Semantic Addressable Encoding

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

Department of Computer Science and Information Engineering
National Taiwan University
cyliou@csie.ntu.edu.tw

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 [4]. Experiments performed on a corpus composed of Shakespeare's writings show its linguistic analysis and categorization abilities.

Index Terms: word perception, authorship, categorization, semantic search, Elman network, linguistic analysis, personalized code, content addressable memory.

1 Introduction

The semantic meaning of a word or a word sequence is often non-quantifiable. A central problem in the analysis of such a sequence is determining how to effectively encoding and extracting its contents. Existing analyses are primarily based on certain statistical linguistic features [2], [7], [20], [21], [22]. The semantic search [23] 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, the process employs the Elman network [3][4], which works well with temporal sequences, as an encoding mechanism. 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 the second section. The semantic search [23] and its notations will be reviewed in this section. Applications to literary works will be presented in the third section.

* Corresponding author.

I. King et al. (Eds.): ICONIP 2006, Part I, LNCS 4232, pp. 183–192, 2006.

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.

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. Its energy function associated with the hidden layer, context layer, and input layer is given in the hairy model [14][15]. 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 [18] 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, 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 $\{w_n, n = 1 \sim N\}$ 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. Initially, $t = 0$, all weights are set to small random numbers. Let $w(t)$ be the current word in a selected sentence at time t, i.e.,

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

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

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

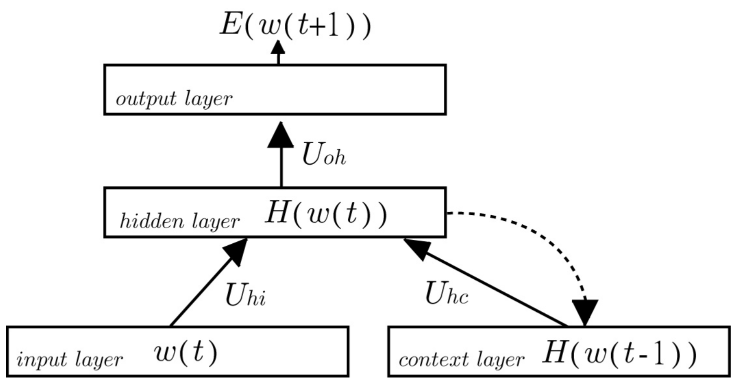


Fig. 1. The Elman Network

where φ is a sigmoid activation function that operates on each element of a vector [18]. 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 and construct a word tree. Before training, he prepares a list of words without inflections or rules. We will follow his preparation on words. All words are coded with certain 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 [4]. 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, $s_n^E = \{H(w(t)) \mid w(t) = w_n\}$. Then he obtains new code, w_n^E, for the n^{th} word by averaging all vectors in set s_n^E:

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

where $|s_n^E|$ is the total number of vectors inside the set s_n^E. Then, he constructs a word tree 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.

Preparation of the Word Corpus

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][17] each word as deep as possible to expose clean relations among words. Note that the degree of stemming is a much discussed lexical issue. For example, it is not clear whether to stem the structure: '-ness,' '-able,' '-tion'.

The Semantic Search

The semantic search [23] 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

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

where w_n, $n = 1 \sim N$, 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:

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

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

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

where

$$F_{R\times R} \equiv [f_1, f_2, \ ..., f_R]_{R\times R}, \quad \|f_r\| = 1, \text{ and } \lambda_r \geq \lambda_{r+1}, \ r = 1 \sim R \ . \tag{8}$$

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

Multidimensional Scaling (MDS) Space

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

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

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, $\{f_r,\ r = 1 \sim R^s\}$, that have large eigenvalues. The MDS space is

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

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

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

or

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

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 [23]. They are the peak-preferred measure,

$$\nu^a_D = [w^a_1, w^a_2, ..., w^a_R]^T;\ \text{where}\quad w^a_r = \max_{w^s_n \in D} |w^s_{rn}|\ ,\ r = 1 \sim R,$$

and the average-preferred measure,

$$\nu^b_D = \sum_{w^s_n \in D} w^s_n = [w^b_1, w^b_2, ..., w^b_R]^T;\ \text{where}\ \ w^b_r = \sum_{w^s_n \in D} w^s_{rn},\ r = 1 \sim R\ . \tag{13}$$

The magnitude is normalized as follows:

$$v_D = \left\| v^b_D \right\|^{-1} v^b_D\ . \tag{14}$$

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 equations (13) and (14).

Relation Comparison

The relation score is defined as follows:

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

Iterative Re-Encoding

Since Elman 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, $w_n^{j=0} = [w_{n1}, w_{n2},...,w_{nR}]^T$. After the j^{th} training epoch, a new raw code is calculated as follows:

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

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, $s_n = \{\varphi(U_{oh}H(w(t-1))) \mid w(t) = w_n$, 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 an updated code from the set s_n, such as the self-organizing map [10], the multi-layer perceptron [12]. After each epoch, all the codes are normalized with the following two equations:

$$W_{R\times N}^{ave} = W_{R\times N}^{raw} - \frac{1}{N} W_{R\times N}^{raw} \begin{bmatrix} 1 & \ldots & 1 \\ \cdot & 1 & \cdot \\ \cdot & & \cdot \\ 1 & \ldots & 1 \end{bmatrix}_{N\times N}, \tag{17}$$

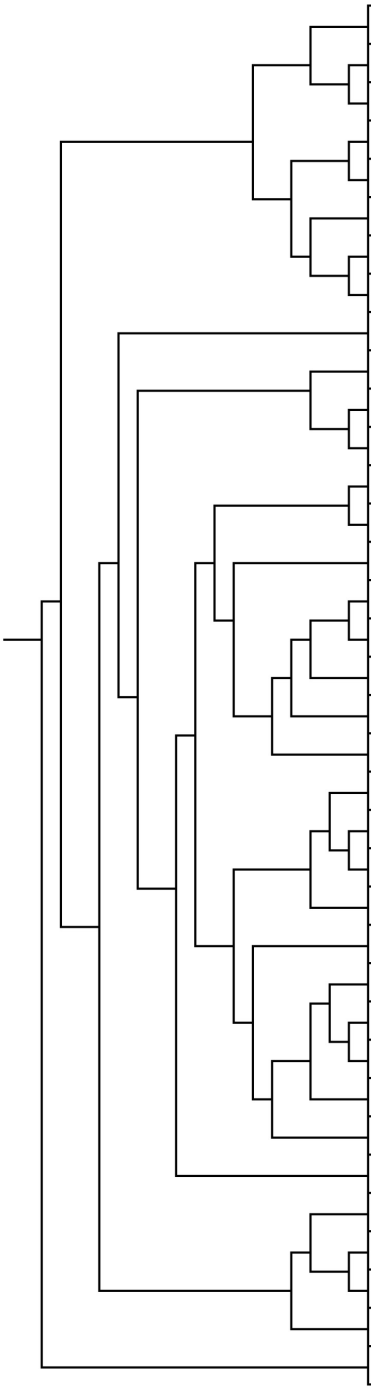
$$w_n^j = w_n^{nom} = \|w_n^{ave}\|^{-1} w_n^{ave}, \quad \text{where } \|w_n\| = (w_n^T w_n)^{0.5}, \quad n = 1 \sim N\ . \tag{18}$$

This normalization can prevent a diminished solution, $\{\|w_n\| \sim 0,\ n = 1 \sim N\}$, 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 [12][13][14][15][18] an Elman network four times. We then compute the new code, w_n^j, for each word using equations (16), (17), and (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 that contain 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, $w_n = w_n^J = [w_{1n}, w_{2n}, , , w_{Rn}]^T$, 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 vector.

3 Example of Literature Categorization

In this experiment, we test the ability to classify 36 plays written by William Shakespeare. 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.



Henry VI, part 2	1590	h
Henry VI, part 1	1589	h
Henry VI, part 3	1591	h
Henry V	1599	h
Richard III	1592	h
Richard II	1595	h
King John	1594	h
Titus Andronicus	1593	t
Jullus Caeser	1599	t
Macbeth	1605	t
Antony and Cleopatra	1606	t
Corlolanus	1607	t
Henry IV, part1	1596	h
Henry IV, part2	1597	h
Romeo and Juliet	1595	t
Hamlet	1600	t
King Lear	1605	t
Cymbeline	1609	r
Pericles	1607	r
The Tempest	1611	r
Timon of Athens	1607	t
Winter's Tale	1610	r
All's Well That Ends Well	1602	c
Measure for Measure	1604	c
Othello	1604	t
Love's Labour's Lost	1595	c
Much Ado About Nothing	1598	c
As You Like it	1599	c
Merchant of Venice	1596	c
Two Gentlemen of Verona	1592	c
Troiles and Cressida	1601	c
Taming of the Shrew	1594	c
Comedy of Errors	1593	c
Twelfth Night	1601	c
Midsummer Night's Dream	1595	c
Merry Wives of Windsor	1597	c

Fig. 2. Categorization of Shakespeare's plays

This tree was constructed by applying the methods in [5][8][19] 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 [1][11][16][22]. In this example, we set $D_i = 1, ..36$, $Q_i = 1, .., 36$, $N = 10,000$ (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 $D_i = 1, ..., 7777$ (the $7,777$ longest conversations in the 23 tragedies and comedies), $N = 10000$, $L_o = L_i = R = 100$, $L_h = L_c = 200$, and $R^S = 64$. Each query indexed one conversation.

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

Summary

In summary, we have explored the concept of semantic addressable encoding and completed a design for it that includes automatic encoding methods. We have applied the methods to study literary works, and we have presented the results. The trained semantic codes can facilitate other research, such as studies on personalized codes, linguistic analysis, authorship identity, categorization, etc. This encoding process can be modified for polysemous words that resolves multiple meaning of a single word.

Acknowledgement

This work was supported by the National Science Council under projects NSC 94-2213-E-002-034 and NSC 92-2213-E-002-002.

References

1. Bloom, H.: Shakespeare: The Invention of Human. Riverhead Books, New York (1998)
2. Burrows, J.: Questions of Athorship: Attribution and Beyond a Lecture Delivered on the Occasion of The Roberto Busa Award ACH-ALLC 2001. New York, Computers and the humanities **37** (2003) 5-32
3. Elman, J.L., Bates, E.A., and Johnson, M.H., Karmiloff-Smith, A., Parisi, D., Plunkett, K.: Rethink Innateness. The MIT Press, Cambridge, Massachusetts (1996)
4. Elman, J.L.: Generalization, Simple Recurrent Networks, and the Emergence of Structure. the 20th Annual Conference of the Cognitive Science Society in Mahway, New Jeresy (1998)
5. Felsenstein, J.: PHYLIP (Phylogeny Inference Package) version 3.5c [Program]. Department of Genetics, University of Washington, Seattle, (1993)
6. Frakes, W.B.: Stemming Algorithms. in Information Retrieval: Data Structures and Algorithms. In: Frakes, W.B., Ricardo, B.-Y. (Eds.), Englewood Cliffs, New Jeresy, Prentice-Hall (1992) 131-160
7. Holmes, D.I.: The Evolution of Stylometry in Humanities Scholarship. Literary and Linguistic Computing **13** (1998) 111
8. Huffman, D.A.: A Method for the Construction of Minimum-redundancy Codes. Proceedings of the I.R.E. **40** (1952) 1098-1102
9. Jordan, M. I.: Serial Order: a Parallel Distributed Processing Approach. Cognitive Science Institute Tech. Rep. 8604, San Diego (1986)
10. Kohonen, T.: Clustering, Taxonomy, and Topological Maps of Patterns. Proceedings of the Sixth Int'l Conference on Pattern Recognition in Silver Spring (1982) 114-125
11. Lee, D.D., and Seung, H.S.: Learning the Parts of Objects by Non-Negative Matrix Factorization. Nature **401** (1999) 788-791
12. Liou, C.-Y., and Yu, W.-J.: Ambiguous Binary Representation in Multilayer Neural Network. Proceedings of Int'l Conference on Neural Networks (ICNN) in Perth, Australia **1** (1995) 379-384
13. Liou, C.-Y., Huang, J.-C., and Kuo, Y.-T.: Geometrical Perspective on Learning Behavior. Journal of Information Science and Engineering **21** (2005) 721-732
14. Liou, C.-Y., and Lin, S.-L.: Finite Memory Loading in Hairy Neurons. Natural Computing **5(1)** (2006) 15-42
15. Liou, C.-Y.: Backbone Structure of Hairy Memory. International Conference on Artificial Neural Networks (ICANN) in Athens, Greece (2006), Lecture Notes in Computer Science, LNCS 4131, Springer
16. McEnery, T., and Oakes, M.: Authorship Identification and Computational Stylometry. in Handbook of Natural Language Processing, Marcel Dekker, Inc. (2000) 545-562
17. Porter, M.F.: An Algorithm for Suffix Stripping. Program **14** (1980) 130-137
18. Rumelhart, D.E., McClelland, J.L., and eds.: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, **1**, Cambridge, MIT Press, Massachusetts (1986)
19. Saitou, N., and Nei, M.: The Neighbor-Joining Method: a New Method for Reconstructing Phylogenetic Trees. Molecular biology and evolution **4** (1987) 406-425
20. Tweedie, F.J., and Baayen, R.H.: How Variable may a Constant be Measures of Lexical Richness in Perspective?. Computers and the Humanities **32** (1998) 323-352

21. William, C.B.: Mendenhall's Studies of Word-Length Distribution in the Works of Shakespeare and Bacon. Biometrika, **62** (1975) 207-212
22. Yang, C.-C., Peng, C.-K., Yien, H.-W., and Goldberger, A.L.: Information Categorization Approach to Literary Authorship Disputes. Physica A **329** (2003) 473-483
23. Yoshida, N., Kiyoki, Y., and Kitagawa, T.: An Associative Search Method Based on Symbolic Filtering and Semantic Ordering for Database Systems. Proceedings of 7th IFIP 2.6 Working Conference on Database Semantics (DS-7) in Leysin, Switzerland (1997) 215-237