COMBINING WORD BASED AND WORD CO-OCCURRENCE BASED SEQUENCE ANALYSIS FOR TEXT CATEGORIZATION

XIAO LUO, A. NUR ZINCIR-HEYWOOD

Faculty of computer science, Dalhousie University, Halifax, NS, Canada
E-MAIL: luo@cs.dal.ca, zincir@cs.dal.ca

Abstract:

This paper represents a text categorization system, which is based on the combination of a hierarchical SOMs encoding architecture and the designed kNN classifier. Through the encoding architecture, a document can be encoded to sequences of neurons so that the sequences of word/word co-occurrence as well as their frequencies are kept. A good performance (Micro average F1-measure 0.98) is achieved on the experimental data set by using this system. This sequence analysis system for text categorization could automatically solve the high dimensionality problem for large data set. And it could be utilized for other data categorization where sequences information is significant and important.

Keywords:

Text Categorization; Sequence Analysis; Self Organization Map

1. Introduction

Text categorization is one of the significant tasks of the content-based document management. Text representation is the necessary and important step for text categorization. Attempts at introducing text representation methods have never ceased since Salton et al [9] introduced the Vector Space Model (VSM). The newly introduced representations include selected n-grams representation [4], Nature Language Processing [1] [7], Bag-Of-Words [5] [8] [11] and Distributional Word Clusters [2]. However, all of these approaches consider term frequencies or words occurrences in documents or categories and therefore ignoring the significance sequences in which they occur. Especially, words sequence or position information is very important to a document when the document is fairly short and the term corpus is small where the words in each of the documents are very similar. Moreover, with the increasing size of the document corpus, each single document usually includes only a small fraction of it; the vector space representation faces the high dimensionality and statistical sparseness problems. We realized that if a document is represented and analyzed in a sequential way, the high dimensionality problem could be solved, and the word sequences or position information could be considered naturally.

In this work, we explored a novel representation of a document, while the sequences of words and word co-occurrences are kept as well as their frequencies. Our approach is quite different from those aforementioned approaches. We investigated a hierarchical SOM architecture to automatically encode the sequence information as a data representation. Thus, by using this approach, a document could be represented by a sequence of neurons on the map; this solves the high dimensionality problem. Moreover, using SOMs – an unsupervised machine learning technique, information sequences are captured automatically in order to detect the characteristics of each category and document without feature selection.

k-Nearest Neighbour (kNN) classifier was employed for the stage of categorization. The results show that this encoding architecture can capture the characteristic words and word co-occurrences sequences for documents and categories. These sequences information can be utilized for document categorization. The results turned out good on the returned Micro F1-measure.

The rest of the paper organized as follows. Section 2 describes the methodology of the encoding architecture and the SOM learning Algorithm. Section 3 presents document representation method. Section 4 describes categorization algorithm. Experiments set up and the categorization results are given in section 5. Finally, conclusions are drawn and future work is discussed in section 6.

2. The methodology and learning algorithm

In this session, a three-level hierarchical SOM architecture is introduced for the process of words encoding and word co-occurrences encoding. Each of the three levels of the SOM hierarchy is employed to discover the patterns of characters, words, and word co-occurrences.

Indeed, pre-processing of data was employed before the encoding process. As usual, all the tags were removed

from the original document. Then a simple part-of-speech (POS) tagging algorithm [3] was used to choose nouns except special nouns from the documents. The chosen nouns are then input to the hierarchical SOM encoding architecture.

2.1 Hierarchical SOMs encoding architecture

Each word in the documents is pre-processed to provide a numerical representation for each character. SOMs may now be used to identify a suitable character encoding, then word encoding, and finally, word co-occurrence encoding. By doing so, the authors of this paper aim to minimize the amount of *a priori* knowledge required to overcome the vocabulary differences. The hierarchical nature of the architecture is shown in Figure 2.

1) Input for the First-Level SOMs: The assumption at this level is that the SOM forms a codebook for the patterns in characters that occur in a specific document category. In order to train an SOM to recognize patterns in characters, the document data must be formatted in such a way as to distinguish characters and highlight the relationships between them. Characters can easily be represented by their ASCII representations. However, for simplicity, we enumerated them by the numbers 1 to 26, i.e. no differentiation between upper and lower case. The relationships between characters are represented by a character's position, or time index, in a word. For example, in the word *"news"*: *"n"* appears at time index 1, *"e"* appears at time index 2, "*w"* appears at time index 3, etc. It should be noted that it is important to repeat these words as many times as they occur in the documents, so that the neurons on the SOM will be more excited by those frequent words and their information will be more accurately encoded. The overall pre-processing process for the first-level SOM is therefore:

- Convert the word's characters to numerical representations between 1 and 26.
- Give the time index to the characters in a word. It is the actual time index plus 2, expect the first character in the word.

The indices of the characters is altered in this way so that when the list is input to an SOM, both data features (enumerated characters and indices) are spread out over a close range. There is no bias on either of the features.

2) Input for the Second-Level SOMs: The assumption at this level is that the SOM forms a codebook for the patterns in words that occur in a specific document category. When a character and its index are run through a trained first-level SOM, the closest neurons (in the Euclidian sense), or *Best Matching Units* (*BMU*s), are used to represent the input space. A two-step process is used to create a vector for each word, *k*, which is input to the first-level SOM of each document:

- Form a vector of size equal to the number of neurons (r) in the first-level SOM, where each entry of the vector corresponds to a neuron on the first-level SOM, and initialize each entry of the vector to 0.
- For each character of word *k*,
  - Observe which neurons $n_1, n_2, ..., n_r$ are affected the most.
  - Increase entries in the vector corresponding to the 3 most affected *BMU*s by $1/j$, $1 \le j \le 3$.

Hence, each vector represents a word through the sum of its characters. The result given by the second-level SOM is clusters of words on the second-level SOM.

3) Input for the Third-Level SOMs: Same as first-level and second-level SOM, the assumption at this level is that the SOM forms a codebook for the patterns in word co-occurrence that occur in a specific document category. In the context of this architecture, word co-occurrence is simply a group of consecutive words in a document. The consecutive words are from a single document with a sliding window of size 3. The input space of the third-level SOM is formed in a similar manner to that in the second-level, except that each word in the word co-occurrences is encoded to the indexes of the 3 most affected *BMU*s resulting from word vectors passed through the second-level SOMs. Thus, the length of the input vector to the third-level SOM is 9. The result given by third-level SOM is clusters of word co-occurrence on the third-level SOM.

2.2 Training SOM – Learning algorithm

The algorithm responsible for the formation of the SOM involves three basic steps after initialization: sampling, similarity matching, and updating. These three steps are repeated until the formation of the feature map has completed [6]. The algorithm is summarized as follows:

- Initialization: Choose random values for the initial weight vectors $w_j(0)$, $j=1,2, \ldots,l$, where l is the number of neurons in the map.
- Sampling: Draw a sample x from the input space with a uniform probability.
- Similarity Matching: Find the best matching neuron $i(x)$ at time step n by using the minimum distance Euclidean criterion:

$$i(x) = \arg\min_j \|x(n) - w_j\|, \qquad j=1,2,\ldots,l \qquad (1)$$

- Updating: Adjust the weight vectors of all neurons by using the update formula:

$$w_j(n+1) = w_j(n) + \eta(n) h_{j,i(x)}(n)(x(n)-w_j(n)) \quad (2)$$

- Continuation: Continue with sampling until no noticeable changes in the feature map are observed.

At the end of the training process, the observed average weight changes of the neurons on the SOM, figure 1, over the training period (in epochs) is analyzed to determine whether the size of the SOM is big enough to represent the pattern distribution of the input data.

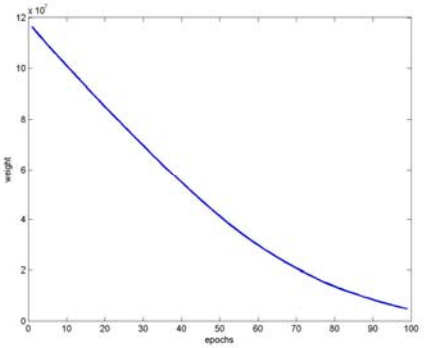

Figure 1. Average weight change of neurons on SOM

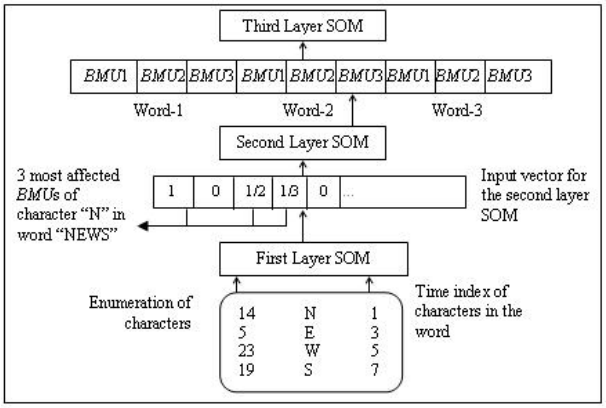

Figure 2. An overview of the hierarchical SOMs encoding architecture

3. Document representations

After training the second and third level of SOMs, we analyzed the hit histograms and neuron sequences for categories and documents on both levels of SOMs. We found that documents from the same category always excite same or close parts of the second and third level SOMs respectively, as well as sharing *BMU* sequences with each other. Figure 3 and 4 gives examples of two document *BMU* sequences on the second and third level SOMs. Both of the documents are from the category "Earn". The dash arrows correspond to the shared *BMU* sequences between the two documents. Moreover, those shared *BMU* sequences between the documents belong to the top frequent *BMU* sequences of their corresponding category. Figure 5 and 6 show the top frequent *BMU* sequences on the second and third level SOMs respectively. It was also shown that different categories have their own characteristic top frequent *BMU* sequences. Thus, it demonstrates that using this architecture to encode the data can capture the characteristic word and word co-occurrence sequences for documents and categories. We hypothesized that the more *BMU* sequences that are shared on both level of SOMs, the more similar the documents. To this end, we proposed the document representation by using sequences of *BMU*s on the SOMs. Thus, each document has two sequence representations; one is based on the second-level SOM, which is the word based sequence representation. The other is based on the third-level SOM, which is the word co-occurrences based sequence representation. The sequence representation is shown in Figure 7. The length of the vector is the number of words/word co-occurrences in the document after pre-processing. The first dimension is the index of the *BMU*s on the second/third level SOM; the second dimension is the Euclidean distances to the corresponding *BMU*s.

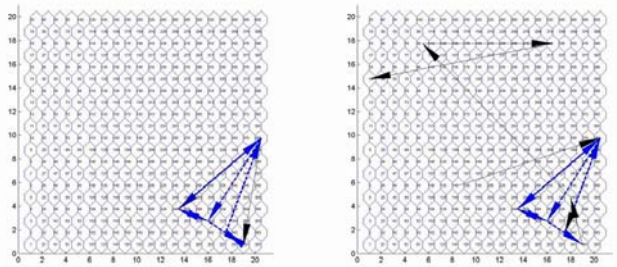

Figure 3. *BMU* sequences of two documents from category "Earn" on the second-level SOM

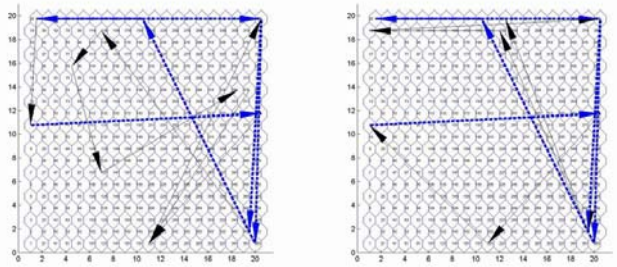

Figure 4. *BMU* sequences of two documents from category "Earn" on the third-level SOM

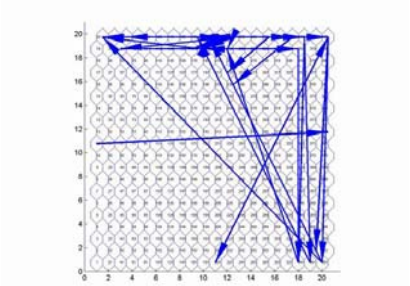

Figure 5. Top frequent *BMU* sequences of category "Earn" with frequency > 150 on the second-level SOM

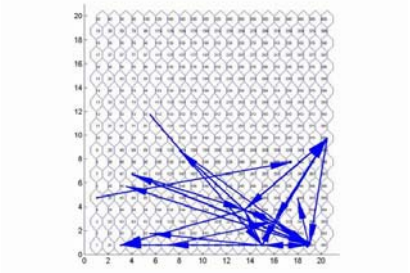

Figure 6. Top frequent *BMU* sequences of category "Earn" with frequency > 150 on the third-level SOM

Index of *BMU*s on SOM:	78	12	3	……
Euclidean distance to the *BMU* (*W*):	7.9	9.8	9.1	……

Figure 7. Example of sequence representation of a document

Thus, we trained a two-level SOMs architecture for the word based sequence representation and a three-level SOMs architecture for the word co-occurrence sequence representation. We considered the balance between the computational cost and the weight change (mentioned in section 3) in choosing the size of a map. The sizes of the maps are shown in Table 1.

Table 1. Size of maps

	Two-level SOMs architecture	**Three-level SOMs architecture**
Level-1	7 by 13	7 by 13
Level-2	20 by 20	8 by 8
Level-3	–	20 by 20

4. Categorization algorithm

kNN stands for k-Nearest Neighbour (kNN) classifier algorithm. It has been studied extensively for text categorization by Yang and Liu [10]. The kNN algorithm is quite simple: To classify a test document, the k-Nearest Neighbour classifier algorithm finds the k nearest neighbours among the training documents, and uses category labels of the k nearest training documents to predict the category of the test document. The similarity in score of each neighbour document to the test document is used as the weight of the categories of the neighbour document [10]. If there are several training documents in the k nearest neighbours that share a category, the category gets a higher weight. Then, it is likely that the test document will to be classified to that category.

In this work, we measure the similarity of documents based on encoded word sequence representation and word co-occurrence sequence representation respectively. However, since neither Euclidean distance nor Cosine distance measurement fits the encoded sequence data representation, we designed similarity measurement formula (3) for word co-occurrences sequence representation and formula (4) for words sequence representation.

$$Sim(D_i, D_j) = \sum_{k=1}^{n} \frac{100}{1 + dist(W_{ik}, W_{jk})} \times n \quad (3)$$

$$Sim(D_i, D_j) = \frac{\sum_{k=1}^{n} \frac{1}{1 + dist(w_{ik} + w_{jk})}}{length(D_j)} \quad (4)$$

D_i : Test document to be categorized.

D_j : Document in the training set.

n : Total number of *BMU*s in common *BMU* sequences of D_i and D_j.

$dist(W_{ik},W_{jk})$: The Euclidean distance between W (defined in Figure 7) of the corresponding *BMU* in the common *BMU* sequences of D_i and D_j. This shows the similarity between words.

$length(D_j)$: Length of the document in the training set. If a test document has the same number and similarity of *BMU*s to more than one training document, then the shorter the training document is, the higher the similarity score it gets.

Both formulas consider the length of the shared *BMU* sequences, in which (3) emphasizes on the similarity degree of the *BMU*s in the sequences, while (4) emphasizes on the length of the training document to be measured.

After weighting the categories by using (3) and (4) for each representation, we normalized the weight of the categories for each representation. Then, we combined the normalized weight of each category by adding them together. To this end, the weighting process is finalized.

5. Experimental setup and categorization results

The practical experiments have been conducted by using the famous data set Reuters-21578 together with the kNN classifier described above. The description of the data set and the corresponding categorization results are given in

the following subsections.

5.1 Experimental data set

In this work, we used the well-known multi-class, multi-labeled document set - Reuters-21578[1], which is a large research-oriented corpus to evaluate the approaches. There are a total of 12612 news stories in this collection. These stories are in English, where 9603 of them are in the training set, and 3299 are in the test set. In our experiments, we made use of all 9603 training files, which belong to 118 categories. Totally, there are 8231 nouns left after pre-processing. Then, the selected nouns of each document are input one by one to the SOMs encoding system. Finally, *BMU* sequence of the third level SOM and the second level SOM is constructed respectively for the two representations of a document.

In general, when a document has more than one label, those category labels are very similar or strongly related to each other. Thus, to automatically explore the relationship of those similar categories becomes a challenge for text categorization. In order to analyze the efficiency of the representation for the multi-class, multi-labeled document categorization, we analyzed complex relationships and overlap between the top 10 categories of this data set. Based on the information we got, we chose 6 of them to test the categorization performance. These categories are "Earn", "Money-fx", "Interest", "Grain", "Wheat" and "Corn". The relationships between these categories in training set are shown in figure 8, and are the same as their relationships in the test set. We could see that "Grain", "Wheat" and "Corn" are strongly related to each other, same as "Money-fx" and "Interest". "Earn", which is the biggest category in the whole corpus, has no overlap with the other five categories. In total, there are 4054 and 1502 multi-labeled and uni-labeled documents in the training set and the test set respectively.

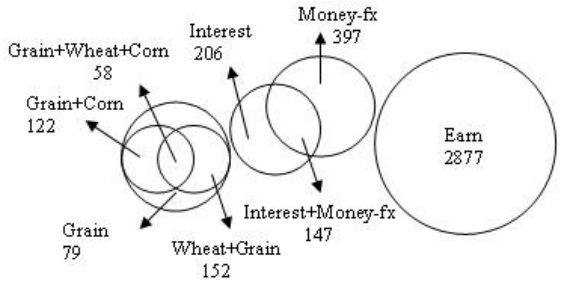

Figure 8. Size and relationship of the six categories in training set

[1] Reuters data set, http://www.daviddlewis.com/resources/testcollections/reuters21578/

5.2 Categorization results and discussion

In our experiments, we set the number of nearest neighbors — k= 3, 5, 10 and 15. It turns out by experiments that k = 5 gives the best performance in terms of the Micro F1-measure score.

Facing a multi-labeled (*N* labels) test document, we classify the test document to the top *N* weighted categories which corresponding to the number of the labels of the test document. The classical effectiveness measurements of multi-labeled text categorization: Recall(R), Precision (P) and F1-measure are used to measure the categorization performance.

$$R = \frac{TP}{TP + FN} \qquad (5) \qquad P = \frac{TP}{TP + FP} \qquad (6)$$

TP : Positive examples, classified to be positive.
FN : Positive examples, classified to be negative.
FP: Negative examples, classified to be positive.

$$F1 - measure = \frac{2RP}{R + P} \qquad (7)$$

Table 2 and 3 summarize the results obtained through the experiments.

Table 2. Categorization results of all six categories

Category	Size in test set	Recall	Precision	F1-Measure
Earn	1087	0.99	0.93	0.96
Money-fx	179	0.78	0.73	0.75
Interest	131	0.57	0.84	0.68
Grain	149	0.87	0.93	0.90
Wheat	71	0.70	0.84	0.76
Corn	56	0.60	0.75	0.67
Micro Average F1-measure:			**0.98**	

Table 3. Categorization results of the multi-labeled documents

Category	F1-Measure
Money-fx+Interest	0.86
Grain+Corn	0.55
Grain+Wheat	0.76
Grain+Corn+Wheat	0.73

From the achieved performance above, this encoding architecture can capture the characteristic sequences for documents and categories. Good performance is achieved by utilizing the sequences information for categorization. However, the results show that it works better for some categories, especially the category "Earn". We hypothesized the reasons behind this is the characteristic word/word co-occurrence sequences information of those categories in the data set is more significant, and we also hypothesized that the sequence analysis system can work

better on the data set where word/word co-occurrence sequence information is more significant and important.

6. Conclusion and future work

Through this work, we explored a sequence analysis system for text representation. This system is based on a hierarchical SOMs encoding architecture. The results (in section 5) show that this architecture works well for capturing the characteristic word and word co-occurrence sequences for documents/categories. Combining the encoding architecture and the designed kNN classifier on top of it, we end up with the Micro average F1-measure for the selected data set at 0.98.

This sequence analysis system naturally solves the high dimensionality problems of the vector space representation for large data sets. Since the order and the frequency of words or word co-occurrences are maintained as they appear before inputting to the encoding architecture, the hierarchy of SOMs can automatically capture the significant characteristics of the data without additional feature selection techniques. Moreover, it can be used for online text categorization without seeing all the words in the whole document corpus. Finally, this mechanism can efficiently work on both textual and non-textual data.

The idea of sequence analysis for text categorization is new, so future work will include performing more experiments and refinements on this system and utilizing it for extensive text categorization as well as classification of data for other applications such as medical or business information systems in which the analysis of sequences information is very important. In addition, other classifiers, which fit to the sequence analysis, will also be analyzed and utilized in the future.

References

[1] R. Basili, A. Moschitti, and M. T. Pazienza, "Language-sensitive text classification", Proceedings of 6th International Conference "Recherche d'Information Assistee par Ordinateur", pp. 331-343, 2000.

[2] R. Bekkerman, R. El-Yaniv, N. Tishby and Y. Winter, "Distributional Word Clusters vs. Words for Text Categorization", Journal of Machine Learning Research, Vol 1, pp. 1-48, 2002.

[3] E. Brill, "A simple rule-based part of speech tagger". Proceedings of 3rd Conference on Applied Natural Language Processing, pp. 152-155, 1992.

[4] M.F. Caropreso, S. Matwin, and F. Sebastiani, "A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization", Text Databases and Document Management: Theory and Practice, pp. 78-102, 2001.

[5] S. T. Dumais, J. Platt, D. Heckerman, and M. Sahami, "Inductive learning algorithms and representations for text categorization", Proceedings of CIKM'98, pp.148-155, 1998.

[6] S. Haykin, "Neural Networks – A comprehensive foundation", Second Edition, Prentice Hall, 1999.

[7] P.S. Jacobs, "Joining statistics with NLP for text categorization", Proceedings of the Third conference on Applied Natural Language Processing, pp. 178-185, 1992.

[8] T. Joachims, "Text Categorization with support vector machines: learning with many relevant features", Proceedings of ECML'98, pp. 137-142, 1998.

[9] G. Salton, A. Wang, and C.S. Yang, "A vector space model for information retrieval", Journal of the American Society for information Science, Vol 18, pp. 613-620, 1975.

[10] Y. Yang, X, Liu, "A re-examination of text categorization methods", Proceedings of SIGIR'99, pp. 42-49, 1999.

[11] S. M. Weiss, C. Apte, F. J. Damerau, D. E. Johnson, F. J. Oles, T. Goetz, and T. Hampp. "Maximizing text-mining performance", IEEE Intelligent Systems, Vol 14, No. 4, pp. 63-69, 1999.