Extracting Features from Protein Sequences Using Chinese Segmentation Techniques for Subcellular Localization

Yang Yang[1,2] and Bao-Liang Lu[1,2]
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University,
800 Dong Chuan Rd., Shanghai 200240, China
[2]Shanghai Institute for Systems Biology, 1954 Hua Shan Rd., Shanghai 200030, China
Email: {alayman, bllu}@sjtu.edu.cn

Abstract **This paper proposes a new method for extracting features from protein sequences to deal with the problem of protein subcellular localization. The idea behind the method arises from Chinese segmentation techniques. We regard the amino acid sequences as text and segment them into words in a non-overlapping way. The words are prede ned in a dictionary, which includes valuable words according to some criteria. Every word in the dictionary will be assigned a weight, and a matching strategy called maximum weight product is adopted for segmentation. By recording word frequencies, a given sequence can be converted into a feature vector. To evaluate the effectiveness of the proposed feature extraction method, two different kinds of classi ers are used to predict protein subcellular locations. The experimental results show that our method is superior to existing approaches in classi cation accuracy and reduces the number of dimensions of feature space at the same time.**

I. Introduction

Subcellular localization of a new protein sequence is very important for understanding its function. Since experimental determination of subcellular location is time consuming and costly, tools for automatic subcellular location prediction have been largely developed in recent years. Sequences of intracellular and extracellular proteins were first analyzed and reported to have different amino acid compositions [1]. In 1994, Nakashima and Nishikawa discriminated those two kinds of proteins successfully by amino acid composition and residue-pair frequencies [2]. After that, a lot of efforts on predicting protein locations in a cell were made. More and more locations can be discriminated. Cedano et al. classified the proteins to five locations, including extracellular, integral membrane, anchored membrane, intracellular and nuclear proteins [3]. Reinhardt and Hubbard used neural network to predict 3 locations of prokaryotic cells and 4 locations of eukaryotic cells [4], and the average accuracies of 80.9% and 66.1% were achieved. Afterwards, proteins in 12 subcellular locations were discriminated [5], [6]. A recent study by Cai and Chou focused on 22 subcellular locations in budding yeast and obtained an overall success rate of 68.36% [7].

Till now, various pattern classification and machine learning methods have been used, such as Mahalanobis distance [3], neural network [4], hidden Markov model (HMM) [9] and support vector machine [10]. But classifiers have limited ability to improve the prediction accuracy. Therefore, how to properly represent proteins becomes more important for subcellular localization. Researchers have developed a lot of feature extraction methods. Besides amino acid composition, Emanuelsson et al. made use of the N-terminal sorting signals [11], which is an efficient method but depends strongly on the leader sequences and often makes mistake when the leader sequences are unreliable. Other methods combine new features, such as hydrophobic [12] information and Zp parameters [13]. Chou introduced the quasi-sequence-order approach [15] and pseudo-amino-acid-composition [14] to incorporate sequence order information. Chou and Cai also developed functional domain composition [17] and gene ontology methods [16] helping to predict protein subcellular locations more precisely. Some of the new features can improve the prediction accuracy significantly. However, we can only obtain corresponding information for a part of proteins. It is difficult to get knowledge like functional composition and gene ontology for new sequences.

Since the numbers of new genomes and protein sequences which are available for prediction have increased dramatically, it is crucial and necessary to perform some deep exploration into the information encoded in the sequences. Similar to analyzing DNA sequences, we can get k-tuple composition of a protein sequence. However, a k-tuple has 20^k different order combinations. One of our goals is to find those useful ones. After finding the valuable k-tuples, we regard the sequences as combinations of words and segment them.

In this paper, we apply Chinese segmentation techniques to analyze amino acid sequences and propose a new method for extracting features from them. This method consists of two steps: dictionary building and segmentation. The k-tuples with highest occurrence frequencies are selected as words in a dictionary, and then a matching strategy called maximum weight product is adopted to segment the sequences. To evaluate the effectiveness of the proposed feature extraction method, two different kinds of classifiers are used to predict protein subcellular locations. The experimental results show that our method is superior to existing approaches in classification

0-7803-9387-2/05/$20.00 ©2005 IEEE

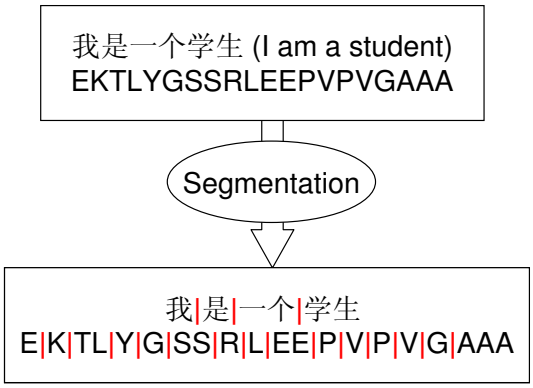

Fig. 1. Similarity between Chinese sentence and protein sequence

accuracy and reduces the number of dimensions of feature space at the same time.

II. Our method

In English text, spaces help to separate the words and understand the sentence well, while Chinese contain no spaces, only punctuations indicating pause or end of a sentence. The automatic analysis of Chinese text has been studied for tens of years. The first step of analysis is Chinese automatic segmentation, which is to separate the character string into meaningful words or phrases [18]. This is an important and basic step for Chinese information processing, such as information retrieval [19] and handwriting recognition [20].

A. Similarity Between Chinese and Protein Sequences

The upper block in Fig. 1 shows an example of Chinese sentence, which means "I am a student". The mark "|" between characters in the lower block denotes a word separator. Compared with a protein sequence, we can find that they are both strings of consecutive characters, written in different languages with respective words. Similar to Chinese text, we model the protein sequences as concatenation of words without any space and punctuation, and try to develop an automatic segmentation technique for them. Thus sequences can be divided into substrings with different lengths. By recording occurrences of all the words in a given sequence, sequence features can be extracted.

In text, words are minimal independent and meaningful language units, and language text usually has a predefined dictionary, i.e., word list. However, protein sequences are written in an unknown language to us at the present state, whose words are not delineated. So we first need to build a dictionary, which is the basis of segmentation. For example, if we have a dictionary $\mathcal{D}$={A, E, P, S, SP, TPT, AAAA} and a sequence $\mathcal{S}$=TPTSPPPAAAAPAE, we can segment $\mathcal{S}$ as TPT|SP|P|P|AAAA|P|A|E.

B. Dictionary Building

Unlike English words, in biosequences, any k-tuple may be meaningful given an alphabet, though it may occur very few times. Since proteins consist of 20 different amino acids, a k-tuple has 20^k different order combinations. This results in a too high dimensionality of the feature space, which can be tens or hundreds of thousands when k 3. It is prohibitively high for many learning algorithms. And it is still difficult to be solved by only counting k-tuples occurring in the data set. Therefore, to develop efficient feature selection methods is a crucial issue.

Our goal is to find out useful words for classification. We suppose the most valuable words are those occurring most frequently in the corpus and put them into the dictionary. To avoid encountering unknown words, all 20 amino acids should be included in the dictionary. And a maximum word length $MaxLen$ should be set, which is the biggest value of k for k-tuples. For every k-tuple with a length less than $MaxLen$, we will count its appearance time in data set in an overlapping way. Then, for every length k, where $1 < k < MaxLen$, add k-tuples with highest frequencies to the dictionary.

C. Segmentation

After building the dictionary, we can match the sequences with words in it. There are thousands of characters usually used in Chinese and every sentence has tens of them at most, while protein sequences usually have hundreds of letters, which composed of only 20 amino acids. Thus, there may be many more ways to segment the sequences into words. To find the best way of segmentation, we first eliminate a large portion of ways by numbers of segments generated, only those which have the least segments remaining. That is to say, long words are preferred to be matched. Then there may still exist multiple ways. We assign a weight for every word in the dictionary and propose a maximum weight product matching method. The details of this matching method are as follows.

For each single letter, let $frequency_{1,i}$ be its occurrence time, and $Freq_1$ be the maximum value of them.

$$Freq_1 = \max_{1 \le i \le 20} frequency_{1,i}. \tag{1}$$

The weights of the 20 single-letter words are defined by:

$$weight_{1,i} = \frac{frequency_{1,i}}{Freq_1}, 1 \le i \le 20. \tag{2}$$

Similarly, for k-tuples, we have

$$Freq_k = \max_{1 \le i \le N} frequency_{k,i}, \tag{3}$$

where N is the number of k-tuples present in the data set. The weights of the k-tuples are defined by:

$$weight_{k,i} = \frac{frequency_{k,i}}{Freq_k} \times C^{k-1}, \tag{4}$$
$$1 \le i \le N, 1 < k \le maxLen, C \quad 1,$$

where C is adjustable, ensuring a part of or most of the long words have bigger weights than short words.

For each way of segmentation with least segments, the weight product is defined by:

$$P_{S,T} = \prod_{\mathcal{W}} weight_w^{n_w}, w \in \mathcal{W} \tag{5}$$

where $\mathcal{W}$ is the set of words matched, and n_w denotes the number of matches of word w in a way of segmentaion T in sequence S. For any given sequence, the optimal segmentation is the one which has the biggest weight product of all the words segmented by itself.

After finishing the segmentation process, appearance time of each word in the dictionary is recorded. Thus, the original sequence can be converted into a vector with the dimensionality of dictionary size.

Let $segNo$ and $wordLen$ be two arrays. $segNo(pos)$ records the number of segments which have been identified till current position pos. $wordLen(pos)$ records the length of the word beginning from current character. $subStr_{len}^{pos}$ stands for the substring of length len starting from position pos. The process of segmentation is described as follows.

Procedure Segment (Data Set $\mathcal{T}$, Dictionary $\mathcal{D}$)

1) For each sequence S in $\mathcal{T}$:
2) $Vector(S) = \vec{0}$,
3) $segNo = \vec{0}$,
4) $wordLen = \vec{0}$.
5) **Search**$(\mathcal{D}, S, 1)$.
$segNo$ and $wordLen$ are evaluated in this function.
6) Set $counter$ = 0.
7) While($counter \leq lengthofS$):
8) w = $subStr_{counter}^{wordLen(counter)}$;
9) $Vector(S)(Id_w)$++;
Id_w is the index of w, $w \in \mathcal{D}$
10) $counter+ = wordLen(counter)$.
11) End While.
12) End For.
13) Return $Vector$.

The function **Search** called at line 5 implement a heuristic process for searching the optimal way of segmentation for a given sequence. It is presented in Fig. 2.

III. Experimental Results

We conducted experiments on the dataset published by Park and Kanehisa [6], which can be obtained at the website http://web.kuicr.kyoto.ac.jp/ park/Seqdata. There are 7579 protein sequences in total, located in 12 subcellular locations: chloroplast, cytoplasmic, cytoskeleton, endoplasmic reticulum, extracellular, Golgi apparatus, lysosomal, mitochondrial, nuclear, peroxisomal, plasma membrane and vacuolar. The distribution of the data set is listed in Table I. All experiments were performed on a 3GHz Pentium 4 PC with 2GB RAM.

We used K-nearest neighbor (KNN) and support vector machine (SVM) to predict. Firstly, protein sequences should be converted into vectors based on our method. Here comes a problem: how many features should be used? This problem refers to two parameters in advance. One is the maximum length of words mentioned in II-B, and the other is the number of words for a definite length. We aimed to reduce the dimensionality of the feature space with no loss in classification accuracy. In order to find the best number of features, plenty of experiments were conducted. There are mainly two lines of experiments, namely searching proper number of words for each length and maximum word length, respectively.

TABLE I
Distribution of The Data Set

Location	Number of sequences
Chloroplast	671
Cytoplasmic	1241
Cytoskeleton	40
Endoplasmic reticulum	114
Extracellular	861
Golgi apparatus	47
Lysosomal	93
Mitochondrial	727
Nuclear	1932
Peroxisomal	125
Plasma membrane	1674
Vacuolar	54
Total	7579

A. Number of Words per Length

Here we specify the maximum word length to 3, the discussion on which will be given in Subsection III-B. So the dictionary includes top n frequently occurring 2-tuples and 3-tuples, respectively, and 20 letters from amino acid alphabet Φ = {A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y}. The number of dimensions of feature vectors, Dim, can be expressed as:

$$Dim = \sum_{l=1}^{3} N_l = 2n + 20, \tag{6}$$

where N_l is the number of words of length l. Let n be 100, 50, 20, 10, 5 and 2, respectively. From (6) we get 220, 120, 60, 40, 30, and 24 features accordingly. The prediction accuracies using the 6 different kinds of feature vectors as inputs for KNN are depicted in Fig. 3. The K value of the KNN classifier varies from 2 to 16. Accuracies are the averages of 5-fold cross-validation results. And the dictionary for segmentation is built only on training data.

As for SVM, one-versus-rest strategy was adopted. We used LibSVM version 2.6 [22] and trained the classifier with a RBF kernel. The kernel parameter γ and penalty parameter C were searched from 225 different combinations: $\gamma = [2^4, 2^3, ..., 2^{-10}]$ and $C = [2^{12}, 2^{11}, 2^{10}, ..., 2^{-2}]$. Fig. 4 shows classification accuracy of every dimensionality using SVM with $\gamma = 0.000977, C = 32$ and KNN with $K = 2$ or 3.

B. Maximum Word Length

In the last subsection, the maximum word length is fixed to 3. Here bigger lengths are assumed. Let $MaxLen$ be 3,4 and 5, respectively, and set the number of words per length to 5. According to the following equation

$$Dim = \sum_{l=1}^{MaxLen} N_l = 20 + (MaxLen \quad 1) \times 5, \tag{7}$$

Procedure Search (Dictionary $\mathcal{D}$, sequence S, Position pos)

1) If $pos = N$: # N is the total length of the sequence
2) $segNo(pos) = 1, wordLen(pos) = 1$.
3) **Return** 1.
4) If $segNo(pos) \neq 0$:
5) **Return** $segNo(pos)$. # the character has been analyzed.
6) Set $tempSegNo = \vec{0}, subStrLen = \vec{0}, counter = 0$.
7) For each $len, 1 \leq len \leq maxLen$:
8) If $subStr_{pos}^{len} \in \mathcal{D}$:
9) $subStrLen(count) = len$.
10) $tempSegNo(count)$=1+**Search**($\mathcal{D}$,S, $pos + len$).
11) $count++$.
12) End If.
13) End For.
14) For each j that $tempSegNo(j) = \min tempSegNo(i), 1 \leq i \leq count$.
15) Set $weightProduct(j) = weight(subStr_{pos}^{subStrLen(j)}), tmpPos = pos + subStrLen(j)$.
16) While($tmpPos \leq N$):
17) $weightProduct(j) = weightProduct(j) \times weight(subStr_{tmpPos}^{wordLen(tmpPos)})$.
18) End While.
19) End for.
20) Find k that $weightProduct(k) = \max_j weightProduct(j)$.
21) Set $wordLen(pos) = subStrLen(k)$, $segNo(pos) = tempSegNo(k)$.
22) **Return** segNo(pos).

Fig. 2. The segmentation algorithm

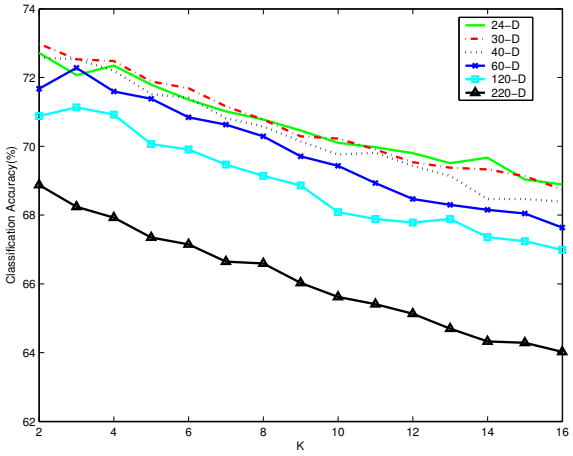

Fig. 3. Classification accuracy for different dimensions using KNN algorithm.

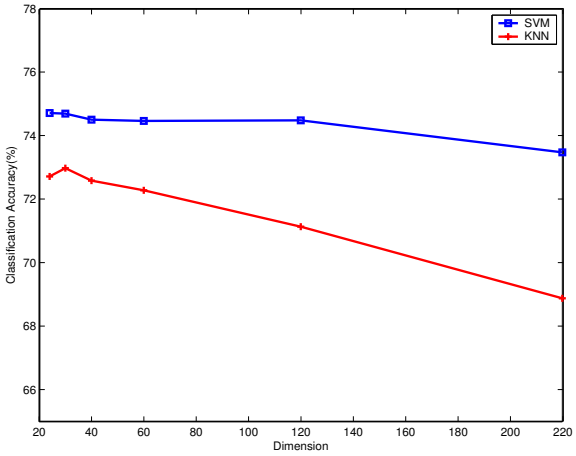

Fig. 4. Classification accuracy for different dimensions using SVM and KNN with $MaxLen$ set to 3.

we can get 30, 35 and 40 features accordingly. The prediction accuracies using the three different kinds of feature vectors as inputs for KNN are shown in Fig. 5.

For evaluating the effectiveness of classification, we use the standard recall, precision and F_1 measure [21] for single classes, and Macroaverage and Microaverage [23] for all classes. Recall is the ratio of samples belonging to class C_i classified correctly compared to the number of samples of C_i. Precision is the ratio of samples belonging to C_i classified correctly to the number of samples classified into C_i. Let R stands for recall and P for precision:

$$R = \frac{tp}{tp + fn}, P = \frac{tp}{tp + fp}, \tag{8}$$

where tp, fp and fn denote the number of true positives, false positives and false negatives, respectively. The F_1 measure corresponds to the harmonic mean of recall and precision in the following form:

$$F_1 = \frac{2RP}{R + P}, \tag{9}$$

Macroaverage is the average of F_1 values of all classes,

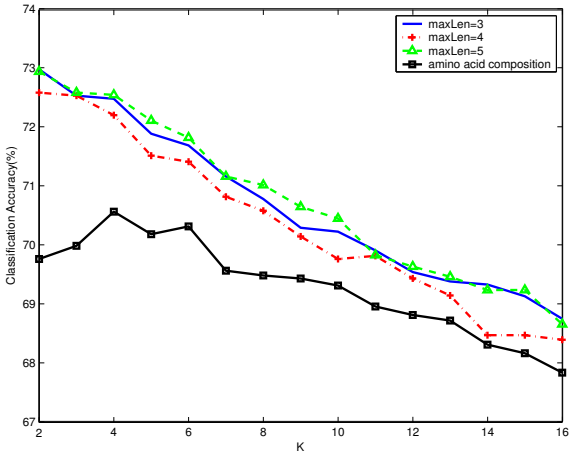

Fig. 5. Classification accuracy for different maximum word-length using KNN algorithm.

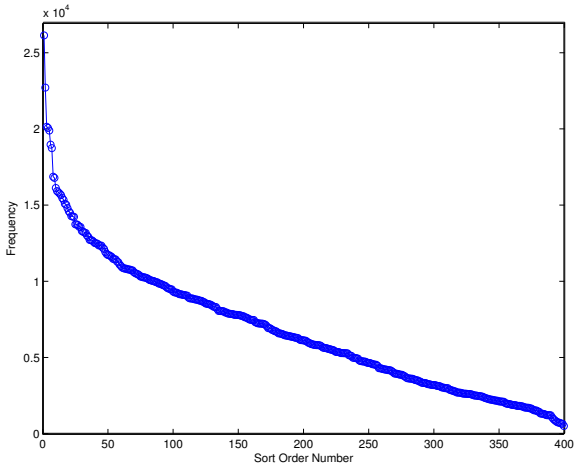

Fig. 6. Frequencies of 400 amino acid pairs.

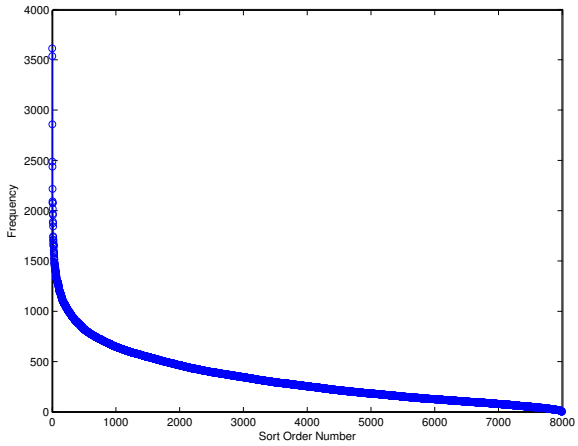

Fig. 7. Frequencies of 8000 3-tuples.

TABLE II
COMPARISON OF MACROAVERAGE AND MICROAVERAGE

	Amino acid composition	Segmentation method		
		3	4	5
Macroaverage	53.4	61.2	60.9	61.4
Microaverage	70.3	74.7	74.5	74.5

and Microaverage can also be calculated by (9) regarding all classes as one.

The detailed values of recall, precision, F_1, Macroaverage and Microaverage of 12 classes are given in Tables II and III, which list three different cases of our method, with maximum word length equal to 3, 4 and 5, respectively, and amino acid composition for comparison.

As a complementary result, the frequencies of all the 2-tuples and 3-tuples are depicted in Figs. 6 and 7. We can find that very few 2-tuples or 3-tuples have extremely high frequencies, and the same situation occurs at other k-tuples with $k > 3$ in our experiment.

From the experimental results, several observations can be made. Generally, SVM performed better than KNN, about 3% higher on accuracy when both use the best parameters. Given a maximum word length, the accuracy drops a little as the number of features increased, and it falls relatively quickly using KNN. So adding more words may induce some noise hurting the accuracy. The maximum word length varying from 3 to 5 differs slightly on the results, which indicates that 3-tuples may be enough for representing the sequences features. Thus we can use very few features for classification.

IV. DISCUSSION

A. Criterion for Building Dictionary

In this paper, we predefine a dictionary containing a number of words for segmenting protein sequences. The criterion for a k-tuple to be included in the dictionary is its frequency. A k-tuple occuring more often in the data set is considered to be a meaningful word which should be separated out. It may take on some important information, though we know little about that kind of knowledge at the present state.

Although the experimental results demonstrate the effectiveness of the features extracted by word frequencies, it may not be considered as the best criterion. As is well known, in document indexing and retrieval, a lot of criteria for selecting useful words, such as document frequency, information gain, mutual information, are adopted rather than word frequency [24] because some of the frequent words are non-informative and can even hurt the classification accuracy.

Therefore, we also conducted an experiment on another measure for word evaluation, $tfidf$ value [25], which is commonly used in text categorization. tf denotes term frequency, and idf denotes inverse document frequency. For the protein subcellular localization problem, we redefine it as follows. Let N be the total number of amino acid sequences, and n_t be the number of sequences in which the k-tuple t appears. $freq_{t,S}$ which stands for tf part is the number of appearance time of t in sequence S and $log\frac{N}{n_t}$ stands for idf part. Then the $tfidf$

TABLE III

SVM CLASSIFICATION ACCURACIES(%) FOR DIFFERENT MAXIMUM WORD LENGTHS, COMPARED WITH AMINO ACID COMPOSITION

Location	Amino acid composition			Maximum word length								
				3			4			5		
	R	P	F_1	R	P	F_1	R	P	F_1	R	P	F_1
Chloroplast	60.1	60.0	60.0	69.0	66.7	**67.8**	68.9	66.7	67.7	68.6	66.2	67.4
Cytoplasmic	62.8	60.7	61.5	66.5	65.5	**66.0**	66.7	65.2	65.9	66.1	65.0	65.5
Cytoskeleton	56.2	63.8	56.0	48.0	89.3	**59.0**	48.0	89.3	**59.0**	45.5	89.3	56.6
Endoplasmic reticulum	49.9	52.9	50.4	55.2	70.4	61.3	55.2	71.1	61.9	56.1	70.8	**62.2**
Extracellular	70.5	72.3	71.4	78.2	77.6	77.9	77.2	77.2	77.2	78.4	78.2	**78.2**
Golgi apparatus	28.0	39.7	30.9	25.6	41.2	31.4	25.6	40.6	31.2	27.8	43.9	**33.8**
Lysosomal	62.3	49.9	55.1	65.4	76.3	**69.7**	65.4	74.1	68.9	66.4	71.6	68.3
Mitochondrial	36.4	51.6	42.4	49.3	60.7	**53.9**	48.9	59.7	53.2	49.2	58.4	53.2
Nuclear	81.5	75.2	78.2	85.2	74.1	79.21	85.0	74.4	79.3	85.0	74.4	**79.4**
Peroxisomal	23.9	33.4	27.4	36.7	56.5	**44.2**	35.2	57.5	43.3	35.9	57.1	43.8
Plasma membrane	90.4	86.8	88.6	88.4	92.5	**90.4**	88.4	92.2	90.2	88.1	92.3	90.1
Vacuolar	18.2	23.3	19.5	28.4	50.7	33.9	28.4	48.7	33.4	32.0	54.7	**38.6**

weight $w_{t,S}$ is calculated as follows:

$$w_{t,S} = freq_{t,S} \times log\frac{N}{n_t}. \tag{10}$$

And the weight of t, w_t, is defined as the maximum value of $w_{t,S}$:

$$w_t = \max_{S \in \mathcal{T}} w_{t,S}, \tag{11}$$

where $\mathcal{T}$ denotes the whole data set.

The dictionary is constructed by collecting words with high $tfidf$ value. Figs. 8 and 9 depict all the $tfidf$ values for 2-tuples and 3-tuples in descending order. It can be observed that only a few k-tuples have extremely big $tfidf$ values.

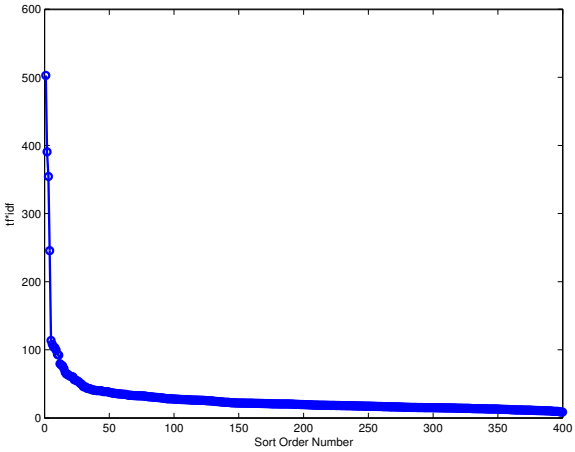

Fig. 8. $tfidf$ values of 400 amino acid pairs

For comparison, we list the words of highest frequencies and $tfidf$ values in Table IV. The top frequent k-tuples are those k-copies of a single letter. This is due to that a big number of substrings like "QQQ...QQ" consisting of only one single letter exist in the data set and we count word frequency in an overlapping way. Most of the words selected by $tfidf$ are different from those by frequency, which indicates that many high frequency words are not concentrated in a few sequences, but instead spreading over all sequences in the data set.

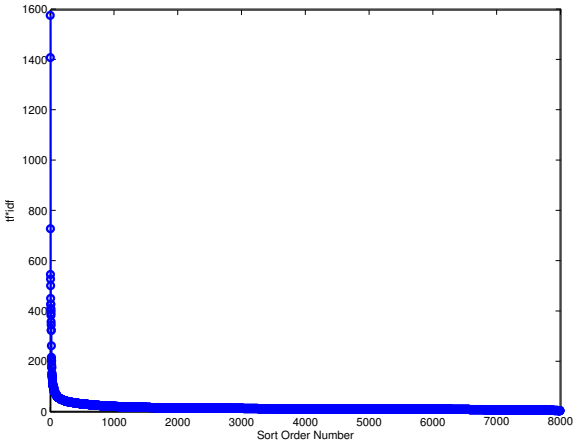

Fig. 9. $tfidf$ values of 8000 3-tuples

For evaluating the $tfidf$ measure, we built a dictionary containing 30 words, with 5 words per length and $maxLen$ equal to 3. The experiment is conducted on the same data set used in Section III. In Table V, F_1 values by $tfidf$ are compared with corresponding ones by frequency.

And the Macroaverage and Microaverage obtained by using $tfidf$ is 60.96% and 74.67%, respectively. These statistics show no improvement for prediction. And the computation of $tfidf$ has a higher time and space complexity than frequency.

Therefore, further studies on evaluating informative words are still needed. Perhaps further improvement may be obtained by modifying the principles of building dictionary and adjusting the number of features.

B. Comparison with other methods

Since our method does not need external reference data set or information but only protein sequences, we made comparisons with other methods based merely on protein sequences. It can be found that our segmentation method outperforms amino acid composition (AAC) on almost all the measures. As for recall value, which indicate location accuracy, segmentation is better than AAC at nine locations, three among which increase

TABLE IV

WORDS SELECTED BY FREQUENCY AND $tfidf$ VALUE

word length	$tfidf$	frequency
2	TT, TP, PT, QQ, QT, NN, TQ, MP, PK, PI	LL, SS , SL, LS, AA, LA, AL, EE, LG, VL
3	TTT, TPT, PTP, TTP, PTT, TQT, GTQ, PIT, QTP, TPI	AAA, LLL, SSS, EEE, GGG, QQQ, LLS, SLL, LLA, SSL
4	TTTT, PTPT, TPTP, PTTT, TTTP, GTQT, TTPI, TPIT, TGTG, TQTP	QQQQ, AAAA, SSSS, GGGG, EEEE, PPPP, LLLL, NNNN, TTTT,RRRR
5	TPTPT, ITTTT, TTTPI, TQTPT, PTTTP, TTPTT, PTPTG, PTGTG, QTPTT, TGTGT	QQQQQ, AAAAA, GGGGG, SSSSS, EEEEE, NNNNN, PPPPP, SPTSP, TPTPT, YSPTS

TABLE V

F1 VALUES USING FREQUENCY AND $tfidf$

Location	$tfidf$	frequency
Chloroplast	66.3	67.8
Cytoplasmic	65.9	66.0
Cytoskeleton	54.7	59.0
Endoplasmic reticulum	**61.9**	61.3
Extracellular	**78.2**	77.9
Golgi apparatus	30.0	31.4
Lysosomal	65.7	69.7
Mitochondrial	**54.1**	53.9
Nuclear	**79.2**	79.2
Peroxisomal	**48.2**	44.2
Plasma membrane	90.3	90.4
Vacuolar	**37.0**	33.9

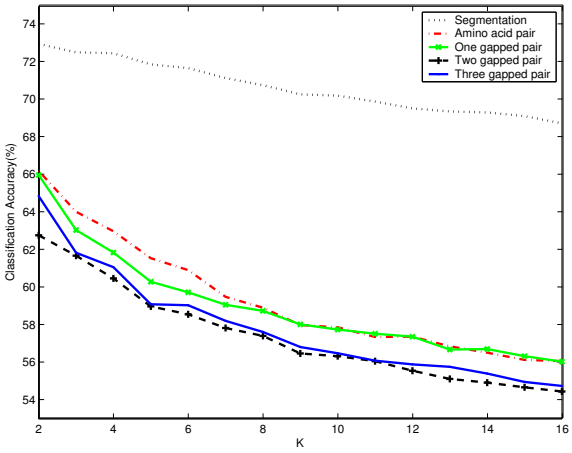

Fig. 10. Comparison with 4 amino acid pair methods using KNN

TABLE VI

MACROAVERAGE AND MICROAVERAGE OF 4 AMINO ACID PAIR METHODS

Method	Dimension	Macroaverage	Microaverage
Amino acid pair	400	58.4	73.6
One gapped pair	400	60.0	73.9
Two gapped pair	400	55.8	72.5
Three gapped pair	400	57.3	72.5
Segmentation	30	61.2	74.7

more than 10%. F_1 is a comprehensive evaluation of recall and precision. All the F_1 values of segmentation are superior to ACC, with four ones over 10% higher.

In addition, a comparison with other four types of compositions namely amino acid pair, one gapped amino acid pair, two gapped amino acid pair and three gapped amino acid pair mentioned in [6] is given. We implemented them under our software and hardware environment.

The most obvious advantage of our method is its condensed feature space, which has much fewer features than the four amino acid pair methods. All of the amino acid pair methods have a feature space of 400 dimensions, while segmentation method achieves satisfactory result only using less than 30 features. Moreover, it performs well under both SVM and KNN, while as shown in Fig. 10, amino acid pair composition cannot achieve good results using KNN, about 7% lower than segmentation at the best situation. The detailed results by SVM are given in Table VI and VII, with $\gamma = 16$ and $C = 256$ which are also choosing from 225 combinations. As for F_1 value, our methods have six ones superior to all the four method with two over 10% higher, and both Microaverage and Macroaverage are 1% to 5% higher than the four methods.

A conclusion can be drawn that our method can gain a high accuracy and reduce the dimensionality of feature space greatly compared with other methods based on counting k-tuples overlappingly.

V. CONCLUSIONS AND FUTURE WORK

This study focuses on seeking efficient feature extraction of protein sequences. We aim to develop a general method for mining the information encoded in enormous protein sequences. Noticing the similarity between Chinese text and protein sequences, a segmentation method is proposed to separate sequences of consecutive characters to words with various lengths. By counting frequencies of the words segmented, a protein sequence is converted into a feature vector.

To demonstrate our method, we use the feature vectors to discriminate proteins in different subcellular locations. The experiments were conducted on a data set of 7579 proteins on 12 locations. Experimental results show its high efficiency especially on feature reduction. It uses very few features, no more than 30, and achieves an equal and even better accuracy than the existing methods based on protein sequences which usually have hundreds of features.

It should be noted that Park and Kanehisa reported that they achieved an accuracy of 78%, and a much higher one of 92.4% is obtained by Chou and Cai on the same data set using GO-FunD-PseAA method [26]. Both of their approaches use much more features than the segmentation method. The former involves a voting scheme of four kinds of

TABLE VII

RESULTS OBTAINED BY 4 AMINO ACID PAIR METHODS

Location	Amino acid pair			One gapped amino acid pair			Two gapped amino acid pair			Three gapped amino acid pair		
	R	P	F_1	R	P	F_1	R	P	F_1	R	P	F_1
Chloroplast	62.8	68.1	**65.2**	62.0	68.3	65.0	63.1	63.8	63.4	63.2	65.5	64.3
Cytoplasmic	65.0	65.1	**65.1**	64.7	63.8	64.2	63.0	63.6	63.3	65.0	63.7	64.3
Cytoskeleton	71.3	68.1	68.1	78.9	67.0	**71.8**	68.5	56.2	61.0	73.8	64.0	67.7
Endoplasmic reticulum	56.2	57.6	56.2	57.8	61.7	**59.0**	49.2	50.2	49.1	53.5	49.7	51.5
Extracellular	69.7	71.0	70.3	71.3	73.6	**72.3**	71.8	72.3	72.0	69.6	72.3	70.9
Golgi apparatus	29.8	35.0	31.5	34.0	38.6	**35.9**	23.6	45.2	29.7	27.3	50.1	30.4
Lysosomal	58.9	55.2	56.6	54.9	56.2	55.0	57.0	59.9	**57.6**	57.0	51.7	53.9
Mitochondrial	50.4	56.0	53.0	52.4	56.5	**54.3**	43.2	53.4	47.7	45.0	55.6	49.6
Nuclear	84.5	79.6	**82.0**	84.1	79.2	81.5	83.9	78.1	80.9	83.8	78.3	80.9
Peroxisomal	27.2	32.9	29.6	29.4	37.0	31.9	27.8	33.1	30.1	29.6	38.4	**32.7**
Plasma membrane	91.8	89.3	90.6	92.1	90.1	**91.1**	92.1	69.0	90.5	91.9	88.7	90.3
Vacuolar	30.0	37.1	32.9	31.8	47.1	**37.4**	21.8	28.5	24.4	26.0	37.5	30.5

amino acid pair compositions. And the latter references other database, which hybridizes gene ontology, functional domain composition and pseudo-amino acid composition approach. As a general method based merely on protein sequences, the segmentation method can reference the voting schemes and other information available to improve prediction accuracy in the future study.

In summary, our method is successful applied to subcellular localization. Without losing generality, it may be useful in solving other protein classification problems. In particular, since a large number of proteins with multiple locations exist, the multi-locational problem is one of our future works.

ACKNOWLEDGEMENTS

This research was partially supported by the National Natural Science Foundation of China via the grants NSFC 60375022 and NSFC 60473040. The authors thank Prof. Liping Zhao and Prof. Zhizhou Zhang for their helpful advice, Dr. Hai Zhao for his suggestion on segmentation techniques and Mr. Zhenhua Hu for result comparison work.

REFERENCES

[1] K. Nishikawa, Y. Kubota and T. Ooi, "Classification of proteins into groups based on amino acid composition and other characters," J. Biochem., vol. 94, 1983, pp. 997-1007.

[2] H. Nakashima and K. Nishikawa, "Discrimination of intracellular and extracellular proteins using amino acid composition and residue-pair frequencies," J. Mol. Biol. 238, 1994, pp. 54-61.

[3] J. Cedano, P. Aloy, J.A. Perez-Pons and E. Querol, "Relation between amino acid composition and cellular location of proteins," J. Mol. Biol., vol. 266, February 1997, pp. 594-600.

[4] A. Reinhardt and T. Hubbard, "Using neural networks for prediction of the subcellular location of proteins," Nucleic Acids Research, vol. 26, 1998, pp. 2230-2236.

[5] K. C. Chou and D. W. Elrod, "Protein subcellular location prediction," Protein Eng., vol. 12, 1999, pp. 107-118.

[6] K. J. Park and M. Kanehisa, "Prediction of protein subcellular locations by support vector machines using compositions of amino acids and amino acid pairs," Bioinformatics, vol. 19, 2003, pp. 1656-1663.

[7] Y. D. Cai and K. C. Chou, "Predicting 22 protein localizations in budding yeast," Biochemical and Biophysical Research Communications, vol. 323, 2004, pp. 425-428.

[8] Z. P. Feng, "An overview on predicting the subcellular location of a protein," In Silico Biology, vol. 2, 2002, pp. 291-303.

[9] Y. Fujiwara, M. Asogawa and K. Nakai, "Prediction of mitochondrial targeting signals using hidden Markov models," Genome Informatics, 1997, pp. 53-60.

[10] S. Hua and Z. Sun, "Support vector machine approach for protein subcellular localization prediction," Bioinformatics, vol. 17, 2001, pp. 721-728.

[11] O. Emanuelsson, H. Nielsen, S. Brunak and G. von Heijne, "Predicting subcellular localization of proteins based on their N-terminal amino acid sequence," J. Mol. Biol., vol. 300, 2000, pp. 1005-1016.

[12] Z. P. Feng and C. T. Zhang, "Prediction of the subcellular localization of prokaryotic proteins based on the hydrophobicity index of amino acids," Int. J. Biol. Macromol., vol. 28, 2001, pp. 255-261.

[13] Z. P. Feng and C. T. Zhang, "A graphic representation of protein sequence and predicting the subcellular localizations of prokaryotic proteins," Int. J. Biochem. Cell Biol, vol.34, 2002, pp. 298-307.

[14] K. C. Chou, "Prediction of protein cellular attributes using pseudo-amino-acid-composition," Proteins, vol. 43, 2001, pp. 246-255.

[15] K. C. Chou, "Prediction of protein subcellular locations by incorporating quasi-sequence-order effect," Biochemical and Biophysical Research Communications vol. 278, 2000, pp. 477-483.

[16] K. C. Chou and Y. D. Cai, "A new hybrid approach to predict subcellular localization of proteins by incorporating gene ontology," Biochemical and Biophysical Research Communication, vol. 311, 2003, pp. 743-747.

[17] K. C. Chou and Y. D. Cai, "Using functional domain composition and support vector machines for prediction of protein subcellular location," J. Biol. Chem., vol. 277, 2002, pp. 45765-45769.

[18] W. Y. Jin, "Chinese Segmentation Disambiguation," Proceedings of the International Computational Linguistics-94 (COLING'94), 1994, pp. 1245-1249.

[19] J. Y. Nie and F. Ren, "Chinese information retrieval:using characters or words? " Information Processing and Management, vol.35, July 1999, pp. 443-462.

[20] C. H. Chang, "Word Class Discovery for Postprocessing Chinese Handwriting Recognition," Proceedings of COLING, 1994, pp. 1221-1225.

[21] D. D. Lewis, "Evaluating and optimizing autonomous text classification systems," Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval(SIGIR 95), 1995, pp. 246-254.

[22] C. C. Chang, and C. J. Lin, "LIBSVM: a library for support vector machines," Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm, 2001.

[23] D. D. Lewis, "Evaluating text categorization," Proceedings of the Speech and Natural Language Workshop, 1991, pp. 312-318.

[24] Y. Yang, "Noise reduction in a statistical approach to text categorization," Proceedings of the 18th Ann Int. ACM SIGIR Conference on Research and Development in Information Retrieval(SIGIR'95), 1995, pp. 256-263.

[25] F. Sebastiani, "Machine learning in automated text categorization," ACM Computing Surveys, vol.34, March 2002, pp. 1-47

[26] K. C. Chou and Y. D. Cai, "Prediction of protein subcellular locations by GO-FunD-PseAA predictor," Biochemical and Biophysical Research Communications, vol. 320, 2004, pp. 1236-1239.