**TELKOMNIKA Indonesian Journal of Electrical Engineering**
Vol.12, No.6, June 2014, pp. 4322 ~ 4329
DOI: 10.11591/telkomnika.v12i6.4689


# A Java Program of Feature Extraction Algorithms for Protein Sequences

**Shanping Qiao*[1,2], Baoqiang Yan[3]**
[1]School of Management Science and Engineering, Shandong Normal University
[2]Shandong Provincial Key Laboratory of Network Based Intelligent Computing,
School of Information Science and Engineering, University of Jinan,
No. 336, West Road of Nan Xinzhuang, Jinan 250022, China, Ph.: +86-53189736503
[3]School of Mathematical Science, Shandong Normal University
No. 88, Cultural East Road, Jinan 250014, China, Ph.: +86-53186182501
*Corresponding author, e-mail: qspzl@hotmail.com[1,2], yanbaoqiang666@gmail.com[3]

***Abstract***

*Prediction of protein subcellular localizations attracted the eyes of many researchers and hence a serial of computational approaches which aimed at designing an effective learning machine to deal with the newly-found protein sequences on the base of the feature vector were developed in the last two decades. The feature extraction algorithm for protein sequences played a vital role actually. The information in the feature vector influenced the performance of the learning algorithm significantly. In order to facilitate users to build predicting system, three feature extraction algorithms about amino acid composition were introduced, improved and implemented in a Java program. By comparing the results with those from some web servers, it was proved that this program ran normally and had good performance both in time costing and user interface. Moreover, the results could be easily saved to the specified file for later use. It was anticipated that this program would give some help to researchers.*

***Keywords****: protein subcellular location prediction, amino acid composition, feature extraction algorithm, Java*



## 1. Introduction

Prediction of protein subcellular localizations is an important and hot topic in bioinformatics. Knowledge of protein subcellular locations often offers important clues toward determining the function of an uncharacterized protein. Therefore, a key step on this way is to determine the subcellular localizations of each protein. The traditional approach to this problem is doing the physicochemical experiments, such as cell fractionation, electron microscopy and fluorescence microscopy. However, it is time-consuming and costly to settle this problem based on experiments purely. Moreover, the number of annotated proteins has increased explosively in the post-genomic age, as illustrated in Table 1. Therefore, experimental annotation of protein subcellular localizations can not keep up with the huge number of sequences that continue to emerge from the genome sequencing projects. To bridge this gap, it is highly desirable to develop the computational methods for predicting protein subcellular localizations. In fact, many efforts have been made in the last two decades [1-5].

Table 1. Number of Protein Sequences in the UniProtKB/Swiss-Prot

| Release Date | Database Version | Total | Experimental Annotations | Non-experimental Annotations |
|---|---|---|---|---|
| 2003-12-15 | 1 | 135,938 | 38,903 | 45,391 |
| 2004-07-05 | 2 | 148,277 | 41,031 | 50,806 |
| 2005-05-10 | 5 | 178,998 | 45,606 | 65,084 |
| 2006-10-31 | 9 | 239,174 | 53,510 | 94,897 |
| 2007-07-24 | 12 | 274,311 | 57,490 | 113,135 |
| 2008-07-22 | 14 | 390,787 | 64,733 | 167,972 |
| 2009-09-01 | 15.7 | 495,368 | 68,029 | 220,091 |
| 2010-07-13 | 2010_08 | 516,934 | 70,180 | 232,546 |
| 2011-07-27 | 2011_08 | 531,326 | 70,552 | 241,226 |
| 2012-05-16 | 2012_05 | 536,029 | 70,868 | 245,342 |

*Received October 10, 2013; Revised December 25, 2013; Accepted January 20, 2014*

These computational methods all aim at designing an effective learning machine to predict the subcellular locations of the newly-found protein sequences on the base of the feature vector. For the feature vector, too many features may lead to a high-dimensional disaster. On the contrary, too few features would lose some necessary information. So the information in the feature vector influences the performance of the learning algorithm significantly. As a result, how to extract the informative features from the protein sequence is a key problem which needs to study deeply. Many feature extraction algorithms have been proposed by now. In this paper, three widely used ones related to amino acid composition were introduced, improved and implemented in a Java program.

The paper is organized as follows. In the next section, we introduce the three algorithms formally. In Section 3, the implementation method based on Java is described. Section 4 presents the result got from this program and gives discussions. Finally, we conclude our work in section 5.

## 2. The Proposed Three Algorithms

There are 20 native amino acids in the nature. The alphabetical order of their single-letter codes are A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, and Y. Amino acid is not only a basic unit composing protein, but is also an important physiological active substance. The features of amino acid determine the attribute of a protein to a great extent. So, the amino acid composition is usually considered into the feature extraction algorithms. A protein sequence which contains N amino acids can be formulated by:

$$R_1 R_2 ... R_N \tag{1}$$

Where $R_1$ represents the first amino acid, $R_2$ the second one, and so forth. On the base of the protein sequence and some physiochemical attributes of amino acids, three algorithms related to amino acid composition are introduced bellow.

### 2.1. Amino Acid Composition

The Amino Acid Composition (AAC) was first proposed by Nakashima [6] in studying the protein folding type problem in 1986. Later, it was used in predicting protein subcellular localizations and some other branches, such as predicting protein structural classes, predicting protein quaternary structure and so on. The form of AAC is a 20-dimensional digital vector defined as follows:

$$X = [\, f_1 \quad f_2 \quad \cdots \quad f_{20} \,]^T \tag{2}$$

Where $X$ denotes the feature vector of a protein, $f_1$, $f_2$, …, $f_{20}$ are the composition components (i.e. frequencies) of the 20 amino acids. The amino acid frequencies were calculated as follows. The percentage of the amino acid $i$ in a protein is defined by:

$$f_i = 100 \frac{n_i}{N} \qquad (i = 1,2,..., 20) \tag{3}$$

Where $n_i$ is the frequency of amino acid $i$, and $N$ is the number of amino acid residues in the protein sequence. AAC is simple and used broadly in the early days. However, the sequence order information is lost completely in AAC. This would result in a low accuracy in predicting protein attributes only using AAC.

### 2.2. Pseudo Amino Acid Composition

The Pseudo Amino Acid Composition (PseAAC) algorithm was proposed in the year 2001 by Chou [7]. It was designed to improve the prediction quality of protein attributes, including the subcellular localization and the membrane protein types. Compared with the conventional AAC, PseAAC not only converts the protein sequences with various lengths to

fixed-length digital vectors, but also keeps considerable sequence order information. The formulation of PseAAC is defined by:

$$X = [x_1 \cdots \quad x_{20} \quad x_{20+1} \cdots \quad x_{20+\lambda}]^T \tag{4}$$

Where,

$$x_u = \begin{cases} \dfrac{f_u}{\sum_{i=1}^{20} f_i + w\sum_{k=1}^{\lambda} \tau_k}, (1 \le u \le 20) \\ \dfrac{w\tau_{u-20}}{\sum_{i=1}^{20} f_i + w\sum_{k=1}^{\lambda} \tau_k}, (20+1 \le u \le 20+\lambda) \end{cases} \tag{5}$$

Where $f_i (i = 1,2,...,20)$ are the normalized occurrence frequency of the 20 amino acids in protein $X$, $w$ is the weight factor for the sequence order effect, and $\tau_k$ is the k-tier sequence correlation factor computed according to Equation (6) - (9) for this protein.

$$\tau_k = \frac{1}{L-k}\sum_{i=1}^{L-k} J_{i,i+k}, (k < N) \tag{6}$$

Where $\tau_1$ is called the first-tier correlation factor that reflects the sequence order correlation between all the most contiguous residues along a protein chain, $\tau_2$ the second-tier correlation factor, and so forth.

$$J_{i,i+k} = \frac{1}{3}\{[H_1(R_{i+k}) - H_1(R_i)]^2 + [H_2(R_{i+k}) - H_2(R_i)]^2 + [M(R_{i+k}) - M(R_i)]^2\} \tag{7}$$

Where $H_1(R_{i+k})$, $H_2(R_{i+k})$ and $M(R_{i+k})$ are, respectively, the hydrophobicity value, hydrophilicity value, and side-chain mass of amino acid $R_{i+k}$; and $H_1(R_i)$, $H_2(R_i)$ and $M(R_i)$ the corresponding values for amino acid $R_i$. These values were gotten by:

$$\begin{cases} H_1(R_i) = \dfrac{H_1^0(R_i) - < H_1^0 >}{SD(H_1^0)} \\ H_2(R_i) = \dfrac{H_2^0(R_i) - < H_2^0 >}{SD(H_2^0)} \\ M(R_i) = \dfrac{M^0(R_i) - < M^0 >}{SD(M^0)} \end{cases} \tag{8}$$

Where $H_1^0(R_i)$ is the original hydrophobicity value of the ith amino acid, $H_2^0(R_i)$ the original hydrophilicity value, and $M^0(R_i)$ the mass of the ith amino acid side chain. These three values are shown in Table 2. In Equation (8), $\langle \cdot \rangle$ denotes the mean of the corresponding value of all the 20 amino acids, and $SD(\cdot)$ means the variance. These two values are defined by Equation (9).

$$\begin{cases} \langle \cdot \rangle = \dfrac{1}{20}\sum_{i=1}^{20} \cdot_i \\ SD(\cdot) = \sqrt{\dfrac{\sum_{i=1}^{20} [\cdot_i - \langle \cdot \rangle]^2}{20}} \end{cases} \tag{9}$$

There are only three physicochemical properties (i.e., hydrophobicity, hydrophilicity and mass) used in the standard PseAAC. In order to add more properties into PseAAC, we enhanced it here. Let n be the number of properties, the new formulations are given as follows:

$$J_{i,i+k} = \frac{1}{n}\{[V_1(R_{i+k}) - V_1(R_i)]^2 + [V_2(R_{i+k}) - V_2(R_i)]^2 + \cdots + [V_n(R_{i+k}) - V_n(R_i)]^2\} \quad (10)$$

Where $V_1(R_{i+k})$, $V_2(R_{i+k})$ , ..., and $V_n(R_{i+k})$ are, respectively, the jth (j=1, 2, ..., n) property value of amino acid $R_{i+k}$; and $V_1(R_i)$, $V_2(R_i)$, ..., and $V_n(R_i)$ the corresponding values for amino acid $R_i$. The vlaues of these properties are gained by:

$$\begin{cases} V_1(R_i) = \dfrac{V_1^0(R_i) - < V_1^0 >}{SD(V_1^0)} \\ V_2(R_i) = \dfrac{V_2^0(R_i) - < V_2^0 >}{SD(V_2^0)} \\ \cdots\cdots\cdots\cdots \\ V_n(R_i) = \dfrac{V_n{}^0(R_i) - < V_n{}^0 >}{SD(V_n{}^0)} \end{cases} \quad (11)$$

The same method as Equation (9) was used to calculate each value in the above equation. After the reformation, any valuable property can be added into Equation (10). This improved the informativity and flexibility of PseAAC significantly.

Table 2. The Original Hydrophobicity, Hydrophilicity and Mass Values of 20 Amino Acids

| Number of amino acid | Letter of amino acid | Hydrophobicity | Hydrophilicity | Mass |
|---|---|---|---|---|
| 1 | A | 0.62 | -0.5 | 15.0 |
| 2 | C | 0.29 | -1.0 | 47.0 |
| 3 | D | -0.90 | 3.0 | 59.0 |
| 4 | E | -0.74 | 3.0 | 73.0 |
| 5 | F | 1.19 | -2.5 | 91.0 |
| 6 | G | 0.48 | 0.0 | 1.0 |
| 7 | H | -0.40 | -0.5 | 82.0 |
| 8 | I | 1.38 | -1.8 | 57.0 |
| 9 | K | -1.50 | 3.0 | 73.0 |
| 10 | L | 1.06 | -1.8 | 57.0 |
| 11 | M | 0.64 | -1.3 | 75.0 |
| 12 | N | -0.78 | 0.2 | 58.0 |
| 13 | P | 0.12 | 0.0 | 42.0 |
| 14 | Q | -0.85 | 0.2 | 72.0 |
| 15 | R | -2.53 | 3.0 | 101.0 |
| 16 | S | -0.18 | 0.3 | 31.0 |
| 17 | T | -0.05 | -0.4 | 45.0 |
| 18 | V | 1.08 | -1.5 | 43.0 |
| 19 | W | 0.81 | -3.4 | 130.0 |
| 20 | Y | 0.26 | -2.3 | 107.0 |

### 2.3. Amphiphilic Pseudo Amino Acid Composition

In 2005, Chou [8] proposed a novel representation of protein feature vector named "Amphiphilic Pseudo Amino Acid Composition" (AmPseAAC) in predicting enzyme subfamily classes. AmPseAAC contains $20+2\lambda$ discrete numbers: the first 20 numbers are the components of the conventional AAC; the next $2\lambda$ numbers are a set of correlation factors that reflect different hydrophobicity and hydrophilicity distribution patterns along a protein chain. The formulation of AmPseAAC is defined by:

$$X = [x_1 \quad \cdots \quad x_{20} \quad x_{20+1} \quad \cdots \quad x_{20+\lambda} \quad x_{20+\lambda+1} \quad \cdots \quad x_{20+2\lambda}]^T \quad (12)$$

Where,

$$x_u = \begin{cases} \dfrac{f_u}{\sum_{i=1}^{20} f_i + w\sum_{k=1}^{2\lambda} \tau_k}, (1 \le u \le 20) \\ \dfrac{w\tau_u}{\sum_{i=1}^{20} f_i + w\sum_{k=1}^{2\lambda} \tau_k}, (20+1 \le u \le 20+2\lambda) \end{cases} \quad (13)$$

Where $f_i (i = 1,2,...,20)$ are the normalized occurrence frequency of the 20 amino acids in the protein $X$, $\tau_k$ is the sequence correlation factor computed according to Equation (14) for this protein, and $w$ is the weight factor.

$$\begin{cases} \tau_1 = \dfrac{1}{N-1}\sum_{k=1}^{N-1} H_{k,k+1}^1 \\ \tau_2 = \dfrac{1}{N-1}\sum_{k=1}^{N-1} H_{k,k+1}^2 \\ \cdots\cdots\cdots\cdots \\ \tau_{2\lambda-1} = \dfrac{1}{N-\lambda}\sum_{k=1}^{N-\lambda} H_{k,k+\lambda}^1 \\ \tau_{2\lambda} = \dfrac{1}{N-\lambda}\sum_{k=1}^{N-\lambda} H_{k,k+\lambda}^2 \end{cases}, (\lambda < N) \quad (14)$$

Where $\tau_1$ and $\tau_2$ are called the first-tier correlation factors that reflect the sequence-order correlation between all the most contiguous residues along a protein chain through hydrophobicity and hydrophilicity, respectively, $\tau_3$ and $\tau_4$ the second-tier correlation factors, and so forth. In Equation (14), $H_{k,j}^1$ and $H_{k,j}^2$ are hydrophobicity and hydrophilicity correlation functions given by:

$$\begin{cases} H_{k,j}^1 = h^1(R_k) \cdot h^1(R_j) \\ H_{k,j}^2 = h^2(R_k) \cdot h^2(R_j) \end{cases} \quad (15)$$

Where $h^1(R_i)$ and $h^2(R_i)$ are hydrophobicity and hydrophilicity values for the ith (i=1, 2, …, N) amino acid in this protein respectively. Note that $h^1(R_i)$ and $h^2(R_i)$ are calculated using the same way as described in section 2.2.

Similarly, there are only two physicochemical properties (i.e., hydrophobicity and hydrophilicity) which are considered in the standard AmPseAAC. Based on the same mode as PseAAC, we enhanced AmPseAAC here also. Let n be the number of properties, the new formulations of Equation (12) - (15) are given as the following four equations:

$$X = [x_1 \quad \cdots \quad x_{20} \quad x_{20+1} \quad \cdots \quad x_{20+\lambda} \quad x_{20+\lambda+1} \quad \cdots \quad x_{20+n\lambda}]^T \quad (16)$$

$$x_u = \begin{cases} \dfrac{f_u}{\sum_{i=1}^{20} f_i + w\sum_{k=1}^{n\lambda} \tau_k}, (1 \le u \le 20) \\ \dfrac{w\tau_u}{\sum_{i=1}^{20} f_i + w\sum_{k=1}^{n\lambda} \tau_k}, (20+1 \le u \le 20+n\lambda) \end{cases} \quad (17)$$

$$\begin{cases} \tau_1 = \frac{1}{N-1}\sum_{k=1}^{N-1} V_{k,k+1}^1 \\ \tau_2 = \frac{1}{N-1}\sum_{k=1}^{N-1} V_{k,k+1}^2 \\ \cdots\cdots\cdots\cdots \\ \tau_n = \frac{1}{N-1}\sum_{k=1}^{N-1} V_{k,k+1}^n \\ \cdots\cdots\cdots\cdots \\ \tau_{n\lambda-(n-1)} = \frac{1}{N-\lambda}\sum_{k=1}^{N-\lambda} V_{k,k+\lambda}^1 \\ \tau_{n\lambda-(n-2)} = \frac{1}{N-\lambda}\sum_{k=1}^{N-\lambda} V_{k,k+\lambda}^2 \\ \cdots\cdots\cdots\cdots \\ \tau_{n\lambda} = \frac{1}{N-\lambda}\sum_{k=1}^{N-\lambda} V_{k,k+\lambda}^n \end{cases}, (\lambda < N) \quad (18)$$

$$\begin{cases} V_{k,j}^1 = v^1(R_k)\cdot v^1(R_j) \\ V_{k,j}^2 = v^2(R_k)\cdot v^2(R_j) \\ \cdots\cdots\cdots\cdots \\ V_{k,j}^n = v^n(R_k)\cdot v^n(R_j) \end{cases} \quad (19)$$

## 3. Research Method

In order to realize the functions of the above three improved algorithms, a class named Representation was defined in Java. At the same time, a flexible mode was adopted to utilize more attributes in the algorithms. That is the number of attributes used in PseAAC and AmPseAAC is no longer limited to three and two. Useres can use any attribute to calculate PseAAC and AmPSeAAC. All the methods in pseudo-code were described bellow in detail.

### 3.1. AAC Feature Vector

```
Input the protein sequence P;
Calculate the sequence length N;
Initialize a vector named vs with the type of double[] and the length of 20;
For each amino acid AAi in P
        Count the number of AAi and save it into vs;
Endfor
Normalize vs into [0, 100) interval according to eq. (3);
Return vs;
```

### 3.2. PseAAC Feature Vector

```
Input the protein sequence P, w, λ and attribute values;
Calculate the sequence length N;
Call the AAC algorithm to get the vector of AAC named vs1;
Initialize a vector named vs2 with the type of double[] and the length of λ;
Convert each attribute value to its standard form according to eq. (9) and (11);
For i = 1 to λ
        For j = 1 to N – i
                For each attribute value
                        Get the two continuous amino acids: j and j + i;
                        Calculate the attribute value according to eq. (10);
                Endfor
        Endfor
        Calculate the i-tier correlation factor according to eq. (6) and save it into vs2;
Endfor
Connect vs1 and vs2 together into a new vector named vs with the length of 20+λ;
Normalize vs into [0, 100) interval;
Return vs;
```

### 3.3. AmPseAAC Feature Vector

Input the protein sequence P, $w$, $\lambda$ and attribute values;
Calculate the sequence length N;
Call the AAC algorithm to get the vector of AAC named vs1;
Calculate the number of attributes n;
Initialize a vector named vs2 with the type of double[] and the length of $n\lambda$;
Convert each attribute value to its standard form according to eq. (9) and (11);
For i = 1 to $n\lambda$
    For each attribute value
        For j = 1 to N – ((i -1) / n + 1)
            Get the two continuous amino acids: j and j + ((i -1) / n + 1);
            Calculate the attribute value according to eq. (19);
        Endfor
    Endfor
    Calculate the i-tier correlation factor according to eq. (18) and save it into vs2;
Endfor
Connect vs1 and vs2 together into a new vector named vs with the length of $20+n\lambda$;
Normalize vs into [0, 100) interval;
Return vs;

## 4. Results and Discussion

In order to use these algorithms easily, a friendly graphical user interface, as shown in Figure 1, was provided.

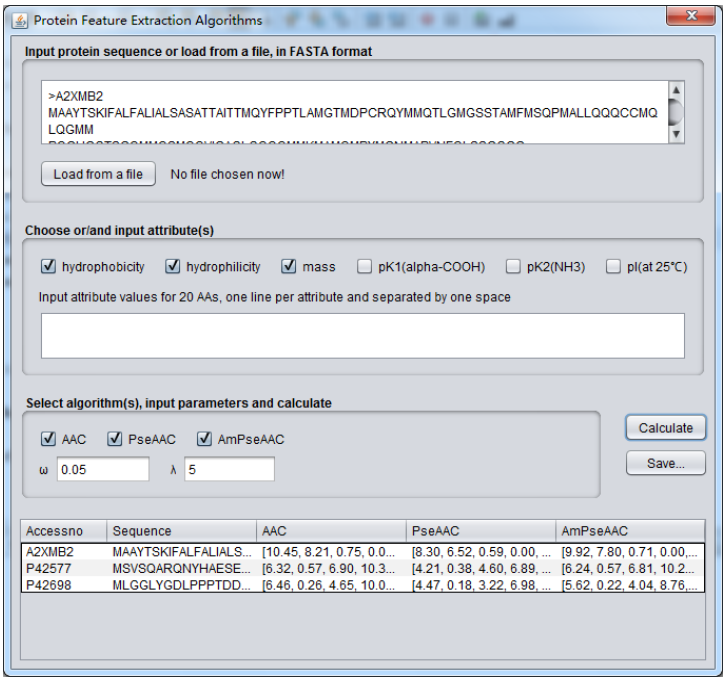


Figure 1. The Graphical User Interface

Through this interface, users can input or load sequences, choose or/and input attribute values, select algorithms and input the parameters. Allow for the batch calculating in some time, users can load a number of protein sequences from a file with the FASTA format. In the interface, the five attributes which are used frequently are provided directly. For more attributes, users can input them in the text area according to the specified format. For PseAAC and AmPseAAC, the parameter $w$ and $\lambda$ are needed while AAC is not. To make it easier for users in operating, the program will enable or disable these two components automatically according

to the option of users. After the necessory data are all given, “Calculate” function will calculate and show the results in a JTable field dynamicly. Each protein contains its access number, sequence and the values of the selected algorithm(s). To press the “Save...” button will save the results to a specified file with the style of one line per protein in text format.

We chose three sequences to test this program. The results proved that it run normally. To verify the results are correct or not, we compared these data with results gotten from http://www.csbio.sjtu.edu.cn/bioinf/PseAAC [9]. The comparing result showed that our program gave the correct answers. Comparing to PseAAC, with the features increased, the consumed time was much shorter in our program than that in PseAAC. Moreover, PseAAC cannot add other features except for the predefined ones in it.

## 5. Conclusion

The feature extraction is a key part in designing the learning algorithm to predict the subcellular localizations of proteins. The three widely used feature extraction algorithms were improved in this work and their corresponding pseudo-codes were depicted in detail. The friendly interface would provide convenience to users. Each algorithm can create a digital feature vector fast. On the base of these vectors, a new vector can be created through the weighting or fusion strategy. This new vector would give a good performance to predicting algorithms. An intelligent computing algorithm, such as PSO [10] and GA [11], for creating a new optimized vector will be developed in the future.

## Acknowledgements

The work was supported by National Natural Science Foundation of China under Grant No. 61302128 and Doctoral Foundation of University of Jinan under Grant No. XBS1318.

## References

[1] Kuochen C, Hongbin S. Recent Progress in Protein Subcellular Location Prediction. *Anal Biochem*. 2007; 370(1): 1-16.
[2] Imai K, Nakai K. Prediction of Subcellular Locations of Proteins: Where to Proceed?. *Proteomics*. 2010; 10(22): 3970-3983.
[3] Kuochen C. Some Remarks on Protein Attribute Prediction and Pseudo Amino Acid Composition. *J Theor Biol*. 2011; 273(1): 236-247.
[4] Pufeng D, Chao X. Predicting Multisite Protein Subcellular Locations: Progress and Challenges. *Expert Rev Proteomics*. 2013; 10(3): 227-237.
[5] Kuochen C. Some Remarks on Predicting Multi-Label Attributes in Molecular Biosystems. *Mol Biosyst*. 2013; 9(6): 1092-1100.
[6] Nakashima H, Nishikawa K, Ooi T. The Folding Type of a Protein Is Relevant to the Amino Acid Composition. *J Biochem*. 1986; 99(1): 153-162.
[7] Kuochen C. Prediction of Protein Cellular Attributes Using Pseudo-Amino Acid Composition. *Proteins: Struct, Funct, Genet*. 2001; 43(3): 246-255.
[8] Kuochen C. Using Amphiphilic Pseudo Amino Acid Composition to Predict Enzyme Subfamily Classes. *Bioinformatics*. 2005; 21(1): 10-19.
[9] Hongbin S, Kuochen C. PseAAC: A Flexible Web Server for Generating Various Kinds of Protein Pseudo Amino Acid Composition. *Anal Biochem*. 2008; 373(2): 386-388.
[10] Yu M, Lichen G. Fuzzy Immune PID Control of Hydraulic System Based on PSO Algorithm. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2013; 11(2): 890-895.
[11] Xuesong Y, Qinghua W, Can Z, etc. An Improved Genetic Algorithm and Its Application. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2012; 10(5): 1081-1086.