Australian Journal of Basic and Applied Sciences, 8(6) April 2014, Pages: 21-29

AENSI Journals

Australian Journal of Basic and Applied Sciences

ISSN:1991-8178

Journal home page: www.ajbasweb.com

Email Spam Detection Generation Algorithm for Negative Selection Algorithm with Hamming Distance Partial Matching Rules

[1]Ismaila Idris, [2]Shafi'i Muhammad Abdulhamid, [3]Abdulmalik Mohammed, [4]Umar Dauda Suleiman, [5]Nicholas Akosu

[1,2]Department of Cyber Security Science, Federal University of Technology, P.M.B 65, Minna, Niger State, Nigeria.
[3]Department of Computer Science, Federal University of Technology, P.M.B 65, Minna, Niger State, Nigeria.
[4]Department of Electrical and Electronics Engineering, Federal University of Technology, P.M.B 65, Minna, Niger State, Nigeria.
[5]Department of Computer Science, Federal Polytechnic Nasarawa, Nasarawa State, Nigeria.

ARTICLE INFO

Article history:
Received 25 January 2014
Received in revised form
8 April 2014
Accepted 20 April 2014
Available online 10 May 2014

Keywords:
Negative selection algorithm, time, accuracy, detector generation, r-chunk, hamming.

ABSTRACT

Negative selection algorithms (NSAs) are inspired by artificial immune system. It creates techniques that aim at developing the immune based model. This is done by distinguishing self from non-self spam in the generation of detectors. In general, NSAs has an exponential run time. This research study the significance of time and accuracy for two commonly used matching rules. The hamming and r-chunk matching rules, based on different threshold values (r) for generating set of fixed number of detectors. The results show the differences between the mean values of time and accuracy for hamming and r-chunk matching rules. Statistical t test shows that the difference between hamming and r-chunk matching rule are insignificant for accuracy while it is significant for time.

To Cite This Article: Ismaila Idris, Shafi'i Muhammad Abdulhamid, Abdulmalik Mohammed, Umar Dauda Suleiman, Nicholas Akosu., Email Spam Detection Generation Algorithm for Negative Selection Algorithm with Hamming Distance Partial Matching Rules. ***Aust. J. Basic & Appl. Sci.***, *8(6): 21-29, 2014*

INTRODUCTION

Negative selection algorithm (NSA) has been used successfully for a broad range of application in the construction of artificial immune system (Balthrop, J., *et al*., 2002). The algorithm is an approach that deals with anomaly detection by the use of negative selection which was originally proposed by Forest *et al* (1994). Negative selection algorithm, will not react to the self cells uses the immune system capability to detect unknown antigens. Its mechanism protects body against self reactive lymphocytes. Receptors are made through a pseudo-random genetic re-arrangement process during the generation of T-cells (Wang, C. and Y. Zhao, 2008); The immature T-cell undergo both positive selection and negative selection procedure in the thymus. The NSA was inspired by the negative selection techniques that exist within the natural immune system (NIS) (Cantu-Ortiz, F.J., 2014; Travé-Massuyès, L., 2014). In this process, T-cell that those not bind to self protein are destroyed. In a nutshell, the immunological function and protection of the body against foreign antigens is possible through circulation of matured T-cells (Zhang, Y., *et al*., 2010). The main concept behind the negative selection algorithm is to generate sets of candidates, c such that $\not\forall x_{i\epsilon} \in s\, f_{MATCH}\left(x_{i,}z_p\right) < r$. The concept is illustrated in figure 1. This paper considered negative selection algorithm in binary classification operating on a string space. Classification is one of the familiar techniques used in machine learning. Patterns belonging to different classes are discriminated due to the generation of decision boundaries.

Single global affinity threshold, r, was originally used by forest et al with the r- contiguous matching rule for each individual artificial lymphocytes (ALC) around the population of the ALCs, C. A process of trial and error is use to determine the affinity threshold, then, the threshold that yields the best system performance will be chosen as the targeted threshold for the system. A frame work that will aid in the chosen of an optimum value for r in conjunction of r-contiguous rule was provided by (Ayara, M., *et al*., 2002). Previous research work that adopts the partial matching rule with hamming distance shows no efficient detector generated (Haiyu Hou and G. Dozier, 2006). This paper analyzes run time for generating efficient detector algorithm that adopt the hamming distance matching rule while comparing its performance with the r-chunk matching rule.

Section 2 presents related work and the contribution of this paper while section 3 discuss the algorithm of the r-chunk matching rule and section 4 discusses the algorithm of the hamming distance matching rules. In

Corresponding Author: Ismaila Idris, Department of Cyber Security Science, Federal University of Technology, P.M.B 65, Minna, Niger State, Nigeria.
E-mail: ismaila.idris95@gmail.com.

section 5 we describe the dataset and in section 6 we present our results as well as discussion and finally we conclude the paper in section 7.

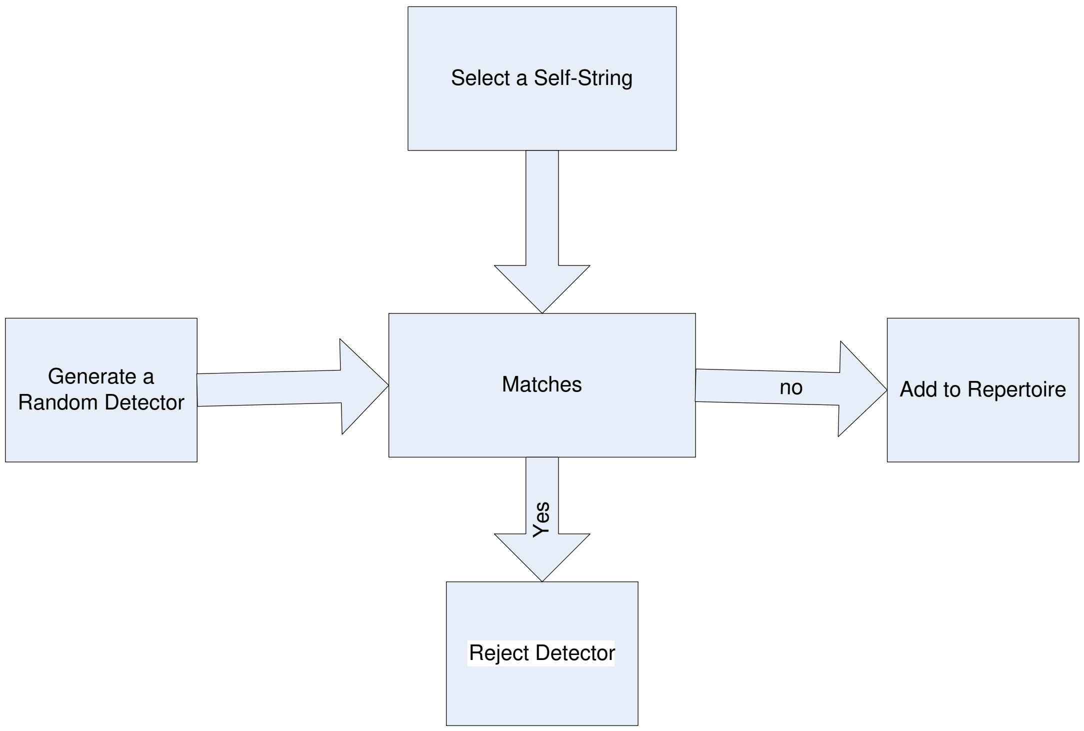

Fig. 1: Detector Set Generation of Negative Selection Algorithm (Forrest *et al*, 1995).

2. Related Work and Contribution:

String based negative selection algorithm with r-chunk and hamming distance was presented in this paper by the application of running time of randomized selection. The technique improved by analyzing the complexity and run time of the proposed hamming distance detector generation algorithm. R-chunk is a string of length r (or r-gram) that matches all string with positive index in which the r-gram occurs at that position while the hamming distance is a string of length (L) and matches string that are not similar in r hamming position.

The research of (Elberfeld, M. and J. Textor, 2009; Elberfeld, M. and J. Textor, 2011) shows that polynomial time NSAs can be obtained by the use of "detector compression" techniques. Though, it was also proved by (Maciej Liśkiewicz and J. Textor, 2010) that detector compressor techniques are not applicable to many interesting type of detector; this therefore creates an assumption of generating an efficient NSA through some other techniques. (Textor, J., 2012) uses markov chain, monte carlo approach to efficiently generate uniform sample of the set of s-consistent detector approximately. These techniques contradict the detector compression techniques that were established in (Maciej Liśkiewicz and J. Textor, 2010). Both (Textor, J., 2012) never demonstrate their approach in practice. A linear bound was proved in addressing the problem of generating detector by (D'Haeseleer, P., 1996; D'Haeseleer, P., *et al*., 1996), a structured exhaustive research algorithm was presented for generating detectors but the algorithm contains a run time exponential in r. More so, (Stibor, T., *et al*., 2004; Wierzchoń, S., 2000) also propose a similar heuristics with an effort of trying to make clear the computational complexity of negative selection algorithm. (Stibor, T., 2007; Stibor, T., 2008) also studied associated decision problem. The time complexity of NSA also brought about some other algorithm such as the greedy algorithms, linear time algorithm, binary template algorithms (Wang, D., *et al*., 2013) and negative mutation with mutation (Hassanien, A.E., *et al*., 2013). All of these algorithms are linear in the size of self-due to its complexity but still having short coming from time complexity and generation of efficient detector.

Balthrop *et al* (2002) first proposed the r-chunk matching rules which are an improvement over the r-contiguous matching rule proposed by (D'Haeseleer, P., *et al*., 1996). In r-contiguous, two elements of the same length match if given r-contiguous character that are similar. The matching rule was theoretical and also investigated practically in (Haiyu Hou and G. Dozier, 2006). In AIS literature, most of the matching rules performances were compared by Gonzalez *et al* (2006). In his research work, he noted that the r-chunk matching rules achieve higher matching performance compared with other matching rules over binary representation. The performance of matching is evaluated by the number of detector generated by the r-chunk matching rule over binary representation. An evaluation of NSA with r-chunk and r-contiguous detector generation was also presented by (Textor, J., 2012), this was achieved by making comparison against other method based on kernels, finite state automata and n-gram frequencies. The evaluation confirm that NSA performed competitively by yielding an average better performance. (Chen, J. and D. Yang, 2011) implement the generation of detectors in three main processes after listing and analyzing the binary matching rules. The process includes gene library, clone selection and negative selection. Several new techniques are adopted to improve the performance of NSA while constructing a co-evolutionary detector generation model. An improved NSA by introducing a novel training is also proposed by (Gong, M., *et al*., 2012). The technique was implemented in the training phase to generate self detectors to cover the self region.

Though this paper focus on traditional NSA with detector generation time for two popular matching rules but other optimized techniques proposed in recent times to improve the detector generation with respect to improving the generation time overlapping of self region and the coverage of non self space are the work of (Aziz, A.S.A., *et al*., 2012; Mohammad, A.H. and R.A. Zitar, 2011). The proposed an improvement of the standard NSA by introducing genetic algorithm for optimization. Optimizing spam detection using genetic algorithm shows a reasonable improvement from the traditional NSA.

3. *Matching Rules:*

Matching rules define the distance measure and matching or recognition which is the foundation in any recognition, classification or detection process based on negative selection algorithm. The matching rules are categories into two phases. The first phase is the detection generation and secondly is the spam detection. Irrespective of the representation, the matching rule is defined formally as: $dMx \leftrightarrow$ distance measure between d and x is between a threshold where as d is the detector and x is a data instance. In this study, two choice of matching rules in string representation is adopted, the r-chunk matching rule and hamming distance matching rule.

3.1 *The R-chunk Matching Rules:*

Let's assume a space U_l^{Σ}, which contains all elements of length l over a string representation $\sum$ and a detection set $D \subset U_l^{\Sigma}$.

Definition 1:

An element $e \in U_l^{\Sigma}$ with $e = e_1, e_2$,---------------e_l and detector $d \in D$ with $d = (p, d_1, d_2, \dots\dots\dots.. d_r$), with $r \leq l,\ p \leq l - r + 1$ match with r-chunk rule $iff\ e_i = d_i\, for\ i = p, \dots\dots\dots.., p + r - 1$.

The detector and the element will match if there exist a sequence of length r at position p where all the characters are similar.

3.2 *Detector Generation Algorithm with R-chunk Matching Rules:*

Input:
r = r-chunk parameter. If the r-chunk value between two strings is smaller than $l - r$, the two string are said to match, therefore, if the two string are similar with less than r bits in the corresponding positions, the two string match.
l = The string length.
D size: No of mature detector required
R = The detector set R is initially an empty set
U_l^{Σ} : Dataset
Output:
$D \subset U_l^{\Sigma}$: Set of matured detector
T: Time elapse in generating detector.
Start:

```
1. Generate self and non-self sample from $U_l^{\Sigma}$
2. R = $\emptyset$
3. While number of generated detector $< Dsize$ // generate a string randomly as a candidate detector.
4.  Generate $D \subset U_l^{\Sigma}$
5. for any self strings in $S$, if $d$ matches, go to (3)
6.  Add generated detector to matured detector set $D \subset U_l^{\Sigma}$
7. $R \leftarrow R \cup \{d\}$
```

```
8. The algorithm terminates if enough size or go to (3)
```

Fig. 2: R-chunk matching rule algorithm.

For efficient detector generation algorithm, the time cost in generating detector requires generating a number of candidate detector which as to be larger than the expected number of detector. This will be applicable to the hamming distance matching rule and the r-chunk matching rule.

Let's assume N_s to denote the size of the self set, N_{R0} represent the number of candidate detectors and N_R indicate the size of the detector set.

Therefore, time cost of the algorithm is proportional to N_{R0} and N_s and the space is determined by N_s

Time complexity: $O(N_{R0}\ .\ N_s) = 0\ \left[\frac{-\ln Pf}{P_m(1-P_m)} N_s\right]$

Space complexity: $O(N_s\ .\ l)$

4. Hamming Distance Matching Rule:

Farmer *et al*. (1986) proposed one of the very first research that modeled Biological Immune System (BIS) concepts in developing pattern recognition. This research work proposed a computational model of the BIS based on the idiotypic network theory of Jerne (), and it was compared with the learning classifier system (Holland, J.H., *et al*., 1987). This model is a binary model that represents antibodies and antigens and it defines a matching rule that is based on Hamming distance. Hamming distance matching rule can be defined as shown below:
given a binary string x = x_1 , x_2 ……x_n and also detector d= d_1, d_2……d_n

d matches x ≡ $\sum x_i \ \oplus \ d_i \ \geq r$

$\oplus$ is the exclusive-or operator, and 0 ≤ r≤_ n is going to be the threshold value.

4.1 Detector Generation Algorithm for Negative Selection Algorithm with Hamming Distance Partial Matching Rule:

Definition 2: Self Space: A self space of order i is a size l string consisting of $l - i$ symbol "*'' lets assume an example *11*11* is a self space of order 4 with three "*''.

A detector is a string that is made up of [0,1,*] in this research work and "*'' can match with ''0'' and ''1''. A self space with ''*'' is a detector if it refuses to match self set. This we refer to as candidate detector.

Assuming if $l = 4, r = 3,$ the self set is said to be {0010,1001}, self space is ''111*" which is said to be a valid detector set.

This definition can enhance the coverage area of a detector and also reduce the number of detectors needed for a given detector rate.

In generating candidate detector for the self space; we assume a self string $V = x_1x_2x_3 \ldots\ldots x_l$, a candidate detector Self Space S_v with order $c(c = l - r + 1)$, randomly selected c bits of v and flip these c bits and let $r - 1$ bits as ''*''.

The S_v will therefore be $\binom{l}{r-1}$.
More so, the candidate detector of both self string and self space is assumed as;

Given a set of self string $v = y_1y_2y_3 \ldots\ldots\ldots y_l$ and a candidate detector self space S with order c, a candidate self space $S_{s,v}$ with order $c + k$ is constructed by the following techniques.

If $k \leq l - c,$ select $k - bits$ of $l - c$ ''*'' bits randomly while setting the value of the bits was selected by flipping corresponding bits of S, and $l - c - k$ bits = ''*''. Meanwhile, if $k > l - c$, the candidate detector self space $S_{s,v}$ will not exist.

Therefore, the possible number of $S_{s,v}$ is $(\frac{l-c}{k})$.

4.2 Detection Generation Algorithm with Hamming Distance Matching Rule:

```
Input:
1.  Denote self set as $s_1, s_2, s_3, \ldots\ldots\ldots s_{NS}$
2.  Initialize $U$ =∅. Valid detectors are stored in $U$
3.  $S_r$ $(1 \leq r \leq N_s)$ is a self string selected at random
Output
```

```
4.  A candidate detector with order $c(c = l - r + 1) of S_r$ was randomly generated by self set. The candidate detector self space is denoted by $d$.
5.  $d = r - 1$ ‘‘*’’. Assume $m = r - 1$
Start
6.  Initialize $i = 0$
7.  Set $i = i + 1$
If $i = r,$ then go to (6)
else,
 $if\ i > N_s,\ U \leftarrow U \cup \{d\}$. If all condition justified and number of mature detector attain.
End, else go to (3),
$if\ i \leq N_s$
(i)   Calculate bits of both $d\ and\ s_i$ (self set that are similar) where $d\ is\ denoted\ by\ k$ and determinate. Therefore ‘‘*’’ is considered when calculating $k$.
(ii)  $if\ k \geq r,$ discard $d$ and go to (3)
(iii) $if\ k = r - 1,$ all ‘‘*’’ bits of $d$ are replaced by corresponding $s_i$ and set $m = 0$ go to (6)
(iv)  $if\ k < r - 1\ and\ k + m \leq r - 1$ Self space $d$ of the candidate detector and ‘‘*’’ $m$ bits remain unchanged, go to (6)
(v)   $if\ k < r - 1\ and\ k + m > r - 1,$ randomly generated candidate detector self space $S$ with order $l - (r - 1 - k)$ of both $d\ and\ s_i$.
```

Fig. 3: Hamming distance matching rule algorithm.

5. *Dataset:*

The corpus bench mark is obtained from spam base data set which is an acquisition from email spam message. In acquiring this email spam message, it is made up of 4601 messages and 1813 (39%) of the message are marked to be spam messages and 2788 (61%) are identified as non-spam which was acquired by (Hopkins, M., *et al*., 1999). The non-spam message was contributed by George Forman, this was acquired from a single mail box. The attribute description is as tabulated in table 1. Preprocessing is very important task that needed to be done before performing any data mining process. The major task is data cleaning, data integration, data reduction and data transformation. This dataset undergo data normalization before performing the feature relevance analysis.

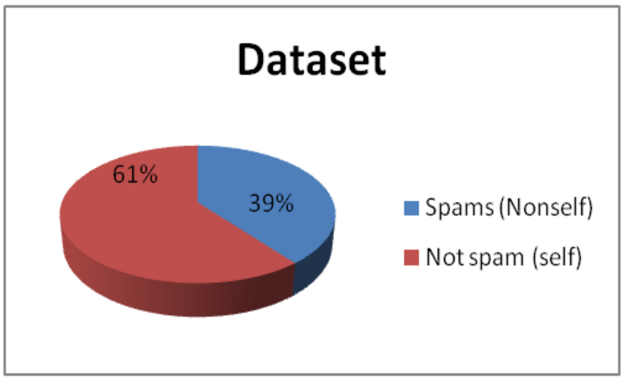

Fig. 4: Dataset Analysis.

5.1 Feature Relevance Analysis:

Table 1: Feature relevance analysis.

Numberof Attribute (NA)	Type of Attribute	Description of Attributes
NA 1 to NA 48	Char_freq_CHAR	Percentage of characters in the email that match CHAR
NA 69 to NA 54	Capital_run_lenght_average	Average length of uninterrupted sequence of capital letters
NA 55	Capital_run_lenght_longest	length of longest uninterrupted sequence of capital letters
NA 56	Capital_run_lenght_longest	length of longest uninterrupted sequence of capital letters
NA 57	Capital_run_length_total	Total no of capital letters in the email
NA 58	Class attribute	Denotes if the email is either spam (1) or non-spam (0)

RESULTS AND DISCUSSION

6.1 Results generated for both Hamming distance matching rule with 1000 generated detectors:

Table 2: Results for hamming distance matching rules.

Matching rule	Threshold	Time	True Positive	False Negative	Accuracy
Hamming	31	28.188	12.56	15.76	78.07
	32	12.594	12.64	15.69	78.18
	33	7.375	12.62	15.70	78.16
	34	5.109	12.33	16.00	77.74
	35	3.906	11.47	16.86	76.55
	36	3.14	10.89	17.43	75.74
	37	2.797	10.11	18.22	74.66
	38	2.579	9.45	18.87	73.74
	39	2.484	7.17	21.15	70.57
	40	2.203	3.30	25.03	65.18
	41	2.141	1.75	26.57	63.03
	42	2.171	0.91	27.42	61.86
	43	2.11	0.84	27.48	61.77
	44	2.157	0.34	27.98	61.07

Results generated for r- chunk matching rule with 1000 generated detectors.

Table 3: Results for r-chunk matching rules.

Matching rule	Threshold	Time	True Positive	False Negative	Accuracy
r-chunk	10	34.282	15.90	12.42	82.72
	11	19.922	16.33	12.00	83.31
	12	14.328	15.67	12.65	82.40
	13	11.406	12.29	16.03	77.70
	14	9.922	9.14	19.18	73.31
	15	8.797	6.05	22.28	69.01
	16	8.375	3.20	25.12	65.05
	17	7.829	2.55	25.78	64.14
	18	7.578	0.78	27.54	61.68
	19	7.296	0.28	28.04	60.99
	20	6.906	0.37	27.95	61.12
	21	6.75	0.03	28.29	60.64
	22	6.594	0.14	28.18	60.79
	23	6.391	0.06	28.26	60.68
	24	6.125	0.02	28.31	60.62

6.2 Time VS Accuracy:

6.2.1 Hamming Distance Matching Rule:

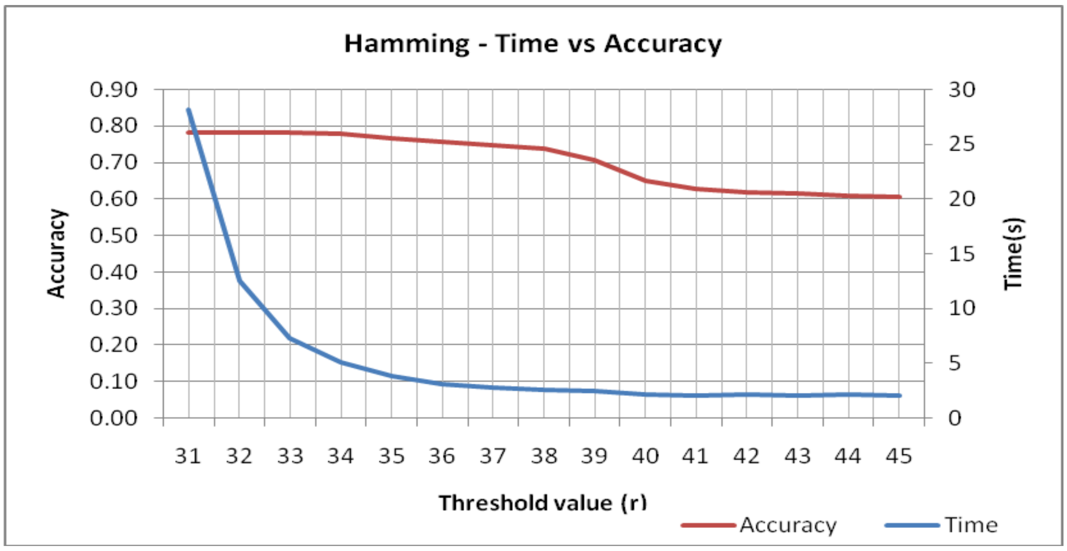

Fig. 5: Time against accuracy in hamming distance matching rule.

Figure 5 shows an increase in threshold causes a decrease in accuracy as well as the time required to generate detectors. Also, decreasing threshold increases the accuracy and the required time. However, on the threshold value of 31, the accuracy was 78% and the time required to generate 1000 detectors was 27 seconds as shown in figure 5. Increasing threshold will cause an exponential increment in time required to generate detector, while accuracy remains constant. The best value of threshold (r) is in the range of 32 and 33, 1000 detectors were generated in less than 10 seconds with accuracy value at 78% in this range.

6.2.2. R-chunk Matching Rule:

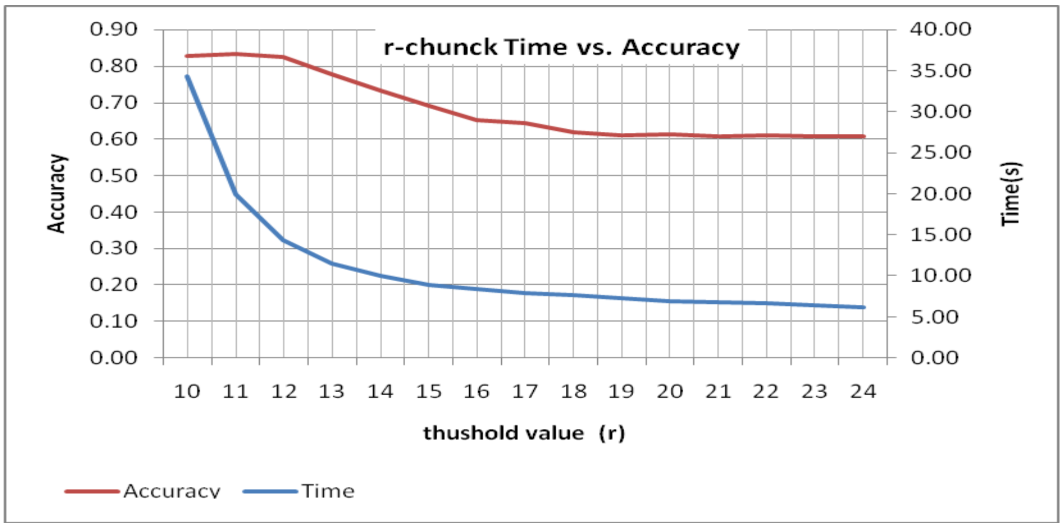

Fig. 6: Time against accuracy in r-chunk matching rule.

Similar to figure 5, Figure 6 shows that an increment in threshold causes a decrement in accuracy and time required to generate detectors. Accuracy and the required time are decreased with an increase in threshold value. Vise versa a decrement in threshold is inversely proportional to accuracy and time. However, for threshold less than 12, accuracy becomes constant at 83% while the time required starts increasing exponentially. The best value of threshold is 12 because 1000 detectors can be generated in less than 15 seconds with accuracy of 82%.

6.2.3 Comparison between Hamming Distance and R-chunk Matching Rules:

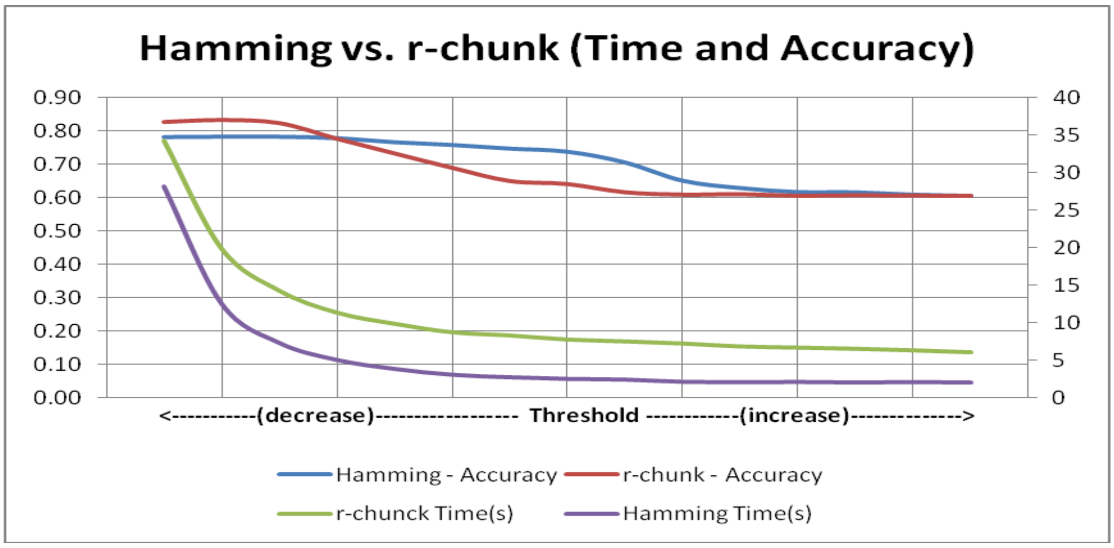

Fig. 7: Result comparison of r-chunk and hamming distance.

Figure 7 discus the comparison between hamming distance and r-chunk matching rules. Increasing threshold causes decrease in accuracy and the time required for both hamming distance and r-chunk matching rules. However, accuracy for r-chunk is greater than hamming distance for some threshold value. Overall average accuracy for hamming is greater than r-chunk average as shown in figure 8.

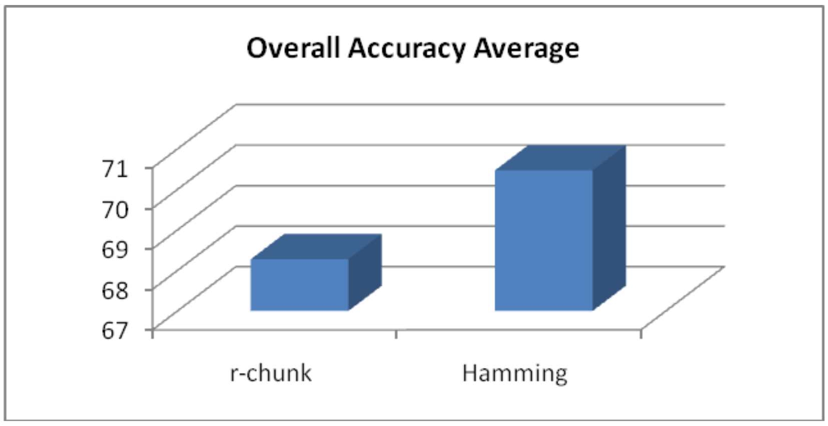

Fig. 8: Overall accuracy average.

In general, time required to generate detector in case of hamming is less than the required time in r-chunk. The time required to generate 1000 detector in r-chunk is 15 seconds with 83% accuracy compared with the time required to generate same number of detector with hamming distance which is 27seconds with 78% accuracy. This shows a good performance base on accuracy vs time, i.e. higher accuracy with less time.

To check for the difference significance in time, true positive, false negative and accuracy, we perform a statistical t-test among the result of both matching rule, next sub-section discuss result of this test.

6.2.4 Statistical t-test:

Table 4: Statistical t-test.

	Matching Rule	Mean	Std. Deviation	Sig. (2-tailed)
Time	Hamming	5.40	6.91	0.048
	r-chunk	10.83	7.47	
True Positive	Hamming	9.87	7.27	0.471
	r-chunk	7.68	9.087	
False Negative	Hamming	29.53	7.27	0.472
	r-chunk	31.72	9.08	
Accuracy	Hamming	70.47	7.27	0.472
	r-chunk	68.28	9.08	

It can be seen from table 4 that there are difference in mean value of both matching rules on time, true positive, false negative and accuracy. The absolute mean difference for time between hamming and r-chunk matching rule is 5.43, however it is 2.19 for true positive, false negative and accuracy. Also, it can be seen that the difference is insignificant for all measurements except time.

7. Conclusion:

In this paper we show that there are difference between the mean value of r-chunk and hamming distance matching rule for accuracy and time. Our statistical test shows that this difference is insignificant for accuracy, true positive, false negative but significant for time. Though, despite the insignificance for other measures, it is best to use r-chunk as a matching rule in generating detectors since time is one of the most important factors in generating detectors. Our future work will focus on the combination of the hamming and r-chunk matching rule in generating detectors to improve the performance of detector generation.

REFERENCES

Ayara, M., *et al*., 2002. "Negative selection: How to generate detector.," 1st International Conference on Artificial Immune Systems, pp: 89-98.

Aziz, A.S.A., *et al*., 2012. "Detectors generation using genetic algorithm for a negative selection inspired anomaly network intrusion detection system," in Federated Conference on Computer Science and Information Systems (FedCSIS), pp: 597-602.

Balthrop, J., *et al*., 2002. "Revisiting LISYS: parameters and normal behavior," in Proceedings of the 2002 Congress on Evolutionary Computing, pp: 1045-1050.

Cantu-Ortiz, F.J., 2014. "Advancing artificial intelligence research and dissemination through conference series: Benchmark, scientific impact and the MICAI experience," Expert Systems with Applications, 41: 781-785.

Chen, J. and D. Yang, 2011. "A study of detector generation algorithms based on artificial immune in intrusion detection system," in Computer Research and Development (ICCRD), 2011 3rd International Conference on, pp: 4-8.

D'Haeseleer, P., 1996. "An immunological approach to change detection: theoretical results," in Proceedings., 9th IEEEComputer Security Foundations Workshop, pp: 18-26.

D'Haeseleer, P., *et al*., 1996. "An immunological approach to change detection: algorithms, analysis and implications," in Proceedings on IEEE Symposium on Security and Privacy, pp: 110-119.

Elberfeld, M. and J. Textor, 2009. "Efficient Algorithms for String-Based Negative Selection," in Artificial Immune Systems. vol. 5666, P. Andrews, et al., Eds., ed: Springer Berlin Heidelberg, pp: 109-121.

Elberfeld, M. and J. Textor, 2011. "Negative selection algorithms on strings with efficient training and linear-time classification," Theoretical Computer Science, 412: 534-542.

Farmer, J.D., *et al*., 1986. "The immune system, adaptation, and machine learning.," Phys. D, 2: 187-204.

Forrest, S. and A. Perelson, 1994. "Self nonself discrimination in computer.,".

Gong, M., *et al*., 2012. "An efficient negative selection algorithm with further training for anomaly detection," Knowledge-Based Systems, 30: 185-191.

Haiyu Hou and G. Dozier, 2006. "An evaluation of negative selection algorithm with constraint-based detectors," Proceedings of the 44th annual Southeast regional conference, pp: 134-139.

Hassanien, A.E., *et al*., 2013. "Computational intelligence techniques in bioinformatics," Computational Biology and Chemistry, 47: 37-47.

Holland, J.H., *et al*., 1987. "Induction: Processes of Inference, Learning, and Discovery," IEEE Expert, 2: 92-93.

Hopkins, M., *et al*., 1999. "Spam Base Dataset," Hewlett-Packard Labs.

Jerne, N.K., "Towards a network theory of the immune system. ," Annales d'immunologie, 125c: 373-389.

Maciej Liśkiewicz and J. Textor, 2010. "Negative selection algorithms without generating detectors," Proceedings of the 12th annual conference on Genetic and evolutionary computation, pp: 1047-1054.

Mohammad, A.H. and R.A. Zitar, 2011. "Application of genetic optimized artificial immune system and neural networks in spam detection," Applied Soft Computing, 11: 3827-3845.

Stibor, T., 2007. "Phase Transition and the Computational Complexity of Generating r-Contiguous Detectors," in Artificial Immune Systems. vol. 4628, L. Castro, et al., Eds., ed: Springer Berlin Heidelberg, pp: 142-155.

Stibor, T., 2008. "Foundations of r-contiguous Matching in Negative Selection for Anomaly Detection," Natural Computing.

Stibor, T., *et al*., 2004. "An Investigation of R-Chunk Detector Generation on Higher Alphabets," in Genetic and Evolutionary Computation – GECCO 2004. vol. 3102, K. Deb, Ed., ed: Springer Berlin Heidelberg, pp: 299-307.

Textor, J., 2012. "A Comparative Study of Negative Selection Based Anomaly Detection in Sequence Data," in Artificial Immune Systems. vol. 7597, C. Coello Coello, et al., Eds., ed: Springer Berlin Heidelberg, pp: 28-41.

Textor, J., 2012. "Efficient Negative Selection Algorithms by Sampling and Approximate Counting," in Parallel Problem Solving from Nature - PPSN XII. vol. 7491, C. C. Coello, et al., Eds., ed: Springer Berlin Heidelberg, pp: 32-41.

Travé-Massuyès, L., 2014. "Bridging control and artificial intelligence theories for diagnosis: A survey," Engineering Applications of Artificial Intelligence, 27: 1-16.

Wang, C. and Y. Zhao, 2008. "A new fault detection method based on artificial immune systems," Asia-Pacific Journal of Chemical Engineering, 3: 706-711.

Wang, D., *et al*., 2013. "Towards enhancing centroid classifier for text classification—A border-instance approach," Neurocomputing, 101: 299-308.

Wierzchoń, S., 2000. "Generating Optimal Repertoire of Antibody Strings in an Artificial Immune System," in Intelligent Information Systems. vol. 4, ed: Physica-Verlag HD, pp: 119-133.

Zhang, Y., *et al*., 2010. "Immunity-based model for malicious code detection," vol. 6215 LNCS, ed. Changsha, pp: 399-406.