First International Congress on Technology, Communication and Knowledge (ICTCK 2014)
November, 26-27, 2014 - Mashhad Branch, Islamic Azad University, Mashhad, Iran

Metamorphic Virus Detection Based on Bayesian Network

NedaShabani
Department of ComputerEngineering
Mashhad Branch,Islamic Azad University
Mashhad, Iran
neda.shabani@yahoo.com

MajidVafaeiJahan
Department of Computer Engineering
Mashhad Branch,Islamic Azad University
Mashhad, Iran
VafaeiJahan@mshdiau.ac.ir

Abstract **- Metamorphic virus detection is one of the most challenging tasks of antivirus software and the most difficult ones are among known viruses. In this article we have used Bayesian network to recognize these kinds of viruses. The body of these virusesis made of assembly codes.At first opcodes are extracted as 1-gram from virus body, theseopcodes are known as the characteristics of Bayesian network, extracting these characteristics reduce dramatically the Computational complexity, memory and time used. After that, it's time to draw the Bayesian network, before drawing,Bayesian network should be training. Bayesian network learning is known as a NP-hard problem because of this utilizing exploratory research has proven that it can be helpful in a lot of cases; in which we have used hill climbing algorithm. This method is compared to different Hidden Markov Model and the methods of role-opcode are also compared. Experimental result shows that, utilizing Bayesian network,the accuracy of virus detection increase, and other classify are not superlative that Bayesian network.**

I. INTRODUCTION

As software are completing, some of developers employ some security related activities to ensure that their products have more security. But viruses are an increasing threat for systems .While technology is developing against viruses; viruses are developing to penetrate in codes. Scanning and methods base on signing are among the first methods developed against viruses. This method were use in the format ofantivirus, but one of the problem of functional programs like antiviruses after performing and appearing in computers . Additionally,since, vulnerability of signing based methods was so high, so this method was not that applicable in next generation ofviruses. In this method was have tried to detection all kinds of destructive methods , viruses and etc. by analyzing files in code level before anything else . There have been a lot of attempt to detection destructive codes via static analyses on files in the binary code level .The idea of using available opcodes was mentioned in assembly code along with analyzing viruses in binary code level . One of the most important activities was done was along with evaluating opcodes and detection computer malware by Billar et al in 2002. They evaluate different kinds of opcodes including more practical and rare sample and they also showed that the abundance of in virus files rather than clean files is more different from more practical opcodes [2]. After that Moskuvi 2002 used n-gram opcode to detection computer malware. They showed that if the percentage of malware software is about 15% of total samples we can reach the accuracy up to 99% [3]. Sentus et al 2010 represented and system to detection the different kind of viruses based on opcodes [4]. He then continued his study about detection malware based on learning methods in one class and also a semisupervision plan according to malware based on opcodes and he also reached to some valuable results [5]. They completed their study about detection malware in 2012. They also represented a method to evaluate and measure the relationship of the scope of each opcode and their work all was based on the scope of appeared opcodes in the next. In this article we have used data exploitation method to detection malware [6].

At the present study we have used exploited opcode from assembly codes the detection metamorphic viruses. These opcodes in an appropriate format are change in the entrance of the software. We have also used normal file to do some compares, such that normal files just like virus files send their opcodes and we exploit them, change them to a good format to detection metamorphic viruses. Following we will explain the process of opcode exploiting and how to format them and we also will explain how to draw a Bayesian network for each viruses and normal file.

II. Metamorphic viruses

These viruses depend on the polymorphic viruses and they change their own destruction in each generation. Assembly codes for each of these containcharacteristics as fallow [13]:

1) Most of the metamorphic viruses put garbage instructions among the core of instructions. Garbage instructionmay not be performed and if so, then they won't have any impact on the result of the program. For example NOP instructions that "sub eax,0" and "add eax,0" are its sample instructions don't affect the results [13].
2) Another way of inserting jump instruction is skip method in which skips from its own code to the next order from virus code [13].
3) Another kind of metamorphic methods is replacement method, in which a group of instructions are replaced by equal replace instructions for example we can replace conditional skip JCC by reverse test condition by JNCC. "Pushebp ,movebp,esp" can be replaced by "push ebp, push esp, pop ebp" .sometimes viruses are performed by opcode instructions and then they change , for example zero in eax fixed variable can change its content to XOR and can be for the result . "Xoreax,eax" can be replaced by "sub eax,eax" [13].
4) Transposition or rearranging is another kind of utilized methods by metamorphic viruses. Instruction is possible when there is no dependency between instructions.

III. Opcode exploition

In order to exploit opcodes,first each file changes to the string of machine instruction or orders by utilizing one disassembler. These orders are made of two parts named operand and opcodes. Fallowing example is a sample of assembly code relating to functional file.

```
pushesi
call SUB_L0100D36F
popecx
testeax,eax
jnz L0100D281
push 0000001Ch
call SUB_L0100D330
popecx
jnz L0100D281
testeax,eax
```

Sinceopcodes can be the results of a result performance of a file so we have skipped analyzing operands and we are just trying to find exploitationopcodes for each file. So the string of exploitationopcode for above example is {push , call, pop, test,jnz,push,call,pop,jnz,test} .After exploiting opcodes from assembly codes, non-irrelativeopcodes are separated from the list and the rest opcodes arrange as the number of non-irrelativeopcodes. These are done as fallow:

```
@attribute push {push,call,pop,test,jnz}
@attribute call {push,call,pop,test,jnz}
@attribute pop {push,call,pop,test,jnz}
@attribute test {push,call,pop,test,jnz}
@attribute jnz {push,call,pop,test,jnz}
@data
Push,call,pop,test,jnz
Push,call,pop,jnz,test
```

Above sample is defined as data set for software. For each of virus and normal files, above stages are done. After doing mentioned stage ,now it's Bayesian network drawing's turn which is related to each file . Next section explains how to draw a Bayesian network.

IV. Bayesian network learning

Bayesian network learning is known as a NP-hard problem, because of this utilizing exploratory research proved that it was helpful in many cases learning approach on accounting point of view is efficient although it doesn't guarantee optimistic result, previous studies have shown that it's a good solution. Hill climbing algorithm is known as a popular algorithm just because of good exchanging between Computational demands and the quality of the model [15].
Generally, there are two main methods for learning Bayesian networks:

1) *Score oriented + research methods*: In this algorithm function F is for network score and DAG (Directed Acyclic Graph) is used considering learning data, and the research method is used for the network by the best score. In this paper we have used local Hill climbing research method of learning for Bayesian network, although other research method are also used, like simulating,Tabu search, Genetic algorithm and ant colony optimization, etc. [15].
2) *Constraint-based methods:* The idea underlying thesemethods is to satisfy as much independence

present in the data as possible. Statistical hypothesis testing is used to determine the validity of conditional independence sentences.

As mentioned above, we focus on local research in DAG space.

A. Learning Bayesian network by local search

The case of learning the structure of Bayesian network can be explained as fallow:
Learning set of D= $\{v^1,\ldots,v^m\}$ and sample V is given , DAG of G^* found such that [15] :

$$G^* = \text{argmax}_{G \square G^n} \; F(G:D) \qquad (1)$$

F(G:D) , is scoring factor to each DAG of G based to data set of D , and if G^n is a sample including all DAG with n node [15].

$$F(G:D) = \sum_{i=1}^{n} F_D(X_i, Pa_G(X_i)) \qquad (2)$$

$$F_D(X_i, Pa_G(X_i)) = F_D(X_i, Pa_G(X_i) : N_{x_i, Pa_G(X_i)}) \quad (3)$$

$N_{x_i, Pa_G(X_i)}$ is static of X_i variable and $Pa_G(X_i)$ in D is the number of the samples in D which can show a incontrast to X_i and $Pa(X_i)$.
Algorithm 1, describe Hill climbing algorithm for learning the structure of Bayesian network. Although DAG (G_0) can be used to search the amount of scoring but the empty graph is used. In this algorithm we assume that in each time a family is scored by $F_D(.)$, then they send it to the hidden memory to gain the amount and they can be a member of that family if they haven't been evaluated before . After accounting the score and using data set of D we inter that into the cache[15].
Hill climbing algorithm to find a model with the highest score is used. Primary state of a Directed Acyclic Graph (DAG) is D empty. The function of finding a collection of neighbors produces D in which each of the neighbors of D are accounted by one of the 3 model of deleting of arc , adding of arc and reversal of arc and each one's functioning is computed as fallow [16]:

1) Addition of $X_j \rightarrow X_i$: $f_D(X_i, Pa(X_i) \cup \{X_j\}) - f_D(X_i, Pa(X_i))$
2) Deletion of $X_j \rightarrow X_i$: $f_D(X_i, Pa(X_i) \setminus \{X_j\}) - f_D(X_i, Pa(X_i))$
3) Reversal of $X_j \rightarrow X_i$: It is obtained as the sequence: deletion($X_j \rightarrow X_i$) plus addition($X_i \rightarrow X_j$), so we compute $[f_D(X_i, Pa(X_i) \setminus \{X_j\}) - f_D(X_i, Pa(X_i))] + [f_D(X_j, Pa(X_j) \cup \{X_i\}) - f_D(X_j, Pa(X_j))]$

Then, in each of stage of algorithm all (local) functioning are analyzed and the highest one is selected.
After generating DAG,then the table of conditional probable for DAG is gained. All h neighbors have a score and the highest score of h' neighbors are compared with h.

- If the score of h' was better than h, then the whole cycle would repeat for h' and so on.
- If it's not so , then the search is stopped [16]

Algorithm 1: *Structural learning of BNs by using a hill climbingalgorithm* [15]

```
Input: D: A dataset defined over variables V={X1, . . . ,Xn}
Input: $G_0$: A DAG defined over V used as the starting point for the search
Output: A DAG G being the graphical part of network B
1)$G \leftarrow G_0$;
2)$f_G \leftarrow f(G : D)$ //*
3)[r] f : decomposable scoring metric   improvement←true;
4)while improvement do
        5)improvement←false; //*
        6)[h]neighbors generated by addition
        7)For each node $X_i$ and each node $X_j \notin Pa_G(X_i)$ such that $X_j \rightarrow X_i$ does not introduce a directed cycle in G, compute the difference
          $diff = f(G + \{X_j \rightarrow X_i\} : D) - f_G$. Store the change which maximizes diff in ($change_a$, $diff_a$);
          //*
        8) [h]neighbors generated by deletion
        9)For each node $X_i$ and each node $X_j \in Pa_G(X_i)$, compute the difference $diff = f(G - \{X_j \rightarrow X_i\} : D) - f_G$. Store the change which maximizes diff in ($change_d$ , $diff_d$);
          //*
        10)[h]neighbors generated by reversal
        11)For each node $X_i$ and each node $X_j \in Pa_G(X_i)$ such that reversing $X_j \rightarrow X_i$ does not introduce a directed cycle in G, compute the difference $diff = d_1 + d_2$ where $d_1$
          corresponds to $f(G - \{Xj \rightarrow Xi\} : D) - f_G$ and $d_2$ corresponds to $f(G + \{Xj \rightarrow Xi\} : D) - f_G$. Store the change which maximizes diff in ( $change_r$ , $diff_r$);
          //*
        12)[h]Checking if improvement
        13)Let $d^* = max_{k=a,d,r} diff_k$ and $move^*$ its corresponding change;
        14) if $d^* > 0$ then
        15)improvement←true;
        16)$G \leftarrow$ apply $move^*$ over G;
        17)$f_G \leftarrow f_G + d^*$;
        18)end
19)end
20)return G;
```

Local search method (especially Hill climbing), are started from a primary method by doing some limited methods and they pass the search distance. In any

stages of algorithm just local changes for example DAG's neighbors and selecting the highest result of F is considered. Algorithm is stopped when there is not local change for optimizing F feedback. In the other hand F is taken in local optimization. Different strategies are used to try to escape from local optimization like, restart, random etc.[15]
Bayesian network learning, local changes are deleted and adding and reversal of the arc are usually done in DAG space. Except for deleting the arc, take care about the acyclic Graph. So, time rating of changes can be O (n^2) in which n shows the number of variables[15].

B. Bayesian network tracing

After consideration needed learning algorithm and suitable data set,now its turn to trace the Bayesian network. As mentioned before, non-irrelativeopcodes are separated from exploited opcodes from assembly code. These opcodes make the nodes for Bayesian network and residual opcodes in the list show the way nodes are connected to each other. In fact non-irrelativeopcodes are the feature of Bayesian network. Feature extraction is one of the important parts in detection, classification,prediction and other way of data mining. Generally the more the numbers of feature , the more the clementine of the space for them and leading to that complexity , the case is increasing , therefore, finding less number of the feature and the most effective one is always one of the most challenging cases in data exploitation [14] . we have tried a lot have to find the most effective feature in viruses and other bad performing files by less number of the features and less Computational burden .
The data may be known but not known network structure .It is the same virus detection; Data or assembly code were identified using the Bayesian network is still not clear. Bayesian network to determine the available data suggest the need for multiple networksBut it is very time consuming. That's why we've used in this article to make hill-climbing algorithm for this network with the highest score achieved and then using (4), we solve the network [17].

$$L_G = (\theta; s_1, s_2, \dots, s_m) = P_G(D|\theta) = \prod_{i=1}^{m} P_G\,(s_i|\theta)(4)$$

Fig.1, is a sample of Bayesian network tracing on virus file by utilizing Hill climbing algorithm. As

NO	push	call	pop	test	jnz
1	push	call	pop	test	jnz
2	push	call	pop	jnz	test

Table I *sample of extracted opcode from the virus body*

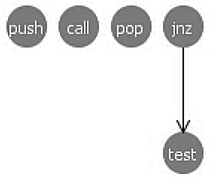

Figure 1. *a sample of traced Bayesian network for virus file utilizing the Hill climbing algorithm*

Shown in fig.1, this virus file includes 5 nodes. On the other way, it has 5 features which are extracted from related assembly code. Table I, shows a sample of extracted opcode from virus assembly code. As mentioned in previous part in order to trace Bayesian network by utilizing Hill climbing algorithm, we need a data set and an empty DAG. Which at first are just nodes. Graph nodes are the same non-irrelativeopcodes extracted from the virus body. In the next step, nodes should be connected to each other. To connect data to each other we use a data set and algorithm 1.
Table I, shows the needed data set to trace the network. As seen in the table I, we have 5 columns which make nodes so our graph has 5 nodes. Look at "push" column , in here push opcode is repeated , so that node should be connected to itself but since Bayesian network is a DAG , we have to use the 8th stage of the algorithm and delete this edge. "Call" and "pop" columns have also the same conditions. So as shown in fig.1, these nodes are traced separated. In the column of test, jnzopcode is seen and in column jnz, test opcode is seen. So these 2 opcode are related to each other. We use the 6th stage of algorithm 1 to connect these two nodes to each other. But since these to opcodes are related to each other we can use reversed edge, to do so, we use the 10th stage of the algorithm.

$$P_G(D|\theta)=\prod_{i=1}^{2} P_G\,(a_i|\theta).\prod_{i=1}^{2} P_G\,(b_i|\theta).\prod_{i=1}^{2} P_G\,(c_i|\theta)\,.\prod_{i=1}^{2} P_G\,(d_i|f_i,\theta)$$

P(push).P(call).P(pop).P(test| jnz) =

P({push,call,pop,test.jnz}|θ)×
P({push,call,pop,jnz,test}|θ)

P(push=push)×P(call=call)×P(pop=pop) ×
P(test=test | jnz=jnz) ×P(push =push) ×
P(call=call)× P(pop=pop) ×P(test =jnz | jnz=test) =

$$\left(\frac{2}{2}\times\frac{2}{2}\times\frac{2}{2}\times\frac{1}{2}\right)\times\left(\frac{2}{2}\times\frac{2}{2}\times\frac{2}{2}\times\frac{1}{2}\right)=0.25$$

Fig.1 is drowning by weka software.

V. Result and Expriments

All results are gained by weka. Used system has 4 gigabyte RAM,5 core processor Intel, and windows 7 64bits. Data set of 350 virus file and 250 normal file are used in this article, that in total they equal to 600 samples. Normal files are test by an antivirus related to Microsoft [11]. The set of viruses are from some reliable sites like virusshare, malwarelu,malshare which received in the first 6 month 0f 2012 [8,9,10].
Fig.2 shows the excitatory results of Bayesian network for normal and virus files. As it's shown in the figure 2 gained probability for normal files and virus files is different and these 2 files are completely separated from each other.

As mentioned in part IV, Bayesian network has different learning algorithm that among them, in this article, Hill climbing algorithm is used. Fig.3 shows the reason of choosing this algorithm. As it seen from the figure this algorithm comparing to others has better accuracy.

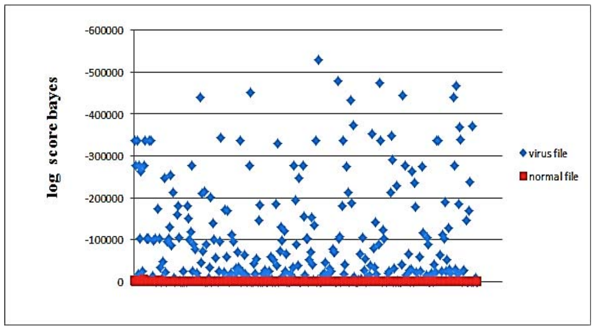

Figure 2. *excitatory results of Bayesian network for normal and virus files*

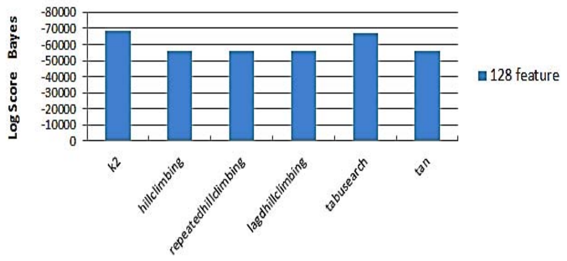

Figure 3. *the chart related to the comparing the accuracy of learning algorithms for Bayesian network*

At first maybe in fig.3, tan algorithm is better than Hill climbing algorithm, but if you pay more attention to fig.4, you will see that the time of making tan algorithm is more than Hill climbing one.
Learning Bayesian network algorithms are compared time build model for construction point of view. As it's shown in the Fig.4, Hill climbing algorithm is better than others, time point of view.

There is a point about fig.3 and Fig.4, K2 algorithm has less time comparing to Hill climbing algorithm, but as seen in fig.3 the accuracy of this algorithm is less than Hill climbing algorithm. Fig.5 shows the accuracy level of different classification for each number of different features.

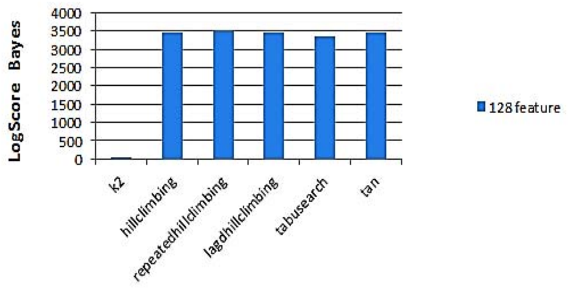

Figure 4. *the chart related to the time build model comparing learning Bayesian network algorithm*

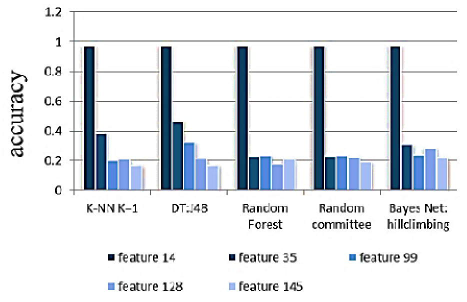

Figure 5. *comparing different classification and Bayesian network*

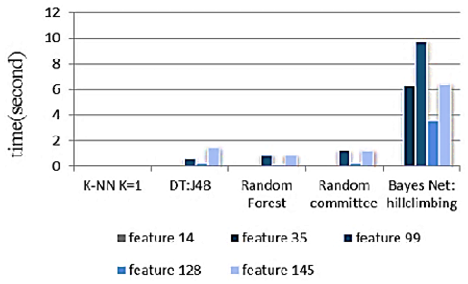

Figure 6. *needed time to build model in different classifiers and Bayesian network*

As shown in fig.5, the accuracy of Bayesian network for feature 145 is in a better situation regarding others. This subject is very important classification in features with high number works well and this is done well just by Bayesian network.
Different classifications are also compared in time to build model point of view. As seen in fig 6, Bayesian network has much time than other classifiers but this duration is to construct the model and by once performing of algorithm we can use a model many times and then this time is neglected.

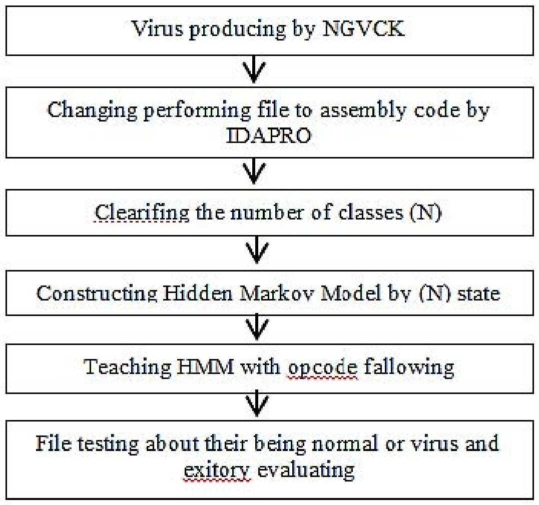
Virus producing by NGVCK
Changing performing file to assembly code by IDAPRO
Clearifing the number of classes (N)
Constructing Hidden Markov Model by (N) state
Teaching HMM with opcode fallowing
File testing about their being normal or virus and exitory evaluating

Figure 7. *The procedure of performing role – opcode*

A.Comparing Bayesian network with Hidden Markov Model

Hidden Markov Model is one of the approach to detection the metamorphic viruses. We compared our method by method in [7]. In [7] method first extractedopcodes classification is done, from files and based on selected classifications, Hidden Markov Model is constructed. The number of different states for HMM is consider based on the number of opcode classifications and just one class of opcodes is shown in each state [7]. Fig.7, shows to procedure of doing this performance.

In this method [7], performing files change to assembly code and the teaching HMM is done by the fallowing made by opcodes [7]. Detection the framework of method [7] is done such this:
All teaching arrival files and passed to make the model as a parameter of HMM, after, teaching HMM to arrival files, excitatory files are used to account log – likelihoods for each opcodes [7].
Experiments are done such that, in which HMM of Viterbi algorithm and back ward algorithm are used to find the best string of the state [7] . The likelihood gained by these two algorithms is shown. Bayesian network is traced like the one described in IV-B part is and its likelihood is gained. We compare these two states; we have tested these likelihoods for 22 extracted features. Excitatory results of these comparisons are shown in fig.8.

These compressions were evaluated for 22 features.

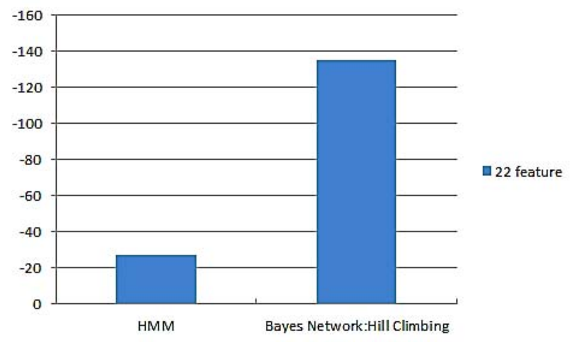

Figure 8. *Comparison of the accuracy of Bayesian network and HMM*

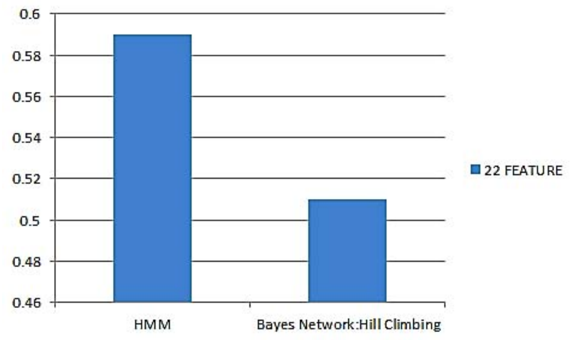

Figure 9. *Time comparing for Bayesian network and HMM model*

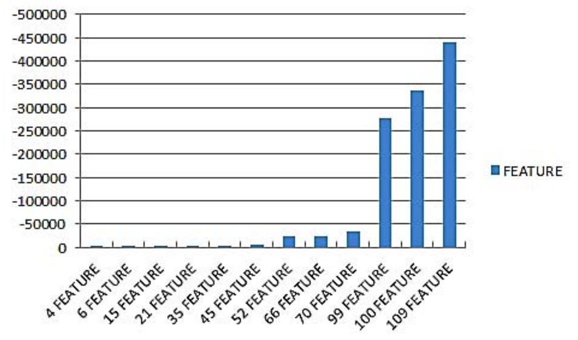

Figure 10. Comparing the accuracy of the features of different Bayesian network

As shown in fig.8 HMM has better accuracy comparing to Bayesian network .The reason HMM uses two algorithms to find the best string in state.
But the time consumed in constructing this model is more than Bayesian network.Fig.9 shows the construction time, HMM and Bayesian model.

B.Compression of the time and accuracy of constructing of model and different features of Bayesian model

As mentioned in section IV-A, the less the number of extracted feature, the more the accuracy. Fig.10 shows the accuracy of Bayesian model for different feature.

As we can see in the picture, the most accuracy is 4 extracted features and the worst is for 109 features which are extracted. And the same case is true for time. Fig.11 shows the constructing time for different features in which the worst time related to 109 features and the best time related to the 4 features.

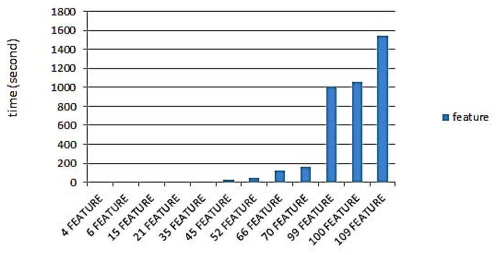

Figure 11. *The chart related to comparing construction time for model and different feature of Bayesian network.*

VI. Conclusion

We have used Bayesian method to detection metamorphic viruses. The procedure was such that first the features were extracted from the virus body as 1-gram. Then by extracting features, Bayesian network is traced. We taught how to trace Bayesian network. That was done by utilizing learning algorithm of Hill climbing. The way of Bayesian network tracing was such that the feature which was extracted as 1-gram made the nodes of Bayesian network. Connecting these nodes to each other was done by utilizing Hill climbing algorithm. Among tests which were done for normal and virus files, Bayesian network could separate these two file from each other. It was also shown that the less the number of features extracted the more accuracy and the less construction time. At the end the Bayesian network was compared to HMM and role – opcode model. The accuracy of HMM model was better than Bayesian network mode in all tests. The reason of this was that HMM used two algorithms to find the best string state of own. But its construction time was not more that Bayesian network. Bayesian network was compared by other classifiers in accuracy and construction time point of view. In which the excitatory of Bayesian network for more features showed better performance comparing to other classifiers .So it can be helpful that utilizing Bayesian network and applying selection methods we can reach to better results in detection malwares.

REFERENCES

[1] Tabish, S. M., Shafiq, M. Z., &Farooq, M. , “Malware detection using statistical analysis of byte-level file content”,ACM SIGKDD Workshop on CyberSecurity and Intelligence Informatics, pp. 23-31. New York, NY, USA,2009

[2] Bilar, D. , “OpCodes as predictor for malware”, International Journal of International Journal of, 1(2), 156-168,2007

[3] Moskovitch, R., Feher, C., Tzachar, N., Berger, E., Gitelman, M., Dolev, S., &Elovici, Y. “Unknown Malcode Detection Using OPCODE Representation”, 1st European Conference on Intelligence and Security Informatics, pp. 204-215,2008

[4] Santos, I., Brezo, F., Nieves, J., Penya, Y. K., &Sanz, B, Idea:” Opcode-sequence-based Malware Detection”, Engineering Secure Software and System,2010

[5] Bringas, I. S. ,” Using opcode sequences in single-class learning to detect unknown malware”, IET Information Security,2011

[6] Santos, I., Brezo, F., Ugarte-Pedrero, X., &Bringas, P. G. ,” Opcode sequences as representation of executables for data-mining-based unknown malware detection”, Information Sciences, 231, 64-82,2013

[7] YassinDehghannaeri,”Detection metamorphic virus by Hidden Markov Model”, MS Thesis, Department of Islamic Azad University of Mashhad, Iran,2013

[8] Roberts, J.-M,VirusShare, Retrieved from http://virusshare.com/, 2013, 4 24

[9] Rascagneres, P, malware.lu, Retrieved from http://www.malware.lu/pages/company.html , 2013

[10] Wood,D.,VirusSign. Retrieved fromhttp://www.virussign.com/downloads.htmlx86 instruction listings, Junuary 2014

[11] kaspersky lab, Retrieved from http://www.kaspersky.com/pure,2013

[12] Remco R. Bouckaert , “Bayesian Network Classifiers in Weka for Version 3-5-7”, University of Waikato, May 12, 2008

[13] W. Wong, “Analysis and detection of metamorphic computer viruses", Master's thesis, San Jose State University, 2006.

[14] Z. Ghezelbiglo, M. VafaeiJahan, “Role-Opcode Vs. Opcode in Maleware Detection” The 11th conference on Computer and Intelligence Systems, Kish, Iran, 2014

[15] José A. Gámez · Juan L. Mateo · José M. Puerta,”Learning Bayesian networks by hill climbing efficient methods based on progressive restriction of the neighborhood”, Data Min Knowl Disc (2011) 22:106–148, 28 April 2010

[16] AntoninoFreno,” Bayesian Networks”,antonino.freno@inria.fr

[17] MajidVafaeiJahan,”an introduction to computer modeling and simulation”, khatavalpuplication,second edition ,2012