Evolving Hidden Markov Models For Network Anomaly Detection

Juan J. Flores, Anastacio Antolino, and Juan M. Garcia

Division de Estudios de Posgrado
Facultad de Ingenieria Electrica
Universidad Michoacana, Mexico
juanf@umich.mx,
{antolino,jmgarcia}@faraday.fie.umich.mx

Abstract. This paper reports the results of a system that performs network anomaly detection through the use of Hidden Markov Models (HMMs). The HMMs used to detect anomalies are designed and trained using Genetic Algorithms (GAs). The use of GAs helps automating the use of HMMs, by liberating users from the need of statistical knowledge, assumed by software that trains HMMs from data. The number of states, connections and weights, and probability distributions of states are determined by the GA. Results are compared to those obtained with the Baum-Welch algorithm, proving that in all cases that we tested GA outperforms Baum-Welch. The best of the evolved HMMs was used to perform anomaly detection in network traffic activity with real data.

Key words: HMMs, GAs, Anomaly Detection, Baum-Welch

1 Introduction

Security threats for computer systems have increased immensely which include virus, denial of service, vulnerability break-in, etc. While many security mechanisms have been introduced to undermine those threats, none can completely prevent all of them. Security threats to the computer systems have raised the importance of anomaly detection [1] .

Anomaly detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of anomaly. Anomaly is defined as attempts to compromise the confidentiality, integrity, availability, or to bypass security mechanism of a computer or network.

Hidden Markov Models (HMMs) are considered a basic component in speech recognition systems. Besides, HMMs are based on a probabilistic finite state machine used to model stochastic sequences [2]. HMMs have many applications in signal processing, pattern recognition, speech and gesture recognition, as well as applications to anomaly detection.

It is important to mention that in an HMM, the estimation of good model parameters affects the performance of the recognition [3] or detection processes. The HMM parameters can be determined during an iterative process called "training". The Baum-Welch algorithm [4] is one method applied in setting an HMM's parameters, but this method has a drawback. The Baum-Welch algorithm, being a gradient-based method, may converge to a local optimum.

Global search techniques can be used to optimize an HMM's parameters. Genetic Algorithms (GAs) are a global optimization technique that can be used to optimize an HMM parameters [3].

GAs are a searching process based on the laws of natural selection and genetics. It emulates the individuals in a natural environment that the natural selection mechanism makes the stronger individuals likely winners in the competing environment. Studies of using GAs to train HMMs are relatively rare, particularly in comparison with the large literature on applying GAs to neural network training [2].

In this paper we present the use of GAs for creation and optimization of an HMM on one step, from a time series data. And then this model is used for anomaly detection for network traffic. And finally, we compare the performance of HMMs created with GA against other created with the Baum-Welch algorithm.

The remainder of the paper is structured as follows. Section 2 introduces the related HMM theory. Section 3 describes GAs for readers not familiar with the topic. Section 4 describes Intrusion Detection theory. Section 5 is the base of the proposal of this paper. Section 6 illustrate the results on comparison HMMs created with GAs against other created with the Baum-Welch algorithm. Section 7 is a brief overview of some approaches and uses of HMMs for anomaly detection and GAs used for optimizing an HMM´s parameters. Section 8 presents our conclusions and future work.

2 Hidden Markov Models

An HMM is a doubly stochastic process with an underlying stochastic process that is not observable, but can only be observed through another set of stochastic processes that produce the sequence of observed symbols. The most common application of HMMs, which are best known for their contribution, is in automatic speech recognition, where HMMs were used to characterize the statistical properties of a signal [5], [4].

Also, HMMs have been working in the bioinformatic area. Recently, HMMs have been applied to a variety of applications outside of speech recognition, such as handwriting recognition, pattern recognition in molecular biology, and fault detection. Variants and extentions of HMMs include econometrics, time series, and signal processing [6].

An HMM is formed by a finite number of states connected by transitions (see Fig. 1), which can generate an observation sequence depending on its transition, and initial probabilities. That means an HMM is represented by a set of three sets of probabilities (initial state probability, transition between states probability, and the probability to observe a symbol in each state). The Markov Model is Hidden because we do not know which state led to each observation.

An HMM is defined, among others, basically for these three parameters:

A = $\{a_{ij}\}$ is the state transition probability matrix.

B = $\{b_j(k)\}$ is the emission probability matrix, indicating the probability of a specified symbol being emitted given that the system is in a particular state.

$\Pi = \{\pi_i\}$ is the initial state probability distribution.

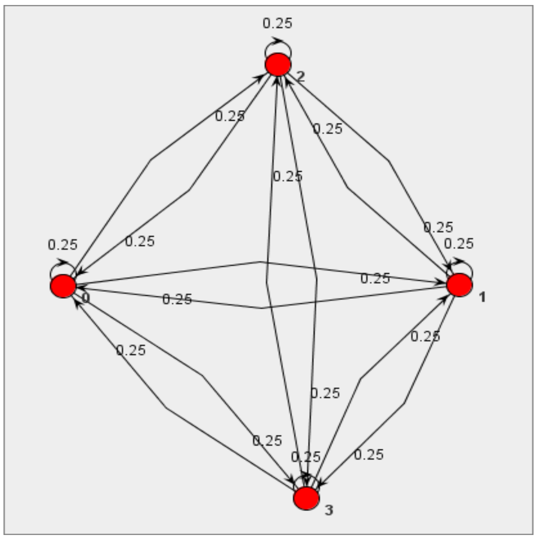


Fig. 1. An HMM example with 4 states.

An HMM can therefore be succinctly defined by the triplet [7]:

$$\lambda = \{A, B, \pi\} \tag{1}$$

There are three basic problems of interest that must be solved for the HMMs to be useful in real-world applications. Mainly HMM applications are reduced for solving three kinds of problems:

1. The probability of a given observation sequence has been generated by λ.
2. The highest-probability state sequence for a given observation sequence and a λ.
3. The adjust or re-estimation of model parameters λ.

HMMs deal with these problems, described above, under a probabilistic or statistical framework. HMMs offer the advantages of having strong and powerful statistical foundations and being computationally efficient to develop and evaluate due to the existence of established training algorithms. But one disadvantage of using HMM is the need for an a priori notion of the model topology. That is, in order to build an HMM we must first decide how many states the model will contain and what transitions between states will be allowed. Once a model structure has been designed, the transition and emission parameters need to be estimated from training data.

The Baum-Welch algorithm is used to train model parameters. The Baum-Welch algorithm is an iterative expectation-maximization algorithm that given an initial parameter configuration, adjusts model parameters to locally maximize the likelihood of data. Baum-Welch training suffers from the fact that it finds local maxima, and is thus sensitive to initial parameters settings. For further information on HMMs, please see [4].

3 Genetic Algorithms

HMMs parameters are determined during an iterative process called "training". One of the conventional methods applied in HMM model parameter values is the Baum-Welch algorithm [4]. One drawback of this method is that it converges to a local optimum. Global search techniques can be used to optimize an HMM parameters. Genetic Algorithm (GA) is a global optimization technique [8], [3] that can be used to optimize the HMM parameters [3].

GAs simulate evolution phenomena of nature. The search space is mapped into a genetic space. The possible solution is encoded into a vector (chromosome), and each element of the vector is called gene. Through continually calculating the fitness of each chromosome, the best chromosome is selected and the optimization solution is obtained [9].

The GA is a robust general purpose optimization technique, which evolves a population of solutions. GA is a search technique that has a representation of the problem sates and also has a set of operations to move through the search space [3].

Selection, crossover, mutation are three main operators of GA, and individual is the objection of operation. These constitute the whole genetic process, making GAs have good characters which other classic methods do not have. You can find the general steps of a GA in [9].

4 Anomaly Detection

Anomaly detection is performed by detecting changes in the patterns of utilization or behavior of the system. Anomaly detection may be performed by building a statistical model that contains metrics derived from system operation and flagging as intrusive any observed metrics that have a significant statistical deviation from the model [1]. In other words, an anomaly detection uses a model of "normal" network behavior to compare to currently observed network behavior. Any behavior that varies from this model is considered a network anomaly and behavior closely matching the model is "normal". In general, the normal behavior of a computing system can be characterized by observing its properties over time.

The problem of detecting anomalies can be viewed as filtering non-permitted deviations of the characteristic properties in the monitored network systems [10].

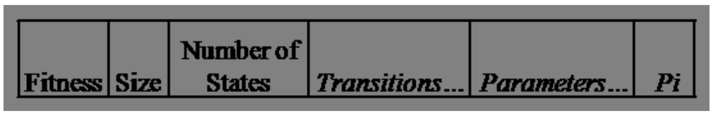
Anomaly detection can also be defined as the process of identifying malicious behavior that targets a network and its resources [11].

Anomaly detection research is carried out in several application domains, such as monitoring business news, epidemic or bioterrorism detection, intrusion detection, hardware fault detection, network alarm monitoring, and fraud detection. The anomaly detection problem involves large volumes of time series data, which has a significant number of entities and activities. The main goal of anomaly detection is to identify as many interesting and rare events as possible with minimum delay and the minimum number of false alarms [12].

5 A Framework for Evolving an HMM

In this section we propose a framework for evolving an HMM parameters. The evolution process consists of:

- A population of chromosomes. One chromosome is constituted as shown in Fig. 2. These genes are described as:
  1. Fitness. This is its fitness evaluation, after been evaluated by the objective function (see Equation 2).
  2. Size. Number of states of the HMM.
  3. Number of States. This is a list of number of elements.
  4. Transitions. This gene has the probability transitions among the states of the HMM (Fig. 3).
  5. Parameters. This gene has the Mean and Variance of a Gaussian distribution function of each state in the HMM (Fig. 4).
  6. Pi. The last set of genes represents the probability of state S$_i$ to start at time t = 1 (Fig. 5).

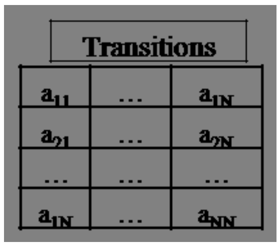


Fitness	Size	Number of States	*Transitions...*	*Parameters...*	*Pi*

Fig. 2. The representation of a Chromosome for evolving HMMs.

Where the chromosomes are the encoded form of the potential solution. Initially, the population is generated randomly and the fitness values of all chromosomes are evaluated by calculating the objective function (Equation 2). This is the Forward variable α that calculates $P(O|\lambda)$ [4], probability that the observation sequence had been generated by model λ (equation 1).

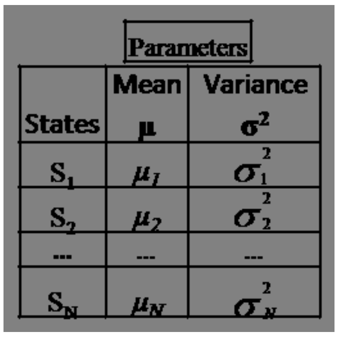
$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i), \tag{2}$$



Transitions

a_{11}	...	a_{1N}
a_{21}	...	a_{2N}
...	...	...
a_{1N}	...	a_{NN}

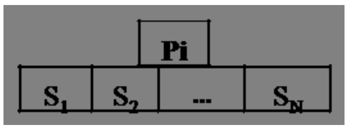
Fig. 3. Representation of a Transitions Gene of the HMM's Chromosome.



Parameters

States	Mean μ	Variance σ^2
S_1	μ_1	σ_1^2
S_2	μ_2	σ_2^2
---	---	---
S_N	μ_N	σ_N^2

Fig. 4. A probabilities Gene of an HMM's Chromosome.



Pi

S_1	S_2	---	S_N

Fig. 5. A Pi Gene of an HMM's Chromosome.

- After the initialization of the population pool, the GA evolution cycle is begun. At the beginning of each generation, the mating pool is formed by selecting some chromosomes from the population. This pool of chromosomes, is used as the parents for the genetic operations to generate the offspring. The fitness values of offspring are also evaluated. Each offspring is a complete HMM, so the system calculates how good this model is with respect to the observation sequence, so we compute its probability and assign this value as its fitness. That is, the topology with the maximum value of likelihood is considered as the best topology.
- At the end of the generation, some chromosomes will be replaced by the offspring, maintaining a total population of constant size across generations, preserving offspring and parents with the best fitnesses.
- This GA is setup to apply some evolutive operators like mutParams (makes a mutation of parameters values mean and variance), mutTransition (mutation or changes of probability values on transition links among states), delTransition (deletes a state transition), addTransition (adds a new state transition), Copy (copies a complete chromosome or HMM), addNode (adds a new node to the model), delNode (deletes a node from the model). This GA builds HMM models ranging from 3 to 15 states. The Mean and Variance are randomly initialized and evolution is driven by mutation magnitude parameters.
- The above generation and genetic operations are repeated until the cycle gets the maximum number of generations. By emulating the natural selection and genetic operations, this process will hopefully determine the best chromosomes or the highly optimized solutions to the problem.

Once GA has finished to evolving an HMM, we take the best model of the last generation and used it to determine if a given observation sequence has anomalies or not. Our new statistic model can examine new observation sequences and determine if a time series of data belongs to this model. That is, if the observation are likely to have been produced by the model. If the observations have a low probability of belonging to the model, it is possible that the sequences present an anomaly.

As we can see in Fig. 6, we have an HMM evolved with GAs. This HMM was evolved based on the observation sequence given by the bandwith used.

The HMM generated and optimized by GAs, is a great help to detect statistically anomalous behavior; its function is to detect anomalous behavior and discriminate over a normal one. HMMs focus on statistics-based anomaly detection techniques. We built a statistics-based normal profile and employed statistical tests to determine whether observed activities deviate significantly from the normal model.

The models generated by GAs analyse incoming time series data, on a sliding time window, and they can detect anomalous behavior or not. The way they analyse an observation sequence is based on deviation from average behavior.

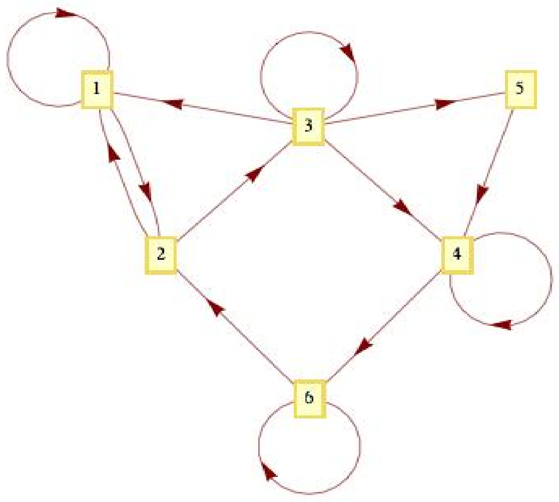


Fig. 6. An HMM Evolved by GAs.

6 Results

This section presents the results of our proposal and compares them with those obtained using the Baum-Welch learning algorithm. We also present the results of using HMMs in anomaly detection.

6.1 HMMs Produced by GAs

Our system evolves HMMs using GAs, with different number of generations and different population sizes. Table 1 describes several examples of HMMs evolved with GAs. The second column of Table 1, shows each model's probabilities for the observation sequence. We selected our HMM from these evolved models. We took the model with the highest probability, and used it in our anomaly detection tests.

Fig. 7 shows how GAs evolve HMMs; each subfigure shows the best individual of corresponding generations. a) Generation 1, produced an HMM with 14 states. b) generation 5, has evolved to 13 states, c) generation 10, HMM has 12 states, d) generation 11, evolved to 11 states, e) in generation 85, has an HMM with 10 states, and f) presents the final HMM evolved with states and transitions. In the last generations, GA has already come to an optimum number of states and focuses on refining the transition probabilities.

Regarding our results, we observed the following points:

Table 1. Several HMM Models Evolved by GAs.

Number of States	$P(O \mid \lambda)$	Population	Number of Generations
5	7.70688×10^{-103}	500	500
5	7.70688×10^{-105}	500	250
6	2.52091×10^{-75}	250	250
6	7.19032×10^{-98}	200	50
8	1.19399×10^{-88}	1000	500

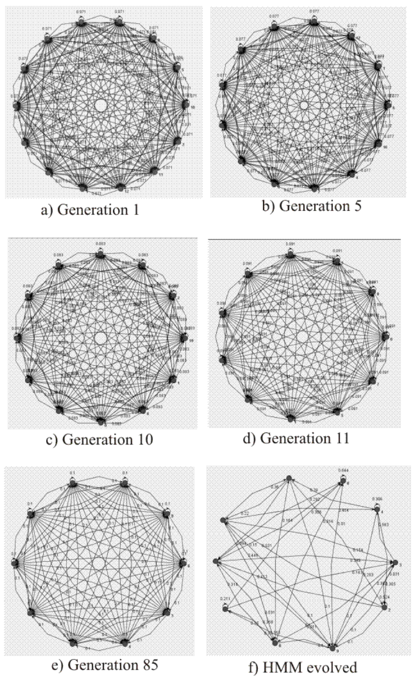


Fig. 7. Evolving an HMM with GAs.

- GA does not assume the user to possess any background knowledge about statistics and probabilities. GA does everything, creates and optimizes an HMM based on the time series data and a probabilistic function to qualify that sequence.
- All this method needs is a time series data, a fitness function, and time for processing the learning method.
- GA builds an HMM with all the required parameters.
- The Baum-Welch algorithm is a good method for learning using time series data, but its problem is that it converges to a local optimum.
- Another problem that the Baum-Welch algorithm presents, is that users need a previous background or knowledge about HMMs theory and how to build it.
- The Baum-Welch algorithm's performance depends a lot on good HMM parameters to re-estimate them.That is, if we give it bad parameters, the method may diverge.

6.2 Baum-Welch Learning

In Fig. 8 we show an HMM structure created with random values. Then we took this HMM and we training it with the Baum-Welch algorithm and we obtained a reestimated HMM. This HMM was trained also, based on the observation sequence given by the bandwith used. A HMM is obtained with the Baum-Welch algorithm training in 50 iterations, as you can see in Fig. 9.

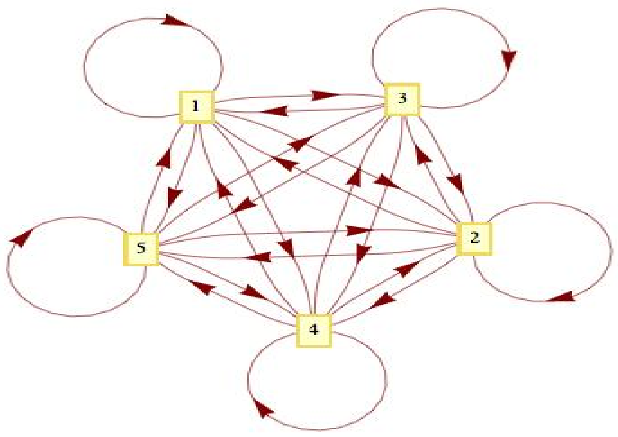


Fig. 8. HMM created with Random Values, before Trainning.

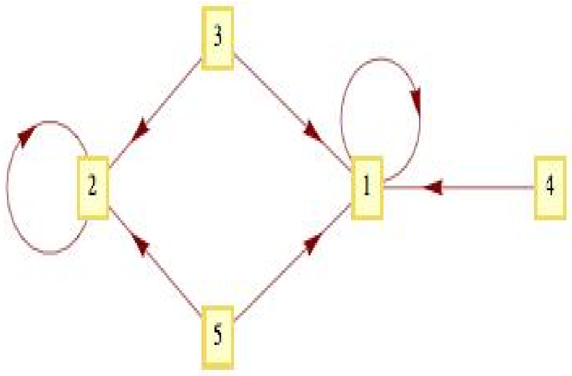


Fig. 9. HMM trained by Baum-Welch Algorithm.

Table 2 describes our tests and results. We created several HMMs and reestimated their parameters using the Baum-Wech algorithm, trying to get better probabilities for our models. We started with a 3-state HMM as you can see in the first column of Table 2. The second column, labeled P(O | λ_1), shows the probability for each model, then we use the Baum-Welch algorithm and we obtained a reestimated model, with a new probability, shown in the third column, labeled as P(O | λ_2). We generated models from 3 to 10 states. In all tests we performed, we applied the Baum-Welch algorithm for 50 iterations. Our initial models were built with random values as initial parameters.

These HMMs present low probabilities because they try to represent the complete likelihood of all observation sequence. It is the probability that this sequence was given o generated by these HMMs.

Table 2. Several HMM Models trained with the Baum-Welch Algorithm.

Number of States	P(O | λ_1)	P(O | λ_2)
3	3.39890 x 10^{-130}	1.8920 x 10^{-123}
4	3.69336 x 10^{-145}	6.44442 x 10^{-175}
5	4.00768 x 10^{-136}	3.43836 x 10^{-137}
6	1.05953 x 10^{-127}	1.07894 x 10^{-127}
7	9.94175 x 10^{-125}	5.90112 x 10^{-125}
8	8.37583 x 10^{-122}	1.09739 x 10^{-119}
9	8.48175 x 10^{-123}	1.22676 x 10^{-119}
10	7.557 x 10^{-116}	6.55885 x 10^{-172}

From our experiments, we realize that the Baum-Welch algorithm needs for its proper usage:

– An initial number of states.

– An initial start point (an estimate for transition and starting point probabilities).
– A Probability Distribution function for each state.

As a conclusion, we noted that the models that we created and we reestimated with the Baum-Welch algorithm did not perform as well as any of the HMM models created with GA, the probability of the best model is 2.5209 x 10^{-75} with 6 states. The Baum-Wech algorithm needs the expertise of the users to create an initial HMM.

We were using Jahmm [13] as our primary tool, but we saw that this tool had problems calculating probabilities on continuous sequences. So we implemented the Baum-Welch algorithms in a Mathematica [14].

Table 3 shows the results from an HMM evolved through GA used in anomaly detection. Our experiments' setup is the following: given an observation sequence, the evolved HMM analyzes a time window of data and it determines the probability of that sequence having been generated by the model. Our time window moves to the next item in the time series, which is verified also by the HMM. And so on, until the whole test time series is traversed and tested. The window sizes we tested were 3, 4, 5. All these probabilities, generated by the sliding time window and the HMM, were compared with the probability of the original sequence and then analyzed to determine if an anomaly exists or not. Finally, we calculated the ratio between the original and the possibly anomalous time series for each window. We do not report all that information for the lack of space. Instead, we report the percentages of hits, false alarms, and unrecognized anomalies. Table 3 shows these results.

The method we are using to analyze and to determine a possible anomaly detection is comparing the probability generated by a time window with a current time serie, against probability given by a time window trained before with our HMM. If these probabilities are different, we mean, current time windows probability are lower, below of our threshold, we flagged as an anomaly behavior.

Table 3. Results of Anomaly Detection using an HMM.

Window Size	%		
	Hits	False Positives	False Negatives
3	71	2	27
4	77	2	21
5	78	2	20

Our experiments report the best result with a window size of 5, with a 78% of positive anomaly detection, 20% of false negative detection, and 2% of False Positive detection. The worst result was produced for window size of 3, with 71% positive anomaly detection, 27% of false negative detection, and 2% of False Positive detection.

In all our experiments we used a time series produced by network bandwidth usage (Fig. 10) in our University. These values recorded bandwidth used for our department. HMMs training used this serie with 48 values, which corresponds to bandwidth used in kbit/second in two consecutive days. The values status fluctuates between 14.625 and 5964.81 kbit/second, considered as a normal behavior.

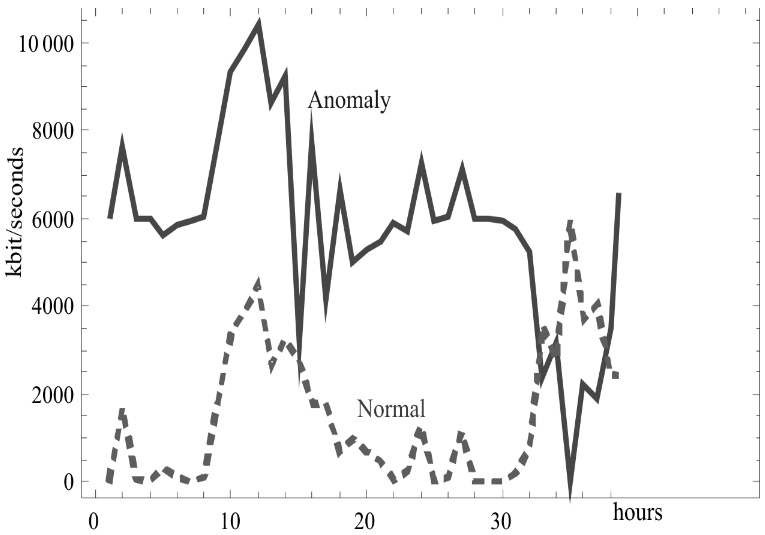


Fig. 10. Network Bandwidth with normal and anomalous behavior.

7 Related Work

Determining the best parameters of an HMM is a hard optimization problem. The performance of the HMM depends largely on these parameters. There are research works in this area, applying GAs as a search method to find the best parameters of an HMM; in this section we mention some of the related work.

We are going to mention some works in GAs aimed to HMM parameter optimization. Most of these papers are focused to optimize or select a good model to use it in speech or gesture recognition.

7.1 HMM Selection with GA

In Bhuriyakorn et al. [8] show several approaches that are extensively used for HMM topology generation. One algorithm predefines rigidly the connectivity among the states; hence, this algorithm only estimates the numbers of states and Gaussian mixtures. Another algorithm, Successive State Splitting, begins allowing one state transition, and grows from one state to the optimal topology. That is, it begins from one state to a complete HMM topology. Another algorithm is also used, State Reduction starts with a fully-connected topology with a predefined maximum number of states, and then, it iteratively reduces a transition until the process terminates. They also describe an algorithm that uses multiple paths in a single HMM, the idea of this algorithm is to improve topologies instead of using significant efforts to adjust the single path for better topologies. Their work focuses basically on the topology selection problem. It is important to mention that these algorithms focus on the task of Thai phoneme recognition only.

Ogawa et al. [15], used a well-known simple GA as a new structure optimization method in which the dependence of the states and the observation of Partly Hidden Markov Model (PHMM) are optimally defined according to each model using the weighted likelihood-ratio maximization.

Won et al. [2] present the use of GAs for evolving HMMs in the prediction of secondary structure information for protein sequences. This hybrid GA was run on a cluster of computers as a Parallel Genetic Algorithm. This paper also mentions that the topology of the HMM was restricted to biologically meaningful building blocks.

Yang et al. [16] treat the problem of optimizing HMMs' parameters. First, they use a GA in order to get into short space solution, where Tabu Search (TS) is applied as a short term memory, recording in a tabu list genetic operators used, and offspring generated. So the GATS (GA and TS combined) algorithm guarantees that GA will not fall in a premature convergence, due to genetic operator used and offspring generated are recorded in a tabu list and can be rejected with a probability in future. So, to improve convergence speed of the GATS-based HMM optimization algorithm, the Baum-Welch algorithm (BW) is combined with it. Generating the hybrid GATSBW algorithm, which combines these three elements mentioned above, with the proposes to train HMMs in continuous speech recognition. This GATSBW algorithm overcomes the shortcoming of the slow convergence speed of the GATS algorithm and also helps the Baum-Welch algorithm escape from local optimum.

7.2 HMMs Used in Anomaly Detection

In this subsection, we provide a brief overview of some research work about anomaly detection using HMM, and see how these systems generally work.

Warrender et al. [17] present a method comparisons used in intrusion detection systems. Among these methods analyzed by Warrender and her team, HMM was analyzed and compared to determine how good was this method used in anomaly detection. Their conclusion were that the performance of HMM depends on its initial parameters, sometimes depends on the experience of who creates the HMM. It was hard and took long time to train an HMM, when this HMM had several states and long observation sequences. But they found that an HMM, once was trained, was fast to determine the probability of an observation sequence. They used programs data and call system, as observation sequences.

Jha et al.[18] present a statistical anomaly detection algorithm based on Markov Chains. The algorithm they propose can be directly applied for intrusion detection by discovering anomalous activities. This algorithm uses the sequence of system calls corresponding to a process as the trace of its activity. The algorithm uses a set of normal system call traces and a statistical model is constructed, then it is used to construct a classifier capable of discriminating between normal and abnormal behavior.

Garca et al. [19] present an approach to anomaly detection based on the construction, hand-designed, of an HMM, trained on processor workload data. For that, this HMM was used to observe a sequence of processor load measurements, where if the probability of the observation of having been generated by the model is lower than it was estimated with the HMM, it was considered as an anomaly.

Khanna et al. [20], discuss an HMM strategy for intrusion detection using a multivariate Gaussian model for observations that are then used to predict an attack that exists in the form of a hidden state. They focus in anomaly detection using users and trends profiles. And any behavior out of this profile is considered as unusual activity, which is recorded as an observation. Once it has these observations, an HMM is used to predict a possible intrusion, based on its hidden states transitions for the most probable intrusion state sequence. These authors are working on ad hoc architecture, so they need a sensor data which one collect and analyze data. This HMM based approach (where the HMM is designed by hand) correlates the system observations (usage and activity profile) and state transitions to predict the most probable intrusion state sequence.

Singh et al. [12] illustrate the capabilities of HMMs, combined with feature-aided tracking, for the detection of asymmetric threats (it refers to tactics employed by subjects to carry out attacks on a superior opponent, while trying to avoid direct confrontation). This approach combines a transactional-based probabilistic model with HMMs and feature-aided tracking.

Most of GAs' work focus on just determining the best HMM's parameters, other are focusing on, once obtained the best parameters, applying HMMs to phonetic or protein sequence research. And most of HMMs used for anomaly detection, described above, were hand-designed. Hand-design implies implicit knowledge derived from previous probabilistic experience.

This work has combined two techniques: GAs for optimization HMM's parameters, and HMMs used for Anomaly Detection. Once an HMM has evolved from GAs, we applied it for Anomaly Detection. This approach allows a user to deploy the system without assuming any background on probability and HMMs theory.

8 Conclusions and Future Work

In this paper, we outlined a framework for a evolving HMMs used for anomaly detection. These models generated and optimized by GAs, provide detection capabilities for anomalous behavior. They are capable of distinguishing between normal and abnormal behaviors. Our experiments show that Hidden Markov Models do well, 75%, over anomaly detection.

Forthcoming experiments include using parallel work on the Baum-Welch algorithm. We hope in a near future, to compare our method with some others, like the one used by [17] and determine how good is our system against others. Finally, we also plan to model several variables to model an HMM.

Our work on HMMs is centered around the task of anomaly detection in a network bandwidth usage. The models we described in this paper are built from time series data of bandwidth used, and the models can be employed by our computer center in a near future.

We expect these improvement to contribute to the development of more accurate models for anomaly detection using HMMs. The next steps are to introduce time in the HMM models, and generalize them to include observations from more than one variable.

Nowadays, completely protect a network from attacks is a very hard task; even heavily protected networks are sometimes penetrated.

References

1. Islam M. H. and Jamil M. Taxonomy of statistical based anomaly detection techniques for intrusion detection. pages 270–276, 2005.
2. Won Kyoung-Jae, Hamelryck Thomas, Prugel-Bennett Adam, and Krogh Anders. Evolving hidden markov models for protein secondary structure prediction. *Evolutionary Computation, IEEE*, 1:1–18, September 2005.
3. Korayem Mohamed, Badr Amr, and Farag Ibrahim. Optimizing hidden markov models using genetic algorithms and artificial immune systems. *Computing and Information Systems*, 11(2), 2007.
4. Rabiner Lawrence R. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2), February 1989.
5. Rabiner L. R. and Juang B. H. An introduction to hidden markov models. *IEEE ASSP Magazine*, 1986.
6. Bengio Yoshua. Markovian models for sequential markovian models for sequencial data. *Statistical Science*, 1997.
7. Nicholl P., Amira A., Bouchaffra D., and Perrot R. H. A statistical multiresolution approach for face recognition using structural hidden markov models. *EURASIP Journal on Advances in Signal Processing, ACM*, 2008:13, 2008.
8. Bhuriyakorn Pattana, Punyabukkana Proadpran, Bhuriyakorn, and Suchato Atiwong. A genetic algorithm-aided hidden markov model topology estimation for phoneme recognition of thai continuous speech. In *Ninth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, pages 475–480. IEEE, 2008.
9. Zhang Xueying, Wang Yiping, and Zhao Zhefeng. A hybrid speech recognition training method for hmm based on genetic algorithm and baum welch algorithm. *IEEE*, pages 572 – 572, 2007.
10. Dasgupta Dipankar and Brian Hal. Mobile security agents for network traffic analysis. *IEEE Transactions on Power Systems*, 2(332-340), 2001.
11. Jemili Farah, Zaghdoud M., and Ben Ahmed M. A framework for an adaptive intrusion detection system using bayesian network. *Intelligence and Security Informatics, IEEE*, 2007.

12. Singh Satnam, Donat William, Pattipati Krishna, and Willet Peter. Anomaly detection via feature-aided tracking and hidden markov model. *Aerospace Conference, IEEE*, pages 1–18, March 2007.
13. Francois Jean-Marc. http://www.run.montefiore.ulg.ac.be/ francois/software/jahmm/, Octuber 2009.
14. Wolfram. http://www.wolfram.com/products/mathematica/index.html, Octuber 2009.
15. Ogawa Tetsuji and Kobayashi Tetsunori. Genetic algorithm based optimization of partly-hidden markov model structure using discriminative criterion. *IEICE TRANS. INF. & SYS.*, E89-D(3), March 2006.
16. Yang Fengqin, Zhang Changhai, and Bai Ge. A novel genetic algorithm on tabu search for hmm optimization. *Fourth International Conference on Natural Computation, IEEE*, 2008.
17. Warrender Christina, Forrest Stephanie, and Pearlmutter Barak. Detecting intrusions using system calls: Alternative data models. In *Security and Privacy, 1999. Proceedings of the 1999 IEEE Symposium on*, pages 133–145. IEEE, May 1999.
18. Jha S., Tan K., and Maxion R. A. Markov chains, classifiers, and intrusion detection. *Computer Security Foundations Workshop*, pages 206–219, 2001.
19. Garcia Juan Manuel, Navarrete Tomas, and Corona Orozco Carlos. Workload hidden markov model for anomaly detection. *SECRYPT 2006*, Proceedings of the International Conference on Security and Cryptography:56–59, August 2006.
20. Khanna Rahul and Liu Huaping. System approach to intrusion detection using hidden markov model. *IWCMC'06, ACM*, July 2006.