Hidden Markov Models For Cyber Security Analysis

Course Project for EE336 Course

VIGHNESH DESHPANDE

200102112

SAMARTH HEGDE

200102081

DEVANSH SHARMA

200102026

The main idea

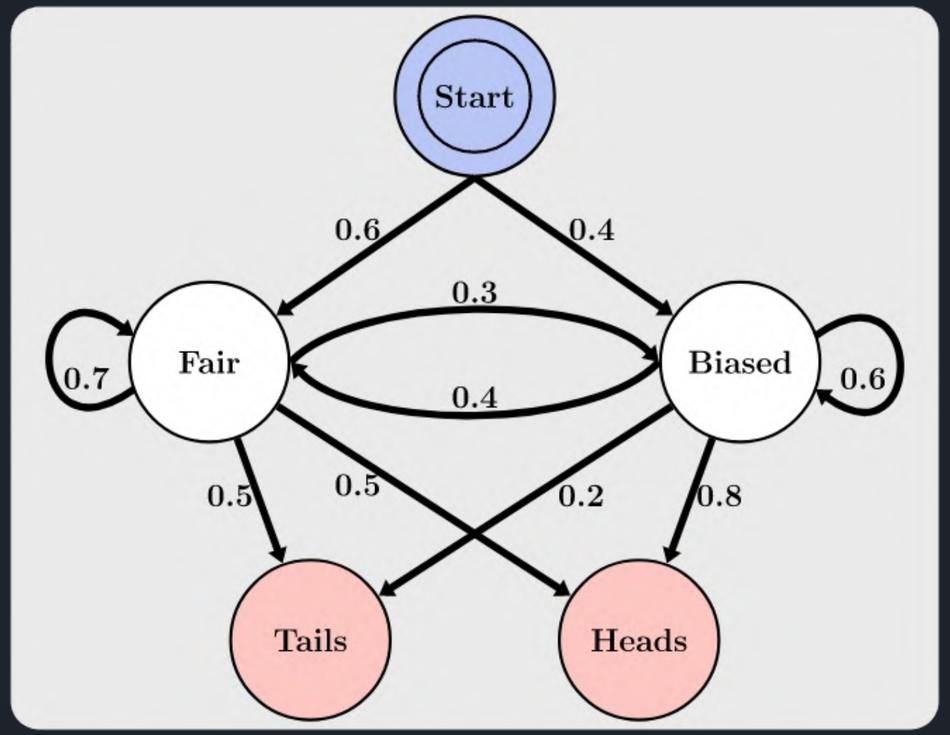
THE CORE IDEA

The main idea behind the project is to predict weather a system is under attack from hackers. Whenever a hacker tries to barge into a system he does so in a series of steps. In each of these steps he leaves behind many alerts which can be used to detect the attack. However, we can't use these alerts directly as they can be caused by a normal user. We need to infer from a sequence of such alerts.

WHY HIDDEN MARKOV MODEL

HMM is a variant of a Markov chain where the states cannot be directly observed but are associated with observable outputs or observations. The model assumes that the observed sequence is generated by a process with an underlying Markov chain that transitions between hidden states, and at each state, produces an observable output.

HMM Parameters



In an HMM, we have the following components:

☑ Hidden States: These are the underlying states of the system that cannot be directly observed. Each state has certain characteristics or properties associated with it.

☑ Observations: These are the observable outputs or measurements associated with each hidden state. Observations provide information about the underlying states, but they may not uniquely determine the current state.

☑ Transition Probabilities: These represent the probabilities of transitioning from one hidden state to another. They capture the dynamics of the system and determine the state sequence.

☑ Emission Probabilities: These represent the probabilities of generating a particular observation given a hidden state. They describe the relationship between the hidden states and the observations.

The alerts/ emissions

	Alerts
0	Consecutive TCP small segments exceeding threshold
1	(http_inspect) NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE
2	Reset outside window
3	(spp_ssh) Protocol mismatch
4	TCP Timestamp is missing
5	(portscan) TCP Portscan
6	(http_inspect) TOO MANY PIPELINED REQUESTS
7	(http_inspect) LONG HEADER
8	(portscan) UDP Distributed Portscan
9	(spp_sdf) SDF Combination Alert
10	(http_inspect) UNESCAPED SPACE IN HTTP URI
11	Bad segment, adjusted size <= 0
12	(portscan) TCP Portsweep
13	(portscan) UDP Portsweep
14	SENSITIVE-DATA Email Addresses
15	(portscan) UDP Portscan
16	(http_inspect) NON-RFC DEFINED CHAR
17	(spp_reputation) packets blacklisted
18	(portscan) TCP Distributed Portscan
19	MALWARE-CNC Win.Trojan.ZeroAccess outbound connection
20	MALWARE-CNC Win.Trojan.ZeroAccess outbound connection

The states

	States
0	Potentially Bad Traffic
1	Unknown Traffic
2	Detection of a non-standard protocol or event
3	Attempted Information Leak
4	Senstive Data
5	A Network Trojan was detected

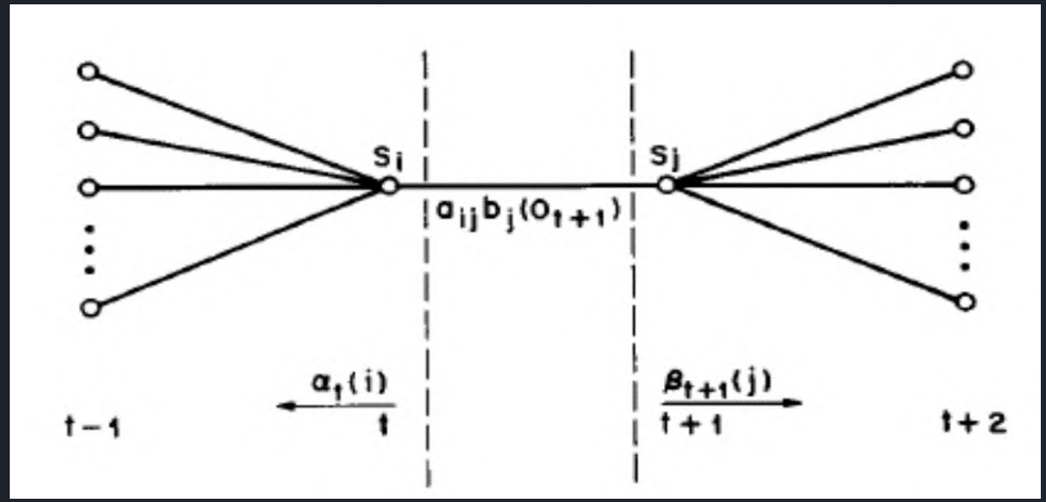
Baum Welch Algorithm

The Baum Welch Algorithm is used to learn the model parameters and return the model parameter which gives best results for a given training sequence.

$$\xi_t(i, j) = P(q_t = S_i, q_{t+1} = S_j | O, \lambda).$$

$$\xi_t(i, j) = \frac{\alpha_t(i)\, a_{ij} b_j(O_{t+1})\, \beta_{t+1}(j)}{P(O|\lambda)}$$

$$= \frac{\alpha_t(i)\, a_{ij} b_j(O_{t+1})\, \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i)\, a_{ij} b_j(O_{t+1})\, \beta_{t+1}(j)}$$



The Pseudocode

Algorithm 1: The Baum-Welch algorithm

Initialization: Θ_0, $\{O_{1:T}\}$

Looping:

for $l = 1, \ldots, l_{\max}$ **do**

1. Forward-Backward calculations:

$$\alpha_1(i) = \pi_i b_i(O_1),\ \beta_T(i) = 1,$$

$$\alpha_t(i) = \Big[\sum_{j=1}^{K} \alpha_{t-1}(j) a_{ji}\Big] b_j(O_t),\ \beta_t(i) = \sum_{j=1}^{K} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)$$

$$\text{for } 1 \le i \le K,\ 1 \le t \le T-1$$

2. E-step:

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{K} \alpha_t(j)\beta_t(j)},\ \xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}\beta_t(j))}$$

$$\text{for } 1 \le i \le K,\ 1 \le j \le K,\ 1 \le t \le T-1$$

3. M-step:

$$\pi_i = \frac{\gamma_1(i)}{\sum_{j=1}^{K} \gamma_1(j)},\ a_{ij} = \frac{\sum_{t=1}^{T} \varepsilon_t(i,j)}{\sum_{k=1}^{K}\sum_{t=1}^{T} \varepsilon_t(i,k)},\ w_{kd} = \frac{\sum_{t=1}^{T} \gamma_t(k,d)}{\sum_{t=1}^{T}\sum_{r=1}^{D} \gamma_t(k,r)}$$

$$\text{for } 1 \le i \le K,\ 1 \le j \le K,\ 1 \le k \le K, 1 \le d \le D$$

end

Result: $\{\Theta_l\}_{l=0}^{l_{\max}}$

Viterbi Algorithm

Viterbi Algorithm is used to predict the most likely sequence of states leading to the given observation

1) choose states that are individually most likely

$$\gamma_t(i) = P(q_t = S_i \mid O, \lambda)$$

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(O \mid \lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

$$\sum_{i=1}^{N} \gamma_t(i) = 1$$

- Initialization $\delta_1(i) = \pi_i b_i(O_1)$

  $\psi_1(i) = 0$

- Inductive step

  $\delta_t(j) = \max_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}] \cdot b_j(O_t)$ $\quad 2 \le t \le T$

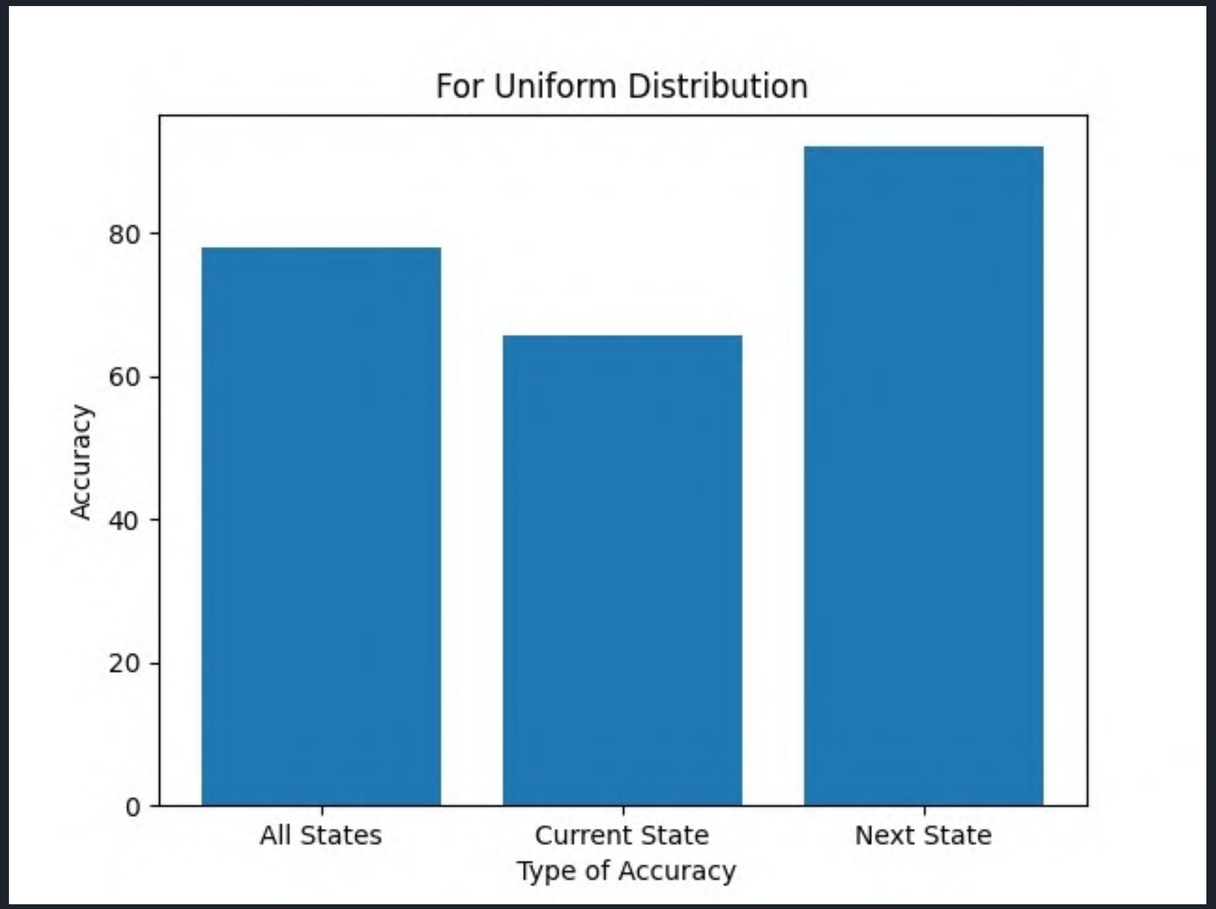
  $\psi_t(j) = \operatorname{argmax}_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}]$ $\quad 1 \le j \le N$

- Termination

  $P^* = \max_{1 \le i \le N} [\delta_T(i)]$

  $q_T^* = \operatorname{argmax}_{1 \le i \le N} [\delta_T(i)]$ $\quad q_t^* = \psi_{t+1}(q_{t+1}^*)$

ACCURACY OF PREDICTION FOR UNIFORM INITIALIZATION



The Hidden Markov Model gives a fairly good accuracy over uniform initialization