공학석사학위논문

침입 탐지 시스템 구현을 위한 Hidden Markov Model의 연구

A Study on Hidden Markov Models for Intrusion Detection

2001년 2월

서울대학교 대학원

컴퓨터공학과

김 인 영

침입 탐지 시스템 구현을 위한 Hidden Markov Model의 연구

A Study on Hidden Markov Models for Intrusion Detection

지도교수 장 병 탁

이 논문을 공학석사 학위논문으로 제출함.
2000년 12월

서울대학교 대학원
컴퓨터공학과
김 인 영

김인영의 공학석사 학위논문을 인준함
2001년 2월

위원장 _____ 인
부위원장 _____ 인
위원 _____ 인

초록

네트워크 기술의 발전으로 거의 모든 시스템이 개방 환경에 놓이게 되면서 시스템 침입과 같은 위협이 증가하고 있으며 여러 가지 기술을 이용한 침입 탐지 시스템이 개발되고 있다. 그러나 대부분의 시스템들에서 사용하는 기술은 기존의 침입 패턴에 대해서만 유효할 뿐 새로운 침입 패턴에 대해서는 그렇지 못하다. 그러므로 점차 변화하는 침입 기술에 대응하기 위해서는 기계 학습 방법이 필요하다.

본 논문에서는 기계 학습 방법을 이용한 침입 탐지 시스템 구축 방법을 연구한다. 크게 두 가지 방법으로 한 가지는 코호넨 신경망을 이용한 클러스터링 기반의 탐지 모델이고 또 다른 한가지는 은닉 마르코프 모델을 이용한 방법이다.

코호넨 신경망을 이용하면 클러스터링뿐만 아니라 그 데이터가 할당된 클러스터의 대표값(Centroid)과의 거리 차이(Quantization Error)를 알아볼 수 있다. 이를 이용하면 어떤 데이터가 정상적인 분포를 따르는지 정상적인 분포에서 벗어나는 비정상적인 데이터인지 알 수 있다. 유닉스 시스템 사용자의 명령어 사용 패턴에 적용하여 어떤 사용자의 명령어 사용 패턴이 정상적인 것인지 비정상적인 것인지 알 수 있다. 또한 은닉 마르코프 모델을 이용하여 각 사용자별 프로파일을 구성하고 이를 이용하여 사용자의 비정상적인 명령어 사용을 탐지한다. 이 경우에는 likelihood값을 이용하여 정상/비정상을 분류한다.

키워드 : 비정상 행위 탐지, 코호넨 신경망, 은닉 마르코프 모델.

차례

그림 차례

표 차례

1. 서론

1.1 연구 목적 및 배경

최근 인터넷, 인트라넷 등과 같은 정보 기술의 활용으로 인해 예전과는 달리 거의 모든 시스템이 개방 환경과 상호 연결성에 노출되게 되었다. 이로 인해 얻게 되는 이익도 많은 반면 시스템 침입에 의한 정보의 유출과 파괴 같은 역기능도 기하 급수적으로 늘고 있다. 검찰과 경찰의 수사 발표 및 한국 정보 보호 센터 기술지원 사항에 근거한 '96년부터 '99년까지 국내 침해 사고의 현황을 살펴보면 [표 1]과 같다[Cert]. 또한 '99년도 기관별 해킹 피해 건수[표 2]와 피해 현황을 통해 우리 나라도 더 이상 컴퓨터 범죄로부터 안전 지대가 아니라는 것과 보안 관련 기술의 낙후로 인한 손실을 예상할 수 있다.

연도	96년도	97년도	98년도	99년도
해킹 사고	147	64	158	572
증가율(%)	-	-44%	247%	362%

[표 1] 연도별 해킹 사고 변화

기관	대학(ac.kr)	기업(co.kr)	비영리(or.kr)	연구소(re.kr)	기타	합계
피해 건수	262	248	22	11	29	572

[표 2] '99년도 기관별 해킹 피해 건수

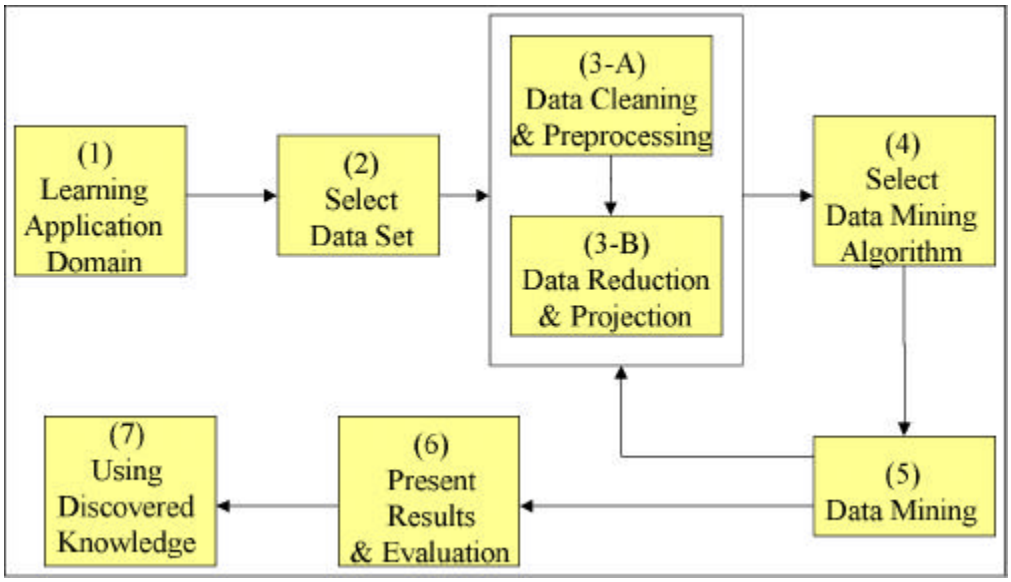
질적으로 양적으로 증가해 가는 컴퓨터 범죄를 막기 위한 국내의 기술 수준은 아주 미약하다고 볼 수 있다. 특히 시스템 불법 침입에 대한 실시간 정보 체계의 기술 개발은 전무한 상태이며 이러한 기술은 기술적 난이도가 크고 개발 비용이 막대한 반면 수요가 별로 없기 때문에 개발을 시도하기에는 많은 어려움이 있다. 침입 탐지 시스템의 구현에는 다음과 같은 어려움이 수반된다.

첫째, 침입 유형의 다양성과 유동성
둘째, 침입 행위의 자동성 및 피해의 광역성
셋째, 범죄 행위의 폐쇄성과 은닉성
넷째, 방대한 양의 데이터 조작

이러한 어려움을 극복하기 위해서는 시스템이 능동적으로 침입 탐지에 필요한 데이터를 획득하고 이를 분석하여 불법 침입의 유형이나 패턴을 예측함으로써 날로 다양해지고 자동화되어 가는 침입 행위에 적극적으로 대응해 나갈 수 있도록 하는 것이 필요하다. 따라서 다양한 침입 형태에 적극적으로 대응해 나갈 수 있는 불법 침입 탐지 시스템을 만들기 위해서는 아래와 같은 기술들이 수반되어야 한다.

첫째, 관련 데이터들을 능동적으로 수집하여 데이터 베이스화 할 수 있는 자동화된 능력
둘째, 방대한 데이터들로부터 효과적으로 내재된 규칙을 찾아낼 수 있는 능력
셋째, 새로운 상황에 대해 끊임없이 정보를 갱신할 수 있는 자동화된 능력

이러한 기술들은 현재 인공지능의 기계학습 기술을 응용한 데이터 마이닝 분야에서 활발이 연구되고 있다. 일반적으로 데이터 마이닝이란 주어진 데이터들로부터 알려지지 않은 유용한 지식 정보를 획득하는 일련의 과정을 일컫는다[그림 1]



[그림 1] 데이터 마이닝의 수행 절차

본 논문에서는 이러한 일반적인 데이터 마이닝 기법을 이용하여 불법 침입에 능동적으로 대응하는 침입 탐지 시스템을 구축하고자 한다.

1.2 연구의 범위

기존의 보안 관련 시스템들에는 보통 능동적인 데이터 습득 및 이를 통한 자동화된 학습 과정이 존재하지 않는다. 대신 흔히 사용되는 방식은 현재 사용되는 해킹의 패턴을 관리자가 분석한 후 이러한 절차 내지는 패턴을 직접 if-then 규칙으로 시스템 안에 심어주고 이러한 규칙에 적용되는 사용자

의 접속은 불법 침입에 해당하는 것이라고 규정하는 것이다. 이러한 경우 기존에 알려진 침입 패턴에 대해서는 탐지할 수 있지만 새로운 침입 패턴으로 접근할 경우에는 침입 여부조차 알 수가 없다. 설령 네트웍 패킷으로부터 데이터를 획득한다고 하더라도 대부분 단순한 통계적인 방법에 의해 누적된 데이터를 피상적으로 분석하는 수준에 그치는 정도였다.

이러한 단점을 보완하기 위해서 최근 일부에서는 기계 학습 기법을 응용한 데이터 마이닝 기법을 사용하여 침입 패턴을 탐지하려는 시도가 있어 왔다[Lee *et al.*, 1997].

본 연구에서는 시스템이 능동적으로 데이터를 획득하고 이렇게 획득된 데이터를 정제 가공하여 기계 학습을 통해 내재된 규칙 내지는 패턴을 자동적으로 학습하도록 함으로써 기존의 침입 탐지 시스템에서의 단점을 극복하고자 한다.

네트웍을 통한 불법적인 시스템 침입을 탐지하기 위한 침입 탐지 시스템(IDS : Intrusion Detection System)은 다음과 같이 분류될 수 있다[표 3].

침입모델(Intrusion Model)을 기반으로 하는 분류	비정상(Anomaly)탐지	비정상적인 행위나 컴퓨터 자원의 사용 탐지
	오용(Misuse) 탐지	시스템과 응용 프로그램의 취약점을 이용한 공격을 탐지
자료 수집원(Data Source)을 기반으로 하는 분류	단일 호스트 기반	단일 호스트로부터 모아진 감사(Audit)데이터를 침입 탐지에 사용
	멀티 호스트 기반	여러 호스트들로부터 생성되고 모아진 감사 데이터를 침입 탐지에 사용
	네트웍 기반	호스트들의 감사 데이터와 네트웍 트래픽 데이터를 모아 침입 탐지에 사용

[표 3] 침입 탐지 시스템의 분류

본 논문에서는 단일 호스트 기반 비정상 탐지(Anomaly Detection)를 위한 모델을 연구한다. 즉, 단일 호스트에서의 사용자들에 대해 데이터를 수집하고 그 사용자들의 비정상적인 행위 탐지를 목적으로 하는 연구를 수행한다. 위에서 설명했듯이 기존의 if-then 규칙에 의한 대부분의 침입 탐지 시스템은 오용(Misuse) 탐지 시스템에 속하며 이러한 시스템의 경우 알려진 침입 패턴에 대해서는 유용하지만 새로운 패턴으로 침입을 시도할 경우 탐지하지 못하게 된다. 반면 비정상 탐지 시스템에서는 각각의 침입 패턴을 기억하고 있을 필요가 없고 사용자의 평소 행동, 즉 명령어 사용 패턴 등을 학습하고 있다가 평소와 다르게 비정상적으로 행동할 경우 이를 탐지하면

된다. 따라서 전혀 새로운 침입 패턴으로 접근해오더라도 이를 비정상적인 행위로 인식하여 탐지할 수 있는 것이다.

1.3 논문의 구성

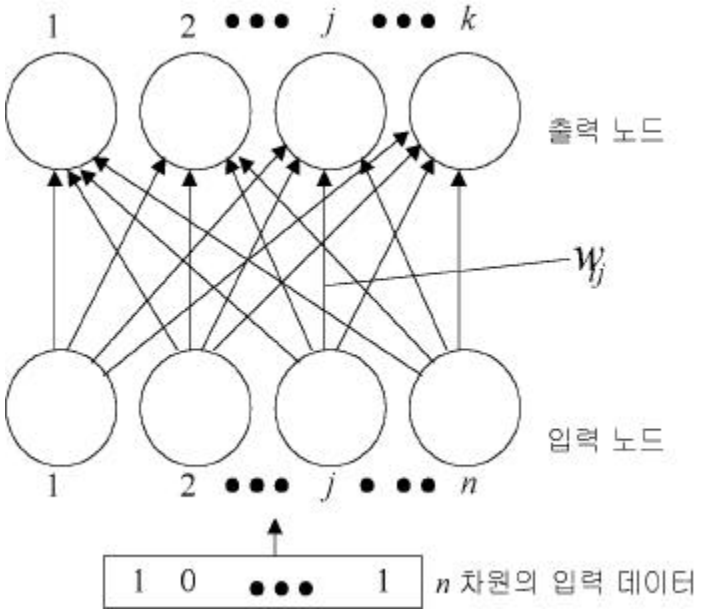
논문의 구성은 다음과 같다. 2장에서는 비정상 탐지 시스템 구축을 위한 여러 가지 학습 방법론에 관한 연구를 다룰 것이다. 3장에서는 코호넨 네트워크를 이용한 비정상 탐지 모델과 은닉 마코프 모델(Hidden Markov Model)을 이용한 비정상 탐지에 대해 설명을 하고 4장에서는 실험 및 결과를 제시할 것이며 마지막으로 5장에서 본 연구의 내용을 요약하고 결론을 맺는다.

2. 관련 연구

이 장에서는 비정상 탐지 시스템을 학습시키기 위한 기계 학습 알고리즘 몇 가지에 대해 설명하고 각각의 학습 방법의 특징에 대해서도 알아보기로 한다.

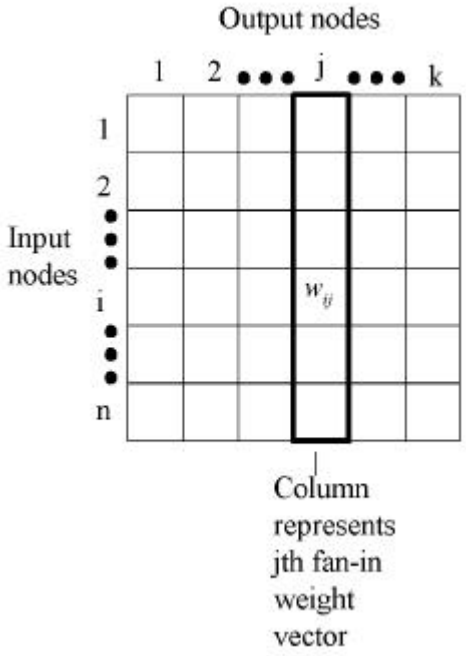
2.1 코호넨 네트워크의 구조 및 학습 알고리즘

코호넨 네트워크(Self-Organizing feature Map)는 무감독 학습(Unsupervised Learning)의 일종으로 스스로 n차원의 입력 데이터들을 클러스터링하여 그 결과를 2차원에 사상시켜준다. 코호넨 네트워크는 2개의 층으로 이루어진 신경망으로 그 구조는 [그림 2]와 같다[Chester, 1993].



[그림2] 코호넨 네크워크의 구조

그림에서 볼 수 있는 것처럼 2개의 층은 각가 입력층(Input Layer)와 출력층(Output Layer)를 나타낸다. 입력층은 n차원의 입력 데이터를 표현하기 위해 n개의 입력 노드로 이루어지고 출력층은 k개의 분류 영역(Decision Region)을 표현하기 위한 k개의 출력 노드로 이루어진다. 각각의 입력 노드들은 모든 출력 노드들과 연결되어 있으며 연결은 입력층에서 출력층의 방향으로 되어 있다. 위의 그림에서 W_{ij}는 입력 노드 i와 출력 노드 j사이의 연결 가중치(weight)를 나타낸다.



[그림 3] 연결 가중치 행렬

[그림 3]는 입력 노드 i와 출력 노드 j를 연결하는 연결 가중치 W_{ij}들의 행렬을 보여준다. 행렬에서 i번째 행은 입력 노드 i로부터 각 출력 노드로 나가는 연결 가중치를 나타내며 j번째 열은 각 입력 노드로부터 출력 노드

j로 들어오는 연결 가중치를 나타낸다. 여기서 j번째 열로 나타내어지는 벡터는 j번째 Fan-in weight vector라고 하며 입력 벡터와의 거리(유클리드 거리) 계산에 쓰인다.

코호넨 네트워크의 학습 알고리즘에 대해서 설명하고자 한다. 코호넨 네트워크를 학습시키기 전에 우선 초기화 과정을 거쳐야 하는데 이 때에는 두 가지에 대해 초기화를 하면 된다. 하나는 연결 가중치 벡터가 임의의 값을 가지면서 적절하게 초기화되어야 하는 것이고 또 다른 하나는 연결 가중치 벡터와 입력 벡터가 통상 0에서 1사이의 정규화된(normalized) 값을 사용하도록 하는 것이다[참고 문헌].

코호넨 네트워크를 이용하여 학습하는 과정은 각 연결 가중치 벡터의 값을 초기화하는 일로 시작된다. 초기 상태에서는 연결 가중치들을 임의로 할당한다. 임의로 연결 가중치를 할당한 후 학습에 참여할 노드들을 선택하게 된다.

코호넨 학습에서의 각 출력 노드들은 학습할 수 있는 특권을 부여받으려고 서로 경쟁하는데 입력 데이터와 가장 유사한 노드가 승자가 된다. 또한 이 승자 노드와 이웃하여 있는 노드들에 대해서도 어느 정도의 학습이 허용된다. 전체 노드들에 대해서 학습(데이터에 적응, 즉 연결 가중치를 조정)하는 것이 아니라 승자 노드와 그 이웃 노드들에 대해서만 학습이 이루어진다는 점에서 다른 신경망과의 차이를 보인다.

승자 노드의 선택을 위해서 각 출력 노드의 연결 가중치 벡터와 입력 벡터와의 유사성을 측정한다. 유사성 측정은 여러 가지 방법으로 할 수 있는데 유클리드 거리를 이용하는 것도 하나의 방법이다. 입력 벡터와 k개의 Fan-in weight vector 사이의 유클리드 거리를 구하여 입력 벡터와 가장 유사한 (유클리드 거리가 가장 작은) j번째 Fan-in weight vector를 찾으면 그 j번째 출력 노드가 승자가 된다.

승자 노드를 결정하고 난 후에는 코호넨 학습 규칙에 따라 연결 가중치를 조정해야 하며 그 값은 다음 식에 따라 변경된다.

$$w^j(t+1) = w^j(t) + \alpha(t)[x(t) - w^j(t)] \quad \text{(식 1)}$$

위의 식에서 $w^j(t)$는 조정되기 이전의 연결 가중치 벡터이고 $w^j(t+1)$는 조정된 후의 연결 가중치 벡터이다. 또한 $\alpha(t)$는 학습 상수(learning rate)이고 $x(t)$는 입력 벡터를 나타낸다. 이 식의 의미는 j번째 출력 노드가 승자가 되었으면 그 노드의 연결 가중치 벡터를 입력 벡터(입력 데이터)쪽으로 이동시켜간다. 즉, Fan-in weight vector를 입력 데이터 벡터와 유사하게 만들어 가는 것이다.

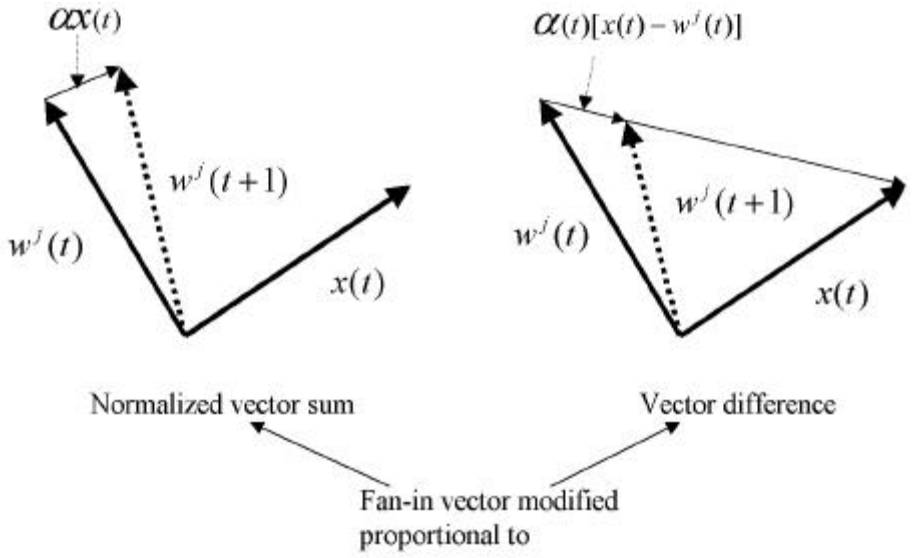
여기서 학습 상수를 고정된 값으로 하지 않고 학습이 진행됨에 따라 변하게 할 수 있는데 예를 들면 다음의 식과 같이 정할 수 있다[Haykin, 1999].

$$\alpha(t) = 0.1(1 - t/10^4) \quad \text{(식 2)}$$

(식2)는 10,000 번의 가중치 조정을 하게 된다는 가정하에서 학습 상수의 초기값을 0.1로 주고 10,000 번째의 사이클에 이르러 학습 상수가 0이 되어 학습이 끝나게 된다는 의미이다.

아래의 [그림 4]는 위의 식을 이용한 연결 가중치 벡터의 갱신 과정을 보여주는데 연결 가중치 벡터가 입력 벡터 쪽으로 움직여 가는 것을 알 수 있다. 그림의 왼쪽은 Normalized Vector Sum을 이용하여 연결 가중치 벡터를

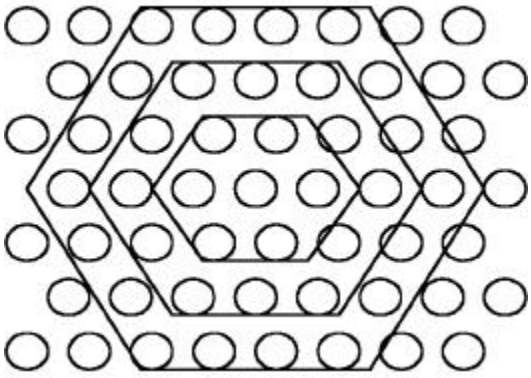
갱신하는 방법이고 그림의 오른쪽은 Vector Difference를 이용하여 갱신하는 방법이다. 두 가지 방법이 서로 다르지만 연결 가중치 벡터 갱신에 미치는 영향은 같다. 그림을 보면 왼쪽의 경우와 오른쪽의 경우 연결 가중치 벡터가 입력 벡터의 방향으로 이동하는 것을 알 수 있다. 따라서 연결 가중치 벡터를 갱신함으로써 얻는 효과는 임의의 값으로 시작한 연결 가중치 벡터가 점차 입력 데이터를 근사하도록 하는 것이다.



[그림 4] Fan-in vector 갱신

학습이 이루어 질 때 승자 노드 주위 얼마만큼의 범위까지 학습에 참여시킬지 정하는 것은 이웃 반경(neighborhood)을 정하는 것이라고 볼 수 있다. 이웃 반경의 크기는 고정된 것이 아니고 학습을 하는 도중에 계속 변할 수 있다. 학습의 초반에는 이웃 반경의 크기를 크게 잡고 학습이 진행됨에 따라 차츰 이웃 반경의 크기를 줄여 나가게 된다. 예를 들어 처음에는 모든 출력들에 대해서 연결 가중치를 조정해나가다가 나중에는 승자 노드와 승자

노드에 바로 인접한 노드들에 대해서만 연결 가중치를 변경시킬 수 있다. 이것을 그림으로 표현한 것이 [그림 5]이다.



[그림5] 이웃 반경의 크기조정

위에서 설명한 학습 과정을 하나의 알고리즘으로 표현하면 아래와 같다.

[단계 1] 연결 가중치를 초기화한다. n개의 입력 노드로부터 k개의 출력 노드 사이의 연결 가중치를 임의의 작은 수로 초기화한다. 초기의 이웃 반경은 충분히 크게 잡았다가 점차로 줄어든다.

[단계 2] 새로운 입력 벡터를 선택한다.

[단계 3] 입력 벡터와 모든 출력 노드 사이의 거리를 계산한다. 입력 벡터와 출력 노드 j 사이의 거리 d_j 는 (식 3)과 같이 계산한다.

$$d_j = \sum_{i=0}^{n-1} (x_i(t) - w_{ij}(t))^2 \qquad \text{(식 3)}$$

여기서 $x_i(t)$ 는 시각 t 에서의 i 번째 입력 벡터이고 $w_{ij}(t)$는 시각 t 에서의 i 번째 입력 벡터와 j 번째 출력 노드 사이의 연결 가중치이다.

[단계 4] d_j 값이 최소인 출력 노드 j^*를 결정한다.

[단계 5] 노드 j^*와 그 이웃 반경 내에 속하는 노드들의 연결 가중치를 다음의 (식 4)에 의해 재조정한다.

$$w_{ij}(t+1) = w_{ij}(t) + \alpha(x_i(t) - w_{ij}(t)) \qquad \text{(식 4)}$$

여기에서 j 는 j^*의 이웃 반경내의 노드이고 i는 0에서 $n-1$까지의 정수이다. α는 0과 1사이의 값을 가지는데 시간이 경과함에 따라 차츰 작아진다.

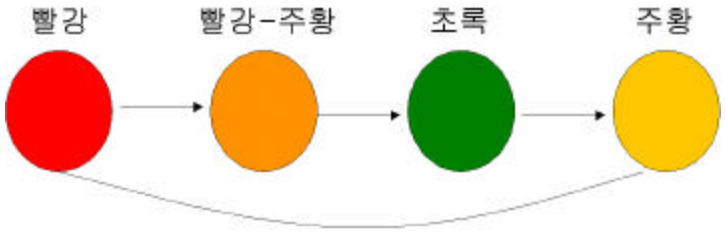
[단계 6] 조건을 만족할 때까지 단계 2로 가서 반복한다. 종료 조건은 두 가지로 생각해볼 수 있다. 하나는 정해진 반복 횟수를 정해놓고 그만큼 반복을 하면 종료하는 것이고 다른 하나는 연결 가중치 벡터가 더 이상 변하지 않고 수렴한 상태일 때 종료하는 것이다.

2.2 은닉 마르코프 모델(Hidden Markov Model)

은닉 마르코프 모델(Hidden Markov Model, 이후로는 간단히 HMM이라 한다.)은 각 상태(state)간 전이 확률을 가지는 유한 상태 기계(finite state machine)이며 각 상태들은 직접적으로 관찰 가능하지는 않으나, 대신

각 상태들이 일정 확률을 가지고 만들어내는 관찰 가능한 결과가 있다.

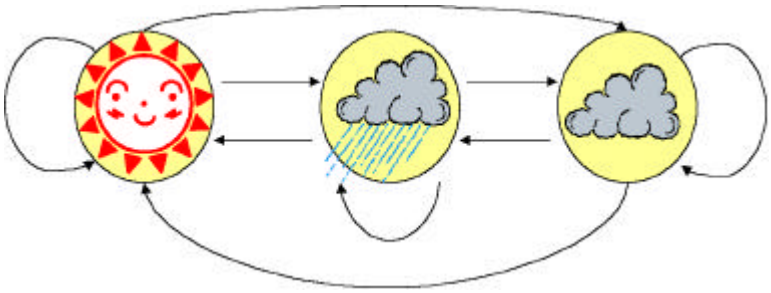
이해를 돕기 위해 HMM을 설명하기에 앞서 마르코프 프로세스(Markov Process)에 대해 먼저 설명하고자 한다[Leeds]. 아래의 [그림 6]을 보자.



[그림 6] 결정 시스템(Deterministic System)

각각의 상태는 전적으로 이전 상태에 의해서 결정된다. 만약 위의 그림처럼 현재 신호등이 초록색이었다면 그 다음은 주황색으로 변할 것이다. 이러한 시스템을 결정 시스템(deterministic system)이라 한다.

그런데, 신호등 예와 달리 날씨 예를 들어보면 내일의 날씨가 전적으로 오늘 날씨에만 의존하여 변한다고 말할 수 없기 때문에 이 시스템은 결정 시스템이 아니다. [그림 7]에서 보는 것처럼 오늘 날씨가 맑았다고 해서 내일 날씨가 맑다고 말할 수 없는 것이다. 확률적으로 맑을 수도 있고 흐릴 수도 있고 비가 올 수도 있다.



[그림 7] 비결정 시스템(Non-deterministic System)

이러한 문제를 풀기 위해 다음과 같은 가정을 한다. “모델의 상태는 오로지 이전 상태들에만 의존한다.” 이 가정을 마르코프 가정(Markov assumption)이라 한다. 마르코프 가정을 하게 되면 복잡한 문제를 굉장히 단순화시킬 수 있다는 장점이 있으나 과도하게 단순화하기 때문에 중요한 정보를 잃을 수도 있다는 단점이 있다. 예를 들어, 날씨 모델에서 내일의 날씨는 과거 며칠동안의 날씨에 의해서 예측할 수 있다는 뜻인데 이 때 바람, 기압 등의 다른 요인들은 고려하지 않는다.

마르코프 프로세스(Markov Process)는 상태간 전이가 오로지 이전 n개의 상태에 의존하여 이루어지는 프로세스를 말한다. 이 때 이 모델을 n차원 모델이라 하는데 n은 다음 상태를 결정하는데 영향을 미치는 상태의 개수를 말한다.

[표 4]의 상태간 전이 확률은 위에서 제시한 날씨 모델을 위한 것으로 날씨의 상태간 전이하는 확률을 보여주고 있다.

		Weather today		
		Sun	Cloud	Rain
Weather yesterday	Sun	0.5	0.25	0.25
	Cloud	0.375	0.125	0.5
	Rain	0.125	0.625	0.25

[표 4] 상태 전이 확률

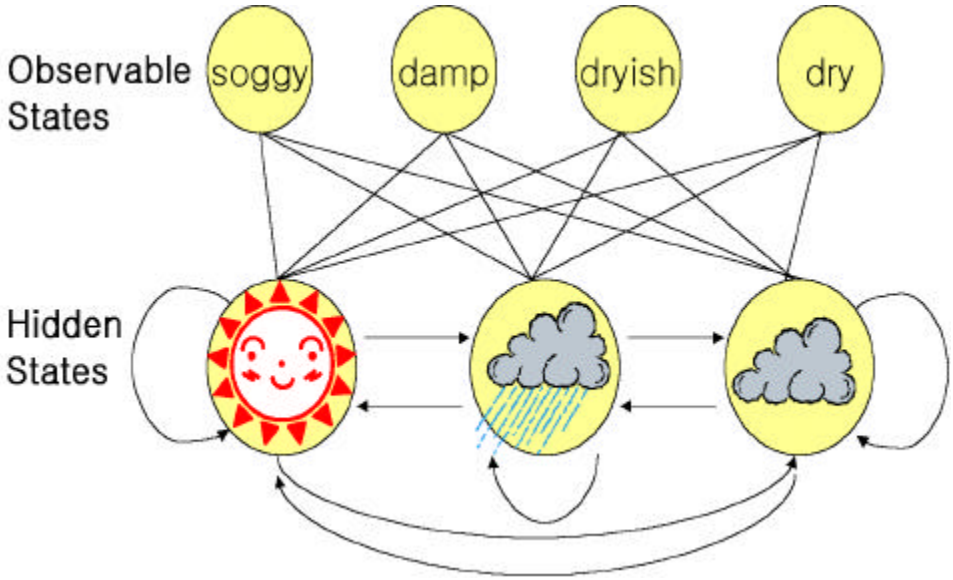
마르코프 프로세스는 다음의 세 가지로 설명될 수 있는 모든 시스템을 말한다[Brown].

상태 집합(states) : **가능한 상태들 - 예**, sunny, cloudy, rainy

π **벡터** : **시스템의 초기화 확률 벡터**

상태 전이 행렬(state transition matrix) : **각 상태간 전이 확률**

그런데, 어떤 경우에는 우리가 알고자 하는 정보를 마르코프 프로세스로부터 얻지 못하는 수가 있다. 예를 들어 직접적인 날씨 정보는 얻을 수 없고 해초의 상태만 알 수 있는 경우를 생각해보자. 즉, 우리가 알고 싶은 정보는 실제 날씨이지만 실제 날씨는 직접적으로 관찰할 수 없고 대신 날씨와 관계가 있는 다른 정보(해초의 마른 정도)를 관찰할 수 있을 때, 관찰 가능한 정보로부터 은닉되어 있는 정보를 얻어낼 수 있도록 하는 모델이 은닉 마르코프 모델이다. 이 때 은닉된 정보는 마르코프 프로세스이다. 날씨 모델에 대한 관찰 가능한 정보들과 은닉 정보들간의 관계가 아래의 [그림 8]로 표현된다.



[그림 8] 날씨 모델의 HMM

위의 그림에서 관찰 가능한 상태들과 은닉 상태들간의 관계는 은닉 상태의 마르코프 프로세스가 주어질 때 특정 관찰 상태를 만들어낼 확률을 나타낸다. 따라서 $\Pr(Obs|Sun)$의 합, $\Pr(Obs|Cloud)$의 합, $\Pr(Obs|Rain)$의 합은 각각 1이 된다.

은닉 상태들간의 마르코프 프로세스를 정의하는 확률 외에 관찰 상태들과 은닉 상태들 사이의 확률을 나타내는 confusion matrix가 새로 추가된다. 그 값은 특정 은닉 상태가 주어질 때 관찰 상태가 나타날 확률을 말한다. 위에서 다룬 날씨 예에 대한 confusion matrix는 [표 5]와 같이 표현할 수 있다. 또한 각 행의 합은 1이 됨을 확인할 수 있다.

		Seaweed			
		Dry	Dryish	Damp	Soggy
Weather	Sun	0.60	0.20	0.15	0.05
	Cloud	0.25	0.25	0.25	0.25
	Rain	0.05	0.10	0.35	0.50

[표 5] output 생성 확률

요약하자면, 은닉 마르코프 모델이란 아래와 같이 두 개의 상태 집합과 세 개의 확률 집합을 가지는 모델이다.

. 은닉 상태(Hidden States) : 마르코프 프로세스로 표현되는 은닉되어 있는 시스템의 상태

. 관찰 가능한 상태(Observable States) : 눈에 보이는 프로세스들의 상태

. π 벡터 : 은닉 상태들의 초기 확률을 나타내는 벡터

. 상태 전이 행렬(State transition matrix) : 은닉 상태들간의 전이 확률, 이전 상태에서 다음 상태로의 전이하는 확률

. Confusion matrix : 특정 은닉 상태에서 특정 관찰 가능 상태가 나타날 확률

은닉 마르코프 모델은 (Π, A, B)로 표현되며 각각은 아래와 같다.

$\Pi = (\pi_{ij})$: 은닉 상태에서 초기 확률

$A = (a_{ij})$: 상태 전이 확률, $\Pr(x_{i_t} | x_{j_{t-1}})$

$B = (b_{ij})$: confusion matrix, $\Pr(y_i | x_j)$

3. 사용자 모델링

비정상 행위 탐지 시스템에서의 사용자 모델링이란 컴퓨터 시스템 사용자의 정상적인 컴퓨터 사용 행위를 나타내는 모델, 내지는 프로파일을 개발하는 일이다. 사용자 프로파일을 만들 때 이용하는 데이터로는 여러 가지가 있을 수 있는데 그 중에서 명령어 사용 패턴을 이용하였다.

3.1절과 3.2절에서는 프로파일을 개발하는 방법에 대해 설명해보겠다.

3.1 다중 프로파일링 (Multi-Profiling)

보통 각 사용자는 하고자 하는 작업에 따라 다른 명령어 사용 패턴을 보인다[Rowe and Schiavo, 1998]. 그리고 한 명의 사용자에 대해서도 역시 여러 가지 다른 명령어 사용 패턴을 발견할 수 있다. 이러한 경우 각 사용자마다 한 가지의 프로파일을 개발하면 사용자의 여러 가지 명령어 사용 패턴을 제대로 표현하기 어렵다. 그러므로 각 사용자의 여러 가지 다른 명령어 사용 패턴을 학습하기 위해 다중 프로파일을 만드는 방법을 고안하게 되었다. 각 사용자마다 여러 개의 패턴을 프로파일에 저장하고 있으면서 새로운 패턴에 대해 학습하고 프로파일을 갱신하는 알고리즘은 아래와 같다 [Chen and Sycara, 1998].

사용자마다 최대 몇 개(M)의 패턴을 프로파일에 저장할 것인지 가정한다. 이 실험에서는 각 사용자마다 최대 10개의 패턴을 프로파일로 가진다고 정하였다. 즉, 각각의 사용자는 최대 10가지의 상이한 명령어 사용 패턴을 보일 수 있다.

ⅰ. M개의 패턴이 되기 전까지는 들어오는 패턴을 모두 프로파일에 저장한

다.

ii. 프로파일에 저장된 패턴의 개수가 M개를 넘은 경우는 다음의 과정을 거친다.

1. 새로 들어온 패턴을 포함한 $M+1$개의 패턴간 차이를 계산한다.
2. 두 패턴간 차이가 가장 작은 두 패턴을 통합한다.

위의 알고리즘을 통해서 M개의 상이한 패턴을 포함하는 프로파일을 유지할 수 있다. 새로 들어온 패턴이 프로파일에 속해 있는 어느 패턴과 유사한 패턴이라면 그 패턴과 함께 통합(Merge)될 것이며 전혀 새로운 패턴이라면 다른 유사한 패턴의 쌍 2개가 통합되면서 새로 들어온 패턴이 프로파일에 추가될 것이다. 이 때 벡터간의 유클리드 거리로 표현되는 패턴간 차이를 보면 새로 들어온 패턴이 기존에 학습된 프로파일에 있는 다른 패턴들과 얼마나 차이가 나는지 알 수 있다. 결국 사용자가 지금까지의 명령어 사용 패턴과는 다른 비정상적인 명령어 사용을 보임을 알 수 있다.

3.2 HMM을 이용한 프로파일링

각 사용자마다 하나의 HMM을 이용하여 프로파일을 만든다. HMM을 표현하기 위해 다음과 같은 변수들을 사용한다[Rabiner, 1989].

q_t : 시각 t일 때 HMM의 state

O_t : 시각 t일 때 관찰된 output

$\bar{q}$: state의 전체 sequence

$\overline{O}$: 관찰된 output의 전체 sequence

π_i : state i의 초기화 확률(prior probability)

a_{ij} : state i에서 j로 전이할 확률(state transition probability)

$b_i(o)$: HMM이 state i에 있을 때 output symbol o를 생성해낼 확률

Π : π_i의 행렬

A : a_{ij}의 행렬

B : $b_i(o)$의 행렬

θ : HMM의 모든 파라미터들의 집합 $\{\Pi, A, B\}$

K : HMM의 state 개수

$|\Sigma|$: 관찰 가능한 output symbol의 개수

여기서 관찰 가능한 output symbol은 각 사용자가 history에 남긴 명령어 데이터라 정의하고 state는 사용자의 명령어 사용 패턴을 의미한다. 그러므로 $|\Sigma|$는 사용자 데이터에 나타난 unique 명령어의 개수를 의미한다.

비정상 행위를 탐지하는 작업은 자기 자신인지(self) 아닌지(non-self)를 구별하는 바이너리 분류라고 볼 수 있다[Kim and Bentley, 1999]. 이러한 분류 작업을 위해서 주어진 output 데이터에 대해 모델 확률을 구하고 그 중 가장 큰 확률값을 가지는 모델에 그 데이터를 할당(assign)할 수 있다. 여기서 모델 확률은 베이즈 규칙(Bayes's rule)에 의해 다음과 같이 사후 확률(posterior probability)로부터 얻을 수 있다.

$$p(\theta|\overline{O}) = \frac{p(\overline{O}|\theta)p(\theta)}{p(\overline{O})} \qquad \text{(식 5)}$$

(식 5)에서 $p(\overline{O})$는 모든 클래스에 공통인 정규화 상수로 볼 수 있고, 모델 확률(model prior probability) $p(\theta)$은 domain knowledge로부터 얻을 수 있으나 여기서는 uniform probability를 사용하였다.

N개의 클래스 분류 문제, 즉 N명의 사용자 데이터에 대한 분류 문제에서는 관찰된 데이터 sequence를 maximum likelihood model에 할당하는 문제로 볼 수 있다.

$$class(\overline{O}) = \arg\max_{i \in 1 \ldots N} p(\theta_i | \overline{O}) \qquad \text{(식 6)}$$

즉, (식 6)과 같이 각 사용자마다 하나의 모델 HMM을 프로파일로 가지고 주어진 데이터를 각 N개의 모델과 비교하여 가장 유사한 모델로 분류하는 문제이다. 그런데 이 때 likelihood값이 임계값 테스트로부터 충분하지 않다는 판정이 나면 이 데이터를 비정상적이라고 판단한다.

4. 실험 및 결과

4.1 데이터

본 실험에서 관심을 가지는 것은 유닉스 시스템 사용자의 행동을 분석하고 각 사용자마다의 특정한 행동 패턴, 즉 명령어 사용에 있어서의 패턴을 학습하는 것이다. 이것은 각 사용자마다 사용하는 명령어와 그 빈도수가 다른 것을 이용하여 사용자의 명령어 사용 패턴을 학습하려는 것이다.

실험에 사용한 데이터는 8명의 사용자를 대상으로 2년 동안 수집된 데이터이다. 이 데이터는 유닉스 시스템에서 각 사용자의 명령어 사용 기록(History)을 이용하였으며 각 세션(Session)마다 구분이 되어 있다. 하나의 세션은 한 번의 로그인부터 로그아웃 사이의 모든 명령어 사용을 저장하고 있다. 전부 9개의 집합(Set)으로 구성되어 있는 이 데이터는 각 사용자마다 하나의 집합을 유지하며 그 중 다른 시스템에서 다른 프로젝트를 수행하던 한 명의 사용자의 경우에 각각을 다른 집합에 저장하였다. 실험에 이용한 데이터는 [Kdd]에서 얻을 수 있다.

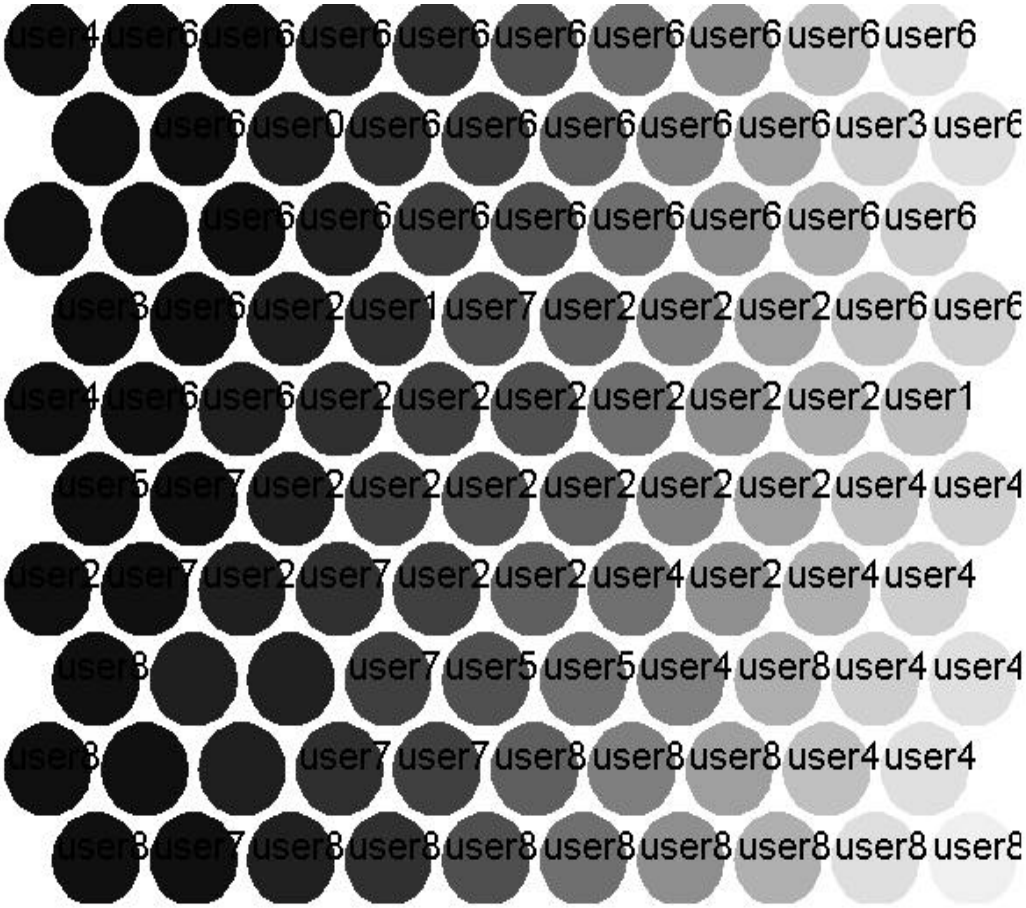
4.2 실험 결과

4.2.1 클러스터링을 이용한 비정상 행위 탐지

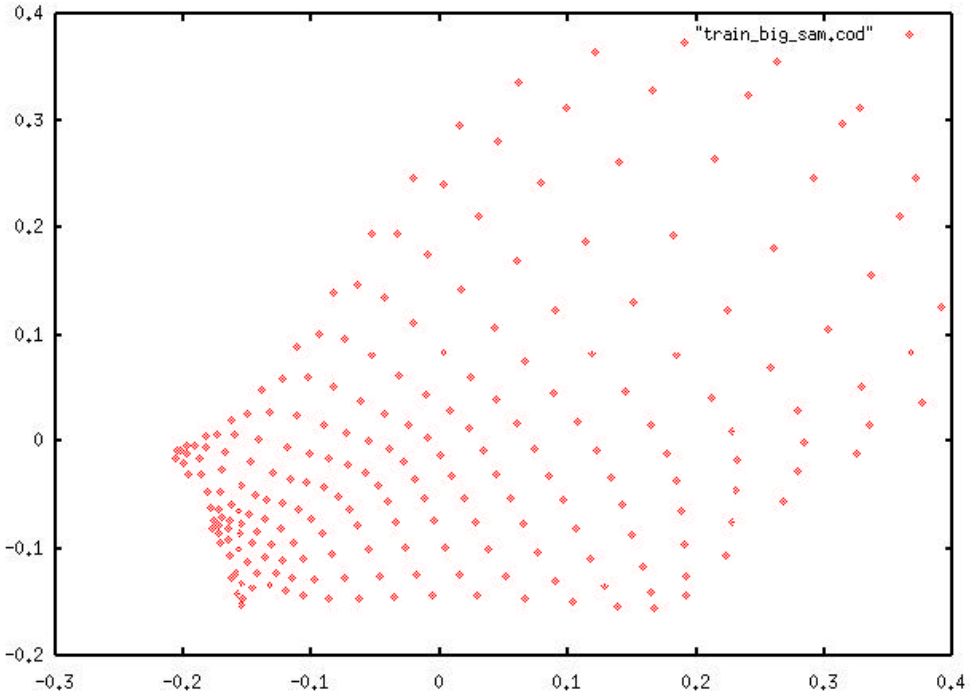
본 실험에서의 데이터 표현은 신경망 등의 학습에서 주로 사용하는 방법인 벡터 표현 방법을 사용하였다[Ryan *et al.,* 1998]. 각 세션별로 하나의 벡터를 만들어 사용하였고 실험에 사용된 벡터는 100개의 명령어에 대한 사용

빈도수를 이용한 100차원의 벡터이다.

실험은 두 가지 방법으로 수행하였다. 첫 번째로 코호넨 네트워크 [Kohonen *et al.,* 1995]를 이용한 실험에서는 전체 사용자들의 명령어 패턴을 이용하여 클러스터링 하였다. 클러스터링 결과 [그림 9]와 [그림 10]을 통해 데이터들의 전체적인 분포를 살펴볼 수 있다. [그림 9]에서 인접한 노드들은 서로 비슷한 유형의 패턴들이 이루는 클러스터들이고 거리가 멀고 명암의 차이가 많이 나는 노드들은 서로 다른 패턴을 나타내는 클러스터들이다. [그림 10]에서는 *n*차원의 데이터 벡터로 표현되는 각각의 클러스터를 2차원으로 표현한 것으로 데이터들의 전체적인 분포와 밀집 정도를 볼 수 있다.


user4 user6 user6 user6 user6 user6 user6 user6 user6 user6
user6 user0 user6 user6 user6 user6 user6 user3 user6
user6 user6 user6 user6 user6 user6 user6 user6
user3 user6 user2 user1 user7 user2 user2 user2 user6 user6
user4 user6 user6 user2 user2 user2 user2 user2 user2 user1
user6 user7 user2 user2 user2 user2 user2 user2 user4 user4
user2 user7 user2 user7 user2 user2 user4 user2 user4 user4
user8 user7 user5 user5 user4 user8 user4 user4
user8 user7 user7 user8 user8 user8 user4 user4
user8 user7 user8 user8 user8 user8 user8 user8 user8 user8

[그림 9] 클러스터링 결과



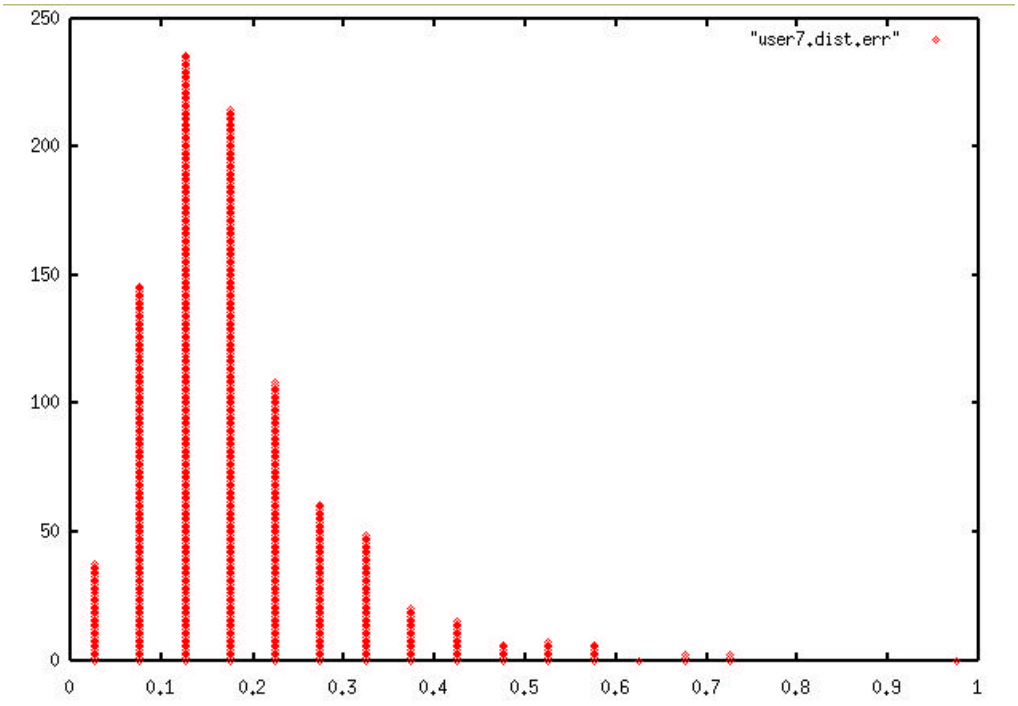
[그림 10] 2차원상의 클러스터 분포

¶Çí한 각 사용자의 명령어 패턴의 분포를 알아보기 위해 각각의 사용자별 클러스터링도 하였다. 이 경우 한 명의 사용자로부터 나온 여러 가지 명령어 패턴에 대해서 클러스터링하며 각 클러스터를 대표하는 Centroid 벡터값과 해당 클러스터에 속한 각각의 데이터와의 차이를 표현하는 Quantization Error[Kohonen, 1997]값을 통해서 비정상적인 패턴을 탐지할 수 있다.

두 번째 실험에서는 통계적인 방법을 이용하여 사용자의 명령어 패턴 학습 및 비정상적인 명령어 사용을 탐지한다. 앞에서 설명한 알고리즘대로 각 사용자마다 최대 M개의 명령어 패턴을 포함하는 프로파일을 유지하고 있다. 새로운 명령어 패턴이 주어지면 프로파일에 속해 있는 다른 명령어 패턴과의 유클리드 거리를 측정하여 그 차이가 Threshold θ 이상이면 비정상

패턴으로 간주한다. 그 후 새로 들어온 패턴을 포함한 $M+1$개 패턴 사이의 유사성을 비교하여 가장 유사한 두 개의 패턴은 하나의 패턴으로 통합한다. 그러면 다시 전체적으로 M개의 패턴이 유지되면서 새로운 패턴에 대한 학습이 끝나게 된다.

[그림 11]은 Threshold θ를 정하기 위해서 새로 들어온 명령어 패턴 벡터와 프로파일내의 M개의 벡터들과의 거리 중 최소 거리를 도수분포표로 나타낸 것이다. 아래 그림에서의 사용자는 새로 들어온 명령어 패턴이 기존의 패턴과 평균적으로 유클리드 거리 0.25의 차이가 나며 분산은 0.15라는 것을 알 수 있는데 만약 새로 들어온 패턴의 최소 거리가 θ 이상이면 비정상적인 명령어 사용이라고 탐지한다.

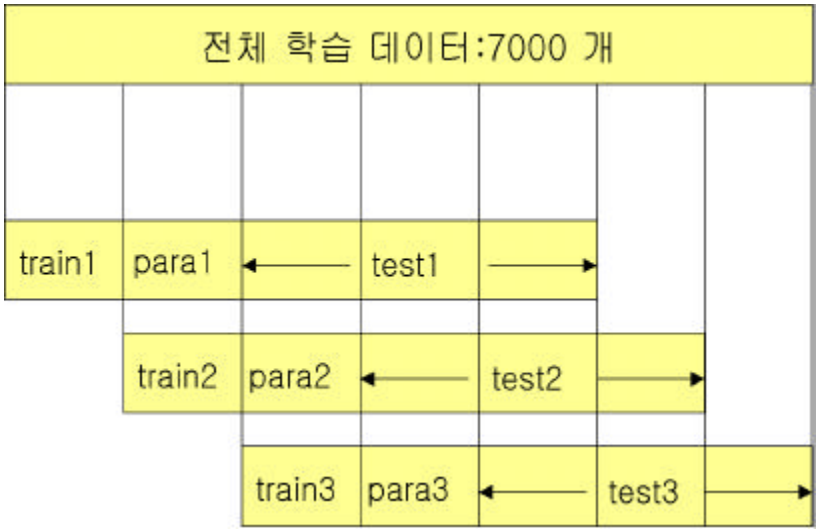


[그림 11] Mean : 0.25, Variance : 0.15

4.2.2 HMM을 이용한 비정상 행위 탐지

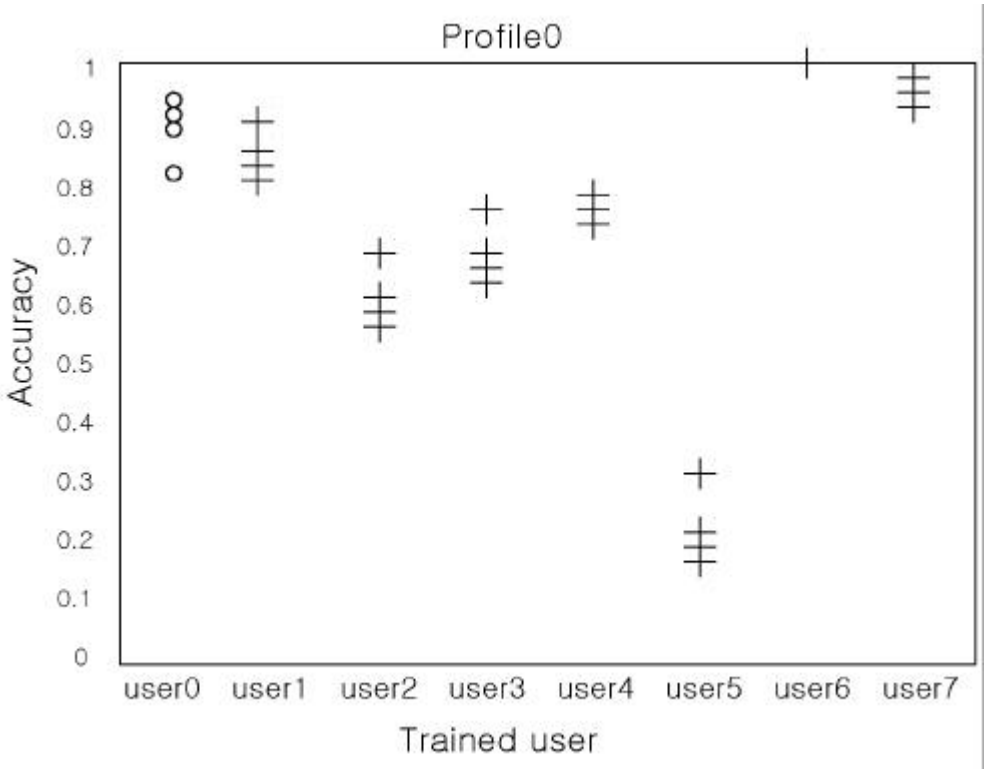
각 사용자마다 7,000개의 토큰으로부터 모델 HMM을 구성한다. 그런데 한 번 구성된 HMM은 마르코프 프로세스의 가정, 즉 각 상태간 전이 확률은 시간에 무관하다는 가정으로 인해 시간이 변함에 따라 변화하는 데이터를 잘 반영하지 못한다는 단점이 있기 때문에 한번 구성된 HMM은 특정한 시간 제한 내에서만 비정상 행위 탐지를 위해 사용된다. 즉, 하나의 고정 모델은 제한된 시간 내에서만 사용되고 그 이후에는 성능이 떨어지기 때문에 변화된 데이터로 다시 HMM을 구성한다.

그리고, 7000개의 명령어 토큰을 아래의 [그림 12]와 같이 나누어서 학습하였다. 세 번 HMM을 구성하였고 각 HMM 구성을 위해 5000개의 토큰 데이터를 1000, 1000, 3000으로 나누어 훈련, 파라미터(비정상 판별을 위한 threshold) 선택, 테스트에 사용하였다.



[그림 12] 훈련, 파라미터, 테스트 데이터

실험 결과는 [그림 13]과 같다. 여기서는 user0의 모델 HMM에 대해서 다른 사용자의 데이터를 적용시켜본 결과를 나타내고 있다. 그림에서 “o"로 나타나는 심볼은 허용 비율 (true accept rate)를 나타낸다. 그리고 ”+“로 나타나는 심볼은 비정상 탐지 비율(true detect rate)를 나타내며 이것은 user0의 모델에 다른 사용자의 데이터를 적용시켰을 때 비정상이라고 판정된 비율을 나타낸다.



[그림 13] user0 모델에 대한 정확도

5. 결론 및 향후 과제

신경망에 근거한 클러스터링 방법과 HMM을 이용한 방법에 근거한 두 가지 침입 탐지 실험으로 확인할 수 있었던 것은 두 가지 모두 비정상적인 명령어 사용을 탐지하는 데 효과적이라는 것이다. 기존에 학습된 프로파일과 상이한 패턴이 들어오면 코호넨 네트워크의 경우 Quantization Error값이 정상 분포에서 벗어나는 것을 알 수 있었고 마찬가지로 HMM에서도 likelihood값이 임계값을 넘지 못해서 비정상적인 데이터로 판정됨을 알 수 있다. 다만, 두 가지 실험의 차이점은 코호넨 네트워크를 이용한 클러스터링 기반 탐지 모델은 명령어의 사용 횟수만을 이용하기 때문에 모델 자체가 단순하지만 HMM을 이용한 모델은 단순한 명령어 사용 횟수뿐만이 아니라 그 순서를 학습할 수 있다는 장점이 있다. 그러므로 HMM을 이용한 모델이 좀 더 성능이 좋다는 결론을 내릴 수 있다.

앞으로 연구할 방향은 좀 더 효율적인 모델을 만들기 위해 HMM에서 적절한 state의 개수를 정하는 문제로 볼 수 있다. 만약, 사용자의 명령어 사용 패턴이 복잡하다면 state의 개수가 많아야 할 것이고 명령어 사용 패턴이 단순한 사용자의 경우는 state의 개수가 많으면 over-fit되는 결과를 초래할 수 있다.

또한 본 논문에서의 모델은 두 가지 모두 off-line 모델이다. 코호넨 네트워크를 이용한 모델이나 HMM 모델에 대해 on-line 탐지하는 문제의 경우 코호넨 신경망은 명령어 사용 횟수를 이용하므로 하나의 세션이 끝난 후에 데이터가 나오므로 온라인으로 적용하기는 어려움이 있을 것이다. 그러나 HMM의 경우는 명령어 사용 회수뿐만 아니라 sequence를 학습 가능하기 때문에 on-line 모델의 개발이 보다 적합하다.

참고 문헌

[Cert] http://www.certcc.or.kr/

[Chester, 1993] Michael Chester. *Neural Networks: A Tutorial,* Prentice Hall. pp. 42-49, 1993.

[Haykin, 1999] Simon Haykin. *Neural Networks: A Comprehensive Foundation,* Prentice Hall, pp. 443-483, 1999.

[Leeds] http://www.scs.leeds.ac.uk/scs-only/teaching-materials/HiddenMarkovModels/html_dev/main.html

[Brown] http://ftp.cs.brown.edu/research/ai/dynamics/tutorial/Documents/HiddenMarkovModels.html

[Chen and Sycara, 1998] Liren Chen and Katia Sycara. WebMate: A personal agent for browsing and searching, *Proceedings of the International Conference on Autonomous Agents (AA-98)*, 1998.

[Rabiner, 1989] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2), 1989

[Kdd] http://kdd.ics.uci.edu/databases/UNIX_user_data/UNIX_user_data.html

[Ryan *et al.*, 1998] Jake Ryan, Meng-Jang Lin, and Risto Miikkulainen. Intrusion Detection with Neural Networks. *Proceedings of NIPS-98*, pp. 943-949, 1998.

[Kohonen *et al.*, 1995] Teuvo Kohonen, Jussi Hynninen, Jari Kangas, and Jorma Laaksonen. *SOM_PAK The Self-Organizing Map Program Package*, 1995.

[Kohonen, 1997] Teuvo Kohonen. *Self-Organizing Maps: Second Edition*, Springer. pp. 48-51, 1997

[Rowe and Schiavo, 1998] Neil C. Rowe and Sandra Schiavo. An intelligent tutor for intrusion detection on computer systems. *Computers & Education 31*. pp. 395-404, 1998.

[Lee *et al.*, 1997] W. Lee, S. J. Stolfo, and P. K. Chen. Learning patterns from unix process execution traces for intrusion detection. In *AAAI Workshop: AI Approaches to Fraud Detection and Risk Management*. pp. 50-56, 1997.

[Kim and Bentley, 1999] Jungwon Kim and Peter Bentley. Negative

selection and niching by artificial immune system for network intrusion detection. *Proceedings of GECCO-99*, pp. 149-158, 1999.

Abstract

Due to increased networking of computer systems, the importance of building intrusion detection systems has increased.

Recently, several intrusion detection systems have been proposed based on various technologies. However, the techniques which have been used in most systems are useful only about the existing patterns of intrusion, not about the new patterns of intrusion. Therefore, it is necessary to use machine learning methods to be prepared to cope with the intrusion techniques that are evolving very fast.

This thesis describes an intrusion detection method that uses machine learning techniques. We study two different techniques. One is Kohonen neural networks, and hidden Markov models.

Using the Kohonen neural networks, we can measure the quatization error which is the difference between the centroid of cluster and the data clustered into that cluster. The quantization error can be used to recognize whether the data is generated from the normal distribution or not. When applied to the data which consist of Unix system user's command, the method could recognize whether the patterns of using commands are normal or abnormal

The profiles of each user has been constructed using hidden Markov models. By comparing the user profile and data, we can do anomaly detection. In this case, we used the likelihood to classify some specific data into normal/abnormal patterns.

Keyword : anomaly detection, Kohonen neural networks, hidden Markov models.

감사의 글

이제 마지막으로 감사의 글을 쓰면 지난 2년여의 석사 과정이 모두 끝나게 됩니다. 언제나 그렇듯이 시간이라는 것은 지나고 난 후에 아쉬움을 남기는군요.

가장 먼저, 저를 지도해주신 장병탁 교수님께 감사의 말씀을 드립니다. 제가 교수님께 배울 수 있었던 것은 학문에의 열의와 학생들에게 보여주신 다정하심이었습니다. 그리고 기존에 알고 있는 지식에 안주하지 않고 끊임없이 새로운 분야를 개척해 나가시는 모습이 지도 학생들에게 때로는 스트레스가 되기도 했지만 진정 학자다운 모습이라고 생각합니다.

다음으로 연구실 동기들와 선후배들에게 감사의 말을 전하고 싶습니다. 밤낮이 뒤바뀐 생활을 하는 선이 오빠, 성실한 재홍이, 저그맨 현구 오빠, 결혼을 앞두고 바쁜 태진 오빠, 엽기적인 승준이, 영우 오빠, 야무진 희주, 94학번 동기들인 동연, 수용, 정집이와 석사 동기들인 바니 형주, 박사장 상욱, 돌고래 호식, 유환이와 자연언어처리 연구실의 여러 선배님들께 감사드립니다. 묵묵히 열심히 일하시는 두 분 양박사님과 제균 오빠, 그리고 웃음 넘치는 후배 상준이, 인지과학 연구실의 언니 오빠들 - 강소영, 신동호, 이광주, 윤호진, 손정우, 하선영 에게도 고마움을 전하고 싶습니다.

마지막으로 우리 가족들에게 감사의 마음을 드리고 싶습니다. 제가 주저없이 공학도의 길을 선택해서 석사 과정까지 오게 된 데에는 아버지의 영향이 컸습니다. 자상한 아버지로, 유능한 엔지니어로 언제나 저에게 모범이 되셨습니다. 그리고 열 아들 부럽지 않은 딸 셋을 낳으시고 키워주신 친구 같은 엄마와 믿음직한 동생 진영이, 얄밉지만 이쁘게 봐줘야만 하는 숙영이에게 고마움을 전합니다. 지금 이 순간 가장 안타까운 것은 아버지께 감사의 말씀을 전할 수 없다는 것입니다.