Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.30, NO.3, Jun. 2020 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2020.30.3.455

비관계형 데이터베이스 환경에서 CNN과 RNN을 활용한 NoSQL 삽입 공격 탐지 모델

서 정 은,[1†] 문 종 섭[2‡]
[1,2]고려대학교 (대학원생, 교수)

Detection of NoSQL Injection Attack in Non-Relational Database Using Convolutional Neural Network and Recurrent Neural Network

Jeong-eun Seo,[1†] Jong-sub Moon[2‡]
[1,2]Korea University (Graduate student, Professor)

요 약

데이터 활용의 다양성이 높아짐에 따라 비관계형 데이터베이스 사용이 증가했으며, 이에 대한 NoSQL 삽입 공격 또한 증가했다. 전통적으로 NoSQL 삽입 공격을 탐지하기 위해 규칙 기반 탐지 방법론이 제안돼왔으나, 이 방식은 규칙의 범위를 벗어나 발생하는 삽입 공격에의 대응이 어렵다는 한계점이 있다. 이에 본 논문에서는 CNN 알고리즘을 이용해 특징을 추출하고, RNN 알고리즘을 활용해 NoSQL 삽입 공격을 탐지하는 기법을 제시한다. 또한, 실험을 통하여 본 논문에서 제시한 모델이 기존의 지도학습을 이용한 가장 우수한 모델보다 정확도는 10%, 정밀도는 4%, 재현율은 14%, F2-score는 0.082만큼 더 높은 비율로 NoSQL 삽입 공격을 탐지함을 보인다.

ABSTRACT

With a variety of data types and high utilization of data, non-relational databases are a popular data storage because it supports better availability and scalability. The increasing use of this technology also brings the risk of NoSQL injection attacks. Existing works mostly discuss the rule-based detection of NoSQL injection attacks that it is hard to deal with NoSQL queries beyond the coverage of the rules. In this paper, we propose a model for detecting NoSQL injection attacks. Our model is based on deep learning algorithms that select features from NoSQL queries using CNN, and classify NoSQL queries using RNN. Also, we experiment the proposed model to compare with existing models, and find that our model outperforms traditional models in terms of detection rate.

Keywords: NoSQL injection, Deep Learning, Convolutional Neural Network(CNN), Recurrent Neural Network(RNN)

I. 서 론

빅데이터 시대의 도래는 정형 데이터뿐만 아니라 반정형 및 비정형 데이터의 증대를 가져왔고, 이 현상으로부터 생성된 대용량 데이터의 실시간 처리기술을 요구했다. 이에 기존의 관계형 데이터베이스를 대체할 수 있는 비관계형 데이터베이스가 주목을 받았다.비관계형 데이터베이스는 분산형 컴퓨팅에 최적화돼 있으므로 대용량 데이터 저장에 유리하고, 다양한 데이터 모델을 지원하여 반정형 및 비정형 데이터 처리가 가능하다는 특징이 있다. 이에 비관계형 데이터베이스의 사용이 증가했는데, 실제로 Google, Facebook,eBay등 세계적인 기업에서 웹 애플리케이션에 비관계형 데이터베이스를 사용하고 있다[1].

Received(03. 30. 2020), Modified(05. 04. 2020), Accepted(05. 26. 2020)
† 주저자, sje5279@korea.ac.kr
‡ 교신저자, jsmoon@korea.ac.kr (Corresponding author)

삽입(injection) 취약점은 현존하는 주요 취약점 중 하나로, 2019 CWE Top25 Most Dangerous Software Error list[2]에 삽입이 포함돼있다. 또한, 가장 큰 오픈소스 웹 애플리케이션 보안 프로젝트인 OWASP(Open Web Application Security Project)가 발표한 웹 애플리케이션 취약점 Top10에 삽입 취약점이 포함돼있는데, 이 취약점의 대표적인 웹 애플리케이션에는 관계형 데이터베이스뿐만 아니라, 비관계형 데이터베이스도 포함돼있다[3]. 실제로 NoSQL(Not-only Structured Query Language) 삽입에 대해 CVE 취약점이 발견됐는데, 비관계형 데이터베이스 솔루션 중 가장 널리 사용되고 있는 MongoDB의 커넥터를 사용하는 IBM API 커넥터 5.0.0.0과 5.0.8.4 버전에서 NoSQL 삽입 공격이 유효함이 확인됐고[4], 또다른 비관계형 데이터베이스 솔루션인 CouchDB의 Sync Gateway 2.1.2 버전에서도 NoSQL 삽입 공격이 유효함이 확인됐다[5]. 이는 NoSQL 삽입 공격에 대한 위험이 존재함을 의미하며, 이 위험에 대응하기 위해서는 NoSQL 삽입 탐지가 우선시 돼야 한다.

기존 연구에서는 NoSQL 삽입 공격 유형을 분류하고[6], 분류된 공격에 대한 규칙 기반 탐지기법을 제시하고 있다[7,8]. 하지만 이는 일부 유형에만 국한돼있다는 한계가 있다.

이에 본 논문에서는 딥러닝(deep learning) 알고리즘에 속하는 CNN(Convolutional Neural Network)을 활용해 특징(feature)을 추출하고, RNN(Recurrent Neural Network)을 이용해 NoSQL 삽입 공격 여부를 판단하는 모델을 제안함으로써, 기존의 규칙 기반 탐지기법의 한계를 극복하고, 삽입 공격을 수행하는 삽입 질의(query)를 효과적으로 탐지하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서 NoSQL 삽입 공격 및 탐지와 관련된 기존 연구와 본 논문에서 사용되는 딥러닝 알고리즘에 관련된 연구를 소개한다. 3장에서는 본 논문에서 제안하는 모델의 구성을 기술하고, 제안된 모델이 삽입 공격 탐지율을 높일 수 있음을 보인다. 4장에서는 3장에서 제안한 모델을 통한 실험 결과를 보이고, 5장에서 결론과 향후 연구 방향을 제시한다.

II. 관련 연구

본 장에서는 NoSQL 삽입 공격 유형과 이 유형에 대한 규칙 기반 탐지(rule-based detection) 모델 및 지도학습(supervised learning) 기반 탐지 모델에 관련된 연구를 살펴보고, 본 논문에서 사용되는 딥러닝 알고리즘에 관련된 연구를 소개한다.

2.1 NoSQL 삽입 공격

비관계형 데이터베이스는 관계형 및 관계형이 아닌 데이터베이스 모두를 아우르는 개념으로, 특정 스키마를 요구하지 않기 때문에 다양한 데이터 모델을 지원한다. 이 때문에 비관계형 데이터베이스가 처음 도입됐을 때는 기존의 관계형 데이터베이스와는 달리 삽입 공격이 불가능할 것으로 판단됐다. 그러나 2015년 독일의 Saarland 대학에서 프랑스 통신사를 포함한 약 4만 개의 MongoDB 인스턴스가 인터넷상에 공개돼 있음과 동시에, 삽입 공격에 취약한 상태의 MongoDB가 상당수 존재함을 밝혔다[9]. Ron[6], Hou[7], Eassa[8] 등 또한 비관계형 데이터베이스가 삽입 공격에 취약함을 주장했다.

2.1.1 NoSQL 삽입 공격 유형

Ron[6]은 관계형 데이터베이스뿐만 아니라, 비관계형 데이터베이스 역시 삽입 공격에 취약함을 주장하고, 공격의 유형을 PHP Array 삽입, NoSQL OR 삽입, JavaScript 기반 삽입, Piggybacked 질의 삽입의 네 가지로 나누어 소개했다.

PHP Array 삽입 공격은 애플리케이션에 PHP 코드를 삽입하여 질의 조건이 수정되도록 하는 공격 유형이고, NoSQL OR 삽입 공격은 'or'을 사용하여 항상 참이 되는 조건을 삽입하여 원하는 질의를 수행하도록 하는 공격 유형이다. JavaScript 기반 삽입 공격은 강제로 true 값을 반환하거나, 항상 true 값을 반환하도록 하는 코드를 삽입하여 원하는 질의를 수행하도록 하는 공격 유형으로, 특히 MongoDB는 JavaScript 없이 질의 대부분을 표현할 수 있으나, JavaScript도 사용하기 때문에 이 공격에 취약하다. 마지막으로, Piggybacked 질의 삽입 공격은 제어 시퀀스(escape sequence)나 종결 문자(termination character)와 같은 특정 문자를 사용해 원하는 질의를 삽입하여 수행하는 공격

유형이다.

2.1.2 규칙 기반 탐지 모델

NoSQL 삽입 공격 유형에 대해 규칙 기반 탐지 기법을 제시하는 연구가 진행됐다.

Hou[7]는 MongoDB에 대해 입력 길이를 제한하고 프리페어드 스테이트먼트(parameterized statement)를 활용한 NoSQL 삽입 탐지 모델을 소개했고, Eassa[8]는 구문 분석(syntactic parser)을 활용한 탐지 모델을 소개했다. 하지만 Hou[7]의 모델은 NoSQL 삽입 공격 유형 중 PHP Array 삽입 공격과 JavaScript 기반 삽입에 대한 탐지에, Eassa[8]의 모델은 PHP Array 삽입에 대한 탐지에 국한된다는 한계가 있다.

2.1.3 지도학습 기반 탐지 모델

기존의 규칙 기반 탐지 모델[7,8]의 한계를 극복하기 위해, Islam[10]은 MongoDB와 CouchDB에 대해 지도학습을 활용한 NoSQL 삽입 탐지 기법을 소개했다. 해당 연구에서는 1,354개의 NoSQL 질의를 생성하고 19개의 특징 선택(feature selection) 후, 의사결정트리(Detecsion Tree) 알고리즘, 랜덤 포레스트(Random Forest) 알고리즘, 에이다부스트(AdaBoost) 알고리즘, 신경망(Neural Network) 알고리즘, SVM(support vector machine) 알고리즘, k-최근접 이웃(k-Nearest Neighbor) 알고리즘, XGBoost(eXtreme Gradient Boosting) 알고리즘을 활용한 7개의 지도학습 모델을 적용했다. Islam[10] 연구의 결과인 지도학습 결과는 Table 6.과 같다.

2.2 딥러닝 알고리즘

딥러닝은 여러 비선형 변환기법의 조합을 통해 추상화(abstraction)를 시도하는 기계학습(machine learning)의 한 분야[11]로, 본 논문에서는 딥러닝 알고리즘 중 CNN과 RNN을 살펴본다.

2.2.1 CNN

CNN은 입력값에 필터링 기법을 적용하여 특징을 추출하는 인공신경망으로, CNN의 일반적인 구조는

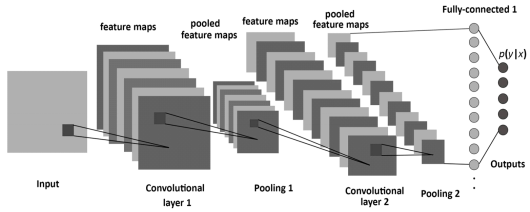


Fig. 1. Structure of CNN, consisting of convolutional, pooling, and fully-connected layers

Fig.1.과 같다[12].

CNN은 합성곱 계층(convolution layer)과 풀링 계층(pooling layer)으로 구성되고, 경우에 따라 완전연결 계층(fully-connected layer)이 추가된다.

합성곱 계층은 입력값에 필터링 기법을 적용하는 계층으로, 입력값과 필터의 가중치를 곱한 값을 모두 더하고, 이 값이 활성화 함수(activation function) 단계를 거쳐 하나의 합성 값을 생성한다. 이때, 하나의 필터는 '가로×세로×이전 층의 채널 개수' 크기의 가중치 행렬이고, 입력값에 대해 필터를 적용할 때는 필터를 일정한 간격으로 움직여가며(stride) 합성 값을 추출한다. 추출된 합성 값은 새로운 크기의 행렬을 형성하고, 이를 특징 맵(feature map)이라고 한다.

풀링 계층은 합성곱 계층의 결괏값에 대해 국소적인 부분을 하나의 대표적인 스칼라값으로 변환함으로써 특징 맵의 차원을 축소하는 역할을 하는 계층이다. 이는 CNN 모델의 학습 시간을 절약하고, 과적합(overfitting)의 가능성을 낮춰줄 수 있다.

여러 차례의 합성곱 계층과 풀링 계층을 거쳐 최종적으로 만들어진 특징 맵은 완전연결 망의 입력으로 들어가게 되고, 완전연결 망에서는 분류(classification) 작업을 수행하게 된다.

2.2.2 RNN

RNN은 시간 단계(timestep)마다 입력을 받아들이고, 현재 시점에서의 판단에 과거 시점의 판단을 고려함으로써 시계열 데이터와 같이 시간의 흐름에 따라 변화하는 데이터를 학습하는 인공신경망이다. LSTM(Long-Short Term Memory)[13]은 RNN의 은닉층에서 사용될 수 있는 셀 유형(cell type) 중 하나로, RNN 모델의 학습이 진행됨에 있

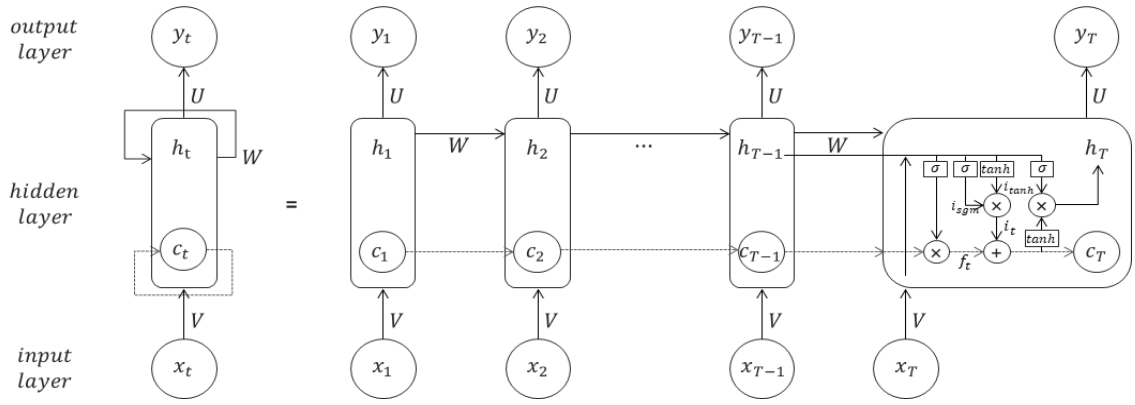


Fig. 2. Structure of RNN, consisting of input layer, hidden layer with LSTM Cell, and output layer

어서, 이전 시간 단계에서 받은 입력층의 정보가 이후 시간 단계의 학습에 미치는 영향이 시간이 지남에 따라 점점 감소하다가 결국에는 사라져버리는 경사도 소멸(vanishing gradient) 현상을 해결한다.

RNN은 크게 입력층(input layer), 은닉층(hidden layer), 출력층(output layer)으로 구성돼 있고, LSTM은 망각 게이트(forget gate) 단계, 입력 게이트(input gate) 단계, 출력 게이트(output gate) 단계로 구성된다. 망각 게이트 단계는 LSTM의 첫 단계로, 과거의 셀 상태(cell state)로부터 어떤 정보만을 남길 것인지를 결정한다. 두 번째 단계인 입력 게이트 단계는 입력되는 새로운 정보 중 어떤 정보만을 상태 값에 저장할 것인지를 결정하는 단계이고, 마지막 단계인 출력 게이트 단계는 상태 값을 얼마나 활성화 값으로 보낼지를 결정하는 단계이다. 각 상태 값은 각 게이트 값의 조합으로 구성되고, 각 게이트 값은 현재 시간 단계의 입력값과 이전 시간 단계의 은닉층 활성화 값에 의해 결정된다.

LSTM 셀을 사용하는 RNN 구조는 Fig.2.와 같다. t는 시간 단계를 의미하며, x_t, y_t, h_t는 각각 t시간 단계에서의 입력값, 출력층의 활성화 값, 은닉층의 활성화 값이자 출력 게이트 값을 의미한다. V, W, U는 각각 입력층과 은닉층을 연결하는 가중치, 은닉층을 서로 연결하는 가중치, 은닉층과 출력층을 연결하는 가중치를 의미한다.

c_t는 t시간 단계에서의 새로운 셀 상태 값으로, LSTM 셀에서 은닉층의 활성화 값의 연산에 사용되는 내부적인 값이다. c_t는 시간이 흐름에 따라 유지되고 전파할 정보를 담고 있으며, h_t는 각 시간 단계에서 출력에 사용할 정보를 담고 있다. f_t, i_t는 각각 t시간 단계에서 은닉층의 망각 게이트 값, 입력 게이트 값을 의미한다. $i_{sgm}, i_{\tanh}$는 각각 입력 게이트에서 새로운 정보의 비율을 결정하는 값, 새로운 정보의 방향을 결정하는 값을 의미한다.

LSTM 셀을 사용하는 RNN에서 t시간 단계에서의 출력 게이트 값 h_t를 구하는 과정은 수식 (1)과 같다. 이때 각 게이트를 계산하는 데 사용된 가중치 V, W 와 편향 값(bias) b를 구별하기 위해, f, g를 아래 첨자로 붙였다.

$$\begin{aligned} f_t &= c_{t-1} \times \sigma(V_f x_t + W_f h_{t-1} + b_f) \\ i_t &= i_{sgm} + i_{\tanh} \\ c_t &= f_t + i_t \\ h_t &= \tanh(c_t) \times \sigma(V_g x_t + W_g h_{t-1} + b_g) \end{aligned} \quad (1)$$

출력층의 출력값 y_t를 구하는 과정은 수식 (2)와 같다. 이때 $g(\bullet)$는 출력층에서의 활성화 함수를 의미한다.

$$y_t = g(Uh_t + b) \quad (2)$$

III. 제안 방법

본 논문에서는 CNN과 RNN을 활용한 NoSQL 삽입 공격 탐지 모델을 제안한다. CNN이나 RNN을 단일 모델로 사용하여 시계열 데이터를 처리하는 많은 연구 결과가 제시됐으나, 최근 연구 결과[14]에서 두 모델을 함께 사용할 경우 모델의 성능이 향상됨을 보였다. 따라서 본 논문에서는 NoSQL 삽입

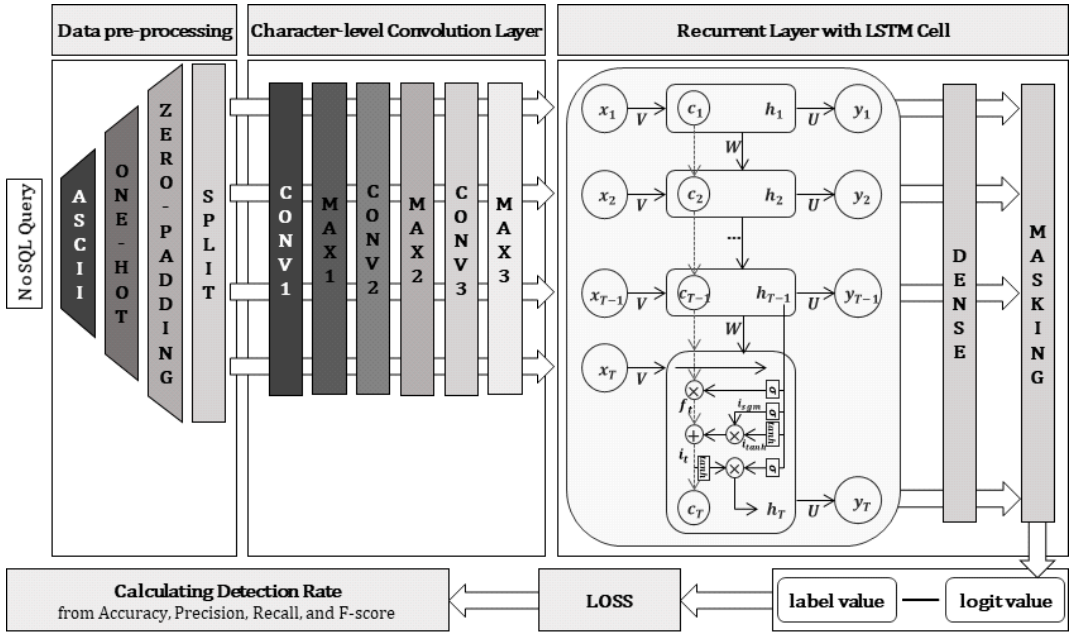


Fig. 3. The Structure of our Proposed Model, consisting of Data pre-processing, CNN, and RNN

공격 탐지 성능을 극대화하기 위해 CNN과 RNN의 융합 모델을 Fig.3.과 같이 설계했다.

제안되는 모델에서는 하나의 완전한 NoSQL 질의를 입력값으로 사용하며 이에 대한 출력값을 생성하므로, 일련의 질의가 입력값으로 전달되면 모델은 각각에 대한 탐지 결과를 만들어낸다. NoSQL 질의에 대해 일련의 전처리 과정(pre-processing)을 수행하고, 이로부터 얻은 데이터들로부터 Zhang [15]의 character-level CNN에서 사용하는 기법을 이용해 특징을 추출하고, 이 특징들을 순서에 맞게 RNN에 전달하여 NoSQL 삽입 질의를 탐지한다. 이때, RNN의 은닉층 활성화 유닛에는 LSTM을 사용한다.

3.1 데이터 전처리

character-level CNN으로의 입력을 위해 NoSQL 질의를 인코딩하여 행렬로 표현한다.

문자인 질의를 딥러닝 모델이 처리하기에 적합한 형태로 변환하기 위해 본 논문에서는 원핫 인코딩(one-hot encoding)을 사용했다. 질의의 각 문자를 0부터 127까지의 아스키 값(ASCII value)으로 변환하고, 변환된 아스키 값은 행렬에서 각각 하나의 행에 해당하며, 해당하는 아스키 값에 대해 원핫 인코딩하여 하나의 질의를 하나의 행렬로 표현한다. 이때 한 질의에 대한 행렬 표현은 '질의의 글자 수×128'의 크기를 갖는다.

질의의 최대 길이를 L이라고 하고, 행렬의 행의 길이를 동일하게 맞춰주기 위해 L만큼 영 패딩(zero-padding) 한다. 이에 모든 행렬은 '$L \times 128$'의 크기를 갖는다.

각 질의에 대한 영 패딩은 실제 데이터들에 더미(dummy) 값을 채워 넣는 것이므로, 더미 값이 분류 결과에 영향을 미치지 않도록 마스크(mask)를 이용하여 RNN에서 더미 값에 의해 만들어진 시간 단계의 출력을 제거한다. 마스크 사용을 위해 '$L \times 128$'행렬을 n개의 '$m \times 128$' 형태로 나누어준

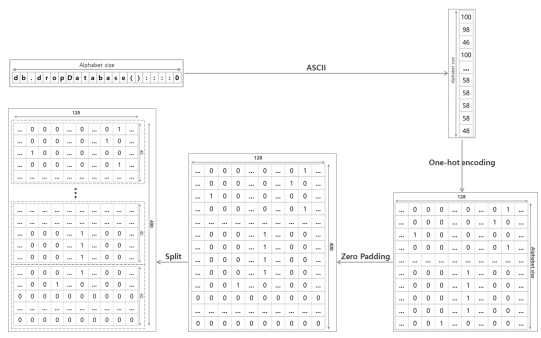


Fig. 4. An example of the Data pre-processing

다. 이때 n과 m은 '$n \times m = L$'을 충족하는 새로운 행렬의 개수 n과 새로운 행렬의 행의 크기 m을 의미한다.

Fig.4.는 제안 모델의 데이터 전처리 단계의 예시로, L은 400, 새로운 행렬의 개수 n은 10, 새로운 행렬의 행의 크기 m은 40이다.

3.2 character-level CNN을 활용한 특징 추출

본 논문에서 제안하는 모델은 특징 추출을 위해 character-level CNN을 활용한다. 이는 해당 모델이 텍스트 데이터의 특징 추출에 뛰어난 성능을 가진 모델임과 동시에, 입력값의 크기를 줄이고 희소한(sparse) 형태의 데이터를 밀도 있는(dense) 형태의 데이터로 만들기 위함이다. 또한, 질의 내에서 의미를 갖는 최소단위인 문자(character)를 모델 입력값의 최소단위로 하여, 삽입 질의의 바이트(byte) 단위의 특징을 민감하게 파악할 수 있도록 했다.

본 논문에서 사용한 character-level CNN의 구조는 Table 1.과 같으며, 활성화 함수로 ReLU(Rectified Linear Unit)[16]를 사용한다. character-level CNN은 원핫 인코딩된 문자로 구성된 텍스트 데이터를 처리하도록 설계됐기 때문에, 일반적인 이미지 처리에 사용되는 2차원 합성곱 연산 대신, 1차원 합성곱 연산을 사용한다.

각 질의의 블록은 컨볼루션 망을 거쳐 '$0.5^3m \times 16$'크기의 행렬이 된다. 이를 펼쳐서 '$j = 0.5^3m \times 16$'차원의 특징 벡터를 만든다.

Table 1. The Structure of Convolution Layers

Layer	Kernel	Stride	Filter	Output
input	-	-	-	$m \times 128$
conv1	3×128	1	64	$j \times 64$
max1	2	2	-	$0.5m \times 64$
conv2	3×64	1	32	$0.5m \times 32$
max2	2	2	-	$0.5^2m \times 32$
conv3	3×32	1	16	$0.5^2m \times 16$
max3	2	2	-	$0.5^3m \times 16$

3.3 RNN을 활용한 NoSQL 삽입 공격 탐지 모델

본 논문에서 제안하는 모델은 순차 데이터에 속하는 NoSQL 질의를 학습하고 분류하기 위해 은닉층 활성화 유닛에 LSTM을 사용하는 RNN을 활용했다. 이는 RNN 모델이 시간 단계마다 입력을 받아 현재 시점에서의 판단에 과거 시점의 판단을 고려함으로써 시계열 데이터를 학습한다는 특징을 반영하기 위함이다. 또한, LSTM 모델이 문자열의 특징 중 하나인 장기의존성(long-term dependency) 문제를 다룰 수 있는 모델이기 때문에 본 논문에 이를 활용했다.

단일 질의를 입력값으로 가지는 RNN은 a개의 은닉층을 가지며, 은닉층 마지막 시간 단계의 LSTM 셀 활성화 값들은 완전연결 계층의 입력값으로 사용돼 스칼라 로짓 값(logit value)으로 변환된다. 이 값과 마스크값의 곱들을 모두 더해 탐지 모델의 최종 스칼라 출력값을 계산한다. 이때 마스크값은 미니배치(mini-batch) 내의 패딩 된 각 질의가 마지막으로 나타난 시간 단계에는 1, 나머지 시간 단계에는 0으로 채워진 벡터이다. 본 논문에서는 모델을 학습하기 위해 손실 함수로 앞서 생성된 출력값과 라벨 사이의 교차 엔트로피(cross entropy)를 사용하였으며, 최적화 알고리즘으로 Adam[17]을 사용했다.

Fig.5.는 제안 모델의 학습 단계의 예시이다.

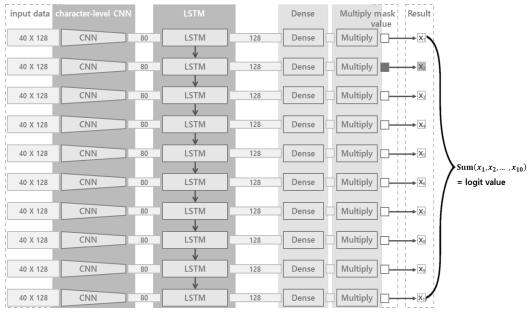


Fig. 5. An Example of the Learning Model

IV. 실험 및 평가

이 장에서는 3장에서 제안한 NoSQL 삽입 공격 탐지 모델의 성능을 측정하고, 측정 결과를 기존 연구 결과와 비교 및 제시한다.

비관계형 데이터베이스는 다양한 데이터 모델을 지원하는데, 가장 일반적인 데이터 모델은 열 기반, 문서 기반, 키 값 매핑 기반, 그래프 기반 모델이다. 본 논문에서는 문서 기반 데이터 모델을 지향하는 대표적인 비관계형 데이터 베이스 솔루션인 MongoDB와 CouchDB의 NoSQL 질의를 대상으로 실험

Table 2. Experiment Environment

OS	Ubuntu 18.04.4
CPU	AMD Ryzen 7 1800X Eight-Core Processor
GPU	NVIDIA GeForce GTX 1080 Ti
RAM	64GB
TensorFlow	2.1.0

을 진행했다.

본 논문에서 제시하는 모델의 학습을 위해 사용한 시스템 환경은 Table 2.와 같다.

4.1 실험 데이터

실험에 사용한 데이터 세트는 Islam[10]의 연구에 사용했던 크로스오버(cross-over) 및 뮤테이션(mutation)를 적용하여 데이터 세트를 오버샘플링(over-sampling)한 데이터 세트를 제공 받아 이를 대상으로 실험을 진행하였고, 그 형태는 Table 3.과 Table 4.와 같다. Islam[10]의 연구에 의하면, 이 데이터 세트는 MongoDB와 CouchDB의 NoSQL 질의에 대해 검증이 완료됨과 동시에, 라벨이 정해진 최초의 데이터 세트로, 본 논문이 제안하는 NoSQL 삽입 공격 탐지 모델의 실험 데이터로 적합하다.

MongoDB 데이터 세트는 203개의 삽입 질의를 포함해 총 1,004개의 질의를 포함하고 있으며, 그중 803개의 질의는 학습을 위해 사용됐고, 나머지 201개의 질의는 성능 평가를 위해 사용됐다.

CouchDB 데이터 세트는 50개의 삽입 질의를 포함해 총 350개의 질의를 포함하고 있으며, 그중 280개의 질의는 학습을 위해 사용됐고, 나머지 70개의 질의는 성능 평가를 위해 사용됐다.

Table 5.와 같이 MongoDB와 CouchDB의 삽입 질의는 총 253개로, 그중 PHP Array 삽입 질의는 134개, NoSQL OR 삽입 질의는 20개, JavaScript 기반 삽입 질의는 43개, Piggybacked 질의는 56개이다.

Table 3. MongoDB Dataset Statistics

	MongoDB		
	Benign	Injection	Total
Training	631	172	803
Test	170	31	201
Total	801	203	1004

Table 4. CouchDB Dataset Statistics

	CouchDB		
	Benign	Injection	Total
Training	240	40	280
Test	60	10	70
Total	300	50	350

Table 5. Statistics of Injection Queries Dataset

Injection Type	# of Samples
PHP Array injection	134
NoSQL OR injection	20
JavaScript based injection	43
Piggybacked queries	56
Total	253

4.2 실험 구성

Islam[10]의 연구에 사용됐던 MongoDB와 CouchDB의 NoSQL 질의를 대상으로 데이터 전처리 후, NoSQL 삽입 공격 탐지를 수행했다.

데이터 전처리는 Fig.4.와 같이 진행됐다. 이때 질의의 최대 길이인 L은 400, 새로운 행렬의 개수 n은 10, 새로운 행렬의 행의 크기 m은 40이다. 따라서 하나의 질의는 '40×128'크기를 가진 10개의 행렬로 변환됐다.

변환된 데이터에 대해 특징 추출과 NoSQL 삽입 공격 탐지는 Fig.5.와 같이 진행됐다. 각 질의의 블록은 character-level CNN을 거치며 '5×16' ($\because m=40$)의 행렬로 변환돼 특징 벡터의 차원 j는 80이다. 이 80차원의 특징 벡터는 a가 1인 1개의 은닉층을 포함하는 RNN을 거치며 128차원 벡터값이 된다. 미니배치 크기는 256, 학습률(learning rate)은 0.01, 에폭(epoch)은 100이다.

4.3 실험 결과

제안하는 모델의 성능을 측정하기 위해 정확도(accuracy), 정밀도(precision), 재현율(recall), F-score를 활용한다. 정확도는 전체 예측값에 대해 올바르게 예측한 비율을, 정밀도는 참이라고 예측한 것 중에서 실제로 참인 것에 대한 비율을, 재현율은 실제 참인 것 중에서 참이라고 예측한 것에 대한 비

Table 6. Comparison Results for Different Methods via Performance Measures

Dataset	Reference	Classifier	Accuracy	Precision	Recall	F_2 Score
MongoDB	Islam[10]	Decision Tree	91.6642%	93.4370%	92.6929%	0.932872
		Random Forest	91.8772%	93.5465%	92.9375%	0.932460
		AdaBoost	91.7880%	93.4722%	92.8735%	0.933518
		Neural Network	91.8772%	93.5537%	92.9392%	0.934302
		SVM	89.4552%	91.0479%	91.5189%	0.910000
		k Nearest Neighbor	91.6196%	93.3030%	92.7668%	0.931952
		XGBoost	89.5101%	90.7910%	87.9179%	0.884429
	our method	**CRNN**	**98.4849%**	**97.0588%**	**94.2857%**	**0.948576**
CouchDB	Islam[10]	Decision Tree	88.3333%	90.7801%	85.3333%	0.896358
		Random Forest	88.5666%	90.8256%	85.8000%	0.897641
		AdaBoost	88.6333%	90.8386%	85.9333%	0.898132
		Neural Network	88.6667%	90.8451%	86.0000%	0.898328
		SVM	85.2000%	84.6685%	85.9667%	0.849250
		k Nearest Neighbor	88.6667%	90.8451%	86.0000%	0.898328
		XGBoost	85.3600%	85.0000%	84.0600%	0.842463
	our method	**CRNN**	**98.5507%**	**90.9091%**	**100.000%**	**0.980392**

율을 나타낸다. F-score는 수식 (3)과 같이 정밀도와 재현율을 이용한 지표로, β값에 따라 F_1-score, F_2-score 등으로 분류된다. 본 논문에서는 F-score의 β값이 2인 F_2-score를 사용했다.

$$F = \frac{(\beta^2+1)\times(\mathrm{Precision}\times Recall)}{(\beta^2\times \mathrm{Precision})+Recall} \qquad (3)$$

Table 6.은 정확도, 정밀도, 재현율, F_2-score 지표에 대해 Islam[10] 연구의 탐지 모델의 성능과 본 논문의 탐지 모델 성능을 비교한 표이다. Islam[10]의 연구 결과를 비교 대상으로 한 이유는, Islam[10]의 연구가 NoSQL 삽입 공격 탐지를 위해 지도학습을 사용한 모델 중에서 탐지율이 가장 높은 모델이기 때문이다. Table 6.에서 확인할 수 있듯이, 모든 지표에 대해 제안된 NoSQL 삽입 질의 탐지 모델이 이전 탐지 모델보다 더 높은 탐지율을 보인다.

본 논문에서 제안하는 모델은 입력값의 최소단위를 단어(word)가 아닌 문자(character) 단위로 인식하는 모델인 character-level CNN을 활용했다. 글자 단위의 인식은 근본적인 언어 구조를 파악할 수 있게 하고, 이는 입력되는 NoSQL 질의의 특징 추출 성능을 높였다. 또한, 시계열 데이터 학습에 적합한 RNN을 활용함과 동시에, RNN의 은닉층에 LSTM 셀을 사용해 장기 의존성 문제를 해결했다. 이는 시계열 데이터의 특징을 가진 NoSQL 질의에 대한 학습률을 높여, 기존 다른 모델의 탐지율보다 높은 탐지율을 보였다.

V. 결 론

본 논문에서는 CNN을 이용해 특징을 추출하고, RNN을 활용해 NoSQL 삽입 공격 탐지 모델을 제시했다.

character-level CNN과 LSTM을 활용한 RNN이 결합된 딥러닝 기반의 분류 모델은 NoSQL 질의와 이 질의의 순서로부터 특징을 추출하여 NoSQL 삽입 질의를 탐지한다. Islam[10]의 연구에서 사용된 NoSQL 질의에 대한 실험을 통해 기존의 모델보다 탐지율이 더 높음을 입증했다.

현재 실험에 사용된 데이터 세트는 이미 알려진 NoSQL 삽입 공격 질의로 구성돼 있으며, 탐지 모델의 학습 및 테스트에 이 데이터 세트를 사용했다. 따라서 실험 결과에서 보듯이, 제안하는 모델은 기존에 공개된 공격 유형에 대해서는 높은 정확도로 탐지할 수 있다. 하지만 알려지지 않은 공격 유형에 대해서는 현재 데이터 세트로는 탐지 성능 측정이 어려우

며, 준 지도학습(semi-supervised learning)이나 추가적인 특징 공학 연구가 고려돼야 한다.

본 논문과 관련해, 경사도 소멸 문제를 해결하기 위한 기법인 어텐션(attention) 활용 등 성능 및 활용도가 높은 딥러닝을 활용해, NoSQL 삽입 공격을 탐지할 수 있는 효율적인 모델에 관한 연구가 가능하다. 또한, 실험에 사용한 데이터가 NoSQL 삽입 공격 질의의 특징을 모두 반영하기에는 적은 양이기 때문에 데이터를 확보할 필요가 있으며, 이때 데이터 증강(data augmentation) 등의 기법을 고려하는 향후 연구를 진행할 수 있다. 뿐만 아니라, 원핫 인코딩 처리로 인해 증가된 데이터 처리량을 감소시킬 수 있는 기법 연구 또한 가능하다.

References

[1] MongoDB, "mongodb all customers," https://www.Mongodb.com/who-uses-Mongodb, 20200302

[2] CWE, "CWE VIEW: Weaknesses in the 2019 CWE Top 25 Most Dangerous Software Errors," https://cwe.mitre.org/data/definitions/1200.html, 20200302

[3] OWASP, "OWASP Top10," https://owasp.org/www-project-top-ten/, 20200228

[4] NIST, "CVE-2018-1784 Detail," https://nvd.nist.gov/vuln/detail/CVE-2018-1784#vulnCurrentDescriptionTitle, 20200228

[5] NIST, "CVE-2019-9039 Detail," https://nvd.nist.gov/vuln/detail/CVE-2019-9039, 20200228

[6] A. Ron, A. Shulman-Peleg, and E. Bronshtein, "No SQL, no injection?," Proceedings of the 36th IEEE Symposium on Web 2.0 Security and Privacy, May. 2015.

[7] B. Hou, K. Qian, L. Li, Y. Shi, L. Tao, and J. Liu, "MongoDB NoSQL Injection Analysis and Detection," Proceedings of the 3rd IEEE International Conference on Cyber Security and Cloud Computing, pp. 75-78, Aug. 2016.

[8] A. M. Eassa, O. H. Al-Tarawneh, and A. S. Salama, "Nosql racket: A testing tool for detecting nosql injection attacks in web applications," International Journal of Advanced Computer Science and Applications, 8(11), pp.614-622, Nov. 2017.

[9] Wired Business Media, "Thousands of MongoDB Databases Found Exposed on the Internet," https://www.securityweek.com/thousands-Mongodb-databases-found-exposed-internet, 20200228

[10] M. R. Ul Islam, M. S. Islam, Z. Ahmed, A. Iqbal, and R. Shahriyar, "Automatic Detection of NoSQL Injection Using Supervised Learning," Proceedings of the 43rd IEEE Annual Computer Software and Applications Conference, pp. 760-769, July. 2019.

[11] Y. Bengio, A. Courville, and P. Vincent, "Representation Learning: A Review and New Perspectives," IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(8), pp.1798-1828, Aug. 2013.

[12] S. Albelwi and A. Mahmood, "A framework for designing the architectures of deep convolutional neural networks," Entropy, 19(6), pp. 242-263, May. 2017.

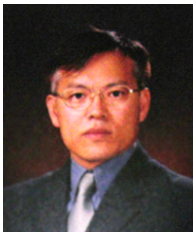
[13] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with LSTM," Proceedings of the 9th International Conference on Artificial Neural Networks, pp. 850-855, Sep. 1999.

[14] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," IEEE transactions on pattern

analysis and machine intelligence, 39(11), pp. 2298-2304, Dec. 2016
[15] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networ-ks for text classification," Advances in Neural Information Processing Systems 28, pp. 649-657, 2015.
[16] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," arXiv preprint arXiv:1505.00853, 2015
[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

〈저자소개〉

서 정 은 (Jeong-eun Seo) 학생회원
2017년 2월: 고려대학교 경영정보학과 학사
2018 9월~현재: 고려대학교 정보보호학과 석사과정
〈관심분야〉 정보보호, 개인정보, DB보안



문 종 섭 (Jong-sub Moon) 종신회원
1981년 2월: 서울대학교 계산통계학과 학사
1983년 2월: 서울대학교 계산통계학과 석사
1991년 2월: Illinois Institute of Technology 전산학과 박사
1993년 3월~현재: 고려대학교 전자 및 정보공학부 교수
2001년 2월~현재: 고려대학교 정보보호대학원 겸임교수
〈관심분야〉 정보보호, 운영체제, 침입탐지