Information Retrieval System for Malwares

Anonymous Author(s)

ABSTRACT

In this paper, we describe the problems that arise when building a malware retrieval system using deep learning, and how to solve them. Multilabel Weighted Centerloss(MWC) is proposed to learn semantic distances between malware samples. Also we visualize the embedding vectors of malware samples and show the retrieval results to prove the synchronization between our perception of malwares and learned embedding space. [**?**]

CCS CONCEPTS

• **Security and privacy → Malware and its mitigation**;

KEYWORDS

Malware, Information Retrieval, Neural Network

1 INTRODUCTION

하루에 30만개 이상의 멀웨어가 생성되고 있고 이 숫자는 계속 늘어나고 있다. 같은 기능을 하지만 해시는 다른 멀웨어들을 자동으로 제너레이션 하는 멀웨어 obfuscation 툴들의 발전 및 배포가 그 이유이다.

반면 보안 전문가의 시간은 늘어나지 않고 있다. 보안 전문가는 크게 두 가지 일을 하는데, 하나는 해시로 걸러지지 않은 악성코드의 카테고리를 분류하는 일이고 두 번째는 주요 악성코드에 deep dive 해서 obfuscation 방법이나 악성 행위 방법을 공부하고 후에 나올 악성코드에 잘 대응할 수 있게 하는 것이다.

이 두 가지 주요 업무 중에서 머신러닝이 쉽게 도움을 줄 수 있는 부분은 두 번째 태스크이다. 사실상 수많은 new 악성코드들을 보안 전문가들이 하나하나 정적 혹은 동적 분석을 통해 분류하는 일은 불가능하다. 따라서 자동으로 멀웨어를 분류하는 시스템은 예로부터 많이 만들었다. PE 스트럭쳐나 콜그래프 등의 정적 피쳐를 받아서 SVM, RF, DNN 등으로 멀웨어를 분류하는 모델이 연구되어왔다.

하지만 이렇게 만들어진 분류기들에는 문제가 있다.그 시스템들은 학습된 샘플들에 대해서는 좋은 정확도를 보이지만 새로운 샘플에 대한 대응이 좋지 못하다. 소분류를 레이블로 주고 가르치면 종류가 너무 많고 계속 추가되는 소분류가 지속해서 생긴다. 대분류를 레이블로 주고 가르치면 정적 피쳐 기준 이너클래스 베리언트가 너무 커지고 레이블의 모호성이 발생한다. 따라서 보안 전문가는 멀웨어 분류를 위해 학습된 머신러닝 모델을 사용하고 나서 그 결과를 확인해야하는 수고를 해야한다. 그리고 그 확인 과정은 비슷한 멀웨어들을 검색해주는 시스템이 구축되어있지 않다면 오래 걸리게 된다.

따라서 classification model 을 검증하는데에 사용할 수 있으면서도 동시에 위에서 언급한 보안 전문가의 두번째 태스크인 deep dive 에도 도움을 줄 수 있는 멀웨어 IR System 에 대한 니즈가 증가하고 있다.

기존 멀웨어 IR system 은 크게 LSH나 imphash와 같은 handcraft Feature representation 을 사용하는 시스템과, ML을 트레이닝시키며 얻은 automatically extracted feature representation 을 사용하는 시스템이 있다. 전자의 장점은 False Negative 가 낮다는 것이지만, generalization 이 잘 안된다는 단점이 있다.[] 더 좋은 generalization 을 위해 후자의 기법들이 개발되었지만, 멀웨어 representation location 이 사람의 시멘틱 스페이스와 어긋나있다는 단점이 IR 결과의 정성 평가 결과(?)를 낮추고 있다.

우리는 이러한 문제들을 해결하고 사람의 인지 스페이스와 멀웨어 임베딩 스페이스가 일치하는 멀웨어 IR 시스템을 제안한다. 또한 우리는 멀웨어 인포메이션 리트리벌 시스템을 만들 때의 어려운 점들과 그 문제들을 어떻게 해결했는지를 설명한다. 그리고 쿼링 결과를 시각적으로 보여서 우리의 인지와 임베딩 스페이스가 얼마나 일치하는지 확인한다.

우리의 메인 컨트리부션은 이렇게 요약할 수 있다.

- Centerloss 를 보완하여 멀웨어 IR 시스템이 더 잘 작동하게 하는 MWC 로스를 제안하였다.
- 멀웨어 딥러닝 based IR 시스템이 어떤 구조를 가져야 하는지 정의했다.
- 멀웨어 IR 시스템이 가지면 좋은 속성들을 정의하고 어떻게 우리의 시스템이 그 속성들을 갖게 만드는지 설명하였다.

페이퍼 구성은 다음과 같다. related work 가 섹션 2에 위치한다. 딥러닝, IR, Metric learning 등 주요 배경 지식에 대한 설명이 섹션 3에 위치한다. 섹션 4은 우리가 풀고자하는 문제 정의와 그에 사용되는 노테이션들을 정의한다. 섹션 5에서는 어떤 멀웨어 IR 시스템이 좋은 시스템인지 그 속성들을 나열하고, 섹션 6은 우리가 제안하는 malware IR 시스템의 디자인과 구현 과정을 설명한다. 구현 과정에는 크게 두 페이즈가 있는데 첫 째는 섹션 7에서 설명할 임베딩 학습 페이즈이고 둘 째는 섹션 8에서 설명할 멀웨어 리트리벌 페이즈이다. 섹션 9에서는 실험에 쓰인 데이터셋과 뉴럴넷 하이퍼파라미터 설정들, 그리고 실험 결과에 대해서 설명하고 임베딩 벡터를 시각화한다. 섹션 9에서는 Future works 그리고 세션 10 에서는 결과 및 discussion 이 위치한다.

2 BACKGROUND

2.1 Deep learning

neural networks는 훈련 데이터의 통계적 특성을 기반으로 특징을 추출할 수 있는 machine learning 방법이다. 또한 DNN은 network를 수 층의 hidden layers로 구성함으로써 hierarchical feature representation을 얻을 수 있다. 일반적으로 DNN은 input layer와 수 개의 hidden layer, output layer로 구성된다. Classification task의 경우, input layer는 task의 목적이 되는 sample의 feature vector representation을 입력으로 받는다. Output layer는 주어진 input vector에 관계된 class의 probability를 vector로 출력한다. 이를 통해 DNN은 input vector에 따른 class를 predict 할 수 있다.

특히 Image classification task의 경우 Convolutional network를 사용함으로써 모델의 성능을 향상시킬 수 있다.[ref?] Convolutional network는 convolutional layer와 pooling layer, 일반적인 neural network layer로 구성되며, convolutional layer의 경우 filter를 input에 대해 convoluting함으로써 feature를 추출한다. Pooling layer의 경우 영향을 적게 주는 feature를 down sampling하는 효과를 얻는다.

2.2 Information Retrieval

Information retrieval 은 본래 유저의 string 쿼리로부터 관련 문서를 찾는 태스크로 자연어 처리 분야에서 발전하였다. 하지만 이를 다른 도메인으로 확장시켜 이미지[? ?], 음악[?], 의료[? ?], 멀웨어[?] 등 다양한 분야에서 사용되고 있다.

Boolean Model. Information Retrieval System 을 만들기위한 모델로, Boolean model 이 제안되었다. boolean model 은 boolean 로직과 집합 이론을 배경으로 하여 문서들과 유저의 쿼리들을 집합으로 대하고 검색을 시행하는 모델이다. 이는 직관적이고 구현하기 쉽다는 장점이 있었지만, 유사도가 Discrete 하게 계산되어 리트리벌 결과가 아주 많아지거나 정확히 일치하지 않으면 검색되지 않는 현상이 일어나고, 모든 term 들이 똑같은 비중을 가지는 한계들이 있었다.[? ?]

Vector Space Model. 또 다른 방법으로 Vector space model 이 있다. 이는 문서들과 유저의 쿼리를 모두 벡터스페이스에서 표현함으로써 벡터 간 거리가 곧 유사도가 되도록 하는 모델이다. 선형 대수에 기반한 간단한 모델로, 팀에 대한 가중치를 다르게 줌으로써 리트리벌 결과를 더 좋게 만들 수 있고, 쿼리의 유사도를 continuous Value 로 나타낼 수 있다는 장점이 있다.[?] 문서와 유저의 쿼리를 벡터로 만드는 방법[]과 벡터 간 거리를 측정하는 방법[]에 따라 여러가지 모델들이 존재한다.

Learn Vector Representation Using Deep Learning. 앞서 설명했듯, 딥러닝을 사용하여 머신러닝 태스크를 수행하면 hierarchical representation 을 배울 수 있기 때문에 Generalization 이 더 잘 되는 장점이 있다. 따라서 Vector space model 을 이용하는 여러 태스크들과 딥러닝은 조합이 좋았고, 이로인해 많은 태스크에서 성능 상 breakthrough 를 이루어냈다.[refs : imagenet, word2vec, speech recognition, ...]. 더불어 Vector space model 을 사용하는 IR 분야 또한 딥러닝과 만나면서 한번 더 Breakthrough를 이루어냈다.[refs : document retrieval, Retrieval based QA, Image retrieval].

벡터 representation 을 학습시키는 방법으로는 supervised 와 unsupervised 가 있다. Unsupervised 로 가르치는 대표적인 방법은 Autoencoder를 이용하는 것으로, 이를 활용해서 멀웨어를 인코딩하려는 시도[ref]가 있었다. Supervised 로 가르치는 방법은 인간이 레이블을 주고 그 레이블로의 Classification 혹은 Regression 태스크를 수행하는 딥러닝 모델을 만들어, 정보를 가장 압축적으로 담고있는 bottleneck layer 의 아웃풋을 해당 샘플의 representation vector 로 보는 것이다. 이 방식은 vector representation 의 위치를 가르치는 사람이 어느정도 컨트롤 할 수 있게 된다. 예컨데 메트릭 러닝을 위한 objective function 을 사용해서 어떤 샘플들을 가깝게 보내고 어떤 샘플들을 멀게 보낼지를 결정할 수 있고[siamese, triplet, centerloss ...], 이는 one shot or few-shot learning 의 태스크를 수행할 수 있기 때문에, 벡터 간 distance 를 기준으로 랭킹을 결정하고 적은 샘플에도 대응할 수 있어야 하는 information retrieval 의 성능을 향상시킬 수 있다.

2.3 Metric Learning

2.4 Semantic Spaces

3 MALWARE INFORMATION RETRIEVAL SYSTEM

이번 세션에서는 제안된 malware information retreival system 의 동작과 구성 그리고 가져야하는 속성들을 설명한다. 우리의 시스템은 크게 학습과 검색(retrieval), 두 페이즈로 나뉜다. 학습 페이즈에서는 원할한 검색을 위해 시멘틱스를 반영하는 학습 데이터의 표현 벡터를 학습한다. 검색 페이즈에서는 유저로부터 입력된 멀웨어 혹은 태그 쿼리와 미리 저장해놓은, 학습된 표현 벡터들로부터 랭킹 결과를 출력한다. 이어지는 서브 섹션에서는 두 페이즈에서 사용될 노테이션들을 정의하고, 이들을 이용해서 각 페이즈에서 어떤 태스크를 행해야하는지를 자세히 설명한다.

Table 1: Notations

Meaning	Notation
Set of Malwares	X
ith malware sample	x_i
Set of single labels	Y_s
Single label of ith malware	y_{si}
Extracted features of ith malware	v_i
Set of extracted features	V
Set of Multilabels	Y_m
Multilabels of ith malware	$\vec{y}_{mi}$
Neural embedder	h
Embedded malware vectors	E
Distance measuring function	d
Ranking Module	R

Source: This is a table sourcenote. This is a table sourcenote. This is a table sourcenote.
Note: This is a table footnote.

3.1 Notations

우리는 먼저 학습 멀웨어 셋으로 $X = \{x_1, x_2, ..., x_N\}$ 를 갖고있다. 그리고 각 멀웨어에 해당하는 싱글레이블 셋 $Y_s = \{y_{s1}, y_{s2}, ..., y_{sN}\}$ 과 멀티레이블 셋 $Y_m = \{\vec{y}_{m1}, \vec{y}_{m2}, ..., \vec{y}_{mN}\}$ 이 있다. 이 레이블을 얻는 방법은 섹션 6.1에서 자세히 설명한다. 그리고 각 멀웨어로부터 손으로 추출된 피쳐들의 셋을 $V = \{v_1, v_2, ..., v_n\}$ 로 정의한다. 어떤 피쳐들을 추출해서 사용했는지는 섹션 6.2에서 설명한다.

멀웨어 IR 시스템은 $[h, E, d, R]$ 4개의 원소를 갖는 tuple 로 정의된다. h 는 멀웨어 피쳐로부터 벡터 representation 을 얻을 수 있는 임베더 함수이다. E 는 검색이 될 멀웨어의 임베딩 벡터 셋이다. d 는 입력된 쿼리와 임베딩 벡터 간 거리를 계산하는 함수이다. R 은 랭킹 모듈이다. 학습 페이즈에서는 적절한 h 를 학습하고 이를 통해 V로부터 E를 얻는다. 리트리벌 페이즈에서는 멀웨어 샘플 쿼리 q 를 입력받고 가장 가까운 k 개의 neighbor 를 랭킹 모듈 R 을 통해 반환한다.

3.2 Tasks

학습페이즈에서는 IR system 에서 사용하기 위한 벡터 리프레젠테이션을 얻기 위한 Auxiliary Task를 수행한다. Auxiliary Task 는 멀웨어 분류 태스크를 학습하는 것으로 싱글 레이블 클래시피케이션이나 멀티레이블 클래시피케이션이 될 수 있다. $f : V \rightarrow Y$ 인 f 는 멀웨어 피쳐로부터 레이블을 추정하는 함수이다. 함수 f 는 다시 g 와 h 의 합성 함수로 정의할 수 있다. h 는 위에서 설명한, 멀웨어 피쳐로부터 vector representation 을 얻을 수 있는 임베딩 함수이다. g는 임베딩으로부터 label 을 추정할 수 있는 클래시파이어이다.

$$f(v_i) = (g \circ h)(v_i) = g(h(v_i)) = g(e_i) = y_i$$

where

$$g(e_i) = \sigma(w * e_i + b)$$

리트리벌 페이즈에서는 멀웨어 샘플 쿼리 q를 입력받아 학습 페이즈에서 구했던 h 함수를 이용하여 임베딩한다. 임베딩된 쿼리 eq

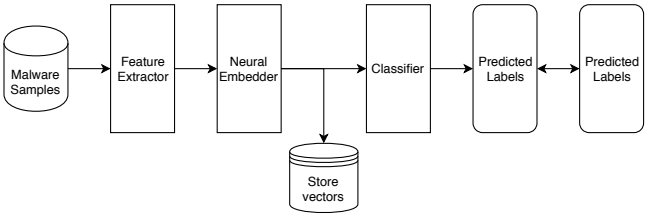

Figure 1: train phase

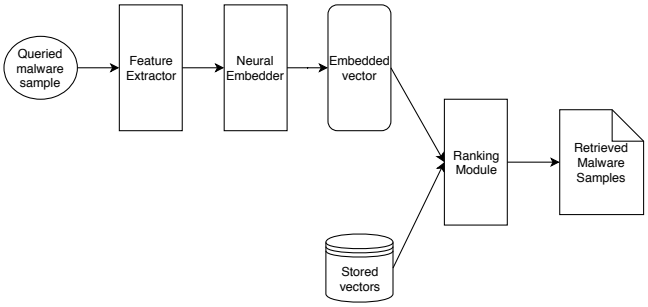

Figure 2: retrieval phase

와 E 의 원소들 간 거리를 d 함수를 통해 측정하고, 가까운 k 개의 neighbor 를 랭킹 모듈 R 을 통해 반환한다.

$$Results = \{x_j | j = argmin_i(-d(e_q, e_i))\}$$

3.3 Modules

두 페이즈의 구조도는 Figure 과 같다. 두 페이즈에서 공통으로 존재하는 모듈은 Feature Extractor와 Neural Embedder 가 있다. 벡터 학습 페이즈에는 Classifier 가, 리트리벌 페이즈에는 랭킹 모듈이 포함되어있다.

Feature Extractor. Feature extractor 는 Malware Rawdata 로부터 Handcrafted feature 를 추출하는 모듈이다. Handcrafted feature 로 PE 같은경우에는 Size 와 Entropy, Histogram of API Calls, … 등을[*] 추출한다. APK 같은 경우에는 추가적으로 Permission, … 등의 피쳐를 추출한다.

Neural Embedder. Neural embedder 는 Feature extractor 모듈에서 추출된 멀웨어의 피쳐들로부터 Representation vector 를 뽑는 모듈이다. theta 로 Parameterized 되어있는 뉴럴 네트워크이며, 파라미터는 벡터 학습 페이즈에서 Auxiliary Task 를 수행하면서 Optimize 된다. 그리고 리트리벌 페이즈에서 파라미터는 freeze 되어 업데이트되지 않는다. Neural Embeder 의 네트워크 구조는 Deep and wide 를 사용하였다. Thumbnail 을 받아서 CNN 5 층을 통과시킨 벡터와 나머지 feature 들을 받아서 fully 2 층을 통과시킨 벡터를 concatenate 하고, fully 2 층을 추가로 쌓은 구조를 사용한다. 이렇게 함으로써 우리가 얻을 수 있는 효과는 feature 의 성격에 따라 다른 네트워크 구조로 임베딩 피쳐벡터를 추출함으로써 더 나은 generalization 성능을 얻을 수 있다는 것이다. thumbnail 피쳐는 [malimg 논문 인용] 악성 코드영역 혹은 악성코드를 특징짓는 영역의 로컬리 시프트 인베리언트한 특성을 담기 위해 CNN 을 사용해서 임베딩 피쳐벡터를 추출한다.

Classifier. Classifier 모듈은 Neural embedder 를 통과해서 나온

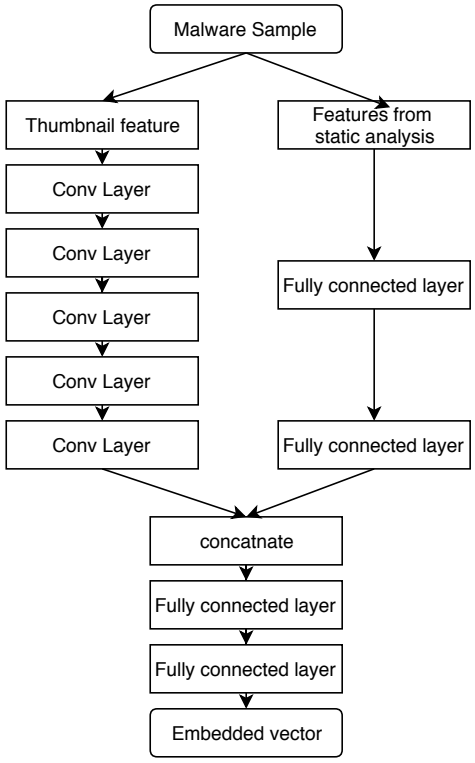

Figure 3: neural embedder

embedding vector 를 받아서 각 레이블 별 확률을 출력해준다. 1층의 fully connected 로 파라미터라이즈드 되어있으며 태스크에 따라 softmax 혹은 sigmoid Activation을 사용하였다. Neural Embeder 모듈과 마찬가지로 벡터 학습 페이즈에서 back propagation 에 의해 파라미터들이 학습되고, 리트리벌 페이즈에서는 프리즈된다.

Ranking Module. 학습 페이즈에 저장해두었던 학습 셋의 임베딩 벡터들과 추출된 벡터의 거리를 거리함수 d 를 사용하여 측정하고 k 개의 nearest neighborhood 를 가까운 순서대로 정렬하고 뱉어낸다.

3.4 desired properties

멀웨어 IR 시스템은 한계들이 존재했다. 특히 Raw binary file 로부터 좋은 피쳐를 뽑기가 수월하지 않고, intraclass variance 는 크고 innerclass variance 는 작으며 계속해서 변종과 새로운 종이 생겨나는 멀웨어 도메인의 특징으로 말미암아 좋은 멀웨어 IR 시스템이 가져야 할 속성들은 다른 도메인의 IR 시스템과는 조금 다르다.

Semantic understanding. 쿼링 샘플에 대해 구조적으로 비슷한 샘플 뿐만 아니라 의미적으로도 비슷한 샘플들을 랭킹하고 retrieve 할 수 있어야 한다. 여기서 의미가 비슷하는 것은 멀웨어의 행동 혹은 사람이 생각하기에 중요한 멀웨어의 속성들이 비슷하다는 것을 의미한다.

Robustness to novelty. 새로운 변종들은 계속해서 나타나기 때문에 이에 빠르게 대응할 수 있어야 한다.

Robustness to rare inputs. 멀웨어 도메인에서의 샘플 수는 그 멀웨어 패밀리의 영향력을 의미하지 않는다. 물론 비례하는 경향이 없는것은 아니지만, 하나의 멀웨어가 전세계적으로 유행할

수도 있고, 변종은 많지만 별로 영향력이 없는 멀웨어일 수도 있다. 따라서 해당 패밀리의 샘플 수가 적다고해서 쿼링 결과에서 랭킹 순위가 밀려서는 안된다. 즉 적은 수의 샘플에 대해서도 모델은 강건해야한다.

Robustness to polymorphism.

Robustness to variable size inputs. 멀웨어의 raw file size Variance 는 매우 크다. 작게는 몇키로바이트부터 크게는 기가단위까지 갈 수 있다. 사이즈에 상관없이 retrieve 를 할 수 있어야 한다.

Efficiency. 세상에 존재하는 멀웨어 샘플 개수는 너무나도 많고 기하급수적으로 늘어나고 있기도 하다. 따라서 수많은 샘플들에서 k 개의 상위 랭크된 결과를 적절한 시간 내에 retrieve 하려면 랭킹 모듈이 효율적이어야한다.

4 SEMANTICS AWARE REPRESENTATIONAL LEARNING

이번 섹션에서는 vector representation E 를 학습시키기 위한 방법들을 설명한다. E를 우리의 의도대로 임베딩시키기 위해 제안되었던 방법들과 각 방법 별 문제점, 그리고 해결하기 위한 방법들을 나열하였다. 그리고 우리가 설계한 최종 목적함수와 constraint 를 소개한다.

4.1 Semantic Spaces for Malwares

4.2 Solution Overview

Multilabel Classification. Singlelabel 멀웨어 샘플들의 vector representation 은 선형 classifier 로 구분가능한 공간에 위치하게 되고, 랭킹 모듈로 리트리벌할 수 있게 된다. 이 모델의 장점은 벡터 스페이스 모델을 사용함으로써 얻는 이점들이다. 특히 continuous similarity 를 사용한 랭킹이 가능해진다는 점이 이전 모델들에 비해 발전이라고 할 수 있다. 하지만 한계들도 많이 존재한다. 같은 레이블이 붙어있다고 해서 정말 의미적으로 가깝거나 다른 레이블이 붙어있다고 해서 정말 의미적으로 다른 샘플이라고 보장할 수 없는 멀웨어 레이블링의 모호성 때문에 싱글레이블만으로 임베딩을 하면 임베딩 공간과 우리의 멀웨어에 대한 인지 공간이 매우 상이하게 된다. single label classification 의 문제를 해결하기 위해 우리는 멀웨어에 멀티레이블링을 한 데이터셋을 멀티레이블 분류 태스크로 학습시켜, 샘플들 간 위치를 좀 더 의미적으로 유사하게 임베딩한다. 이렇게 했을 때, 같은 대분류 내의 소분류 클래스들에 해당하는 샘플들이 가까운 곳에 임베딩되게 된다. objective function 은 다음과 같다.

Centerloss. 하지만 이렇게 임베딩한 샘플들도 inner class variance 가 너무 커서 ranking module 에서 같은 클래스 내 두 샘플의 거리가 서로 다른 클래스 내의 두 샘플 간 거리보다 더 커지는 현상이 발생하게 된다. multilabel classification 에서의 문제를 해결하기 위해 우리는 메트릭 러닝을 위한 목적함수를 추가하여 최종 목적함수를 설계하였다. 센터로스[ref]를 변형한 목적함수를 통해 innerclass variance 를 낮추고 intraclass variance 를 높였다. 최적화할 식은 다음과 같다.

싱글레이블 센터로스를 제안했던 논문에서는 각 레이블 별로 템플릿 벡터를 메모리에 저장해놓고 그 템플릿 벡터와 보틀넥의 거리가 작아지도록 MSE 에러를 로스에 추가했다. 우리도 마찬가지로 각 레이블별로 템플릿 벡터를 메모리에 저장해놓고, 보틀넥이 정답 레이블 벡터들의 합이 되도록 MSE 에러를 로스에 추가했다. 그리고, 보틀넥과 정답 레이블벡터의 합의 차이에 alpha/the number of answer labels 만큼의 곱을 하여 각 레이블을 업데이트 해준다. 알파는 하이퍼파라미터로, 템플릿 벡터와 보틀넥을 얼마나 섞어서 기존 템플릿벡터를 업데이트할지를 결정해주는 값이다. 하지만 섹션 9의 결과에서 볼 수 있듯이, 아직 인간의 인지와 시멘틱 임베딩 공간이 일치하지는 않는다. 그 이유는 각 태그 별 중요도를 반영하여 임베딩하지 않았기 때문이다.

Weighted Centerloss. agent 나 downloader같은 태그들에 대한 innerclass variance 를 낮추는 것보다는 오히려 수는 적지만 중요한 태그들인 ransom, coinminer 등의 태그들의 innerclass variance 를 줄이는 것이 Ranking 할 때 더 의미가 있다. 따라서 태그들의 중요도를 constraint 로 추가한 최종 목적함수를 사용한다. 최적화 할 문제는 다음과 같다.

4.3 Proposed Objective Function: Multilabel Weighted Center Loss (MWC Loss)

$$\min_{\theta,w,b} J(\theta,w,b) = -\sum_i \sum_j y_{mij} \log \hat{y_{ij}} \tag{1}$$

s.t.

$$h(v_i;\theta)) - \sum_k c_{ik} < \epsilon,$$

$$||c_i||_2 = cc_i$$

for

$$i = \{1, 2, 3, ..., N\}$$

위의 constrained optimization problem 의 Lagrangian 은 다음과 같다.

$$\begin{aligned} L(\theta,w,b,\lambda_1,\lambda_2) = -\sum_i [\sum_j y_{mij} \log \hat{y_{ij}} \\ +\lambda_1 * (\sum_k c_{ik} - h(v_i;\theta))^2 + \lambda_2 * (||c_i||_2 - cc_i)^2] \end{aligned} \tag{2}$$

where

$$\hat{y_{ij}} = \frac{exp(w * h(v_i;\theta) + b)}{1 + exp(w * h(v_i;\theta) + b)}$$

실제로 학습할때 메모리에 새로운 레이블 템플릿 벡터를 다음의 방법으로 업데이트하여 두 번째 constraint를 강제로 만족시켰다.

$$c_i \leftarrow c_i + (1-\alpha) * (e_i - \sum_k c_{ik}) \tag{3}$$

The meaning of averaging center vectors.
The meaning of adding center vectors.

5 EVALUATION

이번 섹션에서는 우리가 제안하는 멀웨어 IR 시스템의 성능을 여러 방법으로 평가하여, 앞에서 우리의 시스템의 강점이라고 주장했던 semantic-aware malware retrieval 태스크를 수행하였는지 확인한다.

5.1 Datasets

모델의 학습과 평가에 사용된 dataset은 VirusTotal로부터 crawling 된 샘플 PE 5만여개와 APK 8만여개 중 security expert가 검수하여 확진을 내린 PE 샘플 2천여개 k종의 family, APK 샘플 2만여개 l종의 family로 구성하였다. 또한 VirusTotal에서 제공하는 수십개 이상의 AntiVirus 엔진의 탐지결과로부터 Labeling을 자동화하였다. VirusTotal로부터 각 AntiVirus 엔진들이 탐지해낸 결과를 의미있는 word 단위로 파싱하고, 샘플들로 부터 얻어진 word token들을 대상으로 security expert가 의미있는 토큰과 그렇지 않은 토큰을 선별하는 작업을 수행하였다. 추가로, 의미 있는 토큰에 대해 중요도에 따라 우선순위를 매기도록 하였다. 이를 통해 학습 샘플에 대해 expert가 일일히 수동으로 label을 다는 시간을 줄이면서도 높은 accuracy의 label을 얻을 수 있다. 또한, 하나의 샘플에 single label이 달릴 때에 비해 multi label을 사용 할 경우 악성코드의 다양한 행위 특성을 반영하도록 모델을 학습 시킬 수 있다.

5.2 Hyperparameters

5.3 Auxiliary Task

5.4 Querying Quantitative Test

표 : Performance measures of Malware IR Systems

metric performance measures: acc, auc, tf-idf of labels(?)

row : 기존 방법들, single, multi, weighted multi col : single-label classification accuracy, multilabel classification auc

non-metric performance measure : top query results

5.5 Querying Qualitative Test

Visualizations. semantic synchronization level Visualization of representations

Querying results sorted by distances.

5.6 Generalization

입력 Feature 가 Semantics-aware feature 일 수록 새로운 샘플에 대한 Error 가 더 작다. 우리가 제안한 목적함수는 멀티 레이블과 레이블 별 중요도로부터 해당 샘플이 시멘틱 스페이스에서 어디에 위치해야하는지를 가이드해준다. 따라서 Train Samples 에 대해서는 데이터의 입력 Feature가 Semantic-aware 하지 않더라도 원하는 곳에 특징 표현 벡터를 위치시킬 수 있다. 하지만 새로 보는 샘플의 representation vector 가 우리가 원하는 곳에 위치하게 된다는 보장은 없다. 딥러닝의 하이러키컬 특징 표현 학습이 MWC loss 를 통해 가이드 되는 Semantic을 학습할 수 있도록 하려면 입력 Feature가 충분히 Semantics-aware 해야 하고, 이에 관련된 연구들은 섹션2에서 소개하였다.

우리는 이 차이를 보이기 위해 semantics-aware level 이 다른 두 가지 피쳐에 대해 벡터 표현을 학습시키고, validation set 의 벡터 표현이 Semantic space 위에 잘 위치하는지 확인하는 실험을 한다. 첫 번째 실험에 사용되는 피쳐는 간단한 정적 분석을 통해 얻을 수 있는 Size, entropy[] 이고, 두 번째 실험에 사용되는 피쳐는보다 조금 더 Semantics-aware 한 피쳐인 Thumbnail[] 이다. 위에서 진행한 정량 평가와 정성 평가를 두 실험에 대해 진행해 본 결과 더 semantics-aware한 피쳐를 입력으로 사용할 수록 평가의 결과가 좋았고 이는 즉 Generalization 이 더 된다는 것을 의미한다. 즉, 누구라도 더 Semantics-aware한 피쳐를 입력으로 넣어줄 수 있다면, 간단히 Add-on 가능한 MWC loss 를 사용하여 훌륭한 Semantic space 를 구축할 수 있게 된다.

6 FUTURE WORKS

현재 방법은 멀티 레이블이 다 정확해야 한다. 아니라면 모델이 noise 로부터 오해를 쉽게 한다. 하지만 이 도메인에서 정확한 레이블을 얻기란 쉽지 않다. 따라서 label noise-robust MLC 를 만들고 멀웨어의 feature 로부터 멀웨어 샘플의 리얼 레이블을 추정하고, 이를 통해서 IR 시스템을 만들면 더 좋을 것 같다.

더불어 새로운 샘플에 대한 빠른 대응도 IR 시스템에서 중요한 부분인데, 딥러닝 모델이다보니 학습해야할 시간이 필요하다. continual learning 혹은 online update 를 사용해서 이를 해결하는 것도 우리의 Future work이다.

7 RELATED WORKS

Nataraj et al. 은 large scale malware search and retrieval system을 제안하였다. 이 논문에서는 malware image로 부터 fingerprints를 얻고, 이를 nearest neighbor search를 통해 비슷한 샘플을 retrieval 하는 방법을 제안하였다.

Upchurch et al. 은 similarity testing을 통해 variant malware를 탐지하는 framework를 소개했다. 이 framework는 BitShred, TLSH, sdhash, ssdeep 등의 방법으로 정적 feature를 추출하고 이를 비교함으로써 유사한 malware인지의 여부를 판단하였다.

Palahan et al. 은 system call dependency graph로부터 significant malicious behaviors를 추출하고 이를 비교함으로써 malware간의 similarity를 비교하는 방법을 제시했다.

Neural IR document retrieval domain에서 많은 neural IR 모델들이 제안되었고 그 중 많은 모델들이 text의 좋은 representation을 얻기 위해 연구되었다[1),2),3)].

Multilabel embedding Chih-Kuan et al. 은 multilabel embedding을 얻는 방법으로 label-correlation sensitive loss function을 사용하는 Canonical Correlated AutoEncoder 모델을 학습하였다. 논문에서는 dependency가 있는 label간의 관계를 End-to-End로 학습하고 이를 통해 Multi-label classification task를 해결하였고, 나아가 multilabel classification의 missing label 문제에 효과적임을 입증했다.

8 CONCLUSIONS

우리는 딥러닝을 활용하여 멀웨어 리트리벌 시스템을 만드는 데에서 생겼던 문제들을 새로운 로스를 정의함으로써 해결했다. 특히 기존 방법들보다 [제너럴라이제이션] 측면에서 좋다는 것을 정량적, 시각적으로 증명했다. 이는 수많은 보안 전문가들의 업무 코스트를 낮춰줄 수 있을 것이라 믿는다.

A LOCATION

Note that in the new ACM style, the Appendices come before the References.