Semantics-aware Representation Learning for Malware Information Retrieval

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, Semantic Space, Deep Learning

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 전문가의 업무 중 큰 부분을 차지하는 두 가지 일이 있다. 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 분류기의 정확도가 만족스럽지 못하거나 새로운 샘플에 대해 대응을 잘 하지 못한다면 보안 전문가는 멀웨어 분류를 위해 학습된 머신러닝 모델을 사용하고나서 그 결과를 확인해야하는 수고를 해야한다. 그리고 그 확인 과정은 비슷한 멀웨어들을 검색해주는 시스템이 구축되어있지 않다면 수고스럽고 오래 걸린다.

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 있다.

멀웨어 IR System 은 자연어 혹은 비젼의 IR System 과 마찬가지로 의미적으로 비슷한 멀웨어 샘플들을 랭킹하고, retrieve 해 줄 수 있는 시스템을 말한다. 따라서 우리는 멀웨어의 의미를 담을 수 있도록 멀웨어 IR 시스템을 구축해야한다.

시스템이 Semantics - aware 라는 말은, 사람이 생각하는 멀웨어 샘플들간의 의미론적 관계가 시스템에 담겨있다는 뜻이다. 구조적으로 다르다고 해도 멀웨어의 행위가 일치하면 같은 의미를 갖는 샘플이라고 볼 수 있다. 마찬가지로 구조적으로 거의 같다고 해도 멀웨어의 행위가 전혀 다르다면 두 샘플은 다른 의미를 가졌다고 할 수 있다. 또한 샘플들 간 의미 차이 까지 고려될 수 있다면 그 시스템은 Semantics-aware한 시스템이라고 할 수 있다.

Traditionally Semantic-aware Feature 를 사용해서 Semantics 를 담는 시스템을 구축하고자하는 시도가 있었다. 가장 멀웨어의 행위를 잘 나타낼 수 있는 동적 피쳐는 모든 경우의 수를 탐색하기 어렵기에 얻기가 어렵다는 단점이 있다. 정적 분석으로 얻을 수 있는 Syntatic 피쳐로 멀웨어 샘플의 의미를 표현하기 위해 보안 전문가가 피쳐 엔지니어링을 해서 특정 악성 행위의 공통적인 패턴을 조금 더 잘 검출하는 hand-crafted 피쳐를 획득할 수 있었지만, 복잡한 의미 관계까지 표현하는 머신러닝 모델을 만들기 위해서는 다양한 도메인에서 증명된 딥러닝같은 특징 표현 학습 기법을 멀웨어 도메인에 적용하는것을 고려해야한다.

딥러닝은 보안 전문가에 의해 피쳐 엔지니어링 된 피쳐들 뿐만 아니라, malware 바이너리, 소스코드, 메타데이터, 리소스, 동적 행위로그, 스크린샷 등 엔지니어링 되지 않은 인풋들에 대해서도 같은 의미를 갖는 멀웨어 샘플들을 표현하기 위한 공통 레이턴트 피쳐 표현을 하이러키컬하게 배울 수 있기에 Generalization 관점에서 이점이 있다. 또한 풍부한 Capacity 로 복잡한 의미관계를 학습할 수 있다는 장점도 있다.

그렇다면 좋은 인풋 feature 와 딥러닝의 레이턴트 피쳐 표현 학습 능력이 있다면 멀웨어의 의미 관계까지 배울 수 있는가? 자연어 도메인에서는 의미 관계를 학습하기 위해서 문장에서의 앞 뒤 관계를 활용하거나[word2vec] 컨텍스트에 기반하여 co-occurence 를 확률적으로 표현한 메트릭스를 활용하는[GloVe?] 등, Self- supervised 방법을 활용한다. 하지만 멀웨어는 self-supervised learning 을 할 만한 요소가 없기 때문에, supervised learning 을 통해 의미 관계를 학습시키는 방법을 사용해볼 수 있다. 지금까지 알려진 대부분의 Deep Learning 기반 Malware Classifier는 어떤 파일이 어떤 malware family로 명명되어 있는 지를 분류하도록 가르친다. 하지만 이를 학습시켜서 얻은 representation latent feature 들은 거리가 가까우면 거의 동일한 멀웨어이고 거리가 멀면 동일하지 않은 멀웨어라는 것을 알 수 있을 뿐, 거리의 차이가 의미의 차이를 표현해주지는 않는다.

그러면 어떻게 해야 멀웨어의 시멘틱을 딥러닝으로 학습할 수 있는가? 악성 프로퍼티는 여러개이고 []. 우리는 정제된 멀티 레이블을 제공하여 딥러닝 모델이 학습할 Semantics를 지도해주었다. 그리고 멀웨어의 표현 벡터를 시멘틱 컴퍼넌트 벡터들의 리니어 컴비내이션이 되도록 학습하여 특정 의미를 갖게 하였다. 또한 우리가 제안한 메트릭 러닝 방법을 통해, 벡터 스페이스에서의 멀웨어 샘플들 간 거리가 의미의 차이와 같아지도록 학습하였다. 그리하여 멀웨어 샘플의 의미(Semantics)를 벡터로 표현할 수 있는 멀웨어 시멘틱 스페이스를 디자인할 수 있었다.

이렇게 학습된 Semantic Space에서 의미적으로 유사한 멀웨어 또는 어떤 semantic의 차이를 갖는 malware, 혹은 특정 의미를 갖는 멀웨어들을 검색할 수 있는 유연한 IR system을 소개한다. 그리고 정량 평가와 정성 평가를 통해 시멘틱 스페이스가 얼마나 잘 만들어졌는지 확인한다.

우리의 메인 컨트리부션은 이렇게 요약할 수 있다.

- 우리는 악성코드의 Semantics를 벡터로 모델링하고 어떤 악성코드의 latent feature representation을 semantic components의 linear combination으로 표현할 수 있는 Semantics Space 위에 임베딩했다. 그리고 이를 학습하기 위한 Deep Learning 기반 Malware Classifier의 구조를 제안한다.
- 우리는 Malware Classifier가 학습하는 Semantic Space에서 metric function이 semantics의 차이를 학습하기 위해 multilabel centerloss에 기반한 Metric Learning 기법을 제안한다.
- 학습된 Semantic Space에 존재하는 멀웨어 샘플 벡터들에 대해 멀웨어, 시멘틱, 멀웨어와 시멘틱의 조합 세 가지 방법으로 malware를 검색할 수 있는 유연한 IR system을 소개하고 그 성능을 평가한다.

페이퍼 구성은 다음과 같다. 딥러닝, IR, Metric learning, Semantic Space 등 주요 배경 지식에 대한 설명하는 Background 가 섹션 2에 위치한다. 섹션 3는 멀웨어 IR 시스템에 대한 구조와 desired properties 를 정의한다. 섹션 4에서는 우리가 학습할 멀웨어 시멘틱 스페이스를 설명하고 이를 학습하기 위한 우리의 제안을 보여준다. 섹션 5에서는 실험에 쓰인 데이터셋과 뉴럴넷 하이퍼파라미터 설정들, 그리고 정량 평가와 정성 평가 실험 결과에 대해서 설명한다. 섹션 6에서는 related works 를, 섹션 7에서는 Future works 그리고 세션 8 에서는 conclusion 이 위치한다.

2 BACKGROUND

2.1 Deep learning

neural networks는 훈련 데이터의 통계적 특성을 기반으로 특징을 추출할 수 있는 machine learning 방법이다. 또한 DNN은 network 를 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 network 를 사용함으로써 모델의 성능을 향상시킬 수 있다.[ref?] Convolutional network는 convolutional layer와 pooling layer, 일반적인 neural network layer로 구성되며, convolutional layer의 경우 filter 를 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[1, 8], 음악[7], 의료[2, 4], 멀웨어[6] 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.[3?]

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, 텀에 대한 가중치를 다르게 줌으로써 리트리벌 결과를 더 좋게 만들 수 있고, 쿼리의 유사도를 continuous Value 로 나타낼 수 있다는 장점이 있다.[5] 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.

Learning to rank

2.3 Metric Learning

Metric Learning 은 거리 함수를 학습하는 기법이다. 좀 더 자세히는, 메트릭 러닝을 통해서 어떤 샘플들을 서로 가깝게 볼 것이고 어떤 샘플들을 멀다고 볼 것인지, 혹은 어느정도 다르면 거리가 어느정도 될지를 결정짓는 거리 함수를 학습시킬 수 있다.

딥러닝을 활용한 메트릭 러닝의 대표적인 예는 Siamese Network 이다. 샤미즈 네트워크는 입력 데이터로 동일하다고 볼 샘플의 쌍과 동일하지 않다고 볼 샘플의 쌍을 주고, 동일하다고 볼 샘플의 쌍에 대해서는 특징 표현 벡터의 거리 차가 0으로, 그렇지 않은 샘플의 쌍에 대해서는 거리 차가 1 이상 으로 나도록 가르친다.

또 다른 예시로 Centerloss 가 있다. 이는 각 Single label classification Task 에서, class 별 Template vector 를 만들고 같은 클래스에 속하는 샘플들의 Representation Vector 들이 해당 클래스의 template vector 와 같아지도록 하는 목적 함수를 기존 Cross-entropy 로스에 추가하는 방식으로 학습된다. 이를 통해 intra-class variance 를 낮추어 representation vector 들이 Retrieval와 같은 태스크에 사용될 수 있도록 한다.

이렇게 메트릭 러닝으로 학습된 거리 함수는 유사한 데이터를 찾는 여러 실생활 태스크에 사용되는데, 대표적으로 Face recognition 이나 image retrieval, Text retrieval 등이 있다.

2.4 Semantic Spaces for Natural Language

Semantic spaces 를 만드는 것은 본래 자연어 도메인에서, 의미를 capture 할 수 있는 자연어의 표현을 만드려는 목표를 갖는다. Semantic Spaces의 오리지널 모티베이션은 자연어의 두 가지 챌린지에 있는데 하나는 다른 의미이지만 음이 같은 동음이의어(ambiguity)를 구분 가능하게 표현하는 것이고 다른 하나는 생김새(Syntactic)는 다르지만 의미가 같은 용어들을 동일하게 표현하는(Vocabulary mismatch)것이다.

최근에 뉴럴넷이 다른 새 접근법들과 조합되면서 주된 발전을 이뤘는데, 이를테면 word2vec, glove, fastText 등이 있다. 이렇게 자연어들에 대한 표현을 Semantic spaces 에서 잘 만들어 놓으면, 이를 문서 분류, 문서 검색, Question and Answering, 음성인식, 번역 등 여러 태스크에 사용할 수 있다.

3 MALWARE INFORMATION RETRIEVAL SYSTEM

이번 세션에서는 제안된 malware information retrieval 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.

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. 시멘틱을 내포하는 임베딩 벡터를 학습시키는 방법에 대해서는 섹션4에서 자세히 이야기한다.

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

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.

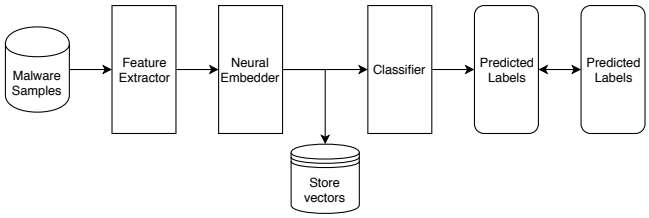

Figure 1: train phase

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 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

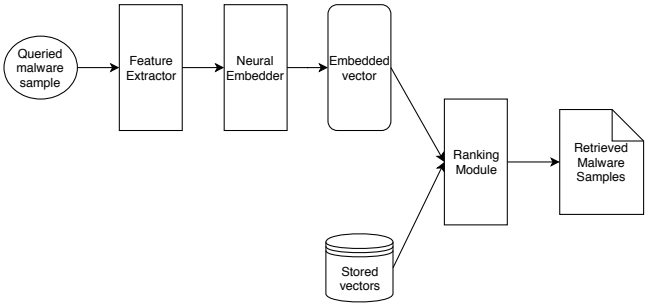

Figure 2: retrieval phase

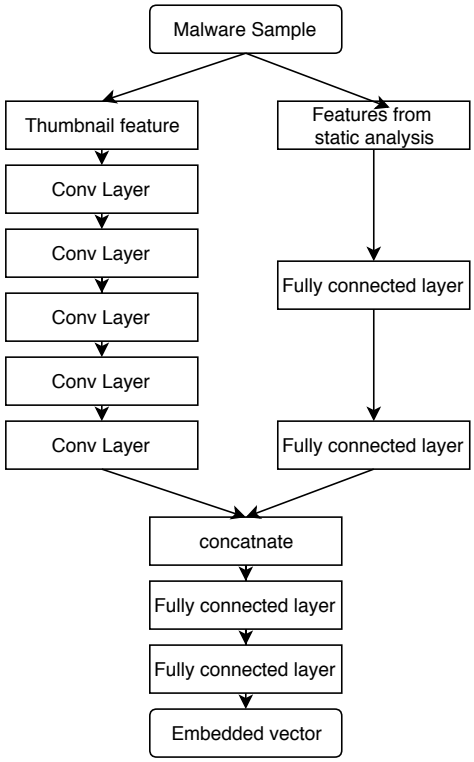

Figure 3: neural embedder

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 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 출력한다.

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. 구조적으로 다르다고 해도 멀웨어의 행위가 일치하면 같은 의미를 갖는 샘플이라고 볼 수 있다. 마찬가지로 구조적으로 거의 같다고 해도 멀웨어의 행위가 전혀 다르다면 두 샘플은 다른 의미를 가졌다고 할 수 있다. 더불어 우리는 Locky 와 Cerber 라는 두 랜섬웨어 패밀리에 해당하는 샘플 간 유사도가 Locky 와 Coinminer 에 해당하는 두 샘플의 유사도보다 더 작다고 생각한다. 이러한 샘플들 간 의미 차이 까지 고려될 수 있다면 그 시스템은 Symantics-aware한 시스템이라고 할 수 있다.

Robustness to novel samples. 멀웨어의 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 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 REPRESENTATION 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

Definition. Let $X = \{x1, \ldots, x_n, \ldots, x_N\}$ be a set of malwares. Let E and S be a subset of vector space of dimension d and let semantic component set $S = \{s_1, ..., s_k, \ldots s_K\}$ be a basis for E. We have constraints for s_i whose l2 norm is one, $||s_i||_2 = 1$ and all semantic components are linearly independent. Then for every $e \in E$, there

is an unique linear combination of the sementic component vectors that equals e.

$$e = c_1 * s_1 + c_2 * s_2 + \ldots + c_k * s_k$$

where c_i is the i'th element of coefficient vector $C = \{c_1, \ldots, c_i, \ldots, c_k\}$ We call a vector e as a semantics of malware x and there is a non-linear mapping from X to E.

Metric function we use an Euclidean Distance $d(e_i, e_j)$ for a metric of a set E and it means the function that defines a distance between semantics of two malwares.

4.2 Solution Overview

피규어다.

Multilabel Classification. 멀웨어 샘플의 의미를 반영하는 Vector representation E 를 학습시키기 위해 멀티레이블 분류 학습을 Auxiliary Task 로 선택하였다. 이 분류 태스크를 학습하기 위해 뉴럴 임베더 h는 표현 벡터들을 선형 분류기가 분류할 수 있도록 위치시킨다. 기존 대부분의 멀웨어 분류기를 학습하는 연구들은 하나의 레이블로 분류하도록 분류기를 학습한다. 하지만 멀웨어 도메인에서 하나의 레이블을 특정하고 이를 기준으로 가르치는 Auxiliary Task로 Malware IR System 을 만든다면, 학습하는 임베딩 벡터가 그 샘플의 의미를 담기에 부족한 점이 있다. 첫 째, 멀웨어에 레이블링을 하는 프로세스가 멀웨어의 의미를 표현하는데에 합리적이지 못하다. 멀웨어를 대분류, 중분류, 소분류 등 하이러키컬한 분류 체계를 갖도록 레이블링 하는 경우가 대부분이다. 하지만 실제로 멀웨어는 여러 의미를 갖고 있고,

첫 째는 하나의 레이블이 만약 Trojan이나 Adware 같은 대분류라면 같은 분류 안의 샘플들을 좀 더 세세하게 구분하여 information retrieval task 를 수행할 때에 정확하게 같은 의미의 샘플을 검색하기가 쉽지 않다. 샘플들의 벡터의 차이가 정확하게 어떤 의미인지를 알 수 없기 때문이다. 레이블이 만약 소분류라면 거의 동일한 샘플들만 같은 분류 안에 속해있을 가능성이 높고, 분류의 개수가 너무 많기 딥러닝 모델의 캐퍼시티가 충분하다면 샘플들을 외우고 하이러키컬 representation 을 학습하지 않아 Generalization 효과가 떨어지게 될 것이다. 따라서 의미적으로 정확하게 같은 샘플들은 검색될 수 있지만 의미가 비슷한, 동일하지 않은 샘플들에 대해서는 검색되지 않는다. 따라서 우리는 같은의미를 가진 샘플들 뿐만 아니라 비슷한 의미를 가진 멀웨어 샘플들에 대해서도 검색이 가능하도록 하기 위해 멀티 레이블을 분류하도록 Auxiliary Task 를 학습시켜 얻은 Vector representation 을 사용한다. 이는 여러 의미 계층에 대한 공통 피쳐를 딥러닝 모델이 모두 학습할 수 있게 되기 때문에 가능해진다.

Centerloss. Auxiliary task 로 임베딩한 샘플들이 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. 이 문제를 해결하기 위해 우리는 메트릭 러닝을 위한 목적함수를 추가하여 최종 목적함수를 세팅하였다. 센터로스[ref]를 변형한 목적함수를 통해 innerclass variance 를 낮추고 상대적으로 인터클래스의 거리는 멀어진다.

싱글레이블 센터로스를 제안했던 논문에서는 각 레이블 별로 템플릿 벡터를 익스터널 메모리에 저장해놓고 그 템플릿 벡터와 보틀넥의 거리가 작아지도록 MSE 에러를 로스에 추가했다. 우리도 마찬가지로 각 레이블별로 시멘틱 컴포넌트 벡터를 익스터널 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 업데이트해준다. 알파는 하이퍼파라미터로, 템플릿 벡터와 보틀넥을 얼마나 섞어서 기존 템플릿벡터를 업데이트할지를 결정해주는 값이다.

Consider the importances of labels. agent 나 downloader 같은 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 Centerloss 를 사용한다. 원래 리트리벌에서 랭킹을 위한 스코어링은 원래 하는짓인데 이걸 딥러닝에서 되게 한건 노블한거다. 연역적으로 왜 이렇게애해야하는지는 자명하다.

뭘 검색할 수 있어야 우리가 앞에서 문제라고 했던 애들이 해결된다. 연산한걸 검색할 수 있고,...

4.3 Proposed Objective Functions : Multilabel Center Loss

Overview 에서 설명했던 방법들을 합쳐서

Multilabel Center Loss

Multilabel Weighted Center 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.

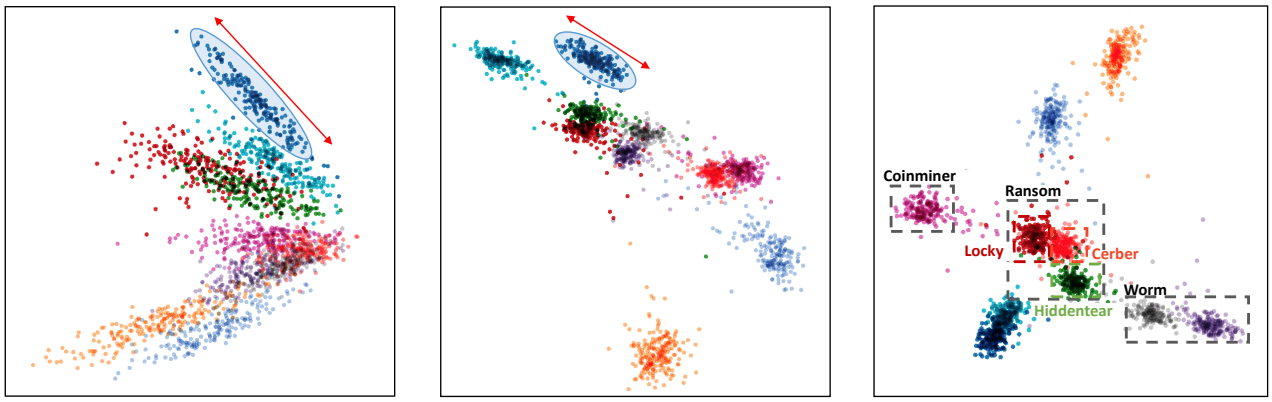

Figure 4: concept

5.1 Implementation and Setup

Datasets. 모델의 학습과 평가에 사용된 dataset은 VirusTotal로부터 [언제부터 언제까지] crawling된 샘플들 중 security expert가 검수하여 확진을 내린 샘플들로 구성되어있다. 또한 VirusTotal에서 제공하는 수십개 이상의 AntiVirus 엔진의 탐지결과로부터 Labeling을 자동화 하였다. VirusTotal로부터 각 AntiVirus 엔진들이 탐지해낸 결과를 의미있는 word 단위로 파싱하고, 샘플들로부터 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 대표 레이블과 멀티레이블 두 종류의 레이블을 부여했으며, 멀티 레이블에는 대표 레이블이 포함된다.

- **Dataset 1. PE1300.** VirusTotal로부터 crawling된 샘플들 PE 5만여개 중 security expert가 검수하여 확진을 내린 PE 샘플 1300여개 11종의 labels. 이 중 8개는 대표 레이블이며 대표 레이블에 대한 샘플들의 수는 거의 같다. 즉 대표 레이블에 대한 샘플의 분포는 유니폼하다.
- **Dataset 2. APK19000.** VirusTotal로부터 crawling된 샘플들 APK 8만여개 중 security expert가 검수하여 확진을 내린 APK 샘플 2만여개 83종의 labels. 이 중 19개는 대표 레이블이며 대표 레이블에 대한 샘플의 수는 1000개로 같다. PE 데이터셋과 마찬가지로 대표 레이블에 대한 샘플의 분포는 유니폼하다.

Hyperparameters. - layers, learning rate, train/valid ratio, Label importances,

Hancrafted Feature Extraction - Thumbnail - Size, Entropy - Call histogram

5.2 Auxiliary Task Evaluation

Auxiliary task 의 정확도는 더 안좋지만 쿼리 결과는 더 좋은 경우가 있다. 이는 Malware Semantic space의 구축 상태를 평가하기에 Auxiliary Task 의 정량 평가만으로는 부족하다는 것을 의미한다.

5.3 Querying Quantitative Test

- metric performance measures - acc, precision, recall, auc, 등 label 겹치는 정도를 수치로 정량화 - inner class variance 측정 후 센터로스 유무에 대해 비교. - Single label, Multi label, Centerloss(Mean, Add) , Weighted Center loss(Mean, Add) - Train data querying , Valid data querying

앞서 Malware Semantic Space 란 사람이 중요하다고 생각하는 레이블이 많이 겹칠 수록 가까이 있도록, 그리고 그렇지 않을 수록 멀리 있도록 멀웨어 샘플들의 표현 벡터를 위치시킨 공간이라고 설명한 바 있다. 그리고 Malware Semantic Space 의 표현 벡터들을 이용하여 MIR 을 구축했을 때 중요한 레이블이 비슷한 샘플들을 검색할 수 있을 것이라고 했다. MWC Loss 를 사용하여 MIR System 을 구축했을 때 위의 정량 수치가 대체적으로 더 높다는 것은 곧 다른 Objective function 을 사용했을 때보다 Semantics 을 더 잘 학습했다고 볼 수 있고 이는 MIR System 의 Desired property 1 번인 Semantic Understanding 속성을 다른 로스들에 비해 더 만족시킨다.

5.4 Querying Qualitative Test

Visualizations. - PCA : Variance of inner class 가 작고 variance of inter class 가 크다는 것을 눈으로 보여줌. 더불어 주요 태그를 함께 보유하고 있다면 보틀넥이 가까운 곳에 위치함을 보여줌으로써 우리의 제안 방법을 통해 Malware Semantic space 가 구축되었다는 것을 보여줌. vector space 에서 연산 관계가 있음을 보여줌.

Querying by malware sample. MWC 를 사용하여 학습시킨 MIR 시스템과 그렇지 않은 시스템에 대하여 멀웨어 샘플을 쿼링하여 얻은 top k 개의 결과를 리스팅하여 기재함으로써 MWC Loss 가 Malware Semantic space 를 구축하는데 도움이 된다는 것을 보여준다.

Querying by labels. 레이블들을 쿼링하여 동일 레이블 조합을 갖고 있는 top k 개의 결과를 리스팅하여 기재함으로써 단순히 Data retrieval system이 아니라 Information retrieval system 을 설계한 것이라는 것을 보여줌.

5.5 Center Vector Combination Methods

Mean, Add 방법의 차이를 위의 결과들로부터 설명.

add center vectors.

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 representation 을 얻기 위해 연구되었다[1),2),3)].

Multilabel embedding Chih-Kuan et al. 은 multilabel embedding 을 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.

REFERENCES

[1] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z Wang. 2008. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys (Csur)* 40, 2 (2008), 5.

[2] Lorraine Goeuriot, Gareth JF Jones, Liadh Kelly, Henning Müller, and Justin Zobel. 2016. Medical information retrieval: introduction to the special issue. *Information Retrieval Journal* 19, 1-2 (2016), 1–5.

[3] Arash Habibi Lashkari, Fereshteh Mahdavi, and Vahid Ghomi. 2009. A boolean model in information retrieval for search engines. In *Information Management and Engineering, 2009. ICIME'09. International Conference on*. IEEE, 385–389.

[4] André Mourão, Flávio Martins, and João Magalhães. 2015. Multimodal medical information retrieval with unsupervised rank fusion. *Computerized Medical Imaging and Graphics* 39 (2015), 35–45.

[5] Gerard Salton, Anita Wong, and Chung-Shu Yang. 1975. A vector space model for automatic indexing. *Commun. ACM* 18, 11 (1975), 613–620.

[6] Igor Santos, Xabier Ugarte-Pedrero, Felix Brezo, Pablo Garcia Bringas, and José María Gómez-Hidalgo. 2013. Noa: An information retrieval based malware detection system. *Computing and Informatics* 32, 1 (2013), 145–174.

[7] Markus Schedl, Emilia Gómez, Julián Urbano, et al. 2014. Music information retrieval: Recent developments and applications. *Foundations and Trends® in Information Retrieval* 8, 2-3 (2014), 127–261.

[8] Jun Yu, Dacheng Tao, Meng Wang, and Yong Rui. 2015. Learning to rank using user clicks and visual features for image retrieval. *IEEE transactions on cybernetics* 45, 4 (2015), 767–779.