Scientific Journal of Electronical & Cyber Defence
Vol. 11, No. 1, Spring 2023, Serial No. 41

A Malware Classification Method Using visualization and Word Embedding Features

E. Bastami, H.Soltanizadeh*,M. Rahmanimanesh, P. Keshavarzi
*Associate Professor, Semnan University, Semnan, Iran
(Received: 19/11/2020, Accepted: 27/02/2021)

ABSTRACT

With the explosive growth of threats to Internet security, malware visualization in malware classification has become a promising study area in security and machine learning. This paper proposes a visualization method for malware analysis based on word embedding features of byte sequences. Based on some assistant information such as word embedding, the basic to a strong malware classification approach is to transfer the learned information from the malware domain to the image domain, which needs correlation modeling between these domains. However, most current methods neglect to model the relationships in an embedding way, ensuing in low performance of malware classification. To catch this challenge, we consider the Word Embeddings duty as a Semantic Information Extraction. Our Proposed method aims to learn effective representations of malware families, which takes as input a set of embedded vectors corresponding to the malware. Word embedding is designed to generate features of a malware sample by leveraging its malware semantics. Our results show that visual models in the domain of images can be used for efficient malware classification. We evaluated our method on the Kaggle dataset of Windows PE file instances, obtaining an average classification accuracy of 0.9896%.

Keywords: Malware Detection, Malware Visualization, Malware Embedding, Static Analysis, CNN Algorithm.

*Corresponding Author Email: H_soltanizadeh@semnan.ac.ir

نشریه علمی "پدافند الکترونیکی و سایبری"
سال یازدهم، شماره ۱، بهار ۱۴۰۲، ص۱-۱۳

علمی – پژوهشی

روش طبقه‌بندی بدافزار با استفاده از ویژگی‌های بصری سازی و تعبیه‌سازی کلمه براساس یادگیری عمیق

اسماعیل بسطامی[۱]، هادی سلطانی‌زاده[۲*]، محمد رحمانی منش[۳]، پرویز کشاورزی[٤]

۱- دانشجوی دکترا، ۲- دانشیار، ۳و٤- استاد، دانشگاه سمنان، سمنان، ایران

(دریافت: ۱٤۰۰/۰۵/۱۹، پذیرش: ۱٤۰۱/۰٦/۲۷)

چکیده

با رشد انفجاری تهدیدات برای امنیت اینترنت، بصری‌سازی بدافزارها در حوزه طبقه‌بندی بدافزارها به یک حوزه مطالعه امیدوارکننده در زمینه امنیت و یادگیری ماشین تبدیل شده است. این مقاله یک روش بصری‌سازی برای تجزیه‌وتحلیل بدافزار را بر اساس ویژگی‌های تعبیه‌سازی دنباله‌های کد دستوری پیشنهاد می‌کند. بر اساس برخی اطلاعات کمکی مانند تعبیه‌سازی کلمه، روش اصلی طبقه‌بندی بدافزار پیشنهادی، انتقال اطلاعات آموخته شده از حوزه بدافزار به حوزه تصویر است که نیاز به مدل‌سازی همبستگی بین این حوزه‌ها دارد. بااین‌حال، اکثر روش‌های فعلی از مدل‌سازی روابط غفلت می‌کنند که منجر به طبقه‌بندی نادرست بدافزارها می‌شود. برای غلبه بر این چالش، ما وظیفه تعبیه‌سازی کلمه[۱] را به‌عنوان استخراج اطلاعات معنایی در نظر می‌گیریم. روش پیشنهادی یک روش طبقه‌بندی بدافزار با استفاده از مفاهیم تعبیه‌سازی کلمات و بصری‌سازی از توالی‌های کد دستور و یک روش شبکه‌های عصبی شامل یادگیری عمیق ($2CNN$) را پیشنهاد می‌کند. نتایج ما نشان می‌دهد که از مدل‌های بصری در حوزه تصاویر می‌توان برای طبقه‌بندی کارآمد بدافزارها استفاده کرد. ما روش خود را بر روی مجموعه داده *kaggle* ارزیابی کردیم و میانگین دقت طبقه‌بندی ۰/۹۸۹۶ و امتیاز $F1$ برابر ۰/۹۸۰۷ به دست آوردیم.

کلیدواژه‌ها: تشخیص بدافزار، بصری‌سازی بدافزار، تعبیه‌سازی بدافزار، تجزیه‌وتحلیل ایستا، الگوریتم CNN

۱- مقدمه

رایانه‌های مدرن و زیرساخت‌های ارتباطی به‌شدت در معرض انواع مختلف حملات امنیتی هستند. یک روش متداول برای انجام این حملات استفاده از نرم‌افزارهای مخرب (بدافزار) مانند کرم‌ها، ویروس‌ها و اسب‌های تروجان است که در صورت انتشار می‌توانند باعث آسیب شدید به کاربران خصوصی، شرکت‌های تجاری و دولت‌ها شود. بدافزار اصطلاحی است که هر برنامه یا کد مخربی را توصیف می‌کند که برای سیستم‌ها مضر است. بدافزار خصمانه، مزاحم و عمداً به دنبال حمله، آسیب رساندن یا غیرفعال‌کردن رایانه‌ها، سیستم‌های رایانه‌ای، شبکه‌ها است که اغلب با در دست‌گرفتن کنترل جزئی بر عملکرد دستگاه‌ها انجام می‌شود. رشد اخیر در اتصالات اینترنتی پرسرعت، بدافزارها را قادر می‌سازد تا خیلی سریع میزبان‌ها را منتشر کرده و آلوده کنند، بنابراین شناسایی و حذف بدافزارها به شیوه‌ای مناسب ضروری است [۱]. یک مسئله اصلی در روش‌های تشخیص الگو برای سیستم‌های طبقه‌بندی و تشخیص بدافزارها، تعیین قابلیت درک ماشین در بدافزار است. برای درک ماشین از بدافزار، توجه به ویژگی‌های مختلف ساختاری و محتوایی بدافزارها موضوع مهمی است. تلاش‌های گسترده‌ای برای تعیین ویژگی‌های بدافزار انجام شده است که بتوانند موارد شهودی دقیقی را ارائه بدهند. در کارهای انجام شده حوزه تشخیص بدافزار، انواع مختلفی از ویژگی‌ها بر اساس ویژگی ارزش‌های آنها وجود دارد. این ویژگی‌ها یا شامل ویژگی‌های عددی هستند که اندازه‌گیری کمی را توصیف می‌کنند یا شامل ویژگی‌های اسمی هستند که مجموعه‌ای از برچسب‌ها را مشخص می‌کنند [۱]. به طور معمول، اکثر سیستم‌های تشخیص بدافزار بر روی ویژگی‌های متداول از جمله *Opcode sequences*[۲ ، ۳]، ویژگی‌های *API call* [۴ ، ۵]، *PE header* [۶] و ویژگی‌های دودویی [۷ ، ۸] تمرکز می‌کنند که شامل ویژگی‌های اساسی نرم‌افزارهای مخرب است. متداول‌ترین روش‌های استخراج ویژگی بدافزارها، تحت دو روش فعال و غیرفعال، قرار می‌گیرد. دسته اول، بر رویکردهای مبتنی بر امضا متمرکز دارند درحالی‌که دسته دوم بر رویکردهای مبتنی بر رفتار متمرکز هستند [۹]. بااین‌حال، در دسته دوم، روش‌های دیگری برای طبقه‌بندی بدافزارها وجود دارد [۱۱-۱۶] که به دودسته ایستا و پویا تقسیم‌بندی می‌شوند [۱۰]. در روش‌های ایستا برای تشخیص بدافزار، فایل بدافزار اجرا نمی‌شود و به‌صورت تحلیل ایستا ویژگی‌ها استخراج می‌شود درصورتی‌که در روش‌های پویا می‌بایست فایل بدافزار اجرا شود. روش‌های ایستا از دستورات

* رایانامه نویسنده مسئول: H_soltanizadeh@semnan.ac.ir

[1] Word Embedding
[2] Convolutional Neural Networks

نحوی ویژگی‌های کد منبع یک برنامه مخرب با مهندسی معکوس hex برنامه و تجزیه‌وتحلیل فایل‌های اجرایی (*PE*)، بدون اجرای آن برای استخراج ویژگی‌ها استفاده می‌شوند[۱۷]. در این روش الگوهای موجود در بدافزار را می‌توان به‌صورت ایستا به دست آورد. این الگوها شامل امضای رشته [۱۸، ۱۹]، گراف جریان کنترل(CFG) [۲۰]، *system API calls* [۲۱]، *opcodefrequency* و *sequence n-grams* [۲۲] است. در روش‌های پویا، یک برنامه مخرب در موقعیت جداگانه یا جعبه شنی[1] اجرا می‌شود و رفتار آن پایش می‌شود و برای بررسی بیشتر گزارش می‌شود. بااین‌حال، ازنقطه‌نظر طبقه‌بندی بدافزارها، بسیاری از روش‌های پویا در حال حاضر دارای برخی از معایب زیر هستند:

۱. این روش، برنامه را اجرا می‌کند که زمان‌بر است و منابع زیادی را در بر می‌گیرد.

۲. اجرای برنامه‌های مخرب همچنین تهدیدی برای ضربه‌زدن به میزبانی است که بدافزار در آن اجرا می‌شود.

۳. بدافزارها در شرایط در حال اجرا اغلب عملکرد کنترل‌شده و باهدف خاصی را به نمایش می‌گذارند و رفتار اصلی خود را پنهان می‌کند.

به دلیل وجود معایب مختلف روش‌های پویا و دارابودن سرعت مناسب تشخیص بدافزار در روش‌های ایستا، در بسیاری از سناریوهای مقابله با بدافزار، از این روش استفاده می‌شود. در بین تمام روش‌های تحلیل ایستا، بصری‌سازی بدافزار به دلیل سرعت‌بخشیدن به عملیات تشخیص بدافزار از رایج‌ترین روش‌ها است. اخیراً، برخی از رویکردهای بصری‌سازی مبتنی بر تحلیل ایستای بدافزارها [۳، ۲۳-۲۵] برای طبقه‌بندی بدافزارها معرفی شده است. این روش‌ها تمرکز بر نمونه‌سازی ویژگی‌های بدافزار به شکلی گرافیکی دارند که می‌تواند برای حمل اطلاعات اضافی بر روی یک بدافزار مورداستفاده قرار گیرد. این روش‌ها به دلیل تغییر دامنه از حوزه گسسته به پیوسته و استفاده از روش‌های پردازش داده‌های پیوسته پرکاربرد و دارای عملکرد مناسب هستند.

بخش‌های مختلف یک فایل اجرایی از جمله قسمت opcodes را می‌توان به‌راحتی به‌عنوان یک تصویر مشاهده کرد [۲۶]. وقتی فایل مخرب به شکل تصویر تبدیل می‌شود، می‌توان ویژگی‌های مبتنی بر تصویر که بعداً می‌توان با یادگیری عمیق استخراج کرد را برای تشخیص و مشخص‌کردن نوع بدافزار استخراج کرد. یک روش منطقی بصری‌سازی بدافزار باید از برخی اطلاعات مفید به‌عنوان ورودی اصلی استفاده کند که باعث ایجاد اهمیت داده-های ارائه شده در حفظ معناداری اطلاعات می‌شود. بااین‌حال، روش‌های تحلیل ایستا از فقدان اطلاعات معنایی[2] رنج می‌برند.

این مقاله یک ویژگی جدید را بر اساس اطلاعات معنایی معرفی می‌کند که معمولاً در طبقه‌بندی دسته صحنه[3] استفاده می‌شود. ما نشان می‌دهیم که روش پیشنهادی یک دامنه تبدیل از حوزه طبقه‌بندی بدافزارها به حوزه پردازش تصویر و مدل‌سازی زبان است. این روش نشان می‌دهد که مدل Word2Vec بر اساس توالی کد دستوری می‌تواند نرخ تشخیص بدافزار را افزایش دهد. تعبیه‌کردن اطلاعات معنایی در تصاویر بدافزار چالشی است که بسیاری از محققین در حال ارائه راهکار برای آن هستند. برای اضافه‌کردن اطلاعات معنایی به تصاویر بدافزار با چالش تجمیع اطلاعات معنایی و بصری مواجه هستیم. این اطلاعات باید به‌گونه‌ای باهم ترکیب شوند که قابل‌استفاده در یادگیری عمیق باشند. روش‌های مبتنی بر *CNN* اطلاعات را به‌صورت ماتریسی به‌عنوان ورودی دریافت می‌کنند. در این مقاله، ما یک رویکرد شناسایی بدافزار بر اساس بصری‌سازی بدافزار و استفاده از اطلاعات معنایی ارائه می‌دهیم. رویکرد پیشنهادی، ابتدا مجموعه بدافزارها را به ساختار برداری تبدیل می‌کند. در ادامه، این بردارهای بدافزاری را به‌صورت تصویر، بصری‌سازی می‌کند. سپس به طور خودکار ویژگی‌های بدافزار را با استفاده از استخراج ویژگی بر اساس شبکه‌های عصبی عمیق (*CNN*) از تصویر استخراج می-کند. در این مقاله، تمرکز بر روی روش شناسایی بدافزار مبتنی بر تعبیه‌سازی برای توسعه موتور تشخیص بدافزار هوشمند است. این مقاله روش پیشنهادی برای طبقه‌بندی فایل‌های PE مخرب را موردبحث قرار می‌دهد. این کار بینشی در مورد مطالعه مداوم در طبقه‌بندی بدافزارها با استفاده از ویژگی‌های بصری و معنایی ارائه می‌دهد. ما عملکرد روش پیشنهادی و روش‌های موجود را بر روی مجموعه داده معیار با هم مقایسه می‌کنیم. نتایج تجربی نشان می‌دهد که رویکرد ما می‌تواند نتایج طبقه‌بندی رقابتی را با پیچیدگی کمتری به دست آورد.

نوآوری و مشارکت‌های این مقاله عبارت‌اند از:

۱) برای اضافه‌کردن اطلاعات معنایی در روش پیشنهادی از رویکردهای پردازش زبان‌های طبیعی برای توالی‌های کد در نمونه‌های بدافزار استفاده می‌کنیم. به‌صورت تجربی نشان می-دهیم که دنباله بایت‌های مختلف دارای تنوع معنی‌داری در بردارهای کلمه در خانواده‌های مختلف بدافزار است.

۲) روش پیشنهادی بردار تعبیه‌شده را به حوزه تصویر تبدیل می-کند. در ادامه در روش پیشنهادی از الگوریتم شبکه عصبی عمیق (*CNN*) برای استخراج ویژگی‌های بصری بدافزار تعبیه شده استفاده شده است.

در ادامه ساختار این مقاله به‌صورت زیر سازماندهی شده است: در بخش ۲، یک مطالعه در مورد تشخیص بدافزارها و بصری‌سازی بدافزارها ارائه شده است. بخش ۳ نمادگذاری و فرمول‌بندی

[1] sandbox
[2] Semantic Information
[3] Scene Category Classification

ناشر: دانشگاه جامع امام حسین (ع)

مسئله را نشان می‌دهد. بخش ۴ روش *word2vec* را نمایش می‌دهد. بخش ۵ روش پیشنهادی را ارائه می‌دهد و ویژگی‌های آن را تحلیل می‌کند. بخش ۶ نتایج تجربی را نشان می‌دهد. در نهایت، بحث و نتیجه‌گیری در بخش ۷ و ۸ آورده شده است.

۲- کارهای مرتبط

در این بخش، ابتدا کارهای مرتبط به کار خود را که شامل تشخیص بدافزار و بصری‌سازی بدافزار است را بررسی می‌کنیم و در نهایت چالش‌های حوزه بصری‌سازی را برجسته می‌کنیم و کارهایی که در این حوزه انجام شده است را تحلیل می‌کنیم.

۲-۱-تشخیص بدافزار

وجود برنامه مخرب در دستگاه قربانی را می‌توان با روش‌های گوناگون شناسایی کرد و می‌توان برنامه را از طریق روش‌ها و رویکردهای مختلف مورد بررسی قرار داد. امروزه بدافزار برای استخراج ویژگی‌هایی که می‌توان برای شناسایی آنها استفاده کرد مورد بررسی قرار گرفته‌اند. به‌طورکلی، دو روش اصلی تجزیه‌وتحلیل بدافزار وجود دارد: تجزیه‌وتحلیل ایستا و تجزیه‌وتحلیل پویا. کارهای انجام شده در حوزه تحلیل ایستا به شرح زیر توضیح داده شده‌اند.

۲-۱-۱- تحلیل ایستای بدافزار

روش‌های مبتنی بر ایستا یک روش تجزیه‌وتحلیل سریع هستند. در تجزیه‌وتحلیل بدافزار به‌صورت ایستا، ویژگی‌های بدافزار بدون اجرای برنامه‌های بدافزار استخراج می‌شوند. هدف تجزیه‌وتحلیل ایستا می‌تواند تحلیل کدهای باینری یا اسمبلی باشد [۲۷]. نمونه‌های بدافزار برای بررسی این فایل‌ها در سطح اسمبلی از کد باینری به منبع کد اسمبلی مهندسی معکوس می‌شوند. کد منبع اسمبلی فایل بدافزار می‌تواند حاوی برخی اطلاعات مهم مانند فراخوانی‌های DLL ها، URLs accessed و مثل این‌ها را در اختیار ما قرار دهد، زیرا کد اسمبلی بدافزار می‌تواند به طور صریح هدف مخرب خود را نشان دهد. ازاین‌رو نویسندگان بدافزارها سعی می‌کنند از بخش‌های مخرب فایل‌های بدافزار محافظت کنند. چندین ویژگی ایستای بدافزار پس از تجزیه‌وتحلیل مانند API Call، N-gram ، گراف‌های جریان کنترل، رشته‌ها، مقدار هش، عملوندها و اطلاعات PE header استخراج می‌شوند[۲۸]. برای متمایزکردن انواع بدافزارها از بقیه، از یک طرح فراخوانی مبتنی بر استخراج گراف کد استفاده می‌کنند که به آن روش minimal contrast frequent subgraph miner می‌گویند. کار پژوهشی [۱۹] از ویژگی‌هایی مانند فهرستی از فراخوان‌های سیستمی، تابع‌های DLL و hex-dump برای تشخیص اجزای بدافزار شناخته نشده استفاده کرده است. کار پژوهشی [۲۹] یک چارچوب MKLDROID با استفاده از یک هسته گراف با چند هسته برای یادگیری مجموعه‌ای از اطلاعات زمینه‌ای و ساختارهای معنایی توسعه داده است. متأسفانه، مطالعات نشان داده‌اند که این ترفندها را می‌توان به‌سادگی با حملات خصمانه در مورد کد مخرب دور زد. دلیل این مسئله این است که وابستگی بین ویژگی‌های مختلف بدافزار به‌واسطه سیستم تشخیص‌دهنده قوی نیست. اخیراً روش‌های قوی مختلفی برای پرداختن به نمونه‌های خصمانه پیشنهاد شده است. این رویکردها به امضای معنایی بستگی دارد و از روش‌های ایستا مانند تعبیه-سازی کلمه استفاده می‌کنند. در این مقاله سعی کردیم که از اطلاعات معنایی استفاده کنیم. منظور از اطلاعات معنایی این است که دستورات کد اسمبلی در کد منبع بدافزار به طور معناداری در کنار هم آمده‌اند. ایده این مفهوم از حوزه پردازش زبان‌های طبیعی آمده است. در حوزه پردازش زبان‌های طبیعی، کلمات حاوی معنا هستند و در کنار همدیگر معنا پیدا می‌کنند. با این استدلال، opcodeها در کد منبع بدافزار حاوی معنا هستند و در کنار همدیگر می‌توانند مشخص کنند الگوی حاکم بر کد منبع بدافزار باشند.

۲-۱-۲- بصری‌سازی بدافزار

هدف بصری‌سازی بدافزار نشان‌دادن ویژگی‌های فایل‌های مخرب در فرم سیگنال‌های بصری است که می‌تواند برای انتقال اطلاعات بیشتر در مورد نرم‌افزارهای مخرب خاص مورداستفاده قرار گیرد [۳۰]. بصری‌سازی بدافزار به طور قابل‌توجهی سریع‌تر و دقیق‌تر از روش‌های تجزیه‌وتحلیل سنتی بدافزارها است [۳۱]. رویکردهای فعلی بصری‌سازی بدافزارها عموماً به دو دسته تقسیم می‌شوند: روش‌های سطح باینری و روش‌های سطح اسمبلی. روش‌های باینری، فایل‌های باینری بدافزار را به‌عنوان تصاویر gray-scale بصری‌سازی می‌کنند [۲۶]. درحالی‌که روش‌های سطح اسمبلی فایل‌های اجرایی را به کدهای اسمبلی تبدیل می-کنند و سپس به تصاویر تبدیل می‌کنند [۳۲]. راهبردهای اولیه بصری‌سازی بدافزار از یک طرح بصری پیروی می‌کنند که ابتدا طبقه‌بندی‌کننده‌های مختلف ویژگی‌ها را می‌آموزند و سپس با مقایسه ویژگی‌های پیش‌بینی شده آن با اطلاعات کلاس‌های دیده نشده، الگوی بصری را مشخص می‌کنند. در حال حاضر، تحقیقات زیادی انجام شده است، از جمله طبقه‌بندی بدافزارها با استفاده از RNNs [1] [۳۳]، CNN[۳۵،۳۴] و مدل‌های ترکیبی [۳۶]. اخیراً، بسیاری از تحقیقات بر مدل‌های شبکه عصبی عمیق مبتنی بر تکنیک‌های بینایی ماشین با CNN برای دسته‌بندی بدافزارها تمرکز کرده‌اند [۳]. کار [۳] در این حوزه پیش‌گام است که کدهای بدافزار را به تصاویر دیجیتالی تبدیل می‌کند و آن‌ها را برای شناسایی به الگوریتم CNN منتقل می‌کند. روش پیشنهادی در این مقاله از کار [۳] الهام گرفته است. آن‌ها از یک الگوریتم simhash برای ساخت بردار بدافزارها استفاده کرده‌اند

[1] Recurent Neural Network

که به دلیل فقدان اطلاعات معنی‌دار، این رویکرد از مشکلاتی رنج می‌برد. در نتیجه، روش‌های مبتنی بر معنا در بصری‌سازی بدافزار برای مدیریت خصوصیات ویژه بدافزارها موردنیاز است. در این مقاله، هدف ما یادگیری اطلاعات معنایی نهفته در کدهای بدافزار و سپس استخراج الگوهای بصری با روش‌های عمیق در یک فایل اسمبلی است.

۳- روش پیشنهادی

در ادامه ابتدا فرمول‌بندی و مدل‌سازی مسئله رو مطرح می‌کنیم و سپس روش پیشنهادی را شرح می‌دهیم. روش پیشنهادی عمدتاً به چهار بخش تقسیم می‌شود: تولید توالی Opcode ، تعبیه‌سازی Word2vec، بصری‌سازی بدافزار و آموزش *CNN*. در قسمت اول یک مرحله پیش‌پردازش به‌نوعی انجام می‌شود و Opcodeهای برنامه استخراج می‌شوند. در این قسمت باید بخشی را که قابل تشخیص است از نمونه‌های بدافزار استخراج کرده و توسط SimHash رمزگذاری کنیم. در بخش دوم موارد استخراج شده از بخش اول، جهت استخراج اطلاعات معنایی، تعبیه‌سازی می‌شوند. در بخش بعدی تصاویری بر پایه اطلاعات مراحل قبلی ایجاد می‌شود. در بخش نهایی این تصاویر برای آموزش مدل CNN به‌صورت مکرر باهدف شناسایی خانواده‌های آن‌ها استفاده می‌شود.

۳-۱- مدل‌سازی مسئله

ما در این مقاله طبقه‌بندی بدافزارها را به‌عنوان یک مسئله چند کلاسه با تعداد کلاس $K = 9$ می‌دانیم. X را به‌عنوان بردار ورودی ویژگی و $Y = \{1,2,3, \dots ,9\}$ را به‌عنوان بردار خروجی ویژگی در نظر می‌گیریم. فرض کنید ما نمونه‌ها را به‌صورت $S = \{z_1 = (x_1, y_1), z_2 = (x_2, y_2), \dots , z_n = (x_n, y_n)\}$ با توزیع ناشناخته μ در $Z = X \times Y$ نمایش می‌دهیم. با توجه به مجموعه‌ای از جفت‌های مرتب $\{(\overrightarrow{x_j}, y_j)\}$ از مجموعه داده‌ی بدافزار که بردار $\overrightarrow{x_j}$ ، موقعیت بدافزار است در بردار ویژگی ورودی، $y_j \in C=[1..n_c]$ برچسب بدافزار است که n_c تعداد برچسب‌ها یا کلاس‌ها است و کلاس‌ها بر اساس توزیع ناشناخته $P(c|\overrightarrow{x_j})$ توزیع می شوند. $c \in [1..n_c]$ نام کلاس و $\overrightarrow{x_j}$ موقعیت در بردار ورودی ویژگی است.

باتوجه به یک نمونه تست $\overrightarrow{x_j}$ ، می‌خواهیم C_i را محاسبه کنیم. باهدف تشخیص هرگونه ویژگی خانواده بدافزار، ما توالی‌های کد را از بخش کد در فایل PE به‌عنوان بردار ویژگی ورودی استخراج می‌کنیم. ویژگی‌ها از برنامه (P) به‌عنوان دنباله‌ای از دستورات I ساخته می‌شوند که $P = (I_1, I_2, I_3, \dots , I_n)$، n تعداد دستورالعمل‌های کد بدافزار است. دستورالعمل شامل یک کد دستور[1] و عملوند[2] است[۳۷] که در این مقاله، ما روی کدهای دستور تمرکز می‌کنیم. ما فرض می‌کنیم که یک برنامه دلخواه از توالی‌های کد دستور تشکیل شده است. بنابراین، برنامه را می‌توان به‌عنوان مجموعه‌ای از توالی‌های کد دستور به‌صورت $P = (o_1, o_2, o_3, \dots , o_m)$ تعریف کرد.

۳-۲- مدل Word2vec

میکولوف و همکاران[۳۹] با فرضیه توزیع شده و مدل‌های زبانی فضای پیوسته که از شبکه‌های عصبی استفاده می‌کنند انگیزه گرفتند و الگوریتم word2vec را برای نمایش کلمات در یک گروه بدون برچسب پیشنهاد کردند. تعبیه‌سازی کلمات با استفاده از معماری شبکه‌عصبی عمیق، هیجان انگیز است زیرا بردارهای تعبیه شده به طور صریح بسیاری از الگوهای دستوری را تعبیه کرده‌اند. این مدل یک معماری شبکه عصبی کم عمق است که با بهینه‌سازی تابع هدف که شامل کلمه زمینه[3] و کلمه هدف[4] است، کلمه embedding را می‌آموزد. مدل Word2vec یک بسته است که شامل دو الگوریتم (CBOW و Skip-gram) و دو مدل آموزشی (softmax سلسله مراتبی و نمونه گیری منفی) است. مدل CBOW قصد دارد یک کلمه مورد نظر را از بازنمایی‌های توزیع شده محیط اطراف بر حسب بردارهای کلمه برآورد کند [۳۹]. به جای برآورد کلمه مرکزی بر اساس کلمات زمینه، Skip-gram توزیع (احتمال) کلمات اطراف را از کلمه مورد نظر برآورد می‌کند [۳۹].

۳-۳- مدل Skip-Gram

وظیفه تابع کلی loss معماری Skip-gram کشف معنای کلمات است. این نمایش برای پیش‌بینی کلمات زمینه‌ای برای هر کلمه مرکزی در یک جمله مفید است. به طور رسمی، باتوجه‌به دنباله‌ای از کلمات $W_1, W_2, W_3, \dots W_T$، هدف از تابع loss در این مدل به حداکثر رساندن میانگین log توزیع فرمول (۱) است

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c, j\neq 0} log\ p(W_{t+j}|W_t) \quad (۱)$$

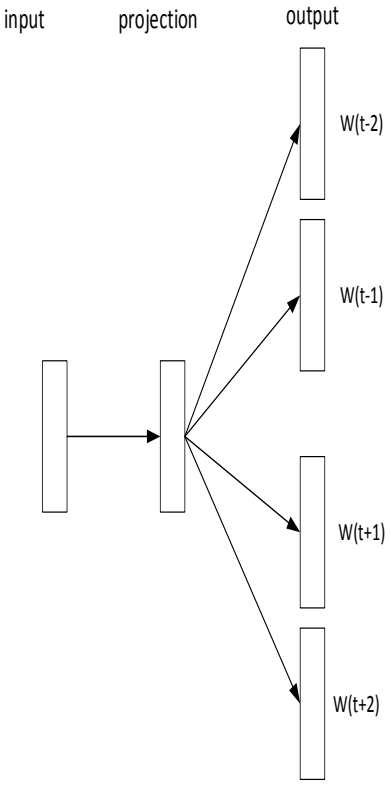
در فرمول(۱)، c طول کلمات زمینه است که می‌تواند تابعی از کلمه مرکزی آموزشی W_t باشد. افزایش قابل ملاحظه در تعداد نمونه‌های آموزشی، می‌تواند باعث افزایش کارایی در بحث هزینه زمان آموزش شود. فرمول معماری Skip-gram مقدار $p(W_{t+j}|W_t)$ را با استفاده از تابع softmax در فرمول (۲) تعیین می‌کند:

$$p(W_O|W_I) = \frac{\exp({v'_{wo}}^T v_{WI})}{\sum_{w=1}^{W}\exp({v'_w}^T v_{WI})} \quad (۲)$$

[1] opcode

[2] operands
[3] context
[4] target

که در فرمول (۲)، v'_w و v_w به ترتیب نمایش بردار خروجی و ورودی برای w هستند. این فرآیند در شکل(۱) آمده است.



شکل (۱): فرایند تعبیه‌سازی در مدل word2vec [۳۱]

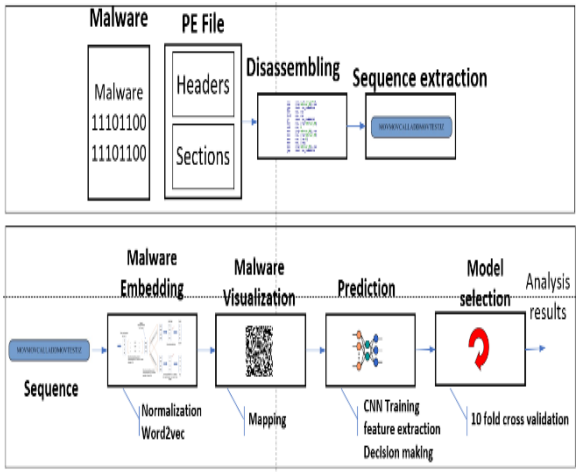
در الگوریتم Word2vec هر کلمه با توزیع وزن در آن عناصر نشان داده می‌شود؛ بنابراین به‌جای نگاشت یک‌به‌یک بین یک عنصر در بردار و یک کلمه، نمایش یک کلمه در تمام عناصر بردار پخش می‌شود و هر عنصر در بردار به تعریف بسیاری از کلمات کمک می‌کند. چنین بردار به نحوی انتزاعی "معنا" یک کلمه را نشان می‌دهد. با بررسی یک corpus ، می‌توان بردارهای کلمه را یاد گرفت که می‌توانند روابط بین کلمات را به شیوه‌ای شگفت‌آور بیان کنند. ما در این مقاله از این بردار استفاده خواهیم کرد تا بتوانیم یک امضا برای یک بدافزار تولید کنیم.

مدل‌های زبانی NLP اجازه می‌دهند تا تعبیه‌سازی مناسب برای بدافزار $x \in X$ به‌عنوان ورودی برای تشخیص‌دهنده‌های مختلف بدافزار مبتنی بر یادگیری عمیق اعمال شوند. این بخش، سازوکار پشت تعبیه‌سازی کلمات را با استفاده از یک الگوریتم پرکاربرد برای تعبیه‌سازی کلمات که به‌عنوان Word2Vec شناخته می‌شود، بیان کرده است. روش پیشنهادی عمدتاً به بخش‌های تولیدکننده توالی کد، تعبیه‌سازی word2vec، ایجاد تصویر بدافزار و پیش‌بینی تقسیم می‌شود. شکل (۲) طرح ساده‌ای از نحوه کاربرد معماری پیشنهادی در سیستم طبقه‌بندی بدافزارها را نشان می‌دهد.

الگوریتم ۱ روش پیشنهادی

inputs: Program P , Embedding size d, Image size b
Output: matrix of malware representations; $\boldsymbol{\Phi} \in \boldsymbol{R}^{|p| \times d}$
1 Initialization
2 Initialize a n -bit vector v as zero vector;
3 Initialize a binary number s as zero;
4 OP = OSG(P);
5 WV= WVE(OP, d)
6 I= MIG(WV, b)
7 Prediction(I)
8 **end**

روش پیشنهادی در الگوریتم ۱ نشان‌داده‌شده است. خطوط ۴-۷ در الگوریتم ۱ پایه روش ما را نشان می‌دهد. در این الگوریتم بخش‌های اساسی این الگوریتم در خطوط ۴ تا ۷ آمده است. این چهار خط بخش‌های اصلی روش پیشنهادی را انجام می‌دهند. در خط ۴، تولیدکننده توالی کد، بخش کد از یک برنامه اجرایی P را به‌عنوان ورودی دریافت می‌کند و دنباله‌ای از کدها را به طور یکنواخت تولید می‌کند. در خط ۵، توالی کدهایی که از نمونه‌های بدافزار قابل‌تشخیص است توسط word2vec تعبیه‌سازی شده است. در خط ۶، این بردارهای word2vec به تصاویر بدافزار تبدیل می‌شوند. در خط ۷، شباهت‌های بین تصاویر بدافزار با آموزش CNN محاسبه شده است. در بخش‌های بعدی، هر مرحله به‌تفصیل توضیح داده شده است. شکل (۲) نمای کلی روش ما را نشان می‌دهد.



شکل (۲): نمای کلی روش پیشنهادی

۳-۴- تولید توالی Opcode

در ابتدا فایل دودویی بدافزار برای به‌دست‌آوردن توالی کد مهندسی معکوس می‌شود. اولین مرحله حیاتی سیستم طبقه‌بندی بدافزارها تعیین نمایش فایل‌های برنامه است. استخراج متمایزسازترین ویژگی‌های موجود در یک مجموعه‌داده که برای نمایش و بیان داده‌ها استفاده می‌شود، استخراج ویژگی نامیده

می‌شود [۴۰]. ویژگی‌ها نقش مهمی در زمینه تشخیص بدافزار برای شناسایی اطلاعات مهم دارند. قبل از استخراج ویژگی‌های بدافزار (استخراج توالی کد دستور)، تکنیک‌های پیش‌پردازش (جداسازی و تمیزسازی) روی فایل برنامه P انجام می‌شود. پس از دریافت برنامه مهندسی معکوس شده، تمام دستورالعمل‌ها (I) استخراج می‌شوند. سپس توالی‌های کد دستورها (O) را با استفاده از یک جداکننده، مشخص می‌کنیم.

۳-۵- تعبیه‌سازی Word2vec

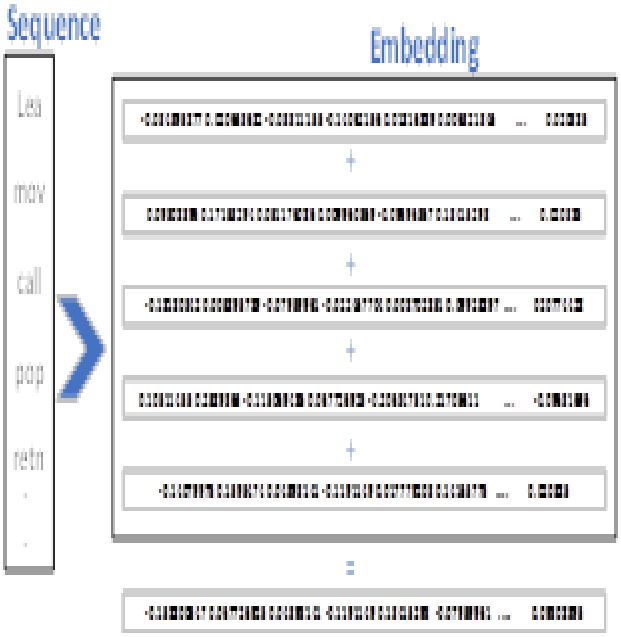
در این مقاله، ما یک برنامه مدل‌سازی زبان را برای تجزیه‌وتحلیل بدافزار از طریق جریانی از کدهای دستوری(o_i) استفاده می‌کنیم. این توالی‌های کد دستوری را می‌توان به‌عنوان جمله در یک زبان خاص در نظر گرفت. هدف از مدل‌سازی زبان، تعیین مقدار احتمال توالی خاصی از کدهایی است که در برنامه ظاهر می‌شود و بسته به هدف مدل زبان متفاوت است. به طور رسمی، با توجه به دنباله‌ای از کدهای دستور $O_1^n = (o_1, o_2, o_3, , , o_n,)$، که در آن $o_i \in P$ برنامه P است)، ما می‌خواهیم $Pr(o_n|o_0, o_1, \dots, o_{n-1})$ را در تمام داده‌های آموزش محاسبه کنیم. کار فعلی در این مقاله در این قسمت در زمینه یادگیری بازنمایی در طبقه‌بندی بدافزارها بر استفاده از شبکه‌های عصبی مصنوعی برای ایجاد بازنمایی کلی از کدهای دستوری که مدل‌سازی زبان را در پشت اهداف اصلی آن گسترش می‌دهند متمرکز شده است. این مقاله از تعمیم روش‌های پردازش زبان طبیعی برای یادگیری الگوهای توالی کد دستور بدافزار استفاده می‌کند. این کدهای دستوری را می‌توان به‌عنوان جملات و عبارات در یک زبان در نظر گرفت. مقایسه مستقیم این است که با توجه به همه کدهای قبلی که تا کنون در برنامه بازدید شده‌اند، احتمال مشاهده کد دستور o_i را تخمین بزنیم. این احتمال در رابطه (۳) آمده است.

$$\Pr(o_i|o_1, o_2, \dots, o_{i-1}) \qquad (۳)$$

هدف ما این است که بازنمایی اصلی P را به دست آوریم، بنابراین از یک تابع نگاشت $\Phi: P \to R^{|p|\times d}$ استفاده می‌کنیم. این تابع نگاشت Φ نشان دهنده عملکرد مربوط به هر O است. بنابراین مسئله طبقه‌بندی بدافزارها تعیین احتمال طبق فرمول (۴) است:

$$Pr(o_i|(\varphi(o_1), \varphi(o_2), \dots, \varphi(o_{m-1}))) \qquad (۴)$$

با ترکیب هر دو دنباله از کدها و مدل‌های زبان عصبی، راهی را بیان می‌کنیم که ویژگی‌های موردنظر ما را برآورده کند. این روش نمونه‌هایی از بدافزارهایی را که در فضای بردار تعبیه شده‌اند نشان می‌دهد. این تعبیه‌سازی‌ها فرم‌های فراخوانی‌های API را رمزگذاری می‌کنند و ازآنجاکه این روش تعبیه‌سازی‌های مفیدی را خروجی می‌دهد، می‌تواند با تغییر ساختارهای بدافزار سازگار شود. تعبیه‌سازی آن ویژگی‌های کد دستوری را رمزگذاری می‌کنند. در این مسئله، مانند هر روش مدل‌سازی زبان عصبی، تنها ورودی موردنیاز، یک واژگان است. روش پیشنهادی مجموعه‌ای از توالی‌های کد دستوری را کدگذاری می‌کند که واژگان $(o_1, o_2, o_3, \dots, o_m)$ به‌عنوان واژگان بدافزار (V = P) دریافت می‌شود. به‌عنوان مثال، ما از پنج کد دستوری اول برنامه در شکل(۳) و اندازه تعبیه ۱۲۸ بیتی برای تعبیه‌سازی استفاده کرده ایم. خروجی الگوریتم Word2vec می‌تواند یک بردار با ابعاد ۱۲۸،۲۵۶ یا ۵۱۲ باشد. در نتایج تجربی این ابعاد خیلی در محاسبات موثر نبودند و به این دلیل ما از ابعاد ۱۲۸ استفاده کردیم.



شکل (۳): فرایند تعبیه‌سازی و تولید بردار

همان‌طور که در شکل (۳) دیده می‌شود، برای هرکدام از کدهای دستوری، بردار تعبیه‌شده را محاسبه می‌کنیم. این بردار به‌صورت پیش‌فرض ۱۲۸ بعدی است. بدین ترتیب ما بدافزارها را به‌صورت $S = \{z_1 = (x_1, y_1), , \dots, z_n = (x_n, y_n)\}$ با توجه به مجموعه‌ای از جفت‌های مرتب $\{(\overrightarrow{x_j}, y_j)\}$ از مجموعه داده‌ی بدافزار که بردار $\overrightarrow{x_j}$ ، موقعیت بدافزار است، نمایش می‌دهیم. در این نمایش هر بدافزار به‌صورت z_i نمایش می‌دهیم.

حال برای اینکه بردار نهایی کل فایل بدافزار را تولید کنیم، کل این بردارها را باهم جمع می‌زنیم تا اینکه بردار نهایی به‌صورت $Z = \sum_{i=1}^{n} z_i$ ایجاد گردد. بردار Z حاوی اطلاعات معنایی شامل اطلاعات نهفته و معنایی در کد مثل نحوه قرارگیری کدهای دستوری کنارهم، نحوه چیدمان فراخوانی‌های APIهای مختلف و ترتیب فراخوانی کدهای دستوری و مثل این‌ها است.

نحوه محاسبه شباهتی که کارپیشنهادی با مقاله[۳] دارد این است که در آنجا از simhash برای این مسئله استفاده شده است که با تغییر جزیی کدهای دستوری و حملات تخاصمی، می‌توان به‌راحتی این الگوریتم را فریب داد که این موضوع در کارهای تجربی نشان داده‌ایم. هم‌چنین الگوریتم word2vec برای محاسبه شباهت بهتر از سایر الگوریتم‌ها عمل می‌کند.

الگوریتم ۲ بصری‌سازی

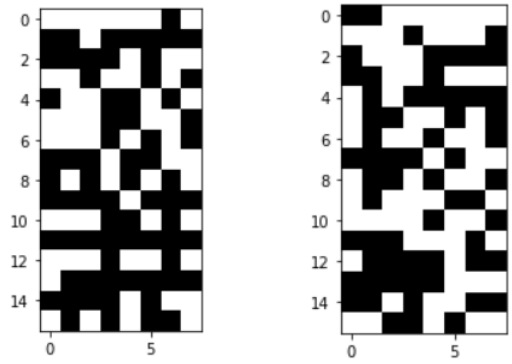
inputs:
set of d-bit Word2Vec algorithm word2vec(Z)
output:
image of all vectors;
1 **Begin**
6 **for** each Vector in Set do
7 I = I + wv[opcode]
8 end for
9 **for** i = 1 to n do
10 **if** I[i] > 0 then
11 I[i] = 1;
12 else
13 I[i] = 0;
14 end if
15 end for
16 end

۳-۶- بصری‌سازی بدافزار

شباهت بصری تصاویر بدافزار ما را برانگیخت تا به طبقه‌بندی بدافزار با استفاده از تکنیک‌های بینایی رایانه نگاه کنیم. همان‌طور که می‌دانیم امروزه طبقه‌بندی اشیا مبتنی بر تصویر به‌خوبی مورد مطالعه قرار گرفته است. تصاویر خانواده‌های خاصی از بدافزارها را می‌توان در شکل (۴) مشاهده کرد.

بردار تعبیه‌شده بدافزار در مرحله قبل (Z) باید به یک تصویر تبدیل شود. هر بدافزار با روشی یکسان با الگوریتم *word2vec* تعبیه‌سازی شده است. یک بدافزار تعبیه شده به‌عنوان یک بردار ۱۲۸ بیتی بیان شده و سپس در یک آرایه دوبعدی سازماندهی می‌شود. فرایند تبدیل هم به این صورت است که یک بردارد ۱۲۸تایی را به یک ماتریس ۱۶×۸ تبدیل می‌کنیم. یعنی در هر سطر ۱۶ بیت قرار می‌دهیم. هر سطر شامل ۸ بیت است؛ بنابراین بیت‌های بردار Z را به تصویر مقیاس خاکستری[1] تبدیل می‌کنیم. ابعاد ماتریس تصویر مقیاس خاکستری به‌اندازه تعبیه شده (d) بستگی دارد؛ بنابراین، تغییرات بدافزار از گروه معادل را می‌توان با روش بینایی ماشین شناسایی کرد. علاوه بر موجود بودن اطلاعات معنایی حال می‌توان اطلاعات شهودی و تصویری را نیز به بردار Z اضافه کرد و آن را به تصویر تبدیل کنیم برای آن که بتوانیم اطلاعات متمایزسازی بیشتری را برای تشخیص بدافزارها تولید کنیم.

[1] grayscale



الف- بدافزار خانواده ۱ ب- بدافزار خانواده ۲

شکل (۴): نمونه تصاویر تولید شده برای دو خانواده متفاوت

همان‌طور که در شکل (۴) دیده می‌شود تصاویری که تولید شده است برای دو خانواده مختلف از بدافزار است. به‌صورت شهودی روش پیشنهادی توانسته است که این تفاوت را ایجاد کند که دو بدافزار از دو خانواده متفاوت الگوی بصری متفاوتی داشته باشند. همان‌طور که واضح است، یک روشی کارآمد است که بتواند الگوهای بصری متفاوت‌تری را برای بدافزارهای مختلف ایجاد کند تا بتوان شباهت بین بدافزارها را به‌صورت مؤثرتری محاسبه کرد. حال برای اینکه این الگوها را استخراج کنیم از الگوریتم یادگیری عمیق *CNN* استفاده می‌کنیم.

۳-۷- آموزش *CNN*

شبکه‌عصبی‌کانولوشنی(*Convolutional Neural Network*) نوعی الگوریتم یادگیری عمیق است که شامل عملیات کانولوشن است که می‌تواند یک ماتریس ورودی را به‌عنوان یک تصویر در نظر گرفته و پارامترهای قابل یادگیری را به ویژگی‌های مختلف ماتریس اختصاص دهد. این روش یکی از روش‌های معمول بینایی رایانه‌ای و یادگیری عمیق است [۴۱، ۴۲]. این معماری بر اساس شبکه عصبی مصنوعی معمولی عمل می‌کند. از این مدل برای تجزیه‌وتحلیل تصویر، طبقه‌بندی تصویر، سیستم‌های توصیه‌گر، مدل‌سازی زبان طبیعی، طبقه‌بندی بدافزارها و غیره استفاده شده است [۴۳-۴۵]. برخلاف شبکه‌های عصبی کاملاً متصل، *CNN*ها عمیق‌تر و بزرگ‌تر شده‌اند تا عملکرد بهتری را برای به‌دست‌آوردن ویژگی‌های سطح بالا مانند ویژگی‌های معنایی از ماتریس ورودی داشته باشند [۴۶]. ما با ارائه یک روش کارآمد برای مدل‌سازی بدافزار شروع می‌کنیم. همان‌طور که در شکل (۲) نشان‌داده‌شده است، روش پیشنهادی، تعبیه‌سازی بدافزار را به‌عنوان ورودی در نظر می‌گیرد و معنای بدافزار را از طریق لایه‌های کانولوشن و تجمیع تا رسیدن به یک نمایش با طول مشخص در لایه نهایی خلاصه می‌کند. در مدل‌های کانولوشنی، معمولاً از واحدهای کانوولوشن، با "*receptive fiel* " محلی و وزن‌های مشترک استفاده می‌شود. *CNN*، بردارهای ورودی را از

نمونه‌های تعبیه‌شده دریافت می‌کند. سپس اندازه این بردارها به یک اندازه تغییر می‌کند. بردارهای معادل به یک لایه *pooling* ارسال می‌شوند. خروجی‌های آخرین لایه‌های *pooling* به یک ساختار عمودی تبدیل می‌شوند و سپس به لایه *feedforward* و *SoftMax* داده می‌شوند. هر یک از لایه‌های *feedforward* شامل ۱۲۸ نورون پنهان است. در اینجا، تابع فعال‌سازی *Relu* را برای لایه مخفی و تابع *softmax* را برای آخرین لایه در شبکه اعمال می‌کنیم.

۴- نتایج آزمایش‌های تجربی

در این بخش، عملکرد روش پیشنهادی خود را بر روی مجموعه داده‌های بدافزار اندازه‌گیری می‌کنیم. برای این منظور، از مجموعه داده *kaggel* [۴۷] برای ارزیابی استفاده می‌کنیم. این مجموعه داده در سایت این سامانه قرار داده شده است. این مجموعه داده یک استاندارد برای اندازه‌گیری روش‌های مختلف تشخیص بدافزار است که بسیاری از مقالات پژوهشی از این مجموعه داده استفاده می‌کنند.

۴-۱- تنظیمات آزمایش‌های تجربی

برای انجام این مطالعه تجربی، ما روش پیشنهادی را با استفاده از پلتفرم *TensorFlow* با *Keras API* پیاده‌سازی کردیم. محیط سخت‌افزار یک پردازنده *Intel (R) Core (TM) i7-10750H* با ۱۶ گیگابایت حافظه اصلی است. ما الگوریتم طبقه‌بندی بدافزار پیشنهادی را با پنج الگوریتم مقایسه می‌کنیم که عبارت‌اند از: *MalConv* [۴۹]، *DeepConv* [۵۰]، *Structural entropy CNN* [۵۱]، *Multiresolution CNN* [۵۱] و *MCSC* [۳]. ما از ۱۰ اعتبارسنجی متقابل برای ارزیابی عملکرد طبقه‌بندی بدافزار روش پیشنهادی استفاده می‌کنیم. ما از *10-fold cross-validation* که یک روش آماری است برای ارزیابی و مقایسه توانایی مدل خود استفاده می‌کنیم. این روش داده‌ها را به بخش‌های آموزش و تست تقسیم می‌کند.

۴-۲- مجموعه داده‌ها و معیارها

این بخش الگوریتم تشخیص بدافزار را با انجام آزمایش‌هایی در زمینه چالش طبقه‌بندی بدافزارها توسط مجموعه داده *Kaggle Microsoft 2015* [۴۸] ارزیابی می‌کند. این مجموعه داده شامل ۱۰،۸۶۸ مورد بدافزار دارای برچسب و ۹ خانواده متفاوت بدافزار از مجموعه داده‌های دارای برچسب است که به طور عمومی در سایت *Kaggle* در دسترس است. هر فایل بدافزار دارای یک شناسه یکتا و یک شناسه کلاس است که این شناسه یک عدد صحیح است که یکی از ۹ شناسه خانواده مربوط به فایل برنامه را نشان می‌دهد (جدول (۱) را ببینید). برای هر فایل بدافزار، مجموعه داده، شامل نمایش هگزادسیمال است. این شامل محتوای باینری فایل بدافزار بدون هدر است.

جدول (۱): جزئیات خانواده‌های مختلف مجموعه داده Kaggle

Family Name	Train Samples	Type
Ramnit	۱۵۴۱	Worm
Lollipop	۲۴۷۸	Adware
Kelihos_ver3	۲۹۴۲	Backdoor
Vundo	۴۷۵	Trojan
Simda	۴۲	Backdoor
Tracur	۷۵۱	TrojanDownloader
Kelihos_ver1	۳۹۸	Backdoor
Obfuscator.ACY	۱۲۲۸	Any kind of obfuscated malware
Gatak	۱۰۱۳	Backdoor

در مرحله اول، پیش‌پردازش با تمیزکردن و تجزیه‌وتحلیل روش‌های مختلف در پایتون انجام می‌شود. در این مرحله کدهای اسمبلی دریافت شدند و کدهای اضافه از آن خارج گردید و خط‌به‌خط کدها خوانده شد و دستورات مربوط به قسمت کد برنامه استخراج گردید. هر بدافزار در مجموعه داده مورداستفاده مربوط به یک خانواده از بدافزار است. بدافزارهای مورداستفاده متعلق به ۹ خانواده مختلف هستند. آمار کامل در هر خانواده بدافزار در جدول (۱) نشان‌داده‌شده است.

جدول (۲): جزئیات نتایج روش پیشنهادی روی خانواده‌های مختلف

	Precision	Recall	F1-score
۱	۰/۹۷	۰/۹۷	۰/۹۷
۲	۰/۹۸	۱	۰/۹۹
۳	۱	۱	۱
۴	۰/۹۷	۱	۰/۹۸
۵	۱	۰/۹۸	۰/۹۹
۶	۰/۹۷	۰/۹۷	۰/۹۷
۷	۰/۹۹	۰/۹۹	۰/۹۹
۸	۰/۹۹	۰/۹۷	۰/۹۸
۹	۱	۰/۹۶	۰/۹۸

جدول (۳): مقایسه روش پیشنهادی با سایر روش‌ها

approaches	Accuracy	Macro F1 score
CNN Entropy [51]	۰/۹۷۰۸	۰/۹۳۱۴
Autoencoder [52]	۰/۹۸۶۱	۰/۹۷۱۹
MalConv [53]	۰/۹۶۴۱	۰/۸۹۰۲
DeepConv [50]	۰/۹۷۵۶	۰/۹۰۷۱
MCSC [3]	۰/۹۸۸۶	۰/۹۸۰۷
Shallow CNN	۰/۹۹۰۳	۰/۹۷۴۳
Our method	**۰/۹۸۹۶**	**۰/۹۸۶۵**

۴-۳- معیار ارزیابی

برای ارزیابی نتایج، از معیارهای *precision*، *F1* ، *recall* استفاده می‌کنیم که دارای تعاریف زیر است:

$$precision = \frac{TP}{TP + FP}$$

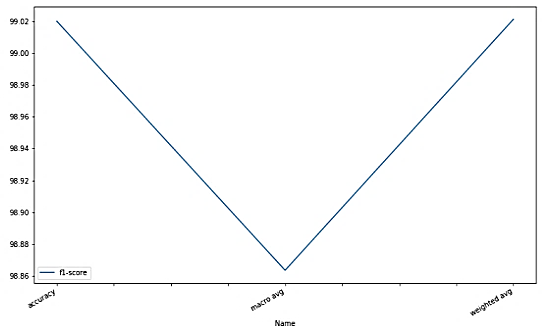
$$Recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times precision \times Recall}{recision + Recall}$$

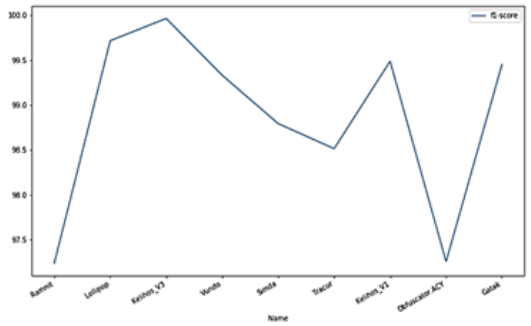
در اینجا ، FP تعداد پیش‌بینی‌های مثبت کاذب، FN تعداد پیش‌بینی‌های منفی کاذب و TP تعداد پیش‌بینی‌های مثبت واقعی است. امتیاز $F1$ میانگین وزنی دقت و فراخوانی است.

۵-۳- تحلیل آزمایش‌های تجربی

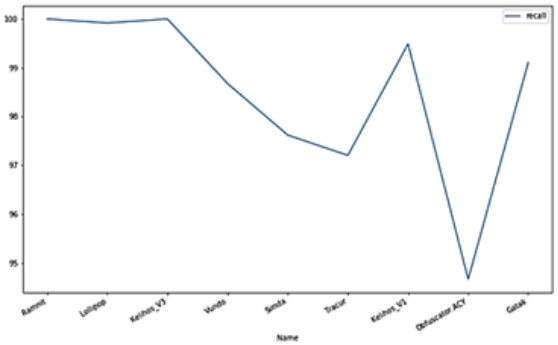
همان‌طور که در جدول ۲ دیده می‌شود روش پیشنهادی را روی مجموعه داده ذکر شده و ۹ خانواده مختلف بدافزار اجرا کردیم که نتایج روش پیشنهادی در این جدول آورده شده است. در این جدول سعی کردیم که نتایج را به تفکیک هر خانواده ارائه دهیم که مشخص شود روش پیشنهادی در کدام خانواده خوب عمل کرده و در کدام خانواده ضعیف عمل کرده است. در جدول ۳ نتایج روش‌های پیشین و روش پیشنهادی در مقایسه با هم آورده شده است. ما مسئله طبقه‌بندی چندکلاسه را به‌عنوان چندین مسئله طبقه‌بندی باینری در نظر می‌گیریم. در مسائل طبقه‌بندی، *precision* و *recall* روش‌های رایج ارزیابی هستند. *precision* احتمال مرتبط بودن یک نمونه بازیابی شده (به طور تصادفی) است. *recall* احتمال این است که یک نمونه مرتبط (به طور تصادفی انتخاب شده) بازیابی شود. همان‌طور که دیده می‌شود روش پیشنهادی از منظر معیار *Accuracy* دارای مقدار ۰/۹۸۹۶ و از منظر معیار *Macro F1 score* دارای مقدار ۰/۹۸۶۵ می‌باشد. همان‌طور که مشهود است از منظر معیار امتیاز *F1* روش پیشنهادی به نسبت سایر روش‌ها دارای عملکرد بهتری است. شکل‌های ۵ معیارهای *Precision* ، *Recall* ، *F1 score* و روش پیشنهادی را نمایش می‌دهد. ازآنجایی‌که تنها ۴۲ نمونه در خانواده *Simda* وجود دارد، اکثر الگوریتم‌ها نمی‌توانند *Simda* را به‌خوبی تشخیص دهند. شایان‌ذکر است که الگوریتم پیشنهادی همچنان عملکرد خوبی را حفظ می‌کند. در خانواده‌هایی که روش پیشنهادی ضعیف عمل کرده تعداد نمونه کم بوده و آن خانواده یکی از بدافزارهایی بوده است که از مکانیزم‌های مخفی‌کردن رفتار اصلی خود استفاده کرده است. به همین علت خیلی از مدل‌ها نتوانسته‌اند که به‌درستی این خانواده را تشخیص دهند. همان‌طور که در شکل‌های ۵-ب و ۵-ج و ۵-ه نمایش داده شده است در این خانواده روش پیشنهادی هم دارای نقص است ولی در قیاس با سایر روش‌ها عملکرد بهتری دارد. شکل ۵-ب به تفکیک هر کدام از ۹ خانواده را نمایش می‌دهد. به این معنی که هر خانواده که تعداد هرکدام از آن‌ها در جدول ۱ آورده شده است را مورد آزمایش قرار دادیم و نتایج خروجی هر خانواده را ثبت کردیم. در مابقی اشکال هم به همین ترتیب نمودارهای *recall* کلی و *recall* به تفکیک هر خانواده و *precision* کلی و *precision* به تفکیک هر خانواده آورده شده است.



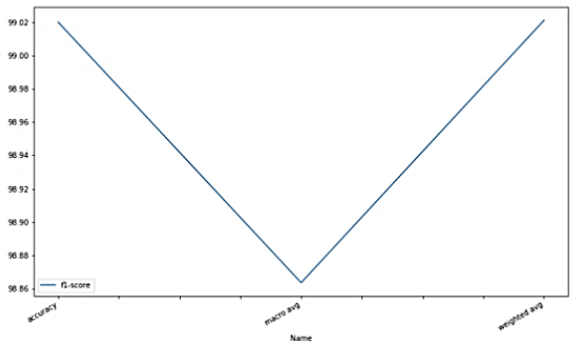
الف) نمودار recall روش پیشنهادی



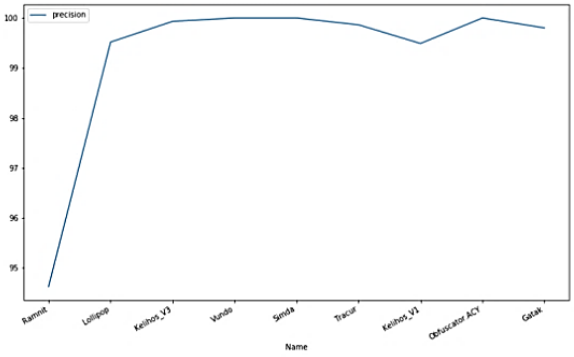
ب) نمودار F1_Score_all بر روی ۹ خانواده



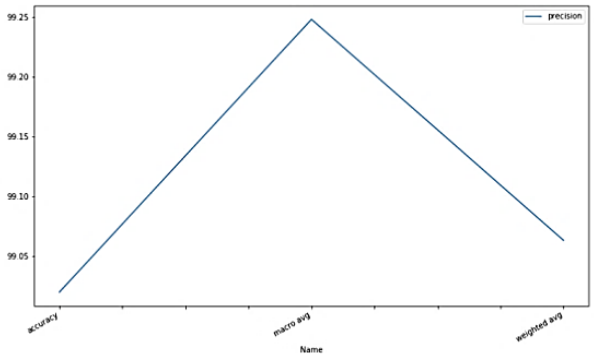
ج) نمودار Recall_all بر روی ۹ خانواده



د) نمودار F1_Score



ه) نمودار Precision_all روی ۹ خانواده



و) نمودار precision

شکل (۵): مقایسه سه معیار accuracy، macro F1 و میانگین وزن‌دار روش پیشنهادی روی خانواده‌های مختلف بدافزار

۶-۳- تأثیر حملات خصمانه بر دقت طبقه‌بندی

مدل‌های تشخیص بدافزار پایه با انجام حملات خصمانه بر روی آن‌ها را می‌توان از نظر استحکام بررسی کرد. ما یک حمله برای تولید نمونه‌های متخاصم با افزودن تعداد کمی اختلالات هوشمند در نمونه‌های بدافزار استفاده کردیم. در این حالت، سعی کردیم که با تغییر بعضی کدهای بدون استفاده و همچنین حذف بعضی کدها به فایل بدافزار حمله کنیم. این فرایند حمله نیازمند مراقبت ویژه در ساخت باینری است که ساختار بدافزار دچار تخریب نشود و سعی شود که حمله به‌گونه‌ای صورت گیرد که منطق و عملکرد برنامه مورد حمله قرار گرفته مشابه نسخه اصلی باشد. عملکرد حمله خصمانه را می‌توان با نرخ فریب، دقت و غیره اندازه‌گیری کرد. هدف حمله خصمانه کاهش دقت مدل‌های تشخیص بدافزار پایه با اضافه‌کردن اختلالات هوشمند در نمونه‌های بدافزار برای طبقه‌بندی اشتباه است و در نتیجه نرخ فریب را افزایش می‌دهد.

جدول (۴): مقایسه روش پیشنهادی با سایر روش‌ها در برابر حمله تخاصمی

نام روش	دقت قبل از حمله	دقت بعد از حمله
CNN Entropy [51]	۰/۹۷۰۸	۰/۹۳۱۴
Autoencoder [52]	۰/۹۸۶۱	۰/۸۴۹۶
MalConv [53]	۰/۹۶۴۱	۰/۷۸۹۵
DeepConv [50]	۰/۹۷۵۶	۰/۸۶۹۸
MCSC [3]	۰/۹۸۸۶	۰/۸۹۶۵
Shallow CNN	۰/۹۹۰۳	۰/۹۰۲۵
Our method	**۰/۹۸۹۶**	**۰/۹۱۸۵**

همان‌طور که در جدول (۴) مشاهده می‌شود با وجود انجام حمله به مدل‌های مختلف تشخیص بدافزار، در حالت برابر، روش پیشنهادی دارای عملکرد بهتری است. باتوجه‌به نتایج مشاهده می‌شود که روش پیشنهادی مقاومت بیشتر از خود در برابر حمله تزریق کد نشان می‌دهد. دلیل مقاومت بالای روش پیشنهادی در برابر حمله تخاصمی، وجود روش استخراج اطلاعات معنایی در کنار ویژگی‌های بصری که الگوریتم CNN این را استخراج می‌کند، در روش پیشنهادی است. در کنار هم بودن این دو مفهوم باعث شده است که نقطه تاب‌آوری روش پیشنهادی به نسبت سایر مدل‌های تشخیص بدافزار بالاتر باشد.

۷- نتیجه‌گیری

در این مقاله، ما یک الگوریتم طبقه‌بندی بدافزار را پیشنهاد کردیم که بصری‌سازی بدافزار و تکنیک‌های یادگیری عمیق را ترکیب می‌کند. دو مفهوم تعبیه‌سازی و بصری‌سازی در طبقه-بندی بدافزارها دو سهم قابل‌توجه در طبقه‌بندی چند کلاسه بدافزارها ارائه می‌دهد. در مرحله اول، تعبیه‌سازی بدافزار یک تعمیم معنایی را ارائه می‌دهد که می‌تواند بر مشکلات مربوط به روابط پیچیده بین کلاس‌ها فایق آید. ثانیاً در بحث بصری‌سازی به ما یادآوری می‌کند که ساخت تصویر می‌تواند برای گسترش هر الگوریتم یادگیری، برای طبقه‌بندی بدافزارها در تنظیمات چندطبقه استفاده شود. در روش پیشنهادی در ابتدا، دنباله‌های کد عملیاتی را از بدافزار استخراج می‌کند و با حفظ ویژگی‌های یکتای بدافزار به طول مساوی با *Word2Vec* کدگذاری می‌کند. با درنظرگرفتن هر مقدار بیت ماحصل تعبیه‌سازی، به‌عنوان یک پیکسل، بیت‌های *Word2Vec* را می‌توان به تصاویر سیاه سفید تبدیل کرد. در ادامه، یک شبکه عصبی کانولوشن برای آموزش تصاویر و شناسایی خانواده‌های بدافزار اتخاذ شد. بخش ۵ نشان داد که روش تعبیه یادگیری نسبت به روش دیگر در مجموعه داده‌های دنیای واقعی و مصنوعی برتری دارد. نتایج تجربی، توانایی تشخیص عالی روش پیشنهادی را نشان می‌دهد. دقت طبقه‌بندی حداکثر ۱۰۰ درصد و میانگین ۰,۹۸۹۶ در مجموعه داده بدافزار از ۱۰۸۰۵ نمونه است که بالاتر از سایر الگوریتم‌های

قرار گرفته است. یکی از نقاط ضعف روش پیشنهادی این است که اگر تعداد نمونه کم باشد روش پیشنهادی دارای عملکردی ضعیف است. همچنین اگر بدافزار از سازوکارهای مخفی‌کردن رفتار اصلی خود استفاده کند منجر به ضعیف‌شدن نتیجه خروجی روش پیشنهادی می‌شود. اگرچه ما برخی شواهد تجربی از رویکرد پیشنهادی را دیده‌ایم، اما این مسئله به مطالعات بیشتری در جهت‌های زیر نیاز دارد و پیشنهاد می‌شود که محققین در این راستاها مطالعه کنند.

(۱) استفاده از روش پیشنهادی در محیط‌های کاربردی در مقیاس بزرگ.
(۲) مطالعه بیشتر برای شناسایی مؤثر، برای زمانی که بدافزار از تکنیک‌های ضد اشکال‌زدایی استفاده می‌کند.
(۳) مطالعه برای حوزه فریب روش‌های بصری‌سازی و مقاوم‌سازی آن‌ها در برابر این حملات.
(۳) شناسایی و طبقه‌بندی سریع‌تر بدافزار با استفاده از محاسبات با کارایی بالا بر اساس *GPU* یا سایر تکنیک‌های موازی‌سازی.
(۴) ادغام روش ایستای پیشنهادی با تحلیل پویا برای گسترش استحکام و سازگاری سیستم تشخیص.

۸- مراجع

[1] A. Shalaginov, S. Banin, A. Dehghantanha, and K. Franke, "Machine learning aided static malware analysis: A survey and tutorial," *Cyber threat intelligence*, pp. 7-45, 2018.

[2] Z. Sun *et al.*, "An opcode sequences analysis method for unknown malware detection," in *Proceedings of the 2019 2nd international conference on geoinformatics and data analysis*, 2019, pp. 15-19.

[3] S. Ni, Q. Qian, and R. Zhang, "Malware identification using visualization images and deep learning," *Computers & Security*, vol. 77, pp. 871-885, 2018.

[4] W. Han, J. Xue, Y. Wang, L. Huang, Z. Kong, and L. Mao, "MalDAE: Detecting and explaining malware based on correlation and fusion of static and dynamic characteristics," *computers & security*, vol. 83, pp. 208-233, 2019.

[5] G. G. Sundarkumar, V. Ravi, I. Nwogu, and V. Govindaraju, "Malware detection via API calls, topic models and machine learning," in *2015 IEEE International Conference on Automation Science and Engineering (CASE)*, 2015: IEEE, pp. 1212-1217.

[6] A. Kumar, K. Kuppusamy, and G. Aghila, "A learning model to detect maliciousness of portable executable using integrated feature set," *Journal of King Saud University-Computer and Information Sciences*, vol. 31, no. 2, pp. 252-265, 2019.

[7] B. Jung, T. Kim, and E. G. Im, "Malware classification using byte sequence information," in *Proceedings of the 2018 Conference on Research in Adaptive and Convergent Systems*, 2018, pp. 143-148.

[8] S. Jain and Y. K. Meena, "Byte level n–gram analysis for malware detection," in *Computer Networks and Intelligent Computing: 5th International Conference on Information Processing, ICIP 2011, Bangalore, India, August 5-7, 2011. Proceedings*, 2011: Springer, pp. 51-59.

[9] M. El Boujnouni, M. Jedra, and N. Zahid, "New malware detection framework based on N-grams and support vector domain description," in *2015 11th international conference on information assurance and security (IAS)*, 2015: IEEE, pp. 123-128.

[10] T. Wuechner, A. Cisłak, M. Ochoa, and A. Pretschner, "Leveraging compression-based graph mining for behavior-based malware detection," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 1, pp. 99-112, 2017.

[11] A. Damodaran, F. D. Troia, C. A. Visaggio, T. H. Austin, and M. Stamp, "A comparison of static, dynamic, and hybrid analysis for malware detection," *Journal of Computer Virology and Hacking Techniques*, vol. 13, pp. 1-12, 2017.

[12] M. Christodorescu and S. Jha, "Static analysis of executables to detect malicious patterns," in *12th USENIX Security Symposium (USENIX Security 03)*, 2003.

[13] M. Egele, T. Scholte, E. Kirda, and C. Kruegel, "A survey on automated dynamic malware-analysis techniques and tools," *ACM computing surveys (CSUR)*, vol. 44, no. 2, pp. 1-42, 2008.

[14] M. Akour, I. Alsmadi, and M. Alazab, "The malware detection challenge of accuracy," in *2016 2nd International Conference on Open Source Software Computing (OSSCOM)*, 2016: IEEE, pp. 1-6.

[15] S. D. Nikolopoulos and I. Polenakis, "A graph-based model for malware detection and classification using system-call groups," *Journal of Computer Virology and Hacking Techniques*, vol. 13, no. 1, pp. 29-46, 2017.

[16] A. P. Namanya, I. U. Awan, J. P. Disso, and M. Younas, "Similarity hash based scoring of portable executable files for efficient malware detection in IoT," *Future Generation Computer Systems*, vol. 110, pp. 824-832, 2020.

[17] C.-I. Fan, H.-W. Hsiao, C.-H. Chou, and Y.-F. Tseng, "Malware detection systems based on API log data mining," in *2015 IEEE 39th annual computer software and applications conference*, 2015, vol. 3: IEEE, pp. 255-260.

[18] K. Griffin, S. Schneider, X. Hu, and T.-c. Chiueh, "Automatic Generation of String Signatures for Malware Detection," in *RAID*, 2009, vol. 5758: Springer, pp. 101-120.

[19] M. G. Schultz, E. Eskin, F. Zadok, and S. J. Stolfo, "Data mining methods for detection of new malicious executables," in *Proceedings 2001 IEEE Symposium on Security and Privacy. S&P 2001*, 2000: IEEE, pp. 38-49.

[20] X. Liu, Q. Lei, and K. Liu, "A graph-based feature generation approach in Android malware detection with machine learning techniques," *Mathematical Problems in Engineering*, vol. 2020, 2020.

Computer Science and Ubiquitous Computing: CSA-CUTE 17, 2018: Springer, pp. 1352-1357.

[36] M. S. I. Sajid, J. Wei, M. R. Alam, E. Aghaei, and E. Al-Shaer, "Dodgetron: Towards autonomous cyber deception using dynamic hybrid analysis of malware," in *2020 IEEE Conference on Communications and Network Security (CNS)*, 2020: IEEE, pp. 1-9.

[37] D. Bilar, "Opcodes as predictor for malware," *International journal of electronic security and digital forensics*, vol. 1, no. 2, pp. 156-168, 2007.

[38] J. Firth, "A synopsis of linguistic theory, 1930-1955," *Studies in linguistic analysis*, pp. 10-32, 1957.

[39] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[40] A. O. Salau and S. Jain, "Feature extraction: a survey of the types, techniques, applications," in *2019 international conference on signal processing and communication (ICSC)*, 2019: IEEE, pp. 158-164.

[41] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[42] J. Gu *et al.*, "Recent advances in convolutional neural networks," *Pattern recognition*, vol. 77, pp. 354-377, 2018.

[43] J. Jeon, J. H. Park, and Y.-S. Jeong, "Dynamic analysis for IoT malware detection with convolution neural network model," *IEEE Access*, vol. 8, pp. 96899-96911, 2020.

[44] X. Meng *et al.*, "MCSMGS: malware classification model based on deep learning," in *2017 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, 2017: IEEE, pp. 272-275.

[45] E. K. Kabanga and C. H. Kim, "Malware images classification using convolutional neural network," *Journal of Computer and Communications*, vol. 6, no. 1, pp. 153-158, 2017.

[46] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning. nature, 521 (7553), 436-444," *Google Scholar Google Scholar Cross Ref Cross Ref*, p. 25, 2015.

[47] R. Ronen, M. Radu, C. Feuerstein, E. Yom-Tov, and M. Ahmadi, "Microsoft malware classification challenge," *arXiv preprint arXiv:1802.10135*, 2018.

[48] M. Kalash, M. Rochan, N. Mohammed, N. D. Bruce, Y. Wang, and F. Iqbal, "Malware classification with deep convolutional neural networks," in *2018 9th IFIP international conference on new technologies, mobility and security (NTMS)*, 2018: IEEE, pp. 1-5.

[49] E. Raff, J. Barker, J. Sylvester, R. Brandon, B. Catanzaro, and C. Nicholas, "Malware detection by eating a whole exe," *arXiv preprint arXiv:1710.09435*, 2017.

[50] M. Krčál, O. Švec, M. Bálek, and O. Jašek, "Deep convolutional malware classifiers can learn from raw executables and labels only," 2018.

[51] D. Gibert, C. Mateu, J. Planes, and R. Vicens, "Classification of malware by using structural entropy on convolutional neural networks," in *Proceedings of the AAAI conference on artificial intelligence*, 2018, vol. 32, no. 1.

[21] E. Amer, S. El-Sappagh, and J. W. Hu, "Contextual identification of windows malware through semantic interpretation of api call sequence," *Applied Sciences*, vol. 10, no. 21, p. 7673, 2020.

[22] R. Sihwail, K. Omar, and K. Z. Ariffin, "A survey on malware analysis techniques: Static, dynamic, hybrid and memory analysis," *Int. J. Adv. Sci. Eng. Inf. Technol*, vol. 8, no. 4-2, pp. 1662-1671, 2018.

[23] V. Verma, S. K. Muttoo, and V. Singh, "Multiclass malware classification via first-and second-order texture statistics," *Computers & Security*, vol. 97, p. 101895, 2020.

[24] R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, and S. Venkatraman, "Robust intelligent malware detection using deep learning," *IEEE Access*, vol. 7, pp. 46717-46738, 2019.

[25] J. Fu, J. Xue, Y. Wang, Z. Liu, and C. Shan, "Malware visualization for fine-grained classification," *IEEE Access*, vol. 6, pp. 14510-14523, 2018.

[26] L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath, "Malware images: visualization and automatic classification," in *Proceedings of the 8th international symposium on visualization for cyber security*, 2011, pp. 1-7.

[27] Y. Ye, T. Li, D. Adjeroh, and S. S. Iyengar, "A survey on malware detection using data mining techniques," *ACM Computing Surveys (CSUR)*, vol. 50, no. 3, pp. 1-40, 2017.

[28] A. Hellal and L. B. Romdhane, "Minimal contrast frequent pattern mining for malware detection," *Computers & Security*, vol. 62, pp. 19-32, 2016.

[29] A. Narayanan, M. Chandramohan, L. Chen, and Y. Liu, "A multi-view context-aware approach to Android malware detection and malicious code localization," *Empirical Software Engineering*, vol. 23, pp. 1222-1274, 2018.

[30] K. Han, J. H. Lim, and E. G. Im, "Malware analysis method using visualization of binary files," in *Proceedings of the 2013 Research in Adaptive and Convergent Systems*, 2013, pp. 317-321.

[31] D. Vasan, M. Alazab, S. Wassan, H. Naeem, B. Safaei, and Q. Zheng, "IMCFN: Image-based malware classification using fine-tuned convolutional neural network architecture," *Computer Networks*, vol. 171, p. 107138, 2020.

[32] J. Zhang, Z. Qin, H. Yin, L. Ou, S. Xiao, and Y. Hu, "Malware variant detection using opcode image recognition with small training sets," in *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, 2016: IEEE, pp. 1-9.

[33] H. HaddadPajouh, A. Dehghantanha, R. Khayami, and K.-K. R. Choo, "A deep recurrent neural network based approach for internet of things malware threat hunting," *Future Generation Computer Systems*, vol. 85, pp. 88-96, 2018.

[34] S. Jha, D. Prashar, H. V. Long, and D. Taniar, "Recurrent neural network for detecting malware," *computers & security*, vol. 99, p. 102037, 2020.

[35] H.-J. Kim, "Image-based malware classification using convolutional neural network," in *Advances in*

[52] V. Kůrková, Y. Manolopoulos, B. Hammer, L. Iliadis, and I. Maglogiannis, *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III*. Springer, 2018.

[53] S. Quynn, "Identifying behaviors in executable binaries with Deep Learning," Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 2021.