Metamorfik Zararlı Yazılımların Derin Öğrenme(LSTM) ile Sınıflandırılması
Classification of Methamorphic Malware with Deep Learning(LSTM)

Ahmet Faruk YAZI
Bilgi Güvenliği Mühendisliği
İstanbul Şehir Üniversitesi
İstanbul, Türkiye
ahmetyazi@std.sehir.edu.tr

Ferhat Özgür ÇATAK
BİLGEM
TUBİTAK
Kocaeli, Türkiye
ozgur.catak@tubitak.gov.tr

Ensar GÜL
Bilgi Güvenliği Mühendisliği
İstanbul Şehir Üniversitesi
İstanbul, Türkiye
ensargul@sehir.edu.tr

***Özetçe* —Günümüzde geleneksel imza tabanlı tespit yöntemleri kullanan anti-virus uygulamaları, metamorfik zararlı yazılımları tespit etmede başarısızdır. Bu nedenle son zamanlarda yapılan tespit ve sınıflandırmaya yönelik çalışmalar, zararlı yazılımların davranışlarını ele almaktadır. Bu çalışma kapsamında, 8 farklı türdeki gerçek zararlı yazılımların API çağrıları kullanılarak, LSTM tabanlı bir sınıflandırma yöntemi geliştirilmiştir. Bu yöntem ile işletim sistemi üzerindeki zararlı yazılım türlerine ait davranışlar modellenmiştir.**

Anahtar Kelimeler—Metamorfik zararlı yazılımlar, Windows API, derin öğrenme, LSTM.

***Abstract*—Nowadays, anti-virus applications using traditional signature-based detection methods fail to detect metamorphic malware. For this reason, recent studies on the detection and classification of malicious software address the behavior of malware. In this study, an LSTM based classification method was developed by using API calls of 8 different types of real malware. With this method, the behaviors of the malware types on the operating system are modeled.**

Keywords—Metamorphic malware, Windows API, deep learning, LSTM.

I. GİRİŞ

Son yıllarda, zararlı yazılımların çok fazla artmasında ve yaygınlaşmasında ki en büyük faktör, büyük bir kitle tarafından çalışılan bir konu olmasıdır. Bu doğrultuda gerek bireysel çalışmalar, gerekse yetkin organizasyonel çalışmalar zararlı yazılımların sürekli gelişmesini sağlamaktadır. Özellikle 2018 yılı ilk 4 ayında, 40,000,000 adet[1] yeni zararlı yazılımın gün yüzüne çıktığı düşünülürse, bu doğrultuda çok büyük emekler sarfedildiği anlaşılmaktadır. Bu nedenle, zararlı yazılımların saldırılarından korunmak için de büyük çabalar gösterilmesi gerekmektedir.

Zararlı yazılımlarından etkili bir şekilde korunabilmek için, öncelikle zararlı yazılımların iyi tanınması ve davranışlarının iyi analiz edilmesi gerekmektedir. Bu doğrultuda sadece zararlı yazılımların tespit edilmesi yeterli olmamaktadır. Bu yazılımların başarı bir şekilde sınıflandırılması da gerekmektedir. Zararlı yazılımların sınıflandırılması, önemli bir konudur. Bir zararlı yazılımın ait olduğu sınıf, zararlı bir davranışı temsil etmektedir. Bu davranışlara karşı alınacak önlemler zararlı yazılımların sınıflarına göre farklılık göstermektedir.

Zararlı yazılımların gelişimleri incelendiğinde, zararlı yazılımların yetkinlik ve etkinliklerinin çok ciddi bir seviyeye ulaştıkları görülmektedir. Bu doğrultuda, zararlı yazılım ailesinin en gelişmiş üyesi olarak metamorfik zararlı yazılımlar karşımıza çıkmaktadır. Bu yazılımlar, güvenlik bileşenlerinden kendilerini, farklı yöntemler kullanarak, farklı zamanlarda kendi kodlarında yaptıkları değişiklikler ile saklayabilmektedirler. Bu sayede kod imzaları sürekli değişmektedir [1]. Bu nedenle, geleneksel imza tabanlı tespit yöntemleri kullanan anti-virus uygulamaları tarafından tespit edilememektedirler. Bu durumun bir sonucu olarak, metamorfik zararlı yazılımların türlerine göre sınıflandırılması da pek mümkün olmamaktadır.

Bir metamorfik zararlı yazılım, farklı ortamlarda farklı kod dizilimleri ile kendini gösterebilmesine rağmen, bütün ortamlarda aynı davranışı göstermek zorundadır. Çünkü belirli bir zararlı eylemi gerçekleştirmek için oluşturulmuştur. Bu bilgi doğrultusunda, metamorfik zararlı yazılımların tespiti ve sınıflandırılması için kullanılan yöntemlerin neredeyse tamamı, zararlı yazılımın yapısal özelliklerini değil, davranışsal özelliklerini ele almaktadırlar. Bu yöntemler ile, genellikle zararlı yazılımların davranışlarını temsil eden, sistem üzerinde yapmış oldukları API çağrıları gibi veriler önemli derecede kullanılmaktadır.

Herhangi bir yazılım tarafından, bir eylemi gerçekleştirmek için yapılan sistem API çağrılarının tamamı, bu yazılımın genel davranışını göstermektedir. Bu gibi davranışların doğru analiz edilmesi ile, davranışa sahip olan yazılımın zararlı bir yazılım olduğu, hangi sınıfa ait bir zararlı eyleme sahip olduğu bilgileri elde edilebilmektedir.

Zararlı yazılımların yapmış oldukları her bir API çağrısı bir veri niteliğinde olduğu gibi bu çağrıların yapılış sırası da önemli olmaktadır. Belirli API çağrılarının belirli sırada

[1] https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-jun-2018.pdf

978-1-7281-1904-5/19/$31.00 ~©~2019 IEEE

yapılması bir davranışı temsil etmektedir. Bu gibi zamana göre sıralı gelen verilerin işlenmesinde derin öğrenme yöntemlerinden uzun-kısa süreli bellek (long-short term memory - LSTM) oldukça yaygın kullanılmaya başlanmıştır.

Bu çalışmanın en önemli katkıları şu şekildedir:

- Zararlı yazılımlar ile ilgili yeni bir veri kümesi oluşturulmuştur. Bu alanda bu kapsamda bir veri kümesi bulunmamaktadır.
- Metamorfik yazılımlar yalıtılmış kumhavuzu ortamlarında çalıştırılarak analiz edilmiş ve API sıralamaları kayıt altına alınmıştır.
- Metin sınıflandırma için kullanılan LSTM algoritmasıyla bir metin sınıflandırma problemi olarak modellenmiş ve zararlı yazılım türü tespit modelleri oluşturulmuştur.
- Çok rastlanan zararlı yazılım türleri olan *Adware*, *Backdoor*, *Downloader*, *Dropper*, *Spyware*, *Trojan*, *Virus* ve *Worm* kullanılmıştır.

II. İlgili Çalışmalar

Klasik imza tabanlı tespit yöntemlerinin, polimorfik ve metamorfik zararlı yazılımları tespitinde yetersiz kalması sonucunda, bu yazılımların davranışlarını temel alan birçok çalışma yapılmaya başlanmıştır. Leder ve arkadaşları, metamorfik zararlı yazılımların geçirmiş oldukları yapısal değişimleri incelemişlerdir. VSA(Value Set Analysis) metodu kullanılarak, zararlı yazılımlardaki değişmeyen kod parçaları çıkarılarak bir tespit yöntemi geliştirmişlerdir [2].

Mehra ve arkadaşları çalışmalarında zararlı yazılımların kontrol akışları ve API çağrı grafik verilerini ele alarak tespit ve sınıflandırma yöntemi geliştirmişlerdir. Sınıflandırma zararlı yazılım ailelerine göre histogram ve Chi-square fark ölçüm formülleri kullanılarak yapılmıştır [3].

Ekhtoom ve arkadaşlarının çalışmalarında ise, metamorfik zararlı yazılım ailelerini belirlemek için sıkıştırma temelli bir sınıflandırma tekniği(Compression-Based Text Classification) kullanılmıştır. 13 farklı, gerçek zararlı yazılım ailesinden oluşan, yazılımlarının binary dosyaları kullanılarak bir veri kümesi oluşturulmuştur. Sınıflandırma için, bir zararlı yazılım dosyası ile farklı ailelere ait veri kümesindeki benzerlikler kullanılmıştır [4].

Prapulla ve arkadaşları ise çalışmalarını, metamorfik zararlı yazılımların yapısal olarak değişikliğe uğrasalarda, temel algoritma yapılarının korunduğu gerçeğine dayandırmışlardır. Bu doğrultuda, yazılım kodları incelenerek, davranışları temsil eden opcode özellikleri çıkarılmıştır. Bu veriler, makina öğrenmesi metodları kullanılarak bir yöntem geliştirilmiştir [5].

Pirscoveanu ve arkadaşları da, makine öğrenmesi metodlarından Random Forests algoritması kullanılarak bir sınıflandırma yöntemi geliştirmişlerdir. Zararlı yazılımların Windows API çağrıları temel sınıflandırma verileri olacak şekilde, DNS istekleri, Accessed Files, Mutexes, Registry Keys verileri kullanılarak zararlı yazılımların özellikleri çıkarılarak, yöntemin girdileri oluşturulmuştur [6].

Athiwaratkun ve Stokes de çalışmalarında, zararlı yazılımların API çağrılarını ele almışlardır. Derin öğrenme yöntemlerinden LSTM, GRU ve CNN kullanılarak farklı farklı tespit yöntemleri oluşturmuşlardır. Bu çalışmada LSTM tabanlı yöntemin, sıralı API çağrılarını analiz etmekte diğerlerine göre daha başarılı olduğu gösterilmektedir. [7].

Fang ve arkadaşları zararlı Javascript kodlarının tespit edilebilmesi için LSTM tabanlı bir yöntem geliştirmişlerdir. Bu yöntemde zararlı Javascript kodlarının, bytecode ve kelime vektörü(word vector) özellikleri çıkarılarak bir veri kümesi oluşturulmuştur. Ayıca bu çalışma içerisinde Random Forest and SVM algoritmalar kullanılarak da farklı farklı yöntemler geliştirilmiştir. LSTM tabanlı tespit yönteminin diğer yöntemlerinden daha başarılı olduğu gösterilmiştir [8].

Ahmed ve arkadaşlarının çalışmaları ise farklı bir yaklaşım sunmaktadır. Yaklaşımın temel mantığı; zararlı yazılımların tespit edilmesinde kullanılacak olan asıl veri, zararlı yazılımların nitelikleri değil, sistem üzerinde yaptıkları etkileridir. Bu nedenle önerdikleri tespit yönteminde, zararlı yazılımların yapısal ve davranışsal özellikleri ele alınmamaktadır. Olağan bir durumda ki sistem üzerinde, anormal olan davranışlar tespit edilmek istenmektedir. Bu sayede, polimorfik, metamorfik gibi gelişmiş zararlı yazılımların, sıfır gün(zero-day) zararlı yazılımların tespitinin yapılabileceği iddia edilmiştir [9].

III. Kullanılan Teknolojiler

Bu çalışma kapsamında; Windows API Çağrıları, Cuckoo Sandbox uygulaması, VirusTotal servisi ve LSTM derin öğrenme yöntemi kullanılmıştır.

A. Windows API Çağrıları

Windows API, Windows işletim sistemi üzerinde uygulama geliştirmek için sunulan bir arayüzdür. İşletim sistemi bir çok hizmetini API olarak sunmaktadır. Bir uygulama, işletim sisteminin sunduğu bir işlevi kullanabilmek için API fonksiyonlarını kullanılmasına API çağrısı denilmektedir.

B. Cuckoo Sandbox

Cuckoo Sandbox uygulaması, herhangi bir yazılımın gerçek bir ortamdaymış gibi çalıştırılmasına olanak sağlayan bir kumhavuzu uygulamasıdır. Bu uygulama ile zararlı yazılımların davranışları ve yapısal özellikleri analiz edilebilmektedir. Zararlı yazılımın API çağrıları, oluşturduğu ağ trafiği, üzerinde çalıştığı sistemin bellek dökümü gibi bilgileri sunmaktadır.

C. Virus Total Servisi

Virus Total Servisi dosyaları veya URL'leri analiz edebilen, internet üzerinden veya bir arayüz üzerinden(Virus Total Public API), çevrimiçi olarak hizmet veren bir servistir. Analiz işlemleri için 67 adet anti-virüs uygulama motoru kullanmaktadır. Herbir anti-virüs uygulama motoru bir analiz raporu oluşturmaktadır.

D. LSTM

LSTM, RNN tabanlı bir derin öğrenme yöntemidir. Uzun süreli bağımlılıkları rastgele aralıklarla hatırlayabilen, öğrenebilen bir mimariye sahiptir. Özellikle zamana göre sıralı bir şekilde gelen verileri veya belirli bir ilişkiye sahip olayları

analiz etmekte oldukça başarılı bir yöntemdir [10]. $\mathbf{x} = \{x_1, \cdots, x_T\}$ şeklinde ilerleyen sıralı bir veri olmak üzere, RNN, $\mathbf{h} = \{h_1, \cdots, h_T\}$ şeklinde ilerleyen gizli vektör sıralamasını ve $\mathbf{y} = \{y_1, \cdots, y_T\}$ şeklinde ilerleyen çıktı vektörü sıralamasını bulmaktadır. Bu hesaplama T adet yineleme ile şu şekilde bulunmaktadır.

$$\begin{aligned} h_t &= \mathcal{H}(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \\ y_t &= \mathcal{F}(W_{hy}h_t + b_y) \end{aligned} \tag{1}$$

W_{xh}, W_{hh}, ve W_{hy} matrisleri, eğitim zamanında hesaplanan bağlantı ağırlıklarıdır. $\mathcal{F}$ ise sigmoid aktivaston fonksiyonudur. Sigmoid şu şekilde tanımlanmıştır.

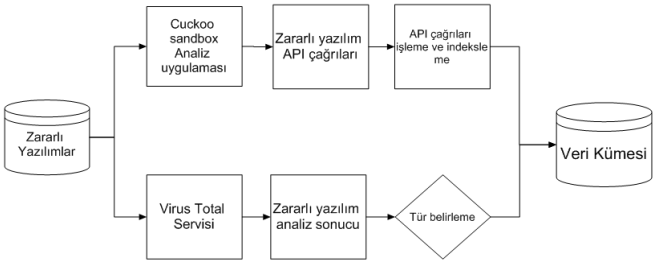
$$\mathcal{F} = \frac{e^{z_m}}{\sum_k e^{z_k}} \tag{2}$$

IV. YÖNTEM

Bu çalışmada, öncelikle bir veri kümesi oluşturulmuştur. Sonrasında ise, bu verileri kullanıarak sınıflandırma modellerinin oluşturulmasını sağlayan bir uygulama geliştirilmiştir.

A. Veri Kümesinin Oluşturulması

Veri kümesinde farklı türlerden toplam 7107 adet zararlı yazılımın davranış ve tür bilgileri bulunmaktadır. Şekil 1'de genel adımlar gösterilmektedir.



Şekil 1. Veri oluşturma

Veri kümesi oluşturulurken 20000' in üzerinde zararlı yazılım Cuckoo Sandbox uygulaması ile çalıştırılarak, zararlı yazılımların Windows işletim sistemi üzerinde yaptıkları API çağrı dizileri elde edilmiştir. Toplam 342 çeşit API çağrısı yapıldığı tespit edilmiştir. Bu çağrı dizileri indekslenerek veri kümesine eklenmiştir.

Virus Total Public API yardımı ile her bir zararlı yazılım dosyası analiz edilmiştir. Her analiz raporunun içerisinde yaklaşık 67 farklı anti-virus uygulama motorunun tespit bilgileri vardır. Bu bilgiler ayrı ayrı olarak işlenerek zararlı yazılımın ait olduğu tür bilgisi belirlenmiştir. Herbir zararlı yazılımın tür bilgisi, ilişkili API çağrıları ile birleştirilerek veri kümesi oluşturulmuştur. Veri kümesi, 379 adet Adware, 1001 adet Backdoor, 1001 adet Downloader, 891 adet Dropper, 832 adet Spyware,1001 adet Trojan, 1001 adet Virus, 1001 adet Worm türünde zararlı yazılım verileri içermektedir.

B. Model Oluşturma

Bu çalışma kapsamında geliştirilen uygulama, zararlı yazılım türlerine göre sınıflandırma yapmamızı sağlamaktadır. Bu uygulama ile sınıflandırmak istediğimiz 8 farklı tür için ayrı ayrı sınıflandırma modelleri oluşturulabilmektedir. Uygulama şu şekilde çalışmaktadır.

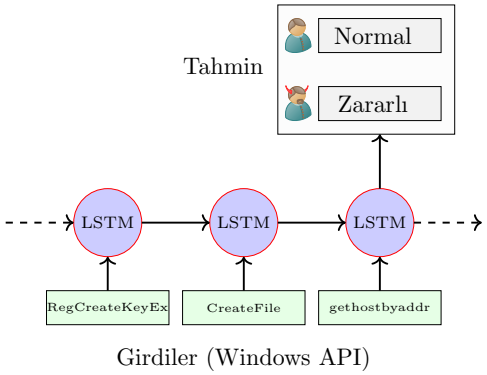
- Veri kümesinde bulunan verileri zararlı yazılım türü için işlemektedir. Modeli oluşturulmak istenilen zararlı yazılım türü bilgisine "1" etiketini, diğer tür bilgilerine ise "0" etiketini atamaktadır. Bu şekilde model çıktısı 0 (türe ait bir davranış değil) veya 1 (türe ait bir davranış) olarak karşımıza çıkacaktır.
- LSTM tabanlı bir sınıflandırma modeli oluşturmaktadır. Oluşturduğumuz veri kümesinin %60'ı eğitim %10'u ise test için ayrılmıştır.
- Yeni bir yazılımın API çağrıları her bir sınıflandırma modeline verilmekte, model sonuçlarının oylanması sonucuyla sınıf etiketi tespit edilmektedir.

Bu adımlar herbir zararlı yazılım türü için ayrı ayrı olarak işletilerek, istenilen modeller oluşturulmaktadır.

Bu uygulama Python programlama dili ve makine öğrenmesi kütüphaneleri olan Keras, Tensorow ve Scikit-learn kütüphaneleri kullanılmıştır. LSTM ağlarının kurulması için Keras kullanılmıştır. İki katmanlı bir LSTM yapısı oluşturulmuştur. Sınıflandırma modelleri, *tanh*, *sigmoid*, *softsign* ve *softmax* aktivasyon fonksiyonları kullanılarak ayrı ayrı olarak eğitilmiştir.

Model eğitimi süreci Şekil 2'de gösterilmiştir. LSTM ağ modeli her bir zararlı yazılımın Windows işletim sisteminde yaptığı API çağrılarını sırası ile almakta ve son adımda tahmin sınıf etiketini, $\hat{y}$ hesaplamaktadır. Her bir sınıf etiketi için model oluşturulmasından dolayı modellerimiz ikili sınıflandırıcılardır. Kayıp fonksiyonu olarak *log loss* kullanılmıştır. Log loss kayıp fonksiyonmu eşitlik 3'de gösterilmiştir.

$$l(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{L} \sum_{l=1}^{l=|L|} -(y_l \log(\hat{y}_l) + (1 - y_l) log(1 - \hat{y}_l)) \tag{3}$$



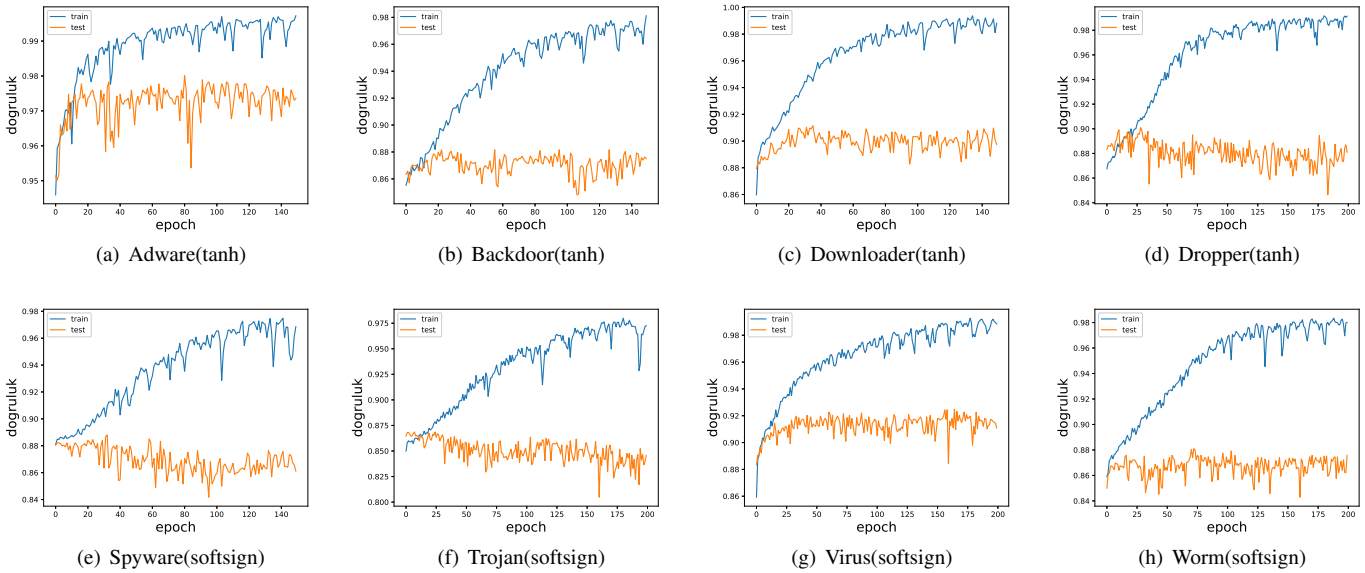
Şekil 2. Windows API çağırımları ile LSTM modeli oluşturulması

V. DENEYSEL SONUÇLAR

Çalışmamızda, herbir zararlı yazılım türü için *tanh*, *sigmoid*, *softsign* ve *softmax* fonksiyonları kullanılarak modeller oluşturulmuştur. Bu modellerden, en iyi sonuç elde ettiğimiz

TABLO I. SINIFLANDIRMA ANALIZ SONUÇLARI

	Zararlı Davranış Türleri							
	Adware	Backdoor	Downloader	Dropper	Spyware	Trojan	Virus	Worm
Doğruluk	0.975	0.875	0.896	0.88	0.86	0.845	0.913	0.876
Kesinlik	0.83	0.60	0.68	0.48	0.46	0.39	0.77	0.60
Anımsama	0.77	0.56	0.59	0.44	0.42	0.28	0.68	0.45
F1	0.80	0.58	0.63	0.46	0.44	0.33	0.72	0.51
Fonksiyon	tanh	tanh	tanh	tanh	softsign	softsign	softsign	softsign
Devir	150	150	150	200	150	200	200	200



(a) Adware(tanh) (b) Backdoor(tanh) (c) Downloader(tanh) (d) Dropper(tanh)

(e) Spyware(softsign) (f) Trojan(softsign) (g) Virus(softsign) (h) Worm(softsign)

Şekil 3. Farklı zararlı yazılım aileleri tespit modellerinin eğitim tarihçeleri

modellerin analiz ve test sonuçları Tablo I 'de gösterilmiştir. Bu modellerin eğitim ve test tarihçeleride Şekil 3 'de verilmiştir.

Tablo I incelendiğinde Adware, Backdoor, Downloader ve Dropper zararlı yazılım türlerinin eğitiminde *tanh* fonksiyonu, Spyware, Trojan, Virus ve Worm türlerinde ise *softsign* fonksiyonu kullanılarak en iyi modelin elde edildiği gözükmektedir. Herbir tür için elde edilen doğruluk değerlerinin 0.845 ile 0.975 arasında olduğu gözlemlenmektedir.

VI. SONUÇ VE GELECEK ÇALIŞMALAR

Bu çalışma kapsamında, 8 farklı türdeki gerçek zararlı yazılımların işletim sistemi üzerinde yaptıkları API çağrıları elde edilerek bir veri kümesi oluşturulmuştur. Bu veri kümesi kullanılarak, 8 farklı zararlı yazılım türü için ayrı ayrı LSTM tabanlı sınıflandırma modelleri oluşturulmuştur. Herbir model, işletim sistemi üzerindeki zararlı bir davranış türünü sınıflandırmaktadır. Çalışma kapsamında elde edilen test sonuçları, bu modellerin sınıflandırma için kullanılabileceğini göstermektedir. Metamorfik zararlı yazılımlar da, diğer aile üyelerinin yaptığı gibi, bir işletim sistemi üzerinde benzer davranışlar sergilemektedirler. Bu nedenle, yapmış olduğumuz çalışma metamorfik zararlı yazılımları sınıflandırmak için kullanılabilir. Gelecek çalışmalarda iki yönlü LSTM modelleri ve bunların Stanford Üniversitesi tarafında önerilen Word2Vec modelleri ile eğitimi konusu araştırılacaktır.

KAYNAKLAR

[1] Babak Bashari Rad, Maslin Masrom ve Suhaimi Ibrahim, “Camouflage in Malware: from Encryption to Metamorphism", IJCSNS International Journal of Computer Science and Network Security, VOL.12 No.8, 2012.

[2] Felix Leder, Bastian Steinbock, Peter Martini, “Classification and Detection of Metamorphic Malware using Value Set Analysis", 2009 4th International Conference on Malicious and Unwanted Software (MALWARE).

[3] Vishakha Mehra, Vinesh Jain, Dolly Uppal, “DaCoMM: Detection and Classification of Metamorphic Malware", 2015 Fifth International Conference on Communication Systems and Network Technologies.

[4] Duaa Ekhtoom, Mahmoud Al-Ayyoub, Mohammed Al-Saleh, Mohammad Alsmirat, Ismail Hmeidip, “A Compression-Based Technique to Classify Metamorphic Malware", 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA).

[5] SB Prapulla, Sharath J Bhat, G Shobha, “Framework for Detecting Metamorphic Malware Based on Opcode Feature Extraction", 2017 2nd International Conference on Computational Systems and Information Technology for Sustainable Solution (CSITSS).

[6] Radu S. Pirscoveanu, Steven S. Hansen, Thor M. T. Larsen, Matija Stevanovic, Jens Myrup Pedersen, Alexandre Czech, “Analysis of Malware Behavior: Type Classification using Machine Learning", 2015 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA).

[7] Ben Athiwaratkun, Jack W. Stokes, “Malware classification with LSTM and GRU language models and a character-level CNN", 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).

[8] Yong Fang, Cheng Huang, Liang Liu, Min Xue, “Research on Malicious JavaScript Detection Technology Based on LSTM", IEEE Access PP(99):1-1 · October 2018.

[9] Muhammad Ejaz Ahmed, Surya Nepal, Hyoungshick Kim, “MEDUSA: Malware Detection Using Statistical Analysis of System’s Behavior", 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC).

[10] Sepp Hochreiter, Jürgen Schmidhuber, “Long Short-term Memory", Neural Computation 9(8):1735-80, December 1997.