Zararlı Powershell Komutlarının Konvolüsyonel Sinir Ağları ile Tespit Edilmesi

Detection of Malicious Powershell Commands with Convolutional Neural Networks

Yazarlar Gizlenmiştir

***Özetçe*—Microsoft PowerShell, Windows makinelerde varsayılan olarak yüklü gelen yeni nesil bir komut satırı uygulamasıdır. Programcıların, sistem yöneticilerinin ve yazılım geliştiricilerin .NET framework'üne ve Windows işletim sisteminin sağlamış olduğu özelliklere doğrudan erişip, bu özellikleri kullanabilmeleri için bir ortam sağlar. Sistem yöneticilerinin, rutin olarak yaptıkları dosya silme/oluşturma, çeşitli servisleri başlatıp/durdurma vb. işlemleri MS PowerShell komutları yardımıyla otomatik hale getirebilmeleri gibi birçok yönüyle Windows işletim sistemi ekosisteminde önemli bir yere sahiptir. Tüm Windows makinelerde yüklü gelmesi, otomatize edilebilmesi, RAM üzerinden direkt çalıştırılabilmesi sayesinde performanslı olması ve çeşitli kod gizleme yöntemleri (obfuscation vb.) kullanılarak siber güvenlik ürünlerinin (antivirus, endpoint detection vb.) atlatılabilmesi gibi nedenlerle siber olaylar sonrası yapılan adli bilişim analizlerinde powershell komutları ile ilgili bir bilgi elde etmek zordur. Bu komutlar, çalıştırıldıktan sonra iz bırakmayacak şekilde rahatlıkla tasarlanabilmektedir. Bu çalışmada, PowerShell komutlarının Doğal Dil işleme (Natural Language Prcessing – NLP) yöntemleri kullanılarak derin öğrenme tabanlı sınıflandırılması gerçekleştirilmiştir.**

Anahtar Kelimeler — PowerShell Komutları, Doğal Dil İşleme, Derin Sinir Ağları, Kelime Vektörü.

***Abstract*—Microsoft PowerShell is a next-generation command-line application that is installed by default on Windows machines. It provides an environment for programmers, system administrators, and software developers to directly access and use the .NET framework and features provided by the Windows operating system. System administrators routinely delete / create files, start / stop various services, and so on. It has an important place in the Windows operating system ecosystem in many aspects such as automating processes with the help of MS PowerShell commands. Forensic information analysis performed after cyber events due to the fact that it is installed on all Windows machines, it can be automated, it can be run directly through RAM and it can perform cyber security products (antivirus, endpoint detection, etc.) using various code hiding methods (obfuscation, etc.) for this reason it is difficult to obtain information about. These commands can be designed to leave no trace after execution. In this study, PowerShell commands are classified as deep learning based on Natural Language Prcessing (NLP).**

Keywords — PowerShell Commands; Natural Language Processing; Deep Neural Networks, Word Embedding.

I. GİRİŞ

Siber suç çeşitli biçimlerde modern dijital toplum için ciddi bir tehdit oluşturmaktadır. Güvenlik şirketleri tarafından yapılan çeşitli raporlar, yapılandırma yönetimi ve görev otomasyonu için kuruluşlarda normalde kullanılan bir komut dosyası olan PowerShell [1-3] kullanımının siber saldırılarındaki popülerliği gözlemlemektedir. PowerShell saldırının bir çok farklı noktasında kişi tarafından veya kötü amaçlı yazılım tarafından saldırılan sistemde hüküm kazanmak için, sunucuyu kontrol etme veya keşif yapma gibi çeşitli kötü amaçlı etkinlikler gerçekleştirmek için kullanılabilir. Kötü amaçlı etkinliklerde PowerShell kullanımının farklılığı ve çeşitliliği, savunucular tarafından ele alınması gereken önemli bir noktadır.

Siber suçluların, PowerShell kullanmalarının nedenleri [4]

- Yeni windows sistemlerde varsayılan olarak yüklü gelmektedir.
- Hafızadan(memory) direkt olarak çalışabilir.
- Varsayılan olarak sistem üzerinde az iz üretir, bu yüzden adli bilişim(forensic) analizlerini zorlaştırır.
- Uzaktan varsayılan olarak şifreli trafik ile erişim imkanı bulunmaktadır.
- Betik (script) gibi çalışır, kolaylıkla karmaşık hale getirilebilir (obfuscate) ve geleneksel güvenlik araçlarıyla tespit edilmesi zordur.
- Birçok sandbox (deneme ortamı) ortamı script tabanlı zararlı yazılımları iyi analiz edemez.

Derin Öğrenme (DL) [5- 7] alanında son yıllardaki bilimsel başarılar, siber savunma için yeni yöntemlerin geliştirilmesinde birçok fırsat sunmaktadır. DL'deki büyük buluşlardan biri, çeşitli Doğal Dil İşleme (Natural Language Processing - NLP) görevlerinde bağlamsal düğünlerin (contextual embeddings)

kullanımı ile ilişkilidir. Son yıllarda kelimeleri vektörlere gömmek için çeşitli yöntemler önerilmiştir [8 - 11].

PowerShell betikleri metinsel veri içerdiğinden, Doğal Dil İşleme (NLP) topluluğu içinde geliştirilen çeşitli yöntemleri kullanarak analizlerini düşünmek doğaldır. Duygu analizi için geliştirilen NLP teknikleri [12], kötü niyetli PowerShell olmayan komut dosyalarını [13] tespit etmek için de uygulanabilinir. Bununla birlikte, NLP tekniklerini kötü niyetli komutları saptamak için uyarlamak kolay değildir, çünkü siber saldırganlar kasıtlı olarak komutları kaldırmaktan kaçınmak için komut komutlarını gizlice karıştırırlar [14].

Zhang ve diğ. [15], metin sınıflandırmada kelime seviyesinden ziyade karakter seviyesinde girişin Konvolüsyonel Sinir Ağları (Convolutional Neural Networks - CNN)'na verildiği bir yaklaşım öne sürmüştür. Derin öğrenme temelli sınıflandırıcılarını, kelime tabanlı geleneksel NLP yöntemleriyle n-gram gibi ve Recurrent Neural Networks (RNN LSTM bloklarını kullanarak) ile karşılaştırdılar. Ampirik değerlendirmeleri, duyarlılık analizi ve konu sınıflandırma veri setleri kullanılarak yapıldı. Elde ettikleri sonuçlar, geleneksel yöntemlerin küçük/orta veri kümelerinde daha iyi performans sunarken karakter tabanlı CNN'lerin daha büyük veri kümelerinde daha iyi performans gösterdiğini göstermektedir.

Hendler ve diğ. [16] iyi huylu ve kötü niyetli PowerShell komutlarını ayırt etmek için birkaç derin öğrenme modeli kullanmayı önerdiler. 6, 290 kötü amaçlı ve 60, 098 temiz PowerShell komutu veri kümesiyle, metin sınıflandırması için karakter düzeyinde Konvolüsyon Sinir Ağlarına (CNN) dayalı hem Doğal Dil İşleme (NLP) tabanlı dedektörleri hem de dedektörleri uyguladılar ve metni ham olarak ele aldılar. Farklı mimariler üzerindeki sonuçlarına göre (9 katmanlı CNN, 4 katmanlı CNN ve uzun kısa süreli bellek ağı dahil), tüm dedektörler 0.985 ve 0.990 arasında yüksek AUC seviyeleri elde etti. Yazarlar en iyi performans gösteren sınıflandırıcının geleneksel NLP tekniklerini CNN tabanlı bir sınıflandırıcı ile birleştiren bir topluluk sınıflandırıcısı olduğunu ileri sürmektedir. Bununla birlikte, daha yüksek yanlış pozitif oranlarda, dışarıda bırakılan test setlerinde daha kötü performans gözlemlenmiştir. Yeni bir blogda, FireEye [17], PowerShell sözdizimi için ağaç (prefix-tree) tabanlı bir saplayıcıdan yararlanan kötü amaçlı PowerShell komutlarını algılamak için denetimli bir sınıflandırıcı uygular. Makine öğrenme modeline girdi, saplanmış tokenlerin vektörize edilmiş bir gösterimidir. Yukarıdaki öneriler, daha zor bir meydan okuma olan komut dosyaları yerine kötü amaçlı PowerShell komutlarını tespit etmeye odaklanmıştır. Dahası, özellikler komutun işlevsel semantiğini yakalayamayabilecek ve karakter frekans kurcalamaya eğilimli olan komutların metin biçiminden türetilmiştir.

Bohannon ve Holmes [18] gizlenmiş PowerShell senaryolarını incelediler. Bir temel karakter frekans analizi sundular ve PowerShell betiklerinde gizlemeyi (obfuscated) tespit etmek için Kosinüs benzerliğini kullandılar. Umut vaat eden ön sonuçları tanımlar ve gizlenmiş ve gizlenmemiş kodlar arasında önemli bir fark olduğunu belirtmektedirler. Çalışmalarında Daniel Bohannon 2016'da oluşturduğu "Invoke-Obfuscation" modülünü herkese açık olarak yayınlamaktadırlar .

JavaScript [19,20] gibi kötü amaçlı komut dosyalarını tespit etmek için bazı çalışmalar yapılmış olsada, PowerShell, siber savaştaki önemli statüsüne rağmen, akademik topluluk tarafından göreceli olarak ilgilenmemiştir. PowerShell'deki çalışmaların çoğu, Symantec [4] ve Palo Alto Networks [21] gibi şirketlerdeki güvenlik uzmanları tarafından yapılır. Bu yayınlar, kötü niyetli PowerShell etkinliklerini saptamak için yaklaşımlar geliştirmek ve değerlendirmek yerine, esas olarak PowerShell tehdidini araştırmaya odaklanır. Kötü niyetli PowerShell komutlarının otomatik olarak algılanması konusundaki araştırma eksikliği ile PowerShell tabanlı kötü amaçlı siber faaliyetlerin yaygınlığı arasındaki fark, bu tür saldırıları tespit etmek için etkili yöntemler geliştirme acil ihtiyacını vurgulamaktadır.

Çalışmanın geri kalanı şu şekilde organize edilmiştir. Bölüm II'de kullanılan veri seti, komutların nasıl ön işleme alındığı ve eğitim setinin nasıl oluşturulduğu açıklanmıştır. Temiz ve kötü amaçlı PowerShell komutlarının NLP yöntemleri kullanılarak derin öğrenme tabanlı dedektörler ile sınıflandırma yöntemi Bölüm III'de verilmiştir. Bölüm IV'de performans değerlendirilmesi yapılmıştır. Bölüm V'de sonuç ve gelecek çalışmalara yer verilmiştir.

II. VERİ SETİ

Çalışmamızda PowerShell derlemi (corpus) için Daniel Bohannon [18] 2016'da oluşturdu ve yarattığı "Invoke-Obfuscation" modülü ve yayınladıkları veri seti kullanılmıştır. Toplamda 365,083 etiketsiz betik (script) dosyası bulunmaktadır. Etiketli veriler elde etmek için toplamda 2,000 komut el ile gözden geçirilerek 1.000 komut kötü amaçlı 1.000 komut temiz olarak etiketlendirilmiştir. Örneğin; şifrelenmiş komut (obfuscated), dosya indirme, web adresine bağlanma gibi şüpheli komut içerek dosyaları zararlı olarak adlandırıldı. Tablo I'de örnek zaralı PowerShell komutları verilmiştir.

TABLO I. ZARARLI KOMUTLARIN KULLANIM ŞEKİLLERİ VE ÖRNEKLERİ

Komut Açıklama	*Örnek Komutlar*
Şifrelenmiş komutların kullanılması	"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -Ex BYPasS -noP -WHIddEN -eCIAVwBtAHAAcgBpAHYAZQBsAGUAZwBlAC4AZQBgAZQAdI=="
İnternet üzerinden dosya indirilmesi	-Invoke-Expression(("New-ObjectNet.WebClient")) .('Downloadfile')
ASCII formatındaki zararlı komutların çalıştırılması	ASCII formatındaki zararlı komutların çalıştırılması
Registry üzerinde kritik değerlerin okunması, eklenmesi, silinmesi	Get-Item -path HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
Komut çalıştırıldığında portu dinlemeye başlaması	New-Object System.Net.HttpListener

Şekil. 1.Akış Diyagramı

A. *Veri Ön İşleme ve Eğitim Seti Oluşturma*

Sınıflandırma modelinde tespit etme (detection) ve değerlendirme (evaluation) sonuçlarının iyileştirilmesi için PowerSehll komutları normalizasyon işlemlerinden geçirildi. İlk olarak IP adresleri, rastgele alan adları (çoğu durumda rakam içermektedir), tarihler, sürüm numaraları vb. verilerle başa çıkmak için rakamlar yıldız (‘*’) işareti ile değiştirildi.

Tablo II’de verilen örnek PowerShell komutunda da görüleceği üzere yorum satırları bulunmaktadır. Ön işlemenin ikinci adımında bu satırlar kaldırılmıştır. Son adımda tüm karakterler küçük harflere dönüştürülmüştür.

TABLO II. ÖRNEK KOMUT

Örnek Temiz PowerShell Komutu
Check if destination file already exists If (!(Test-Path " $SharePoint2013Path\$FileName")) { *## Begin download* Start-BitsTransfer -Source $DownLoadUrl –Destination $SharePoint2013Path\$fileName –DisplayName "Downloading '$FileName`' to $SharePoint2013Path" -Priority High Description "From $DownLoadUrl..." -ErrorVariable err }

III. SINIFLANDIRMA MODELLERİ

Bu bölümde, Şekil 1’de verilen akış diyagramı takip edilerek kötü amaçlı PowerShell komut tespiti için geliştirilen makine öğrenme modellerini açıklanmıştır. Model performansları Bölüm IV'te verilmiştir.

A. *Özellik Çıkarımı Kelime Vektörlerinin Oluşturulması*

Makine Öğrenmesi algoritmaları, dizileri ve dökümanları ham formda işlemede problem yaşamaktadır. Bu algoritmalar, girdi olarak sayı gerektiriler. Bu aşamada PowerShell komutları metin olarak ele alınmıştır. NLP’de Kelime vektörleri (word embeddings) yöntemi genel olarak kelimeleri bir vektörle eşleştirmeye çalışır. Kelime vektörleri, sayılara dönüştürülen metinlerdir. Metinde en sık geçen kelimelere en düşük indeks numarası verilir. Örneğin, Tablo III'de "system" kelimesinin indeks numarası 5 olarak tespit edilmiştir.

TABLO III. ÖRNEK TEMİZ POWERSHELL KOMUTU VE KELİME VEKTÖRÜ

system reflection assembly load with partialname
Microsoft [5, 207, 175, 515, 48,635]

B. *Derin Sinir Ağları*

Çalışmada, kötü ve temiz olarak komutları sınıflandırmak için klasik Derin Sinir Ağları (Deep Neural Network) ile Konvolüsyonel Sinir Ağları (Convolutional Neural Network – ConNN*) uygulanmıştır.

Derin Sinir Ağları modelinde, Tablo IV’de verildiği üzere 4 Katman bulunmaktadır bunlar; Gömme (Embedding), En-Büyük Örneklem (Max-Pooling) ve Yoğun (Dense) katmanlarıdır. Gömme (Embedding) Katmanının iki hizmeti bulunmaktadır. İlk olarak, girdinin boyutsallığını azaltırlar. İkincisi, girdiyi bağlamını koruyacak şekilde temsil ederler. Gömme (Embedding) katmanı, her bir giriş belirtecini (eldeki soruna bağlı olarak genellikle bir sözcük veya karakter) bir vektör temsilcisine dönüştürür. En-Büyük Örneklem (Pooling) katmanı modelin aşırı öğrenmesini engeller.

Derin Sinir Ağları modelinde, Tablo IV’de verildiği üzere 4 Katman bulunmaktadır bunlar; Gömme (Embedding), En-Büyük Örneklem (Max-Pooling) ve Yoğun (Dense) katmanlarıdır. Gömme (Embedding) Katmanının iki hizmeti bulunmaktadır. İlk olarak, girdinin boyutsallığını azaltırlar. İkincisi, girdiyi bağlamını koruyacak şekilde temsil ederler. Gömme (Embedding) katmanı, her bir giriş belirtecini (eldeki soruna bağlı olarak genellikle bir sözcük veya karakter) bir vektör temsilcisine dönüştürür. En-Büyük Örneklem (Pooling) katmanı modelin aşırı öğrenmesini engeller.

TABLO IV. DERİN SİNİR AĞI MİMARİSİ

Katman (Tip)	**Kalıp (Shape)**
embedding_4 (Embedding)	(37839, 50)
global_max_pooling1d_1	(50)
dense	(10)
dense	(1)

Bir diğer uygulanılan derin öğrenme algoritması da ConNN algoritmasıdır. ConNN modelinde, Tablo IV’de verildiği üzere 5 katman bulunmaktadır. Derin Sinir Ağları modelinden farklı olarak konvolüsyonel katmanı bulunur. Bu konvolüsyonel katmanlar görsel veriler üzerinde kenarları, köşeleri ve diğer doku türlerini algılayabilir ve bu da onları özel bir araç yapar.

* Convolutional Neural Network Algoritmasının kısaltması Savaş ve ark.[22]’nın önerdiği ConNN şekilde kullanılacaktır. CNN kısaltması Celluar Neural Network için kullanılmaktadır.

Metin gibi sıralı verilerle çalışırken ise tek boyutlu konvolüsyonel katman kullanılır.

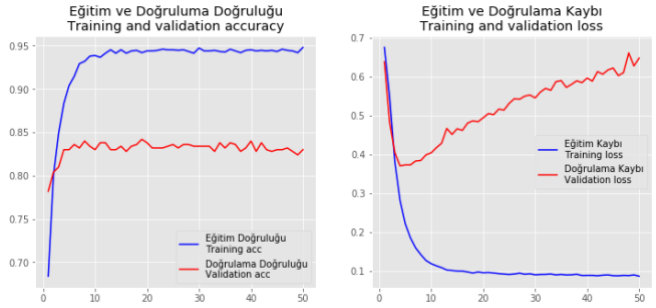
TABLO V. DERİN SİNİR AĞI MİMARİSİ

Katman (Tip)	Kalıp (Shape)
embedding_4 (Embedding)	(37839, 100)
conv1d_1 (Conv1D)	(37835, 128)
global_max_pooling1d_1	(128)
dense	(10)
dense	(1)

IV. DENEYSEL SONUÇLAR VE DEĞERLENDİRME

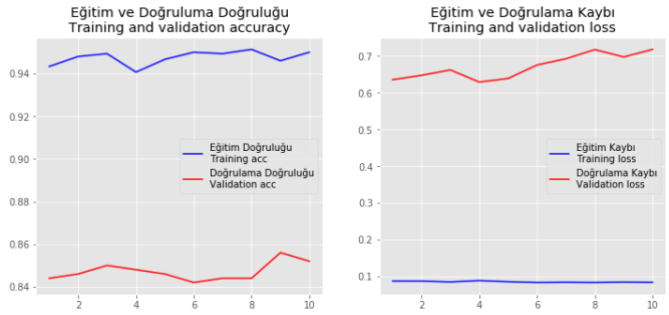
Bu bölümde, sınıflandırma algoritmalarının nasıl sonuç verdiğini değerlendiriyoruz. Algoritmalar, eğitim ve test verileri üzerinde verilen doğruluk (acurracy) ve kayıp (loss) metrikleri ile değerlendirildi. Etiketli veri seti %75-%25 oranında eğitim-test verisi olarak bölündü.

Derin Sinir Ağı algoritması, Eğitimde 0.9480, Testte 0.83 doğruluk ile sonuçlanıştır. Şekil 2'de algoritmaya ait doğruluk ve kayıp grafikleri verilmiştir.



Şekil. 2.Model Doğruluğu ve Kaybı

ConNN, Eğitimde 0.9507, Testte 0.8520 doğruluk ile sonuçlanıştır. Şekil 3'de doğruluk ve kayıp grafikleri verilmiştir.



Şekil. 3.Model Doğruluğu ve Kaybı

V. SONUÇLAR VE GELECEK ÇALIŞMALAR

Bu çalışmada, Güvenlik Hizmetleri, Sağlık Hizmetleri, Eğitim Hizmetleri gibi kuruluşları hedef alan kötü amaçlı PowerShell komutlarının tespiti incelenmiştir. Komutların tespitine yönelik iki farklı derin öğrenme tabanlı sınıflandırıcı incelenerek değerlendirildi. Değerlendirme sonuçlarımız, sınıflandırıcıların yüksek performans verdiğini göstermektedir. En iyi performans ConNN sınıflandırıcısı ile elde edilmiştir. Gelecek çalışmalarda, veri setinin arttırılması ve Recurrent Neural Networks (RNN) sınıflandırıcısı üzerinde çalışılması planlanmaktadır.

KAYNAKLAR

[1] PaloAlto, "Pulling Back the Curtains on EncodedCommand PowerShell Attacks," 2017.

[2] Symantec, "The increased use of Powershell in attacks," 2016.

[3] FireEye, "Malicious PowerShell Detection via Machine Learning," 2018.

[4] Symantec. The increased use of Powershell in attacks. https://www.symantec.com/content/dam/symantec/docs/security-center/white-papers/increased-use-of-powershell-in-attacks-16-en.pdf , 2016.

[5] I. J. Goodfellow, Y. Bengio, and A. C. Courville, Deep Learning, ser. Adaptive computation and machine learning. MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org/

[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[7] J. Schmidhuber, "Deep learning in neural networks: An overview," Neural networks, vol. 61, pp. 85–117, 2015

[8] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in Advances in neural information processing systems. NIPS, 2013, pp. 3111–3119

[9] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.

[10] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," Transactions of the Association for Computational Linguistics, vol. 5, pp. 135–146, 2017.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805, 2018.

[12] Bing Liu and Lei Zhang. A survey of opinion mining and sentiment analysis. In Mining text data, pages 415–463. Springer, 2012.

[13] Yao Wang, Wan-dong Cai, and Peng-cheng Wei. A deep learning approach for detecting malicious javascript code. Security and Communication Networks, 9(11):1520–1534, 2016.

[14] Symantec. The increased use of Powershell in attacks. https://www.symantec.com/content/dam/symantec/docs/security-center/white-papers/increased-use-of-powershell-in-attacks-16-en.pdf , 2016.

[15] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In Advances in neural information processing systems, pages 649–657. NIPS, 2015.

[16] Danny Hendler et al. 2018. Detecting Malicious PowerShell Commands using Deep Neural Networks. In Proceedings of the 2018 on Asia Conference on Computer and Communications Security. ACM, 187–197.

[17] Victor Fang. 2018. Malicious PowerShell Detection via Machine Learning. https://www.fireeye.com/blog/threat-research/2018/07/malicious-powershell- detection-via-machine-learning.html. (2018).

[18] Daniel Bohannon and Lee Holmes. 2017. Revoke-Obfuscation: PowerShell Obfuscation Detection Using Science. https://www.fireeye.com/blog/threat- research/2017/07/revoke-obfuscation-powershell.html. (2017).

[19] Joshua Saxe and Konstantin Berlin. expose: A character-level convolutional neural network with embeddings for detecting malicious urls, file paths and registry keys. arXiv preprint arXiv:1702.08568, 2017.

[20] Yao Wang, Wan-dong Cai, and Peng-cheng Wei. A deep learning approach for detecting malicious javascript code. Security and Communication Networks, 9(11):1520–1534, 2016.

[21] PaloAlto. Pulling Back the Curtains on EncodedCommand PowerShell Attacks. https://researchcenter. paloaltonetworks.com/2017/03/unit42-

pulling-back-the-curtains-on-encodedcommand-powershell-attacks/, 2017.

[22] B. K. Savaş and Y. Becerikli, "Real Time Driver Fatigue Detection System Based on Multi-Task ConNN," in IEEE Access. doi: 10.1109/ACCESS.2020.2963960.

* Convolutional Neural Network Algoritmasının kısaltması Savaş ve ark.[22]'nın önerdiği ConNN şekilde kullanılacaktır. CNN kısaltması Celluar Neural Network için kullanılmaktadır.