TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DERİN SİNİR AĞ TABANLI DOSYA VE VERİ PARÇASI SINIFLANDIRILMASI

YÜKSEK LİSANS TEZİ

Ayşe Sıddıka EROZAN

Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı: Doç. Dr. Hüsrev Taha SENCAR

TEMMUZ 2018

Fen Bilimleri Enstitüsü Onayı

……..
Prof. Dr. Osman EROĞUL
Müdür

Bu tezin Yüksek Lisans derecesinin tüm gereksinimlerini sağladığını onaylarım.

……
Prof. Dr. Oğuz ERGİN
Anabilimdalı Başkanı

TOBB ETÜ, Fen Bilimleri Enstitüsü'nün 141111047 numaralı Yüksek Lisans Öğrencisi **Ayşe Sıddıka EROZAN**'nın ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı **"DERİN SİNİR AĞ TABANLI DOSYA VE VERİ PARÇASI SINIFLANDIRILMASI"** başlıklı tezi **04.07.2018** tarihinde aşağıda imzaları olan jüri tarafından kabul edilmiştir.

Tez Danışmanı : **Doç. Dr. Hüsrev Taha SENCAR** .....
TOBB Ekonomi ve Teknoloji Üniversitesi

Jüri Üyeleri : **Dr. Öğr. Üyesi A. Murat ÖZBAYOĞLU (Başkan)** .....
TOBB Ekonomi ve Teknoloji Üniversitesi

Doç. Dr. Sevil ŞEN .....
Hacettepe Üniversitesi

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, alıntı yapılan kaynaklara eksiksiz atıf yapıldığını, referansların tam olarak belirtildiğini ve ayrıca bu tezin TOBB ETÜ Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırlandığını bildiririm.

Ayşe Sıddıka EROZAN

ÖZET

Yüksek Lisans Tezi

DERİN SİNİR AĞ TABANLI DOSYA VE VERİ PARÇASI SINIFLANDIRILMASI

Ayşe Sıddıka EROZAN

TOBB Ekonomi ve Teknoloji Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Hüsrev Taha SENCAR

Tarih: Temmuz 2018

Bu çalışmada sunulan araştırma, adli bilişim ve bilgi güvenliği uygulamalarında hayati önem taşıyan dosya ve veri türü sınıflandırmasına yönelik bir çözüm önermektedir. Son on beş yılda dosya ve veri türü sınıflandırması araştırmalarında kullanılan yöntemler, dosya uzantısı tabanlı yöntemler, sihirli bayt tabanlı yöntemler ve içerik tabanlı yöntemlerdir. Bu yöntemlerden uzantı tabanlı ve sihirli bayt tabanlı yöntemler, dosya başlığında yer alan sihirli baytlar ve dosya uzantıları kolayca değiştirilebildiğinden dolayı yetersiz yöntemlerdir. İçerik tabanlı yöntemler sihirli bayt ve dosya uzantıları gibi değişikliklere karşı dirençli olduğundan son yıllarda bu alanda yapılan çalışmalar hızlı bir şekilde artmıştır. İçerik tabanlı yöntemlerin kullanıldığı çalışmaların çoğunda çok az sayıda dosya ve veri türü kullanılmaktadır. Bu alanda yapılan çok az sayıda çalışmada ise çok sayıda dosya ve veri türü kullanılmaktadır. Ancak bu çalışmalardaki dosyaların bazıları işletim sistemlerinde çok az kullanılan dosya türleridir. Bu çalışmada en çok kullanılan 15 dosya ve veri türünü içeren içerik tabanlı dosya ve veri parçası sınıflandırma yöntemi sunulmuştur. Sınıflandırma alanında son yıllarda derin sinir ağları yaygın bir şekilde kullanılmaya başlanmıştır. Kullanılan sınıflar eğitim setinde yeterince iyi genellediğinde çok iyi sınıflandırma performansı elde edilmektedir. Bu çalışmada da dosya ve veri sınıflandırması

problemine derin sinir ağ mimarileri kullanılarak çözüm aranmaktadır. Önerilen yöntemde iki seviyeli hiyerarşik model kullanılmakta olup bu hiyerarşik sınıflandırma sisteminde ilk seviyede birkaç alternatif sınıflandırma modeline dayanan deneyler yapılmıştır. Alternatif sınıflandırma modelleri entropi bazlı dört farklı durum ve sınıflandırma bazlı üç farklı algoritma kullanılmaktadır. İkinci seviyede ise kazanan model üzerinden derin sinir ağları kullanılmıştır. İşletim sistemlerinde kullanılan en küçük küme birim büyüklüğü olan 4 kilobayt ve 8 kilobaytlık dosya ve veri parçaları kullanılarak 2-gram analizi ile öznitelikler çıkartılmaktadır. Çıkarılan bu öznitelikler üç farklı makine öğrenmesi algoritması kullanılarak entropiye dayalı olarak gruplara ayrılmaktadır. Daha sonra bu ayrılan gruplar üzerinden dosya ve veriler derin sinir ağlar kullanılarak tür tabanlı sınıflandırma yapılmakladır. 4 kilobayt ve 8 kilobayt için sınıflandırma doğruluk oranları sırasıyla %92,80 ve %94,67'dir. Yapılan bu çalışmada doğruluk oranını önemli ölçüde azaltan şifrelenmiş veri türü olan aes256 kullanılmasına rağmen benzer dosya türü kullanılarak yapılan en iyi çözüm ile karşılaştırıldığında bizim önerdiğimiz yöntem doğruluk oranını %6,87 oranında artırdığı görülmektedir.

Anahtar Kelimeler: Dosya ve veri parçası, İçerik tabanlı yöntemler, Derin sinir ağlar, 2-gram, Adli bilişim.

ABSTRACT

Master of Science

A DEEP NEURAL NETWORK BASED FILE AND DATA FRAGMENT CLASSIFICATION

Ayşe Sıddıka EROZAN

TOBB University of Economics and Technology
Institute of Natural and Applied Sciences
Computer Engineering Science Programme

Supervisor: Assoc. Prof. Dr. Hüsrev Taha SENCAR

Date: July 2018

The research presented in this paper provides a solution for file and data type classification which is crucial digital forensics and information security applications. Over the past fifteen years, the existing methods for file and data type classification are file extension based methods, magic byte based methods and content based methods for file and data type classification. Extension based and magic byte based methods are impotent methods since file extension and magic bytes which is in the file header can be easily changed. Since content-based methods are resistant to changes in magic bytes and file extensions, content-based methods have been frequently investigated in the recent years. Majority of existing studies, where content based methods are used, classify very few file and data types. Only few works classify large number of file and data types. However, these works do not cover the most used file and data types in the well-known operating systems. In this paper, a content based file and data fragment classification method which covers the most used 15 files and data type is presented. In the classification applications, deep neural networks has been widely used in recent years, and great classification results is obtained when the used classes are sufficiently good in the training set. Therefore the proposed method uses deep neural networks for file and data type classification. The proposed method

classifies 15 file and data types by using two level hierarchical model. In this hierarchical classification system, empirical test based on several alternative classification models are performed in the first level. It is used three classification algorithm and entropy based four different cases. In the second level hierarchy, deep neural networks are used on the winning model. 2-gram features are extracted using 4 kilobytes and 8 kilobytes of files and data fragments, which are the smallest cluster sizes used in operating systems. These extracted features are divided into classes based on entropy using three different machine learning algorithms. In the second level, these specified classes are classified to 15 classes by using deep neural networks. The results show that the classification accuracies for 4 kilobytes and 8 kilobytes are 92.80% and 94.67% respectively. Therefore, the proposed method improves the accuracy by 6.87% than the relevant state of the art while it also includes encrypted data type (aes256) which dramatically decreases the classification accuracy since the encryption changes the file content randomly.

Keywords: File and data fragment, Content-based, Deep neural network, 2-gram, Digital forensics.

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren hocam Doç. Dr. Hüsrev Taha SENCAR, kıymetli tecrübelerinden faydalandığım TOBB Ekonomi ve Teknoloji Üniversitesi Bilgisayar Mühendisliği Bölümü öğretim üyelerine teşekkür ederim. Ayrıca eğitim hayatım boyunca her zaman sonsuz destekleri ile yanımda olan aileme çok teşekkür ederim. Son olarak, eşim Ahmet'e çalışmalarım boyunca bana gösterdiği anlayış ve verdiği sonsuz destekten dolayı çok teşekkür ederim.

İÇİNDEKİLER

Sayfa

ÖZET ..... **iv**
ABSTRACT ..... **vi**
TEŞEKKÜR ..... **viii**
İÇİNDEKİLER ..... **ix**
ŞEKİL LİSTESİ ..... **x**
ÇİZELGE LİSTESİ ..... **xi**
KISALTMALAR ..... **xii**
SEMBOL LİSTESİ ..... **xiii**
RESİM LİSTESİ ..... **xiv**
1. GİRİŞ **1**
2. DOSYA TÜRÜ TESPİTİ İÇİN KULLANILAN MEVCUT YÖNTEMLER **7**
2.1 Uzantı Tabanlı Yöntemler 7
2.2 Sihirli Bayt Tabanlı Yöntemler 8
2.3 İçerik Tabanlı Yöntemler 14
2.3.1 Dosya Türü Sınıflandırılması 15
2.3.2 Dosya Parçası Sınıflandırılması 15
3. ARKA PLAN BİLGİSİ **19**
3.1 Google Hacking 19
3.2 JSOUP kütüphanesi 22
3.3 N-Gram Analizi 24
3.4 Rastgele Orman Algoritması 25
3.5 Destek Vektör Makineleri 26
3.6 Derin Sinir Ağlar 28
4. ÖNERİLEN YÖNTEM **31**
4.1 Dosya ve Veri Türü 32
4.2 Veri Toplama ve Hazırlama 34
4.3 Öznitelik Çıkarma 34
4.4 Sınıflandırma 35
5. DENEY SONUÇLARI **37**
6. SONUÇ VE ÖNERİLER **45**
KAYNAKLAR **47**
ÖZGEÇMİŞ **51**

ŞEKİL LİSTESİ

Sayfa

Şekil 1. 1: Dosya türü sınıflandırmada kullanılan yöntemler. 3
Şekil 1. 2: Sürücü içerisindeki bir plaka ve bu plakaya kaydedilmiş dosyalar. 4
Şekil 2. 1: Basit bir dosya yapısı. 8
Şekil 3. 1: 2-gram analizi örneği. 24
Şekil 3. 2: Rastgele orman algoritması karar mekanizması örneği. 26
Şekil 3. 3: Destek vektör makineleri. 28
Şekil 3. 4: Giriş katmanı, çıkış katmanı ve iki gizli katmandan oluşan genel bir derin ağ mimarisi. 28
Şekil 3. 5: Derin sinir ağlarında nöron olarak bilinen işlem birimi. 29
Şekil 4. 1: Önerilen yöntemin akış şeması. 31
Şekil 4. 2: Kullanılan dosya ve veri türleriyle bu veri türlerinin dosya türleri ile ilişkisi..... 32
Şekil 4. 3: Sınıflandırma sisteminin mimarisi. 35
Şekil 5. 1: İlk hiyerarşide kazanan model belirlendikten sonra oluşan akış şeması. . 39

ÇİZELGE LİSTESİ

Sayfa

Çizelge 2. 1: Bazı dosya türlerinin imzaları. 9
Çizelge 3. 1: Google hacking için kullanılan özel kelimeler ve anlamları. 22
Çizelge 3. 2: Jsoup kütüphanesi kullanılarak yazılmış örnek bir java kodu. 22
Çizelge 4. 1: Conti yaklaşımına göre gruplandırılmış dosya türleri. 33
Çizelge 4. 2: Entropi bazlı durumlar. 36
Çizelge 5. 1: 4 KB dosya ve veri parçalarının entropi bazlı durumlar için rastgele orman algoritması ve destek vektör makinesi deneysel test sonuçları. ... 38
Çizelge 5. 2: Orta entropi grubu içerisine giren dosya ve veri türleri için derin sinir ağlarının optimum parametreleri. 39
Çizelge 5. 3: Yüksek ve düşük entropi grubu içerisine giren dosya ve veri türleri için derin sinir ağlarının optimum parametreleri. 40
Çizelge 5. 4: Sonuçlar – 4 KB dosya ve veri parçaları kullanılarak elde edilen tür tabanlı sınıflandırma karışıklık matrisi. 42
Çizelge 5. 5: Sonuçlar – 8 KB dosya ve veri parçaları kullanılarak elde edilen tür tabanlı sınıflandırma karışıklık matrisi. 43

KISALTMALAR

BFD **:** Bayt Frekans Dağılımı (Byte Frequency Distribution)
DE **:** Düşük Entropi (Low Entropy)
DSA **:** Derin Sinir Ağları (Deep Neural Network)
DVM **:** Destek Vektör Makineleri (Support Vector Machine)
ELU **:** Üstel Lineer Birim (Exponential Linear Unit)
KA **:** Karar Ağaçları (Decision Tree)
KB **:** Kilobayt (Kilobyte)
LDA **:** Lineer Diskriminant Analizi (Linear Discriminant Analyses)
OE **:** Orta Entropi (Medium Entropy)
RELU **:** Doğrultulmuş Lineer Birim (Rectified Linear Unit)
RO **:** Rastgele Orman (Random Forest)
SA **:** Sinir Ağları (Neural Network)
SRAT **:** Sınıflandırıcı ve Regresyon Ağacı Tekniği (Classification and Regression Tree)
YE **:** Yüksek Entropi (High Entropy)
GİB : Grafik İşlemci Birimi (Graphics Processing Unit)

SEMBOL LİSTESİ

Bu çalışmada kullanılmış olan simgeler açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklama**
b	Bayas
C	Ceza parametresi
f	Aktivasyon fonksiyonu
M	Geliştirilecek ağaç sayısı
w	Bağlantı ağırlığı
x	Girdi vektörü
y	Etiket değeri
S_N	Veri seti
R^S	S elemanlı reel sayılar kümesi
μ	Her düğümde kullanılan değişken sayısı
K	Çekirdek fonksiyonu
Z	Hata parametresi

RESİM LİSTESİ

Sayfa

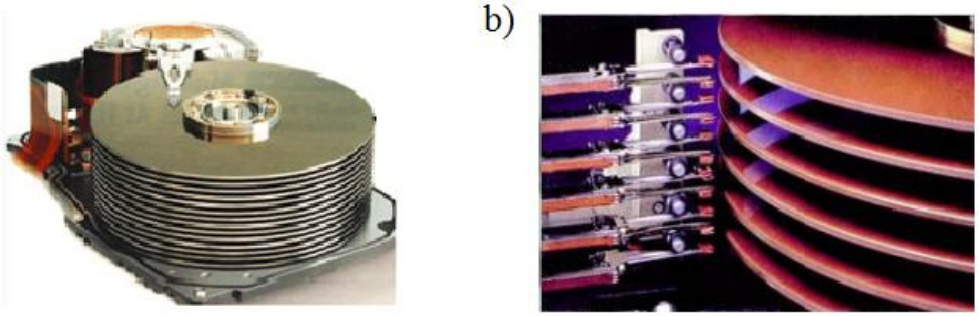
Resim 1. 1: Basit bir sürücünün iç yapısı, b) sürücü içerisinde yer alan plakalar ve okuma yazma kafaları. 1
Resim 2. 1: Türü değiştirilecek dosya ve dosyanın türünün tespit edilmesi. 7
Resim 2. 2: Türü değiştirilmiş dosya. 8
Resim 2. 3: MP3 dosyasının ikili kodları. 9
Resim 2. 4: MP3 dosyasının sihirli baytları değiştirildikten önceki ve değiştirildikten sonraki ikili kodları. 11
Resim 2. 5: MP3 uzantılı bir dosyanın TrID çevrimiçi aracı ile test sonuçları. 12
Resim 2. 6: Uzantısı değiştirilmiş bir dosyanın TrID çevrimiçi aracı ile test sonuçları 12
Resim 2. 7: Sihirli bayt bilgileri değiştirilmiş dosyanın TrID çevrimiçi aracı ile test sonuçları. 13
Resim 2. 8: Dosya parçası için TrID çevrimiçi aracının test sonuçları. 14
Resim 2. 9: İçerik tabanlı dosya türü sınıflandırması alanında yapılan çalışmaların ortalama tahmin doğrulukları. 17
Resim 3. 1: site:etu.edu.tr arama sonuçları. 20
Resim 3. 2: filtype:pdf arama sonuçları. 21
Resim 3. 3: intitle: bilgisayar mühendisliği arama sonuçları. 21
Resim 3. 4: Bir site ve bu siteye ait kaynak kodlar. 23
Resim 5. 1: İçerik tabanlı dosya türü sınıflandırması alanında yapılan çalışmaların ortalama tahmin doğrulukları ve bu çalışma ile elde edilen tahmin doğrulukları. 40

1. GİRİŞ

Modern dünyanın iş hayatımızda ve günlük hayatımızda teknolojinin getirdiği yenilikler ile birlikte dijital cihazların kullanımı hızlı bir şekilde artmıştır. Bu yeniliklerle birlikte dijital verilerde hızlı bir şekilde artmış ve veri kurtarma, saldırı tespit sistemleri, adli bilişim ve bilgi güvenliğinde; dosyaların gerçek türlerinin belirlenmesi çok önemli bir problem haline gelmiştir [1].

Dijital veriler, sabit disk sürücüleri gibi fiziksel ortamlarda depolanmaktadır. Sürücüler üst üste dizilmiş birçok plakadan oluşmaktadır. Resim 1.1 a'da basit bir sürücünün iç yapısı yer almaktadır [2]. Dijital veriler bu plakalara kaydedilmektedir. Sürücülerde elektromanyetik yazma yani bir başka deyişle dijital verilerin kaydedilmesi için okuma yazma kafaları yer almaktadır. Resim 1.1 b'de de görüldüğü üzere her bir plaka için altında ve üstünde olmak üzere iki tane okuma yazma kafaları yer almaktadır [3]. Okuma yazma kafalarının orta yüzeye, kenara ve dışa doğru hareket edebilme kabiliyeti sayesinde plakalar döndüğünde plakaların tüm yüzeylerine erişebilmektedir. Sürücülerde verileri kaydetmek için kullanılan en küçük kayıt birimleri sektör olarak adlandırılmaktadır. En küçük sektör büyüklüğü 256 bayttır. İşletim sistemlerinde ise verileri kaydetmek için daha önceden belirlenmiş olan küme (cluster) büyüklüğü kullanmaktadır. Bu küme büyüklükleri sektör büyüklerinin 2 veya 2'nin katı olmak durumundadır.

a) b)



Resim 1. 1: Basit bir sürücünün iç yapısı, b) sürücü içerisinde yer alan plakalar ve okuma yazma kafaları.

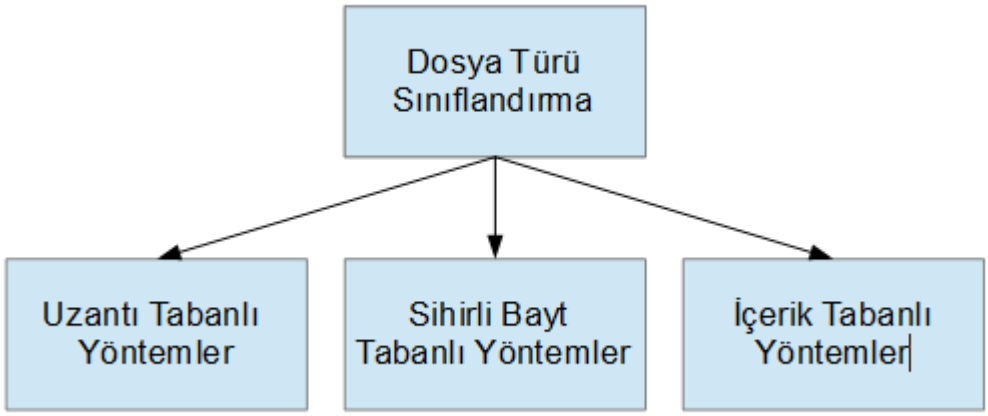
Dijital verilerin kaydedildiği bu sürücülerde yaşlanma veya fiziksel dış etmenlerden dolayı veriler tahrip olabilmekte ve verilerde kayıplar yaşanabilmektedir. Bu gibi etmenlerden dolayı verinin sihirli baytlarının bulunduğu dosya başlığı, dosya tablolarında veya dosya uzantılarında kayıplar meydana gelebilmektedir. Modern işletim sistemlerinin dosyaları çalıştırabilmesi için verinin sihirli baytlarının bulunduğu dosya başlığı ve dosya uzantılarının olması gerekmektedir. Dosya uzantısı ve dosya başlık bilgisi olmadığı veya kaybolduğu durumlarda işletim sistemleri dosyaları tanıyamamakta ve çalıştıramamaktadır. Ham veriler depolandığı yerde erişilebilir durumda ise dosya kurtarma sistemleri bu verilere erişilebilmektedir [4].

Saldırı tespit sistemleri ve güvenlik duvarları ağ üzerinden gelen dosyaların türünü belirleyebilmek için dosyanın uzantısını kontrol etmektedir. Virüs tarama sistemleri ise sadece çalıştırılabilir dosyalar üzerinde zararlı kod içeriğini arayabilmektedir. Zararlı kod içeriğine sahip çalıştırılabilir dosyanın dosya uzantısı çalıştırılamaz bir dosya uzantısına dönüştürüldüğünde, virüs tarama sistemleri bu zararlı içeriği tespit edememektedir [5, 6].

Dijital adli tıp alanında en önemli problemlerden biri dijital delillerin toplanması ve bu delillerin analiz edilmesidir. Dosyalar silinebilmekte, dosya uzantıları değiştirilebilmekte veya dosyaların bulunduğu sürücüler formatlanabilmektedir. Bu gibi durumların meydana gelmesi dijital delillerin kaybolmasına sebep olabilmektedir. Sürücülerdeki dosyalar silindiğinde dosyaları tanımlayan ve sürücülerde nerede bulunduğu bilgisi tutulan dosya sistemi kayıtları silinmektedir [7]. Adli delillerin toplanabilmesi için silinen dosyaların kurtarılması gerekmektedir. Silinen dosyanın ham verileri üzerine herhangi bir yeni dosya yazılmamış ise dosya bulunduğu yerde kalmaya devam etmektedir ve dosya kurtarma sistemleri tarafından kurtarılabilmektedir [1].

Saldırı tespit sistemleri, güvenlik duvarları, virüs tarama sistemleri ve dosya kurtarma sistemlerinin temelini dosya türlerinin belirlenmesi oluşturmaktadır. Dosya türlerini belirlenmesi yani dosya türlerinin sınıflandırılması için kullanılan yöntemler üç sınıfa ayrılmaktadır. Şekil 1.1'de dosya türü sınıflandırma için kullanılan yöntemler yer almaktadır. Bu yöntemler uzantı tabanlı, sihirli bayt tabanlı ve içerik tabanlı yöntemlerdir. Uzantı tabanlı yöntemlerde dosyanın türünü belirlemek için dosyanın uzantısına bakılmaktadır. Sihirli bayt tabanlı yöntemlerde dosya türünü belirlemek için dosyanın başlığında yer alan sihirli baytlar okunmakta ve bu okunan sihirli baytlar

daha önceden tanımlanmış sihirli baytlar ile karşılaştırılmaktadır. İçerik tabanlı yöntemlerde ise dosya türünün belirlenmesi için istatistiksel modelleme teknikleri kullanılmaktadır [3, 4].

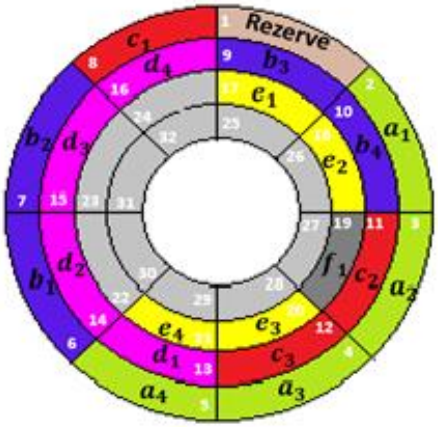


Şekil 1. 1: Dosya türü sınıflandırmada kullanılan yöntemler.

Dosyalar sürücülerde optimal veya parçalanmış bir şekilde saklanabilmektedir. Şekil 1.2'de sürücü içerisindeki bir plaka, bu plakadaki en küçük küme büyüklüğü ve bu plakaya kaydedilmiş dosyalar görülmektedir. Her bir sırada sekiz tane olmak üzere toplamda bu plaka otuz iki tane kümeden oluşmaktadır. Şekildeki bu plaka yer alan dosyalardan a ve d depolanırken birbirine bitişik kümeler dizisi şeklinde depolanmaktadır. Bu tür depolamaya optimal depolama adı verilmektedir. Şekildeki b, c ve e dosyalarında olduğu gibi veriler sürücülerin farklı alanlarında birden çok kümeye bölünüp parçalı bir şekilde de saklanabilmektedir. Bu tür depolama türüne ise parçalanmış (fragmented) depolama adı verilmektedir. Adli olaylardaki dosya türleri ve yapıları incelendiğinde dosyaların çoğunlukla parçalı bir şekilde saklandığı görülmektedir.

Uzantı tabanlı yöntemlerde dosyanın türünün belirlenmesi için dosyanın açılıp okunmasına gerek olmadığından dolayı çok hızlı bir yöntemdir. Fakat dosyaların uzantıları kolaylıkla değiştirilebildiği için güvenli bir yöntem değildir. Aynı şekilde sihirli bayt tabanlı yöntemlerde de dosyanın türünü dosyadaki sihirli bayt bilgilerini kullanarak kolaylıkla belirleyebilmektedir. Bu yöntemde sihirli baytların kontrol edilmesi için dosyanın açılması ve sihirli baytların okunması gerekmektedir. Okunan sihirli baytlar daha önceden tanımlanmış sihirli baytlar ile karşılaştırılmaktadır. Bazı dosya türleri için tanımlanmış sihirli baytların olmaması ve sihirli baytlar dosyaların boyutlarına göre değişebildiğinden dolayı bu yöntem de güvenilir bir yöntem değildir.

Bu iki yöntem güvenli olmamakla beraber optimal bir şekilde saklanan dosyaların türlerini belirlemekte kullanılabilmektedir. Parçalı bir şekilde saklanmış dosyalarda dosyanın uzantı ve sihirli bayt bilgileri eksik veya bozuk olabilmektedir. Parçalı dosyalarda silinme veya başka dış etmenlerden dolayı dosya kümeleri aralarındaki fiziksel bağlantılar ve verilerin konum bilgileri kaybolabilmektedir. Bu gibi durumlarda uzantı tabanlı ve sihirli bayt tabanlı yöntemler etkisiz hale gelmektedir [8].



Şekil 1. 2: Sürücü içerisindeki bir plaka ve bu plakaya kaydedilmiş dosyalar.

İçerik tabanlı yöntemlerde dosya türünü belirlemek için istatistiksel modelleme teknikleri kullanılmaktadır. Dosyanın uzantısının değişmesi veya sihirli bayt bilgilerinin değişmesi dosyanın içerik bilgilerini değiştirmemektedir. Bu yöntem ile optimal ve parçalı biçimde saklanmış dosyaların türlerini belirlemek mümkün olabilmektedir. Dosya parçaları dosyanın bir alt kümesi olduğu için dosya türü hakkında bilgi taşımaya devam etmektedir. Tüm dosya veya dosyanın bir parçasından dosyanın türünü belirlemek için dosyanın içerik bilgileri kullanıldığından dolayı dosya uzantısı ve sihirli bayt tabanlı yaklaşımlara göre daha güvenilir bir yöntemdir. Fakat dosyanın içeriğinde, dosyanın uzantısı veya sihirli baytlara göre dosyanın türüne dair daha az bilgi taşıdığı için bu yöntem ile dosyanın türünü tespit etmek çok daha zordur [7, 9].

Dosya türünü tespit etmenin veya belirlemenin ilk adımını dosyanın sınıflandırması oluşturmaktadır. Dosyaların sınıflandırılabilmesi için son yıllarda makine öğrenmesi algoritmaları yaygın olarak kullanılmaktadır. Grafik işlemci birimlerinin (GİB) gelişmesi, GİB'lerin hesaplama işlemlerinde kullanılması ve bilgisayarların çalışma hızlarının artması ile beraber makine öğrenmesi algoritmalarından derin sinir ağ yaklaşımını ön plana çıkmaktadır. Derin sinir ağ yaklaşımının son yıllardaki uygulama

alanları da günden güne artmaktadır ve sınıflandırma problemlerinde yaygın olarak kullanılmaktadır. Derin sinir ağları kullanılmaya başlandığı çoğu alanda diğer istatistiksel veya diğer makine öğrenmesi algoritmalarına göre daha iyi performans elde edildiği görülmektedir. Hesaplama hızlarının artması ile beraber derin sinir ağlar ile veriden anlamlı bilgi çıkarma çok daha kolay ve daha az maliyetli hale gelmektedir. Teknolojinin getirdiği bu yenilikler göz önüne alındığında derin sinir ağlarının içerik tabanlı dosya türü tespitinde kullanılmasının doğruluk oranını artıracağı öngörülmektedir.

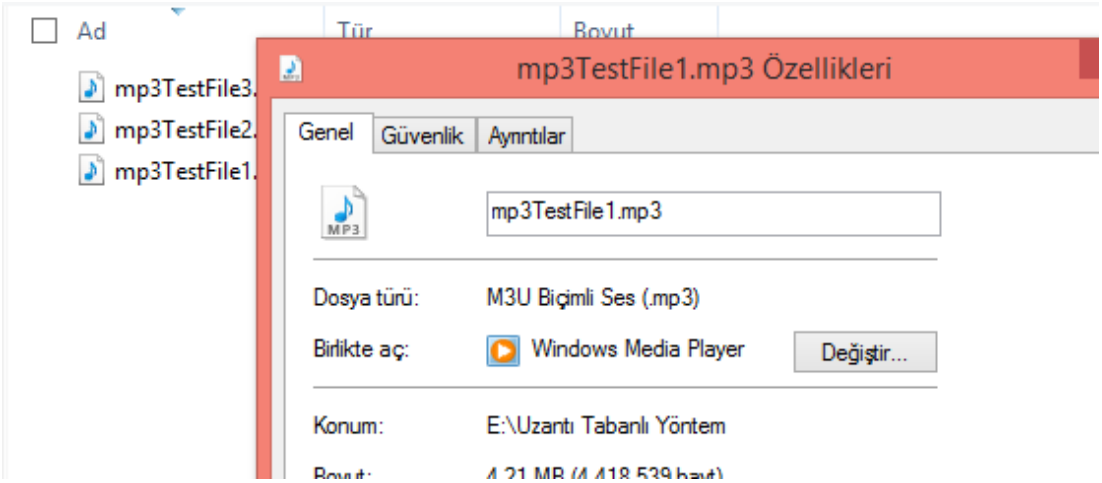
Günümüzde bilgisayarlarda, telefonlarda ve fotoğraf makinalarında yaygın olarak kullanılan görüntü, ses ve metin tabanlı 24 dosya türü belirlenmiştir. Bu çalışmada bu 24 dosya türü 15 veri türüne dönüştürülmüş olup daha sonra bu veri türleri sınıflandırılırmıştır. Önerilen yöntemde sınıflandırma probleminin çözümü için hiyerarşik bir sınıflandırma modeli ve derin sinir ağ yaklaşımı kullanılmıştır. İşletim sistemlerince kullanılan en küçük parça boyutu olan 4 kilobayt ve 8 kilobaytlık dosya parçaları üzerinde derin sinir ağları eğitilmiş ve test edilmiştir. Bu çalışmanın, dosya parçası kullanan içerik tabanlı dosya veya veri türü sınıflandırma çalışmaları ile performansı karşılaştırılmış ve bu karşılaştırma sonucunda doğruluk oranının arttığı görülmüştür.

2. DOSYA TÜRÜ TESPİTİ İÇİN KULLANILAN MEVCUT YÖNTEMLER

Dosya türü tespiti adli bilişim ve bilgi güvenliği için önemli bir konu olmakla beraber araştırmacılar bu konu üzerinde yıllardır çalışmaktadır. Dosya türü tespiti için kullanılan yöntemler uzantı tabanlı, sihirli bayt tabanlı ve içerik tabanlı yöntemlerdir. Bölüm 2.1'de uzantı tabanlı yöntemler anlatılmaktadır. Bölüm 2.2'de sihirli bayt tabanlı yöntemler ve bölüm 2.3'te de içerik tabanlı yöntemler anlatılmaktadır.

2.1 Uzantı Tabanlı Yöntemler

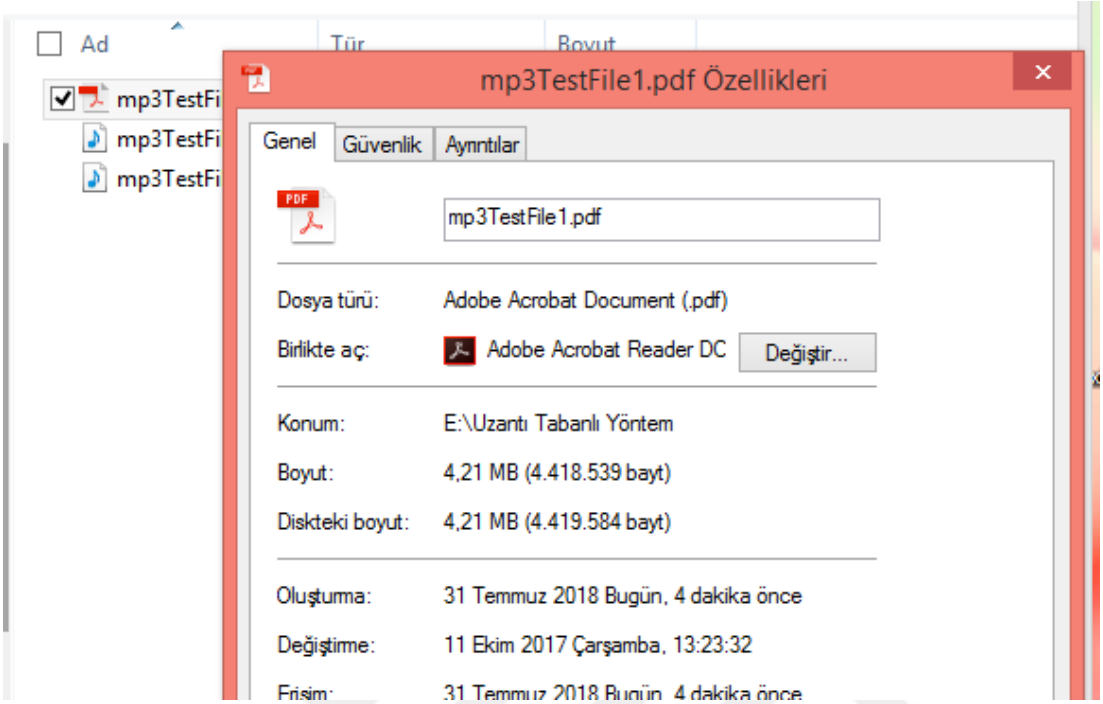
Dosya türü tespitinde kullanılan en basit yöntem uzantı tabanlı yöntemdir. Her dosyanın bir uzantısı vardır ve bu uzantı dosyayı uygun yazılımla ilişkilendirmektedir. Bu yöntemde dosyanın türünün belirlenmesi için dosyanın açılıp okumaya gerek yoktur. Dosyanın uzantı bilgisine bakılarak dosyanın türü kolaylıkla tespit edilebilmektedir. Resim 2.1'de türü değiştirilecek bir dosya ve bu dosyanın türünün nasıl tespit edileceği görülmektedir. Windows işletim sistemlerinde dosyanın özelliklerinden dosyanın türü kolaylıkla belirlenebilmektedir.



Resim 2. 1: Türü değiştirilecek dosya ve dosyanın türünün tespit edilmesi.

Bu yöntemin en büyük dezavantajı uzantısının kolaylıkla değiştirilebilmesidir. Resim 2.1'deki mp3TestFile1.mp3 dosyasının uzantısı Resim 2.2'de uzantısı değiştirilmiş

dosya ve bu dosyanın türü yer almaktadır. Dosyanın uzantısının değiştirilmesi çok kolay bir işlemdir ve herkes tarafından kolaylıkla yapılabilmektedir.



Resim 2. 2: Türü değiştirilmiş dosya.

Dosyaların uzantılarının değiştirilmesi kanıtları gizlemek için kullanılabilecek en kolay yoldur. Uzantısı değiştirilmiş bir dosya ile adli bilişim uzmanları kolaylıkla kandırılabilmektedir. Ancak uzantısı değiştirilmiş dosyaları tespit etmek için adli bilişim uzmanlarının kullandığı yazılımlar vardır. Encase ve Autopsy bu tür yazılımlara örnektir ve türü değiştirilmiş dosyaları kolaylıkla tespit edilebilmektedir.

2.2 Sihirli Bayt Tabanlı Yöntemler

Dosya türü tespiti için kullanılacak bir başka yöntem sihirli bayt tabanlı yöntemlerdir. Sihirli baytlar dosyanın başlık bölümünde yer almaktadırlar. Şekil 2.1'de basit bir dosya yapısı görülmektedir. Şekilde de görüldüğü üzere dosyanın başlığı dosyanın ilk kısmındadır ve dosyanın meta verilerini içermektedir. Meta veriler ise dosyanın içeriği hakkında bilgiler içermektedir.

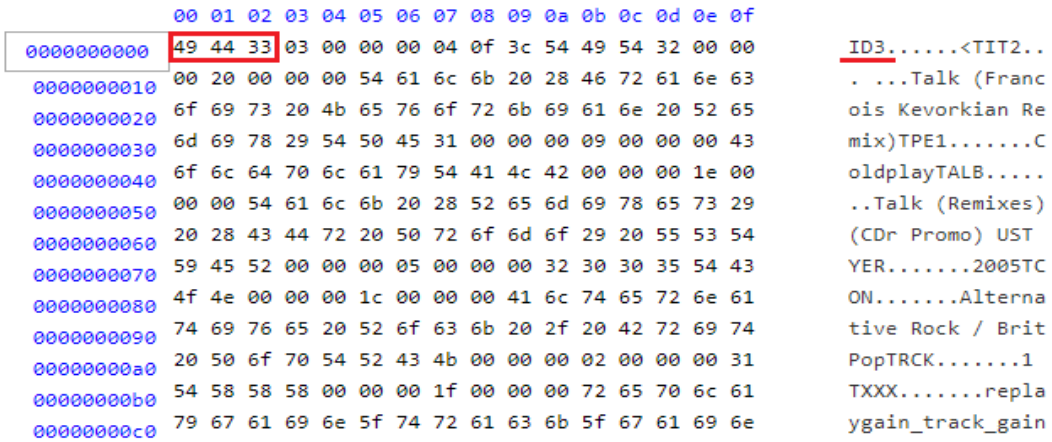
Dosya Başlığı	Dosya Gövdesi	Dosya Altbilgisi

Şekil 2. 1: Basit bir dosya yapısı.

Sihirli baytlar tabanlı yöntemler kullanılarak dosyanın türünün tespit edilebilmesi için öncelikle dosyanın açılması ve dosyanın başlığında yer alan sihirli baytlarının okunması gerekmektedir. Sihirli baytlar dosya imzaları olarak da adlandırılmaktadır. Sihirli baytlar daha önceden tanımlanmış sihirli baytlar ile karşılaştırılarak dosyanın türü tespit edilebilmektedir.

Çizelge 2 1: Bazı dosya türlerinin imzaları.

Dosya Türü	Dosya İmzaları
JPG	FF D8 FF E0 FF D8 FF E1 FF D8 FF E8
PNG	89 50 4E 47 0D 0A 1A 0A
GIF	47 49 46 38
BMP	42 4D
M4A	00 00 00 20 66 74 79 70 4D 34 41
PDF	25 50 44 46
MP3	49 44 33



Resim 2. 3: MP3 dosyasının ikili kodları.

Çizelge 2.1'de bazı dosya türlerinin önceden tanımlanmış imza bilgileri yer almaktadır [10]. Bu tablodaki değerler on altılı sayı (hexadecimal) türündendir. JPEG dosya türü için belirlenmiş 3 tane imza vardır. Diğer dosya türleri için bir tane imza bilgisi olmasına karşın her dosya türünün imza uzunlukları farklılık göstermektedir.

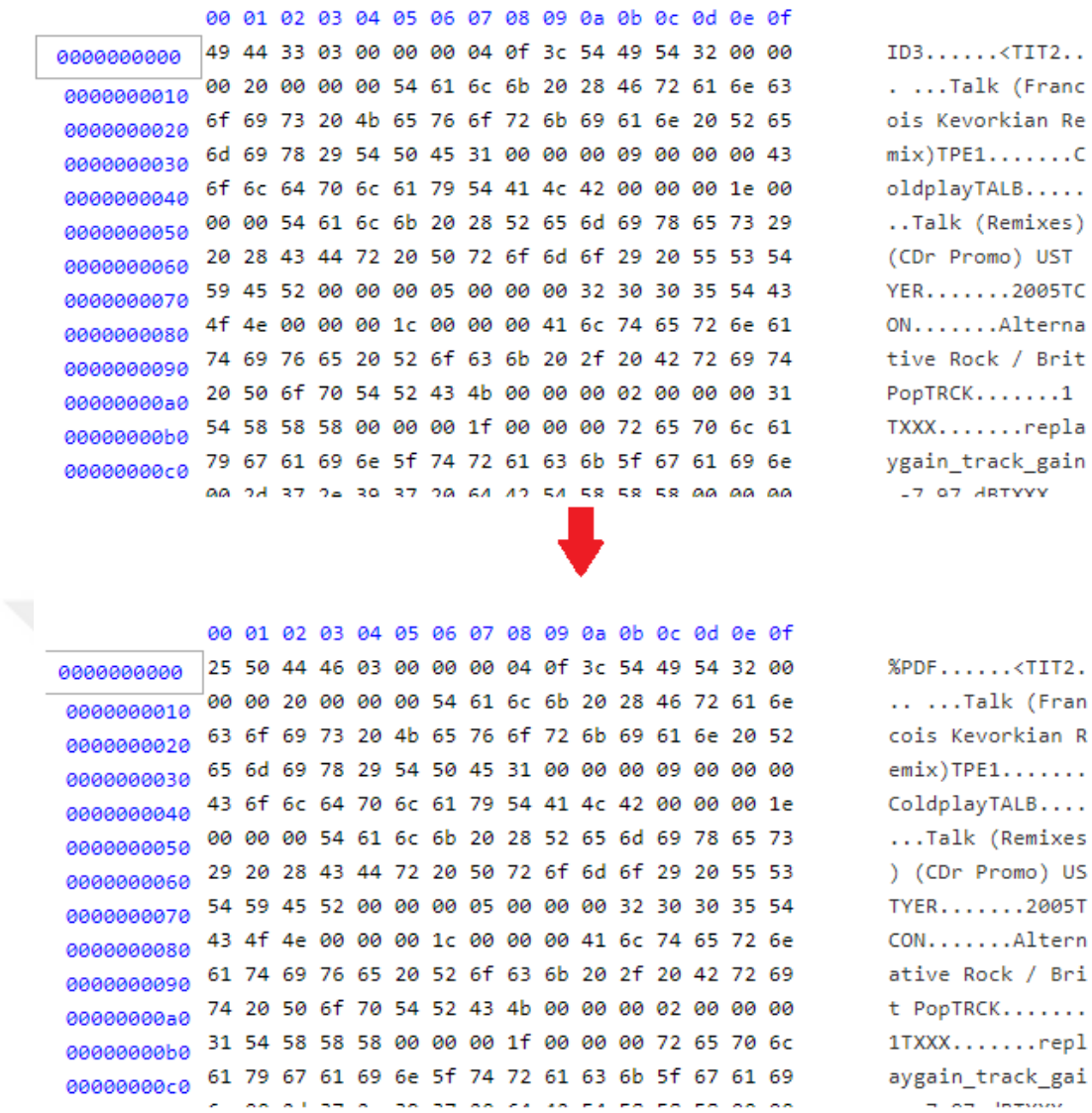
Resim 2.3'te MP3 dosyasının ikili kodları yer almaktadır [11]. Bu ikili kodlar incelendiğinde ikili kodların başlarında on altılı sayı türünden 49 44 33 yer aldığı görülmektedir. Bu on altılı sayı daha önceden tanımlanmış imzalarla karşılaştırıldığında MP3 dosyasının imzası ile eşleşmektedir.

Uzantı tabanlı yöntemlerde olduğu gibi sihirli bayt tabanlı yöntemlerde de dosya üzerinde değişiklik yapılarak yetkililerin kandırılması çok kolaydır. Dosyanın imzalarının değiştirilebilmesi için öncelikle dosyanın ikili kodlarının okunması gerekmektedir. İkili kodlar okunduktan sonra dosyanın imzaları istenilen dosyanın imzalarına dönüştürülebilir veya imzalar silinebilir.

Sihirli bayt tabanlı yöntemlerin hızlı bir yöntem olmasına karşın dezavantajları da vardır. Dosya imzaları için oluşturulmuş farklı farklı veri tabanları vardır ve bu veri tabanlarında bazı dosya türlerinin imza bilgileri farklılık göstermektedir. Örneğin M4A dosyasının imzası bir veri tanından 00 00 00 20 66 74 79 70 4D 34 41 [10] iken diğer veri tabanında 66 74 79 70 4D 34 41 20'dir [12]. Ayrıca her dosya türünün imzası olmadığı gibi bazı dosya çeşitlerinin boyutlarına göre değişebildiği için her dosya türü için kullanışlı bir yöntem değildir.

Resim 2.4'te sihirli baytları değiştirilmiş bir MP3 dosyasının sihirli baytları değiştirilmeden önceki ve değiştirildikten sonraki ikili kodları yer almaktadır [11]. Deneylerde kullanılan MP3 uzantılı dosya okunup ilk üç sırada yer alan 49 44 43 (MP3 dosyasının imzası) olan baytları 25 50 44 46 (PDF) ile değiştirilip tekrardan kaydedilmiştir. Yapılan çalışmada göstermektedir ki sihirli bayt tabanlı yöntemlerde hem dosyanın türünü değiştirmek hem de dosyanın türünü belirlemek uzantı tabanlı yöntemlere göre daha maliyetli bir işlemdir.

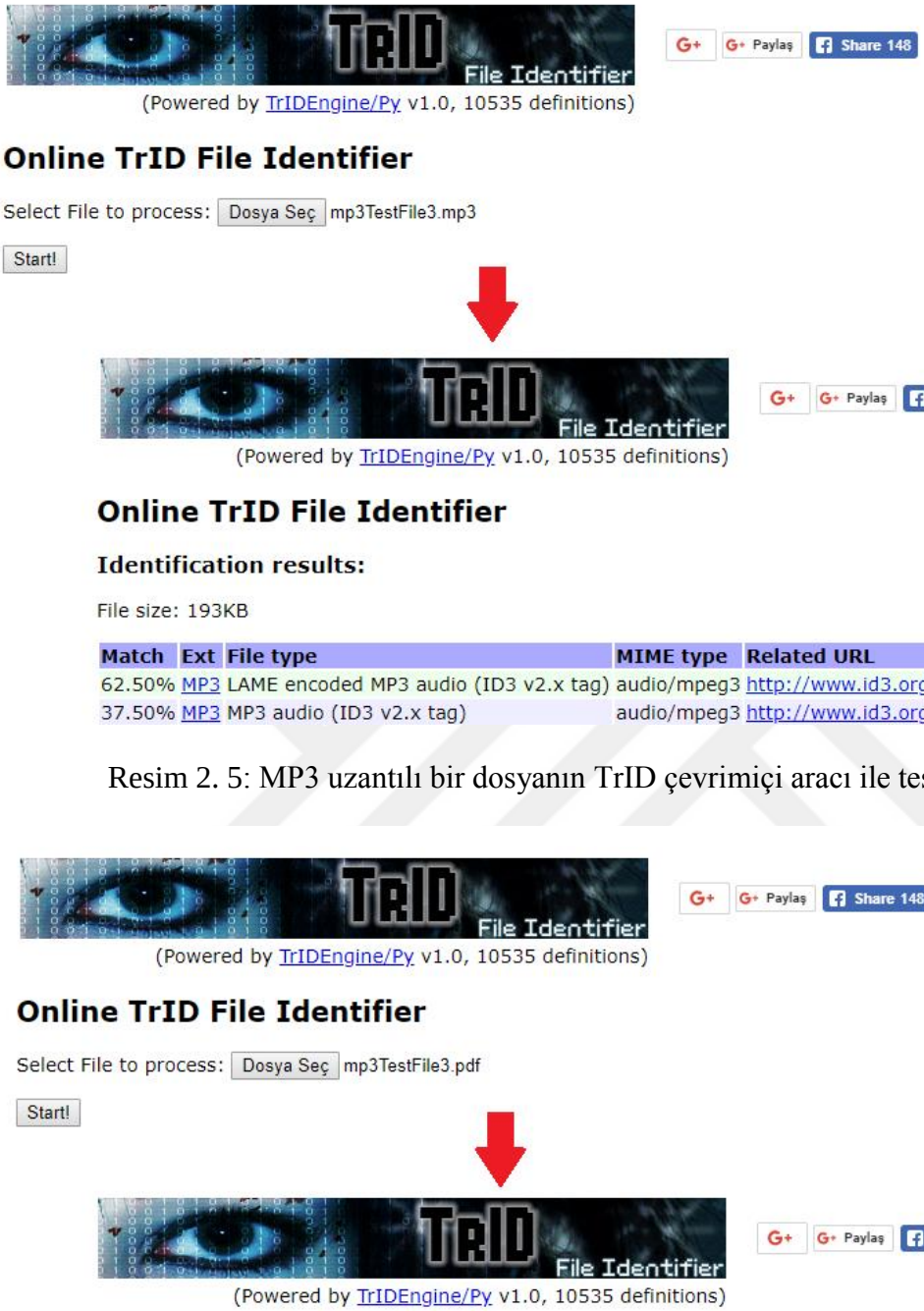
Dosya türlerini belirleyebilmek için TrID ve file gibi açık kaynak kodlu araçlar bulunmaktadır. Bu araçlar dosyanın sihirli bayt bilgilerini kullanılarak dosyaların türünü tespit etmektedir.



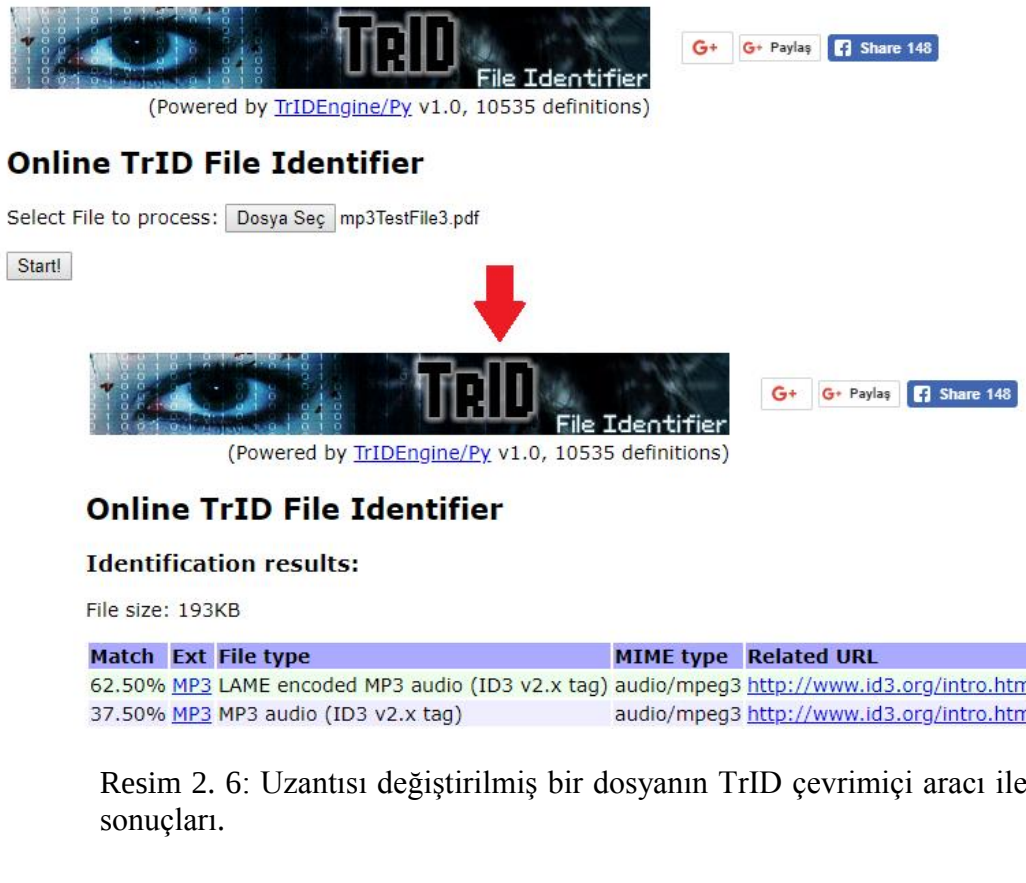
Resim 2. 4: MP3 dosyasının sihirli baytları değiştirildikten önceki ve değiştirildikten sonraki ikili kodları.

Resim 2.5'de MP3 uzantılı bir dosyanın TrID çevrimiçi aracı ile test sonuçları yer almaktadır [13]. Bu dosya %62,5 oranında LAME şifrelenmiş MP3 ses dosyası ve %37,5 oranında MP3 ses dosyası olarak bulunmuştur. Toplamda %100 MP3 ses dosyasıdır.

Resim 2.6'da ise bir önceki testte kullanılan MP3 dosyasının uzantısı PDF olarak değiştirilmiş ve uzantısı değiştirilmiş bu dosyanın TrID çevrimiçi aracı ile test sonuçları yer almaktadır [13]. Sonuçta da görüldüğü üzere uzantısı değiştirilmiş dosyada %100 oranında MP3 dosyası olarak bulunmuştur. Bu iki testinde sonuçlarının aynı çıkmış olması bu aracın analiz yaparken dosyanın uzantı bilgilerini kullanmadığını göstermektedir.



Resim 2. 5: MP3 uzantılı bir dosyanın TrID çevrimiçi aracı ile test sonuçları.

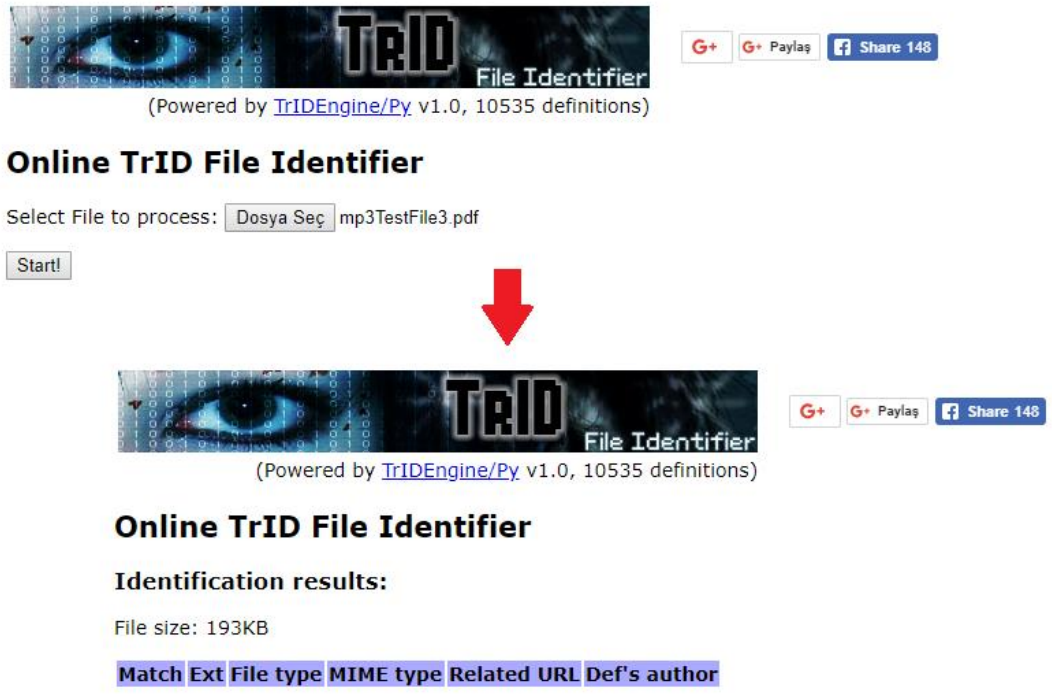


Resim 2. 6: Uzantısı değiştirilmiş bir dosyanın TrID çevrimiçi aracı ile test sonuçları.

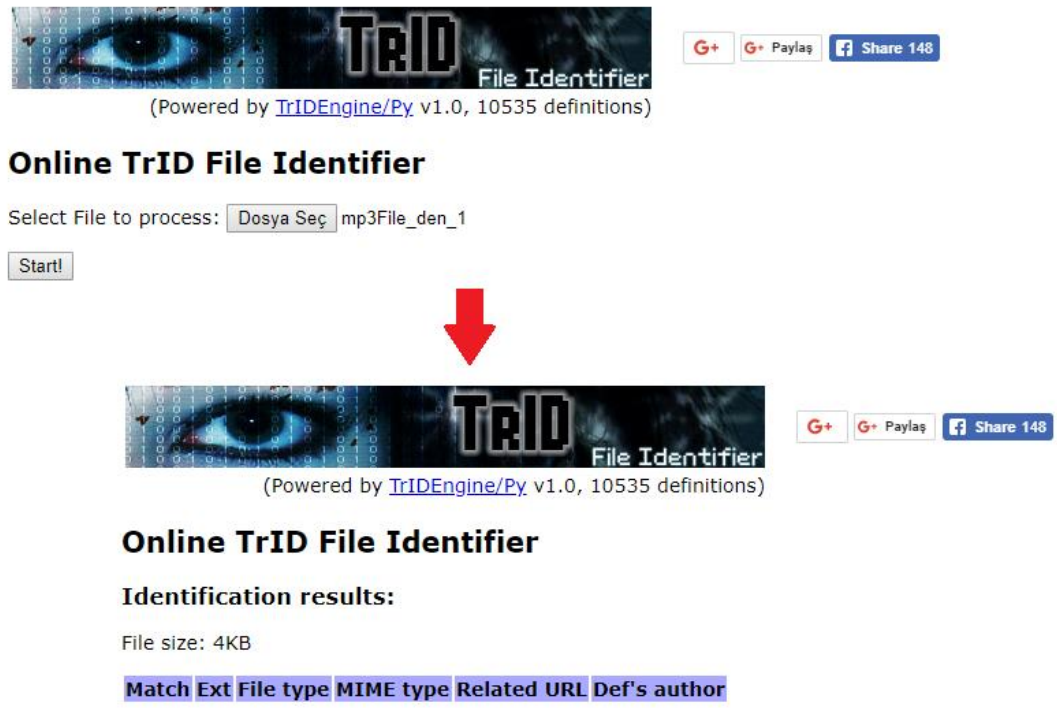
Deneylerde kullanılan MP3 uzantılı dosyanın sihirli baytları değiştirildikten sonra (PDF – 25 50 44 46) dosya tekrardan TrID aracı ile analiz edilmiştir. Analiz sonuçlar Resim 2.7'de yer almaktadır [13]. TrID çevrim içi aracı ile sihirli baytları değiştirilmiş bir dosyanın türü tespit edilememektedir.

TrID aracı ile son olarak parçalanmış dosyaların türünün belirleyip belirlemeyeceğini test etmek için 4 KB'lık bir MP3 dosya parçası üzerinde test edilmiştir. Resim 2.8'de parçalı dosyanın tür tespit sonuçları yer almaktadır [13]. Analiz sonuçlarına göre parçalı dosyalarda da dosyanın türü tespit edilememiştir.

Bu testler sonucunda TrID çevrimiçi aracının sihirli bayt bilgileri değiştirildiğinde veya olmadığı durumlarda dosyanın türünün tespit edilemeyeceğini göstermektedir. TrID aracı için sadece sihirli bayt bilgilerine bağlı denilmesi mümkün olmamakla beraber dosyanın türünün tespitinin önemli bir kısmını sihirli bayt bilgileri oluşturmaktadır.



Resim 2. 7: Sihirli bayt bilgileri değiştirilmiş dosyanın TrID çevrimiçi aracı ile test sonuçları.



Resim 2. 8: Dosya parçası için TrID çevrimiçi aracının test sonuçları.

2.3 İçerik Tabanlı Yöntemler

Dosya türü tespitinde kullanılan üçüncü yöntem istatistiksel modelleme tekniklerinin kullanıldığı içerik tabanlı yöntemlerdir. Bu yöntem ile dosyaların uzantıları veya sihirli baytları değiştirilmiş, silinmiş ve sürücülerde bir kısmı kalmış dosyaların içerik bilgileri kullanılarak gerçek türünün tespit edilebileceği tek yöntemdir.

Dosya türü tespiti için içerik tabanlı yaklaşımı öneren McDaniel ve Heydari'dir [6]. İçerik tabanlı yöntemin önerilmesi ile beraber bu alanda pek çok araştırma yapılmıştır. Bu alandaki yapılan 28 çalışma incelendiğinde 8 çalışmada dosya başlıkları da dahil olmak üzere dosyaların tümünün ele alındığı çalışmalardır (optimal saklanmış dosyalar) [5-7, 9-17]. Yapılan çalışmaların 3'ünde hem tüm dosya bilgileri hem de dosya parça bilgileri kullanılmıştır [4, 18, 19]. Diğer 17 çalışmada ise sadece dosya parça bilgileri kullanılmıştır [1, 8, 20-34]. Dosya parçası türü çalışmaları ile yalnızca bilgisayarın dosya türü sınıflandırması için değil, aynı zamanda veri paketleri ve dosya parçalarının tür sınıflandırması içinde kullanılabilmektedir.

Literatürde yer alan çalışmaları tüm dosyaların kullanıldığı ve dosya parçalarının kullanıldığı çalışmalar olmak üzere ikiye ayrılmaktadır ve bölüm 2.3.1 ve 2.3.2'de daha ayrıntılı olarak anlatılmaktadır.

2.3.1 Dosya Türü Sınıflandırılması

Dosya türü sınıflandırması dosyanın başlık bilgileri dahil tüm dosya bilgilerinin kullanılarak yapıldığı çalışmalar olarak tanımlanabilmektedir. Dosyanın başlık bilgisindeki sihirli baytlar dosya türü hakkında ayırt edici bilgiler içermektedir. Literatürde yer alan içerik tabanlı dosya türü sınıflandırması çalışmalarından sekizinde tüm dosyanın içerik bilgileri kullanılmıştır. Bu yapılan çalışmaların yedisinde on veya daha az dosya türü kullanılmıştır [5, 7, 9-17]. McDaniel ve diğ. [6] yaptığı çalışmada otuz dosya türü kullanılmış ve öznitelik olarak bayt frekans dağılımı (BFD), bayt frekans çapraz korelasyonu ve dosya başlık/fragmanı da kullanılmıştır. Bu yapılan çalışmaya göre dosya başlık/fragman algoritması ile çıkarılan öznitelikler ile daha iyi sınıflandırma başarısı elde edildiği belirtilmiştir. Bayt frekans dağılımı bu alandaki diğer çalışmalarda da kullanılmış olup Cao [15] ve Dunham'ın [16] çalışmalarında bu özniteliklere ek olarak çeşitli karmaşıklık ölçütleri de kullanılmıştır. Bazı çalışmalarda öznitelik seçiminden sonra temel bileşen analizi ile birlikte sinir ağları, genetik algoritma ve gram dağılımı temelli öznitelik seçim algoritmaları kullanılmıştır. Bu yapılan çalışmalarda sınıflandırma işlemi için sinir ağları (SA), lineer diskriminat analizi (LDA), kosinüs benzerliği ve Mahalanobis uzaklığı gibi algoritmalar kullanılmıştır. Bu yapılan çalışmaların hepsinde 90% üzerinde doğruluk oranı elde edilmiştir.

2.3.2 Dosya Parçası Sınıflandırılması

Tüm dosya ve dosya parçalarının beraber incelendiği çalışmalarda dosya parçasından dosya türü tespit performansında ciddi düşüşler yaşandığı görülmüştür ancak dijital adli tıp uygulamaları için dosya parçasından dosya türü tespiti daha önemli bir konu olduğu için araştırmacıların çoğu bu alana odaklanmışlardır [4, 18, 19].

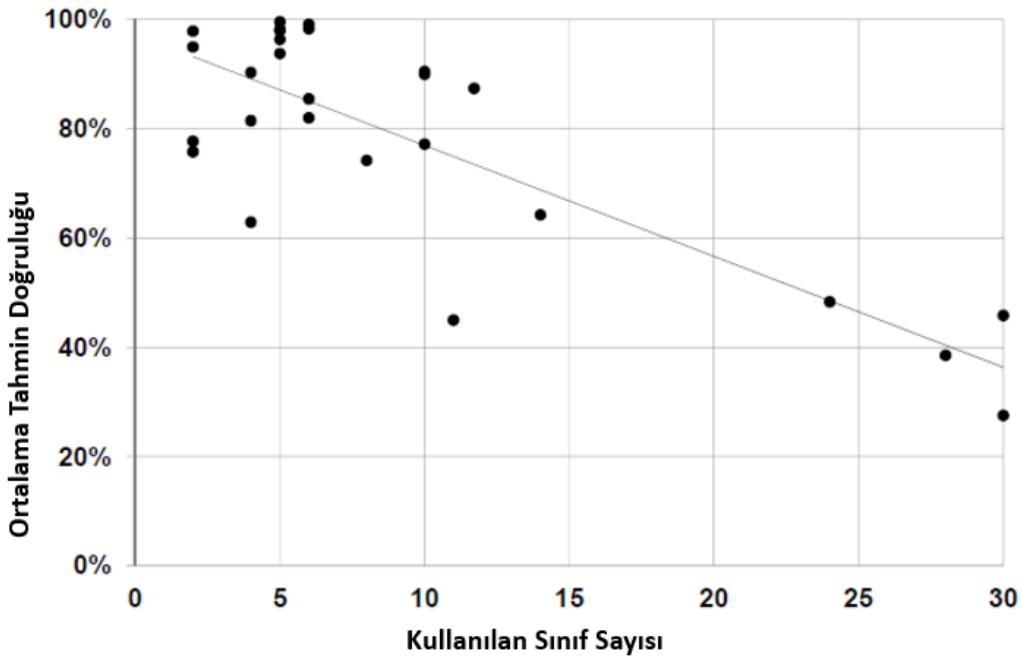
Hem tüm dosyanın hem de dosya parçalarının beraber incelendiği çalışmalarda BFD ile öznitelikler çıkarılmış ve daha sonra farklı sınıflandırma algoritmaları kullanılarak karşılaştırmalı değerlendirmeler yapılmıştır. Ahmed ve diğ. [18, 19] yaptığı çalışmada 6 farklı sınıflandırma algoritması kullanılmıştır. Sinir ağları (SA), lineer diskriminat analizi (LDA), K ortalamalar, K en yakın komşuluk, karar ağaçları (KA) ve destek vektör makineleri (DVM) algoritmalarının performansları karşılaştırmıştır ve K en yakın komşuluk algoritmasının diğer algoritmalara göre daha iyi performans verdiği gösterilmiştir. Amirani ve diğ. [4] yaptığı çalışmada sınıflandırma için SA ve DVM

kullanılmıştır. Hem tüm dosyanın hem de dosya parçalarının sınıflandırmasında DVM'ının daha iyi performans verdiği gösterilmiştir.

Dosya parçası türü tespiti için 15 yıldır çalışmalar hızlı bir şekilde devam etmektedir. Yapılan bu çalışmalarda öznitelik çıkarmak için çeşitli yöntemler kullanılmaktadır. Bu alanda yapılan çalışmalarda BFD, 2-gram, Shannon entropy, ortalama bayt değeri, Kolmogorov karmaşıklığı ve Hamming ağırlığı gibi öznitelik çıkarma yöntemleri kullanılmaktadır [1, 8]. Bu yöntemlerden en yaygın olanı BFD'dir ve 9 çalışmada öznitelik çıkarmak için bu yöntem kullanılmıştır [1, 8, 22, 25, 28-34]. Gopal [27] ve Fitzgerald'in [26] yaptığı çalışmalarda BDF (1-gram) ile 2-gram'ın karşılaştırmalı sonuçları verilmiştir. Bu iki araştırmacının verdiği sonuçlara göre 2-gram ile elde edilen özniteliklerle daha başarılı sonuçlar elde edilmiştir. Diğer çalışmalarda da çeşitli karmaşıklık ölçüleri kullanılmıştır [20, 21, 23, 24]

Öznitelikler çıkartıldıktan sonra sınıflandırma algoritmaları uygulanmıştır. Yapılan bu çalışmalarda sınıflandırma işlemi için üç algoritma öne çıkmaktadır. Bu algoritmalar DVM, LDA ve K- en yakın komşuluk algoritmalarıdır. Bu yapılan çalışmalardan büyük bir kısmında DVM kullanılmıştır [1, 8, 20, 26, 27, 30]. Bu çalışmaların sonucuna göre lineer çekirdek fonksiyonunun diğer çekirdek fonksiyonlarına (polinom, radyan temelli ve sigmoid) göre daha etkili olduğu vurgulanmıştır [1, 8, 26]. 4 çalışmada K- en yakın komşuluk algoritması kullanılmış ve diğer sınıflandırma algoritmaları ile karşılaştırıldığında daha gürbüz olduğu belirlenmiştir [4, 21, 24, 27]. 3 çalışmada LDA algoritması kullanılmıştır [20, 23, 33] ve kalan çalışmalarda ise diğer makine öğrenmesi algoritmaları ya da istatistiksel yöntemler kullanılmıştır.

Resim 2.8’de içerik tabanlı dosya türü sınıflandırması alanında yapılan çalışmaların ortalama tahmin doğrulukları yer almaktadır [22]. Resimde de görüleceği üzere 10 ve altı dosya türü kullanılarak yapılan çalışmaların çoğunda doğruluk oranları %80 ve üzerindedir. Dosya sayısının artması ile beraber doğruluk oranları düşmektedir. 10’dan fazla dosya türü kullanılan çalışmalar sınırlı sayıda olmasıyla beraber bu çalışmalarda doğruluk oranları %70 seviyelerine düşmektedir [1, 8, 22, 24, 26, 33].



Resim 2. 9: İçerik tabanlı dosya türü sınıflandırması alanında yapılan çalışmaların ortalama tahmin doğrulukları.

2007 yılı itibariyle Erbaher ve Mulholland [25] tarafından dosya türü ve veri türü olmak üzere iki farklı tanım yapılmıştır. Dosya türü, dosyayı oluşturmak veya dosyaya erişmek için kullanılan uygulamanın belirlediği genel dosya türü olarak tanımlanmıştır. Veri türü ise dosyaya gömülmüş verilerin türü olarak tanımlanmıştır. Örneğin Microsoft Word dosyası metin, görüntü veya tablo içerebilmektedir. Böylesi bir dosyada dosya türü tek iken içerisinde birden çok veri türü vardır. Bu tanımın ortaya çıkması ile beraber bazı çalışmalarda dosya ve veri türü sınıflandırmasına odaklanılmıştır [1, 8, 22, 25, 31]. Zheng ve diğ. [1] yaptığı çalışmada dosya türü ile veri türü arasında performans karşılaştırılması yapılmıştır. Dosya türleri veri türlerine dönüştürülmüş ve bu dönüşüm sonucunda doğruluk oranının %21 arttığı gösterilmiştir. Dosya ve veri türü sınıflandırması yapan diğer çalışmalarda ise dosya türleri ve veri türleri beraber incelenmiştir.

Bizim çalışmamızda da en çok kullanılan dosya türleri belirlenmiş ve bu dosya türleri veri türlerine dönüştürülmüştür. Türler belirlendikten sonra rastgele bu türlerden veri toplamak için Google hacking yöntemi ile ilgili türlerin bulunabileceği web adresleri belirlenmiştir. Bu web adreslerindeki verilerin otomatik olarak indirilebilmesi için Jsoup yani hazır Java kütüphanesi kullanılmıştır. Veri türleri toplandıktan sonra eğitim ve test olmak üzere toplanan veriler ikiye ayrılmıştır. Eğitim veri seti içerisindeki her bir veride istenilen boyutta fragmanlar seçildikten sonra n-gram analizi ile öznitelikler

çıkartılıp bu öznitelikleri kullanıp sınıflandırma için hiyerarşik bir sınıflandırma modeli kullanılmış olup ilk hiyerarşide çeşitli makine öğrenmesi algoritmaları test edilip ikinci hiyerarşide ise derin sinir ağlar ile daha başarılı cevap verilebileceği varsayılmıştır.

3. ARKA PLAN BİLGİSİ

Bölüm 3.1'de tür tespiti yapılacak dosyaların internet ortamından bulunması için kullanılan Google hacking yöntemi, bölüm 3.2'de Google hacking yöntemi ile bulunan dosyaları indirmek için kullanılan html ayrıştırma kütüphanesi olan JSOUP, bölüm 3.3'te öznitelik seçimi için kullanılan n-gram analizi, Bölüm 3.4'de sınıflandırma sistemleri için kullanılan rastgele orman algoritması, 3.5'te destek vektör makineleri ve 3.6'da derin sinir ağlar anlatılmıştır.

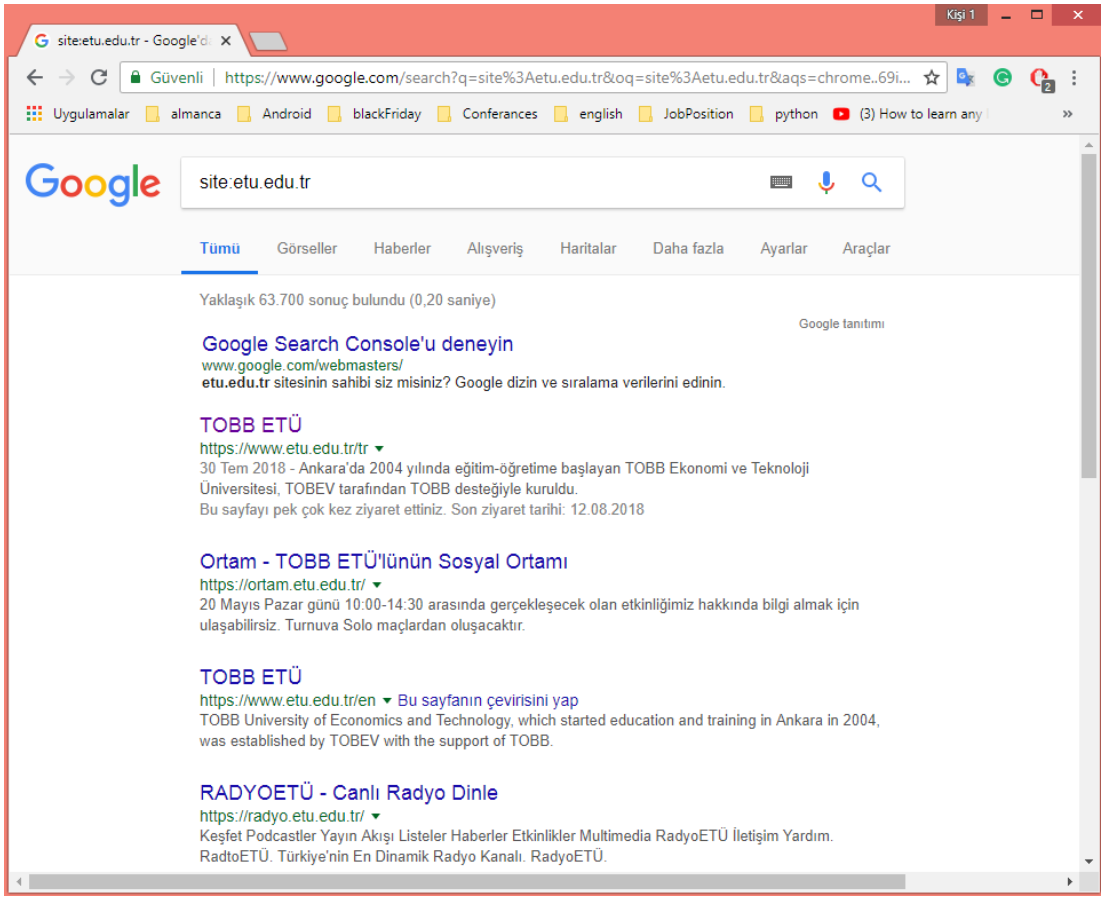
3.1 Google Hacking

Google web'de (WWW – World wide web) arama yapmak ve dizine eklenecek belgeleri bulmak için otomatik örümcekleri (spiders) veya Google robotlarını (Googlebots) kullanmaktadır. Google arama motoru kullanıldığında, kullanıcılar aslında Google dizini aramaktadır. Google dizinleri etkin hale getirmek için bulduğu her sayfanın bir kopyasını oluşturmaktadır ve bu kopyaları Google önbelleğine yerleştirmektedir. Aslında kullanıcıların kaynak dizine yönlendirilmek yerine dosyanın Google önbelleğine alınmış versiyonunu görüntüleme seçeneği de vardır.

Google hacking, bilgisayar güvenliği ile ilgili bilgileri bulmak için çok sayıda arama sonucunu filtrelemek amacıyla özel olarak oluşturulmuş karmaşık arama motoru sorguları oluşturma olarak tanımlanabilir. Bir site kendi sitesinden bazı bilgileri kaldırmış veya erişilemez hale getirmiş olabilmektedir. Kaldırılmış veya erişilemez hale getirilmiş hassas bilgilere genellikle Google önbelleğinden erişilebilmektedir. Yani İnternet'teki güvenlik sorunlarını bulmak için Google hacking teknikleri kullanılabilir [35, 36, 37].

Beyaz şapkalı hacker olan Johnn Long Google hacking veri tabanını (GHVT) oluşturan kişidir. GHVT web'de hassas verilerin yerini bilinen bir Google arama sorgusudur. Bu özel sorguları yapmak için kullanılan arama kelimelerin birkaçına örnekler verilecektir.

Site kelimesi ile arama yapıldığında belirtilen alan adına sahip sitelerde arama yapmaktadır. Resim 3.1'de "site:etu.edu.tr" alan adına sahip sitelerin sonuçları yer almaktadır. Bu alan adına sahip 63.700 sonuç bulunmuştur.

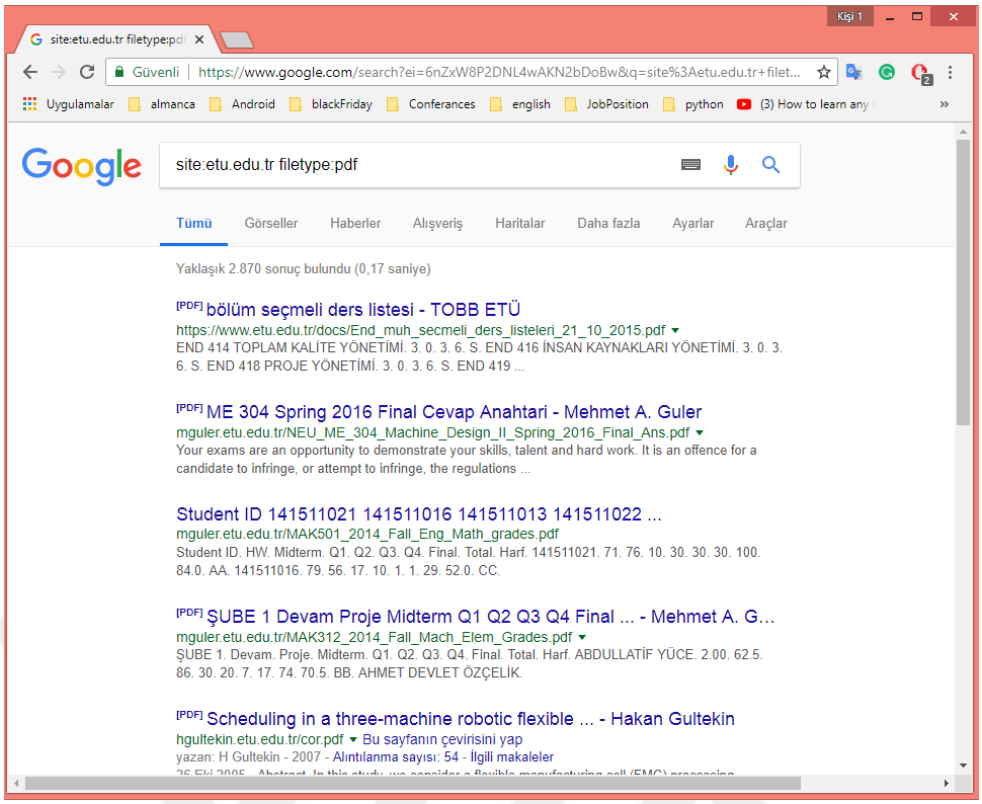


Resim 3. 1: site:etu.edu.tr arama sonuçları.

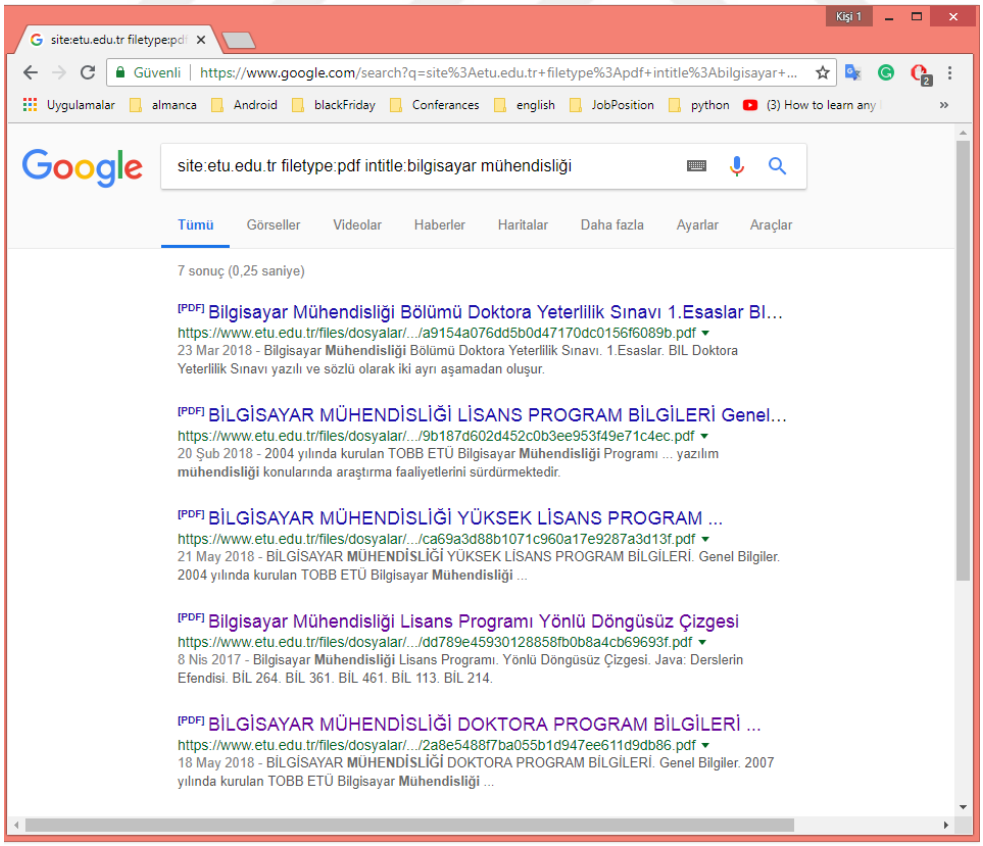
İstenilen dosya türünden verileri elde edebilmek için filetype anahtar kelimesi kullanılarak verilerin bulunduğu internet siteleri bulunmuştur. Bir önceki arama sonucuna "filetype:pdf" eklendiğinde etu.edu.tr adresindeki tüm pdf içeren siteler listelenmektedir. Resim 3.2'de filetype:pdf aramasının sonuçları yer almaktadır.

Belirlenen kelimeler intitle arama kelimesi ile arandığında sayfa başlığında bu kelimeler aranır ve listelenir. Resim3.3'de intitle:bilgisayar mühendisliği arama sonuçları yer almaktadır.

Site, filetype, intitle gibi daha pek çok özel arama kelimeleri ile istenilen verilere çok daha hızlı şekilde erişim sağlanabilmektedir. Diğer kullanılan arama kelimelerinin bazıları Çizelge 3.1'de yer almaktadır.



Resim 3. 2: filtype:pdf arama sonuçları.



Resim 3. 3: intitle: bilgisayar mühendisliği arama sonuçları.

Çizelge 3. 1: Google hacking için kullanılan özel kelimeler ve anlamları.

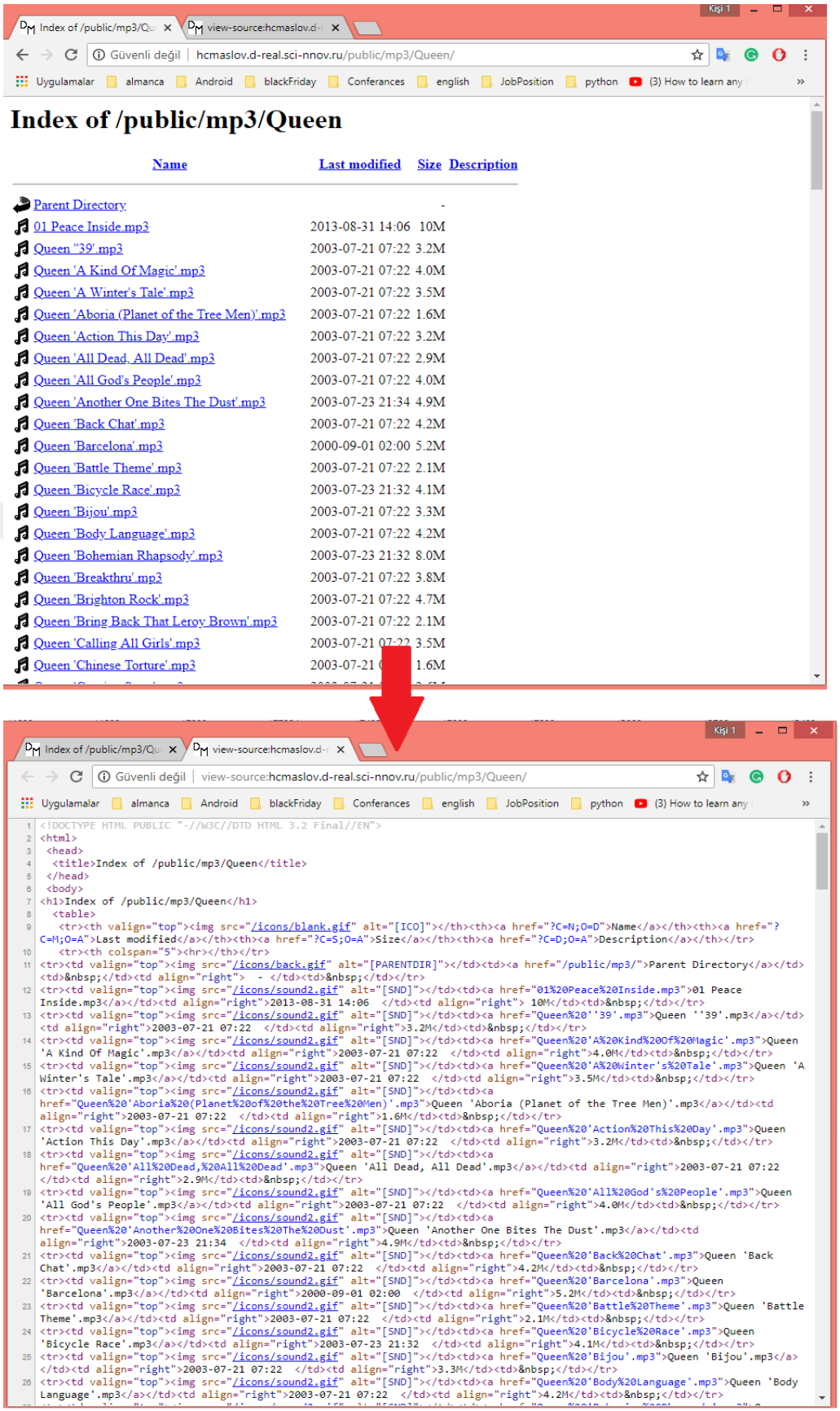
Anahtar Kelime	Kelimenin Anlamı
Site	İstenilen adrese ait sayfalar listelenir
Filetype	İstenilen uzantıya sahip dosyalar listelenir
İntitle	İstenilen kelimeyi sayfa başlığı içerisinde aranır ve sonuçlar listelenir
İnurl	İstenilen kelimeyi web sitesinin URL'sinin herhangi bir yerinde aranır ve sonuçlar listelenir
İntext	İstenilen kelimeler web sayfasının içerisinde aranır ve sonuçlar listelenir.
Link	İstenilen linke ait sayfalar listelenir

3.2 JSOUP kütüphanesi

Jsoup [38], HTML çözümlemeye yarayan bir Java kütüphanesidir. Bu kütüphane ilgili URL'ye erişen ve önceden ayarlanmış tarama parametrelerine dayanarak bir ön seçim yapmaktadır. Resim 3.4'de bir site ve bu siteye ait kaynak kodları yer almaktadır [39]. HTML sayfasının başka bir sayfaya bağlantısını sağlayan link <a> etiketi ile belirtilmektedir. Bu kaynak kodlar içerisindeki başka bir kaynağa veya sayfaya bağlantı <a> etiketi ile sağlanmıştır.

Çizelge 3. 2: Jsoup kütüphanesi kullanılarak yazılmış örnek bir Java kodu.

```
Document doc = Jsoup.connect
                ("http://hcmaslov.d-real.sci-nnov.ru/public/mp3/Queen/").get ();
Elements links = doc.select("a");
for (Element e: links){
        String newURL = e.attr("abs:href");
        String src = newURL.substring
                (newURL.lastIndexOf(".")+1, newURL.length());
        String src1 = "mp3";
        int result = src.compareTo(src1);
        if (result==1){
                getImages(newURL)}}
```

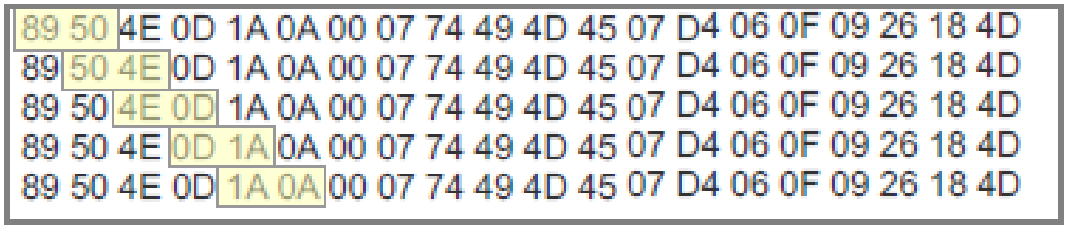


Resim 3. 4: Bir site ve bu siteye ait kaynak kodlar.

Çizelge 3.2'de jsoup kütüphanesi kullanılarak yazılmış örnek bir kod yapısı bu kaynak kodlarla incelenmektedir. URL'ye bağlantı yapmak için Jsoup.coonect() fonksiyonun içerisine istenilen URL adresi yazılarak bağlantı sağlanmaktadır. HTML sayfasındaki bütün linkleri almak için doc.select() fonksiyonu kullanılmaktadır. <a> etiketi ile başlayan bütün linkler bu kod ile alınmaktadır. İstenen dosyaları elde etmek için ilgili linkin son harfleri karşılaştırılır ve istenen uzantılı dosya ise getImages fonksiyonu ile kaydedilmektedir. getImages fonksiyonu istenilen linki kaydetmek için genel Java komutları kullanılarak yazılmış bir fonksiyon olduğu için burada ayrıntıları verilmemektedir.

3.3 N-Gram Analizi

N-gram analizinin çeşitli kullanım alanları vardır. Bu alanlara hesaplamalı dilbilim, olasılık ve hesaplamalı biyoloji gibi örnekler verilebilmektedir. N-gram veri üzerinde arama yapmak, karşılaştırmak veya tekrar sayısını belirlemek için kullanılan bir yöntemdir. N-gram, belirli bir dizinin n tane elemanlı bir alt dizisi olarak tanımlanır [15]. N-gram dağılımını hesaplamak için sabit boyutlu pencere veri seti üzerinde kaydırılır ve her bir değerin kaç kere tekrarlandığını hesaplamaktadır. Şekil 3.1'de 2-gram analizinin bir örneği gösterilmektedir. Bu örnekte her bir ikili sayı onaltılık sayı (hexadecimal) türünden bir bayta karşılık gelmektedir. Verilen dizi boyunca 2 boyutlu pencereler 1'er adım mesafe ile kaydırılarak baytların sıklık değerleri bulunur ve 2-gram sonuçları elde edilir [30].



Şekil 3. 1: 2-gram analizi örneği.

N gram analizinde kullanılan öğeler kelimeler, harfler veya baytlar olabilmektedir. Kullanılan bütün öğeler aynı alfabeye ait olmalıdır. Bu makalede kullanılan öğeler baytlardır ve 256 olası değeri vardır (0, 1, ..., 255). N-gram için, uzay 256^N dir (Unigram (n=1) veya bayt frekans dağılımı olarak adlandırılan durumda 256 olası değer vardır. Bigram veya 2- gram olarak adlandırılan ve n değerinin 2'ye eşit olduğu durumda 256^2 olası değer vardır). Bayt dizisi bilgileri tutulduğundan dolayı n sayısının

artması ile beraber önemli özniteliklerin sayısı da artmaktadır. Ancak hesaplama maliyeti de N'in artması ile artmaktadır. 1-gram'da sadece bayt sayısı önemliyken 2-gram'da baytların sayısı ve baytların sıra bilgileri de önem kazanmıştır.

3.4 Rastgele Orman Algoritması

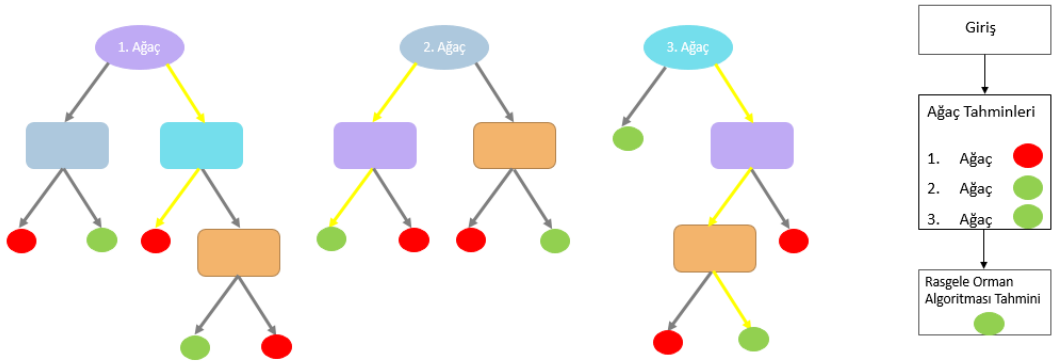
Toplu sınıflandırma yöntemlerinden biri olan rastgele orman (RO) algoritması Breiman tarafından geliştirilmiştir [40]. Rastgelelik özelliği eklenerek torbalama yönteminin geliştirilmiş bir versiyonu olarak kabul edilen bu yöntemde bir sınıflandırıcı yani bir karar ağacı yerine birden çok karar ağacı üretilmektedir. Karar ağaçlarında budama işlemi yapılmamaktadır. Ağaçlar maksimum boyuta üretilmektedir ve budamanın olmaması rastgele orman algoritmasını diğer karar ağacı algoritmalarından daha avantajlı hale getirmektedir.

Algoritma 1'de rastgele orman algoritması sözde kod biçiminde özetlenmektedir. Bu algoritmada S_N, M ve μ olmak üzere üç tane giriş parametresi bulunmaktadır. S_N veri setini temsil etmektedir ve her bir veri N tane örnek noktadan oluşmaktadır ($(x, y), x \in R^S$). Diğer iki parametre ise kullanıcılar tarafından belirlenmektedir. μ en iyi bölünmeyi belirlemek için her bir düğümde kullanılan değişkenlerin sayısını temsil ederken M ise geliştirilecek ağaçların sayısını temsil etmektedir.

Algoritma 1 Rastgele Orman Algoritması (S_N, M, μ)
Girdi: Veri seti S_N, M ağaç sayısı, μ alt uzay boyutu
1. for i = 1'den M kadar
2. $S_i \leftarrow S_N$'den μ elemanlı alt uzay oluştur
3. SRAT ile S_i alt uzayı kullanılarak sınıflandırıcı oluştur.
4. Geri Dönme: Tüm ağaç modellerini dön

Rastgele orman algoritmasında öncelikle gerçek veri setinden μ elemanlı yeni bir eğitim veri seti oluşturulmaktadır. Ardından rastgele özellik seçimi kullanılarak yeni eğitim setinden bir ağaç geliştirilmektedir. Ağaç üretmek için sınıflandırıcı ve regresyon ağacı tekniği (SRAT) kullanılmaktadır. SRAT metodolojisini kullanarak bir ağaç budanmadan maksimum boyutta üretilmektedir. Algoritma çıktısı olarak M tane ağaç modeli geri dönmektedir. Bu üretilen ağaç modellerinden elde edilen tahminler

kullanılarak, yeni veri oylama veya ortalama alma yöntemleriyle sınıflandırılmaktadır [41].



Şekil 3. 2: Rastgele orman algoritması karar mekanizması örneği.

Şekil 3.2'de rastgele orman algoritması için basit bir örnek yer almaktadır. M ağaç sayısı 3 olarak belirlenmiş ve SRAT yöntemi ile μ elemanlı veri seti kullanılarak yeni ağaçlar üretilmiştir. Şekilde yeni gelen bir örnek sarı oklarla gösterilen yolu takip ederek her bir ağaç için hangi sınıfın içine düştüğü bulunmuştur. İlk ağaç için kırmızı, ikinci ve üçüncü ağaç için ise yeşil sınıfının içine düştüğü belirlenmiştir. Bu üç sınıfın elde ettiği tahminlerin ortalamasının alınması ile yeni gelen örneğin yeşil sınıfının içine dahil edildiği görülmektedir. Yani genel olarak özetlemek gerekirse yeni gelen bir örneğin üç ağaç için de tahmin sonuçları elde edilmiştir ve bu tahmin sonuçlarının ortalaması alınarak hangi sınıfın içine dahil edildiğine karar verilmiştir.

3.5 Destek Vektör Makineleri

Gözetimli öğrenme algoritmalarından biri olan destek vektör makineleri istatistiksel öğrenme kuramına dayanmaktadır ve Vapnik [42] tarafından geliştirilmiştir [8]. İki sınıfın birbirinden en uygun şekilde ayrıştırılması prensibine dayanmaktadır. Sınıflar giriş uzayında doğrusal olarak ayrıştırılamazlarsa, veriler destek vektör makineleri tarafından yüksek boyutlu öznitelik uzayına taşınmaktadır. Bu verileri birbirlerinden ayırmak için hiper-düzlemler yani başka bir ifadeyle yüksek boyutlu uzayda sınıflar arasında maksimum sınırlar kullanılmaktadır. Veri seti çekirdek fonksiyonları olarak bilinen doğrusal olmayan fonksiyonlar yardımıyla yüksek boyutlu öznitelik uzayına taşınır.

N hacimli bir eğitim kümesi S_N, (x, y) ikililerinden oluşmaktadır. Burada x ϵ R^S olup N boyutlu uzayı, y ϵ {-1,1} ise sınıf etiketlerini göstermektedir. Destek vektör makineleri Denklem (3.1) -(3.3) arasındaki optimizasyon probleminin çözülmesi ile elde edilmektedir.

$$\min \frac{1}{2} w^T w + \sum_{i=1}^{N} \xi_i \tag{3,1}$$

$$y_i(w^T \phi(x_i) + b) \geq 1 \quad i = 1, \dots, N \tag{3,2}$$

$$\xi_i \geq 0 \tag{3.3}$$

b ve w hiper-düzlem parametreleri olup bu parametreler eğitim verileri yardımıyla bulunmaktadır. Burada x girdi vektörleri, ϕ fonksiyonu ile daha yüksek boyutlu bir öznitelik uzayına taşınmaktadır. $K(x_i x_j) = \phi(x_i)^T \phi(x_j)$ çekirdek fonksiyonunu olarak adlandırılmaktadır. Denklem (3.4) – (3.7) arasında literatürde yaygın olarak kullanılan çekirdek fonksiyonları yer almaktadır. ξ ve C sırasıyla hata ve ceza parametrelerini belirtmekte olup C parametresi kullanıcı tarafından belirlenmektedir.

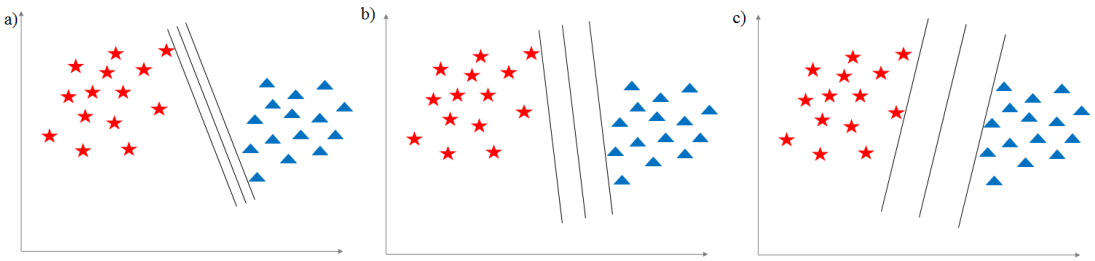
Lineer $$K(x_i x_j) = x_i^T x_j \tag{3.4}$$

Polinom $$K(x_i x_j) = (\gamma x_i^T x_j + r)^d, \ \gamma > 0 \tag{3.5}$$

Radyal Tabanlı Fonksiyon $$K(x_i x_j) = exp\left(-\gamma \left\| x_i - x_j \right\|^2\right), \gamma > 0 \tag{3.6}$$

Sigmoid $$K(x_i x_j) = tanh(\gamma x_i^T x_j + r) \tag{3.7}$$

Şekil 3.3'te iki sınıflı bir destek vektör makineleri örneği yer almaktadır. Bu şekilde iki sınıfı birbirinden ayıran üç farklı hiper-düzlem vardır. Hiper-düzlemlere en yakın elemanlar destek noktaları olarak tanımlanır. Destek vektör makinelerindeki amaç hiper-düzlem ve destek noktalar arasındaki mesafenin maksimum olmasıdır. Destek noktalarının hiper-düzleme maksimum uzaklıkta olması yeni gelen bir verinin doğru sınıflandırılma olasılığını artırmaktadır.

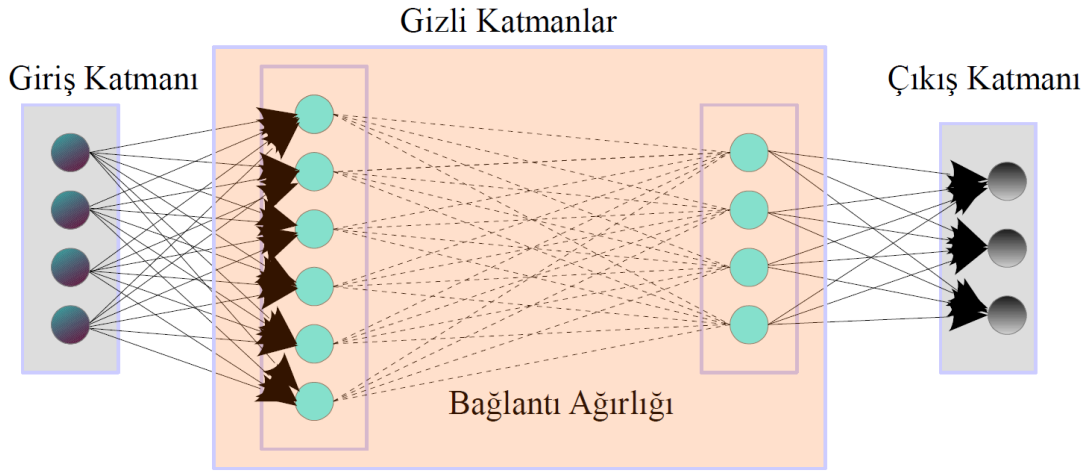


Şekil 3. 3: Destek vektör makineleri.

3.6 Derin Sinir Ağlar

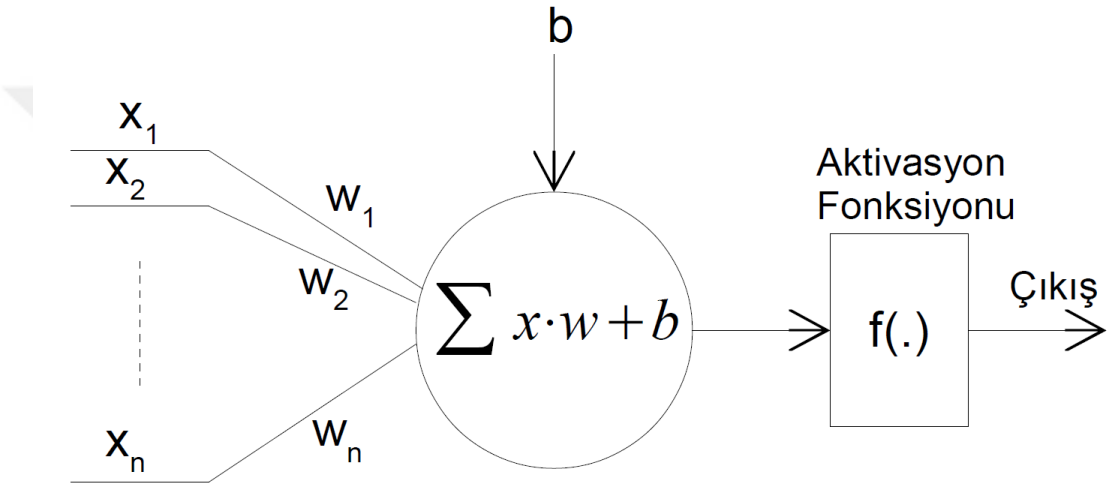
İnsan beyninin yaptığı gibi giriş sinyalleri derin sinir ağlar (DSA) tarafından işlenmektedir. DSA'lar nöron olarak bilinen çok sayıda homojen hesaplama elemanından oluşmaktadır. Bu nöronlar paralel olarak çalışmakta olup çoklu girişler ve tek bir çıkış içermektedir.

DSA'lar tipik olarak katmanlı bir nöron yapısı kullanmaktadır. l. katman nöronlarının çıkışı, l+1. katmandaki tüm nöronlara bağlanmaktadır. Hiyerarşik öğrenmeyi sağlamak için DSA'lar çoklu katman yapısını kullanmaktadır. Şekil 3. 4'te birinci katmanı giriş katmanı, son katmanı çıkış katmanı ve geri kalan katmanları gizli katmanlar olarak adlandırılan dört katmana sahip derin sinir ağ mimarisi yer almaktadır. DSA'lar ileri beslemeli ağ mimarisini kullanmakta olup bu mimaride geri besleme bağlantısı yoktur. Giriş sinyalleri ilk önce giriş katmanı sonra gizli katmandan ve en son olarak da çıkış katmanına doğru hareket etmektedir.



Şekil 3. 4: Giriş katmanı, çıkış katmanı ve iki gizli katmandan oluşan genel bir derin ağ mimarisi.

Bir nöron bir bayas b ile bir dizi bağlantı ağırlıkları w = (w_1, w_2, ..., w_n), ve aktivasyon fonksiyonundan f oluşmaktadır. Şekil 3.5'te DSA'larda kullanılan nöron olarak bilinen işlem birimi görülmektedir. Dıştan gelen giriş vektörü x= (x_1, x_2, ..., x_N) ile bağlantı ağırlıkları arasında skaler çarpım gerçekleştikten sonra bayas eklenmektedir. Girişin ağırlıklı toplamı hesaplandıktan sonra Denklem 3.4'te verilen çıktıyı üreten f (.) fonksiyonu uygulanır. Bu aktivasyon fonksiyonları nöronların davranışlarını belirlemektedir. Literatürde birçok aktivasyon fonksiyonu bulunmaktadır. Bu fonksiyonlardan DSA'larda en popüler olanları tanh (f (x) = (2 / (1 + e^{-2x}))- 1) ve sigmoid (f (x) = 1 / (1 + e^{-x})) fonksiyonlarıdır.



Şekil 3. 5: Derin sinir ağlarında nöron olarak bilinen işlem birimi.

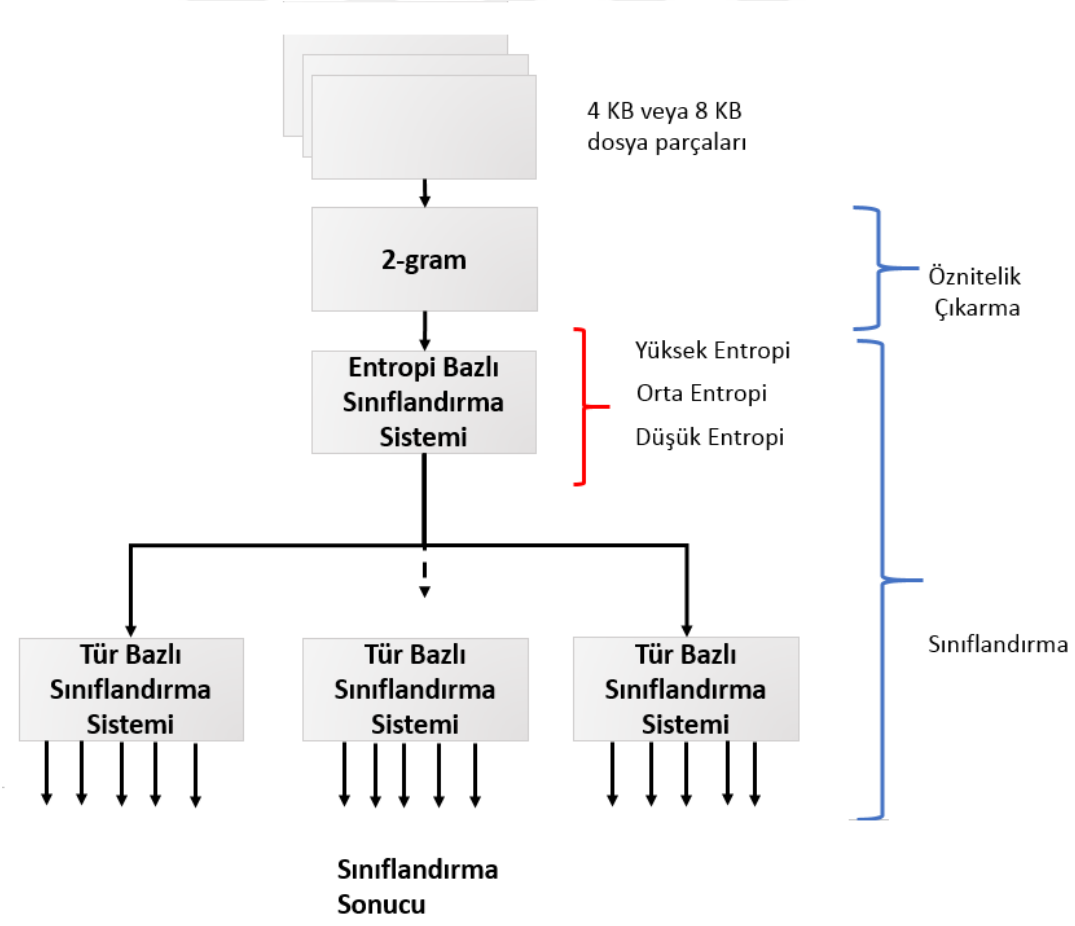
$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{3.4}$$

Öğrenme görevinde en önemli parametreler ağ bilgisini depolayan bağlantı ağırlıkları ve bayas değerleridir. Bağlantı ağırlıkları ve bayaslar stokastik gradyan iniş, adam veya adagrad gibi optimizasyon fonksiyonları ile öğrenilmektedir. Bu değişkenler karesel hata ve çapraz entropi gibi önceden tanımlanmış bir maliyet işlevini en aza indirgemek için kullanılmaktadır. Optimizasyon fonksiyonun çözümü için öncelikle parametreler rastgele değerler ile başlamaktadır, daha sonra bu parametreler global minimum seviyeye ulaşana kadar iteratif olarak değişmektedir. Her iterasyonda maliyet fonksiyonunun türevi hesaplanır ve parametrelerin değerleri güncellenmektedir [43, 44]. Daha fazla teknik detay için Yu ve Deng tarafından yazılan kitabın 4. ve 5. bölümlerine bakılabilir [44].

4. ÖNERİLEN YÖNTEM

Dosya ve veri türlerini sınıflandırmak için istatistiksel bilgilerin kullanıldığı matematiksel bir model önerilmiştir. Önerilen yöntemde uygulanan işlemler aşağıdaki gibi altı adımda özetlenebilmektedir.

1. Veri seti oluşturma
2. Veri setini eğitim veri seti ve test veri seti olmak üzere ikiye böl
3. Her bir veri setinden 4KB ve 8KB'lık dosya parçalarını rastgele seç
4. Her dosya parçasının 2-gram analizi ile özniteliklerini çıkar
5. Eğitim veri setindeki elemanları kullanarak bir sınıflandırma modeli oluştur
6. Sınıflandırıcıyı test etmek için test veri setindeki elemanları kulan

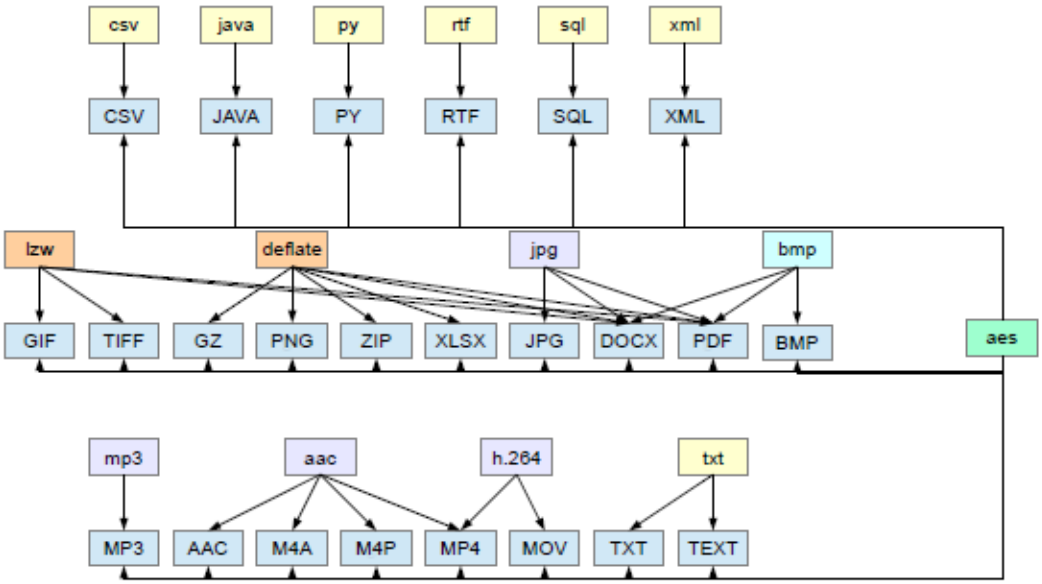


Şekil 4. 1: Önerilen yöntemin akış şeması.

Bu yöntemin akış şeması Şekil 4.1'deki gibidir. Sınıflandırma işlemi için iki seviyeli hiyerarşik bir model kullanılacak olup ilk hiyerarşide Entropi bazlı deneysel bir yaklaşım benimsenmektedir. Entropi bazlı sınıflandırma sonucunda 2 sınıflı veya 3 sınıflı ayrışma olacağı için sınıflandırma sisteminin ikinci hiyerarşisindeki ayrışmanın nasıl olacağı yapılan testler sonucunda netlik kazanmaktadır Yapılan deneyler sonucunda kazanan model ile ayrışma sağlandıktan sonra tür bazlı sınıflandırma yapılmaktadır. Sınıflandırma doğruluğunu değerlendirmek için karşılaştırmalı değerlendirme yapılmaktadır

4.1 Dosya ve Veri Türü

Bu çalışma için bilgisayarlarımız, telefonlarımız ve fotoğraf makinelerinde en çok kullanılan dosya türleri belirlenmiştir. Bu dosya türleri AAC, BMP, CSV, DOCX, GIF, GZ, JAVA, JPG, M4A, M4P, MP3, MP4, MOV, PDF, PNG, PY, RTF, SQL, TEXT, TIFF, TXT, XLSX, XML ve ZIP dir.



Şekil 4. 2: Kullanılan dosya ve veri türleriyle bu veri türlerinin dosya türleri ile ilişkisi.

Bu belirlenen dosya türlerinin bazılarında birden fazla veri türü olduğu için bu dosya türleri veri türlerine dönüştürülmüştür. Dosya türü ve veri türü tanımı içerik tabanlı yöntemlerde parçalı dosya türlerinin anlatıldığı bölümde ayrıntılı bir şekilde anlatıldığı için bu bölümde dosya türü ve veri türü bir örnek üzerinden anlatılacaktır. Örneğin DOCX dosya türünün içinde metin, resim, tablo gibi farklı veri türleri

barındırabilmektedir ve ayrıca deflate sıkıştırma algoritması ile sıkıştırıldığından bu dosya türü birden fazla veri sınıfının içine girmektedir. Birden fazla veri türü barındıran dosya türlerinin kullanılması sınıflandırma aşamasında karışıklığa neden olacağı için dosya türleri veri türlerine dönüştürülmüştür. Şekil 4.2'de seçilen dosya türleri ile veri türleri arasındaki ilişki gösterilmektedir. Dosya türleri veri türlerine dönüştürüldükten sonra 15 veri türü ortaya çıkmıştır. Bu veri türleri aac, aes, bmp, csv, deflate, h.264, java, jpeg, lzw, mp3, py, rtf, sql, txt ve xml dir.

Çizelge 4. 1: Conti yaklaşımına göre gruplandırılmış dosya türleri.

METİN Comma Separated Values (.csv) Java Source Code (.java) Python Script (.py) Rich Text Language (.rtf) Structured Query Language (.sql) Plain Text (.txt) Extensible Markup Lanquage (.xml)
İKİLİ Bitmap Image (.bmp)
RASTGELE Encrypted (AES256)
SIKIŞTIRMA – KAYIPLI Advanced Audio Coding (.aac) JPEG Image(.jpeg) MP3 Audio File(.mp3) H.264 (.mp4)
SIKIŞTIRMA – KAYIPSIZ Lempel-Ziv-Welch (.gif) Deflate (.png, .zip, .gz)

Conti'nin yaklaşımına göre dosya ve veri türleri, yüksek entropi, orta entropi ve düşük entropi olmak üzere üç gruba ayrılmaktadır. Bu gruplar Çizelge 4.1'de gösterilmiştir. Sıkıştırılmış, şifrelenmiş ve rastgele veriler yüksek entropi grubu içindedir. Makine kodu, programlama dili, işaret dili ve veri yapıları orta entropi grubu içerisine girmektedir. Bitmap de düşük entropi grubuna içerisindedir.

4.2 Veri Toplama ve Hazırlama

Günümüzde kişisel bilgisayarlarımızda ve android ve ios işletim sistemlerine sahip telefonlarımızda en çok kullanılan 24 dosya türü seçilmiş ve bu dosya türleri 15 veri türüne çevrilmiştir. Bu 15 veri türünden çok farklı kaynaklardan veri toplamak için Google hacking yöntemi ile internet ortamından veriler toplanmıştır. İstenilen formatta verileri içeren siteler bulunduktan sonra bu verileri toplamak için ECLIPSE entegre geliştirme ortamında JAVA programlama dili ve JSOUP kütüphanesi kullanılarak veriler toplanmıştır.

Her veri türü için iki bin örnek toplanmaktadır. Veri kümesinin yarısı eğitim için kullanılmaktadır (%80 eğitim, %20 doğrulama) ve kalan yarısı da test için kullanılmaktadır.

İşletim sistemlerince veri kaydetmek için kullanılan en küçük küme boyutu 4 KB ve 8 KB olduğundan, parça boyutu olarak 4 KB ve 8 KB seçilmiştir. Bu boyut belirlendikten sonra verilerden rastgele 4 ve 8 KB uzunluğunda dosya parçaları seçilmiştir. Her dosya sadece bir kere kullanılmıştır ve toplamda atmış bin dosya parçası elde edilmiştir. Bu dosya parçalarının yarısı 4 kilobayt, diğer yarısı ise 8 kilobayttır.

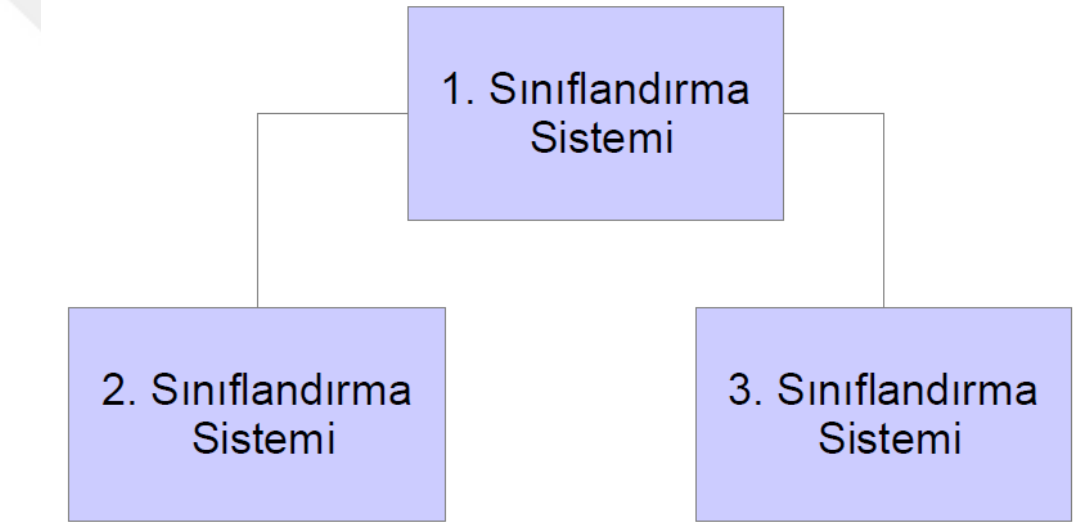
4.3 Öznitelik Çıkarma

Dosya ve veri türü sınıflandırması alanında öznitelik çıkarmak için birbirlerinden farklı öznitelik çıkarma yöntemleri kullanılmıştır. Bu yöntemlerden en yaygın olanı bayt frekans dağılımıdır. Bayt frekans dağılımı ile 2-gram analizi arasında karşılaştırma yapan araştırmacılar 2-gram analizinin daha etkili bir yöntem olduğunu vurgulamışlardır. 2-gram analizi ile sadece baytların sıklığı değil aynı zamanda baytlarında sırası önem kazanmaktadır. 2-gram analizi ile daha özgül öznitelikler çıkarılacağından dolayı bizde öznitelik çıkarmak için bu analizini seçtik. 4 KB ve 8

KB dosya ve veri parçaları seçildikten sonra bu seçilen yöntem ile veri parçalarının öznitelikleri çıkartılmıştır.

4.4 Sınıflandırma

Dosya ve veri sınıflandırması için makine öğrenmesi ve istatiksel bilgiye dayalı yöntemler yaygın bir şekilde kullanılmıştır. Bu çalışmada sınıflandırma için hiyerarşik bir yapı kullanılmıştır. Şekil 4.3'te sınıflandırma sisteminin yapısı yer almaktadır. İlk sınıflandırma sistemi için üç farklı algoritma dört farklı durum için test edilmiştir. Bu durumlar Çizelge 4.2'de yer almaktadır ve bu durumlar iki veya üç sınıftan oluşmaktadır. Bu sınıflar entropi bazlı durumların kombinasyonundan oluşmaktadır.



Şekil 4. 3: Sınıflandırma sisteminin mimarisi.

Entropi bazlı sınıflandırma için üç farklı algoritma seçilmiştir. Bu algoritmalar rastgele orman algoritması, destek vektör makineleri ve derin sinir ağlarıdır. Destek vektör makinelerinin seçilmesinin temel sebebi dosya ve veri sınıflandırmasında kullanılan en yaygın algoritma olmasıdır. Lineer çekirdek fonksiyonunun diğer çekirdek fonksiyonlarından daha etkili olduğu belirtildiği için sadece lineer çekirdek fonksiyonu ile test edilmiştir. Rastgele orman algoritması ve derin sinir ağlarda sınıflandırma alanında yaygın kullanılan diğer algoritmalar olduğu için bu iki algoritmada ilk sınıflandırma sisteminde test edilmiştir. İlk hiyerarşide bu üç sınıflandırma algoritması Çizelge 4.2'de yer alan 4 farklı durum için test edilmektedir.

Çizelge 4. 2: Entropi bazlı durumlar.

1. durum	1- Yüksek Entropi 2- Orta Entropi 3- Düşük Entropi
2. durum	1- Yüksek ve Orta Entropi 2- Düşük Entropi
3. durum	1- Yüksek ve Düşük Entropi 2- Orta Entropi
4. durum	1- Yüksek Entropi 2- Orta ve Düşük Entropi

Sınıflandırma alanında derin sinir ağları yaygın bir şekilde kullanılmaya başlanmıştır ve eğitim veri seti yeterince iyi genellendiğinde çok iyi sınıflandırma performansı elde edilmektedir. Bu çalışmanın amacı da derin sinir ağlarının dosya ve veri türü sınıflandırma alanında yapılan çalışmalarda da uygulanabileceğini göstermektir.

5. DENEY SONUÇLARI

Bu bölümde önerilen yöntem için elde edilen deney sonuçları sunulmaktadır. Bu önerilen yöntemde sınıflandırma sisteminde deneysel sonuçlar içermektedir. İlk hiyerarşide entropi bazlı sınıflandırma yapılmıştır. 4 farklı durum ve 3 farklı algoritma ile test edilmiştir. Entropi bazlı sınıflamanın sonucu optimum parametreleriyle Çizelge 5.1'de rapor edilmiştir. DVM'de lineer çekirdek fonksiyonu diğer çekirdek fonksiyonlarından daha verimli bulunmuştur, bu nedenle sadece lineer çekirdek fonksiyonu kullanılır ve C parametreleri yani ceza parametresi ızgara arama algoritması ile aranır. Rastgele orman algoritması için max_depth, criter, n_estimator ve random_state ızgara arama algoritması ile aranmaktadır. max_depth üretilecek ağacın en fazla ne kadar olabileceğinin limiti, n_estimetor üretilecek ağaç sayısını, ağacı üretmek için kullanılacak kriter criter parametreleri ile belirlenmektedir.

Entropi bazlı sınıflandırmada derin sinir ağları yeterince iyi sonuçlar vermediği için derin sinir ağı algoritması sonuçları bu tabloya dahil edilmemiştir. Derin sinir ağlarında optimum parametreleri bulmak için ızgara araması algoritması kullanılmıştır ve hesaplama maliyeti nedeniyle altı katmana kadar arama yapılmıştır. Arama sonucunda yeterli olacak ağ mimarisi bulunamamıştır ve ağ trendinin sadece bir tarafta olduğu görülmüştür. Entropi bazlı sınıflandırma problemini çözmek için derin sinir ağını kullanarak daha derin katmanlı yapıda ağa ihtiyacımız vardır. Bu sınıflandırma algoritmalarının sonuçlarını karşılaştırdığımızda rastgele orman algoritması ile en başarılı sonuçlar elde edilmiştir.

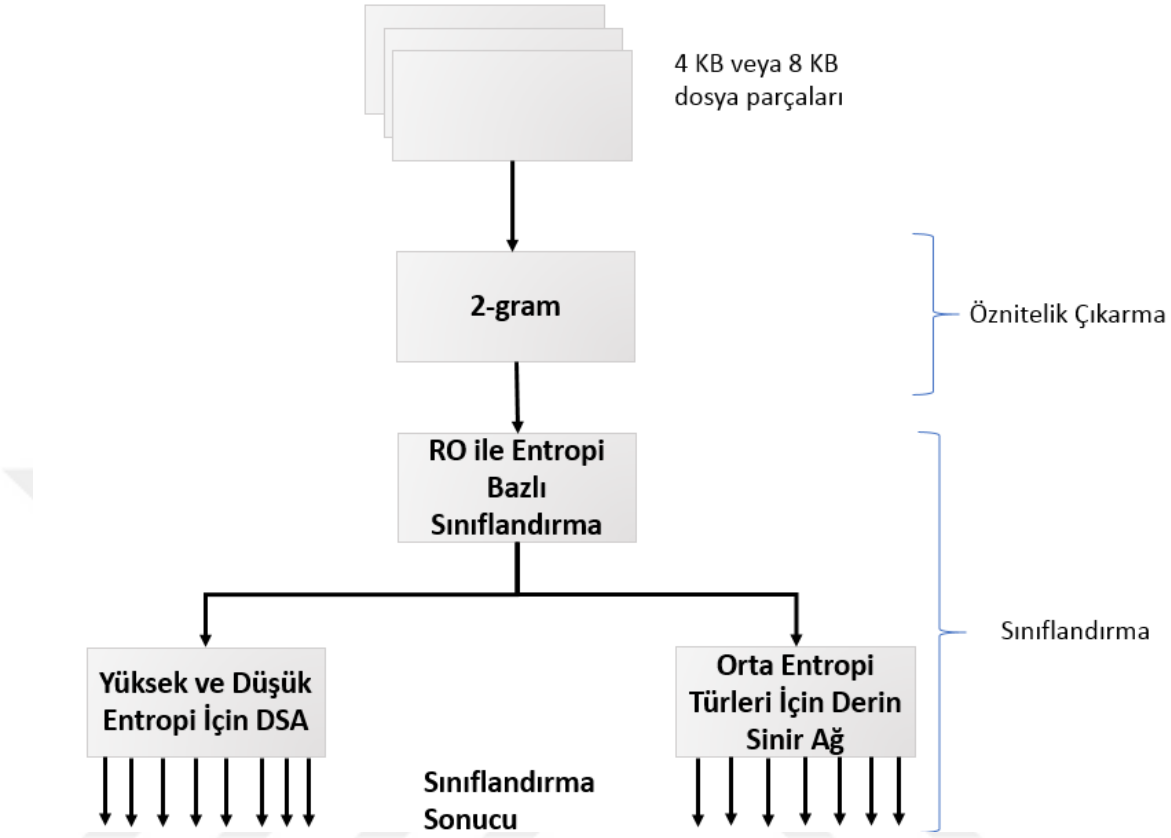
Rastgele orman algoritması ve ikili sınıflandırma durumu ile en başarılı sonuçlar elde edilmiştir. İkili sınıflandırmada bir tarafta yüksek ve düşük entropi, diğer tarafta ise orta entropi yer almaktadır. Şekil 5.1’de kazanan model ve kazana durum belli olduktan sonra önerilen akış şamasının son hali yer almaktadır.

Çizelge 5. 1: 4 KB dosya ve veri parçalarının entropi bazlı durumlar için rastgele orman algoritması ve destek vektör makinesi deneysel test sonuçları.

Yöntem	Sınıf Sayısı	Sınıf Bilgisi	Parametreler	Ortalama Başarı
RO	3 sınıf	1. durum	max_depth = 100 criter = entropy n_estimator = 100 random_state = 25	97.18
RO	2 sınıf	2. durum	max_depth = 100 criter = entropy n_estimator = 200 random_state = 2	97.67
RO	2 sınıf	3. durum	max_depth = 75 criter = gini n_estimator = 300 random_state = 10	99.52
RO	2 sınıf	4. durum	max_depth = 50 criter = entropy n_estimator = 25 random_state = 5	97.48
DVM	3 sınıf	1. durum	C = 2	98.72
DVM	2 sınıf	2. durum	C = 256	98.90
DVM	2 sınıf	3. durum	C = 256	98.74
DVM	2 sınıf	4. durum	C = 2	98.92

Kazanan model belirlendikten sonra, tür tabanlı sınıflandırma için derin sinir ağı kullanılmaktadır ve en iyi parametreleri bulmak için altı katmana kadar 5 kat çapraz korelasyon ve ızgarası araştırması ile bir ağ aranmaktadır. Çizelge 5.2'de orta entropili dosya ve veri türleri için 4 KB dosya parçaları kullanılarak elde edilen derin sinir ağı optimum parametreleri yer almaktadır. Giriş katmanı 65536 nöron içermektedir. Gizli katmanlar ızgara araması sonucunda sırasıyla 32, 64, 32, 64 olarak bulunmuştur ve bu katmanlar sırasıyla 32, 64, 32, 64 nöron içermektedir. Çıktı katmanı 7 dosya ve veri türü kullanıldığından 7 nöron içermektedir. Gizli katmanlar aktivasyon fonksiyonu RELU (Rectified Linear Unit-Doğrultulmuş Lineer Birim) ve çıktı katmanı aktivasyon

fonksiyonu ise softmax'tır. Optimizer fonksiyonu adam, devir sayısı ve küme büyüklükleri sırasıyla 30 ve 32'dir.



Şekil 5. 1: İlk hiyerarşide kazanan model belirlendikten sonra oluşan akış şeması.

Çizelge 5. 2: Orta entropi grubu içerisine giren dosya ve veri türleri için derin sinir ağlarının optimum parametreleri.

Katman Seviyesi	Nöron Sayısı	Aktivasyon
Giriş Katmanı	65536	-
1. Gizli Katman	32	Relu
2. Gizli Katman	64	Relu
3. Gizli Katman	32	Relu
4. Gizli Katman	64	Relu
Çıkış Katmanı	7	Softmax

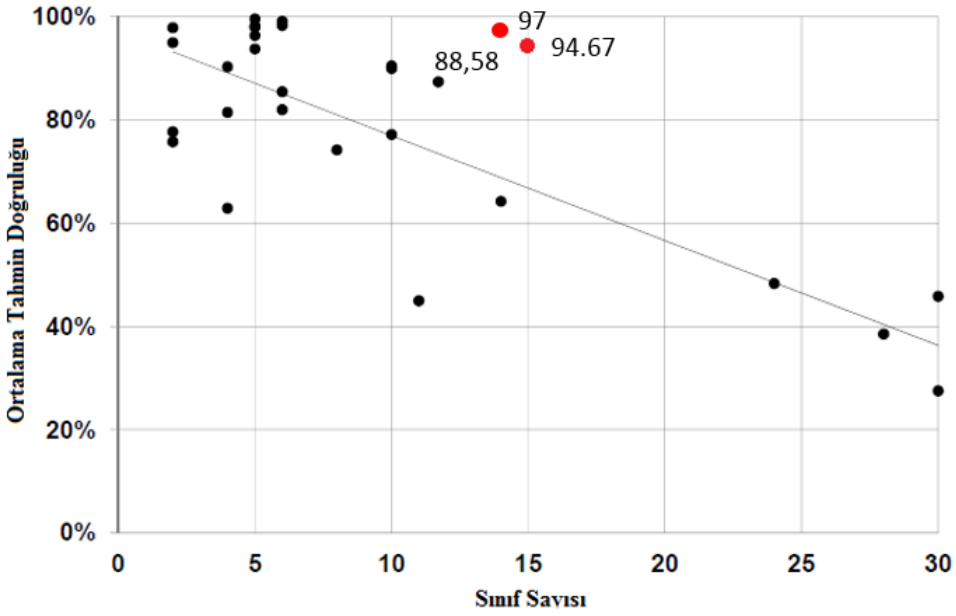
Aynı işlem adımları yüksek ve düşük entropili veri türleri için 4 KB dosya ve veri parçalarına uygulanmaktadır. Çizelge 5.3'te yüksek ve düşük entropili dosya ve veri parçaları kullanılarak eğitilen derin sinir ağının optimum parametreleri yer almaktadır. Giriş katmanı 65536 nöron içermektedir. Gizli katmanlar sırasıyla 32, 64, 128, 128

nöron olarak bulunmuştur ve aktivasyon fonksiyonu ELU'dur (Exponential Linear Unit- Üstel Lineer Birim). Optimizasyon fonksiyonu adamax olduğu çıktı katmanları için 8 düğüm kullanılmaktadır. Devir sayısı ve küme büyüklükleri sırasıyla 25 ve 32'dir.

En uygun parametreler belirlendikten sonra ağ 4 KB ve 8 KB dosya parçaları ile derin sinir ağları eğitilmiştir. Eğitilmiş ağlar kullanılarak test verilerinin hangi sınıfın içerisine girdiği tahmin edilmiştir. Çizelge 5.4 ve 5.5'te test verileri kullanılarak elde edilen karmaşıklık matrisi yer almaktadır. Tür seviyesinde doğruluk oranları 4 KB için %92,8 ve 8 KB için %94,67'dir.

Çizelge 5. 3: Yüksek ve düşük entropi grubu içerisine giren dosya ve veri türleri için derin sinir ağlarının optimum parametreleri.

Katman Seviyesi	Nöron Sayısı	Aktivasyon
Giriş Katmanı	65536	-
1. Gizli Katman	32	Elu
2. Gizli Katman	64	Elu
3. Gizli Katman	128	Elu
4. Gizli Katman	128	Elu
Çıkış Katmanı	8	Softmax



Resim 5. 1: İçerik tabanlı dosya türü sınıflandırması alanında yapılan çalışmaların ortalama tahmin doğrulukları ve bu çalışma ile elde edilen tahmin doğrulukları.

Resim 5.1 içerik tabanlı dosya türü sınıflandırması alanında yapılan tahmin doğruluk oranlarının olduğu grafiktir. Bu grafikte kırmızı ile gösterilen sonuçlar bu çalışma sonucunda elde edilen tahmin doğruluklarıdır. Bu çalışmalarda aes şifreleme veri türü olmadan yapılan ilk çalışmada ortalama %97 doğruluk oranı elde edilmiştir. Adli bilişim olaylarında şifrelenmiş veri türü önemli bir tür olduğu için yapılan çalışma sonradan genişletilmiştir. Bu dosya türü eklendikten sonra ortalama doğruluk oranları 4 KB için %92,8 ve 8 KB için %94,67'dir.

Çalışmamızda derin sinir ağ ve hiyerarşik bir model kullanılmıştır. En çok kullanılan metin, resim ve ses dosya türleri veri türlerine dönüştürülmüş ve sınıflandırma için kullanılmıştır. Çalışmamızı literatürdeki benzer sayıda dosya ve benzer dosya türü kullanılarak yapılan çalışma ile karşılaştırıldığında %6,87 oranında doğruluk oranını artırdığı görülmüştür. Literatürdeki en gelişmiş yöntemin kullandığı dosya türlerinin dışında bizim çalışmamızda şifrelenmiş veri türü de kullanılmıştır. Şifrelenmiş veri türü ile deflate veri türü birbiri ile karıştığı görülmüş ve doğruluk oranını ciddi şekilde düşürmüştür.

Çizelge 5. 4: Sonuçlar – 4 KB dosya ve veri parçaları kullanılarak elde edilen tür tabanlı sınıflandırma karışıklık matrisi.

KM	AAC	AES	BMP	CSV	DEF	H.264	JAVA	JPEG	LZW	MP3	PY	RTF	SQL	TXT	XML
AAC	975	8	2	0	5	1	0	9	0	0	0	0	0	0	0
AES	3	902	0	0	79	15	0	1	0	0	0	0	0	0	0
BMP	8	7	908	0	32	3	0	22	5	10	0	0	0	0	0
CSV	0	0	0	980	0	0	1	0	0	0	0	0	2	7	1
DEF	24	393	1	0	498	31	0	10	19	2	0	0	0	0	0
H.264	1	85	0	0	8	905	0	0	0	1	0	0	0	0	0
JAVA	0	0	0	1	0	0	986	0	0	0	0	0	13	0	0
JPEG	5	14	26	0	12	0	0	924	8	2	0	0	0	0	0
LZW	2	31	0	0	7	2	0	0	958	0	0	0	0	0	0
MP3	0	1	1	5	0	0	0	6	0	987	0	0	0	0	0
PY	0	0	0	2	0	0	0	0	0	0	989	0	4	5	0
RTF	0	0	0	2	0	0	4	0	0	0	0	978	2	4	0
SQL	0	0	0	0	0	0	3	0	0	0	2	0	992	0	1
TXT	0	0	0	11	0	0	0	0	0	0	9	2	2	958	10
XML	0	0	0	1	0	0	0	0	0	0	1	0	6	4	981

Çizelge 5. 5: Sonuçlar – 8 KB dosya ve veri parçaları kullanılarak elde edilen tür tabanlı sınıflandırma karışıklık matrisi.

KM	AAC	AES	BMP	CSV	DEF	H.264	JAVA	JPEG	LZW	MP3	PY	RTF	SQL	TXT	XML
AAC	983	0	8	0	5	0	0	3	0	1	0	0	0	0	0
AES	0	926	0	0	74	0	0	0	0	0	0	0	0	0	0
BMP	26	8	899	0	20	2	0	28	11	4	0	0	0	0	0
CSV	0	0	0	982	0	0	0	0	0	0	3	0	3	8	1
DEF	4	302	1	0	659	3	0	4	9	1	0	0	0	0	0
H.264	0	46	1	0	14	937	0	0	2	0	0	0	0	0	0
JAVA	0	0	0	0	0	0	998	0	0	0	0	0	2	0	0
JPEG	2	13	21	0	13	1	0	938	2	7	0	0	0	0	0
LZW	0	18	4	0	8	0	0	0	969	1	0	0	0	0	0
MP3	4	1	0	0	0	0	0	2	1	992	0	0	0	0	0
PY	0	0	0	2	0	0	0	0	0	0	990	0	2	6	0
RTF	0	0	0	1	0	0	0	0	0	0	1	981	6	2	1
SQL	0	0	0	0	0	0	2	0	0	0	3	0	992	0	1
TXT	0	0	0	7	0	0	0	0	0	0	13	5	0	968	4
XML	0	0	0	1	0	0	1	0	0	0	2	0	0	1	987

6. SONUÇ VE ÖNERİLER

Dosya ve veri türü sınıflandırması adli bilişim ve bilgi güvenliği için önemli bir problemdir. Adli bilişim çalışmaları incelendiğinde dosyalar genellikle parçalı bir şekilde saklanmaktadır. Parçalı bir şekilde saklanmış dosyalarda sadece ilk parçasında sihirli bayt bilgileri yer alır. Diğer parçalarda ise dosya türüne ait bilgiler olmamaktadır. Parçalı bir şekilde saklanmış dosyalar için dosya türünü belirlemek çok zorlu bir süreçtir. Bu alanda son yıllarda araştırılması hızlı bir şekilde artan içerik tabanlı yöntemlerde dosya içerik bilgileri analiz edilerek dosyanın türü belirlemektedir. Ayrıca içerik tabanlı yöntemler dosya uzantısı ve sihirli bayt bilgilerinin değişmesine karşı dirençli olduğu için güvenilir bir yöntemdir. Bu çalışmada da hiyerarşik model kullanılarak içerik tabanlı dosya ve veri sınıflandırması yöntemi sunulmuştur. Son yıllarda derin sinir ağlar sınıflandırma alanında yaygın bir şekilde kullanılmaktadır. Kullanıldığı çoğu alanda performans başarısını artırmaktadır. Bu çalışmada dosya ve veri türü sınıflandırmasına derin sinir ağlar perspektifinden bakılmış ve bu yönde bir çözüm önerisi sunulmuştur. Öncelikle 2-gram analizi ile dosya ve veri parçalarının öznitelikleri çıkarılmıştır. Hiyerarşik sınıflandırma sisteminde ilk seviyede entropi bazlı sınıflandırma işlemi rastgele orman algoritması ile yapılmıştır. İkinci seviyede ise tür bazlı sınıflandırma işlemi derin sinir ağlar kullanılarak yapılmıştır. 4 KB ve 8 KB'lık dosya parçaları için %92,8 ve %94,67 doğruluk oranları elde edilmiştir. Adli bilişim alanında önemli dosya türlerinden olan şifrelenmiş türde dosyalarda veri setine eklenmiştir. Şifrelenmiş verinin eklenmesi doğruluk oranını ciddi şekilde düşürmesine rağmen literatürdeki benzer dosya sayısı ve benzer dosya türleri kullanılarak yapılan en gelişmiş çalışma ile karşılaştırıldığında %6,87 oranında doğruluk oranını arttırdığımız görülmüştür. Sonuç olarak önerilen bu yöntem yaygın olarak kullanılan dosya ve veri türlerini sınıflandırma için başarılı bir şekilde kullanılabileceği gösterilmiştir.

Elde edilen sonuçlar bu sistemin dosya türlerini ayırmada kullanılabileceğini göstermektedir. Bu sistemi daha da geliştirmek için öznitelikler çıkartıldıktan sonra benzer başarım oranlarının elde edilebileceği öznitelikler seçilip ızgara araması

yöntemi ile daha derin katmanlarda arama yapılabilecek ve daha derin katmanlarda sistemin doğruluk oranının artacağı öngörülmektedir. Sistem daha da geliştirilerek sık kullanılan dosya türlerini tespit eden bir ürün haline getirilebilir. Uzantısı değiştirilmiş, sihirli baytları değiştirilmiş veya sürücülerden silinmiş ama bulunduğu yerde kalmaya devam eden dosyaların gerçek türlerini belirlemede adli bilişim uzmanlarının yararlanabileceği ve adli bilişim alanında delillerin toplanıp analiz edilmesine önemli bir katkı sağlayabilir.

.

KAYNAKLAR

[1] **Zheng, N., Wang, J., Wu, T., Xu, M.A.,** (2015). A fragment classification method depending on data type, *In Computer and Information Technology*, 1948-1953.

[2] Platter, https://www.pcmag.com/encyclopedia/term/49369/platter.

[3] Harddiskler, http://teknomerkez.net/index.php?git=1086.

[4] **Amirani, M.C., Toorani, M., Mihandoost, S.,** (2013). Feature-based type identification of file fragments, *Security and Communicatio Networks* 6, 115-128.

[5] **Amirani, M. C., Toorani, M., And Beheshti, A.,** (2008). A new approach to content-based file type detection, *In Computers and Communications, Symposium on IEEE*, 1103–1108.

[6] **Mcdaniel, M., And Heydari, M. H.,** (2003). Content based file type detection algorithms, *In System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on IEEE*.

[7] **Karampidis, K., And Papadourakis, G.,** (2017). File type identificationcomputational intelligence for digital forensics, *Journal of Digital Forensics, Security and Law 12*, 2, 6.

[8] **Beebe, N. L., Maddox, L. A., Liu, L., And Sun, M.,** (2013). Sceadan: using concatenated n-gram vectors for improved file and data type classification, *IEEE Transactions on Information Forensics and Security 8*, 1519–1530.

[9] **Ahmed, I., Lhee, K.-S., Shin, H., And Hong, M.,** (2010). Content-based filetype identification using cosine similarity and a divide-and-conquer approach, *IETE Technical Review 27*, 465–477.

[10] File signatures database, https://www.filesignatures.net/index.php?page=search.

[11] Hex editor, https://www.onlinehexeditor.com/.

[12] File signature database, https://www.garykessler.net/library/file_sigs.html.

[13] TrID, http://mark0.net/onlinetrid.html.

[14] **Ahmed, I., Lhee, K.-S., Shin, H., And Hong, M.,** (2009). On improving the accuracy and performance of content-based file type identification, *In Australasian Conference on Information Security and Privacy, Springer*, 44–59.

[15] **Cao, D., Luo, J., Yin, M., And Yang, H.,** (2010). Feature selection based filetype identification algorithm, *In Intelligent Computing and Intelligent Systems (ICIS), 2010 IEEE International Conference on*, vol. 3, 58–62.

[16] **Dunham, J. G., Sun, M.-T., And Tseng, J. C.,** (2005). Classifying file type of stream ciphers in depth using neural networks, *In Computer Systems and Applications, The 3rd ACS/IEEE International Conference on IEEE*, 97.

[17] **Li, W.-J., Wang, K., Stolfo, S. J., And Herzog, B.,** (2005). Fileprints: Identifying file types by n-gram analysis, *In Information Assurance Workshop, Proceedings from the Sixth Annual IEEE SMC*, 64–71.

[18] **Ahmed, I., Lhee, K.-S., Shin, H., And Hong, M.,** (2010). Fast file-type identification, *In Proceedings of the 2010 ACM Symposium on Applied Computing, ACM*, 1601–1602.

[19] **Ahmed, I., Lhee, K.-S., Shin, H.-J., And Hong, M.-P.,** (2011). Fast contentbased file type identification, *In IFIP International Conference on Digital Forensics, Springer*, 65–75.

[20] **Alamri, N. S., And Allen, W. H.,** (2015). A comparative study of file type identification techniques, *In SoutheastCon, IEEE*.

[21] **Axelsson, S.,** (2010). Using normalized compression distance for classifying file fragments, *In Availability, Reliability, and Security, 2010. ARES'10 International Conference on, IEEE*, 641–646.

[22] **Beebe, N., Liu, L., And Sun, M.,** (2016). Data type classification: Hierarchical class-to-type modeling, *In IFIP International Conference on Digital Forensics, Springer*, 325–343.

[23] **Calhoun, W. C., And Coles, D.,** (2008). Predicting the types of file fragments, *Digital investigation 5*, 14–20.

[24] **Conti, G., Bratus, S., Shubina, A., Sangster, B., Ragsdale, R., Supan, M., Lichtenberg, A., And Perez-Alemany, R.,** (2010). Automated mapping of large binary objects using primitive fragment type classification, *Digital investigation 7*, 3–12.

[25] **Erbacher, R. F., And Mulholland, J.,** (2007). Identification and localization of data types within large-scale file systems, *In Systematic Approaches to Digital Forensic Engineering, Second International Workshop on*, 55–70.

[26] **Fitzgerald, S., Mathews, G., Morris, C., And Zhulyn, O.,** (2012). Using nlp techniques for file fragment classification, *Digital Investigation 9*, 44–49.

[27] **Gopal, S., Yang, Y., Salomatin, K., And Carbonell, J.,** (2011). Statistical learning for file-type identification, *In Machine Learning and Applications and Workshops, 10th International Conference on*, vol. 1, 68–73.

[28] **Karresand, M., And Shahmehri, N.,** (2006). File type identification of data fragments by their binary structure, *In Proceedings of the IEEE Information Assurance Workshop*, 140–147.

[29] **Karresand, M., And Shahmehri, N.,** (2006). Oscar file type identification of binary data in disk clusters and ram pages, *In IFIP International Information Security Conference, Springer.*

[30] **Li, Q., Ong, A., Suganthan, P., And Thing, V.,** (2011). A novel support vector machine approach to high entropy data fragment classification, *In Proceedings of the South African Information Security Multi-Conf (SAISMC)*, pp. 236–247.

[31] **Moody, S. J., And Erbacher, R. F.,** (2008). Sádi-statistical analysis for data type identification, *In Systematic Approaches to Digital Forensic Engineering, 2008. SADFE'08. Third International Workshop on IEEE*, 41–54.

[32] **Roussev, V., And Garfinkel, S. L.,** (2009). File fragment classification-the case for specialized approaches, *In Systematic Approaches to Digital Forensic Engineering, Fourth International IEEE Workshop on IEEE*, pp. 3–14.

[33] **Veenman, C. J.,** (2007). Statistical disk cluster classification for file carving, *In Information Assurance and Security, Third International Symposium on IEEE*, 393–398.

[34] **Zhang, L., And White, G. B.,** (2007). An approach to detect executable content for anomaly based network intrusion detection, *In Parallel and Distributed Processing Symposium, IEEE International*, 1–8.

[35] **McGuffee, J. W., And Hanebutte, N.,** (2013). Google hacking as a general education tool, *Consortium for Computing Sciences in Colleges*, 81-85.

[36] **Lancor, L., And Workman, R.,** (2007). Using Google hacking to enhance defense strategies, *ACM SIGCSE Bulletin*, 491-495.

[37] **Billing, J., Danilchenko, Y., And Frank, C. E.,** (2008). Evaluation of Google hacking, *Proceedings of the 5th annual conference on Information security curriculum development*, 27-32.

[38] Jsoup, https://jsoup.org/.

[39] http://hcmaslov.d-real.sci-nnov.ru/public/mp3/Queen/.

[40] **Breiman, L.,** (2001). Random forests, *Machine learning 45*, 5–32.

[41] **Breiman, L.,** (2017). Classification and regression trees, *Routledge*.

[42] **Vapnik, V.,** (2013). The nature of statistical learning theory, *Springer science & business media*.

[43] **Kim, P.,** (2017). Matlab deep learning: With machine learning, neural networks and artificial intelligence, *Apress*.

[44] **Yu, D., And Deng, L.,** (2016). Authomatic Speech Recognition, *Springer*.

ÖZGEÇMİŞ

Ad-Soyad : Ayşe Sıddıka EROZAN

Uyruğu : T.C.

Doğum Tarihi ve Yeri : 10.12.1989 Sincan

E-posta : a.aydogdu@etu.edu.tr

ÖĞRENİM DURUMU:

- **Lisans** : 2013, İstanbul Teknik Üniversitesi, Elektrik Elektronik Fakültesi, Telekomünikasyon Mühendisliği
- **Yükseklisans** : 2018, TOBB Ekonomi ve Teknoloji Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği

MESLEKİ DENEYİM VE ÖDÜLLER:

Yıl	Yer	Görev
2018-…	KIT	Öğretim Asistanı
2014-2017	HAVELSAN	Araştırma ve Geliştirme Mühendisi
2013	PIWORKS	Müşteri Destek Mühendisi

YABANCI DİL: İngilizce

TEZDEN TÜRETİLEN YAYINLAR, SUNUMLAR VE PATENTLER:

- **Aydoğdu Erozan, A. S**., *2018* File Fragment Type Detection By Neural Network, *IEEE Signal Processing and Communications Applications Conference*, May 2-5, İzmir, Turkey.

DİĞER YAYINLAR, SUNUMLAR VE PATENTLER:

- **Aydoğdu, A. S**., Hatipoğlu, P. U., Özparlak, L. and Yüksel, S. E., *2015* LWIR and MWIR Images Dimension Reduction and Anomaly Detection with Locally Linear Embedding, *IEEE Signal Processing and Communications Applications Conference*, May 16-19, Malatya, Turkey.

- Erozan, A. T., **Aydoğdu, A. S**., Ors, B., *2015* Application specific processor design for DCT based aaplications. *IEEE Signal Processing and Communications Applications Conference,* May 16-19, Malatya, Turkey