T.C.
BURSA ULUDAĞ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DERİN ÖĞRENME TABANLI ADLİ ANALİZ UYGULAMALARI

Ahmet Gökhan POYRAZ

Doç. Dr. Ahmet Emir DİRİK

(Danışman)

YÜKSEK LİSANS TEZİ
ELEKTRONİK MÜHENDİSLİĞİ

BURSA - 2019

TEZ ONAYI

Ahmet Gökhan POYRAZ tarafından hazırlanan "Derin Öğrenme Tabanlı Adli Analiz Uygulamaları" adlı tez çalışması aşağıdaki jüri tarafından oy birliği ile Bursa Uludağ Üniversitesi Fen Bilimleri Enstitüsü Elektronik Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Danışman : Doç. Dr. Ahmet Emir DİRİK

Başkan : Doç. Dr. Ahmet Emir DİRİK
Bursa Uludağ Üniversitesi
Mühendislik Fakültesi
Bilgisayar Mühendisliği Bölümü

Üye: Doç. Dr. Ersen Yılmaz
Bursa Uludağ Üniversitesi
Mühendislik Fakültesi
Elektrik-Elektronik Mühendisliği Bölümü

Üye: Doç. Dr. Cemal HANİLÇİ
Bursa Teknik Üniversitesi
Mühendislik ve Doğa Bilimleri Fakültesi
Elektrik-Elektronik Mühendisliği Bölümü

Yukarıdaki sonucu onaylarım.

Prof. Dr. Hüseyin Aksel EREN
Enstitü Müdürü
05/7/2019 (Tarih)

B.U.Ü. Fen Bilimleri Enstitüsü, tez yazım kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- tez içindeki bütün bilgi ve belgeleri akademik kurallar çerçevesinde elde ettiğimi,
- görsel, işitsel ve yazılı tüm bilgi ve sonuçları bilimsel ahlak kurallarına uygun olarak sunduğumu,
- başkalarının eserlerinden yararlanılması durumunda ilgili eserlere bilimsel normlara uygun olarak atıfta bulunduğumu,
- atıfta bulunduğum eserlerin tümünü kaynak olarak gösterdiğimi,
- kullanılan verilerde herhangi bir tahrifat yapmadığımı,
- ve bu tezin herhangi bir bölümünü bu üniversite veya başka bir üniversitede başka bir tez çalışması olarak sunmadığımı

beyan ederim.

01/02/2019
Ahmet Gökhan POYRAZ
İmza

ÖZET

Yüksek Lisans Tezi

DERİN ÖĞRENME TABANLI
ADLİ ANALİZ UYGULAMALARI

Ahmet Gökhan POYRAZ

Bursa Uludağ Üniversitesi
Fen Bilimleri Enstitüsü
Elektronik Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Ahmet Emir DİRİK

Günümüzde sayısal resimler üzerinde oynama yapabilmek oldukça kolay bir işlem haline gelmiştir. Bu oynama işlemleri genellikle kötü bir niyet taşımaksızın yapılmaktadır. Fakat bazı durumlarda bir resmin oynanıp oynanmadığı oldukça önem arz etmektedir. Özellikle siyasi kişilere karşı manipülasyon yapabilmek için sahte resimler oluşturulmaktadır. Bu bağlamda resimlerin güvenilirliği günümüzde adli kanıt olması açısından oldukça önem arz etmektedir. Mevcut adli analiz yöntemleri bazı durumlarda iyi sonuçlar üretebilmektedir. Ancak çoğu oynama çeşidinde mevcut yöntemler yetersiz kalmaktadır. Literatürdeki PRNU tabanlı kaynak cihaz tanıma yöntemi, adli bilişim alanında çalışanlar tarafından kabul görmüş ve eşdeğer yöntemler arasındaki en iyi yöntem olarak kabul edilir. Ayrıca son zamanlarda bu alana farklı bir açıdan çözüm getiren derin öğrenme tabanlı kamera model sınıflandırıcısı yöntemi de adli bilişim alanında başarısını kanıtlamaktadır. Bu çalışmada PRNU tabanlı yöntem ile derin öğrenme tabanlı yöntem irdelenmiş ve özel bir yaklaşımla birleştirilerek yeni bir yöntem önerilmiştir. Bu yöntem ile sayısal resimler üzerindeki oynanan bölgeler, eşdeğer yöntemlere göre daha doğru bir şekilde tespit edilebilmektedir. Hatta 100×100 piksel boyutundaki müdahalelerde dahi iyi derecede çalışmaktadır.

Anahtar Kelimeler: PRNU, CNN, Oynama Tespiti, Derin Öğrenme, Adli Bilişim
2019, x + 82 sayfa

ABSTRACT

M.Sc. Thesis

DEEP LEARNING BASED
FORENSIC APPLICATIONS

Ahmet Gökhan POYRAZ

Bursa Uludağ University
Graduate School of Natural and Applied Sciences
Department of Electronic Engineering

Supervisor: Assoc. Prof. Dr. Ahmet Emir DİRİK

Nowadays it has become very easy altering the contents of digital images. These alterations are usually carried out without a bad intention. But in some cases, it is very important to know that a picture is altered or not. Particularly, fake images are created to manipulate political figures. In this context, the trustworthiness of the images is very important in terms of forensic evidence. Current forensic detection methods can produce good results in some cases. However, there are insufficient methods available against most types of alterations. The PRNU-based source device identification method in the forensic detection literature is the most accepted method among similar methods by forensic analysts. In addition, deep learning-based camera model classifier method, which has recently been offered as a solution in this area, proved its success in forensic field. In this study, the deep learning based forensic detection method and the PRNU-based method is examined and a new method based on a special fusion approach is proposed. With this method, the tampered regions on digital images can be detected more accurately than the methods in the literature, even so, the proposed method works well in detecting small-scale forgeries with the size of 100 x 100 pixels.

Keywords: PRNU, CNN, Tamper Detection, Deep Learning, Image Forensics
2019, x + 82 pages

TEŞEKKÜR

Bu tezi yapmamda ve yazmamda bana her daim desteği olan ve beni teşvik eden öncelikle anneme ve babama, sağlamış olduğu imkanlardan, yapmış olduğu desteklerden ve vermiş olduğu eğitimlerden ötürü danışmanım Doç. Dr. Ahmet Emir DİRİK'e ve yardımları için Ahmet KARAKÜÇÜK'e teşekkürlerimi bir borç bilirim.

Ahmet Gökhan POYRAZ
.../.../2019

İÇİNDEKİLER

SİMGELER VE KISALTMALAR DİZİNİ

Semboller	**Açıklama**
λ	Öğrenme katsayısı
ϕ	ESA tarafından üretilen olasılık değeri
θ	Sunulan yöntemdeki YSA tarafından üretilen olasılık değeri
ρ	PRNU yöntemi tarafından üretilen korelasyon değeri

Kısaltmalar	**Açıklama**
YSA	Yapay Sinir Ağı
ESA	Evrişimsel Sinir Ağı
ROC	Receiver Operating Characteristic
MRF	Markov Random Field
CRF	Conditioanl Random Field
TP	True Positive
FP	False Positive
MLE	Maximum Likelihood Estimation
AUC	Area Under the Curve

Çeviriler	**Açıklama**
Batch	Yığın
Dropout	Seyreltme
Node	Düğüm
Wavelet	Dalgacık
Threshold	Eşikleme
Digital	Sayısal
Activation	Aktivasyon
Regression	Regresyon
Exponential	Üstel
Gradient Descent	Gradyan İnişi
Label	Etiket
Linear	Doğrusal
Stochastic	Stokastik
Graph	Graf

Momentum	Ağırlık düşürme katsayısı
Convolution	Evrişim
Pooling	Örnekleme
Fully-Connected	Bütün bağlı
Tamper	Oynama
Learning Rate	Öğrenme Katsayısı
Zero-Padding	Sıfır Ekleme
Architecture	Mimari
Stride	Kaydırma
Early Stopping	Erken Durdurma
Backpropagation	Geriye Yayılım
Epoch	Tur
Majority	Çoğunluk
Multiple	Çoklu
Noise	Gürültü
Maksimum Likelihood	En büyük olabilirlik kestirimi

ŞEKİLLER DİZİNİ

Sayfa

ÇİZELGELER DİZİNİ

Sayfa

1 GİRİŞ

Günümüzde sosyal medya üzerinde milyonlarca sayısal görüntü ve video paylaşılmaktadır. Haber sitelerindeki sayısal resimler bize çokça bilgi vermektedir. Ancak gelişen yazılımlar ile sayısal resimler üzerinde değişiklik yapmak oldukça kolay bir işlem haline gelmiştir. Bu durum, paylaşılan görüntülerin güvenilirliğini tartışmaya açmıştır. Bazı durumlarda resimlerin gerçek olup olmadığı ya da resim içerisinde bir bölgenin oynanıp oynanmadığı bilgi doğruluğu açısından önem arz etmektedir. Haber sitelerinde bize yüzlerce haber sunulmaktadır. Bu haberler içerisinde, bir futbol transfer haberini ele alacak olursak, dünyaca ünlü bir futbolcunun Türkiye'deki bir takıma transfer olduğunun haberinin yapıldığını varsayalım. Haberdeki resimde takımın forması, transfer edilmiş gibi gösterilen futbolcunun üzerine oynama yapılarak giydirilebilmektedir. Balon haber yapmak aslında bu kadar kolaydır. Az önceki basit bir örnekti. Ancak bazı durumlarda, örneğin sayısal ortamda evrakta sahtecilik yapıldığı zaman bu durumun tespit edilmesi adli bir vaka haline gelir. Bu adli vakalar ise adli bilişim alanının konusuna dahil olmaktadır.

Konunun öneminin daha iyi anlaşılması açısından şu şekilde bir kurgusal örnek verebiliriz. Dünyadaki bir terör örgütü lideri hakkında istihbari bir bilgiyi ele alalım. Diyelim ki istihbarat birimlerinin eline güvenilir bir kaynaktan sayısal resimler geçti. Resimler içerisinde bu terör örgütü liderinin Konya'da olduğu görülüyor. Eğer bu resimler doğruysa devlet yetkilileri, polis ve askeri birliklerini bu bölgeye kaydırması gerekir. Ancak resimlerin gerçek olmadığı ispatlanabilirse ya da resim üzerindeki şahsın sadece yüz bölgesinin oynanmış olduğu kanıtlanırsa gereksiz yere askeri operasyonlar yapılmayacaktır. Ya da bir ülkenin başbakanının uygunsuz bir yerdeki fotoğrafının gerçek olup olmadığı, ülkenin yönetimi açısından oldukça önemlidir.

Adli bilişim alanı sayısal ortamdaki kanıtların analiz edilerek belirli bilgilere ulaşmayı hedeflemesinden ötürü oldukça geniş uygulama alanlarına sahiptir. Bu tez kapsamında sayısal görüntüler kullanılarak kamera model sınıflandırma, kaynak cihaz tanıma problemleri ve sayısal resimler üzerinde yapılan müdahale işleminin tespit edilmesi anlatılacaktır.

Her bir problemin kendi içerisinde önemli bir uygulama alanı bulunmaktadır. Örneğin kaynak cihaz tanıma problemini ele alalım. Olayın önemini belirtebilmek adına ve uygulama alanının daha iyi anlaşılması açısından tekrardan şu şekilde kurgusal bir örnek verilebilir. İşlenen bir cinayet olayını inceleyelim. Diyelim ki bir odada 3 kişi var bu 3 kişiden biri odada bulunan diğer şahıslardan birini öldürüyor. Şahıslardan biri ise olayı cep telefonuyla görüntüsünü alıyor. Görüntülerin internet ortamında paylaşıldığını varsayalım. Akşam haberlerinde bu tarz haberlere denk gelindiğinden mantıklı bir varsayımdır. Görüntülere bakılarak ölen ve öldürülen kişi belirlenebilir. Ancak cinayete yardım eden ya da ses çıkarmayan kişi çekimi yaptığı için şahsı kesin olarak tespit etmek mümkün değildir. Tam olarak bu durumda eğer polis kuvvetleri olası şüphelileri belirlerse, şüphelilerin cep telefonu kameralarıyla, çekilen görüntüler eşleştirilebilmektedir. Bu örnekte kaynak cihazı tespit ederek suçlu olan 3. şahsı tespit etmek mümkündür. Bu sadece basit bir örnekti. Ancak kaynağı tespit etmenin önemini vurgulamaktadır.

Sayısal görüntüler üzerinde yapılan bölgesel oynamaları tespit edebilmek, az önce verilen örnektekinden kat ve kat daha zor bir problemdir. Kötü niyetli kimseler sayısal resmin sadece belirli bir bölgesinde oynama yaparak sahte evrak oluşturabilir, yasadışı kanıtlar üretebilir, ya da masum bir kimseyi pornografik bir görüntü üzerine yerleştirebilirler.

Bu çalışmada öncelikle bölgesel oynama tespiti için kullanılan 2 ana yöntem detaylıca incelenmiştir. İlk ana yöntem olan PRNU tabanlı kamera kaynak cihaz tanıma yöntemi, kameranın değişmeyen sensör özelliklerini kullanarak kameraya ait parmak izini tahmin edilebilmektedir. Elde edilen parmak izi ile test edilen resim eşleştirildiğinde, uyuşmayan bölgelerin oynanmış bölge olduğu yorumu yapılabilmektedir. İkinci ana yöntem olarak, adli bilişim alanında çokça kullanılmaya başlanılan derin öğrenme tabanlı kamera model sınıflandırıcısı yöntemi detaylıca incelenmiştir. Bu yöntemde kamera modeline ait mozaiklendirme filtresi, vb. kamera içi işlemlere ait hassas filtreler öğrenilmekte, bu sayede kamera modelleri arasında bir sınıflandırma yapılabilmektedir. Bu yöntem ile çalışma boyutu olarak 96×96 piksel gibi küçük bloklar seçildiği zaman resim üzerindeki oynanmış bölgenin tespiti de mümkündür. Literatürdeki bu iki ana yöntem oynanmış bölgeleri tespit edebilme açısından hata oranı yüksek sonuçlar üretmektedirler. Bu duruma daha

iyi bir çözüm sunabilmek için bu tez çalışmasında, PRNU tabanlı kamera kaynak cihaz tanıma yöntemi ile derin öğrenme tabanlı kamera model sınıflandırıcısı yöntemi özel bir yaklaşımla birleştirilerek yeni bir yöntem önerilmiştir. Önerilen yöntem ile kameranın hem fiziki sensörlerindeki örüntü bilgisini hem de kamera içerisinde bulunan hassas filtrelerdeki örüntü bilgisi birlikte kullanılmaktadır. Önerilen yöntemdeki birleştirme işlemi için, iki yöntemden elde edilen bulgular 2 katmanlı bir yapay sinir ağı eğitiminde kullanılmıştır. Böylelikle kamera cihazının iki farklı örüntü bilgisi dengeli bir şekilde tek bir yöntem içerisinde kullanılmıştır. Ek olarak önerilen yöntem, literatürdeki eşdeğer yöntemler ile karşılaştırılmıştır. Karşılaştırma neticesinde önerilen yöntemin diğer yöntemlere göre daha başarılı olduğu deneysel sonuçlar ile gözlemlenmiştir. Önerilen bu yeni yöntem ile sayısal görüntülerdeki adli bilişim alanına katkıda bulunulması hedeflenmiştir.

Bu tez 6 ana bölümden oluşmaktadır. İlk bölüm giriş bölümüdür. Bu bölümlerin başlıkları ve içerikleri şu şekildedir.

1-GİRİŞ: Bu bölümde tez hakkında genel bilgilendirme yapılmıştır.

2-KAYNAK ARAŞTIRMASI VE KAYNAK ARAŞTIRMASI: Bu bölümde adli bilişim alanında sayısal görüntüler üzerindeki oynamaları tespit eden diğer yöntemler araştırılmış ve incelenmiştir. Ayrıca tez içerisinde önerilen yöntem için gerekli olan yapay sinir ağları ve evrişimsel sinir ağları örnekler verilerek anlatılmıştır.

3-MATERYAL ve YÖNTEM: Bu bölümde PRNU tabanlı ve derin öğrenme tabanlı yöntemler ile önerilen yöntem detaylıca verilmiştir. Ek olarak deneylerde kullanılan veri seti de bu bölümde bahsedilmiştir.

4-BULGULAR: Bu bölüm tez kapsamında yapılan deneyler ve elde edilen sonuçları karşılaştırma yaparak vermektedir.

5-TARTIŞMA ve SONUÇ: Bu bölümde elde edilen sonuçların yorumlanması, önerilen yöntemin iyi ve eksik yanları, çalışma aralığı ve son olarak yöntemin yorumlaması yapılmıştır.

2 KURAMSAL TEMELLER VE KAYNAK ARAŞTIRMASI

Adli bilişim alanında sayısal resimlerdeki oynama işleminde 2 önemli problem bulumaktadır. Birincisi sayısal resim oynanmış mı? İkincisi ise sayısal resim üzerinde oynanan bölgenin tespiti nasıl yapılır? Bu iki problem için birçok çalışma gerçekleştirilmiştir. Resmin genelinde bir oynanma işlemi olup olmadığının tespiti için farklı açıdan yaklaşan araştırmalar bulunmaktadır. Örneğin, Popescu ve Farid (2005) ve Kirchner (2008) sayısal bir resmin çözünürlüğü değiştirildiyse, (özellikle büyütme işlemi için) resim üzerindeki çözünürlük değişimi işleminden kalan kalıntıları tespit ederek resmin geneli hakkında bir yorum yapmamıza olanak sağlamıştır. Cao ve ark. (2010) ve Kirchner ve Fridrich (2010) resim üzerine uygulanmış ortalama filtre işlemini açığa çıkarmaktadır. Bianchi ve Piva (2012), Conotter ve ark. (2013) ve Thai ve ark. (2017) ise sayısal resimlerdeki JPEG sıkıştırmasından kaynaklanan kalıntılara bakarak resmin sıkıştırılıp sıkıştırılmadığı üzerinde araştırma yapmışlardır.

Lukas ve ark. (2005) ve Lukáš ve ark. (2006) kamera model tanımada oldukça iyi çalışan PRNU (Photo Response Non-Uniformity) tabanlı bir yöntem önermişlerdir. Bu yöntem ile resimlerin hangi kamera cihazından çekildiği bulunabilmektedir. Bu yöntem ayrıca Kaynak Araştırması bölümünde detaylıca anlatılacaktır. Bölgesel oynama tespiti için ise literatürde görüntü üzerindeki adli bilişim alanında farklı çözümler sunulmuştur. Lin ve ark. (2009)'nin önerdiği ADQ1 (Aligned Double Quantization) yöntemi, oynanmış 8×8 görüntü blokları üzerindeki çifte JPEG sıkıştırılmasından kalan izleri tespit etmeyi amaçlamıştır. Bütün görüntü blokları için toplanan ayrık kosinüs dönüşümü histogramları değerlendirilerek her bir bölge için bir olasılık değeri üretilir. Ancak orijinal resimlerlerin JPEG formatında olup oynama işleminden sonra tekrardan JPEG formatında kayıt edilmesi gerekmektedir. Li ve ark. (2009)'in önerdiği BLK yöntemi, temel olarak JPEG formatının çalışma mantığına dayanmaktadır. JPEG kendi doğası gereği, resim üzerinde sıkıştırma işleminden kaynaklı 8×8' lik periodik izler bırakmaktadır. Bu periodik izler içerisindeki bozuklukları tespit ederek resim üzerindeki oynanmış ya da kopyalanıp yapıştırılmış bölgelerdeki izleri tespit edebilmektedir. Ferrara ve ark. (2012)'in önerdiği

CFA1 (Color Filter Array) yöntemi, bir resmin çekilmesi esnasındaki renk filtersininden sonraki ara kestirim işlemini baz almaktadır. Bu ara kestirim işlemi resim üzerinde bir iz bırakmaktadır. Resim üzerindeki bu izlerin bozukluğu bölgesel olarak oynama tespiti yapılabilmesine olanak sağlamıştır. Dirik ve Memon (2009)'in yöntemi (CFA2) bir önceki yöntemde bahsedilen CFA (Renk filtresi) örüntülerini tahmin ederek oynanmış bölgeyi tespit etmeyi amaçlamıştır. Ye ve ark. (2007)'in DCT yöntemi, resmin güvenilir bölgelerinden kuantalama matrisi tahmin ederek, ayrık kosinüs dönüşümü değerlerinin histogramındaki süreksizliği tespit etmeyi önermiştir. Böylelikle resim üzerindeki oynanmış bölgenin bulunabileceğini göstermiştir. Krawetz ve Solutions (2007)'in ELA (Error Level Analysis) yöntemi, resmin belirli bölgesinin diğer bölgelerine göre daha düşük bir JPEG sıkıştırmasına maruz bırakıldığında bu bölgeyi hata analiz yöntemiyle tespit etmeyi amaçlamaktadır. Farid (2009)'in GHO (Ghosts detection) yöntemi, daha önce farklı bir kalitedeki JPEG değeriyle sıkıştırılan resmin belirli bir bölgesini bulmayı hedeflemektedir. Temel olarak analiz edilen resim ile resmin farklı kalitede tekrardan kaydedilmiş değişen JPEG versiyonları arasındaki karesel farka bakarak oynanmış bölge tespiti yapılmak istenmiştir.

PRNU yöntemi kaynak cihaz tanıma probleminde kullanılmasının yanında, bölgesel oynama tespiti probleminde de kullanılmaktadır. Ancak analiz edilen görüntü bloğunun çözünürlüğü küçüldükçe üretilen sonucun güvenilirliği azalmaktadır. Bu yöntemi daha iyi hale getirmek için Chierchia ve ark. (2014) PRNU tabanlı bölgesel oynama tespit edici önermişlerdir. Klasik PRNU yöntemine ek olarak Markov Random Field (Markov Rastgele Alanı, MRF) kullanarak her bir blok için karar vermek yerine resmin genelinde tek seferde bir karar vermektedir. Bu yöntemde sabit bit eşikleme değeri seçmek yerine Bayes kuralı ile otomatik olarak oynanmış bölgeler hesaplanmaktadır. Ayrıca standart PRNU yönteminde kullanılan dalgacık dönüşümü yerine BM3D filtresi kullanmaktadır. Bu yönteme ek olarak Korus ve Huang (2017) klasik PRNU yöntemini farklı boyutlardaki pencerelerden elde ettiği sonuçları matematiksel bir işlem ile birleştirerek PRNU yöntemi ile oldukça başarılı çalışan bir bölgesel oynama tespit edici önermiştir. Korus ve Huang (2017) ve Chierchia ve ark. (2014) yöntemleri ayrıca karşılaştırma bölümünde de

anlatılacaktır.

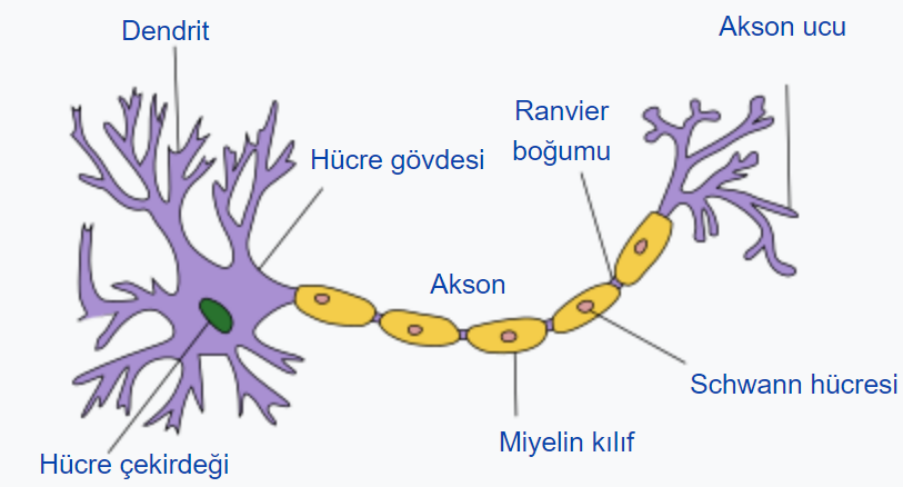
Bu yöntemlere ek olarak son yıllarda derin öğrenme tabanlı oynama tespit eden yöntemler önerilmiştir. Özellikle Bayar ve Stamm (2016) genel olarak çalışabilen derin öğrenme tabanlı bir yaklaşım sunmuştur. Sunduğu yaklaşım ile belirlenen manipülasyon çeşitlerini öğrenebilen derin öğrenme makinesi eğitilebilmiştir. Daha sonrasında ise Tuama ve ark. (2016) ve Bondi ve ark. (2017b) derin öğrenme tabanlı kamera model sınıflandırıcısını başarılı bir şekilde modellemişlerdir. Bu fikri baz alarak Bondi ve ark. (2017c) başka bir kamera modelinden gelen kopyala yapıştır oynama çeşidi için genel çalışabilen derin öğrenme tabanlı bir yöntem önermiştir. Önerilen yöntem eğitilen derin öğrenme tabanlı kamera model sınıflandırıcısını özellik çıkaran bir makine olarak kullanmıştır. Bu makinenin üretmiş olduğu çıktıları iteratif bir algoritma ile kümeleme yaparak çıktı resmi üzerinde oynanmış ve oynanmamış bölgeyi tespit edebilmiştir.

Bu tez kapsamında standart PRNU tabanlı kaynak cihaz tanıma yöntemi ile derin öğrenme tabanlı kamera model sınıflandırıcısı yöntemi özel bir yaklaşımla birleştirilerek yeni bir yöntem önerildiği için PRNU tabanlı yöntem ile derin öğrenme tabanlı kamera model sınıflandırıcısı yöntemi sonraki bölümde detaylıca anlatılacaktır. Ayrıca PRNU tabanlı kaynak cihaz tanıma yöntemi, tez içerisinde PRNU yöntemi olarak ifade edilecektir.

2.1 Yapay Sinir Ağları (YSA)

Yapay sinir ağları (YSA), insan sinir sistemindeki sinir hücrelerinin matematiksel modelini basit bir şekilde çıkararak oluşturulan ve bazı problemlerde oldukça başarılı sonuçlar üreten bir yöntemdir. YSA temel olarak insan beynindeki sinir sisteminden ilham alınarak oluşturulmuştur. İnsan beyninde bulunan basit bir sinir hücresini Şekil 2.1'de görebiliriz.

Bir sinir hücresinin çalışma mantığı temel olarak şu şekildedir: elektrik sinyali dentrit'e gelir. Dentritten hücre çekirdeğine iletilir. Buradan da akson yardımıyla bir diğer sinir hücresine iletilir. Burada belirtilmesi gereken birkaç husus vardır. Bir sinir hücresine



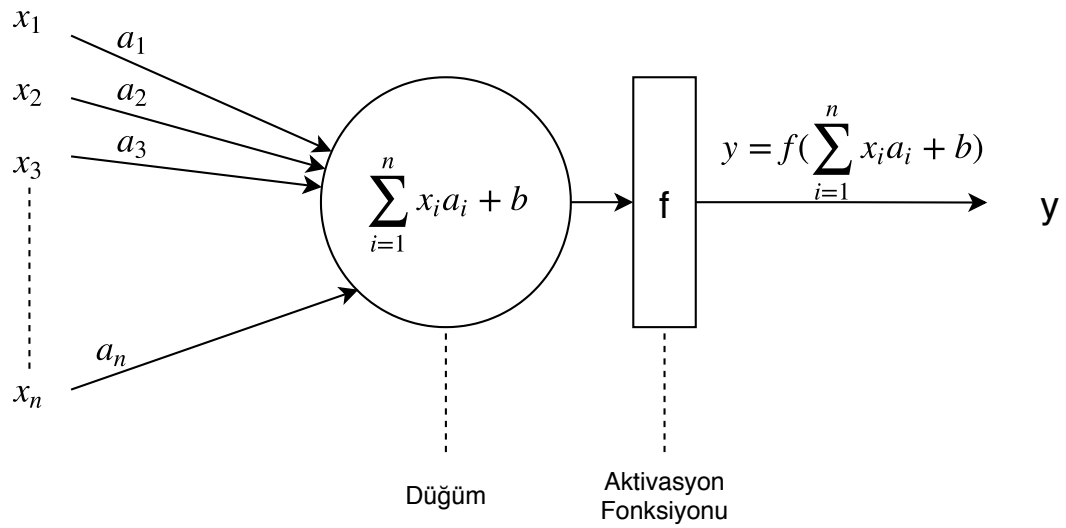
Şekil 2.1. Sinir hücresi (Anonim, 2019)

aynı anda birden fazla sinyal gelebilir. Bu durumda sinir hücresinin bütün bu sinyalleri harmanlayıp tek bir sinyal haline getirmesi gerekmektedir. İşte bu işlemi hücre çekirdeği yapmaktadır. Bir sinir hücresinin bu yapısını basit bir şekilde modelleyecek olursak (bkz Şekil 2.2) x_i değeri girdi sinyalleri a_i her bir girdinin ağırlığı, n girdi sayısı, b bias değeri ve f aktivasyon fonksiyonu olmak üzere y çıktısını

$$y = f(\sum_{i=1}^{n} x_i a_i + b) \tag{2.1}$$

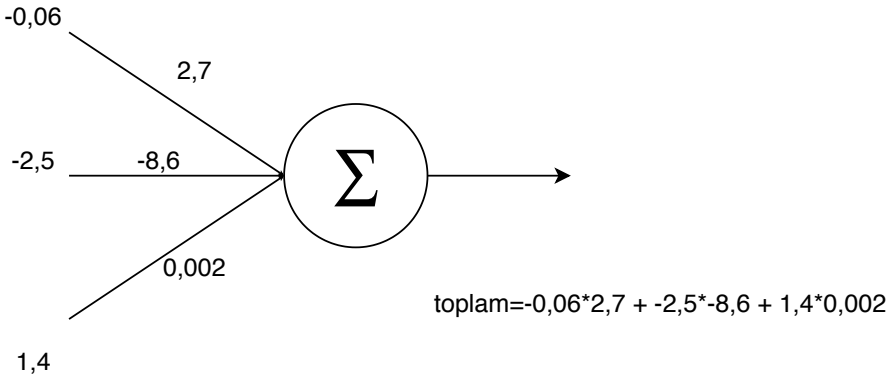
şeklinde yazabiliriz. Genel olarak bu harmanlama işlemi x girdi sinyallerinin a ağırlık sinyalleriyle çarpımının toplamı şeklindedir. Bir sinir düğümü ek olarak bu toplama bir aktivasyon fonksiyonuyla normalleştirme işlemi yapabilir. Aktivasyon fonksiyonu sayesinde üretilen çıktının daha anlamlı değerler olması sağlanır. Yukarıda verilen denklem 2.1'de b bias değeri bulunmaktadır. Bu değer ile ayrıca hesaplanan değerin normalleştirilmesi yapılabilir. Örneğin $x_i \times a_i$ çarpımından sıfır değeri gelirse, bu düğüm etkisini kaybedicektir. Bu duruma engel olabilmek adına b bias değeri eklenebilir.

Bir düğüm için basit bir örnek verelim. x girdi sinyallerimiz -0,06, -2,5 ve 1,4 olsun. Bu sinyallerin gelmiş olduğu kanalların a ağırlık değerleri 2,7, -8,6 ve 0,002 olsun. Örneğin görsel versiyonu Şekil 2.3'de verilmiştir. Bias değeri sıfır olarak kabul edilirse, x değeri Şekil 2.3'deki gibi hesaplanır. Yani her bir girdi sinyali ile bu sinyallerin gelmiş olduğu ağırlık değerlerinin çarpımlarının toplamı olarak bulunur. Bu adımdan sonra belirlenen



Şekil 2.2. Yapay sinir hücresi modeli

aktivasyon fonksiyonu hesaplanarak çıktı değeri üretilir.



Şekil 2.3. Yapay sinir ağı örnek hesaplama görseli

Az önce verilen örnek tamamen rastgele seçilen değerlerden oluşturulmuştur. Ancak bu değerlerin gerçek bir uygulamada kullanılan değerler olduğunu düşünelim. Her bir ağırlık değeri birbirinden farklı olarak verilmiş. Bunun anlamı her bir kanalın aslında birbirlerinden daha az veya daha fazla öneme sahip oldukları anlamına gelmektedir. Yani şöyle ki 2.3'deki düğüm için aslında en önemli kanal a_1'in geldiği kanaldır. Çünkü en büyük ağırlık değerini o kanal almıştır. a_3 değeri neredeyse sıfıra yakın olduğundan bizim düğümümüz için aslında buradan gelecek olan girdinin çok da önemli olmadığı anlaşılmaktadır. Son olarak a_2 ağırlığının negatif değerde olması demek, bu kanaldan

gelen sinyalin ters etki yapacağını göstermektedir (Şekil 2.3'de hem a_2 hem de x_2 değeri negatif olduğundan bu negatif etki pozitife dönüşmüştür).

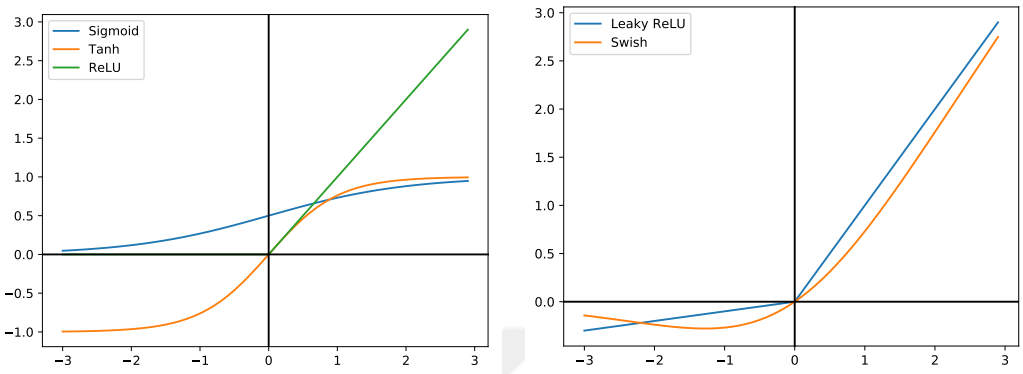
Aktivasyon Fonksiyonu: Girdi sinyallerinin ağırlık değerleriyle çarpımından elde edilen değer çıktı olarak verilmeden önce bir aktivasyon fonksiyonuna girdi olarak verilebilir. Bu adımın yapılmasındaki amaç hem sistemin doğrusallığını değiştirebilmek hem de üretilen değerin anlamlı bir hale getirilmesini sağlamaktır. Örneğin basit bir karar mekanizması için bir düğümün en fazla 1 en az 0 değeri üretmesi olasılıksal hesaplama yapmak için mantıklıdır. Dolayısıyla üretilen çok yüksek veya çok düşük çarpım değerini basit bir sigmoid fonksiyonuyla normalleştirebiliriz.

Çizelge 2.1. Aktivasyon fonksiyonlarının matematiksel ifadeleri

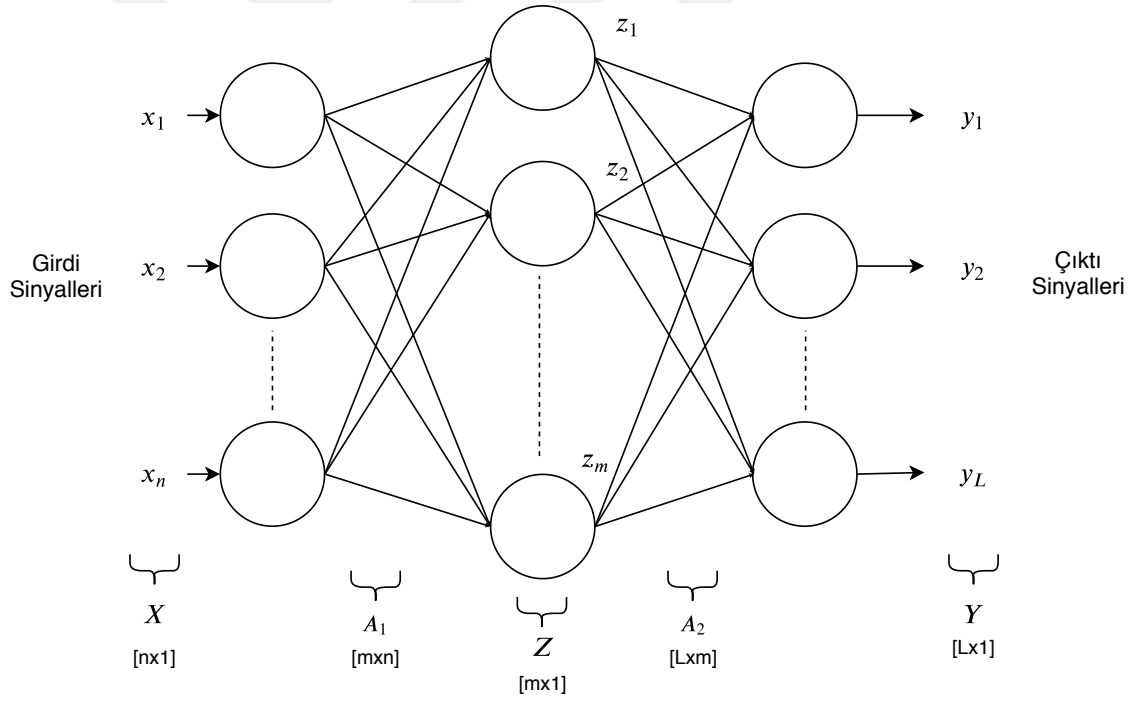
Aktivasyon Fonksiyonu	**Denklem**	**Aralık**
Doğrusal Fonksiyon	$f(x) = x$	(-∞, ∞)
Basamak Fonksiyonu	$f(x) = \begin{cases} x < 0 \text{ için } 0 \\ x \geq 0 \text{ için } 1 \end{cases}$	{0,1}
Sigmoid Fonksiyon	$f(x) = \frac{1}{1+e^{-x}}$	(0,1)
Hiperbolik Tanjant Fonksiyonu	$f(x) = tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	(-1,1)
ReLU	$f(x) = \begin{cases} x < 0 \text{ için } 0 \\ x \geq 0 \text{ için } x \end{cases}$	[0,∞]
Leaky(Sızıntı) ReLU	$f(x) = \begin{cases} x < 0 \text{ için } 0.01x \\ x \geq 0 \text{ için } x \end{cases}$	(-∞, ∞)
Swish Fonksiyonu	$f(x) = x \cdot sigmoid(\beta x) = \frac{x}{1+e^{-\beta x}}$	(-∞, ∞)

Çizelge 2.1'de swish fonksiyonunda verilen β, öğrenilebilir bir katsayıdır. Normalleştirme işlemi her problem için iyi sonuç üretmeyebilir. Dolayısıyla literatürde birçok aktivasyon fonksiyonu bulunmaktadır. Bu fonksiyonlardan en çok kullanılanları Çizelge 2.1'da gösterilmiştir. Aktivasyon fonksiyonlarının grafikleri Şekil 2.4'de gösterilmiştir.

YSA Mimarisi ve Türleri: Yapay sinir ağlarının bir ağ olabilmesi için öncelikle birden fazla düğüme sahip olması gerekmektedir. Bir yapay sinir ağında düğümler girişten çıkışa doğru katmanlar şeklindedir. Daha karmaşık modellerde ara katmanlar bulunabilmektedir. Şekil 2.5'de örnek bir yapay sinir ağı modeli gösterilmiştir.



Şekil 2.4. Aktivasyon fonksiyonları (Kizrak, 2019)



Şekil 2.5. Örnek bir yapay sinir ağı modeli

Şekil 2.5'de x girdi sinyallerini ve y çıktı sinyallerinin her birini belirtmektedir. X, girdilerin oluşturduğu vektör ve A_1 ağırlıkların bulunduğu matris olmak üzere ara katmanda üretilen değerler vektörü (Z), $Z = f(A_1X + b_1)$ şeklinde yazılabilir. Benzer şekilde girdi

olarak Z vektörünü alan son katmanda üretilen değerler $Y = f(A_2Z + b_2)$ olarak yazılabilir. Her zaman b bias değeri olmak zorunda değildir. Şekil 2.5'deki ağda bias değerinin eklendiği varsayılmıştır. Dolayısıyla konunun anlaşılması ve denklemlerin basitleştirilmesi açısından bundan sonra yapılacak olan örneklerde ve denklemlerde bias değeri kullanılmayacaktır.

YSA, ileri beslemeli ve geri beslemeli olmak üzere temel olarak ikiye ayrılır. İleri beslemeli ağlarda giriş düğümlerinden gelen sinyaller gizli katmanı geçerek çıkış katmanına doğru gider. Geri beslemeli ağlarda ise hem ileri hem de geriye doğru gidebilmektedir. Şekil 2.5'de görüldüğü gibi çoğu ağda her bir düğüm birbirine bağlı vaziyettedir. Daha önce Bölüm 2.3'de anlatıldığı gibi her bir düğüme gelen girdi sinyali ağırlık değeriyle çarpılıp aktivasyon fonksiyonuna verilir. Elde edilen değer bir sonraki düğüme aktarılır. Bu işlem çıkış katmanına gelene kadar yapılır.

YSA'da Öğrenme: Yapay sinir ağlarında şu ana kadar bir ağın nasıl çalıştığı ve yapısı anlatılmıştır. Ancak buradaki asıl zor olan kısım bir yapay sinir ağının iyi çalışabilmesi için belirlenmesi gereken ağırlık değerleridir. Yani daha önceki örnekte ağırlık değerleri direk verilmişti. Fakat bu ağırlık değerleri gerçek bir problemde verilmemektedir. Dolayısıyla bu değerleri en iyi şekilde belirlememiz gerekmektedir. Bu değerleri belirlerken makine öğrenmesi metotları kullanılmaktadır.

Konunun daha iyi anlaşılması açısından makine öğrenmesinden kısaca şu şekilde bahsedebiliriz. Bir sistemin mevcut verilere göre istenen en iyi sonucu üretebilecek hale getirilmesine makine öğrenmesi denebilir. Makine öğrenmesi gözetimli ve gözetimsiz olarak ikiye ayrılmaktadır. Gözetimsiz öğrenmede sistem elindeki verileri bir referans olmaksızın ayrıştırma veya derecelendirme yapmaktadır. Gözetimli öğrenmede ise verilen referans verilerine göre olabilecek en iyi çalışan modeli üretmeye çalışmaktadır. Örneğin insan tanıma problemini düşünelim. Biz eğer makineye içerisinde insan olan resimler için 'insan' referansı, insan olmayan resimler için 'farklı' referansını verirsek, bu gözetimli öğrenme olmuş olur. Ancak biz referansları vermeksizin sistemin kendi kendine bazı objeleri ayırt

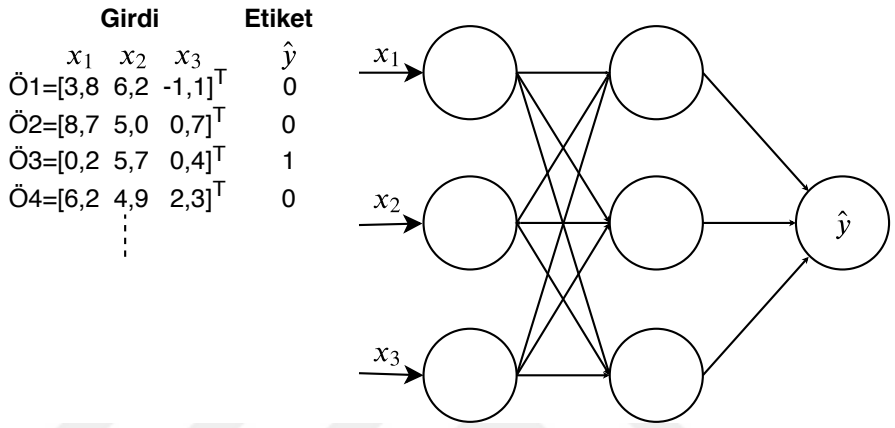
etmesini istersek bu da gözetimsiz öğrenme olur. Yani burada temel olarak problemimize göre veriler ve eğer gözetimli çalışacaksak bu verilere ait referans bilgileri gerekmektedir. Bu verilere sahip olduktan sonra makineye sadece bu verilerin hangi referansa ait olduğunu söylüyoruz. Geriye kalan kısmını algoritma hızlı bir şekilde yapmaktadır.

Ağın Eğitilmesi: Her bir düğüm için ağırlık değerlerinin belirlenmesi işlemine ağın eğitilmesi denir. Eğitim başlamadan önce bu ağırlık değerleri tamamiyle rastgele olarak verilir. Yapay sinir ağları kendilerine örnek gösterildikçe bu ağırlık değerlerini güncelleme algoritması yardımıyla değiştirirler. Burada amaç ağa gösterilen örneklerin tamamı için doğru sonucun üretilebileceği ağırlıkları belirlemektir. Ağa defalarca örnekler gösterilerek en doğru ağırlık değerleri bulunmaya çalışılır. Mantıksal açıdan düşünüldüğünde aslında yapay sinir ağına olabilecek bütün ihtimalleri gösteriyoruz. Yani milyonlarca ihtimali, hesaplayabildiğimiz her bir ihtimalin, doğru olup olmadığını hesaplıyoruz. En sonunda eğitilen ağ verilen örnek yığını için olabilecek en doğru şekilde çalışan ağırlıklara sahip olmuş olur.

Konunun iyi anlaşılabilmesi açısından basit bir ağ eğitimi örneği güncelleme algoritmasının detayına girmeden (sonraki bölümlerde ayrıca anlatılacaktır) şu şekilde verilebilir:

Elimizde ikili sınıflandırma yapabilmemiz için bir veri seti olsun. Bu veri seti içerisinde her bir örnek için 3 adet özellik tanımlanmış olsun. Mesela insanın kan değerlerindeki 3 özelliğe bakarak kişinin hasta olup olmadığını bulduğumuzu varsayalım. Elimizde her bir insan için bu üç özellik ve bu insanların hasta olup olmadıkları bilinmektedir. Her özelliği Ö olarak ifade edelim. Bu veriler ile bir yapay sinir ağı eğitebiliriz. YSA'da öğrenme gerçekleştirilirken, elimizdeki verileri 3 farklı şekilde YSA'ya öğretebiliriz. İlk olarak bütün verileri aynı anda YSA'ya vererek güncelleme gerçekleştirebiliriz. Bu şekilde yapılırsa bütün veriler için ortak bir hata elde edilmiş olacaktır. İkinci olarak elimizderi verileri (örnekleri) teker teker ağa göstererek her adımda ağırlıkları güncelleyebiliriz. Eğer veriler teker teker YSA'ya verilirse, her bir örnek için hata hesaplanıp ağırlıklar

her örnek sonunda güncellenektir. Son olarak belirli sayıda örnekleri yığınlar halinde ağa vererek ağırlıkları güncelleyebiliriz. Bu kurgusal örnekte her bir örnek YSA'ya teker teker verilerek ağırlıklar güncellenecektir. Yani her bir Ö verisinden sonra ağırlıklar güncellenecektir. İlk olarak bir yapay sinir ağı modeli belirlememiz gerekiyor. En temel ve çoğu basit problemde yüksek başarı veren tek ara katmanlı modelimiz Şekil 2.6'de verilmiştir. İlk adım olarak bu ağ üzerindeki ağırlıklar rastgele olarak belirlenir.



Şekil 2.6. YSA örnek eğitimi birinci adım: YSA modeli oluşturma

Şekil 2.7'de 4 insan için özellikler ve sınıf değeri verilmiştir (Hasta ve değil sınıfları 0 ve 1 olarak gösterilmiştir). Burada amaç verilen her bir örnek için istenen sınıf değerini üretebilecek ağırlıklara sahip bir ağ eğitebilmektir. Şekil 2.7'de gösterilen 2. adımda birinci örnek sınıfı için rastgele olarak belirlenmiş ağırlıklarla çıktı hesaplanır. Bir ağın hesaplama yaptığı bölüme skor fonksiyonu, S_A diyelim. S_A fonksiyonunu denklem 2.2'deki gibi tanımlayabiliriz.

$$y = S_A(X) \tag{2.2}$$

Hesaplanan değer 0,8 olsun. Ancak biz ağımızın 0 değerini üretmesini istiyoruz. Dolayısıyla basit bir şekilde hata fonksiyonumuzu h hata fonksiyonu, y üretilen değer ve $\hat{y}$ üretilmesi gereken sınıf değeri olmak üzere denklem 2.3'deki gibi yazabiliriz.

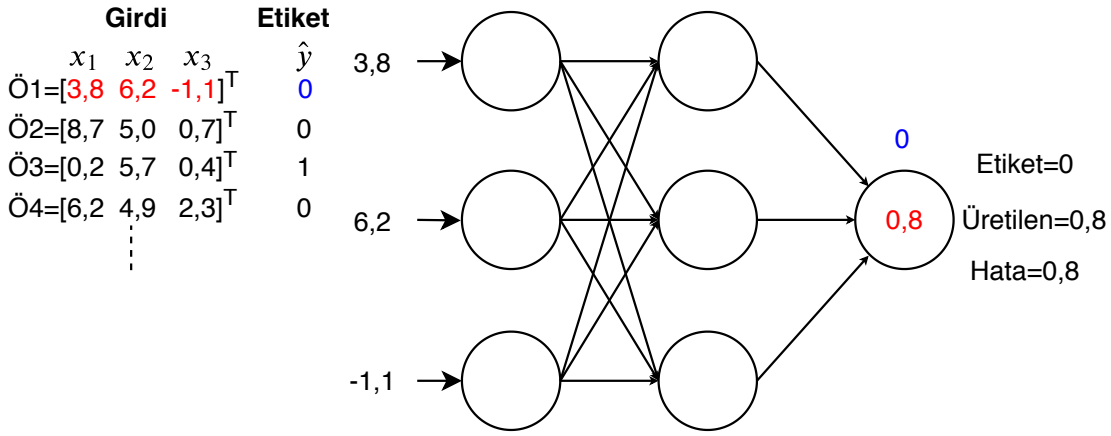
$$\begin{aligned} h_A(X) &= S_A(X) - \hat{y} \\ h_A(X) &= y - \hat{y} \end{aligned} \tag{2.3}$$

Aslında ağırlık değerlerini en uygun duruma getirme işlemi hata fonksiyonumuzu sıfır

veya sıfıra yakın bir değere getirmek ile gerçekleşir. Dolayısıyla bizim amacımız kayıp fonksiyonunu mümkün olduğunca azaltabilmektir. Hata fonksiyonu genellikle tek bir örnek için hesaplanır. Yani verilen örnek için hesaplanan değer ile etiket değeri arasındaki fark hata fonksiyonunu oluşturur. Bu hata karesel fark olabileceği gibi SVM yönteminde kullanılan 'hinge' (Anonim, 2019) hatası şeklinde de olabilir. Hata fonksiyonuna ek olarak bir de kayıp fonksiyonu bulunmaktadır. Kayıp fonksiyonu, hata fonksiyonuna göre daha geneldir. Bütün eğitim seti için elde edilen hataların toplamından oluşabilir. $K(A)$ kayıp fonksiyonu olmak üzere karesel kayıp fonksiyonu denklem 2.4'de gösterilmiştir.

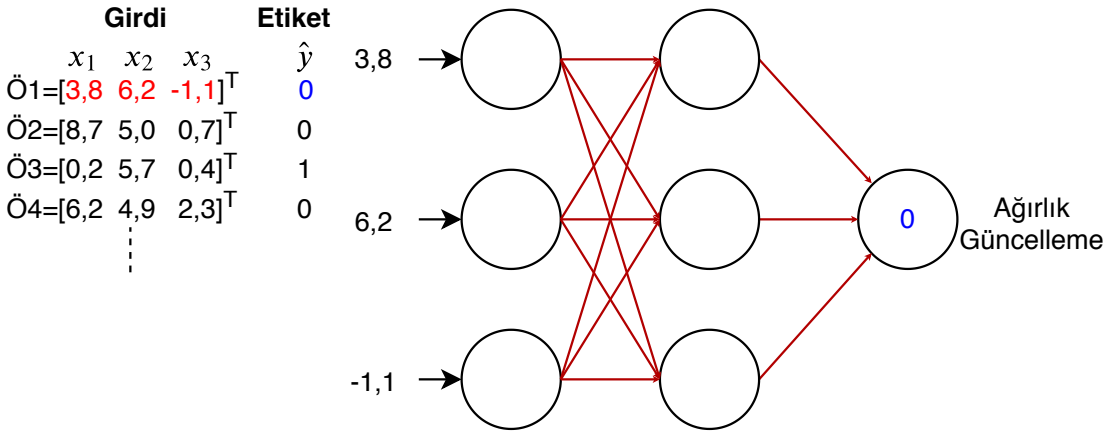
$$K(A) = \frac{1}{2}\sum_{i=1}^{m}\left(S_A(X^{(i)}) - \hat{y}^{(i)}\right)^2 \tag{2.4}$$

Denklem 2.4'deki m örnek sayısını, $X^{(i)}$ i'ninci girdi vektörünü, $\hat{y}^{(i)}$ ise $X^{(i)}$ için etiket değerini belirtmektedir. Öncelikli amaç hata fonksiyonunu, daha sonrasında ise kayıp fonksiyonunu azaltmaktır. Hata fonksiyonunda da kayıp fonksiyonunda da amaç verilen girdi için en iyi ağırlık değerlerinin belirlenip belirlenmediğini ölçmektir. Bu sebepten ötürü farklı kayıp ve hata fonksiyonları bulunmaktadır. Kayıp fonksiyonu zaten hata fonksiyonunu içerisinde barındırdığı için asıl amaç kayıp fonksiyonunu azaltmaktır.



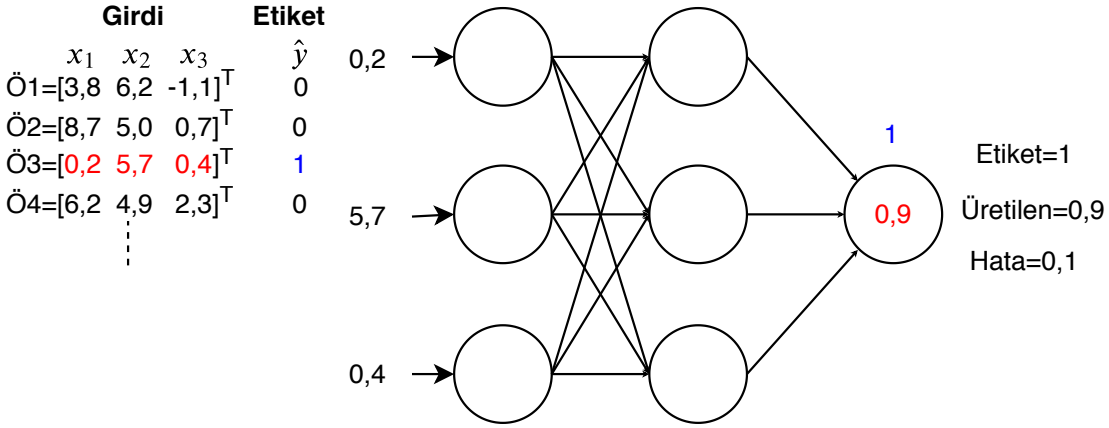
Şekil 2.7. YSA örnek eğitimi ikinci adım: Hesaplama

Kayıp fonksiyonu hesaplandıktan sonra Şekil 2.8'de verilen 3. adımda güncelleme algoritmasıyla ağırlık değerleri kayıp fonksiyonunu sıfıra yakın yapacak şekilde düzenlenir. Şekil 2.7'deki T, transpoz işaretidir. Bir sonraki adıma ise güncellenen ağırlık değerleri aktarılır.



Şekil 2.8. YSA örnek eğitimi üçüncü adım: Güncelleme

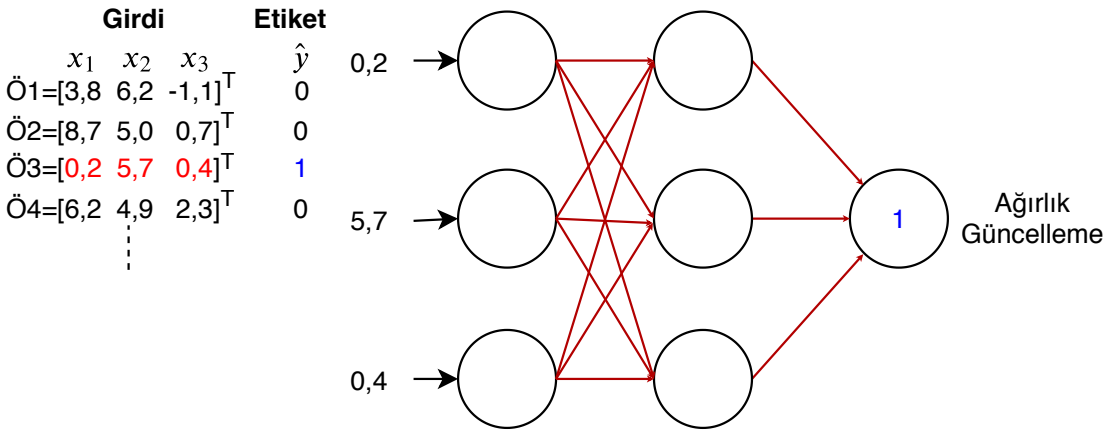
İlk veriye güncelleme işlemi yapıldıktan sonra 4. adımda (Şekil 2.9) ikinci veriye ait olan hata hesaplanır. Burada olması gereken değer 1, ancak üretilen değer 0,9 olduğundan hata 0,1 olarak hesaplanır.



Şekil 2.9. YSA örnek eğitimi dördüncü adım: Hesaplama

Şekil 2.10'de verilen 5. adımda ise tekrardan kayıp fonksiyonunu sıfıra indirecek ağırlık değerlerini belirleyebilmek için güncelleme algoritması uygulanır.

Adım 2-3 ve adım 4-5 aslında birbirlerinin tekrarını içermektedir. Örneğin iyi anlaşılması açısından fazladan 2 adım gösterilmiştir. Bu adımlar veri kümesindeki bütün örnekler için onlarca defa tekrarlanır. Böylelikle ağımız, elimizdeki veri kümesi için istenen değerleri üretebilen bir ağ haline gelmiş olur. Sistemimiz, elimizdeki veri kümesi ile



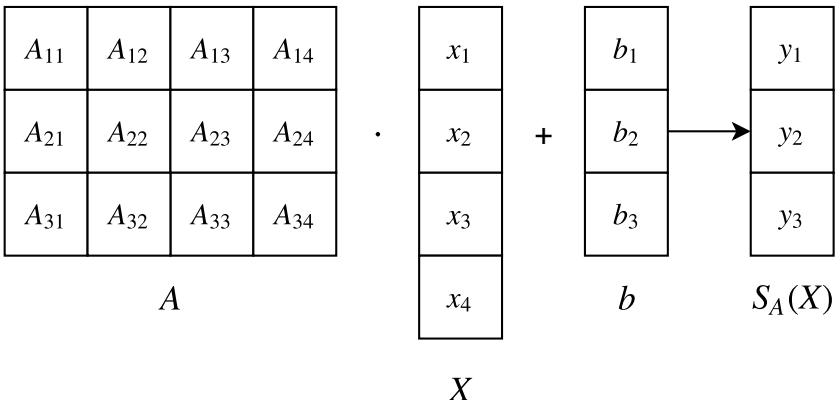
Şekil 2.10. YSA örnek eğitimi beşinci adım: Güncelleme

eğitildi. Sistemin başarısını ölçmek için eğitim esnasında hiç kullanılmayan ek bir veri seti için $S_A(X)$ fonksiyonu güncelleme işlemine tabi tutulmaksızın hesaplanır. Hesaplanan ortalama hata değeri sistemin başarısını verir. Yani eğer biz ağımıza yeterince farklı örnekler gösterirsek, ağımız en farklı örneklerde dahi doğru sonucu üretecektir. Burada önemli olan veri kümesindeki örneklerin çok ve çeşitli olmasıdır.

Çok Sınıflı Eğitim İçin Skor Fonksiyonu: Şu ana kadar olan kısımda regresyon probleminde kullanılan YSA'lar için skor ve kayıp fonksiyonları tanımlandı. Regresyon probleminde çıktı olarak çoklu sınıf değerleri yerine tek bir değer üretilmektedir. Sınıflandırma probleminde de skor fonksiyonu ufak birkaç farklılıkla aynıdır. Şöyle ki her bir sınıf için ayrı bir skor hesaplanır. Şekil 2.11'de çok sınıflı problem için skor fonksiyonu hesaplama işlemi gösterilmiştir.

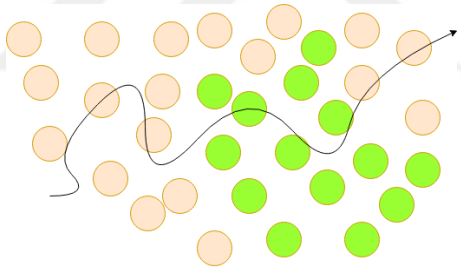
Şekil 2.11'de 4 girdili ve 3 sınıflı çıktı için örnek skor fonksiyonu hesaplaması gösterilmiştir. Her bir sınıf için ayrı y değeri üretilir. A matrisinin satır değerleri her bir sınıf için bulunması gereken ağırlık değerlerini barındırmaktadır. Denklem 2.4'de verilen kayıp fonksiyonundan farkı A'nın matris olması ve çıktı olarak Y vektörü üretilmesidir. Üretilen değer denklem denklemi 2.5'de verilmiştir.

$$Y = S_A(x) = AX + b \tag{2.5}$$



Şekil 2.11. Çok sınıflı skor fonksiyonu hesaplama örneği

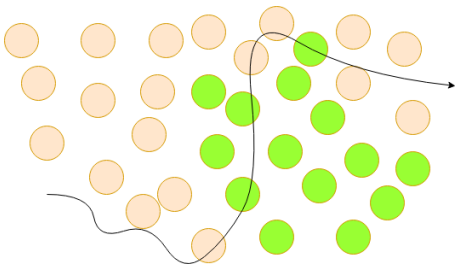
Karar Çizgisi: Öğrenme adımını verilerin birbirinden en iyi şekilde ayrılması olarak düşünebiliriz. Veri kümemizi uzay düzlemine dağıttığımızı düşünelim. Örneğin Şekil 2.12'de kırmızı noktalar birinci sınıfa, yeşil noktalar ikinci sınıfa ait olsun. Öğrenme adımında bizim yapmak istediğimiz bu iki veriyi ayıracak en doğru karar çizgisini belirleyebilmektir. Biz ağırlıkları ilk adımda rastgele olarak atadığımızda yeşil ve kırmızı noktaları ayırması gereken karar çizgisi Şekil 2.12'deki gibidir.



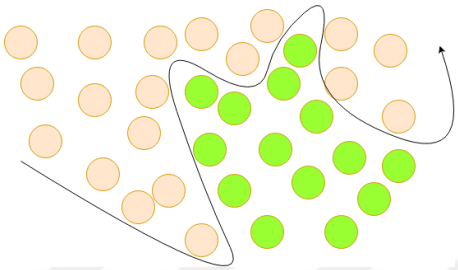
Şekil 2.12. Öğrenme gerçekleşmeden önceki karar çizgisi

Ancak öğrenme adımı başladıktan sonra yani hatayı sıfıra yaklaştırmak için ağırlık değerlerini değiştirme işlemi tekrarlandıkça bu karar çizgisi 2.13'deki gibi daha ayırt edici hale gelmektedir.

Eğitim işlemi bitirildiğinde ise bizim ağımız, elimizdeki veri setini en iyi şekilde ayırabilen bir karar çizgisini Şekil 2.14'deki gibi oluşturmuş olur.

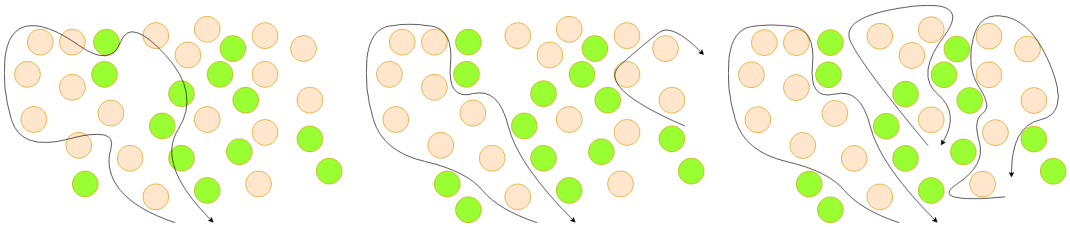


Şekil 2.13. Öğrenme gerçekleşirkenki karar çizgisi



Şekil 2.14. Öğrenme gerçekleştikten sonraki karar çizgisi

Katman Sayısını ve Boyutunu Belirlemek: YSA'da katman sayısı belirlemenin net bir yöntemi tam olarak bulunmamaktadır. Burada problemin karmaşıklığına göre modelin katman sayısını arttırmak daha doğru olabilir. Şekil 2.15'de ikili sınıflandırma için farklı katman sayılarındaki karar veren ayırımları görebiliriz.



(a) 3 katman **(b)** 6 katman **(c)** 15 katman

Şekil 2.15. Katman sayısının eğitime etkisi

Fark edildiği üzere çok katmanlı modelde verinin tamamı için doğru sonucu üretecek model oluşturulmuştur. Ancak bu durum aşırı öğrenme sorununu doğurmaktadır. Aşırı öğrenme modelin eğitimde kullanılan veri setini çok iyi öğrenip test setinde iyi çalışmaması durumudur. Yalnızca eğitim verilerinde doğru karar verdiği için buna aşırı öğrenme denmiştir. Eğer bizim problemimiz genel çalışabilen basit ikili bir sınıflandırma yapmak ise veri dağılımı 2.15'de olduğu gibi değil daha basit olacaktır. Ancak basit bir problem

için bile çok katmanlı bir model kullanmak problemi çözemeyebilir, çözse bile zor yoldan çözüme ulaşılmış demektir. Şekil 2.15'deki gösterildiği gibi katman sayısının artması demek karmaşık fonksiyonları barındırabilen bir sistemin var olması demektir. Problemimiz karmaşık ise çok katman kullanmak daha avantajlı olacaktır.

Ağırlık Değerlerinin Düşürülmesi: Ağ eğitilirken denklem 2.4'de verilen kayıp fonksiyonu kullanılmaktadır. Ancak ağırlıklar güncellenirken belirli bir kanalın ağırlık değeri sürekli olarak artabilir. Bu durumda ise girdi vektöründeki tek bir değer tek başına sonucu büyük ölçüde etkileyebilir. Bu durumda diğer kanallardan gelen bilginin önemi kalmamaktadır. Dolayısıyla aslında skor fonksiyonu ile beklenen değer arasındaki hata gibi ağırlık değerlerinde de bir hata oluşmaktadır. Bu hatanın eğitim esnasında giderilmesi edilmesi gerekir. Yani tek başına sürekli olarak büyüyen bir ağırlığı engelleyerek daha küçük ağırlıkların da söz sahibi olması sağlanır. En çok kullanılan ağırlık düşürme fonksiyonu denklem 2.6'de verilmiştir.

$$V(a) = \sum_i a_i^2 \tag{2.6}$$

Ağırlık düşürme fonksiyonu girdi verisi tabanlı değil sadece ağırlık tabanlı bir fonksiyondur. Bu fonksiyon kayıp fonksiyonuna eklenerek ağırlık değerlerinin hem aşırı büyümesi engellemek hem de daha geniş bir aralığa yayılmasını sağlamaktadır. Dolayısıyla yeni kayıp fonksiyonu $K(A)$, S_A skor fonksiyonu, a ağırlık değeri, $X^{(i)}$ i'ninci girdi vektörünü, $y^{(i)}$, $X^{(i)}$ için etiket değerini, λ ağırlık düşürme parametresini, m toplam örnek sayısını belirtmek üzere denklem 2.7'de verilmiştir.

$$K(A) = \frac{1}{m}\sum_{i=1}^{m}\left(S_A(X^{(i)}) - \hat{y}^{(i)}\right)^2 + \lambda\sum_{i=1}^{k} a_i^2 \tag{2.7}$$

Örneğin girdi vektörümüx $x = [1,1,1,1]$ olsun. 2 adet ağırlık vektörümüz ise $a_1 = [1,0,0,0]$ ve $a_2 = [0,250,250,250,25]$ olsun. Girdi vektörünün hem a_1 hem de a_2 ile noktasal çarpımı 1 sonucunu vermektedir. Ancak a_1 için $V(a_1)$ 1 değerini üretirken, a_2 için $V(a_2)$ 0,25 değerini üretmektedir. Dolayısıyla daha düşük hata veren a_2 ağırlıkları tercih edilecektir. Ağırlık düşürme fonksiyonu daha küçük ve daha eşit dağılımlı ağırlıklar belirleyeceği için, sadece değeri büyük girdilere odaklanmak yerine bütün girdi değerlerine

odaklanacaktır. Ayrıca bu durum ağın aşırı öğrenmesini de engellemektedir.

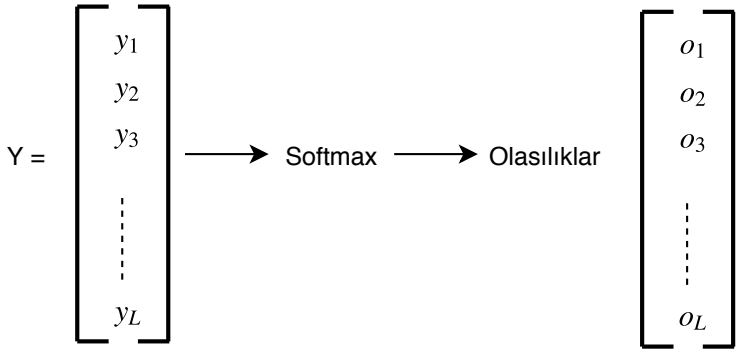
Softmax Sınıflandırıcısı: SVM ve softmax sınıflandırıcıları yapay sinir ağlarındaki en popüler sınıflandırıcılardır. Bu tez kapsamında SVM kullanılmadığından SVM konusu anlatılmayacaktır. Aslında softmax sınıflandırıcısı ikili mantıksal regresyonun ikiden fazla sınıflandırmayı yapabilen versiyonudur. Softmax sınıflandırıcıda $S_A(X)$ skor fonksiyonu sonucunda elde edilen değerleri normalize ederek her bir sınıf için olasılık değeri üretmektedir. Tek bir örnek için üretilen olasılık değerleri toplamı 1 olacak şekilde normalize işlemi gerçekleştirilir. Softmax fonksiyonu, denklem 2.8'de verilmiştir.

$$P(y_i) = Softmax(y_i) = \frac{e^{y_i}}{\sum_{j=1}^{L} e^{y_j}}, 1 \leq i \leq L \tag{2.8}$$

Softmax fonksiyonu ile elde edilen olasılık değerleri cross entropy (CE) kayıp fonksiyonunu kullanmaktadır. CE fonksiyonu denklem 2.9'de verilmiştir.

$$CE(y) = -\sum_{i=1}^{L} y_i log\big(P(y_i)\big), 1 \leq i \leq L \tag{2.9}$$

Burada y_i skor fonksiyonu tarafından hedef sınıf için üretilen değeri, e üstel değeri, L sınıf sayısını, $\sum_j e^{y_j}$ toplamı ise bütün sınıflar için üretilen değerlerin toplamını göstermektedir. Tek bir sınıf için üretilen değeri bütün değerler toplamına bölerek olasılık değerini bulmuş oluyoruz. Bu değerin eksi logaritması ise bize hatayı vermektedir. Şekil 2.16'de softmax ile üretilen değerlerin olasılık değerine dönüşümünü görsel olarak verilmiştir.



Şekil 2.16. Softmax ile olasılık hesaplama

Gradyan İnişi Algoritması: Yapay sinir ağını eğitirken kullanılan çeşitli güncelleme algoritmaları bulunmaktadır. Bu algoritmalar içerisinde Gradyan İniş Algoritması en yaygın olarak kullanılan algoritmadır. Bu tez kapsamında bu algoritma kullanılacağından bu bölümde bu algoritmanın matematiksel açıklaması ve bir adet örnek gösterilecektir. Ayrıca şu ana kadarki anlatılan kısımların uygulamasını göreceğiz.

Temel olarak gözetimli bir öğrenme yaparken öncelikle skor fonksiyonunun belirlenmesi gerekmektedir. Algoritmanın çalışmasını anlaşılır seviyede gösterebilmek için doğrusal bir fonksiyon olarak tanımlayalım.YSA'da girdiler ile girdilerin ağırlıkları çarpımlarının toplamı skor fonksiyonunu verdiğinden aslında pek bir fark bulunmamaktadır. $f_s(x)$ skor fonksiyonu, a_i her bir ağırlık değeri, x_i her bir girdimiz olmak üzere

$$y = S_A(X) = a_0x_0 + a_1x_1 + a_2x_2 \tag{2.10}$$

şeklinde bir skor fonksiyonumuz olsun. Denklem 2.10'deki x_1 girdimizin birinci değerini x_2 ise ikinci değerini göstermektedir. Denklemin sadeleştirilmesi adına x_0 değeri 1 olarak belirlenmiştir. Denklem 2.10'i kısaca

$$S_A(X) = \sum_{i=0}^{n} a_ix_i = AX \tag{2.11}$$

yazılabilir. Denklem 2.11'daki X ve A değerleri vektör türündendir. Buradaki n değeri girdi sayısının bir eksiğidir. Amacımız, verilen eğitim seti için (örnek kısmında gösterilecek) denklem 2.10'nin istenen etiket değerlerine en yakın değeri üretmesi için a_i değerlerinin belirlemesi gerekiyor. İstenen değer ile üretilen değerin birbirine ne kadar yakın olduğunu bulabilmemiz için kayıp fonksiyonuna ihtiyacımız var. Bu problem için karesel kayıp fonksiyonunu kullanacağız. Karesel kayıp fonksiyonu denklem 2.12'de verilmiştir.

$$K(A) = \sum_{i=1}^{m} \frac{1}{2}\left(S_A(X^{(i)}) - y^{(i)}\right)^2 \tag{2.12}$$

Denklem 2.12'deki m değeri veri setindeki örnek sayısını, $X^{(i)}$ i'ninci örnek için girdi vektörünü, $y^{(i)}$, $X^{(i)}$ için etiket değerini sembolize etmektedir. Hatayı matematiksel olarak ifade ettikten sonra artık $K(A)$ fonksiyonunu küçültecek en uygun a_i değerlerini bulabilmek için Gradyan iniş algoritması kullanılacaktır. Gradyan iniş algoritması denklem

2.13'de verilmiştir.

$$a_j := a_j - \eta \frac{\partial}{\partial a_j} K(A) \tag{2.13}$$

Denklem 2.13'deki a_j her bir girdi değerini belirtmektedir. Güncelleme işlemi bütün j değerleri için (j=0,1,2,3,4,....,n) hesaplanır. Buradaki η değeri öğrenme katsayısıdır. Bu katsayı bizim öğrenme adımının büyüklüğünü belirlememize olanak sağlar. Eğer çok büyük bir değer seçilirse öğrenme gerçekleşmez. Eğer çok küçük bir değer seçilirse öğrenme çok yavaş gerçekleşir. Dolayısıyla bu değerin oldukça hassas bir şekilde belirlenmesi gerekmektedir.

Denklem 2.13'i uygulayabilmemiz için denklem içerisindeki kismi türev değerlerinin hesaplanması gerekmektedir. İfadelerdeki karmaşıklığın azaltılması için bu hesaplama işlemini yalnızca tek bir örnek olduğu varsayılarak türetilecektir. Ardından birden fazla örnek için olan formulü yazılacaktır (Sadece x ve y).

$$\begin{aligned}
\frac{\partial}{\partial a_j} K(A) &= \frac{\partial}{\partial a_j} \frac{1}{2} \big(S_A(X) - \hat{y}\big)^2 \\
&= 2 \cdot \frac{1}{2} \big(S_A(X) - \hat{y}\big) \cdot \frac{\partial}{\partial a_j} \big(S_A(X) - \hat{y}\big) \\
&= \big(S_A(X) - \hat{y}\big) \cdot \frac{\partial}{\partial a_j} \Big(\sum_{i=0}^{n} a_i x_i - \hat{y}\Big) \\
&= \big(S_A(X) - \hat{y}\big) x_j
\end{aligned} \tag{2.14}$$

Tek bir örnek için (i=1) formulün son hali denklem 2.15'de verilmiştir. Denklem 2.14'deki eksi ifade denklem 2.15'deki ikinci ifadenin içerisine dağıtılmıştır. (Hatırlatma X vektör formundadır)

$$a_j := a_j + \eta \big(\hat{y} - S_A(X)\big) x_j \tag{2.15}$$

Eğer birden fazla örnek varsa bu durumda yeni denklemimiz

```
for j=0:n
```

$$a_j(2) := a_j(1) + \frac{1}{m} \eta \sum_{i=1}^{m} \big(\hat{y}^{(i)} - S_A(X^{(i)})\big) x_j^{(i)} \tag{2.16}$$

```
end
```

Denklem 2.16'deki i örnek sayısını, j girdi elemanını (indisi) belirtmektedir. Denklem 2.16'deki ağırlık değerleri yakınsayana kadar güncelleme işlemi tekrarlanır. Bu şekildeki

güncelleme işlemine yığın-gradyan inişi denmektedir. Her adımda eğitim seti içerisindeki bütün örneklere bakarak ağırlık değeri güncellenir. Bu işlem her bir j değeri için (j=1,2,3,..) yapılır. $a_j(1)$ birinci iterasyondan elde edilen ağırlık değerini, $a_j(2)$ ise ikinci iterasyondan elde edilen ağırlık değerini göstermektedir.

Basit bir örnek yapalım. Üç girdili 3 adet örnek için verilen etiket değerlerini sağlayacak ağırlık değerlerini gradyan iniş yöntemi ile bulalım. Diyelim ki kan değerlerimizdeki 3 özelliğe bakarak kanda bulunan şeker miktarını bulabilen bir regresyon modeli eğittiğimizi varsayalım. Buradaki değerler tamamen rastgele olarak verilmiştir. Sadece algoritmanın çalışma mantığının anlatılması için bir kurgu oluşturulmuştur. Örneğin,

$$X^{(1)} = [3,4,5]^T,\ \hat{y}^{(1)} = 53$$

$$X^{(2)} = [7,10,4]^T,\ \hat{y}^{(2)} = 120$$

$$X^{(3)} = [1,2,3]^T,\ \hat{y}^{(3)} = 25$$

olsun. Burada x bizim girdilerimiz y bizim etiket değerlerimizdir. Bu değerleri gerçekleyecek lineer denklemimizi ise

$$S_A(X) = a_1x_1 + a_2x_2 + a_3x_3$$

olarak belirleyelim. İlk olarak her bir ağırlık değeri rastgele olarak atanır. $a_1 = 5$, $a_2 = 6$, $a_3 = 2$ olsun. Bu adımdan sonra yapılması gereken sadece denklem 2.16'i uygulamaktır.

$$a_j := a_j + \eta \frac{1}{m} \sum_{i=1}^{m} \left(\hat{y}^{(i)} - S_A(X^{(i)})\right) x_j^{(i)}$$

$$a_1 = a_1 + \eta \left((\hat{y}^{(1)} - S_A(X^{(1)}))x_1^{(1)} + (\hat{y}^{(2)} - S_A(X^{(2)}))x_1^{(2)} + (\hat{y}^{(3)} - S_A(X^{(3)}))x_1^{(3)} \right)$$

$$a_1 = 5 + \frac{1}{300}[(53-49)\cdot 3 + (120-103)\cdot 7 + (25-23)\cdot 1] = 5,44$$

$$a_2 = 6 + \frac{1}{300}[4\cdot 4 + 17\cdot 10 + 2\cdot 2] = 6,63$$

$$a_3 = 2 + \frac{1}{300}[4\cdot 5 + 17\cdot 4 + 2\cdot 3] = 2,92$$

Elde edilen yeni ağırlıklar ile tekrar hesaplama yapıldığında

$$S_A(X_1) = (5,44 \cdot 3) + (6,63 \cdot 4) + (2,3 \cdot 5) = 54,34$$
$$S_A(X_2) = (5,44 \cdot 7) + (6,63 \cdot 10) + (2,3 \cdot 4) = 113,58$$
$$S_A(X_3) = (5,44 \cdot 1) + (6,63 \cdot 2) + (2,3 \cdot 3) = 25,6$$

Görüldüğü gibi yeni etiket değeri ile güncellenen ağırlık değerleri arasındaki fark tek bir güncelleme adımından sonra bile azalmıştır. Bu güncelleme adımını tekrar tekrar yaparak en doğru ağırlık değerleri bulunur.

Yığın gradyan inişi yöntemi tek bir adımda eğitim seti içerisindeki bütün örneklere bakmaktadır. Bu işlem m değeri büyük olduğunda oldukça fazla işlem maliyeti getirmektedir. Dolayısıyla bu duruma çözüm olarak stokastik-gradyan inişi yöntemi önerilmiştir. Bu yöntemde her seferinde tek bir örneğe bakarak ağırlıklar güncellenir. Her bir örneğe bakıldıktan sonra bir adım gerçekleşmiş olur. m örnek toplam örnek sayısı olmak üzere, tek bir tur için şu şekilde formulize edebiliriz:

```
for i=1:m
    for j=0:n
```

$$a_j(2) := a_j(1) + \eta\left(\hat{y}^{(i)} - S_A(X^{(i)})\right)x_j^{(i)} \tag{2.17}$$

```
    end
end
```

Bu işlem, ağırlıklar yakınsanana kadar devam eder. Aynı örneği stokastik Gradyan inişi yöntemi ile yaptığımız zaman ilk önce $X^{(1)}$ için a_1,a_2 ve a_3 ağırlıkları hesaplanır. Daha sonra $X^{(2)}$ için a_1, a_2 ve a_3 ağırlıkları hesaplanarak bütün $X^{(i)}$ girdileri için bu işlem devam eder. Bütün girdilerin işleme sokularak ağırların belirlenmesi ile bir tur tamamlanmış olur bu işlem ağımız yakınsayıncaya kadar devam eder. Örnek güncelleme adımları aşağıdaki gibidir.

Birinci tur: $X^{(1)}$ için

$$a_1 = 5 + \frac{1}{100}(53 - 49) \cdot 3 = 5,12$$
$$a_2 = 6 + \frac{1}{100}(4) \cdot 4 = 6,16$$

$$a_3 = 2 + \frac{1}{100}(4) \cdot 5 = 2,2$$

$$5,12 \cdot 3 + 6,16 \cdot 4 + 2,2 \cdot 5 = 51$$

$$5,12 \cdot 7 + 6,16 \cdot 10 + 2,2 \cdot 4 = 106,24$$

$$5,12 \cdot 1 + 6,16 \cdot 2 + 2,2 \cdot 3 = 24,04$$

$X^{(2)}$ için

$$a_1 = 5,12 + \frac{1}{100}(13,75) \cdot 7 = 6,08$$

$$a_2 = 6,16 + \frac{1}{100}(13,75) \cdot 10 = 7,53$$

$$a_3 = 2,2 + \frac{1}{100}(13,75) \cdot 4 = 2,75$$

$$6,08 \cdot 3 + 7,53 \cdot 4 + 2,75 \cdot 5 = 62,11$$

$$6,08 \cdot 7 + 7,53 \cdot 10 + 2,75 \cdot 4 = 128,86$$

$$6,08 \cdot 1 + 7,53 \cdot 2 + 2,75 \cdot 3 = 29,39$$

$X^{(3)}$ için

$$a_1 = 6,08 + \frac{1}{100}(-4,4) \cdot 1 = 6,04$$

$$a_2 = 7,53 + \frac{1}{100}(-4,4) \cdot 2 = 7,45$$

$$a_3 = 2,75 + \frac{1}{100}(-4,4) \cdot 3 = 2,62$$

$$6,04 \cdot 3 + 7,45 \cdot 4 + 2,62 \cdot 5 = 61,02$$

$$6,04 \cdot 7 + 7,45 \cdot 10 + 2,62 \cdot 4 = 127,26$$

$$6,04 \cdot 1 + 7,45 \cdot 2 + 2,62 \cdot 3 = 28,8$$

İkinci tur: $X^{(1)}$ için

$$a_1 = 6,04 + \frac{1}{100}(-8,02)\cdot 3 = 5,8$$
$$a_2 = 7,45 + \frac{1}{100}(-8,02)\cdot 4 = 7,13$$
$$a_3 = 2,62 + \frac{1}{100}(-8,02)\cdot 5 = 2,22$$

$$5,8\cdot 3 + 7,13\cdot 4 + 2,22\cdot 5 = 57,02$$
$$5,8\cdot 7 + 7,13\cdot 10 + 2,22\cdot 4 = 120,78$$
$$5,8\cdot 1 + 7,13\cdot 2 + 2,22\cdot 3 = 26,66$$

$X^{(2)}$ için

$$a_1 = 5,8 + \frac{1}{100}(-0,78)\cdot 7 = 5,75$$
$$a_2 = 7,13 + \frac{1}{100}(-0,78)\cdot 10 = 7,05$$
$$a_3 = 2,22 + \frac{1}{100}(-0,78)\cdot 4 = 2,19$$

$$5,75\cdot 3 + 7,05\cdot 4 + 2,19\cdot 5 = 60,78$$
$$5,75\cdot 7 + 7,05\cdot 10 + 2,19\cdot 4 = 119,51$$
$$5,75\cdot 1 + 7,05\cdot 2 + 2,19\cdot 3 = 26,42$$

$X^{(3)}$ için

$$a_1 = 5,75 + \frac{1}{100}(-1,42)\cdot 1 = 5,74$$
$$a_2 = 7,05 + \frac{1}{100}(-1,42)\cdot 2 = 7,02$$
$$a_3 = 2,19 + \frac{1}{100}(-1,42)\cdot 3 = 2,15$$

$$5,74\cdot 3 + 7,13\cdot 4 + 2,22\cdot 5 = 56,05$$

$$5,74 \cdot 7 + 7,13 \cdot 10 + 2,22 \cdot 4 = 118,98$$

$$5,74 \cdot 1 + 7,13 \cdot 2 + 2,22 \cdot 3 = 26,23$$

Olması gereken değerlerin $[53, 120, 25]$ olduğunu düşünürsek algoritmanın 2 turda bile istenen değerlere yakınsadığı görülmüştür. Bu yöntem oldukça başarılı olmasına karşın işlem maliyetini daha da azaltmak ve başarıyı biraz daha arttırmak adına Mini-yığın gradyan iniş yöntemi önerilmiştir. Bu yöntemde tek seferde bir tek örnek almak yerine küçük bir örnek seti alınarak işleme tabi tutulur. Denklem 2.18'de mini-yığın gradyan iniş yöntemi verilmiştir. Yukarıda anlatılan gradyan iniş yönteminin 3 farklı versiyonu kısaca aşağıdaki gibi özetlenebilir.

```
for i=1:b:m
    for j=0:n
```

$$a_j(2) := a_j(1) + \eta \sum_{k=i}^{i+b} \left(\hat{y}^k - S_A(X^{(k)})\right) x_j^{(k)} \tag{2.18}$$

```
    end
end
```

Denklem 2.18'deki b, mini-yığın boyutunu, m ise toplam örnek sayısını belirtmektedir.

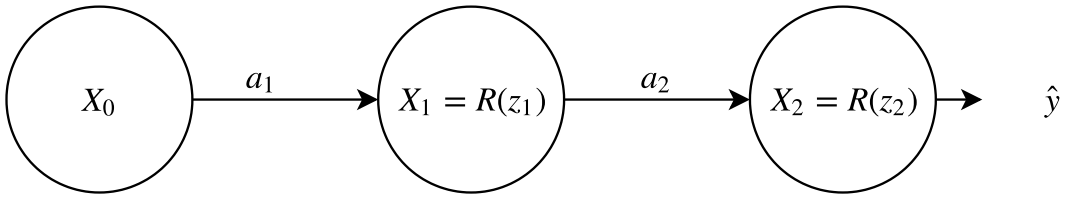
Geriye Yayılım Algoritması: Şu ana kadar tek katmanlı yapay sinir ağlarından bahsettik. Fakat YSA, insan beyninin sinir sistemi yapısından ilham alması bakımından tek katman yerine çok katmanlı sistemlerin kullanılması fikrini de gündeme getirmüştir. Benzer şekilde yapay sinir ağı çok katmanlı tasarlanmaya çalışılsa da güncelleme işleminde asıl denklemin türevinin alınması gerekmektedir. Geriye yayılım algoritmasının çok katmanlı sistemlerde işlem maliyetinin fazla olmasından ötürü, çok katmanlı sistemler için bu algoritma kullanılamamıştır.

Geriye yayılım algoritması yapay sinir ağı gibi katmanlı yapılardaki birbirlerine bağlı olan düğümlerin zincirleme bir şekilde hesaplanmasında kullanılan, türevleri hesaplama yöntemidir. YSA gibi birbirine bağlı yapılarda oldukça etkilidir. Geriye yayılım algoritması

bir öğrenme metodu değildir. Aksine öğrenme yönteminde sıkça kullanılan, öğrenme adımındaki işlemleri hesaplama yöntemidir. Temel olarak türevlerdeki zincir kuralının bir uygulaması olan geriye yayılım algoritması, graf (graph) yapıdaki derin sistemlerde gerekli olan bütün kısmi türevleri doğrusal (lineer) zamanda hesaplama kolaylığını sunmaktadır. Çünkü bütün denklemlerin türevlerinin hesaplanması, ağaç yapısının derinliği sebebiyle üstel boyutta zaman alacaktır. Bu yöntem bu noktada kolaylık sağladığından ötürü derin ağlarda güncelleme algoritması, geriye yayılım algoritmasını kullanarak doğrusal zamanda sonuç verebilmektedirler. Bu konuda sıkça yapılan bir yanlış bulunmaktadır. İnsanlar bir modeli eğittiği zaman geriye yayılım algoritması ile eğitilmiştir ifadesini kullanmaktadır. Ancak doğrusu, türev hesaplama tekniği olan geriye yayılım algoritmasını kullanarak, stokastik gradyan inişi güncelleme algoritması ile eğitilmiştir olmalıdır.

Zincir kuralı türev hesaplamada bize kolaylık sağlayan bir yöntemdir. Diyelim ki skor fonksiyonumuz $S_A(X) = A(B(C(X)))$ şeklinde olsun. Buradaki A, B ve C fonksiyonularını her bir katmanda bulunan aktivasyon fonksiyonu olarak düşünebiliriz. Zincir kuralını kullanarak $S_A(X)$ fonksiyonunun X'e göre türevini kolaylıkla, $S'_A(X) = S'_A(A) \cdot A'(B) \cdot B'(C) \cdot C'(X)$ yazarak bulabiliriz.

Bir giriş, bir orta katman ve bir çıkıştan oluşan basit bir YSA modeli üzerinde geriye yayılım algoritmasını uygulayalım. Şekil 2.17'de örnek uygulamanın modeli gösterilmiştir.



Şekil 2.17. 3 katmanlı sinir ağı

Şekil 2.17'deki X_0 YSA'nın girdi değeri, $\hat{y}$ YSA'nın ürettiği değer, a_i ağırlıklar ve R fonksiyonu ReLU aktivasyon fonksiyonunu temsil etmektedir. Birinci düğümden ikinci düğüme giderken hesaplanan değer z_1 ve ikinci düğümden üçüncü düğüme giderken

hesaplanan değer z_2 olarak gösterilmiştir. ReLU aktivasyon fonksiyonunun sonuçları bir sonraki düğüm için girdi olacağı için X_i olarak belirtilmiştir. Bu basit ağı eğitirken gradyan iniş güncelleme algoritmasını ve karesel kayıp fonksiyonunu kullanalım. Gradyan iniş algoritmasında kayıp fonksiyonunun türevinin ağırlıklara göre alınması gerekmektedir. ($a_j := a_j - \eta \frac{\partial}{\partial a_j} K(A)$, bkz 2.13). Türev alınırken zincir kuralı kullanılarak kısmi türevlerin çarpımı şeklinde ifade edebiliriz. 2.17'de örnek için ele alalım. a_2 ağırlığı için türev;

$$\begin{aligned} K'(a_2) &= K'(\hat{y}) \cdot \hat{y}'(z_2) \cdot z_2'(a_2) \\ &= (y - \hat{y}) \cdot R'(z_2) \cdot X_1 \end{aligned} \tag{2.19}$$

Denklem 2.19, sözlü olarak ifade edilecek olursa, hatanın a_2 ağırlığına göre türevi eşittir, hatanın çıktıya göre türevi çarpı, çıktının hesaplanan z_2 değerine göre türevi çarpı, z_2'nin a_2 ağırlığına göre türevi olarak söylenebilir. a_1 ağırlığı için türev:

$$\begin{aligned} K'(a_1) &= K'(\hat{y}) \cdot \hat{y}'(z_2) \cdot z_2'(X_1) \cdot X_1'(z_1) \cdot z_1'(a_1) \\ &= (y - \hat{y}) \cdot R'(z_2) \cdot a_2 \cdot R'(z_1) \cdot X_0 \end{aligned} \tag{2.20}$$

Denklem 2.19 ve 2.20'den görüldüğü üzere ortak alınması gereken türevler bulunmaktadır. Yani şöyle ki a_2 ağırlık değeri için alınan kısmi türevler, a_1 için alınan kısmi türevlerin bir kısmını oluşturmaktadır. Bu işlem ağ derinleştikçe bir kurala bağlı olarak devam etmektedir. Bu kural denklem 2.21'de görülmektedir.

$$\begin{aligned} K(a_3) &= (y - \hat{y}) \cdot R'(z_3) \cdot X_2 \\ K(a_2) &= (y - \hat{y}) \cdot R'(z_3) \cdot a_3 \cdot R'(z_2) \cdot X_1 \\ K(a_1) &= (y - \hat{y}) \cdot R'(z_3) \cdot a_3 \cdot R'(z_2) \cdot a_2 \cdot R'(z_1) \cdot X_0 \end{aligned} \tag{2.21}$$

Dolayısıyla türev alma işlemleri yapılırken hesaplanan kısmi türevler kendisinden bir önceki katmana aktarılır. Böylelikle işlem maliyetinden büyük ölçüde kazanç sağlanır. Zincir kuralının ve hesaplanan kısmi türevlerin bir sonraki hesaplama için hafızada tutulması geriye yayılım algoritmasını oluşturmaktadır. Bu yöntem ile derin ağları doğrusal zamanda eğitebilmek mümkündür.

Veri Önişleme: Yapay sinir ağları kullanılarak girdi ve etiketi olan bütün veriler için eğitim gerçekleştirilebilir. Ancak verilerin girdilerinin ve etiketinin olması demek YSA'nın

bu veri kümesini gerçekleyecek ağırlıkları bulabileceği anlamına gelmemektedir. Yani YSA yakınsamayabilir. Eğitim yapılırken asıl dikkat edilmesi gereken husus verilerin düzenli ve ayrılabilir olmasıdır. Model eğitilirken hassas parametreler bölümünde eğitim başarısını etkileyen durumlardan bahsedilecektir ancak bu parametrelere geçilmeden önce verilerin eğitime uygun hale getirilmesi eğitimin en önemli aşamasını oluşturmaktadır.

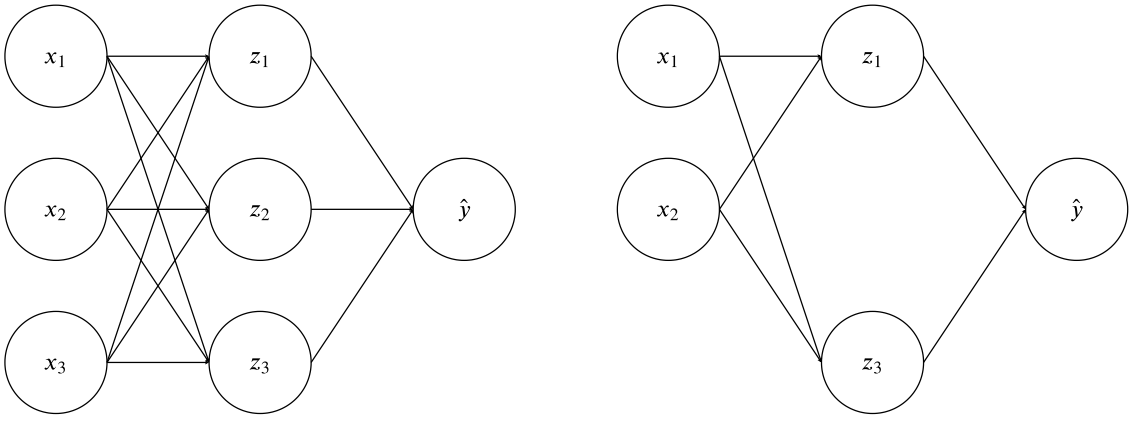
Veri önişleme yöntemlerinden en yaygın olarak kullanılanı, verilerin tamamının (özelliklerin, girdilerin) ortalamasını verilerin her birinden çıkarmaktır. Böylelikle yöntem, veri bulutunu orijin etrafına her boyutta merkezleyerek geometrik olarak yorumlayabilecektir. Bu dönüşümden sonra merkezi 0 olan veri kümesine sahip oluruz. İkinci olarak kullanılan önişleme yöntemi ise normalleme işlemidir. Birçok şekilde normalleme işlemi yapılabilmektedir. Normalleme işlemi her bir verinin yaklaşık olarak aynı boyutta olabilmesi için yapılan düzenlemedir. Örneğin bazı veriler için verilerin tamamı -1 ile 1 arasındaki değerlere dönüştürülür. Bazı durumlarda ise merkezleme işleminden sonra giriş verileri standart sapma değerlerine bölünür.

Ağırlıkların İlk Değerini Belirleme: Eğitim yapılmadan önce her bir ağırlık değerine bir ilk değer verilmesi gerektiği önceki bölümlerde bahsedilmişti. Ancak bu ilk değerlerin nasıl verileceği eğitimin başarısı açısından önem arz etmektedir. İlk olarak akla gelen her bir ağırlık değerine 0 değerini vermektir. Ancak bu yanlış bir yaklaşımdır. Çünkü eğer ağ içerisindeki her bir düğüm aynı çıktıyı üretirse, güncelleme adımlarındaki türev işlemleri de aynı sonucu üretecektir ve dolayısıyla aynı düğümler aynı ağırlık değerlerine sahip olacaktır. Yani Ağ içerisinde farklı farklı ağırlık değerleri olması gerekirken, aynı ağırlık değeri bütün ağ genelinde olacaktır.

Her bir ağırlık değerine sıfır değeri vermek yerine sıfıra çok yakın ancak eşit dağılımlı veriler verirsek her bir ağırlık değerinin aynı değil farklı değerler almasını sağlayabiliriz. En çok tercih edilen ilk ağırlık verme yönteminde ilk değerler, sıfır ortalamalı ve standart sapması 1 olan Gauss dağılımından elde edilen rastgele değerlerin 0,01 gibi sıfıra yakın

bir ek değer ile çarpılmasıyla elde edilir. Bu formül ile her bir düğüm rastgele sıfıra yakın değerler alarak, girdi uzayında farklı yönlere yönlendirilirler. İlk ağırlık değerlerini verirken eşit dağılımlı küçük sayılar da vermek mümkündür. Ancak uygulamada, bu yaklaşımın sistem başarısına çok düşük bir etkisi olmaktadır.

Seyreltme: YSA'da eğitim yapılırken belirli bir veri kümesi için ağırlıklar güncellenir, bu ağırlıklar ile de eğitim esnasında kullanılmayan bir veri setinde test yapılır. Test setinde elde edilen başarı sistemin başarısını vermektedir. Ancak mevcut ağırlıklar sadece verilen veri kümesi için belirlendiğinden bazı durumlarda aşırı öğrenme söz konusu olabilmektedir. Aşırı öğrenme ağın verilen veri kümesini yüzde yüz başarıyla öğrenmesi ancak başka bir veri kümesinde ise çalışmaması ya da düşük bir başarıyla çalışması demektir. Örneğin insan ayırt edebilen bir YSA eğitimi yapıldığını varsayalım. Girdi olarak elinde elma olan bir insan resmi olsun. YSA'nın özellik olarak insana ait el kol bölgelerine gelen piksel değerleri için ağırlık değerlerini yüksek olarak belirlemesini bekleriz. Ancak aşırı öğrenme durumunda ağ, yalnızca elmanın bulunduğu bölgeye bakarak bu bölgede elma varsa resim içinde insan vardır diyebilir. İşte bu alakasız detay bilgilere bağlı olmasını önlemek için, yani aşırı öğrenmeyi önlemek amacıyla seyreltme (dropout) yöntemi önerilmiştir.



Şekil 2.18. Seyreltme yöntemi

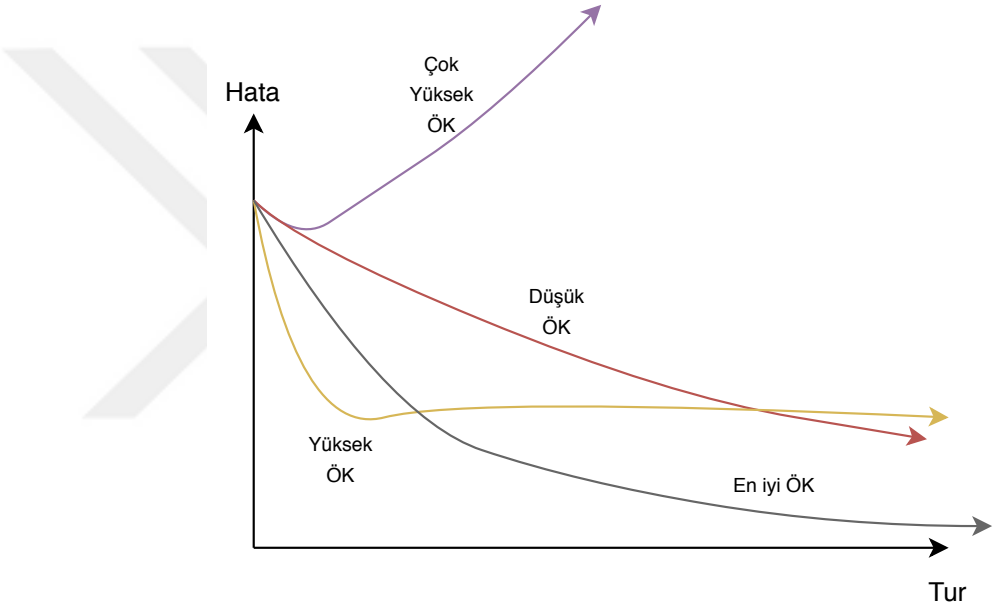
Bu yöntemde YSA içerisindeki bazı bağlantıların rastgele olarak hesaplama işlemine katılmaksızın sonuç değeri üretilir (Şekil 2.18). Böylelikle genel bir yargıya yalnızca tek bir koldan gelen ağırlık değerine bağlı kalmadan ulaşılabilmektir.

Hassas Parametreler: Yapay sinir ağlarında eğitim yapılırken belirlenmesi gereken bazı hassas parametreler bulunmaktadır. Bu değerlerin bütün ağlarda çalışabilecek ortak ve sabit bir değeri malesef bulunmamaktadır. Dolayısıyla bu değerler çoğunlukla deneme yanılma yöntemiyle belirlenir. Bir önceki bölümde sistemin başarısını, eğitimde kullanılmayan bir test setiyle belirlendiğini söyledik. Hassas parametlerin sistemi en yüksek performansta çalıştıracak şekilde belirlenebilmesi için aynı test setini kullanmak hatalı olacaktır. Çünkü elimizde test seti için en iyi parametreler seçilmiş olur ve biz sistemimizi yüksek performansta çalışıyor olarak zannederiz. Halbuki test işlemi yalnızca en son adımda sadece sistemin son başarısını ölçmek için yapılır. Bu noktada ağımızın hassas parametlerini belirleyebilmek için elimizdeki verileri 3 parçaya bölmeliyiz. Bunları, eğitim kümesi , test kümesi ve ön test kümesi olarak adlandırabiliriz. Ön test kümesi kullanılarak bu hassas parametre değerleriyle oynanır ve ağın ön test kümesinde yakalayabileceği en yüksek başarıyı yakalaması sağlanır. Son olarak da ağın başarısı test setiyle test edilerek ölçülür.

Ağın başarısını etkilen hassas parametleri şu şekilde sıralayabiliriz: Gizli katman ve her bir katmandaki düğüm sayısı, öğrenme katsayısı, seyreltme katsayısı, aktivasyon fonksiyonu çeşidi, ağırlık düşürme katsayısı (momentum) ve yığın boyutu. Ağımızdaki gizli katman sayısı ve düğüm sayısı yapmak istediğimiz uygulamanın karmaşıklığına ve elimizdeki verilere bağlıdır. Basit bir problem için basit bir mimari kullanmak hem sistemin çalışma hızı açısından hem de başarısı açısından daha mantıklıdır. Ağ derinleştikçe sistemin başarısı artar düşüncesi uygulamadan uygulamaya değiştiği için doğru bir yargı değildir. Bu yüzden ağın kaç adet düğüme ve gizli katmana sahip olması gerektiği hassas bir şekilde belirlenmelidir.

Öğrenme katsayısı ağırlıkların güncellenmesi aşamasında deneme yanılma yöntemiyle belirlenmektedir. Genellikle 0,1 ile 0,001 arasında değişen öğrenme katsayısı, girdilerin değerinin büyük veya küçük olmasına ya da yığın boyutuna bağlı olarak en uygun değeri değişebilmektedir. Eğer olması gerekenden büyük öğrenme katsayısı seçilirse, ağırlıklar yakınsamak yerine ıraksayacaktır. Eğer çok küçük seçilirse yakınsama işlemi hem çok yavaş olacaktır hem de lokal minimum noktasına takılma ihtimali doğacaktır.

Eğer ikisinin arasında ıraksamayacak ancak çok da yakınsamayacak bir değer seçilirse sistem eğitilir fakat performans olarak en iyi başarıyı sağlayamayacaktır. Dolayısıyla bir kaç tur deneme yanılma işleminden sonra yani ıraksamayacak değeri belirledikten sonra en uygun öğrenme katsayısı her bir tur için azaltarak belirlemek olacaktır. Şekil 2.19'de öğrenme katsayılarının davranışı gösterilmiştir. Yani 0,1'i yakınsayacak sınır değeri olduğunu varsayarsak 0,1'den başlayıp her bir turda azaltarak 0,001'e kadar (bu değer bize bağlı) seçmek en doğru yaklaşım olacaktır. Bu sayede hem lokal minimum noktalarına takılmamış hem de ağı başarılı kılacak en hassas ağırlık değerlerine sahip olunacaktır.



Şekil 2.19. Öğrenme katsayısı davranışı (ÖK: Öğrenme Katsayısı)

Hassas parametrelerden seyreltme katsayısı genellikle 0,5 olarak seçilir. Bu değerden daha fazla seçilmesi demek hesaplamaya girecek olan düğüm sayısının azalması demektir. Bu da faydadan çok zarar getirecektir. Bu yüzden 0,5 değerinden başlayarak sıfıra doğru denenebilir. Diğer hassas parametrelerden ağırlık düşürme katsayısı, yığın boyutu ve aktivasyon fonksiyonunun çeşidini belirlemek de çoğunlukla deneme yoluyla belirlendiği için kesin olarak seçilememektedir.

2.2 Evrişimsel Sinir Ağları (ESA)

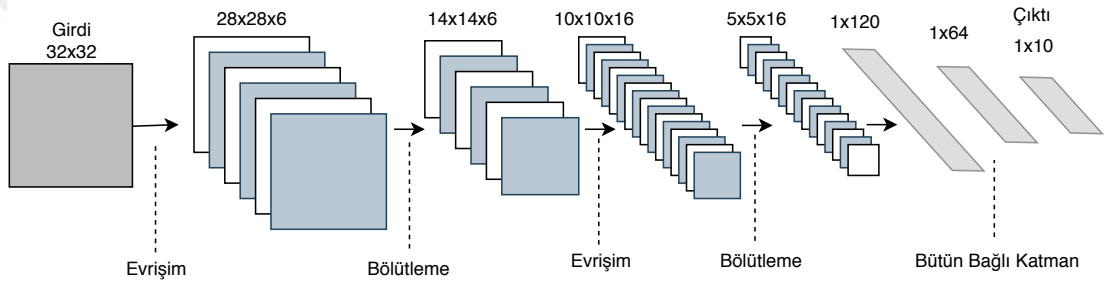
Evrişimsel sinir ağı (ESA) yapay sinir ağlarından türetilen girdi olarak resimlerin piksellerini aldığını varsayan, görüntü sınıflandırma için tasarlanmış bir yaklaşımdır. Bu yaklaşımda YSA'daki gibi kayıp fonksiyonu, türevlenebilir skor fonksiyonu ve öğrenilebilen ağırlık değerleri bulunmaktadır. Bu yaklaşım görüntü sınıflandırma problaminde standart YSA'ya göre daha başarılı olmasından ötürü daha çok tercih edilmektedir.

Genel Yapı: Sıradan yapay sinir ağında girdi olarak bir vektör verilir, bu girdiye dönüşüm (çarpım) uygulanarak gizli katmana aktarılır. Benzer dönüşümle gizli katmanlar arasında ilerledikten sonra sınıflandırma katmanına gelerek sonucu üretir. YSA'daki her bir düğüm birbirine bağlıdır. YSA'daki girdinin görüntü olduğunu düşünürsek her bir düğüm birbirine bağlı olduğu için küçük bir görüntüde bile oldukça fazla parametre çarpımıyla karşı karşıya kalınacaktır. Bu durum hem işlem maliyetini arttırır hem de aşırı öğrenmeye sebebiyet verebilmektedir. Evrişimsel sinir bu noktada bize büyük bir avantaj sunmaktadır. Evrişimsel sinir ağında her bir katman YSA'dakinden farklı olarak 3 boyutlu olarak tanımlanmıştır.

Her bir katmandaki düğümler YSA'dan farklı olarak kendisinden önceki katmanın sadece küçük bir kısmına bağlanmıştır. Bu sayede hesaplanması gereken parametre sayısı düşmektedir. Her bir katman 3 boyutlu girdi alıp 3 boyutlu çıktı üretmektedir. Ek olarak ESA çıktı katmanı olarak, içerisinde sınıf değerleri olan bir vektör üretecektir (sınıflandırma problemi için).

ESA'da Kullanılan Katmanlar: Basit bir ESA modeli bir takım katmanlardan oluşmaktadır. Her bir katman kendisinden bir önceki katmanın ürettiği çıktı değerini girdi olarak alır. ESA modelini 3 ana katmana ayırabiliriz, bunlar: evrişim katmanı, örnekleme katmanı ve bütün bağlı katmandır. Bu üç katman ESA modelinin mimarisi oluşturmaktadır. Örnek bir ESA mimarisi verelim, yapımız Girdi-Evrişim-ReLU-Örnekleme-Bütün bağlı olmak üzere 5 katmandan oluşsun. $24 \times 24 \times 3$ boyutunda 3 kanaldan oluşan (Kırmızı,

Yeşil, Mavi) resimleri düşünelim. Bu durumda girdimiz eni 24, boyu 24 ve derinliği 3 olan bir matris olacaktır. Evrişim katmanı, girdinin lokal bölgesine bağlı olan düğümlerin çıktısını hesaplar. Her bir işlem düğümlerin ağırlıkları ve girdinin küçük bir bölgesinin değerlerinin noktasal çarpımıyla gerçekleşir. 10 tane filtre kullandığımızı varsayarsak evrişim katmanından sonra çıktımızın boyutu $24 \times 24 \times 10$ olur. ReLU katmanında yalnızca aktivasyon fonksiyonu uygulanır. Bu katmandan sonra boyut değişmez. Örnekleme katmanında boyutu düşürme işlemi yapılır. Çıktı olarak $12 \times 12 \times 10$ gibi yarıya inmiş veya daha da az hacime sahip çıktı elde edilir. Bütün bağlı katmanda ise sınıf sayısı kadar hacme sahip bir vektör üretilir. Sınıf sayımızın 5 olduğunu varsayarsak $1 \times 1 \times 5$ boyutunda vektör, her bir sınıf değerinin olası skor değerlerini barındırır.



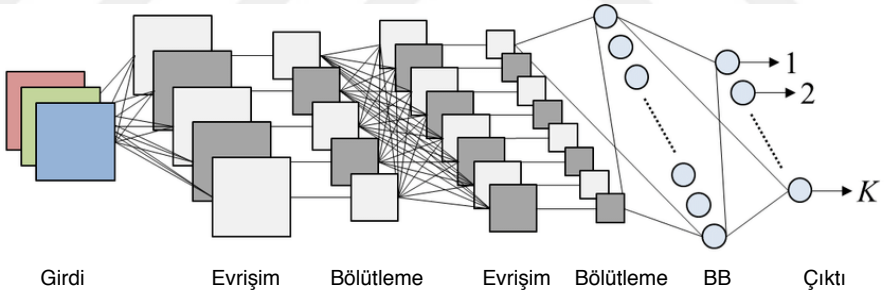
Şekil 2.20. LeNet mimarisi LeCun ve ark. (1998)

ESA yöntemiyle orijinal resim girdisi, katman katman ilerleyerek her bir sınıf için skor bilgisini üretmektedir. Parametreler YSA'dakine benzer şekilde, Gradyan iniş yöntemiyle güncellenir. LeNEt için LeCun ve ark. (1998) ESA mimarisi Şekil 2.20'de verilmiştir. Şekil 2.20'de görüldüğü gibi 32×32 boyutunda tek kanallı görüntüler, katmanlardan geçerek sınıf skoru üretmektedir. Her bir katman sonraki bölümlerde sırasıyla anlatılacaktır. Dolayısıyla sadece şu ana kadar anlatılan ESA mimarisinin akılda canlanması açısından Şekil 2.20 verilmiştir.

Evrişim Katmanı: Evrişim katmanındaki parametreler bir takım öğrenilebilir filtrelerden oluşmaktadır. Her bir filtre belirlenen en ve boya ve girdinin derinliğiyle aynı derinliğe sahiptir. RGB (Kırmızı, Yeşil, Mavi) kanallı bir görüntü girdisi için ilk evrişim katmanı $5 \times 5 \times 3$ olabilir. 5 değerleri farklı değerler alabilir ancak derinlik bilgisi olan 3

değeri çarpım işleminin gerçekleşebilmesi için girdi ile aynı boyutta olmak zorundadır. Skor fonksiyonunun hesaplanması kısmında her bir filtre girdi üzerinde kaydırılarak ilerler ve her bir filtre girdinin o anki pozisyonundaki değerlerle noktasal olarak çarpılır. Her bir filtre girdi matrisinin en ve boy düzleminde kaydırılarak çarpıldıkça, her bir özel pozisyonun çarpımını veren 2 boyutlu özellik haritası elde edilmiş olur.

ESA mimarisinde filtrelerin 2 veya 3 boyutlu olduğu daha önce bahsedilmişti. Her bir filtre resimler üzerindeki çeşitli özellikleri öğrenmektedir. Yani filtrenin bir tanesi belirli bir açıdaki kenar özelliğini veya belirli bir mozaikleme filtresini öğrenmektedir. Dolayısıyla ESA mimarisi içerisindeki filtreler resimlerin istenen sınıflandırma için özelliklerini çıkarmaktadırlar. Bu sebepten ötürü tek bir filtre kullanmak yerine çoklu filtre kullanılmaktadır. Bu sayede eğitilen ESA modeli, farklı özellikleri testpit edebilen filtreleri öğrenebilmektedir. Örneğin Şekil 2.20'de ilk evrişim katmanından sonra 6 adet özellik haritası üretilmiştir. Bu 6 haritayı üreten 6 adet filtre bulunmaktadır. Her bir filtre ise probleme bağlı olarak farklı bir özelliği öğrenmiştir. Şekil 2.21'de evrişim işlemindeki bölgesel çarpım işlemi görsel olarak verilmiştir.



Şekil 2.21. Örnek ESA mimarisi, BB:Bütün-bağlı

YSA'dan farklı olarak ESA mimarisinde bölgesel olarak bağlılık bulunmaktadır. Yani diyelim ki girdimizin boyutu $20 \times 20 \times 3$, filtre ise 5×5 boyutunda olsun. Toplamda her bir düğüm için $25 \cdot 3 = 75$ adet ağırlık değeri bulunmaktadır. Ya da $16 \times 16 \times 20$ girdi boyutu ve 3×3 filtre boyutu kullanılırsa (bu durumda derinlik 20 olduğundan ara katmanlardan birinde işlem yapılıyor demektir) $3 \cdot 3 \cdot 20 = 180$ adet ağırlık değeri her bir düğüm için bulunamktadır.

Çıktı Boyutu Düzenleme: Şu ana kadar evrişim katmanının temel olarak her bir düğümün girdiye nasıl bağlandığı anlatıldı. Ancak evrişim işleminden sonra olması gereken çıktı boyutunun ne olduğundan bahsedilmedi. Çıktı boyutunu belirleyen 3 parametre bulunmaktadır: derinlik, kaydırma miktarı, sıfır-ekleme.

Derinlik parametresi kullanmak istediğimiz filtre sayısıyla alakalıdır. Örneğin ilk evrişim katmanı girdi olarak resim piksellerini almaktadır. Bu katmandaki her bir filtre farklı bir özelliği öğrenmektedir. İlk filtre yuvarlak bölgelerin tespit edebilecek özelliği öğrenirken ikinci ve üçüncü filtreler ise renkleri, köşeleri veya kenar bölgeleri tespit edebilecek özelliğe bürünebilirler. Dolayısıyla filtre sayısını arttırmak daha fazla özellik filtresi öğrenmek demektir. Ancak belirli bir noktadan sonra aynı örüntüyü öğrenmeye başlayan filtreler olacaktır. Aynı zamanda filtre sayısını arttırmak işlem maliyetini arttıracaktır.

İkinci olarak kaydırma miktarının belirlenmesi gerekmektedir. Evrişim işlemi yapılırken resim üzerinde kaydırılarak noktasal çarpım yapılmaktadır. Dolayısıyla bu kaydırma miktarı hassas bir şekilde belirlenmelidir. Kaydırma miktarının 1 olması demek her seferinde 1 piksel kaydırılarak yeni bölgenin filtre ile işleme tabi tutulması demektir.

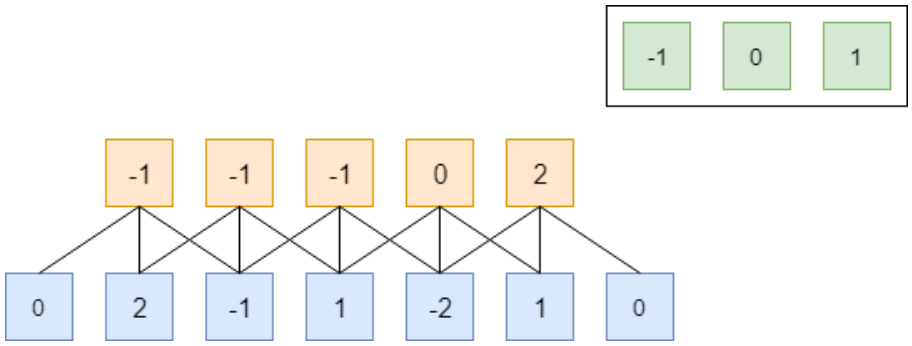
Son olarak bazı durumlarda sıfır ekleme yapılmaktadır. Genellikle girdi ile çıktının aynı boyutta olması istendiğinde üretilen özellik haritasının kenar bölgelerine girdi ile aynı boyutta olacak şekilde sıfır değeri eklenir.

Çıktı boyutu belirlenirken yukarıda verilen üç parametreye göre işlem yapılmaktadır. Girdi boyutu W, filtre boyutu F, kaydırma miktarı S ve sıfır ekleme P ile gösterilmek üzere, çıktı boyutu denklem 2.22'de verilmiştir.

$$C = (W - F + 2P)/S + 1 \tag{2.22}$$

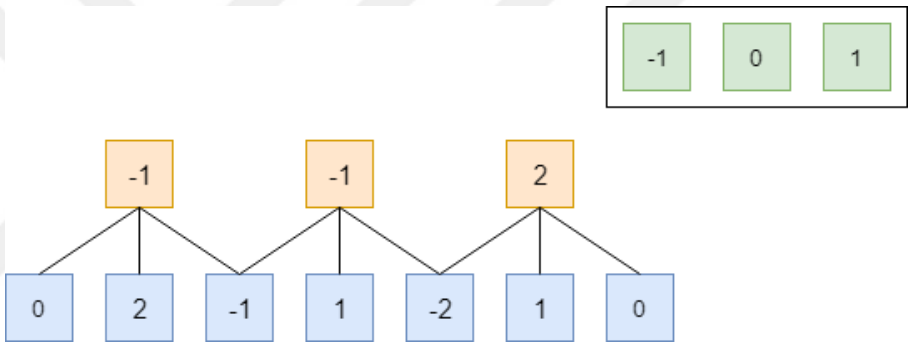
Örneğin 7×7 girdi, 3×3 filtre, 1 kaydırma ve 0 sıfır ekleme boyutu olursa üretilen çıktı boyutu 5×5 olacaktır. Şekil 2.22'de girdi boyutu 5×1, filtre boyutu 3×1, sıfır ekleme değeri 1 ve kaydırma miktarı 1 için evrişim işlemi gösterilmiştir. Yeşil ile gösterilen filtreyi temsil etmektedir. Filtre girdi (mavi) üzerinde 1 piksel miktarı kaydırılarak noktasal çarpım yapılır. Üretilen değer çıktının (kırmızı) ilgili yerine yazılır. Şekil 2.22'de görüldüğü gibi

formül uygulandığında çıktı (5-3+2)/1+1=5 olmaktadır.



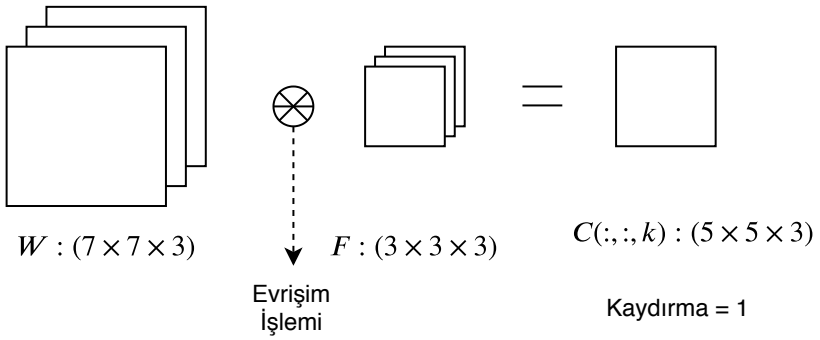
Şekil 2.22. Çıktı boyutu hesaplama örneği, kaydırma=1, sıfır ekleme=1

Sadece kaydırma miktarının 2 olduğu diğer parametrelerin aynı olduğunu düşünürsek Şekil 2.23'de görüldüğü gibi çıktı boyutu $(5-3+2)/2+1=3$ olacaktır.



Şekil 2.23. Çıktı boyutu hesaplama örneği, kaydırma=2, sıfır ekleme=1

Kaydırma miktarı eğer girdi ile uyumlu seçilmezse üretilen çıktı değeri tam sayı olmayacaktır. Yani diyelim ki girdi 10, filtre 3, sıfır ekleme 0 ve kaydırma miktarı 2 olsun. Bu durumda $(10-3+0)/2+1=4.5$ değeri üretilir. Mevcut derin öğrenme kütüphaneleri küsüratlı olan bu gibi durumlarda kendilerine göre hesaba katmama ya da sıfır ekleme gibi yöntemler uygulayarak tam sayı haline getirmektedir. Dolayısıyla istenmeyen durumlar ortaya çıkabilir. Bu yüzden yukarıda belirtilen 3 parametre, çıktıyı tam sayı yapacak sayılardan seçilmelidir. Temel olarak evrişim işlemi Şekil 2.24'deki gibi gösterilebilir. Şekil 2.24'de $7\times7\times3$ boyutundaki girdi W, $3\times3\times3$ boyutundaki filtre F ile evrişim işlemine sokularak C çıktısı elde ediliyor. k ise elde edilen filtre indeksini belirtmektedir.



Şekil 2.24. Evrişim işlemi örneği ve tensör boyutları

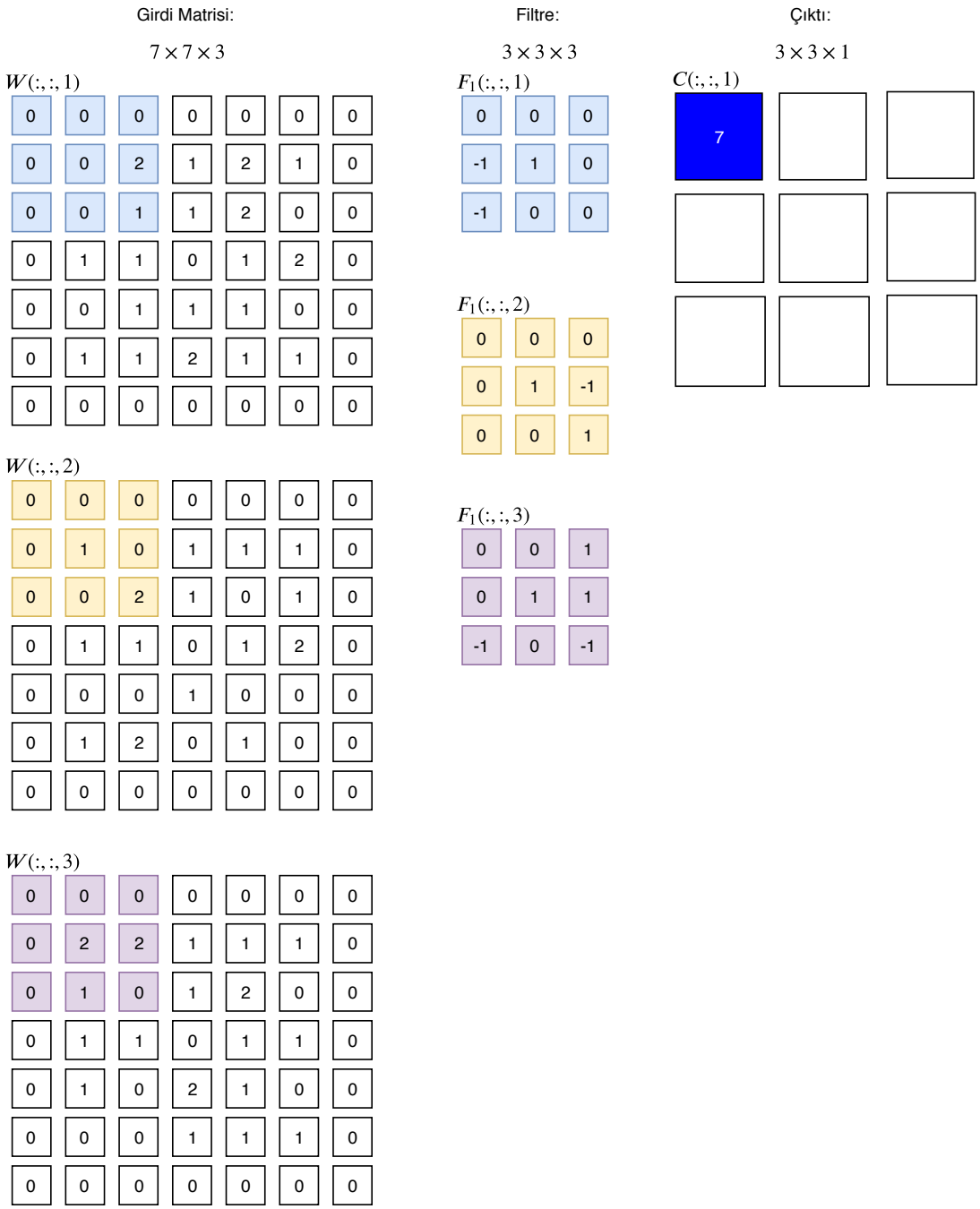
Evrişim İşlemi Örneği: Evrişim işleminin daha iyi anlaşılması açısından görsel olarak bir örnek verilmesi daha iyi olacaktır. Fakat 3 boyutlu bir örneği görselleştirmek zor olduğundan her bir boyut ayrı renkler ile ifade edilecektir.

Girdimizi boyutu $5 \times 5 \times 3$, filtre parametreleri ise K=2, F=3, S=2, P=1 şeklinde olsun. Buradaki K parametresi filtre sayısını göstermektedir. Dolayısıtla elimizde 2 adet 3×3'lük filtre, 2'li kaydırma miktarıyla uygulanacaktır. Çıktı boyutu (5 - 3 + 2)/2 + 1 = 3 olarak hesaplanır. Şekil 2.25'de girdi filtre ve çarpım işleminin ilk adımı gösterilmiştir. Şekil 2.25'de kenar değerlerin tamamının 0 olduğu görülüyor. Sıfır ekleme değerinin değerinin 1 olmasından ötürü girdi etradına 1 katman olacak şekilde sıfır eklenmiştir.

Şekil 2.25'de aynı renkteki matrisler birbirleriyle eleman bazlı olarak çarpıldıktan sonra toplanır. Yani adım 1'deki mavi bölgelerin eleman bazlı çarpımı sıfır değerini üretmektedir. Kırmızı bölgelerin çarpımından 3 değeri elde edilir. Mor bölgeden ise 2+2=4 değeri elde edilir. Son olarak elde edilen bu 3 değer toplarak koyu mavi olarak belirtilen bölgeye yazılır. Yazma işlemiyle birlikte ilk lokal evrişim işlemi yapılmış olur.

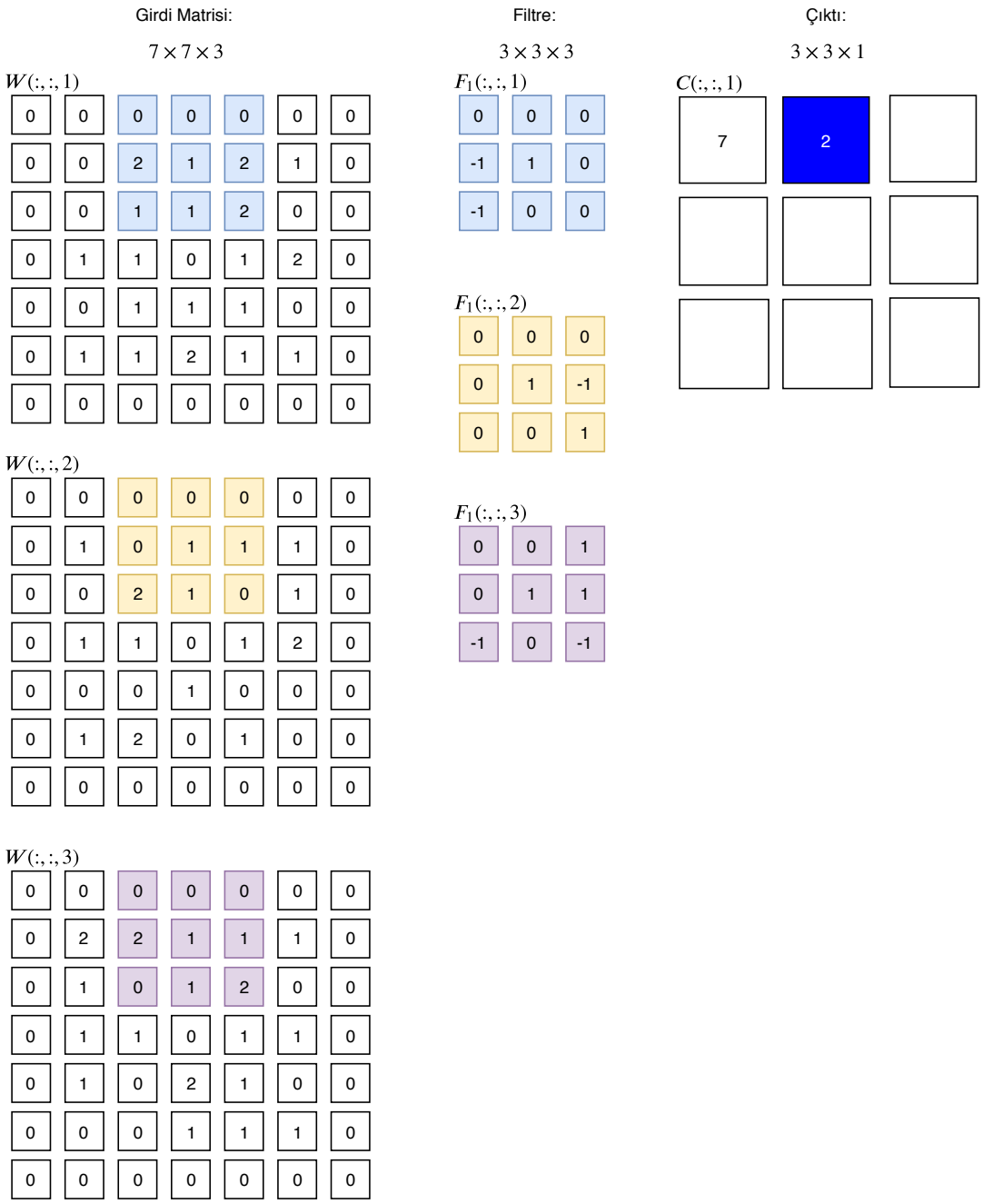
Şekil 2.26'de evrişim işlemi bir sonraki lokal bölge için aynı şekilde yapılır. Dikkat edilmesi gereken nokta kaydırma miktarı kadar atlama yapılarak çapılacak bölgenin doğru olarak belirlenmesidir. Bu örnekte kaydırma miktarı olarak 2 belirlendiği için 2 piksel yana kayarak evrişim işlemi gerçekleştirilmiştir.

Şekil 2.27'de ise ikinci filtre için üretilen çıktı haritasını görebiliriz. Kaydırma miktarı



Şekil 2.25. Evrişim işlemi örneği, adım 1

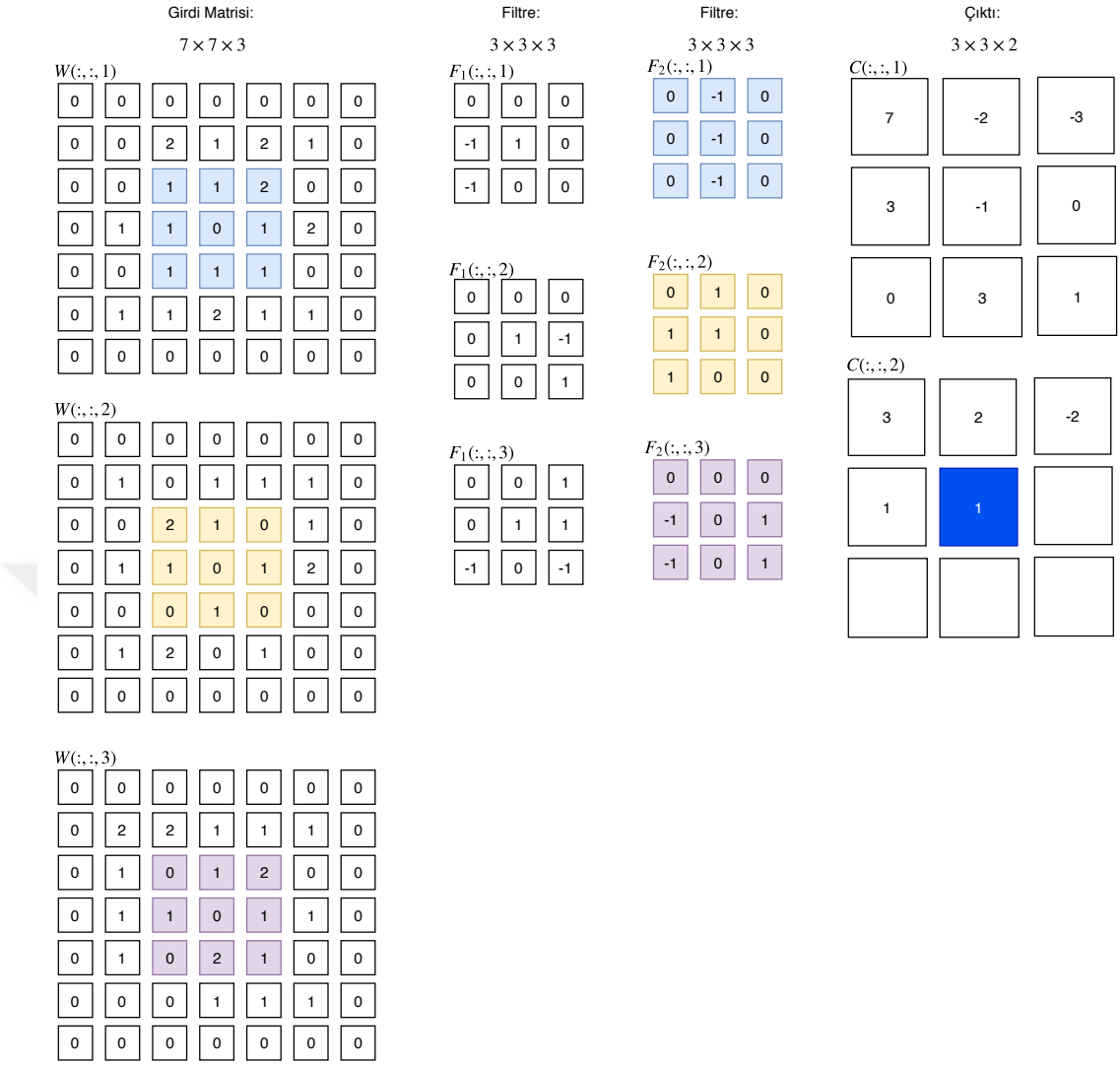
hem ene hem de boya doğru ilerlemektedir. Dolayısıyla aşağıya doğru kaydırılrken de 2 piksel kaydırılmıştır. Şekil 2.27'de de benzer şekilde aynı renkler birbirleriyle eleman bazlı olarak çarpılıp toplanır. Her renk için elde edilen değerlerin toplamı o lokal bölge



Şekil 2.26. Evrişim işlemi örneği, adım 2

için çıktıyı verir.

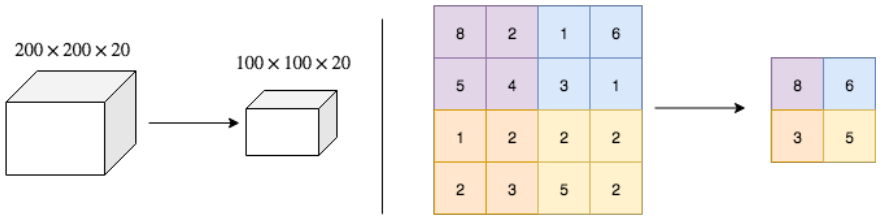
Örnekleme Katmanı: Her bir evrişim işleminden sonra çıktının boyutunun hesaplanması yukarıdaki bölümlerde anlatılmıştır. Her bir evrişim işleminden sonra boyut belirli



Şekil 2.27. Evrişim işlemi örneği, adım 3

oranda genellikle azalmaktadır. Ancak girdinin yüksek çözünürlüklü bir resim olduğunu düşünürsek son katmana yani bütün bağlı katmana gelebilmesi için çokça evrişim katmanına ihtiyaç duyulmaktadır. Bu durum ağın daha detay bilgileri bulabilmesini sağlayabilir. Ancak bu durum beraberinde oldukça fazla işlem maliyeti getirmektedir. Ayrıca parametre sayısının artması, aşırı öğrenmeye sebebiyet verebilir. Bu problemlere çözüm olarak örnekleme katmanı sunulmuştur. Örnekleme katmanında genellikle ara katmandaki girdilerin boyutu azaltılır. Boyut azalmasıyla hem işlem maliyeti azalır hem de aşırı öğrenmenin önüne geçilmiş olunur.

Boyut azaltma işlemi genellikle 2×2 filtre boyutuyla 2, kaydırma miktarını kullanır. Bu filtre ve kaydırma miktarı girdiyi tam olarak yarı boyutuna indirir (tek boyut için). Ek olarak örnekleme, girdinin her bir boyutunda yapıldığı için derinlik boyutu sabit kalır. Yani $200 \times 200 \times 3$ boyutundaki bir girdi, 2×2 lik filtre ve 2'lik kaydırma miktarıyla örnekleme işlemi gerçekleştirilirse, $100 \times 100 \times 3$ boyutunda değer üretilir.



Şekil 2.28. Maksimum bölütleme (Max-Pooling)

Örnekleme işleminin en büyük değeri alma ya da ortalama değeri alma gibi farklı versiyonları vardır. Çoğunlukla en büyük değeri alma fonksiyonu kullanılmaktadır. En büyük değeri alma işlemi filtre içerisine denk gelen girdi değerlerinin en büyüğünü alıp çıktı matrisinin içerisine yazmaktır. Örnek örnekleme işlemi Şekil 2.28'de verilmiştir. Şekil 2.28'de çizginin sağ tarafında 2×2'lik filtre ile 2'lik kaydırma miktarı kullanılarak en büyük değeri alarak örnekleme işlemi gerçekleştirilmiştir.

Bütün Bağlı Katman: Bütün bağlı katmanda her bir düğüm kendisinden bir önceki katmanda bulunan her bir düğüme bağlıdır. Bu kısım klasik yapay sinir ağıdır. Evrişim ve örnekleme işlemlerinden sonra çıktı olarak 3 boyutlu değil de 2 boyuta düştüğü anda artık klasik sinir ağı uygulanabilir. Örneğin $200 \times 200 \times 3$ piksellik bir girdi, evrişim ve örnekleme işlemlerinin ardından son olarak $1 \times 1 \times 2000$ (2000 sayısı örneğe göre değişebilir) boyutuna indirgenir. Bu kısımdan sonrası girdi olarak 2000 boyutlu vektör alan klasik bir yapay sinir ağıdır. YSA daha önce anlatıldığından burada tekrar anlatılmayacaktır.

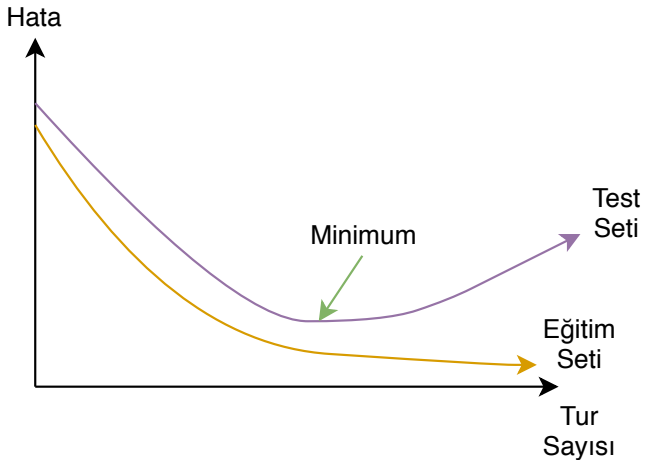
Veri Çoğaltma: Yeterli miktarda veri bulunmadığında ya da girdinin belirsiz olduğu durumlarda veri çoğaltma işlemi yapılabilir. Derin öğrenmede eğitilen ağ, ne kadar fazla veri ile eğitim yapılırsa olabilecek bütün ihtimalleri gördüğü için başarısı bir o

kadar artacaktır. Kedi köpek ayrımı yapabilen bir ağ eğittimizi düşünelim. Makineye ne kadar çok kedi ve köpek çeşidi gösterirsek, makinenin kararı o kadar doğru olacaktır. Olası bütün ihtimallerin gösterilmesi gerekir. Ancak bazı durumlarda yeterli miktarda veri bulunamamaktadır. Yeterli miktarda veri bulunmadığında mevcut resimler bazı operasyonlardan geçirilerek çoğaltılır. Örneğin az önceki misalde bahsedilen kedi resmi üzerine gürültü eklenirse bu yine kedi resmidir. Ancak orijinalden farklıdır. Ya da resmin x eksenine göre simetriği alındığında resim yine kedi resmidir. Dolayısıyla veri miktarı azlığında gürültü ekleme, ortalama filtre uygulama, simetriğini alma, vb. gibi yöntemlerle veriler çoğaltılabilir.

Veri çoğaltma işlemi aynı zamanda makinenin girdinin varyasyonlarını tanıyabilmesi için bir iyileştirme yöntemidir. Kedi-köpek ayıran bir ağa resmi yan çevirip girdi olarak verdiğimizde eğer eğitimde yan durmuş bir kedi yoksa tanıyamayacaktır. Ek olarak gürültü ekleme ya da ortalama filtre uygulayarak resmi ufak miktarda değiştirmek aşırı öğrenmeyi de engelleyecktir.

Erken Durdurma: Eğitim esnasında güncelleme işlemi için tekrarlanan turlar bazen ağı iyileştirmek yerine kötüleştirebilmektedir. Ağ eğitilirken amacımız doğrulama kümesinin en az hatayı verecek şekilde eğitim seti üzerinde değişiklikler yapmaktır. Yapılan değişiklikler neticesinde Şekil 2.29'deki gibi eğitim verisinin hatası düşmeye devam ederken ara-test setinin hatası bir noktadan sonra artmaya başlayabilir. Burada yapılması gereken işlem eğitimi erken durdurmaktır. Böylelikle ağ, olabilecek en iyi durumdayken güncelleme işlemi bitirilecektir.

Şekil 2.29'de turuncu ile gösterilen eğitim verilerinin hatası azalmaya devam ederken, eflatun ile gösterilen ara-test verilerinin hatası bir noktadan sonra artmaya başlamıştır. En iyi modeli elde edebilmek için eğitim yeşil ok ile gösterilen turda durdurularak en uygun ağırlık değerlerine sahip olunur.



Şekil 2.29. Erken durdurma

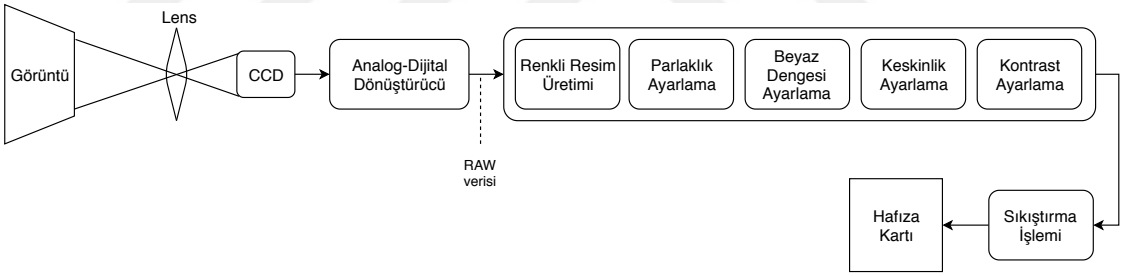
Yığın Normallame: Yığın normalleme (batch normalization), her yığın içindeki iç değişken değişimini azaltarak yakınsamayı hızlandırır. Yığın içindeki bireysel gözlemler oldukça farklıysa, gradyan güncellemeleri dalgalı olur ve yakınsaması daha uzun sürer. Yığın normallame katmanı aktivasyon fonksiyonundan gelen çıktıyı normalleyerek, ortalaması sıfir ve standart sapması 1 olan yeni bir yığın üretir. Bu normalleme işlemi, yığın'in ortalamasını yığının her bir elemanından çıkardıktan sonra yığının standart sapmasına bölünerek elde edilir.

3 MATERYAL VE YÖNTEM

3.1 PRNU Tabanlı Kaynak Cihaz Tanıma

Bu tez kapsamında PRNU yöntemi, önerilen yöntem içerisinde kullanıldığından PRNU yönteminin çalışma mantığı bu bölümde anlatılacaktır. Kaynak cihaz tanıma problemlerinde çekilen resmin kaynağını tespit edebilmek oldukça zordur. Adli bilişim alanında çalışanlar dahi bu konuda yüksek başarıya sahip bir yöntem henüz bulamamışlardır. Kaynak cihaz tanıma probleminin çözülebilmesi, resmin çekilme aşamalarının iyi anlaşılmasıyla başlar. İlk olarak resmin çekilme aşamaları bir sonraki alt başlıkta anlatılacaktır.

Resmin Oluşum Aşamaları: Sayısal bir resim kameradan çekilip bizim baktığımız ekrana gelinceye kadar bir takım işlemlerden geçmektedir. Resmin çekilme aşamaları Şekil 3.1'de gösterilmiştir.



Şekil 3.1. Sayısal resmin oluşum aşamaları

Çekilecek olan sahneden ışık ilk olarak kameranın lensine gelir. Lens yardımıyla odaklanan ışık renk filtresinden geçerek ışık miktarını ölçen sensörler üzerine düşer. BU sensörler ışık miktarını elektrik sinyaline dönüştürürler. Işık miktarının fazla olması demek daha yüksek elektrik sinyali olduğunu anlamına gelir. Ardından sayısal hale gelen görüntü mikroişlemci yardımıyla 3 kanallı görüntü haline çevirilir. Son olarak istenen formatlarda (JPEG, PNG, TIFF vb.) kayıt edilerek depolama biriminde saklanır.

Resim Üzerindeki Sensör Kaynaklı Sayısal Kalıntılar: Resmin çekilmesi esnasında yapılan her bir işlem resim üzerinde bir iz bırakmaktadır. PRNU yöntemi sensörlerden kaynaklı kalıntıları baz almaktadırlar. Işığın geldiği sensör yüzeyinde birbirine eş birçok sensör bulunmaktadır. Ancak fabrikasyon ürünü olan eş sensörler her ne kadar birbirinin aynısı olarak üretilse de malzemeden kaynaklı sebeplerden ötürü aslında birbirinin aynısı değillerdir. Yani 0-255 arasında bir sinyal değeri üretmesini beklerken sensörlerden birisi hiç bir zaman 0 değerini üretmeyebilir veya olması gerekenden hep 1 birim fazla üretebilir. Bahsedilen farklılıklar gözle görünemeyecek kadar küçük farklılıklardır. Yani her bir sensör ufak da olsa bir iz bırakmaktadırlar. Bazıları çok çok küçük bazıları ise küçük izler bırakabilir. Her bir sensörün kendine has çalışma karakteristiği bulunduğundan kamera içerisinde bulunan sensörler yüzeyi her bir cihaz için eşsiz özelliktedir. Dolayısıyla her bir cihazın kendine ait kamera parmak izini çıkartmak mümkündür.

Sensörlerden Parmak İzi Çıkarımı: Kameraya ait değişmeyen parmak izini tam olarak bulabilmek henüz mümkün değildir. Yalnızca bazı varsayımlar yapılarak parmak izi tahmini yapılabilmektedir. Kamera çıktısı I olmak üzere ışık denklemi 3.1'de verilmiştir.

$$I = I_0 + I_0 \cdot F + \phi \tag{3.1}$$

Denklem 3.1'deki I_0 gerçek sahneyi (gürültülerden bozukluklardan arınmış gözle görünen sahne), F sensörden kaynaklanan değişmeyen gürültü miktarını, ϕ ise dışarıdan gelen harici gürültüyü belirtmektedir. Amacımız F değişmeyen parmak izini tam olarak bulmak ya da en yakını tahmin etmektir. Bunun için ilk olarak ölçülen görüntü gürültüden temizlenir. Gürültüden temizleme işlemi için dalgacık gürültü çıkarıcısı kullanılır (Lukáš ve ark. 2006). Gürültüden temizlenmiş görüntü $\hat{I}$ = gürültü filtresi (I) şeklinde gösterilebilir. Gürültüden temizlenmiş görüntü ile temizlenmemiş arasındaki fark bize gürültüyü verir. W gürültü olmak üzere

$$W = I - \hat{I}$$

olarak hesaplanır. Gürültü denklemini, denklem 3.1'i kullanarak açarsak ve ek olarak $+IF - IF$ eklersek,

$$W = I - \hat{I} = I_0 + I_0 \cdot F + \phi - \hat{I} + IF - IF$$

$$W = IF + I_0 - \hat{I} + (I_0 - I)F + \phi$$

olarak düzenlenebilir. Burada $I_0 - \hat{I}$ ve $(I_0 - I)F$ terimleri ihmal edilebilir boyutta bir gürültü olduğundan gürültüyü denklem 3.2'deki gibi yazabiliriz.

$$W = IF + \Xi \tag{3.2}$$

Ξ gürültüsü, ϕ harici gürültüsü ile diğer iki terimin toplamından oluşmaktadır. IF PRNU sinyalinin enerjisi, Ξ gürültüsünden daha küçüktür. PRNU parmak izi tahmini yapılırken I yerine W kullanılır. Bunun sebebi W'nun, resmin içeriğine daha az bağımlı olmasıdır. Bu sayede daha iyi PRNU parmak izi tahmini yapılabilmektedir. F değerini elde edebilmek için Ξ gürültüsünden ve I teriminden kurtulunması gerekmektedir. Denklem 3.2'deki Ξ, her resim için farklı özellikte olacaktır. Dolayısıyla birden fazla aynı kameradan çekilmiş resim alınarak bu resimlerden gürültüler çıkarılır. Eğer kamera cihazına fiziki olarak erişim sağlanabiliniyor ise, bu cihazdan gökyüzü gibi daha eşit dağılımlı görüntüler çekilmesi tavsiye edilmektedir. Bu sayede elde edilecek olan parmak izinin kalitesi artmış olur. Kullanılması gereken resim sayısının ideal olarak en az 50 adet olması gerekmektedir. Bu aşamadan sonra parmak izi tahmini yapılırken en büyük olabilirlik kestirimi (MLE) yöntemi kullanılır. MLE'nin amacı gözlemlenen veriler için olasılığı maksimum yapan değerleri bulabilmektir.

Kısaca MLE yöntemi, belirlenen model için parametreleri bulmamızı sağlar. Bu yöntemde elimizdeki verileri seçilen dağılım üzerine en iyi şekilde oturtmayı deniyoruz. Bu sayede elimizdeki veriler ile bilinmeyen parametrelerin en iyi tahminini bulabiliyoruz. MLE yönteminde, öncelikle verilerin üzerinde oturtulacağı bir model belirlenmesi gerekmektedir. Parmak izi tahmininde seçilen model çeşidi Gauss dağılımıdır. Gauss dağılımında belirlenmesi gereken iki adet değişken bulunmaktadır. Bunlar ortalama (μ) ve varyans (σ^2) değeridir. Ayrıca her bir veri birbirinden bağımsız olduğundan MLE yöntemi bu

problem için uygulanabilir bir yöntemdir. MLE yöntemi ile Gauss dağılımı modeli için en genel denklem $x_i \sim N(\mu, \sigma^2)$ olarak yazılır. Buradaki N normal dağılımı belirtmektedir. Her bir μ değeri için olasılık değeri hesaplanır. Olasılık değerleri, P olasılık, x_i girdi olmak üzere denklem 3.3'de verilmiştir.

$$\mathrm{P}(x_1, x_2, .., x_T; \mu) \tag{3.3}$$

Denklem 3.3'in türevi alınarak sıfıra eşitlenirse maksimum olasığı veren tahmini μ değeri bulunmuş olur. Yani elimizdeki verileri Gauss dağılımına oturttuğumuzda olması gereken μ değerini yaklaşık olarak hesaplamış oluruz. A olasıkların toplamı olmak üzere

$$A = \prod_{i=1}^{T} \mathrm{P}(x_i; \mu)$$

$$A = \prod_{i=1}^{T} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(\frac{x_i-\mu}{\sigma})^2}$$

olarak yazılır. A'nın μ'ye göre türevi sıfıra eşitlenerek en uygun parametreler tahmin edilir. Ancak A'nın açılımında çarpım olarak çokça ifade olacağından bu ifadeyi hesaplamak matematiksel olarak zor olacaktır. Bunun için Log-MLE metodunu sunmuşlardır. Bu metodda tek fark fark denklemin her iki tarafının logaritmasını alarak işleme devam edilir. $L = \ln A$ olmak üzere $\frac{\partial A}{\partial \mu} = 0$ ifadesi $\frac{\partial L}{\partial \mu} = 0$ olarak tekrar yazılır. Bu işlem sonucu etkilememektedir. Denklemin logaritması alındığında, logaritmanın özelliği olarak çarpım olarak yazılan ifadeler toplama dönüşür. Türev adımları ise su şekildedir:

$$L = \sum_{i=1}^{T} \ln \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(\frac{x_i-\mu}{\sigma})^2}$$

$$L = \sum_{i=1}^{T} \ln \frac{1}{\sqrt{2\pi\sigma^2}} + \sum_{i=1}^{N} -\frac{1}{2}(\frac{x_i-\mu}{\sigma})^2$$

ilk terim içersinde μ bulunmadığından türevi sıfır olur. Böylelikle

$$\frac{\partial L}{\partial \mu} = \sum_{i=1}^{T} -(\frac{x_i-\mu}{\sigma})(-\frac{1}{\sigma})$$

$$\frac{\partial L}{\partial \mu} = \sum_{i=1}^{T} \frac{\mu - x_i}{\sigma^2} \tag{3.4}$$

olarak bulunur. Bu denklem sıfıra eşitlenerek σ ve μ parametreleri hesaplanır. Aynı kameradan çekilen N tane resim $I^{(1)}, I^{(2)}, ..., I^{(T)}$ ve bu görüntülerden elde edilen gürültüleri $W^{(1)}, W^{(2)}, ..., W^{(T)}$ olarak tanımlayalım. PRNU parmak izi hesaplanırken $k = 1, 2, ...T$ olmak üzere denklem 3.2 aşağıdaki gibi düzenlenerek F tek bırakılır.

$$\frac{W^{(k)}}{I^{(k)}} = F + \frac{\Xi}{I^{(k)}} \tag{3.5}$$

Denklem 3.5'de bulunan Ξ ifadesi Gauss dağılımına benzer özellik taşır. Dolayısıyla Ξ için ortalaması sıfır ve varyansı σ^2 olan normal dağılım

$$\Xi \sim N(0, \sigma^2)$$

şeklinde olur. F sabit bir sayı ve $I^{(k)}$ da Ξ'nin çarpanı olduğundan $\frac{W^{(k)}}{I^{(k)}}$ ifadesi de Gauss özelliği taşımaktadır. Dolayısıyla $\frac{W^{(k)}}{I^{(k)}}$ için Gauss dağılımı yaklaşık olarak

$$\frac{W^{(k)}}{I^{(k)}} \sim N\big(F, \frac{\sigma^2}{(I^{(k)})^2}\big)$$

şeklinde yazılır (Chen ve ark. 2008). Bu varsayım genel olarak doğrudur. Çünkü F sabit bir sayı olduğundan ortalamayı etkileyecektir. Ξ Gauss dağılımı göstermektedir ve $I^{(k)}$ ise çarpan olduğundan varyansı etkileyecektir. Denklem 3.4'deki x_i yerine $\frac{W^{(k)}}{I^{(k)}}$, μ yerine F ve σ^2 yerine $\frac{\sigma^2}{I^{(k)}}$ yazılarak sıfıra eşitlenirse istenen en iyi tahmini F değeri bulunabilir. Yeni denklemi sıfıra eşitlersek:

$$\sum_{k=1}^{T} \frac{F - \frac{W^{(k)}}{I^{(k)}}}{\frac{\sigma^2}{(I^{(k)})^2}} = 0$$

$$\sum_{k=1}^{T} \frac{\frac{W^{(k)}}{I^{(k)}}}{\frac{\sigma^2}{(I^{(k)})^2}} = \sum_{k=1}^{T} \frac{F}{\frac{\sigma^2}{(I^{(k)})^2}}$$

olarak yazılır. Son olarak tahmin edilen yaklaşık parmak izi denklem 3.6'deki gibi yazılır (Lukáš ve ark. 2006).

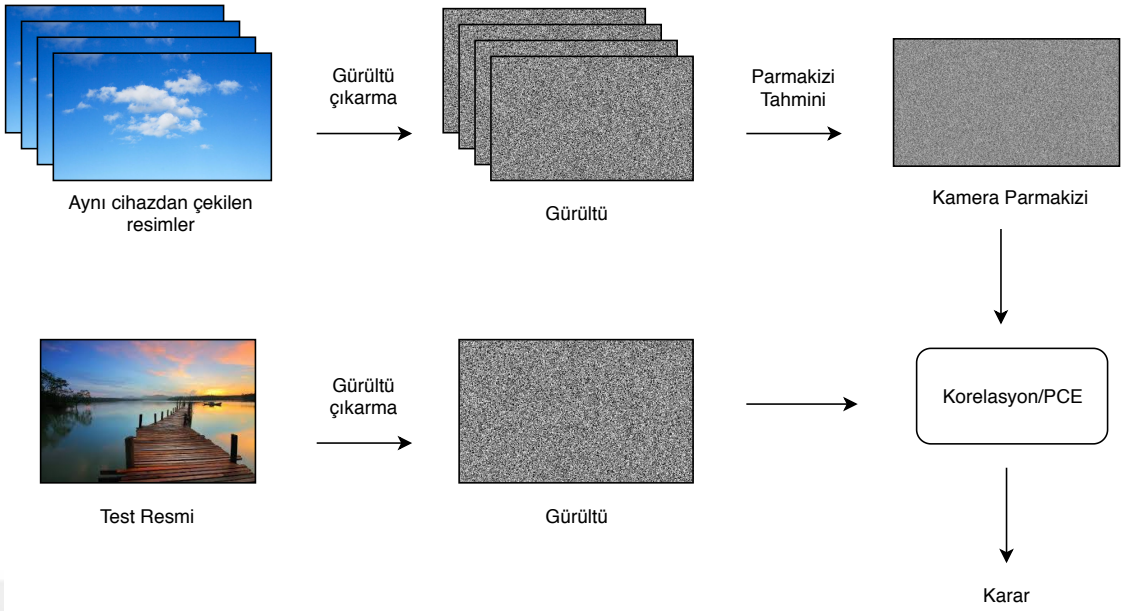
$$\hat{F} = \frac{\sum_{k=1}^{T} W^{(k)} I^{(k)}}{\sum_{k=1}^{T} (I^{(k)})^2} \tag{3.6}$$

PRNU Tabanlı Kaynak Cihaz Tanıma: Lukas ve ark. (2005) kamera içerisindeki fiziki noksanlıklardan faydalanarak PRNU yöntemini önermişlerdir. Tahmini $\hat{F}$ değeri ile bir resmin ilgili kamera cihazına ait olup olmadığının bulunabilmesi için, sorgulanan resmin gürültüsü çıkarılır. Çıkarılan gürültü ile tahmini parmak izinin korelasyonuna bakılır. Eğer korelasyon yüksek çıkarsa sorgulanan resim ilgili kamera cihazına aittir yargısına varırız. Ancak düşük korelasyon değeri üretilirse test edilen resim ilgili kamera cihazına ait olmadığı anlaşılır. Korelasyon hesabı yapılırken kullanılan denklem 3.7'de verilmiştir (Lukáš ve ark. 2006).

$$\rho(r,c) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} \left(F(i,j) - \bar{F}\right) \cdot \left(W(i+r, j+c) - \bar{W}\right)}{||F - \bar{F}|| \cdot ||W - \bar{W}||} \tag{3.7}$$

Denklem 3.7'deki F tahmin edilen parmak izi, W test görüntüsünden elde edilen gürültü, $\bar{F}$ ve $\bar{W}$ sırasıyla parmak izi ve gürültü matrislerinin ortalamalarıdır. Ayrıca $||$ işareti L-2 norm uzaklığını, r ve c ise kaydırma parametrelerini belirtmektedir. PRNU yöntemi ile kaynak cihaz tanıma yönteminin adımlar Şekil 3.2'de gösterilmiştir.

Test edilen resmin boyutu parmak izi boyutuyla uyuşmuyorsa ilk aşamada test resmi parmak izinin çıkarıldığı kamera cihazına ait değil yargısı oluşabilir. Ancak test edilen resmin belki bir kısmı kesilmiştir. Dolayısıyla korelasyon denklemine test görüntüsünün gürültüsü verilmeden önce sıfır ekleme yöntemiyle boyutlar eşitlenir. Denklem 3.7'deki kaydırma parametreleri sayesinde gürültü, parmak izi üzerinde dolaştırılır ve her bir pozisyon için korelasyon üretir. Sonuçta bize çıktı matrisi verir. Eğer parmak izi ve gürültü aynı boyuttaysa ve test edilen resim ilgili kamera cihazına aitse çıktı matrisinin (0,0) konumunda yüksek korelasyon değeri elde edilir. Eğer test edilen resim kırpılmış ve ilgili kamera modeline aitse kaydırma sayesinde çıktı matrisinin (0,0)'dan farklı bir konumda yüksek korelasyon değeri gözlemlenecektir.



Şekil 3.2. PRNU tabanlı yöntemin kaynak cihaz tanıma süreci

Şu ana kadar korelasyonun yüksek ve düşük olmasından bahsedildi. Ancak test edilen resmin çözünürlüğü elde edilecek olan korelasyonun büyüklüğünü etkileyecektir. Çözünürlükten bağımsız bir değerlendirme yapılabilmesi için Goljan ve ark. (2009) PCE formülünü önermişlerdir (denklem 3.8). Bu formül ile çözünürlük bilgisinden bağımsız olarak bir resmin ilgili cihazdan çekilip çekilmediğine karar verilmektedir.

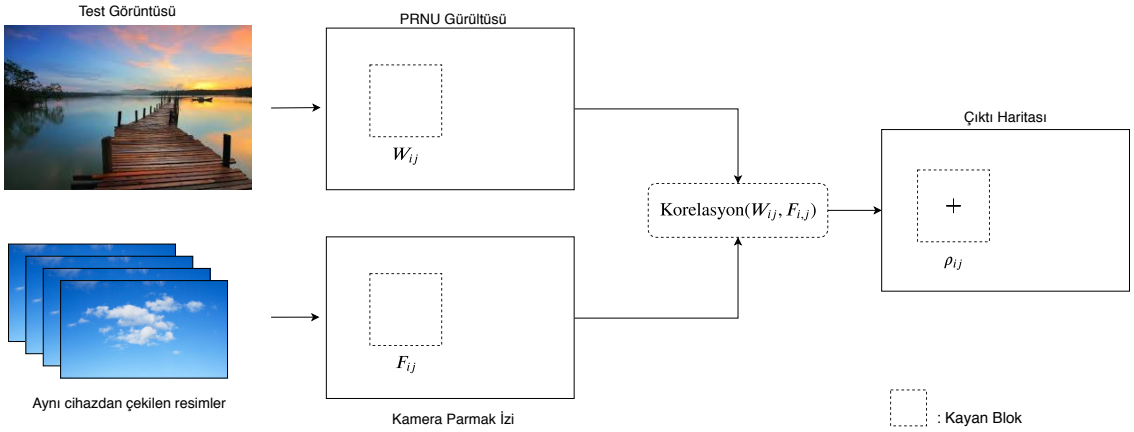
$$\mathrm{PCE}(\rho) = \frac{\rho_{max}^2}{\frac{1}{mn-|S|}\sum_{r,c\notin S}\rho(r,c)^2} \tag{3.8}$$

Denklem 3.8'deki ρ_max elde edilen korelasyon matrisinin (ρ) en yüksek değerini göstermektedir. S değeri, ρ matrisi içerisindeki en yüksek değer alan nokta ve etrafındaki noktaları temsil etmektedir. Yani S bölgesi eşleşen nokta ve etradındaki birkaç piksellik noktalardan oluşmaktadır. Renkli görüntülerde her bir kanal için (Kırmızı, Yeşil ve Mavi kanalları) parmak izi çıkarılarak (F_k, F_y, F_m) denklem 3.9'e göre birleştirilir.

$$\hat{F} = 0,29F_k + 0,58F_y + 0,11F_m \tag{3.9}$$

Eşleşme durumunda yani PCE değeri 50'nin üzerindeyse ya da korelasyon belirlenen eşikleme değerinin üzerindeyse bu duruma H_1 durumu tam tersiyse H_0 durumu denir.

PRNU Tabanlı Oynanmış Bölge Tespiti: Görüntüler üzerindeki oynanmış bölgeyi bulmak kaynak cihaz tespitinden daha karmaşık ve zor bir problemdir. PRNU yöntemiyle resmin geneli hakkında bir yorum yapılabildiği gibi resmin bölgesel parçaları hakkında da bilgiye sahip olmak mümkündür. Denklem 3.7'de verilen korelasyon formülüyle test resminin gürültüsü ile kameranın parmak izi arasındaki eşleşmeye bakılarak kaynak tespiti gerçekleştiriliyor. Eğer test edilen görüntünün bir kısmı oynanmış ise, PCE değeri olması gerekenden düşük ancak 50 değerinden yüksek çıkacaktır. Dolayısıyla bütün resmin tek bir defa eşleşme sonucuna bakılarak yapılan hesaplama yalnızca kaynağı doğrulayacaktır. Ancak eşleşme için resmin tamamını tek bir defa kullanmak yerine belirli bir boyutta pencere seçilerek her bir pencere için korelasyon değerleri üretilebilir. Seçilen pencere boyutuyla resim üzerinde kaydırılarak korelasyonlar üretilir. Korelasyonların düşük olduğu bölgelerde oynanma ihtimali yüksektir yargısına ulaşabiliriz. Anlatılan yöntem Şekil 3.3'de gösterilmiştir.



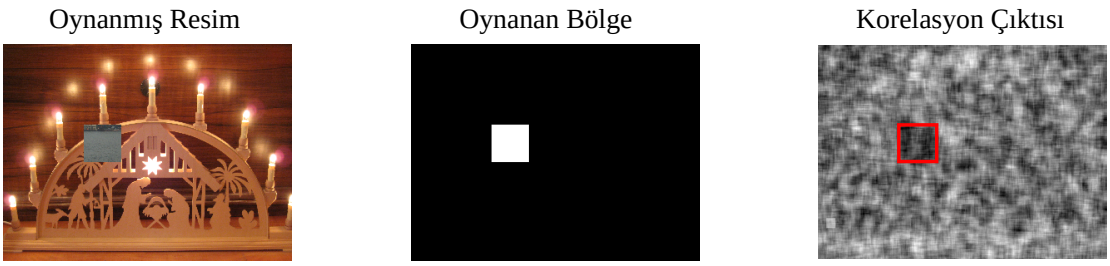
Şekil 3.3. Oynanmış bölge tespiti için PRNU tabanlı yöntemin işleyişi

Şekil 3.3'de anlatılan PRNU yöntemiyle oynanmış bölge tespitinde ilk olarak test resminden W gürültüsü çıkarılır. Ardından elde edilen gürültü üzerindeki küçük bölgelerde yani W_{ij} ve F_{ij} bölgeleri girdi olaracak şekilde denklem 3.7 kullanılarak korelasyonlar ρ_{ij} hesaplanır.

Bir önceki alt başlıkta test resminin çözünürlük bilgisinin elde edilen korelasyon değerini etkilediğinden bahsedildi. Test resminin çözünürlüğünün küçük olması demek H_1 durumunda üretilen korelasyon değerinin de H_0 durumunda üretilene yakın olması demektir.

Yani çözünürlük küçüldükçe karar verme eşiklemesi güvenilirliğini kaybetmektedir. Ancak kaydırma penceresinin küçük seçilmesi, daha küçük boyutta oynanmış bölgeleri tespit edebilmemizi sağlamaktadır. Bu yüzden deneysel olarak tespit edilen uygun pencere boyutları genellikle 64×64, 96×96, 128×128, 192×192 ve 256×256'dır. 64×64'ten küçük pencere boyutu seçildiğinde H_1 ile H_0 durumu için üretilen korelasyon değerleri neredeyse birbirine eşit derecede olmaktadır. Dolayısıyla yapılan tespit güvensiz olacaktır. 256×256 boyutundan ise tespit edilecek olan oynanmış bölgenin en az 256×256 piksel boyutunda olması gerekir. Bu durumda da çalışma aralığı düşecektir.

PRNU yöntemi ile oynanmış bölge tespiti birçok oynama çeşidinde başarılı olarak çalışmaktadır. Çünkü resmin bir bölgesinde herhangi bir değişiklik yapıldığında (klonlama, kopyala yapıştır, silme, ekleme, boyut değiştirip ekleme) o bölgenin parmak iziyle eşleşen özellikleri kaybolur. Dolayısıyla oynanan bölge düşük korelasyon değeri üretir. Bu yüzden çoğu oynama çeşidinde çalışabilmektedir. Şekil 3.4'de PRNU tabanlı oynanmış bölge tespiti örneği verilmiştir. Şekil 3.4'da verilen korelasyon çıktısı 96×96 piksel boyutunda pencere seçilerek oluşturulmuştur. Korelasyon çıktısı Şekil 3.3'de verilen işlemlere göre elde edilmiştir. Şekil 3.4'daki korelasyon çıktısında siyah olan bölgeler düşük korelasyonu, beyaz olan bölgeler ise yüksek korelasyonu göstermektedir. Oynanmış bölge siyah olarak görülebilmektedir.



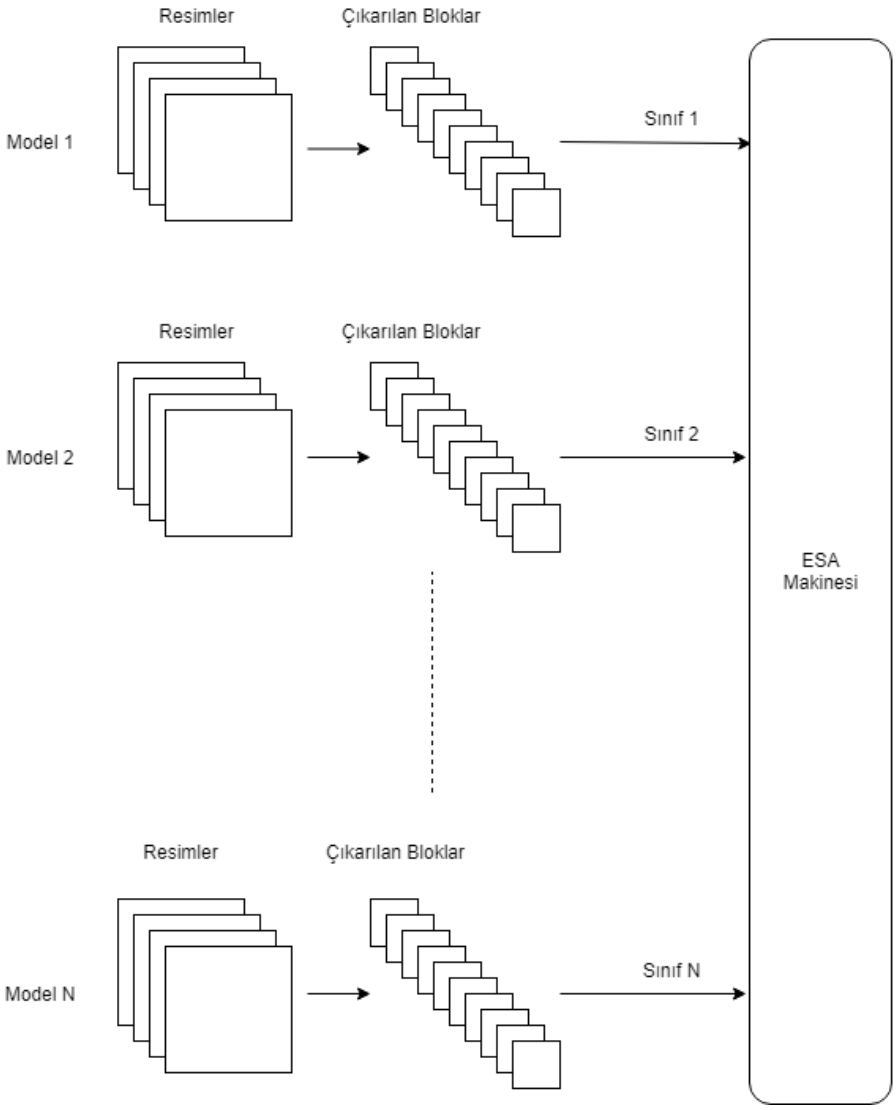
Şekil 3.4. PRNU tabanlı oynanmış bölge tespiti örneği

3.2 ESA Tabanlı Kamera Model Sınıflandırma

Son yıllarda evrişimsel sinir ağları ile karmaşık problemlere çözüm getirilmiştir. Bu durum adli bilişim alanında çalışan kimseler tarafından da kullanılmaya başlanmıştır. Temel problemlerden biri olan kamera kaynak cihaz tespiti için Bondi ve ark. (2017b) ESA tabanlı sınıflandırıcı kullanarak bir resmin hangi model kameradan çekildiğinin bulunabildiğini kanıtlamışlardır. Bu yöntemde ayrıca cihaza ya da cihazın parmaksizine ihtiyaç duyulmamaktadır. Bu bölümde ilk olarak ESA tabanlı kamera model tanıma probleminin nasıl çözüldüğü ardından oynanmış bölge tespitinin kamera model tanıma yöntemiyle nasıl yapıldığı anlatılacaktır.

ESA Tabanlı Çoklu Kamera Model Sınıflandırma Adımları: ESA tabanlı yöntemler temel olarak çokça veriye ve verilerin etiket değerlerine ihtiyaç duyarlar. Kamera model sınıflandırma için de bu durum geçerlidir. Elimizde farklı kamera modellerinden çekilmiş resimler var ise bunları ESA kullanarak ayırt etmek mümkündür. Bondi ve ark. (2017b) 18 adet kamera modeli için çalışabilen bir sınıflandırıcı eğitmişlerdir. Temel olarak camera model sınıflandırıcısı yapılırken uygulanan adımlar şu şekilde sıralanabilir:

1. Her bir sınıfı oluşturacak kamera modelleri belirlenir.
2. Her bir kamera modelinden farklı sahneler içerecek şekilde resimler seçilir
3. Her bir resimden belirlenen boyutta bloklar çıkarılır (64, 96, 128 ... gibi)
4. Eğitim için uygun ESA modeli belirlenir.
5. Her bir model için çıkarılan bloklar ile ESA eğitimi yapılır.
6. Eğitim sonrasında elde edilen ESA tabanlı makine test edilecek resimden girdi boyutunda olacak şekilde bloklar üzerinde çalıştırılır.
7. Çoklu oylama yöntemiyle test edilen resmin hangi sınıfa ait olduğu belirlenir.



Şekil 3.5. ESA tabanlı kamera model sınıflandırıcı eğitimi (çoklu sınıf)

Şekil 3.5'de N adet kamera modelini sınıflandırabilecek ESA eğitiminin süreç haritası gösterilmiştir. ESA makinesi girdi olarak belirlenen boyuttaki blokları almaktadır. Bunun 2 sebebi bulunmaktadır. Birincisi orijinal resimlerinin çözünürlüğünün büyük olması sebebiyle oluşturulacak ESA modeli, eğitimi yapabilmek için belki 1 ay belki daha fazla süreye ihtiyacı olacaktır. İkincisi ise veri miktarını arttırmaktır. ESA tabanlı yöntemlerin verilerin çok olduğu durumlarda daha başarılı çalıştığı bilinmektedir. Resimlerde blok çıkarmak ayrıca mantıklı bir işlemdir. Örneğin A modelinden çekilen bir resim doğal olarak A sınıfına aittir. Ek olarak aynı resimden çıkarılan bloklar da A sınıfına aittir. Çünkü

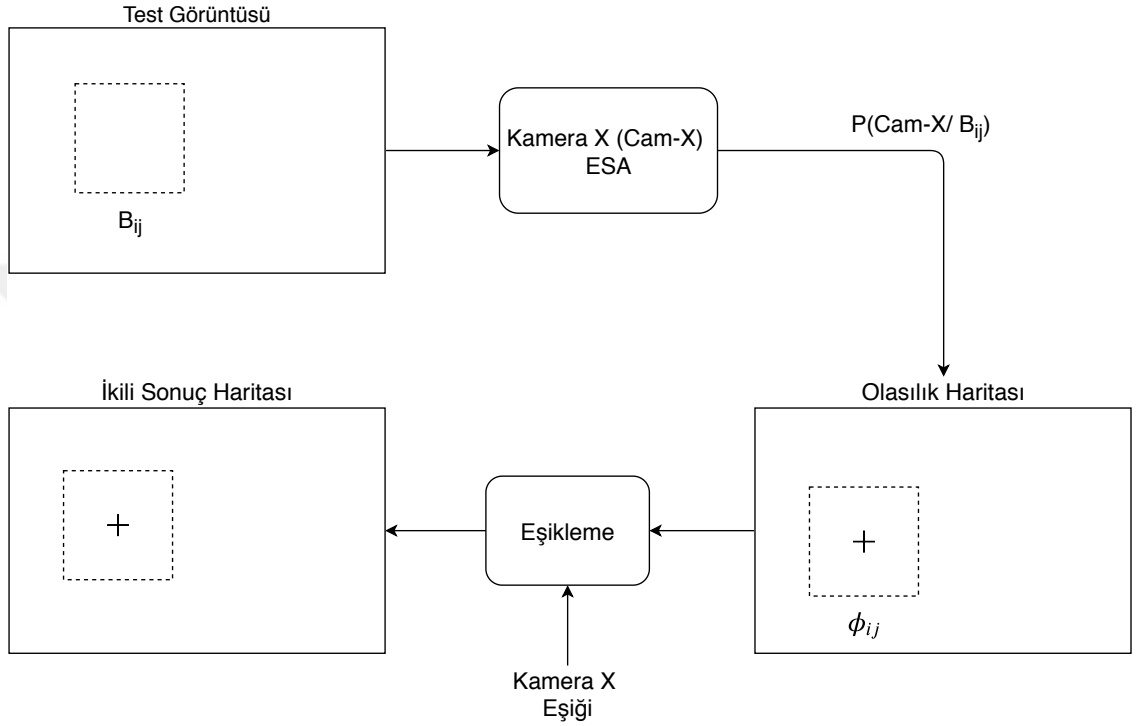
A modelinden çekilmiştir. Dolayısıyla resimlerden çıkarılan bloklar da aynı modelden çekilmiş orjinal resimler gibi değerlendirilir.

Kamera model sınıflandırıcısı eğitiminde sistemin başarısını arttırmak birkaç hususa bağlıdır. İlk olarak kullanılacak resimler farklı sahne içeriğine sahip olmalıdırlar. Aksi taktirde aşırı öğrenme sorunuyla karşı karşıya kalınabilir. Aşırı öğrenme olmasa bile tamamiyle benzer sahneler ile eğitilen ESA makinesi farklı bir sahne ile karşılaştığında hatalı sonuç üretecektir. Ek olarak seçilen resimler çok parlak veya çok karanlık olmaması gerekmektedir. Çünkü bu tarz resimler içerisinde yeteri miktarda istatistiki bilgi bulundurmamaktadırlar.

ESA eğitiminde sonra kamera model tespiti için test edilecek resimlerden girdi boyutunda bloklar çıkarılır. Çıkarılan blok sayısı resmin çözünürlük bilgisine bağlı olarak deneysel olarak belirlenir. 25-100 arası çıkarım genellikle yeterlidir. Çıkarılan bloklar için ESA makinesinin ürettiği değerlere bakılır. En çok ürettiği değer test edilen resmin ana sınıfını yani modelini belirtmektedir. Uygulanan bu yönteme çoklu oylama yöntemi denir. Böylelikle yalnızca tek bir bloğun üretebileceği yanlış sonuç çoklu oylama yapılarak düzeltilir.

PRNU yöntemi Şekil 3.1'da gösterilen kamera cihazının değişmeyen sensör özelliklerini modelleyerek kaynak cihazını tespit edebilmektir. ESA tabanlı kamera kaynak tanımada ise ESA makinesinin 3.1'deki bütün adımları öğrenmesi beklenir. Fakat Bondi ve ark. (2017b) ESA yönteminin aynı model iki cihazı ayıramadığını farklı iki modeli başarılı bir şekilde ayıtabildiğini göstermişlerdir. Buradan yola çıkılarak ESA makinesinin 3.1'de verilen adımlardan mozaikleme filtresi adımını öğrendiği yorumu yapılabilir. Çünkü aynı model iki cihaz aynı mozaikleme filtresine sahiptir. Dolayısıyla ESA taradınfan aynı cihaz olarak değerlendirilmesi normaldir. Yani PRNU yöntemi sensör özelliklerini kullanırken, ESA tabanlı yöntem mozaikleme filtresinin özelliklerini kullanarak kaynak tanımaktadırlar.

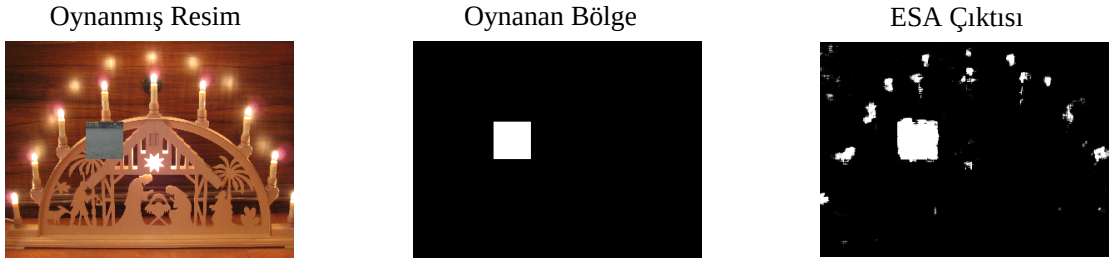
ESA Tabanlı Oynanmış Bölge Tespiti: PRNU yönteminde olduğu gibi ESA tabanlı kamera model sınıflandırıcısını kullanarak oynanmış bölge tespit etmek mümkündür. ESA makinesi eğitildikten sonra test resmi üzerinde girdi boyutundaki pencere kaydırılarak sonuç haritası çıkarılır. ESA tabanlı oynanmış bölge tespiti görseli Şekil 3.6'de gösterilmiştir. Şekil 3.6'de verilen P, B_{ij}'ye göre Cam-X olma olasılığını belirtmektedir.



Şekil 3.6. ESA tabanlı oynanmış bölge tespiti

Şekil 3.6'de verilen işlemler şu şekilde sıralanabilir: Test görüntüsünden alınan blok B_{ij}, eğitilen ESA modeline girdi olarak verilir. ESA modeli aldığı girdiye göre, eğitildiği kamera modeline ait olup olmama olasılığını ϕ üretir. Ürettiği değer test görüntüsünün ilgili bölgesine denk gelecek şekilde olasılık haritası üzerine yazılır. Test görüntüsünden elde edilen bütün bloklar ile olasılık haritası oluşturulur. Oluşturulan harita, en iyi eşikleme değeri ile eşiklenerek oynanmış bölgeler tespit edilir. Test edilen resmin her bir girdi boyutlu penceresi için ESA modelinin ürettiği sonuç, sonuç haritası üzerine yazılır. Böylelikle resim içerisinde farklı kamera modelinden bir bölge bulunuyorsa bu bölgeyi tespit etmek mümkündür. PRNU yöntemi birçok oynama çeşidinde çalışmaktadır.

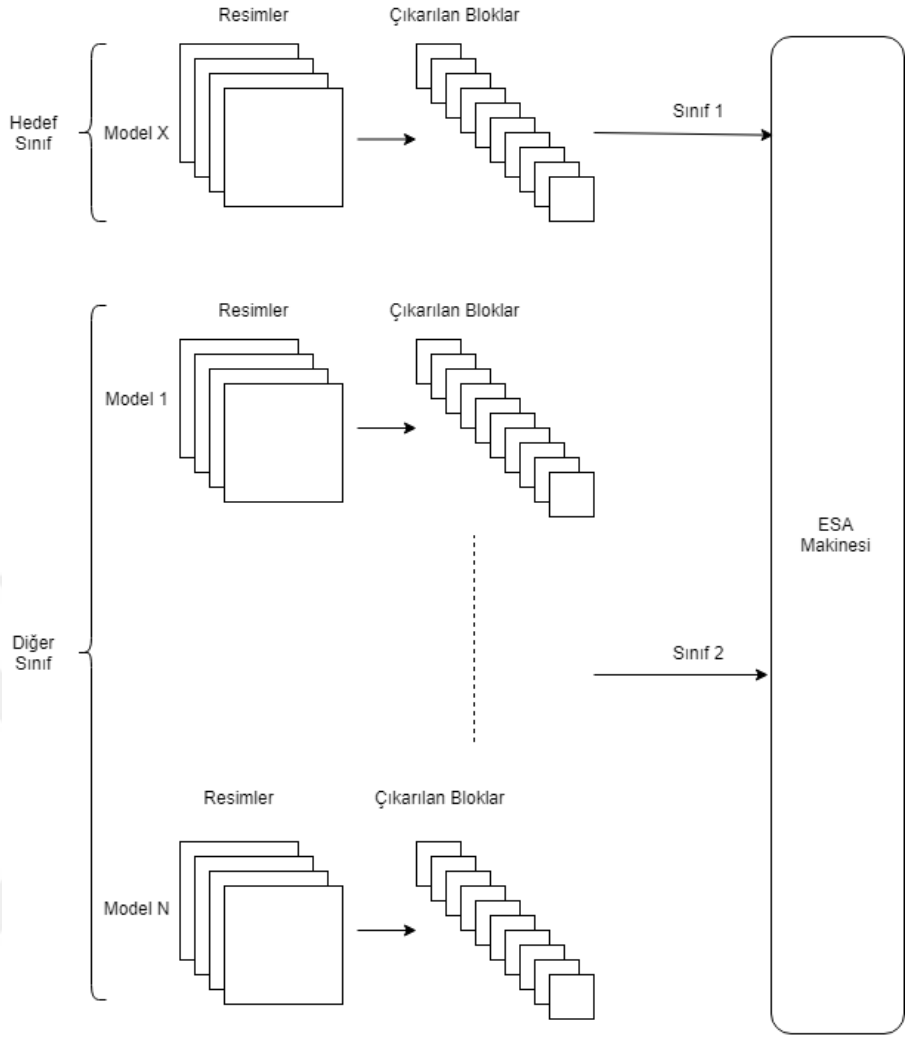
Aynısını ESA tabanlı oynama tespiti yöntemi için söyleyemeyiz. Çünkü ESA makinesi kamera modellerinin özelliklerini öğrenir. Dolayısıyla kamera model farklarında kaynaklı oynamaları tespit edebilir. Eğer farklı bir modelden kopyalanıp test resmi üzerinde yapıştırılırsa yöntemin çalışacağını biliyoruz. Ancak aynı modelden kopyalama ya da resmin kendi içerisinden klonlama işlemlerinde çalışamayacaktır. ESA tabanlı kamera model sınıflandırıcısı kullanılarak yapılan oynanmış bölge tespiti örneği Şekil 3.7'de verilmiştir. Şekil 3.7'de gösterilen ESA çıktısında oynanmış olarak bulunan yerler beyaz renk ile gösterilmiştir. Ayrıca uygun bir threshod değeri seçilerek hatalı bulunan hatalı bölgeler elimine edilebilir.



Şekil 3.7. ESA tabanlı oynanmış bölge tespiti örneği

ESA Tabanlı İkili Kamera Model Sınıflandırıcı ESA tabanlı kamera model sınıfladırıcısı verilen kamera modellerini ayırt etmek için tasarlanmıştır. Benzer şekilde sadece tek bir kamera modelini ayırt edebilen bir makine eğitimi yapmak mümkündür. Çoklu sınıflandırıcıdan tek farkı verilen sayıdaki kamera modelini tespit yerine bizim belirlediğimiz kamera modelini tespit eder. Yani ESA eğitimiyle iPhone X tespit edici eğitimi yapılabilir. Tek bir model için yapılan eğitim süreci Şekil 3.8'de verilmiştir.

Şekil 3.8'de görüldüğü gibi, hedef bir model seçilir (Model X) ve bu modelin etiket değerine 1 denir. Ardından elimizde bulunan diğer modellerden diğer sınıfı oluşturulur. Diğer sınıfın etiket değerine 0 verilir. Böylelikle eğitim sonunda elimizde ESA tabanlı iPhone X saptayıcı olur. İlk bakışta çoklu sınıfladırıcı yapmak daha avantajlı olarak gözükecektir. Ancak bir sonraki bölümde anlatılacak olan birleşim yönteminde bu yaklaşımın birleştirme işlemi için daha uygun olduğu anlaşılacaktır.



Şekil 3.8. ESA tabanlı kamera model sınıflandırıcı eğitimi (ikili sınıf)

Oynanmış bölge tespiti yapabilmek için Şekil 3.1'de verilen adımların çok iyi analiz edilmesi gerekir. PRNU yöntemi bu adımlardan sensör kısmını kendisine baz almıştır. Kamera cihazı içerisinde bulunan sensörlerin değişmeyen özelliğinden faydalanarak kameraya ait parmak izi oluşturulabileceğini ve bu parmak izi kullanılarak oynanmış bölgelerin tespitinin mümkün olduğunu göstermiştir.

ESA tabanlı kamera model sınıflandırıcısı ise Şekil 3.1'deki adımlardan mozaikleme filtresini kendisine baz almıştır. Her bir kamera modelinde farklı bir mozaikleme filtresi bulunduğunu ve bu filtrelerin farklı olmasından faydalanarak bir ayırım yapılabileceğini göstermiştir. ESA tabanlı bu yöntem, resim üzerindeki o modele ait olmayan bölgerin

tespit edilmesinin mümkün olduğunu ayrıca göstermiştir.

Her iki yöntem birbirinden farklı tabanları baz almaktadır. Dolayısıyla elimizde birbirinden farklı 2 tane bilgi bulunuyor diyebiliriz. Bu tezde PRNU yönteminin bilgisi ile ESA tabanlı kamera model sınıflandırıcısının bilgisi özel bir yaklaşımla birleştirilerek daha iyi bir oynama tespit edici önerilmiştir.

3.3 Önerilen Yöntem (Birleşim Yöntemi)

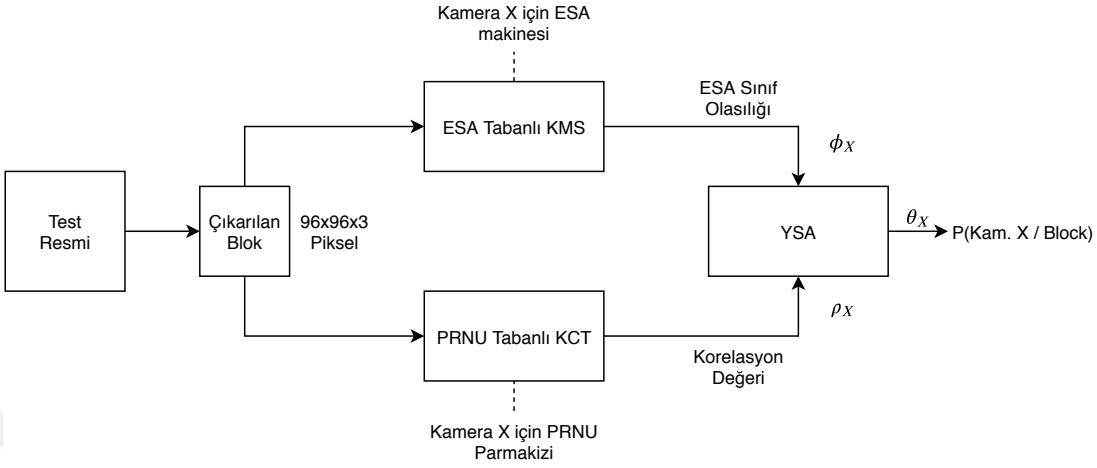
PRNU yöntemi ile ESA tabanlı yöntemin birleştirilebilmesi için öncelikle yöntemin çıktılarının iyi analiz edilmesi gerekmektedir. PRNU yöntemi hem PCE değeri hem de korelasyon değeri üretebilir. Benzer şekilde ESA tabanlı yöntem hem sınıf değerini hem de sınıfa ait olasılık değerini üretmektedir. Burada belirtilmesi gereken alt bir husus bulunmaktadır. ESA tabanlı yöntemde Şekil 3.8'de gösterilen kamera model saptayıcı tercih edilmiştir. Bunun sebebi PRNU yönteminin tek bir cihaz üzerinde sonuç üretmesinden kaynaklamaktadır. Madem PRNU yöntemi tek bir cihaz üzerinde sonuç üretiyor, biz de sadece o cihazın modelini tespit edebilecek ESA tabanlı KMS (Kamera Model Saptayıcı) yaparak aynı cihaz için iki farklı bilgi üretilmesini sağlayabiliriz. ESA tabanlı çoklu sınıflandırıcı içerisinden de hedef olarak seçilen kamera modelinin olasılık değerini elde edebiliriz. Ancak yöntemin basite indirgenmesi yani yöntemin model bazlı çalıştığının gösterilebilmesi için ESA eğitiminde KMS yaklaşımı tercih edilmiştir.

Önerilen birleşim yönteminin adımları tek bir model için şu şekildedir:

1. Seçilen hedef cihazına ait parmak izi elde edilir.
2. Seçilen hedef cihazının modelini tespit edebilen ESA makinesi Şekil 3.8'deki adımlara göre eğitilir.
3. ESA eğitimi ve parmak izi tahmininde kullanılmayan test blokları ESA tabanlı KMS yöntemi ve PRNU tabanlı KCT yöntemiyle test edilerek sırasıyla ϕ korelasyon ve ρ olasılık değerleri üretilir.

4. Üretilen ϕ ve ρ değerleri kullanılarak yapay sinir ağı eğitimi yapılır.

Birleşim yönteminin süreç haritası Şekil 3.9'de gösterilmiştir.



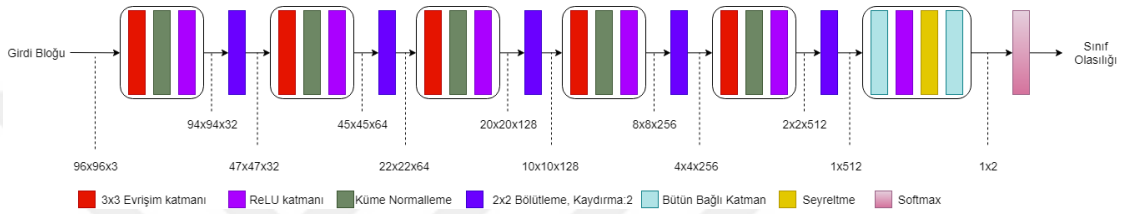
Şekil 3.9. Önerilen yöntemin blok diyagramı (KMS: Kamera Model Sınıflandırıcı, KCT: Kaynak Cihaz Tanıma, YSA: Yapay Sinir Ağı)

Bu tez kapsamında girdi bloğu olarak 96×96 piksel boyutu tercih edilmiştir. Bu sebepten ötürü Şekil 3.9'de verilen diyagramda girdi boyutu 96×96 piksel olarak belirtilmiştir. YSA eğitimi neticesinde eğitilen sistem, test edilen bloklar için θ ilgili modele ait olma olasılığını üretmektedir. Yöntemlerden birisi cihaz tabanlı diğeri ise model tabanlı olarak çalışmaktadır. Dolayısıyla önerilen birleşim yöntemi daha genel olan model için olasılık değerini ürettiğini varsayıyoruz. Mantık olarak bir modele ait olma olasılığına, aynı modelin cihazına ait olma olasığı eklenebilir.

Somut bir örnek verelim. Elimizde iPhone 5'e ait bir cihaz olduğunu varsayalım. Sırasıyla bu cihaza ait parmak izi bulunur. Cihazın modeline ait kamera cihaz saptayıcı eğitilir (iPhone 5 tespit edici). O zaman test edeceğimiz resim bloğunun hem korelasyon değeri hem de iPhone 5'e ait olma değeri bize o blok hakkında bilgi verecektir. O cihaza ait bir görüntü bloğu, aynı cihazdan çekildiği için yüksek korelasyon değeri üretecektir. Benzer şekilde O görüntü bloğu iPhone 5 modeline ait olduğundan yüksek olasılık değeri üretecektir. Aksi durumda yani farklı bir modele ait olma durumunda ise hem korelasyon hem de olasılık değeri düşük olacaktır. Akla gelen ilk soru muhtemelen 'aynı model farklı

cihazda nasıl davranacaktır?' olacaktır. Bu durumda olasılık yüksek fakat korelasyon düşük çıkacaktır. Bu sebepten ötürü önerilen birleşim yöntemi cihaz tabanlı değil model tabanlı olarak çalışmaktadır.

ESA Mimarisi Bondi ve ark. (2017a) kamera model sınıflandırıcısı için basit bir modelin dahi oldukça iyi sonuçlar ürettiğini gözlemlemişlerdir. Bu çalışmada ise VGG Simonyan ve Zisserman (2014)'dan ilham alınarak Şekil 3.10'deki model oluşturulmuştur.

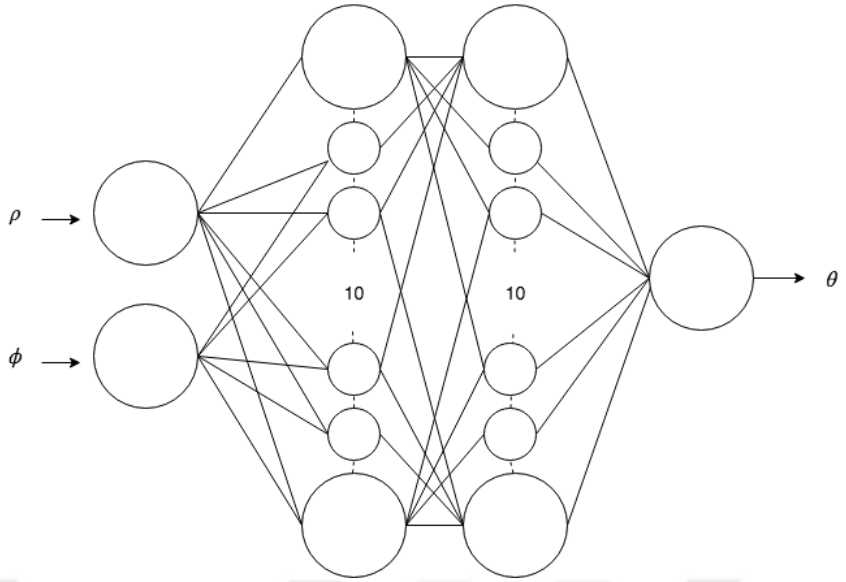


Şekil 3.10. ESA mimarisi

Şekil 3.10'de verilen ESA modelinde her bir evrişim katmanından sonra yığın normallame katmanı eğitim süresi kısaltmak için eklenmiştir. Her yığın normallame katmanından sonra ReLU aktivasyon fonksiyonu ardından ise örnekleme katmanı kullanılmıştır. Son olarak bütün bağlı katman yüzde 50 seyreltme miktarıyla softmax sınıflandırıcısına aktarılmıştır.

YSA Mimarisi Elde edilen ρ ve ϕ değerleri basit bir yasap sinir ağı ile eğitilerek önerilen birleşim yöntemini oluşturmaktadır. YSA'nın mimarisi Şekil 3.11'de verilmiştir.

Şekil 3.11'de görüldüğü gibi YSA modeli 2 girdili, iki ara katmanlı ve tek çıktılıdır. Ara katmanlar 10 adet düğüm içermektedir. Oldukça basit bir model olmasıda rağmen test sonuçlarında (Bulgular bölümünde bahsedilecektir) iki temel yöntemin üzerine çıkmıştır. Daha karmaşık bir modelin daha başarılı olacağı yargısı bu yöntem için geçerli diyemeyiz. Çünkü her bir katmanında 10 düğüm bulunan tek ara katmanlı ysa modeliyle iki ara katmanlı ysa modelinin performansı neredeyse birbirlerinin aynısıdır. Sonuç olarak YSA'nın iç mimarisinin birleşim yöntemine büyük bir etkisi bulunmamaktadır.



Şekil 3.11. Önerilen birleşim yönteminde kullanılan YSA mimarisi

3.4 Veri Seti

Bütün çalışmalarda Shullani ve ark. (2017) hazırladığı VISION veriseti kullanılmıştır. Bu veri seti içerisinde güncel cep telefonu markalarının da olduğu 29 adet kamera modeli bulunmaktadır. Set, içerisinde her bir kamera cihazı için düz (flat), ve doğal (nat) olmak üzere resimler barındırıyor. Düz olarak adlandırılan resimler, çoğunlukla gözyüzü resmi veya duvar resimlerinden oluşuyor. Doğal resimler ise güncel hayattan manzara veya hayatın içinden resimler bulunduruyor.

Her bir kamera cihazında önerilen birleşim yöntemini, dolayısıyla PRNU yönteminin uygulanabilmesi için her cihaz için parmak izi tahmini yapılması gerekmektedir. Veri seti içerisindeki flat resimler, içerisinde bulundurduğu gökyüzü ve duvar resimlerinden ötürü parmak izi tahmini yapılırken flat resimler tercih edilmiştir. Ayrıca bütün cihazlarda en az 70 adet flat resim olması parmak izi tahmini için yeterli bir miktardır. Ek olarak parmak izi tahmini gökyüzü ve duvar resimlerinden daha iyi elde edilmektedir. Veri setini hazırlayan Shullani ve ark. (2017) muhtemelen bu durumu göz önünde bulundurmuşlardır.

Bu çalışmada Vision veri seti içerisindeki 17 adet kamera modelinden 21 adet cihaz kullanılmıştır. Her bir kamera modelinden yalnızca 1 adet cihaz hedef kamera olarak

seçilip birleşim yönteminin eğitiminde kullanılmıştır. Geriye kalan 4 adet cihazın ise 'Eğitilen modelin farklı cihaz ile testi' bölümünde nasıl test edildiği anlatıcalaktır.

parmak izi çıkarımı haricindeki eğitim ve test kısımlarında, her bir cihaz için toplamda 160 adet görüntü kullanılmıştır. 160 adet görüntünün 100 tanesi ESA eğitiminde (yalnızca hedef kamera için), 50 tanesi YSA eğitimi ve yöntem karşılaştırması kısmında kullanılmıştır. Geriye kalan 10 adet resim ise oynanmış bölge tespiti yapılırken en uygun eşikleme değerinin bulunması kısmında kullanılmıştır.

Resimlerden Blok Çıkarma İşlemi: ESA tabanlı yöntem ve PRNU yöntemi girdi olarak görüntü bloğu aldığı için her bir kamera modeli için seçilen 160 adet resimden 96×96 piksel boyutunda bloklar çıkarılmıştır. ESA eğitiminde, hedef ve diğer olmak üzere toplamda 2 adet sınıf bulunmaktadır. Dolayısıyla 17 adet kamera modelinden 1 tanesi hedef, geri kalanlar ise diğer sınıfa seçilmiştir. Hedef olarak seçilen kamera modelindeki resimlerin her birinden 500 adet 96×96 blok, diğer sınıfında ki kamera modellerinden ise 50'şer adet blok çıkarılmıştır. Her kamera modeli için aynı işlem gerçekleştirilmiştir. Örneğin ilk kamera modelini hedef olarak seçelim. Bu durumda geri kalan 16 model diğer sınıfına ait olur.

Bütün eğitim işlemleri neticesinde elimizde ilk kamera modeli için oluşturulmuş oynama tespit edici makine olacaktır. Sonrasında ikinci kamera modeli hedef olarak seçilir. Aynı işlemler neticesinde elimizde ikinci kamera modeli için oynanmış bölge tespit edici sistem olmuş olacaktır. Bu çalışmada 17 adet kamera modelinin her biri için birleşim yöntemi modeli oluşturulmuştur.

Kullanılan blok sayılarının ve kullanıldığı yerlerin karışmaması açısından ESA eğitiminde kullanılan görüntü ve görüntü blokları C, YSA eğitimde kullanılan görüntü blokları S_{tr}, YSA testinde kullanılan (aynı zamanda ESA ve PRNU ile karşılaştırırken kullanılan) görüntü blokları S_{ts} olarak adlandırıldı. ESA eğitiminde her bir model için kullanılan blok ve görüntü sayıları Çizelge 3.2'de verilmiştir.

Çizelge 3.1. Çalışmada kullanılan kamera modelleri

Etiket	Marka	Model	Çözünürlük
K-1	Samsung	S3 Mini	2560×1920
K-2	Apple	iPhone 4s	3264×2448
K-3	Huawei	P9	3969×2976
K-4	LG	D290	3264×2448
K-5	Apple	iPhone 5c	3264×2448
K-6	Samsung	Tab3	2048×1536
K-7	Apple	iPhone 4	2592×1936
K-8	Samsung	Galaxy S3	3264×2448
K-9	Sony-Xperia	Z1 Compact	5248×3936
K-10	Apple	iPad2	960×720
K-11	Huawei	P9 Lite	4160×3120
K-12	Microsoft	Lumia 640 LTE	3264×1840
K-13	Apple	iPhone 6 Plus	3264×2448
K-14	Apple	iPad Mini	2592×1936
K-15	Wiko	Ridge 4G	3264×2448
K-16	Samsung	Trend Plus	2560×1920
K-17	One Plus	A3000	4640×3480

Çizelge 3.2. ESA eğitiminde kullanılan blok ve görüntü sayıları

	Resim sayısı		Blok sayısı	
Set İsmi	Hedef Kamera	Diğer Kameralar	Hedef Kamera	Diğer Kameralar
C_{tr}	80	1280	40k	64k
C_{ts}	20	320	10k	16k
C	100	1700	50k	80k

Çizelge 3.2'deki C_{tr} ESA eğitiminde kullanılan görüntü bloklarını, C_{ts} ESA ara-testinde kullanılan görüntü bloklarını göstermektedir. Her bir blok 96×96 piksel boyutundadır. "k" sayısı bin ile çarpım anlamına gelmektedir. Çizelge 3.2, tek bir hedef kamera için gösterilmiştir. Aynı parametreler 17 kamera modeli için geçerlidir. Bütün testlerde kullanılan kamera modelleri Çizelge 3.1'de verilmiştir.

4 BULGULAR

Birleşim yöntemi PRNU tabanlı ve ESA tabanlı oynanmış bölge tespit ediceleri YSA yardımıyla birleştirerek daha iyi bir oynanmış bölge tespit edici sunmaktadır. Yöntemin daha iyi olarak çalıştığı öncelikle birleşim yönteminin PRNU ve ESA tabanlı yöntem ile karşılaştırılarak gösterilmiştir. Daha sonra birleşim yönteminin JPEG sıkıştırma formatına karşı dayanıklılığı test edilmiştir. Ardından eğitilen modellerin farklı bir cihaz ile testi yapılarak modellerin genellenebilir olduğu gösterilmiştir. Son olarak literatürdeki PRNU tabanlı çalışan oynanmış bölgeyi tespit edebilen diğer yönetemler ile karşılaştırılmıştır.

4.1 ESA Eğitim Sonuçları

Her bir model için belirlenen C_{tr} seti ile ESA tabanlı kamera model saptayıcı makinesi eğitimi gerçekleştirilmiştir. ESA makinesinin başarılı olup olmadığının görülebilmesi için C_{tr} seti ile ara test gerçekleştirilmiştir. Her bir eğitim gerçekleştirilirken 1 adet kamera modeli hedef olarak seçildi. Geriye kalan bütün modeller ise diğer sınıfına ait olarak değerlendirildi. Böylelikle seçilen hedef kamera modelini tespit edebilen ESA makinesi eğitimi gerçekleştirilmiş oldu. Yapılan ara test sonuçları Çizelge 4.2'de verilmiştir.

4.2 Önerilen Yöntemin Başarım Sonuçları

Yöntemlerin kendi içerisindeki testi gerçekleştirilirken, karşılaştırma metriği olan ROC eğrileri kullanılmıştır. Çünkü elimizde 3 yöntem için ϕ, ρ ve θ değerleri bulunmaktadır. ϕ ve θ değeri 0 ile 1 arasındaki olasılık değerini, ρ ise genellikle -0,2 ile 0,3 arasında değişen korelasyon değerini göstermektedir. Resim üzerindeki oynanmış bölge tespiti için normal şartlarda oynanmış resimlerden oluşan bir test seti belirlenir. Her bir yöntem aynı test seti üzerinde değerlendirilir. Elde edilen olasılık veya korelasyon haritalarında her bir eşikleme değeri (olasılık için 0'dan 1'e, korelasyon için -0,3'ten 0,2'ye) için Çizelge 4.1'de verilen hesaplama parametreleri bulunur. Bu değerler ile istenen karşılaştırma

Çizelge 4.1. Hesaplama parametreleri

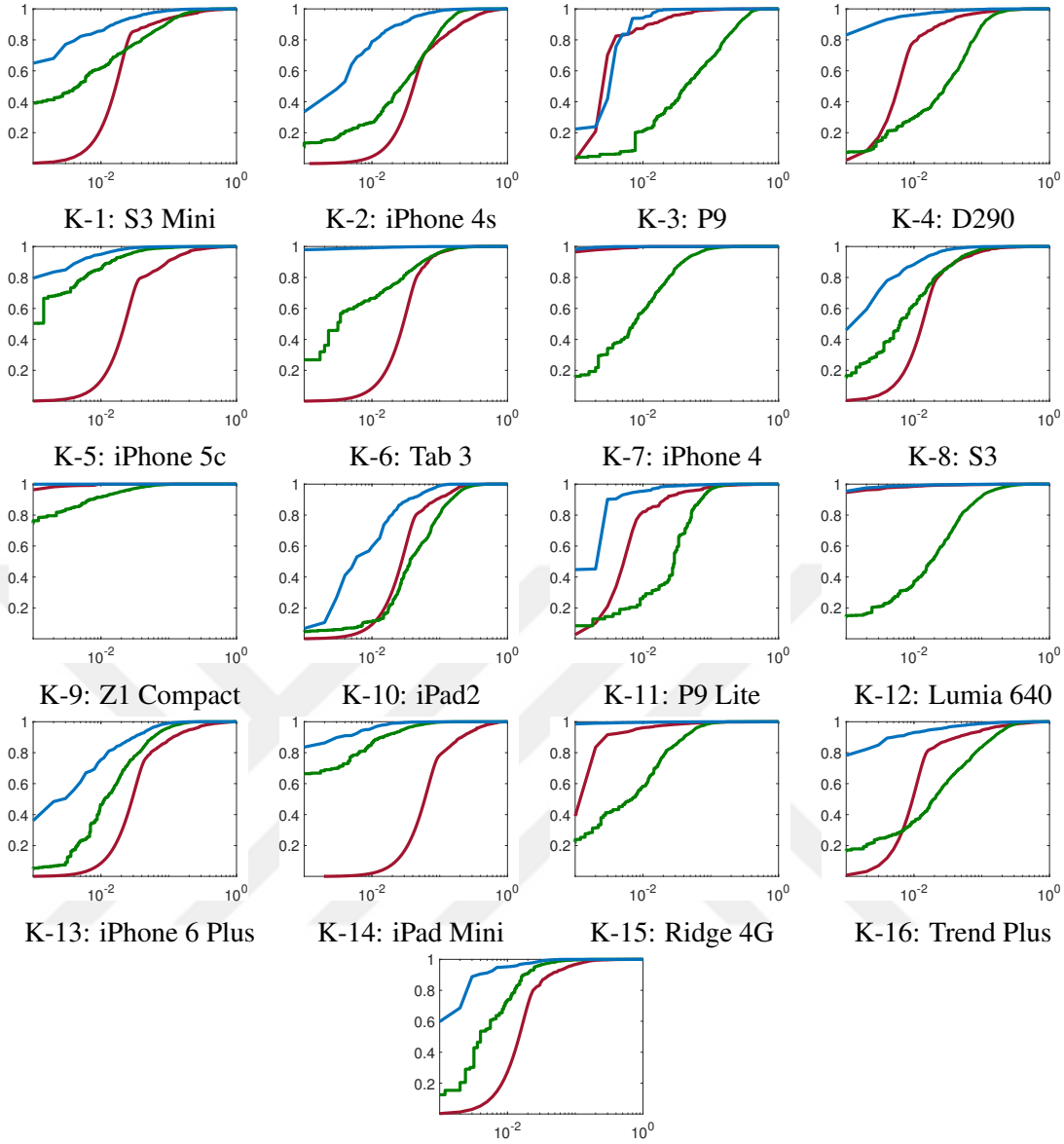
		Gerçek Değer	
		Pozitif	Negatif
Tahmin değeri	Pozitif	TP	FP
	Negatif	FN	TN

yöntemiyle karşılaştırmak mümkündür.

ROC eğrileriyle karşılaştırma yönteminde her bir eşikleme değeri için hesaplanan TP, FP değerleri çizdirilir. Çizilen bu grafik altında kalan en büyük alan 1'dir. ROC eğrisinin altında kalan alan AUC olarak adlandırılacaktır. ESA, PRNU ve birleşim yöntemi karşılaştırılırken S_{ts} setindeki bloklar kullanılmıştır. Ancak S_{ts} setini oluşturan görüntülerden bazıları Vision datasetinden kaynaklanan karışıklık nedeniyle 50'den düşük PCE değeri üretmektedir. Bir resmin 50'den daha az PCE değeri üretmesi, resmin belirtilen cihaza ait olmadığı demektir. Dolayısıyla 50'den düşük PCE değerine sahip görüntüler S_{ts} seti içerisinden çıkarılmıştır. Dolayısıyla TP ve FP için hesaplama işlemi şu şekilde belirlenmiştir. TP: diğer sınıfına ait bloğa, diğer sınıfına ait diyebilme ve FP: hedef sınıfına ait bloğa, diğer sınıfına ait diyebilme, olarak yazılabilir. Yukarıda açıklanan TP ve FP değerleriyle her bir yöntem için hesaplanan ROC eğrileri Şekil 4.1'de verilmiştir.

Şekil 4.1'de görüldüğü gibi bütün kamera modellerinde birleşim yöntemi, ESA tabanlı ve PRN tabanlı yöntemlerden daha iyi performans göstermiştir. Her bir kamera modeline ait ROC eğrilerinin altında kalan AUC değerleri ayrıca Çizelge 4.2'de gösterilmiştir.

Çizelge 4.2'deki sonuçlar önerilen yöntemin mevcut iki yöntemin de önüne geçtiğini göstermektedir. Çizelge 4.2'deki AUC değerleri Şekil 4.1'de verilen eğrilerden hesaplanmıştır. $|S_{\text{ts}}|$ test setindeki görüntü bloklarının sayısını, $|S_{\text{tr}}|$ ise eğitim setindeki görüntü bloklarının sayısını göstermektedir. Ek olarak bazı kamera modelleri için PRNU, ESA ve birleşim yönteminin karşılaştırması Şekil 4.2'de gösterilmiştir. Şekil 4.2'de girdi bloğunun boyutu 96×96 ve kaydırma miktarı 8 olarak seçilmiştir. Son 3 sütundaki görüntüler % 1 FP değerine göre seçilmiş eşikleme değeriyle hesaplanmıştır. Yeşil olan pikseller TP, Kırmızı olan pikseller FP ve mavi olan pikseller ise bulunamayan pikselleri göstermektedir.



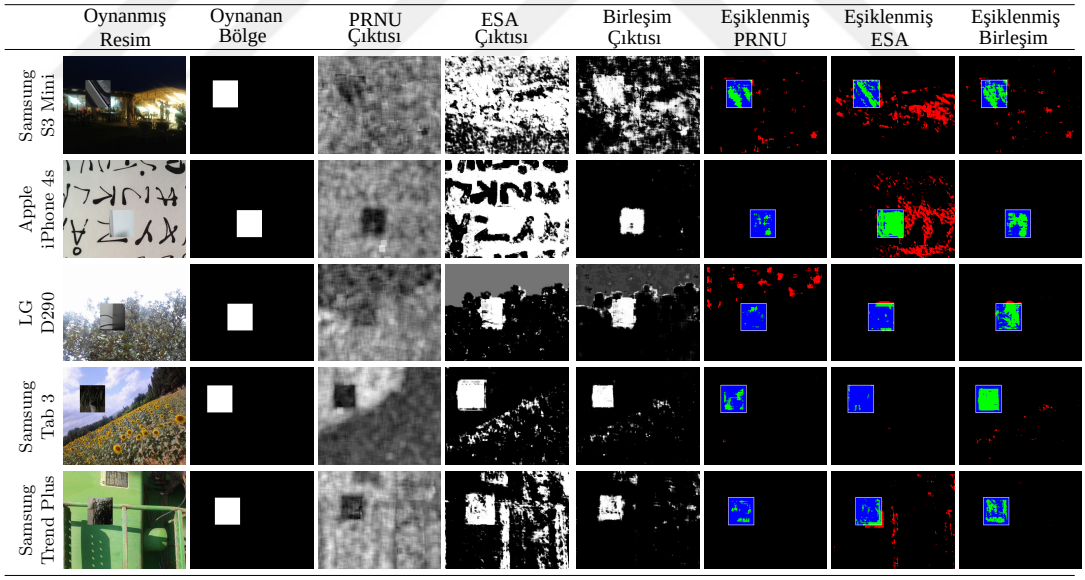
Şekil 4.1. Her bir modelin ROC eğrileri. X ekseni TP oranını, Y ekseni ise FP oranını göstermektedir (Mavi: Birleşim yöntemi, Yeşil: PRNU yöntemi, Kırmızı: ESA yöntemi)

4.3 JPEG Sıkıştırmasının Önerilen Yöntem Başarısına Etkisi

Sayısal görüntülerde herhangi bir manipülasyon işlemi yapıldıktan sonra yapılan işlemin kalıntılarının gizlenebilmesi için JPEG sıkıştırması çokça tercih edilen bir yöntemdir. Ayrıca manipülasyon yapılmasa dahi görüntülerin kapladıkları depolama yerini azaltmak için JPEG sıkıştırması yapılmaktadır. Çoğu kamera modelinde otomatik olarak JPEG

Çizelge 4.2. Birleşim yöntemi ile yöntemi oluşturan iç yöntemlerin karşılaştırılması

Etiket	C_{ts} için Baş. yüzdesi	Blok Sayıları $\mid S_{ts} \mid / \mid S_{tr} \mid$	S_{ts} için AUC değerleri		
			PRNU	ESA	Birleşim
K-1	%92	13,9k / 55,6k	0,97	0,96	0,99
K-2	%80	14,5k / 58k	0,95	0,92	0,99
K-3	%95	10,8k / 43,2k	0,91	0,99	0,99
K-4	%94	14,2k / 56,8k	0,95	0,98	0,99
K-5	%93	13,7k / 54,8k	0,99	0,96	0,99
K-6	%97	13k / 52k	0,98	0,96	0,99
K-7	%97	14,5k / 58k	0,98	0,99	0,99
K-8	%94	14,5k / 58k	0,98	0,98	0,99
K-9	%99	14,5k / 58k	0,99	0,99	0,99
K-10	%85	14,5k / 58k	0,94	0,96	0,98
K-11	%96	10,6k / 42,4k	0,97	0,99	0,99
K-12	%95	14,5k / 58k	0,96	0,99	0,99
K-13	%89	14,5k / 58k	0,98	0,94	0,99
K-14	%85	13,5k / 54k	0,99	0,91	0,99
K-15	%92	14,5k / 58k	0,98	0,99	0,99
K-16	%85	14,5k / 58k	0,95	0,97	0,99
K-17	%95	12k / 48k	0,99	0,98	0,99



Şekil 4.2. Oynama tespiti için örnek sonuçlar

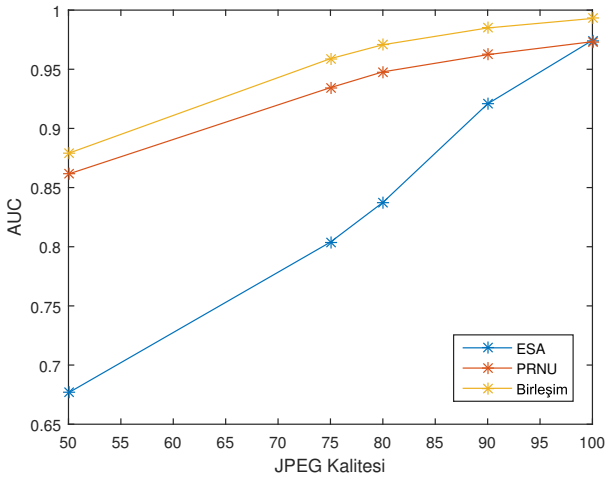
sıkıştırması özelliği bulunmaktadır. Dolayısıyla önerilen yöntemin ve yöntemi oluşturulan 2 temel yöntemin JPEG sıkıştırmasına karşı dayanıklılığı test edilmiştir. JPEG dayanıklılık

testi sonuçları Çizelge 4.3'de verilmiştir.

Çizelge 4.3. Her bir kamera modeli için 100, 90, 80, 75, 50 JPEG kalitesindeki AUC değerleri

	Yöntem	ESA					PRNU					Birleşim				
	Kalite	100	90	80	75	50	100	90	80	75	50	100	90	80	75	50
Kamera Model Etiketi	K-1	0,95	0,66	0,55	0,54	0,52	0,97	0,93	0,90	0,89	0,81	0,99	0,94	0,90	0,89	0,81
	K-2	0,92	0,90	0,80	0,78	0,66	0,95	0,94	0,92	0,92	0,87	0,99	0,98	0,96	0,95	0,90
	K-3	1,00	0,97	0,87	0,85	0,83	0,93	0,92	0,90	0,88	0,81	1,00	0,99	0,97	0,96	0,90
	K-4	0,99	0,98	0,95	0,92	0,83	0,96	0,94	0,92	0,89	0,80	1,00	1,00	0,98	0,97	0,90
	K-5	0,95	0,85	0,79	0,75	0,65	0,99	0,98	0,97	0,95	0,86	1,00	0,99	0,98	0,96	0,86
	K-6	0,96	0,83	0,70	0,66	0,59	0,98	0,95	0,92	0,88	0,79	1,00	0,98	0,93	0,89	0,78
	K-7	1,00	0,96	0,83	0,82	0,58	0,98	0,97	0,97	0,96	0,92	1,00	1,00	0,99	0,99	0,93
	K-8	0,98	0,92	0,86	0,79	0,61	0,98	0,98	0,97	0,96	0,89	1,00	0,99	0,98	0,98	0,90
	K-9	1,00	0,99	0,88	0,84	0,70	1,00	0,99	0,98	0,98	0,94	1,00	1,00	0,99	0,98	0,95
	K-10	0,96	0,94	0,84	0,83	0,71	0,93	0,93	0,93	0,93	0,86	0,99	0,98	0,98	0,97	0,90
	K-11	0,99	0,95	0,93	0,92	0,65	0,97	0,95	0,94	0,93	0,79	1,00	0,98	0,98	0,97	0,82
	K-12	1,00	0,99	0,99	0,98	0,70	0,96	0,94	0,93	0,91	0,81	1,00	0,99	0,99	0,98	0,84
	K-13	0,95	0,93	0,85	0,83	0,64	0,98	0,97	0,97	0,96	0,90	0,99	0,99	0,98	0,97	0,90
	K-14	0,92	0,90	0,85	0,83	0,77	0,99	0,99	0,98	0,97	0,94	1,00	1,00	0,99	0,99	0,96
	K-15	1,00	0,91	0,96	0,90	0,72	0,99	0,98	0,95	0,90	0,74	1,00	1,00	0,99	0,97	0,81
	K-16	0,97	0,94	0,88	0,83	0,75	0,95	0,94	0,92	0,91	0,85	0,99	0,99	0,97	0,96	0,89
	K-17	0,98	0,98	0,96	0,93	0,85	0,98	0,97	0,97	0,95	0,89	0,99	0,99	0,99	0,98	0,93

Bütün kamera modelleri için birleşim yönteminin JPEG sıkıştırmasına karşı dayanıklılığı ayrıca Şekil 4.3'de gösterilmiştir.



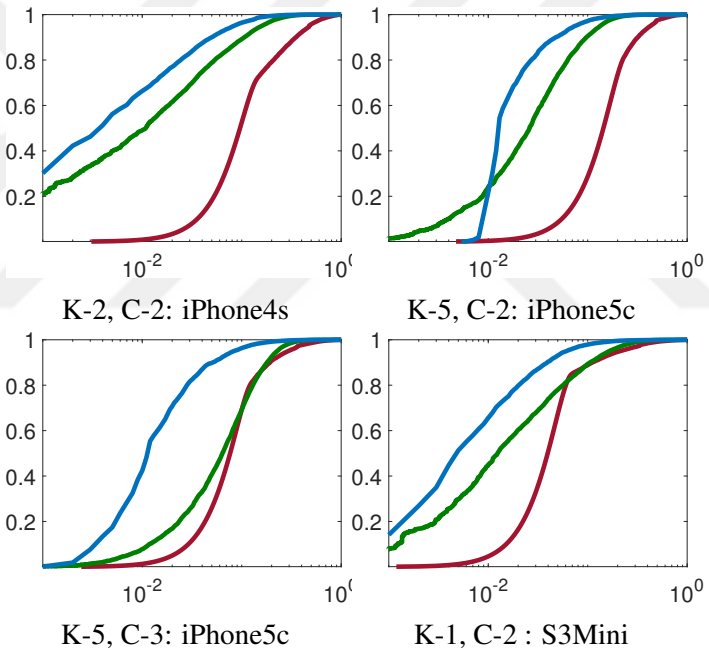
Şekil 4.3. JPEG sıkıştırmasına karşı bütün kamera modellerinin ortalama AUC değerleri

PRNU yönteminin JPEG sıkıştırmasına dayanıklı olduğu biliniyor ancak ESA tabanlı yöntemin JPEG sıkıştırmasına karşı dayanıklılığı literatürde araştırılmamış bir konudur. Dolayısıyla hem ESA tabanlı yöntemin hem de birleşim yönteminin JPEG'e karşı dayanıklılığı bu test ile gösterilmiştir. Şekil 4.3'de görüldüğü gibi PRNU yöntemi tek başına

dayanıklı olmasında karşın ESA yöntemi tek başına JPEG sıkıştırmasında dayanıklı olmadığı görülüyor. Ancak birleşim yöntemi biri dayanıklı biri dayanıksız olan bu yöntemleri birleştiriken JPEG'e karşı dayanıklı olabilme özelliğini kazanmıştır. Dolayısıyla bu açıdan da hem PRNU yönteminden hem de ESA tabanlı yöntemden üstündür.

4.4 Önerilen Yöntemin Aynı Model Farklı Cihazlardaki Başarımı

Birleşim yöntemi içerisinde hem hedef kamera için ESA makinesi hem de korelasyon ve olasılık değeriyle çalışan YSA makinesi eğitimi yapılmaktadır. Sunulan yöntemin model tabanlı olarak çalıştığını iddia ettiğimiz için eğitilen modellerin farklı aynı model fakat farklı cihazlarda da test edildiği zaman çalışacağı bu test ile gösterilecektir.



Şekil 4.4. Birleşim yönteminin aynı model farklı cihazlar için elde edilen ROC eğrileri. Kamera isimleri için lütfen Çizelge 4.4'e bakınız. (Mavi eğri: Birleşim yöntemi, Yeşil eğri: PRNU yöntemi, Kırmızı eğri: ESA yöntemi)

Veri setini açıklarken toplamda 21 adet cihaz ve 17 adet kamera modelinin kullanıldığı söylendi. Daha önceki testlerde kullanılmayan 4 adet cihaz ki bu cihazların modeli 17 adet kamera modelindeki modellerin bazılarından oluşuyor, bu testte kullanılmıştır. Amacımız eğitilen modellerin farklı cihazlarda da çalışabilirliğini test etmektir. Test

Çizelge 4.4. Daha önce eğitilen ESA ve YSA modeliyle, aynı model farklı cihazların test sonuçları (C: Cihaz, K: Kamera)

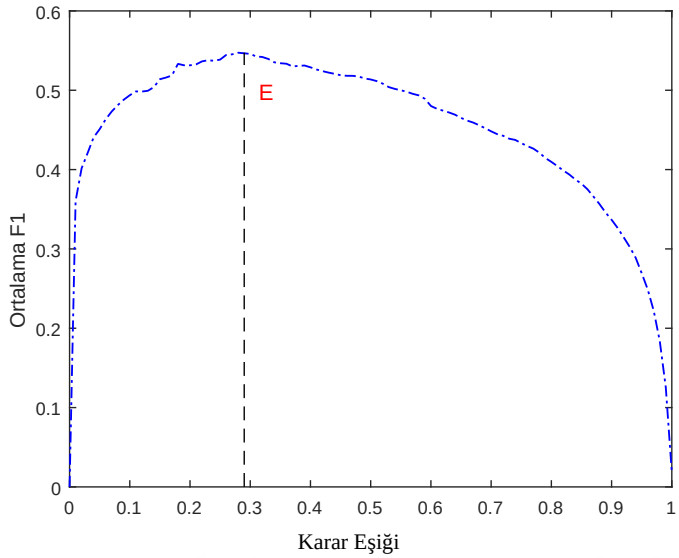
Cihaz Etiketi	Kamera Modeli	S_{ts} için AUC değerleri		
		PRNU	ESA	Birleşim
K-2,C-2	iPhone 4s	0,965	0,855	0,982
K-5,C-2	iPhone 5c	0,959	0,837	0,976
K-5,C-3	iPhone 5c	0,916	0,902	0,976
K-1,C-2	S3 Mini	0,962	0,939	0,984

işleminde kullanılan kamera modelleri ve 3 temel yöntem için ROC altında kalan AUC değerleri Çizelge 4.4'de gösterilmiştir. Çizelge 4.4'deki AUC değerleri Şekil 4.1'deki ROC eğrilerinden hesaplanmıştır.

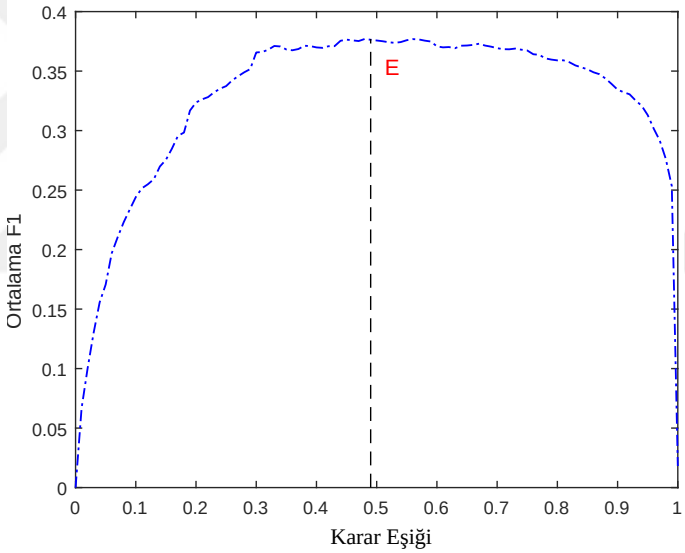
Test işlemi yapılırken daha önce eğitilen modeller, üzerinde hiç bir değişiklik yapılmadan kullanıldı. Tek farklılık test edilen cihazlardan parmak izi çıkarılmıştır. Parmak izi her cihaz için tekil bir özellik olmasından ötürü farklı bir cihazın parmak izini kullanmak mantıksız olacaktır. Test edilen bloklardan yeni oluşturulan parmak izi ile korelasyon hesaplanmıştır. Ardından hazır olarak alınan ESA modeliyle bloklar için olasılık değerleri hesaplanmıştır. Son olarak hazır olarak alınan YSA modeliyle de θ olasılık değeri üretilmiştir. Elde edilen ROC sonuçları Şekil 4.4'de gösterilmiştir.

Eşikleme Değeri Belirleme Deneylerde kullanılan veri setleri anlatılırken toplamda 160 resim 100, 50 ve 10 olarak ayırılmıştı. 100 ve 50 olarak ayırılan resimler ile test ve eğitim işlemleri gerçekleştirildi. 10 adet resim ise en iyi eşikleme değerinin bulunabilmesi için ayırıldı.

Sunulan birleşim yöntemi girdi olarak kabul ettiği girdi boyutundan (96×96) daha büyük çözünürlükte görüntü aldığı zaman, bu büyük görüntü üzerinde kaydırma ve hesaplama yaparak bütün görüntü için olasılık değerinden oluşan sonuç haritası üretmektedir. Sunulan yöntem literatürdeki diğer yöntemlerle karşılaştırılırken sabit bir eşikleme değerinin belirlenmesi gerekmektedir. Bu değerin belirlenebilmesi için ekstra 10 adet resim 400×400, 200×200 ve 100×100 boyutlarında manipüle edilerek (kopyala yapıştır) toplamda



Şekil 4.5. Kamera 5 için en yüksek F1 değerini veren eşik değeri (E) belirleme



Şekil 4.6. Kamera 11 için en yüksek F1 değerini veren eşik değeri (E) belirleme

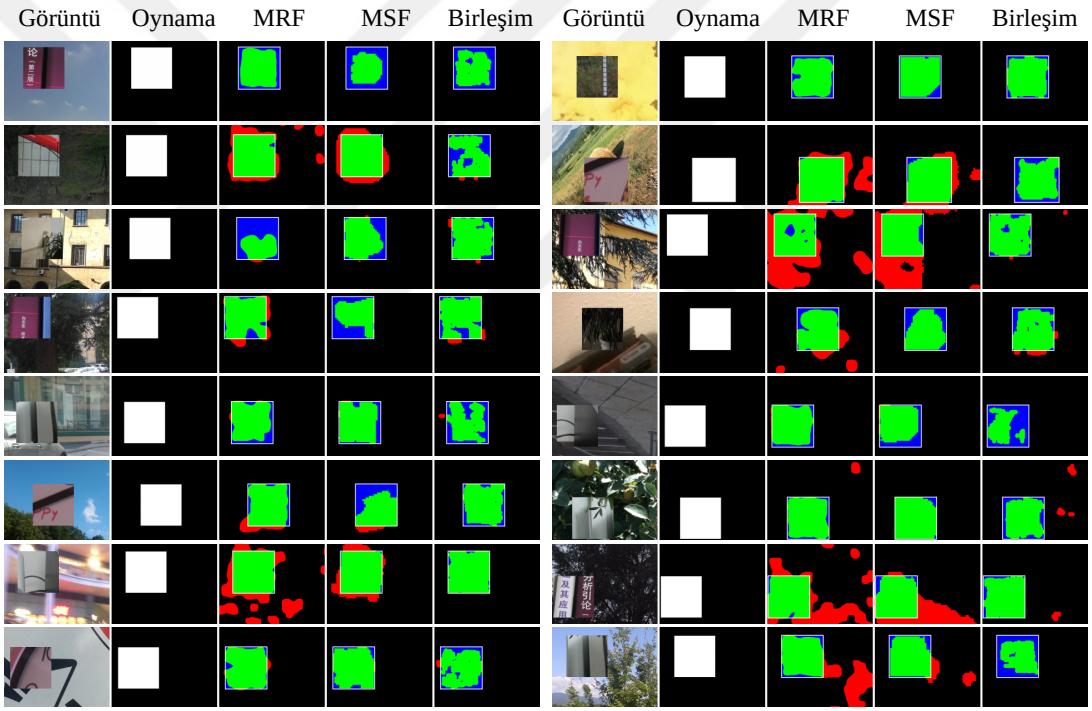
30 adet ekstra oynanmış görüntü elde edilmiştir. Elde edilen bu 30 görüntü özel birleşil yöntemiyle test edilmiştir. Test neticesinde elde edilen olasılık haritalarında her bir eşiklem değeri için F1 (F-skor) ölçme değeri (Denklem 4.1) hesaplanmıştır. Her bir model için en uygun eşikleme değeri 2 kamera modeli için Şekil 4.5 ve 4.6'deki gibi karşılaştırma için belirlenmiştir. Denklem 4.1 içerisindeki TP, FP, FN ve TN kısaltmaları Çizelge 4.1'de

verilmiştir.

$$\text{F-değeri (F1)} = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}} \tag{4.1}$$

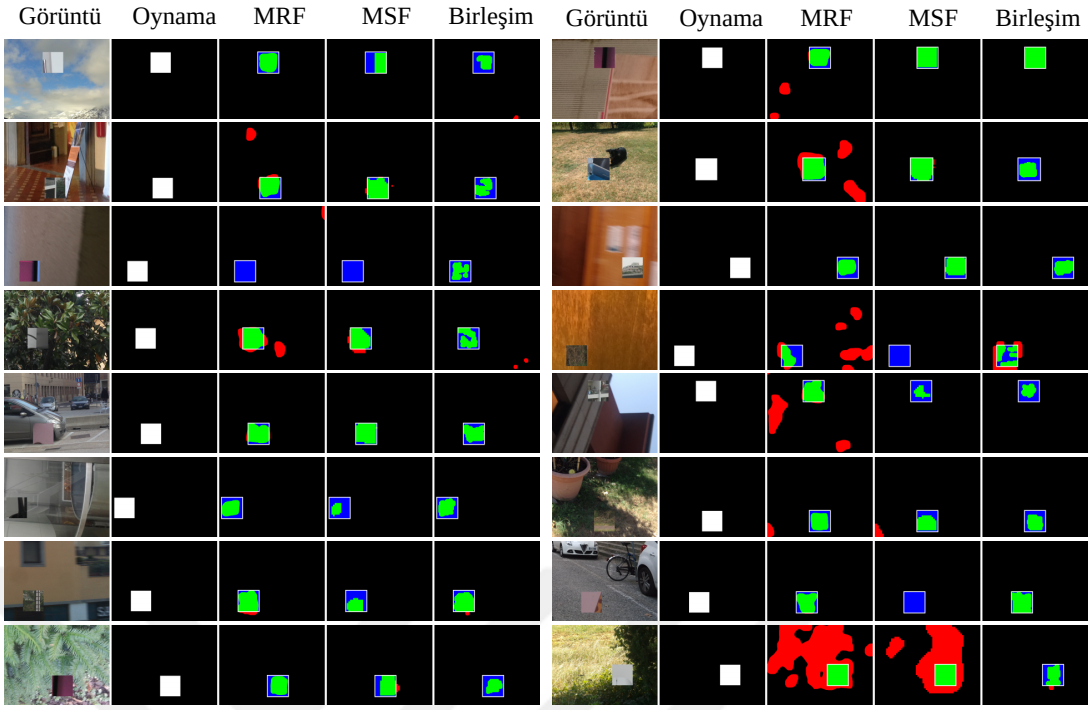
4.5 Önerilen Yöntem ile Resim İçi Oynamaların Tespiti

Birleşim yöntemi 2 farklı yöntemin birleşmesinden oluşan bölgesel oynamaları tespit edebilen bir yöntemdir. Literatürde lokal oynamaları tespit edebilen birçok yöntem bulunmaktadır. Ancak sunulan yöntem literatürdeki en iyi yüzdeyle çalışan yöntemlerden türetilmiştir. Literatürdeki ESA tabanlı yöntemler ile karşılaştırılmak istendiğinde adil bir karşılaştırma olmayacaktır.



Şekil 4.7. 400×400 piksel boyutundaki oynamalar için örnekler

Çünkü sunulan yöntem hem model bilgisini hem de cihaz bilgisini kullanarak iyileştirme yapmaktadır. Dolayısıyla literatürdeki cihaz bilgisini kullanan yani PRNU tabanlı yöntemler ile karşılaştırılımıştır. Literatürde PRNU yöntemini geliştiren 2 tane yöntem bulunmaktadır. Bunlardan birincisi Korus ve Huang (2017)'nin sunmuş olduğu yöntemdir. Bu yöntemde test görüntüsü, standart PRNU yöntemi ile farklı boyutlardaki pencerelerde

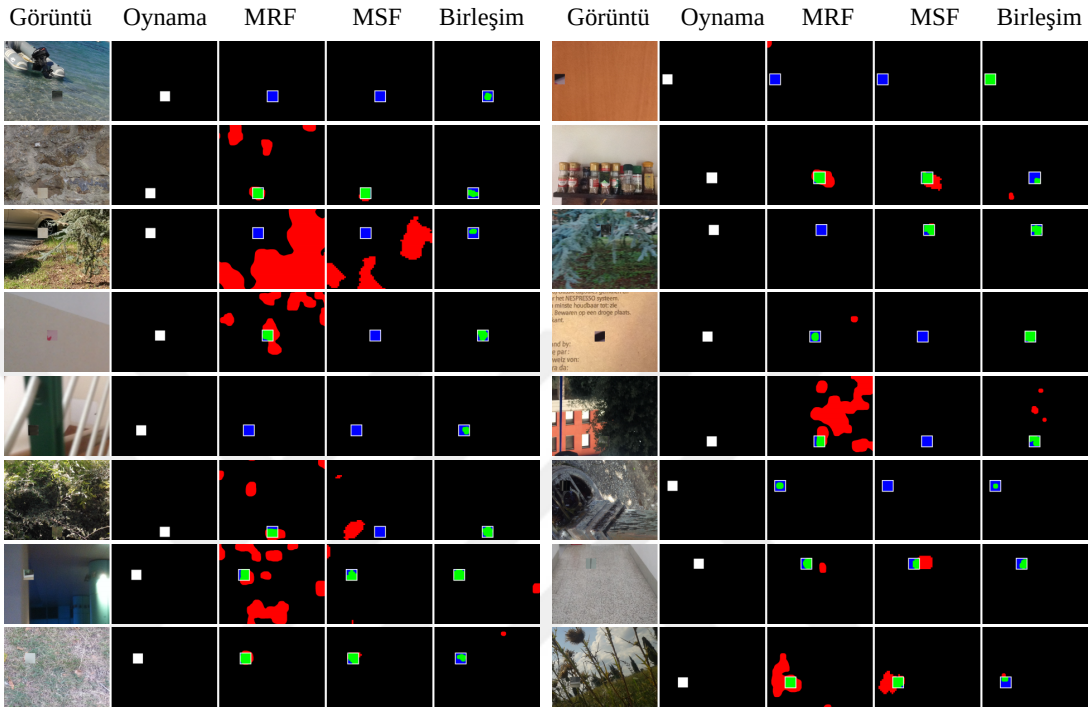


Şekil 4.8. 200×200 piksel boyutundaki oynamalar için örnekler

test edilerek birbirinden farklı aday sonuç haritaları üretmektedir. Bu aday haritaları MSF (Multi Scale Fusion) adı verilen yaklaşım ile birleştirilmektedir. Bu yöntem MSF yöntemi olarak adlandırılacaktır. Her bir kamera modeli için örnek oynama tespiti görüntüleri Şekil 4.7, 4.8 ve 4.9'de verilmiştir. Şekil 4.7, 4.8 ve 4.9'de yeşil doğru karar verilen bölgeyi, kırmızı yanlış kararı ve mavi saptanamayan bölgeyi göstermektedir.

MSF yöntemini daha detaylı açıklayacak olursak, 32×32, 48×48, 64×64, 96×96, 128×128, 192×192 ve 256×256 boyutlarındaki pencereler ile test görüntüsü üzerinde kaydırma ve korelasyon hesaplama işlemi yapılarak 7 farklı aday haritası üretmektedir. Her harita olası oynanmış bölgeler için bilgi barındırmaktadır. Büyük boyuttaki pencereler daha doğru ancak sadece büyük oynamaları tespit ederken, küçük boyutlu pencereler daha küçük oynalamarı hata payı artacak şekilde üretmektedir. Bu noktada MSF yöntemi, 7 adet aday haritayı MSF (Conditional Random Field) yaklaşımıyla birleştirilerek oldukça başarılı sonuç üreten bir yöntem olmuştur. Karşılaştırılacak olan ikinci yöntem ise PRNU yöntemini daha iyi hale getirmek için Chierchia ve ark. (2014) PRNU tabanlı bölgesel

oynama tespit edici önermiştir. Klasik PRNU yöntemine ek olarak Markov Randov Field kullanan bu yöntem her bir blok için karar vermek yerine resmin genelinde tek seferde bir karar vermektedir. Ek olarak resimden gürültü çıkarırken Lukas ve ark. (2005)'de önerilen gürültü çıkarıcıdan farklı çıkarıcı kullanmaktadır.



Şekil 4.9. 100×100 piksel boyutundaki oynamalar için örnekler

Birleşim yöntemi ile MSF ve MRF yöntemleri test edilirken S_{ts} setini oluşturan görüntüler oynanmış resim oluşturmak için seçilmiştir. 17 kamera modeli için toplamda 154 adet görüntü üzerine 400×400, 200×200 ve 100×100 boyutunda farklı kameradan kopyala yapıştır yöntemiyle oynama işlemi gerçekleştirilmiştir. Toplamda 462 adet oynanmış görüntü ile karşılaştıma işlemi yapılmıştır. Her bir model için ortalama 25 adet oynanmış görüntü karşılaştırma için kullanılmıştır. Birleşim yönteminden elde edilen sonuç haritaları belirlenen eşikleme değeriyle (bir önceki başlıkta anlatıldı) eşiklenmiştir. 3 yöntem için karşılaştırma sonuçları Çizelge 4.5'de verilmiştir. Çizelge 4.5'de verilen kamera cihazlarının (K-1, ... , K-17) detayları Çizelge 3.1'de verilmiştir. MSF Korus ve Huang (2017)'un yöntemini, MRF ise Chierchia ve ark. (2014)'in yöntemini göstermektedir.

Çizelge 4.5. Karşılaştırılan yöntemlerin oynanmış bölge boyutuna göre ortalama F1 sonuçları

	Oynanmış Bölge Boyutu								
	400×400			200×200			100×100		
Kamera #	Birleşim	MRF	MSF	Birleşim	MRF	MSF	Birleşim	MRF	MSF
K-1	0,60	0,52	**0,71**	0,33	0,27	**0,46**	**0,12**	0,03	0,07
K-2	**0,53**	0,41	0,51	**0,41**	0,30	0,33	**0,18**	0,09	0,07
K-3	**0,64**	0,16	0,39	**0,38**	0,00	0,02	**0,09**	0,00	0,00
K-4	**0,86**	0,42	0,64	**0,72**	0,23	0,44	**0,56**	0,05	0,22
K-5	0,69	0,75	**0,84**	0,58	0,58	**0,69**	**0,20**	0,07	0,13
K-6	**0,87**	0,73	0,56	**0,73**	0,47	0,50	**0,30**	0,16	0,19
K-7	**0,92**	0,69	0,82	**0,83**	0,51	0,69	0,31	0,16	**0,36**
K-8	0,64	0,78	**0,81**	0,52	0,59	**0,66**	0,13	**0,29**	0,26
K-9	0,86	0,53	**0,88**	**0,75**	0,35	0,73	**0,58**	0,03	0,23
K-10	0,79	0,76	**0,82**	0,52	0,52	**0,67**	0,07	0,32	**0,44**
K-11	**0,57**	0,48	0,55	0,30	**0,30**	0,29	**0,07**	0,00	0,01
K-12	**0,81**	0,49	0,79	**0,62**	0,26	0,53	**0,36**	0,05	0,14
K-13	**0,67**	0,49	0,57	**0,45**	0,29	0,43	**0,19**	0,15	0,15
K-14	0,71	0,51	**0,81**	**0,38**	0,21	0,37	**0,03**	0,03	0,02
K-15	**0,83**	0,76	0,77	**0,70**	0,62	0,65	**0,48**	0,38	0,30
K-16	0,59	0,54	**0,70**	0,41	0,34	**0,55**	**0,21**	0,11	0,15
K-17	**0,75**	0,33	0,53	**0,53**	0,10	0,31	0,13	0,02	**0,16**
Ortalama	**0,73**	0,57	0,70	**0,56**	0,38	0,53	**0,26**	0,13	0,19

Çizelge 4.5'de verilen karşılaştırma sonuçları, belirlenen oynama boyutuna göre ayrı ayrı listelenmiştir. Çünkü MRF ve birleşim yöntemi pencere boyutu olarak 96×96 kullanmasına karşın MSF yöntemi 32'den 256'a kadar değişen çoklu pencere boyutları kullanmaktadır. Normal şartlarda MSF yönteminin birleşim yöntemiyle karşılaştırılması adil değildir. Ancak sunulan birleşim yöntemi bu adil olmayan durumda dahi diğer yöntemlerden üstün olduğunu kanıtlamıştır. Özellikle 100×100 boyutundaki oynamalarda ortalamada yüzde MSF yöntemine göre %37, MRF yöntemine göre %92 daha başarılıdır.

5 TARTIŞMA VE SONUÇ

Bu tezde standart PRNU tabanlı kaynak cihaz tanıma yöntemiyle ESA tabanlı kamera model sınıflandırıcısı birleştirilerek, oynanmış bölge tespiti problemine daha iyi bir çözüm sunabilen yeni bir yöntem önerilmiştir. PRNU tabanlı kamera kaynak cihaz tanıma yöntemi, kamera cihazının değişmeyen sensör özelliklerinden faydalanırken, ESA tabanlı kamera model sınıflandırıcısı, kameranın mozaikleme filtresininizlerinden faydalanmaktadır. Dolayısıyla iki yöntemin, kameranın farklı özelliklerini taban alarak çalışması, bu iki yöntemin birleştirilebileceği düşüncesini ortaya çıkarmıştır. Bu tezde, kameranın 2 farklı özelliğini tek bir yöntem içerisinde kullanarak yeni bir yöntem oluşturulması hedeflenmiştir.

Önerilen yöntem, PRNU yönteminden gelen korelasyon değeri ile ESA makinesinin ürettiği olasılık değerinin birleştirilmesi olarak tasarlanmıştır. Üretilen bu değerlerin biri diğerine göre daha doğru sonuç üretiyor olabilir. Dolayısıyla hangisine daha çok önem verileceği deneme yanılma yöntemiyle değil bir YSA modeli oluşturularak bulunmuştur. Böylelikle her kamera için özel olarak eğitilen bir yapay sinir ağı en iyi sonucu üretmiştir. Çünkü bazı kameralarda parmak izi yeterince güçlü olmayabilir. Bu durumda YSA, paylaştıracağı ağırlığın büyük kısmını ESA çıktısı olan olasılık değerine verecektir. Benzer şekilde ESA modeli iyi eğitilememiş olabilir. Bu durumda ise ağırlığın büyük kısmı korelasyon değerine verilecektir.

Elde edilen sonuçlara göre birleşim yöntemi hem kendisini oluşturan PRNU ve ESA tabanlı yöntemden hem de literatürdeki PRNU tabanlı 2 yöntemden daha iyi çalışmaktadır. Literatüe ile yapılan karşılaştırma sonuçları, JPEG 95-100 kalitesindeki resimlerden elde edilmiştir. Fakat birleşim yönteminin JPEG sıkıştırmasına karşı PRNU yöntemi kadar dayanıklı olduğu gösterilmiştir. Tek başına JPEG sıkıştırmasına dayanıklı olamayan ESA makinesinin bu eksikliği PRNU yönteminin dayanıklı olması sayesinde birleşim yöntemi içerisinde giderilmiştir.

Literatürdeki 2 yöntemle karşılaştırma yapılırken 3 farklı oynama boyutu için karşılaştırma

gerçekleştirilmiştir(400×400,200×200 ve 100×100). MSF yöntemi 7 farklı boyuttaki pencereleri kullanarak sonuç üretmektedir. Dolayısıyla Önerilen yöntem ile MSF yönteminin karşılaştırılması çok da adil değildir. Ancak bu dezavantajlı konumda dahi MSF yöntemi, önerilen yöntemden daha düşük performans göstermiştir. Elde edilen sonuçlarla, kamera cihazının değişmeyen sensör özellikleri ile cihazın renk filterisindeki bilgilerin toplanarak kullanılabileceği kanıtlanmıştır.

PRNU tabanlı yöntem kameranın sensör özelliklerini baz aldığından, resim üzerinde yapılan hemen hemen bütün oynama çeşitlerinde çalışabilmektedir. Özellikle kopyala yapıştır oynama çeşidinde ve türevlerinde iyi çalışmaktadır. Çünkü oynanan bölgedeki her bir pikselin, tahmin edilen parmak izi ile eşleşmesine bakıldığından, kopyalanan bölge rahatlıkla tespit edilebilmektedir. Kopyala yapıştır oynama tipinin türevlerinde farklı kameradan (farklı model veya farklı cihaz) kopyalama veya resmin kendi içerisinden kopyalama versiyonları bulunmaktadır. PRNU yöntemi çoğu oynama probleminde çalışabilmesine karşın ESA tabanlı yöntem yalnızca farklı kamera modelinden kopyala yapıştır yapılırsa çalışabilmektedir. Çünkü ESA tabanlı sınıflandırıcı yalnızca kamera modelleri arasında bir ayrım yapabilmektedir. Bunun sebebi ise aynı kamera modellerine, üretim yapılırken aynı mozaikleme filtresinın koyulmasıdır. ESA tabanlı yöntem resim içerisinde farklı modele ait bir bölge olup olmadığını tespit ederek oynamaları bulabilmektedir. Dolayısıyla resmin kendi içerisinden bir bölgeyi, yine resmin içerisinde başka bir bölgeye yapıştırma işlemini ESA tabanlı yöntem tespit edemeyecektir. Bunun sebebi resim içerisinde farklı modele ait bir mozaikleme filtresi özelliği bulunmamasıdır. Benzer şekilde aynı model ve farklı cihazdan bir bölge resim içerisine yapıştırıldığında da ESA tabanlı yöntem ile tespit edilemez.

Sunulan birleşim yöntemi ise hem ESA tabanlı yönteme hem de PRNU tabanlı yönteme bağlıdır. Dolayısıyla birleşim yöntemi, ESA tabanlı yöntemin çalıştığı oynama çeşitlerinde çalışabilmektedir. Yani resim üzerinde hem mozaikleme filtresinin özelliğini hem de PRNU bilgisinin özelliğini bozabilecek oynama çeşitlerinde çalışacaktır.

Sunulan yöntem içerisinde hem parmak izi tahmini hem de ESA tabanlı eğitim olduğu

için kamera cihazının kendisine ya da kamera cihazından çekilmiş en az 100-150 adet resime ihtiyaç duyulmaktadır. Bu resimlerin bir kısmı ile parmak izi tahmini bir kısmı ile ESA eğitimi yapılmalıdır. Bu açıdan PRNU yönteminde olduğu gibi cihaz bilgisine ihtiyaç duyulmaktadır. Ek olarak eğer cihaza fiziki olarak erişim sağlanamıyorsa (yeni resimler çekilemiyorsa), görüntü çeşitliliğin az olması yöntemin başarısını etkilemektedir. Çünkü ESA tabanlı kamera model sınıflandırıcısı eğitilirken, kullanılması gereken görüntülerin sahne olarak birbirinden farklı olması ve aşırı karanlık ya da aşırı parlak resimlerden oluşmaması gerekmektedir. Bu açılardan PRNU yönteminden biraz daha farklılık göstermektedir. Çünkü PRNU yönteminde parmak izi tahmini yapılırken, ESA eğitiminin aksine kullanılan resimlerin düz içeriğe sahip olması (gökyüzü ve duvar gibi) parmak izi kalitesini arttırmaktadır.

Sunulan yöntem içerisindeki ESA modeli deneme yanılma yöntemiyle oluşturulmuş en yüksek başarıyı veren modeldir. Fakat bu konuda daha derin araştırma yapılması gerekmektedir. Oluşturulabilecek çok daha başarılı bir ESA yapısı ile birleşim yöntemi, kat ve kat daha yüksek performans sergileyebilir. Bu açıdan sunulan birleşim yöntemi geliştirmeye açık bir yöntemdir. Ayrıca sunulan yöntem sadece 96×96 boyutunda pencere kullanılarak test edilmiştir. Ancak bu boyutu arttırarak daha iyi çalışabilen makineler eğitilebilir. Dolayısıyla sunulan yöntem sabit bir pencere boyutu için değil, bütün pencere boyutları için geçerlidir. Burada sadece ESA modelinin seçilen pencere boyutuna göre uyumlu olması gerekmektedir. Bu tez kapsamında sunulan ESA mimarisi 96×96 piksel boyutundaki girdiler için tasarlanmıştır. Bu durum res-net gibi daha karmaşık modeller kullanılarak hangi çözünürlük olursa olsun otomatik olarak ayarlanabilir.

Gelecek çalışma olarak, yöntemin geliştirilebilmesi açısından MSF yönteminde yapıldığı gibi farklı boyutlarda pencere kullanılarak aday haritalar üretilip birleştirilebilir. Yani 7 (bu sayı değişken) farklı birleşim haritası ile mevcut birleşim yönteminden daha başarılı karmaşık bir metot geliştirilebilir. Bu şekilde işlem maliyeti çokça artsa dahi mevcut birleşim yönteminden daha iyi sonuçlar elde edilecektir.

KAYNAKLAR

Anonim, 2019. Sinir hücresi https://tr.wikipedia.org/wiki/Sinir_hücresi (Erişim tarihi: 01.04.2019).

Anonim, 2019. Hinge hatası https://en.wikipedia.org/wiki/Hinge_loss (Erişim tarihi: 01.07.2019).

Bayar, B. ve Stamm, M. C. 2016. A deep learning approach to universal image manipulation detection using a new convolutional layer. Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security. ACM, 5–10.

Bianchi, T. ve Piva, A. 2012. Detection of nonaligned double JPEG compression based on integer periodicity maps. *IEEE transactions on Information Forensics and Security* 7(2): 842–848.

Bondi, L., Güera, D., Baroffio, L., Bestagini, P., Delp, E. J. ve Tubaro, S. 2017a. A preliminary study on convolutional neural networks for camera model identification. *Electronic Imaging* 2017(7): 67–76.

Bondi, L., Baroffio, L., Güera, D., Bestagini, P., Delp, E. J. ve Tubaro, S. 2017b. First steps toward camera model identification with convolutional neural networks. *IEEE Signal Processing Letters* 24(3): 259–263.

Bondi, L., Lameri, S., Güera, D., Bestagini, P., Delp, E. J. ve Tubaro, S. 2017c. Tampering detection and localization through clustering of camera-based CNN features. 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). IEEE, 1855–1864.

Cao, G., Zhao, Y., Ni, R., Yu, L. ve Tian, H. 2010. Forensic detection of median filtering in digital images. 2010 IEEE International Conference on Multimedia and Expo. IEEE, 89–94.

Chen, M., Fridrich, J., Goljan, M. ve Lukás, J. 2008. Determining image origin and integrity using sensor noise. *IEEE Transactions on information forensics and security* 3(1): 74–90.

Chierchia, G., Poggi, G., Sansone, C. ve Verdoliva, L. 2014. A Bayesian-MRF approach for PRNU-based image forgery detection. *IEEE Transactions on Information Forensics and Security* 9(4): 554–567.

Conotter, V., Comesaña, P. ve Pérez-González, F. 2013. Forensic analysis of fullframe linearly filtered JPEG images 2013 IEEE International Conference on Image Processing. IEEE, 4517–4521.

Dirik, A. E. ve Memon, N. 2009. Image tamper detection based on demosaicing artifacts. 2009 16th IEEE International Conference on Image Processing (ICIP). IEEE, 1497–1500.

Farid, H. (2009). Exposing digital forgeries from JPEG ghosts. *IEEE transactions on information forensics and security* 4(1): 154–160.

Ferrara, P., Bianchi, T., De Rosa, A. ve Piva, A. 2012. Image forgery localization via fine-grained analysis of CFA artifacts. *IEEE Transactions on Information Forensics and Security* 7(5): 1566–1577.

Goljan, M., Fridrich, J. ve Filler, T. 2009. Large scale test of sensor fingerprint camera identification, Media forensics and security. *International Society for Optics and Photonics*.

Kirchner, M. 2008. Fast and reliable resampling detection by spectral analysis of fixed linear predictor residue. Proceedings of the 10th ACM workshop on Multimedia and security. ACM, 11–20.

Kirchner, M. ve Fridrich, J. 2010. On detection of median filtering in digital images. *Media forensics and security*

Kizrak, M. 2019. Aktivasyon grafigi çizdirme kodu. http://bit.do/ePTp3 (Erişim tarihi: 02.04.2019)

Korus, P. ve Huang, J. 2017. Multi-scale analysis strategies in PRNU-based tampering localization. *IEEE Transactions on Information Forensics and Security* 12(4): 809–824.

Krawetz, N. ve Solutions, H. F. 2007. A Picture's Worth. *Hacker Factor Solutions* 6.

Kurita, T. 2019. CNN Yapısı. https://bit.ly/2UQ39mp (Erişim tarihi: 02.05.2019)

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., ve ark. 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11) 2278–2324.

Li, W., Yuan, Y. ve Yu, N. 2009. Passiand detection of doctored JPEG image via block artifact grid extraction. *Signal Processing* 89(9): 1821–1829.

Lin, Z., He, J., Tang, X. ve Tang, C.-K. 2009. Fast, automatic and fine-grained tampered JPEG image detection via DCT coefficient analysis. *Pattern Recognition* 42(11): 2492–2501.

Lukas, J., Fridrich, J. ve Goljan, M. 2005. Determining digital image origin using sensor imperfections. *Image and Video Communications and Processing 2005*. 249–261.

Lukáš, J., Fridrich, J. ve Goljan, M. 2006. Digital camera identification from sensor pattern noise. *IEEE Transactions on Information Forensics and Security* 1(2): 205–214.

Mahdian, B. ve Saic, S. 2009. Using noise inconsistencies for blind image forensics. *Image and Vision Computing* 27(10): 1497–1503.

Popescu, A. C. ve Farid, H. 2005. Exposing digital forgeries by detecting traces of resampling. *IEEE Transactions on signal processing* 53(2): 758–767.

Shullani, D., Fontani, M., Iuliani, M., Al Shaya, O. ve Piva, A. 2017. VISION: a video and image dataset for source identification. *EURASIP Journal on Information Security* 2017(1): 15

Simonyan, K. ve Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *ArXiv preprint arXiv*:1409.1556.

Thai, T. H., Cogranne, R., Retraint, F., ve ark. 2017. JPEG quantization step estimation and its applications to digital image forensics. *IEEE Transactions on Information Forensics and Security* 12(1): 123–133.

Tuama, A., Comby, F. ve Chaumont, M. 2016. Camera model identification with the use of deep convolutional neural networks. *2016 IEEE International workshop on information forensics and security (WIFS)*. IEEE, 1–6.

Ye, S. Sun, Q. ve Chang, E.-C. 2007. Detecting digital image forgeries by measuring inconsistencies of blocking artifact. 2007 IEEE International Conference on Multimedia and Expo. IEEE, 12–15.

ÖZGEÇMİŞ

Adı Soyadı	: Ahmet Gökhan POYRAZ
Doğum Yeri ve Tarihi	: Elazığ, 1991
Yabancı Dil	: İngilizce

Eğitim Durumu (Kurum ve Yıl)

Lise	: Ulubatlı Hasan Anadolu Lisesi, 2009
Lisans	: Anadolu Üniversitesi Elektrik-Elektronik Mühendisliği, 2015

İletişim : agpoyraz@gmail.com

Tezden Yapılan Yayınlar

Poyraz, A. G. 2019. Sayısal imgeler için PRNU ve CNN tabanlı bölgesel müdahale tespiti. *Uludağ University Journal of The Faculty of Engineering*. (Kabul Tarihi: 30.05.2019)

Poyraz, A. G., Dirik, A. E., Karaküçük, A., Memon, N. 2019. PRNU based Tamper Detection with Convolutional Neural Networks. *Multimedia Tools and Applications* (Gönderim Tarihi: 30.04.2019)