DOKUZ EYLÜL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MAKİNE ÖĞRENMESİ YÖNTEMLERİ İLE VİDEO GÖRÜNTÜLERİNDEKİ FİZİKSEL ŞİDDETİN TESPİT EDİLMESİ

Muhammet Fatih POLAT

Eylül, 2022
İZMİR

MAKİNE ÖĞRENMESİ YÖNTEMLERİ İLE VİDEO GÖRÜNTÜLERİNDEKİ FİZİKSEL ŞİDDETİN TESPİT EDİLMESİ

Dokuz Eylül Üniversitesi Fen Bilimleri Enstitüsü

Yüksek Lisans Tezi

İstatistik Anabilim Dalı, Veri Bilimi Yüksek Lisans Programı

Muhammet Fatih POLAT

Eylül, 2022

İZMİR

YÜKSEK LİSANS TEZİ SINAV SONUÇ FORMU

MUHAMMET FATİH POLAT tarafından **PROF. DR. AYLİN ALIN** yönetiminde hazırlanan "**MAKİNE ÖĞRENMESİ YÖNTEMLERİ İLE VİDEO GÖRÜNTÜLERİNDEKİ FİZİKSEL ŞİDDETİN TESPİT EDİLMESİ**" başlıklı tez tarafımızdan okunmuş, kapsamı ve niteliği açısından bir Yüksek Lisans tezi olarak kabul edilmiştir.

……..

Prof. Dr. Aylin ALIN

Yönetici

……..	……..
Doç. Dr. Ufuk BEYAZTAŞ	Dr. Öğr. Üyesi Barış Tekin TEZEL
----------	**----------**
Jüri Üyesi	Jüri Üyesi

Prof. Dr. Okan FISTIKOĞLU

Müdür

Fen Bilimleri Enstitüsü

TEŞEKKÜR

Tez çalışmamın yürütülmesinde, yardımlarını esirgemeyen ve sabırla çalışmama destek sağlayan danışmanım ve hocam sayın Prof. Dr. Aylin Alın'a, eğitim hayatım boyunca bana rehberlik eden kardeşim Gül Ece Hızal'a ve küçüklüğümden itibaren bilime olan merakımı destekleyen annem Dilek Polat'a teşekkürü bir borç bilirim.

Muhammet Fatih POLAT

MAKİNE ÖĞRENMESİ YÖNTEMLERİ İLE VİDEO GÖRÜNTÜLERİNDEKİ FİZİKSEL ŞİDDETİN TESPİT EDİLMESİ

ÖZ

Video görüntülerinin şiddet içerip içermemesinin bulunmasına yönelik birçok çalışma yapılmıştır. Yapılan çalışmalarda yüksek doğruluk oranlarında sonuçlar alınmıştır. Bu tez kapsamında da şiddet içerikli video sınıflandırılması amacıyla evrişimli sinir ağları ve uzun-kısa süreli bellekten oluşan bir sistem oluşturulmuştur. Ayrıca bu sistemi aktifleştirmesi için de bir tetikleyici fonksiyon önerisinde bulunulmuştur. Önerilen yöntem hedeflenen doğrultuda sonuçlar üretmiştir. Şiddet eylemlerinin sınıflandırılmasında *Real Life Violence Situations* veri seti kullanılmıştır.

Anahtar kelimeler: Şiddet tespiti, derin öğrenme, aşırı hareket

VIDEO BASED PHYSICAL VIOLATION DETECTION USING MACHINE LEARNING METHODS

ABSTRACT

Many studies have been made to find out whether video images contain violence or not. Studies have obtained results with high accuracy rates. Within the scope of this thesis, a system consisting of convolutional neural networks and long-short term memory was created in order to classify violent video. In addition, a trigger function has been suggested to activate this system. The proposed method produced results in the targeted direction. The Real-Life Violence Situations data set was used for the classification of acts of violence.

Keywords: Violence detection, deep learning, extreme movement

İÇİNDEKİLER

Sayfa

ŞEKİLLER LİSTESİ

Sayfa

TABLOLAR LİSTESİ

Sayfa

KISALTMALAR

AI	: Artificial intelligence
BiConvLSTM	: Bidirectional convolutional long-short term memory
H&E	: Hematoksilen ve eozin
CRF	: Conditional random field
MAP	: Maksimum a posteriori
mAP	: Mean average precision
MSE	: Mean squared error
AdaGrad	: Adaptive gradient
RMSProp	: Root Mean Squared Propagation
Adam	: Adaptive moment
GPU	: Graphical processing unit
API	: Application programming interface
BSD	: Berkeley Software Distribution
2D	: İki boyutlu
BRNN	: Bidirectional recurrent neural network
GRU	: Gated recurrent unit
LSTM	: Long-short term memory
ReLU	: Rectified linear unit
LRN	: Local response normalization
ILSVRC	: ImageNet Large Scale Visual Recognition Challenge
ConvLSTM	: Convolutional + LSTM neural network

BÖLÜM 1
GİRİŞ

Geçmişten bugüne şiddet insanlar ile birlikte süregelmiştir. Modern toplum içerisinde şiddet yasalar ile yasaklanmıştır. Bu yasaklara uymayan bireyler, toplum için birer tehdit olmaktadır. Bu tehditlerin otonom sistemler ile tespit edilmesi günümüzde mümkün hale gelmiştir. Bu tezde önerilen aşırı hareketi tespit etmesi için kullanılan öncül fonksiyonun yanı sıra aşırılıktan önce hareketin tespitine yönelik çalışmalardan bazıları şunlardır;

Tian ve Hanpapur (2005), videolardaki ilginç hareketleri tespit eden bir algoritma önerisinde bulunmuşlardır. Bu algoritma arka plan çıkartımı yönteminde olduğu gibi eğitilmeye ihtiyaç duymamaktadır. Ayrıca nesne hakkında öncül bilgi sahibi olması da gerekmemektedir. Zhan ve arkadaşları (2007), video kareleri arasındaki farka ve kenar tespitine dayalı bir hareketli obje tespit algoritması önermişlerdir. Bu algoritma iki video karesinde *Canny* dedektörü ile kenar tespiti yapmakta ve kenar tespiti yapılmış karelerin farkını almaktadır. Bu fark görüntüsü küçük bloklara bölünmekte ve bir eşik işleminden geçmektedir. Bu işlemler sonucunda da hareket tespit edilmektedir. Cheng, Huang ve Ruan (2010), zamansal ve mekânsal öğelerden faydalanarak bir arka plan modeli geliştirmişlerdir. Bu model sonucunda ikili tabanda bir hareket maskesi oluşturulmaktadır. Bu yöntem, diğer yöntemlerden daha iyi bir sonuç vererek F1 skorunda %79'a yakın bir doğruluk oranına ulaşmaktadır. Du ve arkadaşları (2010), optik akış ve temel bileşenler analizini beraber kullanarak bir yöntem önerisinde bulunmuşlardır. Burada temel bileşenler analizi optik akışlar içerisinden hareket eden objelere ait büyük akışların yerel pencerede daha iyi çıkartılmasında rol oynamaktadır. Bu yöntem, dış ortam ve düşük kalite videolarda kullanışlı olduğu gözlemlenmiştir. Ayrıca, durağan ve hareketli arka planlarda da çalışabilmektedir. Huang (2011), otomatik güvenlik kameraları için bir hareket tespiti yaklaşımı önerisinde bulunmuştur. Bu yaklaşım, üç ana modülden oluşmaktadır. Bunlar, arka plan modeli modülü, alarm tetikleyici ve nesne çıkartımı modülüdür. Bu yaklaşım ile beraber %53,43'e varan F1 doğruluk oranına ulaşılmıştır. Li, Ng ve Yuan (2015), bozulmaya uğramış veya gürültü içeren video görüntülerinden durağan arka plan

çıkartımı için bir yöntem önerisinde bulunmuşlardır. Bu yöntem, matris gösteriminde durağan arka planın birebir aynı sütunlara sahip olmasına dayanmaktadır. Daha önceden yapılan çalışmalarda kullanılan dayanıklı temel bileşenler analizinde karşılaşılan hesaplama bakımından fazla kaynak tüketen tekil değer ayrışımının etkisi bu yöntemdeki iteratiflik ile aşılabilmektedir.

Video görüntülerindeki fiziksel şiddetin tespitine yönelik farklı makine öğrenimi yöntemleri ve bu yöntemlerin farklı kombinasyonları kullanılmıştır. Yapılan bu çalışmalardan bazıları aşağıda kısaca açıklanmaktadır.

Horhan ve Eidenberger (2013), video görüntülerinde gerçek şiddet eylemi ile savunma sporlarını ayırmaya yönelik iki aşamalı makine öğrenmesi yöntemi üzerinde çalışmışlardır. Bunlara ek olarak seğirme tespitine ve yerel ilgi noktalarına yönelik iki dönüşüm önerisinde bulunmuşlardır. Çalışma sonucunda kabul edilebilir düzeyde kesinlik ve duyarlılık değerleri elde edilmiştir. Deniz ve arkadaşları (2014), aşırı hızlanma mekanizmasını ana özellik olarak kullandıkları bir yöntem önerisinde bulunmuşlardır. Aşırı hızlanmanın etkili bir biçimde yaklaşımı için de Radon dönüşümünden yararlanmışlardır. Goto ve Aoki (2014), görsel ve ses özelliklerinin bir kombine olarak kullanılan bir yapıyı temel alan makine öğrenmesi sistemi üzerine bir yöntem geliştirmişlerdir. Bu çalışmada çoklu çekirdek öğrenme yöntemini kullanmışlardır. Bununla birlikte yaptıkları çalışma etiketleme için bir iş yükü oluşturmayan bir çeşit kümeleme yöntemidir. Bu çalışma kapsamında da *MediaEval 2013 Affect Task* veri setini kullanmışlardır. Hassner, Itcher ve Kliper-Gross (2014), kalabalık insan gruplarının içinde bulunduğu video görüntülerindeki şiddetin tespiti için bir yöntem önerisinde bulunmuşlardır. Bu yöntemde, optik akış yöntemleri ve lineer destek vektör makineleri ile bir sınıflandırıcı oluşturmuşlardır. Bu yöntemi de kendi oluşturdukları veri seti üzerinde kullanmışlarıdır. Schedi ve arkadaşları (2015), bir *VSD2014* adında bir veri seti ortaya koymuşlardır. Bu veri seti içerisinde Hollywood filmlerinden ve internet üzerindeki birçok video kaynağından görüntüler içermektedir. Bilinski ve Bremont (2016), çalışmalarında video görüntülerinde şiddetin varlığı ile şiddetin ne zaman başladığı konularını ele almışlardır. Çalışma kapsamında geliştirilmiş Fisher vektörlerinden yararlanmışlardır ve bu yönteme bir

eklenti önerisinde bulunmuşlardır. Arceda ve arkadaşları (2016), optik akış için iteratif yeniden ağırlıklandırılmış en küçük kareler yöntemi yerine Horn-Schunk yöntemini ve sınıflandırma için de destek vektör makinelerini kullandıkları bir sistem geliştirmişlerdir. Geliştirdikleri bu sistem her veri seti üzerinde olmasa da bazı veri setlerinde iteratif yeniden ağırlıklandırılmış en küçük kareler yönteminden daha yüksek doğruluk oranına sahip olduğu görülmüştür. Sudhakaran ve Lanz (2017), şiddet içerikli videoların tespit edilmesine yönelik olarak önerdikleri yöntemde görüntü içerisindeki özelliklerin çıkartımı için evrişimli sinir ağlarını kullanmışlardır. Bu özellik çıkartımı sonrasında ise uzun-kısa süreli bellek ile şiddetin varlığını tespit etmişlerdir. Ayrıca bu çalışma normal video karelerinin üzerinde yaptıkları gibi ardışık video karelerinin farkları üzerinde de gerçekleştirmişlerdir. Kare farkları üzerinde yaptıkları çalışmada %97,1±0,55 sınıflandırma doğruluk oranına ulaşmışlardır. Normal video kareleri üzerinde ise %96±0,35 doğruluk oranına ulaşılmıştır. Ha ve arkadaşları (2018), dört aşamalı bir sistem ortaya koymuşlardır. Bu sistem, arka plan çıkartımı, nesne tespiti ve hareket vektörlerinin analizi üzerine kurulmuştur. Zou ve arkadaşları (2018), geniş alanlara ait video görüntülerindeki şiddetin tespiti için sinirsel-bulanık bir model önerisinde bulunmuşlardır. Bu modeli, *VSD2014* ve *UCR-Videoweb* veri setleri üzerinde test etmişler ve umut verici sonuçlar bulmuşlardır. Zhou ve arkadaşları (2018), aşamalı bir şiddet tespiti sistemi geliştirmişlerdir. Öncelikle video görüntüsünde yönlü eğimin yerel histogramı ve optik akışın yerel histogramı ile düşük seviyeli özellikler ve optik akış özellikleri çıkartılmaktadır. Ardından kelime çantası yöntemi ile gereksiz bilgiler elenmesi ile destek vektör makineleri ile sınıflandırma işlemi gerçekleştirilmektedir. Dorogyy, Kolisnichenko ve Levchenko (2018), farklı tipte verilerden yararlanarak bir şiddet tespiti sistemi önerisinde bulunmuşlardır. Bu sistem, yüz ifadelerinden duygu tespiti, yüz tanıma ve eşleştirme ile kişilerin sağlık ve sabıka kaydı bilgilerini sisteme veri girdisi gibi çeşitli özelliklere sahiptir. Ditsanthia, Pipanmaekaporn ve Kamonsantiroj (2019), kapalı devre televizyon kamerasından gelen görüntüler içerisinde şiddet tespiti için bir sistem önerisinde bulunmuşlardır. Bu sistem öncelikli olarak evrişimli sinir ağları ile görüntü özelliklerini çıkarmakta ardından çift yönlü uzun-kısa süreli bellek ile zamansal özellikleri çıkartarak bir sınıflandırma yapmaktadır. Bu çalışmada evrişimli sinir ağı yapılarından *ResNet50*, *VGG19* ve *Xception*'ı kullanmışlardır. *The Movie* veri seti

üzerinde en yüksek doğruluk oranına (%88,74) *ResNet50* ile ulaşmışladır. Hanson ve arkadaşları (2019), evrişimli çift yönlü uzun-kısa süreli bellek (*BiConvLSTM*) mimarisi üzerinde inşa edilmiş uzay-zamansal bir kodlayıcı önermişlerdir. Bu yöntem, her iki zamansal yönde uzun-dönemli (*long range*) bilgilerden yararlanmak üzerine kuruludur. Çalışma sonucunda *Hockey Fights* veri seti üzerinde %96,96 ± 1,08, *Violent Flows* veri setinde %92,18 ± 3,29 ve *The Movies* veri setinde %100,0 doğruluk oranına ulaşılmıştır. Ullah ve arkadaşları (2019), üç aşamalı bir yöntem önerisinde bulunmuşlardır. Bu yöntemde yararlı olmayan kareler daha hafif bir evrişimli sinir ağı ile insan tespiti yapılarak çıkartılmaktadır. Böylece işlenmesi gereken veri miktarı azalmaktadır. Tespit edilen insanların bulunduğu 16 karelik diziler *softmax* sınıflandırıcısına sahip 3 boyutlu evrişimli sinir ağına aktarılır. Yapılan sınıflandırma çalışmaları sonucunda, *Violent Crowd* veri seti üzerinde %98, *Violence in Movies* veri seti üzerinde %99,9 ve Hockey Fights veri seti üzerinde %96 doğruluğa ulaşılmıştır. Abdali ve Al-Tuma (2019), gerçek zamanlı olarak kullanılabilecek bir model önerisinde bulunmuşlardır. Bu modelde mekansal özellik çıkarıcı olarak evrişimli sinir ağı ve zamansal ilişkiyi çıkarmak amacıyla da uzun-kısa süreli bellek kullanılmıştır. Yapılan çalışma sonucunda *Hockey Fights* veri seti üzerinde saniyede 131 kare hızla %98 doğruluğa ulaşılmıştır. Li ve arkadaşları (2019), güvenlik kameralarında kaydedilen şiddet görüntülerinin sınıflandırılmasına yönelik bir model geliştirmişlerdir. Bu model, genel olarak iki boyutlu görüntülerdeki özelliklerin çıkarımında kullanılan evrişimli sinir ağını üç boyutlu olarak düzenlenerek oluşturulmuştur. Bu model, daha önce yapılan çalışmalarda olduğu gibi özyineli ağlara ve farklı tarzda temel özellik çıkartıcı unsurlara ihtiyaç duymamaktadır.

Bu tez kapsamında aktarımlı öğrenme için kullanılan VGG-19 ve *EfficientNet-B0* evrişimli sinir ağları ile oluşturulmuş sınıflandırma modellerinin karşılaştırılması doğruluk oranı metriği kullanılarak yapılmıştır. Literatür araştırması sırasında bu iki sinir ağının karşılaştırıldığı bir çalışmaya rastlanmamıştır. Karşılaştırma sonucunda da *EfficientNet-B0* ile oluşturulmuş sınıflandırıcının daha başarılı olduğu ortaya çıkmıştır. Sınıflandırıcıların karşılaştırılması dışında bu tez kapsamında fiziksel şiddet tespiti sistemleri için öncül fonksiyon önerisinde bulunulmuştur. Zamana bağlı bir tür varyans

işlemi ile video görüntülerindeki aşırı hareket olarak nitelendirilebilecek durumların tespitine yönelik bir geliştirme yapılmıştır.

Bu bölümün devamında, bilgisayarlı görüden, görüntü işlemeden, sinir ağlarından ve onların kullanım alanlarından bahsedilecektir. Materyal ve Yöntem bölümünde geliştirme yapılan ortam, kullanılan programlama dili, çalışmanın temelinde yer alan yazılım kütüphaneleri, deneyin yapımında kullanılan veri seti ve çalışmada kullanılmış olan sinir ağı yöntemleri tanıtılmıştır. Üçüncü bölümde numerik çalışmanın bulgularının tartışılmasının ardından tez Sonuçlar bölümü ile sonlandırılacaktır.

1.1 Bilgisayarlı Görü

Bilgisayarlı görme, bilgisayarların ve diğer hesaplayıcı içeren sistemlerin dijital görüntü ve görsellerden veya videolardan anlamlı bilgiler türetmesini ve bu bilgilere dayalı olarak eylem ve önerilerde bulunmasını sağlayan bir yapay zekânın (AI) bir alt araştırma alanıdır. Yapay zekâ, bilgisayarların insan beynini taklit ederek düşünmesini sağlamaktadır. Bilgisayarlı görme ise onların görmelerini, gözlemlemelerini ve anlamalarını sağlamaktadır (IBM, b.t.). Bilgisayarlı görme, makineleri bu işlevleri yerine getirmek için eğitmektedir, ancak insanlar gibi retinalar, optik sinirler ve görsel bir korteks yerine bu işlemi kameralar, veriler ve algoritmalar ile kısa sürede yapmaktadır. Örneğin, ürünleri denetlemek veya bir üretim varlığını izlemek için eğitilmiş bir sistem, bir dakika içinde binlerce ürünü veya süreci analiz ederek, algılanamayan kusurları veya sorunları fark edebildiğinden, insan yeteneklerini hızla aşmaktadır. Bilgisayarlı görme, tarımdan imalata birçok farklı sektörde kullanılmaktadır (IBM, b.t.).

1.2 Görüntü İşleme

Görüntü işleme, temelinde iki boyutlu olan verilerin işlenmesi ve değişiklere uğratılması ile ilgilenmektedir. Görüntü işleme teknikleri birçok alanda uygulanmaktadır. Bu alanlara görüntü iyileştirme, resimsel örüntü tanıma (*Image Pattern Recognition*) ve aktarım veya depolama için resimlerin verimli kodlanması olarak örneklendirilmektedir. Görüntü işleme alanındaki bazı teknikler aşağıda sunulmuştur.

1.2.1 Görüntü Dönüştürme

Bir görüntüye bir etki alanından diğerine dönüştürmek için bir görüntü dönüşümü uygulanabilmektedir. Fourier dönüşümü, iyileştirme, restorasyon, kodlama ve tanımlama amacıyla görüntülere uygulanabilen bazı iki boyutlu operatörler dışında, yaygın bir yöntem olarak kabul görmektedir. Bir görüntüyü frekans veya Hough alanı gibi alanlarda görüntülemek, uzamsal alanda kolayca tespit edilemeyen özelliklerin de tanımlanmasını sağlar. Yaygın görüntü dönüşümleri şunları içerir:

- Bir görüntüdeki çizgileri bulmak için Hough Dönüşümü kullanılmaktadır.
- Radon Dönüşümü, fan ışını ve paralel ışın projeksiyon verilerinden görüntüleri yeniden oluşturmak için kullanılmaktadır.
- Görüntü ve video sıkıştırmada Ayrık Kosinüs Dönüşümü kullanılmaktadır.
- Filtreleme ve frekans analizinde Ayrık Fourier Dönüşümü kullanılmaktadır.
- Ayrık dalgacık analizi, gürültü giderme ve sigorta görüntüleri gerçekleştirmek için Dalgacık Dönüşümü kullanılmaktadır (MATLAB & Simulink, b.t.).

1.2.2 Görüntü İyileştirme

Huang ve arkadaşlarının (1971) yaptıkları çalışmaya göre görüntü iyileştirme, bir görüntüyü sonucun belirli bir uygulama için orijinal görüntüden daha uygun olacak şekilde işlenmesidir. Bu kategorideki teknikler genelde problem odaklıdır. Hiçbir görüntüleme sistemi mükemmel kalitede görüntüler vermemektedir. Görüntü iyileştirmede, görüntü kalitesinin arttırılması amacıyla görüntüde değişiklikler gerçekleştirilmektedir. Bu değişikliklere örnek olarak, ekinleri sınıflandırmak için yapılan bir hava keşfinde veya gezegenleri atmosferden fotoğraflanırken elde edilen resimler atmosferik türbülans, optik sistemdeki sapma ve kamera ile nesne arasındaki bağıl hareket nedeniyle bozulabilmektedir. Tıp alanında ise genellikle düşük çözünürlüklü radyograflarda birçok elektron mikrografı elektron merceğindeki küresel sapma nedeniyle bozulmaktadır. Bu örnekler ve bunlar gibi durumlarda görüntünün kalitesinin arttırılması için görüntü iyileştirme üzerinde çalışılmasının gerekliliği söz konusu olmaktadır.

1.2.3 Görüntü Restorasyonu

Mishra ve arkadaşlarının (2019) yaptığı çalışmada yer alan bilgilere göre görüntü restorasyon teknikleri, bozuk bir görüntüyü yüksek bir kalitede yeniden oluşturmaya çalışmak olduğundan görüntü iyileştirme tekniklerine benzerlik göstermektedir. Görüntü analizi, dijital medya restorasyonu, astronomik görüntüleme ve tıbbi görüntüleme gibi farklı alanlarda görüntü restorasyonu kapsamında çeşitli uygulamalar yapılmaktadır. Bu uygulamalarda yakalama ve kayıt sırasında istenmeyen etkiler tarafından oluşturulan kusurlar görüntünün bozulmasına ve verilerdeki (görüntü) bilginin azalmasına neden olmaktadır. Görüntülerin birçoğunda görüntü üzerindeki kusurların işlenmesi ve düzeltilmesi yapılması istenen görevin başarı ile gerçekleştirilmesi konusunda kritik öneme sahiptir. Görüntü üzerindeki kusurlar veya bozulmalar, bulanık görüntü ve/veya sensörlerde oluşan gürültüden kaynaklanabilmektedir. Ayrıca kameranın nesneye göre olan hareketi, rastgele atmosferik türbülans ve yanlış kamera odaklaması neden olabilmektedir. Bu nedenle, görüntü restorasyonu, bulanıklığı ve gürültünün modelleyerek görüntü bulanıklığını ve paraziti gidermek amacıyla tersine modellemeye (*inverse modelling*) odaklanmaktadır.

Yine Mishra ve arkadaşlarının (2019) yaptığı çalışmaya göre görüntü restorasyon işlemi, belli yöntemler ile görüntüyü yeniden oluşturmak veya kurtarmak için kullanılmaktadır. Görüntüdeki bulanıklık ve gürültü, görüntünün bozulmamış halinin tahminlenmesi ve yeniden oluşturulması ile belli oranlarda giderilebilmektedir. Görüntü üzerinde yapılacak işlemler ve restorasyon süreci, görüntü üzerinde bulunan gürültü ve bozulmanın yapısına bağlı olarak değişiklik gösterecektir.

1.2.4 Görüntü Bölütleme (Segmentasyon)

Abadi ve arkadaşlarının yaptıkları çalışmaya (2015) göre bir görüntü sınıflandırma görevinde yapay sinir ağı, her bir görüntü girdisine bir etiket (sınıf) atanmaktadır. Bunun yanı sıra herhangi bir pikselin görüntü içerisindeki hangi nesneye ait olduğunun bulunması görevinde, her bir piksel bir etiket (sınıf) atanması gerekmektedir. Bu görevin adı görüntü segmentasyonu (bölütleme) olarak

adlandırılmaktadır. Görüntü segmentasyonu, tıbbi görüntüleme, sürücüsüz otonom araçlarda ve uydu görüntüleme gibi farklı uygulama alanlarına sahiptir.

Janai ve arkadaşlarına göre (2017) görüntü bölütleme tekniği, bir görüntüdeki piksellerin özelliklerine dayalı olarak birden çok bölgeye ayırmak için dijital görüntü işleme ve analizinde kullanılmaktadır. Görüntü segmentasyonu, ön planın (*foreground*) arka plandan (*background*) ayrılmasını veya renk ve şekildeki benzerliklere dayalı olarak belirli alandaki piksellerin kümelenmesi işlemlerini de içermektedir. Uygulama alanına özgü bilgiler kullanılarak segmentasyon problemlerinin çözümü için çeşitli algoritmalar ve teknikler geliştirilmiştir. Bu algoritmaların uygulama alanlarına örnek olarak, tıbbi görüntüleme, otonom sürüş, video takibi ve makine görüşü verilebilir. Medikal uygulama örneklerinden biri dokuların segmentlere ayrılmasıdır. Vücut dokusu kanserin tıbbi teşhisi esnasında patologlar tarafından hematoksilen ve eozin (H&E) ile boyanmaktadır. Görüntülerde doku türlerinin tanımlanması için görüntü bölütlemede kullanılan bir kümeleme algoritması kullanılmaktadır. Kümeleme algoritması ile görüntü üzerindeki nesne grupları ayrıştırılmaktadır.

Yine Janai ve arkadaşlarının (2017) yaptığı çalışmaya göre bir çeşit görüntü bölütleme yöntemi olan semantik segmentasyon, sürücüsüz arabalar gibi otonom araçlarda sistemin diğer araçları ve yol üzerinde veya kenarındaki nesneleri tanımlamasına ve tespit etmesinde kullanılmaktadır. Semantik segmentasyon, bilgisayarlı görü alanında (*computer vision*) temel problemlerden bir tanesidir. Çözülmeye çalışılan bu problem, görüntü anlamlandırılması ve sensörlere dayalı motor kontrolü gibi daha yüksek seviyeli problemlerin çözümüne yönelik bir ara hedeftir. Semantik segmentasyon, görüntü üzerindeki her piksele önceden tanımlanmış kategoriler kümesi içerisinden bir etiket atanmasını amaçlamaktadır. Görüntüler, araba, yaya veya yol gibi tipik bir sokak sahnesinde bulunabilen nesneler için anlamlı bölgelere ayrılarak, otonom navigasyonda çevrenin kapsamlı şekilde anlaşılmasına olanak sağlamaktadır. Bu görevin zorluğu sahnenin karmaşıklığı, karmaşık nesne sınırları, küçük nesneler ve büyük etiket alan boyutları nedeniyle yüksektir. Semantik segmentasyonda amaç, bir görüntüye ait her pikselin semantik olarak anlamlı bir etiket

değerinin (örneğin yol, kaldırım, yaya, gökyüzü gibi) atanmasıdır. Geleneksel olarak bu problemin çözümü için pikseller üzerinden tanımlanan bir koşullu rastgele alanda (*CRF*) maksimum a posteriori (*MAP*) çıkarımı olarak tanımlanmaktadır. Son yıllarda görüntü sınıflandırma ve nesne algılama problemlerinin çözümünde derin evrişimli sinir ağlarının yüksek tahmin başarısı nedeniyle semantik segmentasyon gibi piksel düzeyindeki görevlerin çözümünde de kullanılmaktadır. Evrişimli sinir ağları görüntü sınıflandırması için çözünürlüğü düşüren ardışık havuzlama (*Pooling*) ve alt örnekleme katmanları ile çok ölçekli bağlamsal bilgileri birleştirmektedir. Bununla birlikte semantik segmentasyon, tam çözünürlükte tahmin ve çok ölçekli bağlamsal ilişkilendirme üzerine kurgulandığından bu problemin çözümü için evrişimli sinir ağları kullanımını mümkün kılmaktadır.

1.2.5 Özellik Çıkartımı

Morfolojik, dokusal, fraktal ve yoğunluk tabanlı yöntemlerdir. Tanıma ve yorumlama yöntemleri, otomatik görüntü analizi gerektiren uygulamalar için özellikle uygundur ve genellikle bir dereceye kadar yapay zekâ içermektedir.

1.2.6 Örüntü Tanıma

Örüntü tanıma, verilerdeki desenleri otomatik olarak tanımak için makine öğrenimi algoritmalarını kullanan bir görüntü analizi yöntemidir. Örüntü tanıma sistemleri, sisteme tanımlı desenleri hızlı ve doğru şekilde tanımlayabilmektedir. Bunun yanı sıra tanıdık olmayan nesneleri tanıyabilmekte ve sınıflandırabilmektedir. Şekilleri ve nesneleri farklı açılardaki görünümlerini ve belli bir kısmının göründüğü durumlarda da tanıma işlemini gerçekleştirebilmektedir (Arm®, b.t.).

1.2.7 Nesne Tespiti

Chahal ve Dey'in (2018) yaptıkları çalışmaya göre nesne algılama veya diğer adıyla nesne tespiti, bir görüntüdeki bir nesnenin konumu ve sınıflandırması ile birlikte tanımlanmasıdır. Bu görevleri gerçekleştiren sistemlere nesne detektörleri denir. İnsan denetimi gerektiren çok sayıda görev, görüntülerdeki nesneleri algılayabilen bir yazılım sistemi ile otomatik hale getirilebilmektedir. Bu sistemler video izleme,

hastalık tanımlama ve otonom araç sürüşü alanlarında da kullanılmaktadır. Derin öğrenme tabanlı nesne detektörleri, ilk örneklerinin sunulduğu 2015 yılından itibaren doğruluk, hız ve bellek ayak izi konularında ilerleme kaydetmiştir. Derin öğrenme nesne detektörlerinin ilk örneklerinden birisinin bir görüntüyü işlemesi 47 saniye sürmüştür. Son yıllardaki çalışmalarla bu süre 30 ms'yeye indirilmiştir. Hız ile birlikte doğruluk üzerinde de iyileşmeler gerçekleşmiştir. İlk örneklerde 29 mAP algılama doğruluğu alınırken günümüzde nesne detektörleri 43 mAP'ye ulaşmıştır. Nesne detektörlerinin boyutları, modellerin tasarımı sayesinde düşük güçlü telefonlarda da çalışabilmektedir. Tensorflow ve Caffe gibi yazılım geliştirme çerçeveleri (*framework*) telefonlarda model çalıştırma desteğini sunmaktadır.

1.2.8 Hareket Tespiti

Kurylyak (2009), hareket tespitini, ardışık video karelerinde oluşan anlamlı veya anlamsız değişikliklerin tespiti olarak tanımlamaktadır. Bu problemin çözümü için önerilen yöntemlerden bazıları; zamansal fark (*Temporal Difference*), optik akış (*Optical Flow*) ve arka plan çıkartımı (*Background Subtraction*) olarak verilebilir. Arka plan çıkartımı, yüksek performansı ve düşük bellek gereksinimi nedeniyle sabit statik kameralarda en yaygın kullanılan yöntemdir. Arka plan çıkartımı sistemi, arka planı modellemeyi ve mevcut görüntü ile karşılaştırmaktadır. Bu sayede ön plandaki hareketli nesnelerin tespiti gerçekleşmektedir. Bu yöntemin güvenilir olabilmesi için arka plan ve kameranın sabit olması gerekmektedir.

1.2.9 Optik Akış

Lee ve Bovik'in (2009) tanımlamasına göre optik akış (*Optical-flow*) veya hareket tahmini (*Motion Estimation*), nesnelerin video görüntüsü üzerinde hareketten kaynaklı denilebilecek görüntü yoğunluklarının hareketini hesaplamanın temel bir yöntemidir. Turaga ve arkadaşları (2010) ise optik akışı, görüntü düzlemindeki her pikselin gözlemlenebilen hareketi veya görüntü düzlemine yansıtılan fiziksel hareketin bir tahmini olarak tanımlamaktadır. Optik akışı hesaplayan çoğu yöntem, bir pikselin renginin/yoğunluğunun ardışık video kareleri içerisinde yer değiştirme altında değişmez olduğunu varsaymaktadır. Yine Lee ve Bovik'in (2009) yaptığı çalışmaya göre optik akış yöntemlerine, birçok video işleme algoritması içerisinde

başvurulmaktadır. Bu yöntemler videonun farklı karelerinde nesnelere ait yoğunlukların hareket tahminlerine dayanmaktadır. Optik akış, görüntüyü kaydeden kameranın sabit olmaması durumunda da kullanıma uygundur. Kamera hareketi, kameradan bağımsız harekete sahip cisimlerin algılanmasını ve tanımlanmasını yüksek oranda etkilememektedir. Optik akış, işlem karmaşıklığının yüksekliği nedeniyle gerçek dünya uygulamalarında hızlı bir donanım ve yazılım altyapısına sahip olması gerekmektedir. Bu yöntem, temelinde diferansiyel işlemler barındırması sebebiyle görüntü üzerinde oluşabilecek gürültüye karşı hassastır. Gürültü duyarlılığının azaltılması mümkün olmakla birlikte yapılan bu geliştirmeler sistem karmaşıklığını da arttıracaktır.

1.3 Yapay Zekâ, Makine Öğrenmesi ve Derin Öğrenme

François Chollet'e (2017) göre kısa tanımı zekâ gerektiren işlerin otomatikleştirme çabası olan yapay zekâ kavramı, 1950'lerde ortaya çıkmıştır. Yapay Zekâ, makine öğrenimi ve derin öğrenme gibi konseptleri kapsıyor olması ile birlikte herhangi bir öğrenme işlemi içermeyen yaklaşımlar da yapay zekâ olarak adlandırılmaktadır. Burada öğrenme işlemi içermeyen yaklaşımlar programcılar tarafından oluşturulmuş kural tabanlı sistemler olurken, bu sistemler makine öğrenimi olarak sınıflandırılmamaktadır. 1950'lerden 1980'lere kadar baskın bir paradigma olan sembolik yapay zekâ olarak da tanımlanan sistemler, programcılar tarafından yeterince geniş kurallar dizisi oluşturularak kurgulanmıştır. Sembolik yapay zekâ, satranç gibi mantıksal sorunları çözmek için uygun olsa da görüntü sınıflandırma, konuşma tanıma veya dil çevirisi gibi karmaşık ve bulanık sorunları çözmek gibi problemlere karşı açık kurallar tanımlanması mümkün olmamaktadır. Bunlar gibi karmaşık problemlerin çözümü, makine öğrenimi adı verilen öğrenme odaklı yöntemler ile sağlanabilmektedir.

1.3.1 Makine Öğrenmesi

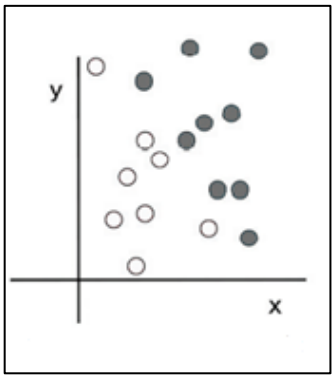
Chollet'in (2017) yazdığı kitaba göre Victoria İngiltere'sinde Leydi Ada Lovelace, genel amaçlı kullanıma yönelik bilinen ilk mekanik bilgisayar olan analitik motorun tasarımcısı ve mucidi olan Charles Babbage'ın arkadaşı ve ortağıydı. Analitik motor, 1830 ve 1840'larda genel amaçlı kullanım konseptinin o yıllarda icat edilmemesinden

dolayı genel amaçlı bilgisayar olarak nitelendirilmemiş ve matematiksel analiz alanındaki bazı hesaplamaların otomatikleştirilmesini sağlayacak bir yol olarak düşünülmüştür. Ada Lovelace'ın 1843'te, analitik motorun yeni şeyler ortaya çıkarma iddiasında olmadığını, Verilen komutları yerine getirebileceğini dile getirmiş ve analitik motorun görevinin bilinen işlerde kolaylık sağlamak olduğunu ifade etmiştir. Ada Lovelace'ın bu sözleri, Alan Turing tarafından 1950'de yayımlanan "Bilgisayar Makineleri ve Akıl" makalede "Leydi Lovelace'ın İtirazı" olarak alıntılanmıştır.

Yine François Chollet'in (2017) yaptığı derlemeye göre klasik programlama ve sembolik yapay zekâ paradigmasında programcılar kurallar oluşturmakta, bu kurallar gelen veriye uygun bir cevap üretmektedir. Makine öğreniminde ise temel yaklaşımlardan biri olan gözetimli öğrenmede (*Supervised Learning*) sisteme veriler ve bu verilerin çıktıları verilerek sembolik yapay zekadaki kurallar ortaya çıkarılması amaçlanmaktadır. Ortaya çıkarılan bu kurallar yeni veriler ile kullanılarak yeni cevaplar üretilmesi sağlanmaktadır. Bir makine öğrenimi sistemi, doğrudan kurallar verilerek programlanmamakta, bir görev ile ilgili gözlemler verilerek uygun istatistiksel yöntemlerle kuralların bulunup bu görevin otomatikleştirilmesinin temel parçalarından biri olarak yerini almaktadır.

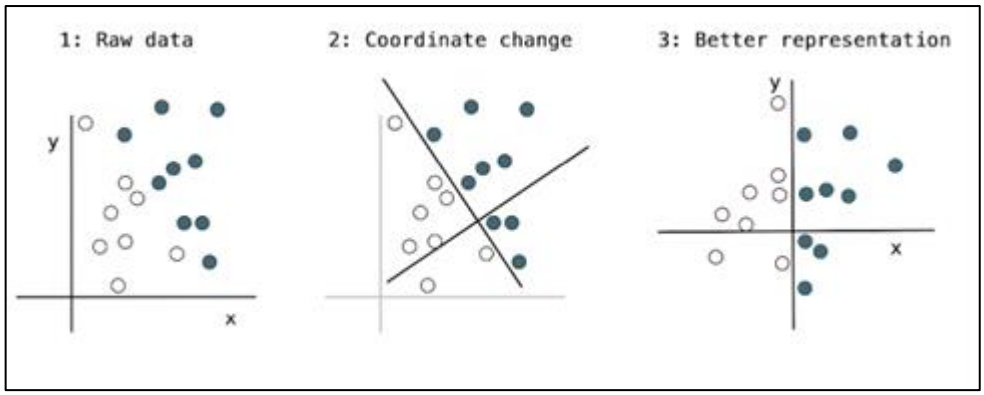
1.3.2 Öğrenme Konsepti

Chollet'in (2017) kitabına göre bir makine öğrenimi modeli, girdiler ve bu girdilere ait çıktı örneklerine maruz bırakılarak bir "öğrenme" eylemi gerçekleştirmektedir. Makine öğrenimi veya derin öğrenmede verilerin ayırt edici özelliklerinden beklenen çıktıya yaklaşmak için gerekli kurallara yaklaşım sağlamak veya gerekli dönüşümleri sağlayabilmek temel amaçtır. Bir diğer anlamıyla verinin temsilini ortaya koymaktır. Verinin temsili, verilerin farklı şekillerde ifade edilmesidir. Makine öğrenimi modelleri, girdiler için uygun temsiller bulma ve eldeki verileri istenen göreve uygun olarak verilerin dönüştürülmeleri ile ilgilidir.



Şekil 1.1 İki sınıflı veri görseli (Chollet, 2017)

Bir x ekseni ve y ekseni ile (x, y) koordinat sisteminde temsil edilen bazı noktalar yukarıdaki Şekil 1.1'de görüldüğü gibidir. Burada beyaz ve siyah noktaların ayrıştırılmasına yönelik model geliştirilmesinin beklendiği bir örnek verilecek olursa; nokta koordinatları girdileri, noktaların renkleri beklenen çıktıları ifade edecektir. Model performansı da doğru sınıflandırma yüzdesi ile ölçülebilir hale gelecektir. Beyaz ve siyah noktaları en temiz şekilde ayıran yeni koordinatları sağlayan koordinat değişimi verinin temsili olarak adlandırılabilir. Koordinatların değişimi ve yeni temsil aşağıdaki şekildeki gibi olacaktır.



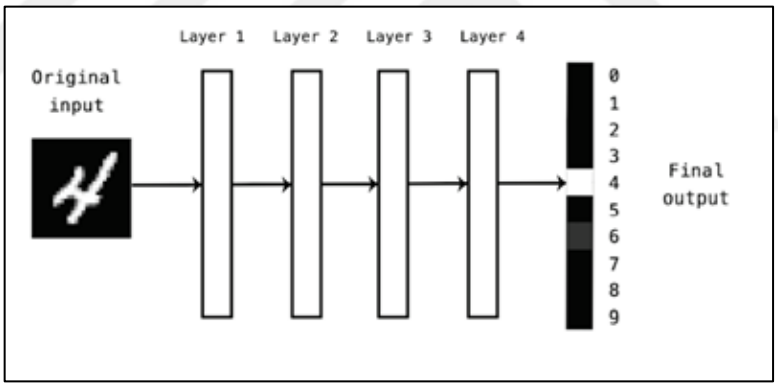
Şekil 1.2 İki sınıflı veri için koordinat değişimi (Chollet, 2017)

Şekil 1.2'deki gibi yeni temsilde beyaz noktalar x<0 ile siyah noktalar da x>0 ile kuralları ile tespit edilmektedir. Bu örnek için model kuralı x'in 0'dan büyük veya küçük olma durumu olmaktadır. Bu temsil sistemi doğru sınıflandırma yüzdesi ile ilgili bir geri bildirim kullanılarak kurgulandığı takdirde "öğrenme" işlemi gerçekleşmekte ve bu sistem bir makine öğrenmesi sistemi olarak adlandırılmaktadır. Olasılıklar alanı

içerisinde geri besleme sinyalleri kullanılarak girdilerin faydalı temsillerini aramak makine öğreniminin alternatif tanımı olarak verilebilmektedir.

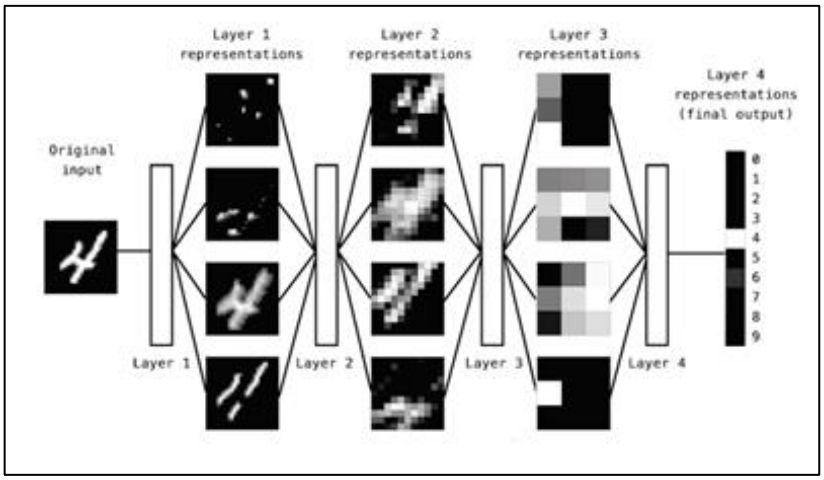
1.3.3 Derin Öğrenme

Chollet (2017) kitabında derin öğrenmeyi makine öğreniminin bir alt dalı olarak ifade etmektedir. Yapay sinir ağları kullanılarak oluşturulmuş katmanların sayılarının arttırılması bu makine öğrenmesi yöntemine "derin" anlamı katmaktadır. Makine öğrenmesi yönteminin derin sinir ağları yapısı bir alt dal olarak kabul edilmiştir. Bu yöntem derin öğrenme adını almıştır. Sinir ağı terimi nörobiyolojiye bir referanstır. Bu referansın sebebi de derin öğrenmedeki bazı kavramların beynin çalışma yapısından ilham alınarak ortaya çıkarılmış olmasıdır. İlham alınmış olmasına rağmen yapay sinir ağlarının çalışma şeklinin beyinde de olduğuna dair kesin kanıtlar bulunmamaktadır. Derin öğrenme, verilerden anlamlı temsiller çıkarılabilmesi amacıyla oluşturulmuş matematiksel bir çerçevedir. Şekil 1.3'te çok katmanlı bir sinir ağı örneği verilmiştir.



Şekil 1.3 Yapay sinir ağı konsept örneği (Chollet, 2017)

Verilen sinir ağı örneğinde verilen rakam resmi girdisinin katmanlarda farklı katmanlarda oluşturulan temsilleri Şekil 1.4'te gösterilmiştir.

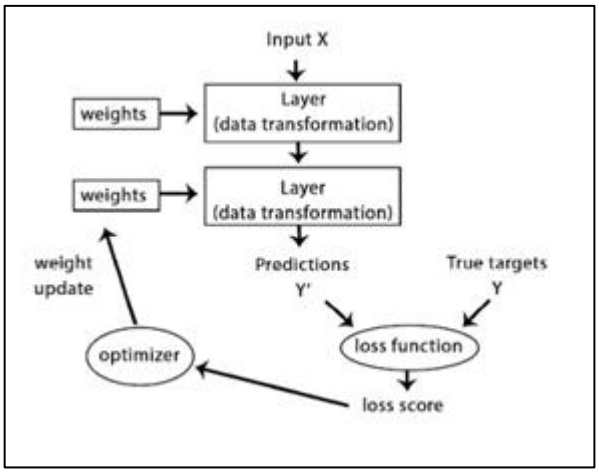


Şekil 1.4 Evrişimli sinir ağı aşamalarına ait örnek (Chollet, 2017)

Şekil 1.4 ile verilen ağ, orijinal görüntüden her kademe farklı temsillere dönüştürmektedir. Derin bir sinir ağı yapısı, bilginin ardışık filtrelerden geçtiği ve giderek daha fazla bilginin ortaya çıkarıldığı bir süreç olarak da kabul edilmektedir. Derin öğrenme, verilerin farklı ve kademeli temsillerinden "öğrenme" işlemi olarak da kabul edilebilir.

1.3.4 Derin Öğrenme Çalışma Prensibi

François Chollet'e (2017) göre bir derin sinir ağı yapısında katmanlarda gerçekleştirilen dönüşüm işlemi ağırlık adı verilen matrisler ile parametrelendirilmektedir. Bu bağlamda öğrenme işlemi ağırlık dizilerinin değerlerinin bulunması anlamına gelmektedir. Ağırlık dizilerinin değerlerinin bulunması ile örnek girdi verileri ilişkili hedefler ile eşlenebilmektedir. Bir sinir ağının çıktılarının kontrol edilebilmesi ve öğrenme işleminin gerçekleştirilebilmesi için beklenen çıktılar ile sinir ağı çıktılarının ne kadar farklı olduğunun ölçülmesi gerekmektedir. Bu farklılığın ölçümü de kayıp fonksiyonu (*Loss Function*) tarafından gerçekleştirilmektedir. Kayıp fonksiyonu, yapay sinir ağının tahminlerini ve hedef çıktıları kendisine girdi olarak almakta ve belirlenen farklılık metriğine göre bir puan hesaplaması yapmaktadır. Derin öğrenme sistemi, hesaplanan bu kayıp puanını düşürmek amacıyla barındırdığı ağırlıklar üzerinde güncelleme yapmaktadır. Sistem bu kayıp puanını bir geri besleme sinyali olarak kullanmaktadır. Yapılan bu güncelleme işlemi, geri yayılım algoritması ile gerçekleştirilmektedir.



Şekil 1.5 Geri yayılım (*back-propogation*) ile ağırlık güncelleme (Chollet, 2017)

Derin sinir ağı ilk kurulduğunda ağırlıkları rastgele olarak atanması nedeniyle eğitimin ilk aşamalarında derin sinir ağının tahmin çıktısı ile ideal çıktı arasındaki kayıp puanı çok yüksek olmaktadır. Şekil 1.5'te görüldüğü üzere yapay sinir ağı, her eğitim döngüsünde ağırlıklar kayıp puanını azaltacak yönde güncellenmektedir. Kayıp puanının en düşük seviyeye ulaştığında eğitim sonlanır ve sonlanan eğitimin ardından yapay sinir ağının tahmin çıktıları hedeflenen çıktılara en yakın değerleri üretir duruma gelmektedir.

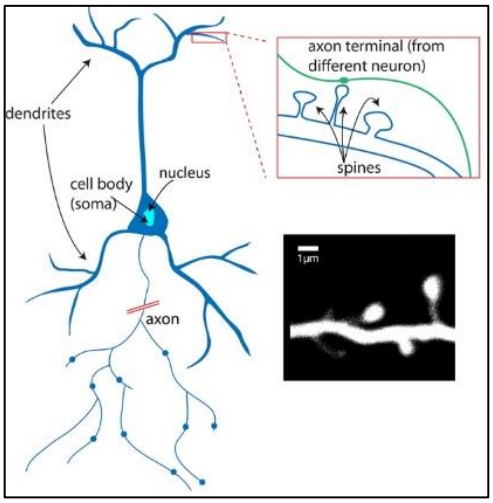
1.3.5 Biyolojide Sinir Ağları

İnsan vücudu, eylemleri, refleksleri ve algıyı koordine eden karmaşık yapı olan sinir sistemi olmadan bilinçli varlığını sürdürememektedir. Bu sinir sistemi iki ana bölümden oluşmaktadır. Bunlar merkezi sinir sistemi ve çevresel sinir sistemleridir. Beyni ve omuriliği barındıran merkezi sinir sistemi vücudun işlem operasyonlarının merkezidir. Beyin ve omurilik meninks adı verilen üç katlı bir yapı ile sarmalanmıştır. Daha güçlü bir koruma için beyin kafatası gibi sert bir kemik yapı içerisinde yer almaktadır. Omurilik ise kemikli bir yapı olan omurga içerisinde korunmaktadır. Üçüncü bir koruma da beyin-omurilik sıvısı adı verilen beyin ile kafatası arasında ve omurilik ile omurga arasında bulunan etkileşimi sınırlandıran ve tampon görevi gören sıvıdır (University of Queensland - Queensland Brain Institute, b.t.).

Dokusal yapı olarak bakıldığında, merkezi sinir sistemi gri madde (*grey matter*) ve beyaz madde (*white matter*) olarak ikiye ayrılmaktadır. Gri madde, nöron hücre

gövdelerini ve bu hücrelerin dendritlerini, glial hücrelerini ve kılcal damarlarını içermektedir. Beyaz madde ise merkezi sinir sisteminin nöronlardan uzanan aksonları barındıran bölgesidir. Aksonların çoğu, sinyal iletimini hızlandırabilen ve sinyalin güvenilirliğini sağlayan yağlı bir dokuya sahip olan miyelin tabakası ile kaplıdır (University of Queensland - Queensland Brain Institute, b.t.).

Beyinde beyaz madde gri maddenin altında gömülüdür ve oluşan sinyalleri beynin farklı bölümlerine taşımaktadır. Omurilikte ise beyaz madde gri maddeyi kaplayan dış katman olmaktadır. Beyin temel olarak ön beyin, orta beyin ve arka beyin olmak üzere üç ana bölümden oluşmaktadır. Gelişmekte olan beyindeki prosensefalondan türemiş olan ön beyin bu üç bölümden en büyüğüdür. Beynin en dış ve en büyük tabakası olan serebral korteksi, merkezine doğru talamus, hipotalamus ve epifiz bezi gibi yapıları içermektedir. Gelişmekte olan beyindeki mezensefalondan türemiş olan orta beyin, ön beyin ile arka beyin arasındaki bağlantıyı sağlamakta ve aynı zamanda beyni omuriliğe bağlayan beyin sapının üst kısmını oluşturmaktadır. Arka beyin, beyin sapının hemen arkasına yerleştirilmiş küçük bir yoğun beyin dokusu topu olan serebellumu içeren beynin en alt arka kısmıdır. Bu büyük yapıyı oluşturan en temel parça sinir hücreleridir. Sinir hücreleri (nöronlar), beynin ve sinir sisteminin temel birimleridir, dış dünyadan duyusal girdi almaktan, kaslarımıza motor komutları göndermekten ve aradaki her adımda elektrik sinyallerini dönüştürmek ve iletmekten sorumlu hücrelerdir. Bir sinir hücresinin (nöronun) üç ana bölümü vardır: dendritler, akson ve hücre gövdesi veya soma. Bir dendrit, bir nöronun diğer hücrelerden girdi aldığı yerdir. Şekil 1.6'da bu yapı görülmektedir (University of Queensland - Queensland Brain Institute, b.t.).

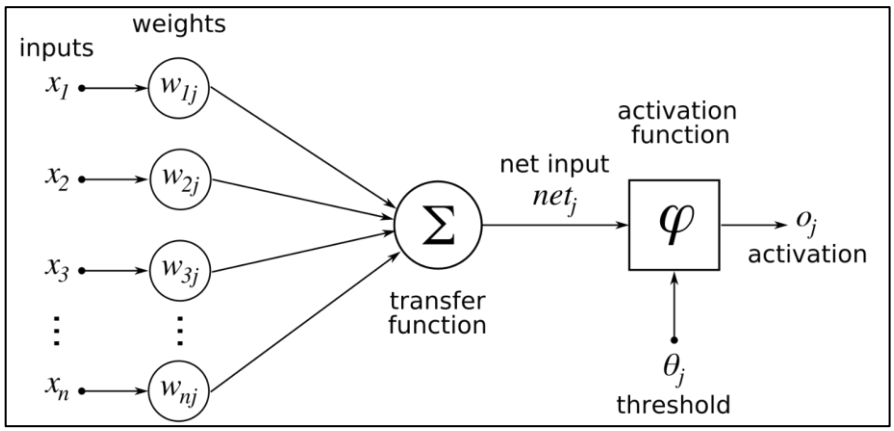


Şekil 1.6 Sinir hücresi yapısı (University of Queensland - Queensland Brain Institute, b.t.)

Akson, nöronun sinyal çıktısını ileten yapısıdır; Bir nöron başka bir nöronla iletişim kurmak istediğinde, tüm akson boyunca aksiyon potansiyeli adı verilen bir elektrik mesajı gönderir. Soma, çekirdeğin bulunduğu, nöronun DNA'sının barındırıldığı ve proteinlerin akson ve dendritler boyunca taşınmasının sağlandığı yerdir (University of Queensland - Queensland Brain Institute, b.t.).

1.3.6 Yapay Sinir Ağı Hücresi (Yapay Nöron)

Yapay sinir ağı hücresi (*artificial neuron*), yapay sinir ağlarının en temel parçasıdır. Çalışma prensibi aynı olmamakla birlikte biyolojik sinir hücrelerinden esinlenilmiştir. 3 ana kısımdan oluşmaktadır.



Şekil 1.7 Yapay nöron yapısı (Ren vd., 2018)

Şekil 1.7'de görülen bu 3 ana kısım; girdiler (*inputs*), ağırlıklar (*weights*) ve aktivasyon fonksiyonu (*activation function*) olarak adlandırılmaktadır. Yapay nöron, girdi değerleri ile ağırlıkların çarpımlarının toplamını aktivasyon fonksiyonundan geçirerek bir çıktı oluşturmaktadır. Yapay nöronun yaptığı değer çıkartımı işlemi eşitlik 1.1 ve eşitlik 1.2'de *sigmoid* aktivasyon fonksiyonlu örneği verilmiştir.

$$z = (w_1 x_1 + \cdots + w_n x_n) \tag{1.1}$$

$$\hat{y} = \frac{1}{1 + e^{-z}} \tag{1.2}$$

Yapay nöronların bir araya getirilmesi ile yapay sinir ağları oluşmaktadır. Oluşan sinir ağlarında bir tahmin ($\hat{y}$) yapıldığında gerçek hedef değeri ile aralarında oluşan farklılık bir kaybı ortaya koymaktadır (Analyticsvidhya, b.t.).

1.3.6.1 Kayıp (Loss)

Kayıp, yapılan çalışmanın türüne göre farklılık göstermek ile birlikte yapılan tahminin gerçek değerden ne kadar farklı olduğudur. Yapay sinir ağlarında regresyon veya sınıflandırma gibi çalışmalarda bu kaybın hesaplanması için farklı kayıp fonksiyonları (*loss function*) bulunmaktadır. Bazı kayıp fonksiyonları aşağıdaki gibidir (Keras, b.t.).

- *Mean Squared Error* (MSE)
- *Mean Squared Logarithmic Error*
- *Mean Absolute Error* (MAE)
- *Binary Cross Entropy* (*Log Loss*)
- *Hinge Loss*

Bu tezde yapılan çalışmada *binary cross entropy* (*log loss*) kayıp fonksiyonu kullanılmıştır.

Logaritmik kayıp fonksiyonu (*log loss*), sınıflandırma problemlerinde kullanılan bir kayıp fonksiyonudur. Logaritmik kayıp, model tarafından yapılan sınıflandırma olasılığının gerçek etiket değerine ne kadar yakın olduğunun ifadesidir.

$$L_{log}(y,p) = -(\mathrm{y} * \log(p) + (1-y) * \log(1-p)) \quad (1.3)$$

Eşitlik 1.3'te y gerçek etiket değerini, p ise tahmin sonucundaki olasılık değerini ifade etmektedir (Scikit-Learn, b.t.).

1.3.6.2 Optimizasyon

Tahminde ortaya çıkan kaybın azaltılması ve ağırlıkların en az hatayı verebilecek değerine hızlı şekilde ulaşabilmesi için optimizasyon fonksiyonu kullanılmaktadır. Optimizasyon fonksiyonları farklı şekillerde ağırlık güncellemesi yapabilmektedir. Aşağıda bazı optimizasyon yöntemleri listelenmiştir.

- *Stochastic Gradient Descent*
- *AdaGrad*
- *RMSProp*
- *Adam*

Bu tez kapsamında yapılan çalışmalarda *Adam* optimizasyonu kullanılmıştır. Kingma ve Ba'nın (2015) geliştirmiş olduğu *Adam* (*Adaptive Moment Estimation*), gradyan inişi hesaplamasında kullanılan bir algoritmadır. Bu yöntem büyük çok boyutlu veri kümesi içeren veya çok fazla parametre içeren problemler üzerinde etkilidir. Daha az bellek (hafıza) gerektirdiğinden verimliliği de yüksektir. Adam iki farklı yöntem olan momentumlu gradyan inişi ile *RMSProp* algoritmalarının bir kombinasyonudur.

Momentum, ağırlık güncellemesi miktarının alınan ağırlık değişiminde hatanın değişimi ile birlikte değiştiği optimizasyon yöntemidir.

$$w_{t+1} = w_t - \alpha * m_t \tag{1.4}$$

$$m_t = \beta * m_{t-1} + (1 - \beta) * \frac{\delta L}{\delta w_t} \tag{1.5}$$

Eşitlik 1.4 ve eşitlik 1.5'te, m_t t anındaki eğimi, m_{t-1} t-1 anındaki eğimi, w_t t anındaki ağırlık değerlerini, w_{t+1} t+1 anındaki ağırlık değerlerini, α öğrenme oranını, δL hata fonksiyonunun türevini, δw_t t anındaki ağırlıkların türevini ve β hareketli ortalama parametresini ifade etmektedir.

RMSProp (*Root Mean Square Propogation*), *AdaGrad* algoritmasını geliştiren ve iyileştiren adaptif bir optimizasyon algoritmasıdır. *AdaGrad* algoritması gibi gradyanların karelerinin birikimli toplamını kullanmak yerine üstel hareketli ortalamayı kullanmaktadır.

$$w_{t+1} = w_t - \frac{\alpha_t}{(v_t + \varepsilon)^{1/2}} * \frac{\delta L}{\delta w_t} \tag{1.6}$$

$$v_t = \beta * v_{t-1} + (1 - \beta) * \frac{\delta L}{\delta w_t}^2 \tag{1.7}$$

Eşitlik 1.6 ve eşitlik 1.7'de, w_t t anındaki ağırlık değerlerini, w_{t+1} t+1 anındaki ağırlık değerlerini, α öğrenme oranını, δL hata fonksiyonunun türevini, δw_t t anındaki ağırlıkların türevini, β hareketli ortalama parametresini, v_t eski gradyanların karelerinin toplamını ve ε küçük bir pozitif sabiti ifade etmektedir.

Adam algoritması bu iki yöntemi birleştirerek global minimumu hızlı bir şekilde bulmayı amaçlamaktadır.

$$m_t = \beta_1 * m_{t-1} + (1 - \beta_1) * \frac{\delta L}{\delta w_t} * v_t \tag{1.8}$$

$$m_t = \beta_2 * v_{t-1} + (1 - \beta_2) * \frac{\delta L}{\delta w_t}^2 \tag{1.9}$$

Eşitlik 1.8 ve eşitlik 1.9 denklemleri kurulurken ε sıfır ile bölünme probleminden etkilenmemek üzere seçilen çok küçük pozitif bir değerdir. β_1 ve β_2 Adam algoritmasının yararlandığı algoritmaların hareketli ortalama parametreleridir. m_t ve v_t, başlangıç durumlarında 0 (sıfır) olarak alındığından β_1 ve $\beta_2 \approx 1$ için 0'a karşı yanlılık göstermektedirler. Bu durumdan kaçınmak için m_t ve v_t için yanlılık düzeltme (*bias correction*) işlemi uygulanmaktadır.

$$\widehat{m_t} = \frac{m_t}{1 - \beta_1^t} * v_t \tag{1.10}$$

$$\widehat{m_t} = \frac{v_t}{1 - \beta_2^t} \tag{1.11}$$

Yanlılık düzeltme işleminin (eşitlik 1.10 ve eşitlik 1.11) sonunda oluşan aşağıdaki matematiksel ifade (eşitlik 1.12) ile ağırlık güncellemesi gerçekleştirilmektedir (GeeksforGeeks, b.t.).

$$w_{t+1} = w_t - \widehat{m_t}\left(\frac{\alpha}{\sqrt{\widehat{v_t}} + \varepsilon}\right) \tag{1.12}$$

BÖLÜM 2
MATERYAL VE YÖNTEM

Bu bölüm iki parça olarak hazırlanmıştır. “Materyal” bölümünde geliştirme yapılan ortam, kullanılan programlama dili, çalışmanın temelinde yer alan yazılım kütüphaneleri ve deneyin yapıldığı veri seti tanıtılmıştır. “Yöntem” bölümünde ise çalışmada kullanılmış olan sinir ağı yöntemleri ile aktarımlı öğrenme yolu ile kullanılmış olan iki sinir ağı modeli tanıtılmıştır.

2.1 Materyal

2.1.1 Google Colaboratory

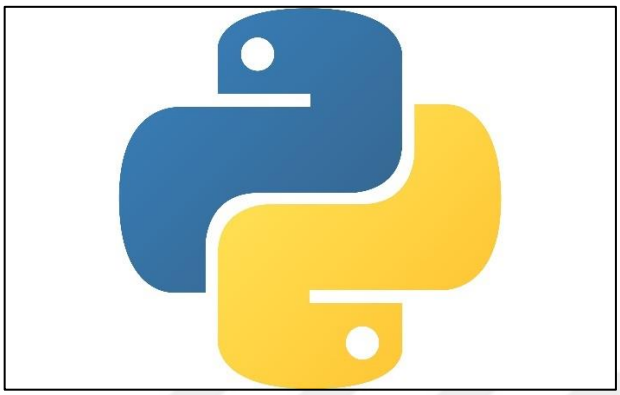
Colaboratory veya kısaca Colab (2022), Google Research’ün bir ürünüdür. İnternet tarayıcısı üzerinden Python programlama dilinde kod yazmaya ve çalıştırmaya olanak sağlamaktadır. Makine öğrenmesi, veri analizi gibi alanlarda kullanılabilmektedir. Teknik anlamda Colab, içerisinde Jupyter Notebook servisini içerisinde barındırmaktadır ve kullanıcıların yaralanması için ücretsiz *GPU* (*Graphical Processing Unit*) erişimi sağlamaktadır. Şekil 2.1’de Google Colab’ın logosu bulunmaktadır (Google Colab, b.t.).

Şekil 2.1 Google Colaboratory logosu (Google Colab, b.t.)

2.1.2 Python

Python, bir yorumlanmış yüksek seviyeli genel kullanım programlama dilidir. Guido van Rossum tarafından 1990'ların başında Hollanda'daki Stichting

Mathematisch Centrum'da oluşturulmuştur. Dizayn felsefesi kod okunabilirliği üzerinedir ve bunu kendine özel girintiler ile oluşturmaktadır. Şekil 2.2'da Python programlama dilinin logosu bulunmaktadır.



Şekil 2.2 Python programlama dili logosu (Python, b.t.)

Genel özellikleri;

- Yorumlanmış bir dil olsa da alt seviyelerde byte kodu otomatik olarak derlenmektedir. Bu özelliği nedeniyle *scripting language* (komut dosyası dili) olarak *web* uygulamaları için kullanılmaktadır.
- C ve C++ gibi diller ile genişletilebiliyor oluşu, Python'ın yüksek hız gerektiren hesaplamalarda da kullanılmasını mümkün hale getirmektedir.
- Obje kullanımına olanak verdiği güçlü yapısı ve esnek programlama kabiliyeti ile nesne yönelimli programlama için de uygun bir dil olarak kabul görmektedir (Python, b.t.).

2.1.3 Scikit-Learn

Pedragosa ve arkadaşlarının (2011) geliştirdiği Scikit-Learn, orta ölçekli denetimli ve denetimsiz problemler için çok çeşitli son teknoloji makine öğrenme algoritmalarını entegre eden bir Python modülüdür. Bu paket, genel amaçlı üst düzey bir dil kullanarak makine öğrenimini uzman olmayan kişilere sunmaya odaklanmaktadır. Kullanım kolaylığı, performans, belgelendirme ve *API* tutarlılığı ön plana çıkan özellikleridir. Bulundurduğu minimum bağımlılık ve basitleştirilmiş BSD lisansı ile hem akademik

hem de ticari kullanımı teşvik etmektedir. Şekil 2.3'te Scikit-Learn'ün logosu bulunmaktadır.

Şekil 2.3 Scikit-Learn logosu (Scikit-Learn, b.t.)

Scikit-learn kütüphanesinin fonksiyon kullanım alanları;

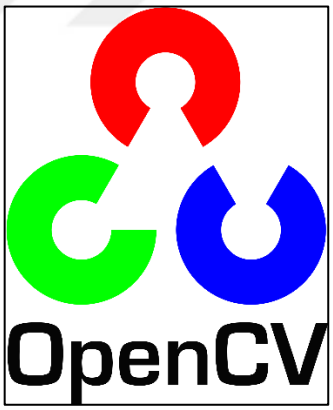
- Sınıflandırma (*Classification*)
- Regresyon (*Regression*)
- Kümeleme (*Clustering*)
- Boyut indirgeme (*Dimensionality Reduction*)
- Model seçimi (*Model Selection*)
- Veri ön işleme (*Preprocessing*)

2.1.4 OpenCV

OpenCV'nin (*Open Source Computer Vision Library*), bir açık kaynaklı bilgisayarlı görü (*Computer Vision*) ve makine öğrenmesi (*Machine Learning*) kütüphanesidir. OpenCV, bilgisayarlı görü uygulamaları için ortak bir altyapı sağlamak ve makine algısını ticari ürünlerde kullanımının arttırılması amacıyla oluşturulmuştur. BSD lisanslı bir ürün olması OpenCV'yi şirketler için kullanımını ve modifiye edilmesini kolay hale getirmektedir. Bu kütüphane, çok kapsamlı son gelişmeleri içerisinde barındıran bilgisayarlı görü ve makine öğrenmesi algoritmalarını da içeren 2500'den fazla optimize edilmiş algoritma barındırmaktadır. Bahsi geçen algoritmalar yüz tespiti ve yüz tanıma, nesne tanımlama, video içerisindeki insan hareketlerini tanımlama, kamera hareketini takip etme, nesne

hareketini takip etme, ikili kameralardan (*stereo cameras*) 3 boyutlu nesne çıkartımı ve 3 boyutlu nesne bulutu oluşturma, resimleri birleştirerek daha yüksek çözünürlükte resim oluşturma, resim veri tabanından benzer resimleri bulma, fotoğraf makinesi flaşından kaynaklı kırmızı göz görüntüsünü silme, göz hareketini takip etme, manzara tanımlama, arttırılmış gerçeklik vb. uygulamalarda kullanılmaktadır. OpenCV, 47 binden fazla kullanıcı içeren topluluğa ve dünya çapında 18 milyon indirme sayısına ulaşmıştır (OpenCV, b.t.).

Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota gibi büyük ve yerleşik şirketlerin bu kütüphaneyi kullandığı gibi, *start-up* şirketler de OpenCV'nin farklı kullanımları bilinmektedir. OpenCV'nin İsrail'de güvenlik kameralarından izinsiz giriş takibi, Çin'de maden ekipmanlarının takibi, George Garage'da yardımcı robotların yön bulma ve nesneleri hareket ettirmesi, Avrupa'daki yüzme havuzlarında boğulmaların tespit edilmesi, İspanya ve New York'ta interaktif sanat faaliyetleri, dünya çapında fabrikalarda ürünlerin etiketlerinin incelenmesi, Japonya'da hızlı yüz tanıma gibi aktif uygulamaları bulunmaktadır (OpenCV, b.t.).

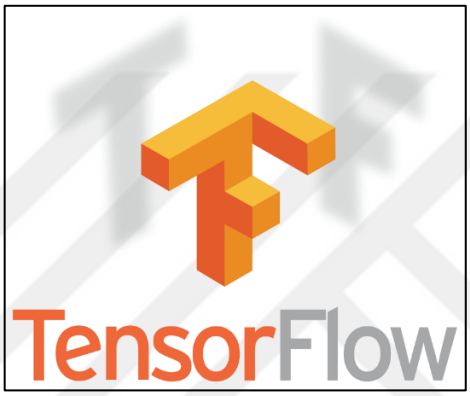


Şekil 2.4 OpenCV logosu (OpenCV, b.t.)

Şekil 2.4'te logosu bulunan OpenCV'nin C++, Java, Python ve MATLAB için ara yüzleri ve Windows, Linux, MacOS ve Android işletim sistemlerine desteği bulunmaktadır. OpenCV daha çok gerçek zamanlı görüntü işleme uygulamaları üzerine yoğunlaşmıştır. Tam özellikli *CUDA* ve *OpenCL* ara yüzleri geliştirilmektedir. OpenCV, yerel olarak C++ ile yazılmıştır ve STL *containers* ile sorunsuz bir şekilde çalışan şablonlu bir ara yüze sahiptir (OpenCV, b.t.).

2.1.5 Tensorflow

Abadi ve arkadaşları (2015) tarafından ortaya konan Tensorflow (*Large-Scale Machine Learning on Heterogeneous Distributed Systems*), makine öğrenimi algoritmalarını ifade etmek için geliştirilmiş bir ara yüz ve bu tür algoritmaları yürütmek için Google tarafından geliştirilen bir uygulamadır. Tensorflow ile geliştirilen uygulamalar ve yapılan hesaplama çalışmaları, değişiklik yapılmadan veya çok az değişim yapılarak mobil ve GPU gibi çeşitli heterojen sistemler üzerinde çalışabilmektedir. Şekil 2.5'te Tensorflow'un logosu görülmektedir.



Şekil 2.5 Tensorflow logosu (Wikimedia Commons, b.t.)

Yine Abadi ve arkadaşlarının (2015) yaptığı çalışmaya göre sistemin esnekliği sayesinde Tensorflow, derin sinir ağı modelleri için eğitim (*training*) ve çıkarım (*inference*) algoritmaları dahil olmak üzere çok çeşitli algoritmaları ifade etmek için ve çok sayıda alanda araştırma yapmak ve makine öğrenimi sistemlerini üretime yerleştirmek için kullanılabilmektedir. Tensorflow'un kullanıldığı bazı alanlar;

- Konuşma tanıma (*Speech Recognition*)
- Bilgisayarlı görü (*Computer Vision*)
- Robotik (*Robotics*)
- Bilgi edinimi (*Information Retrival*)
- Doğal dil işleme (*Natural Language Processing*)
- Coğrafi bilgi çıkarma (*Geographic Information Extraction*)

- Hesaplamalı ilaç keşfi (*Computational Drug Discovery*)

2.1.6 Veri Seti

Soliman ve arkadaşlarının (2019) oluşturduğu *Real Life Violence Situations Dataset*, şiddet tespiti çalışma kapsamında kullanılan veri setidir. İçerisindeki şiddet içerikli görüntülerde farklı ortamlardaki sokak kavgalarından görüntüler bulunmasıyla birlikte şiddet içermeyen görüntülerde yürüyüş, şarkı söyleme ve çeşitli spor etkinliklerinden görüntüler barındırmaktadır. Şekil 2.6 ve Şekil 2.7'de sırasıyla şiddet içermeyen ve şiddet içeren videolara ait kareler bulunmaktadır.

Şekil 2.6 Fiziksel şiddet içermeyen video karesi (Soliman vd., 2019)

Şekil 2.7 Fiziksel şiddet içeren video karesi (Soliman vd., 2019)

Veri setinin başlıca özellikleri;

- 1000 adet şiddet içerikli, 1000 adet şiddet içerikli olmayan toplam 2000 adet video görüntüsü bulunmaktadır.
- Video görüntüleri arasında gri ölçekli örnekler bulunmasında rağmen veri setinin büyük çoğunluğu üç kanal içeren renkli videolardır.
- Video karelerinin boyutları değişkenlik göstermektedir.
- Videoların toplam kare sayıları değişkenlik göstermektedir.
- Veri seti içerisinde sabit ve sabit olmayan kamera ile çekilmiş video görüntüleri karışık olarak yer almaktadır.

Veri setinin ana modele aktarılmadan önce uygulanan görüntü işleme adımları;

- Video kareleri 80x80 boyutlarına ölçeklendirilmiştir.
- Toplam kare sayısı farklı çalışmalar için 100 ve 34 olarak ayarlanmıştır.

2.2 Yöntem

2.2.1 Evrişimli Sinir Ağları

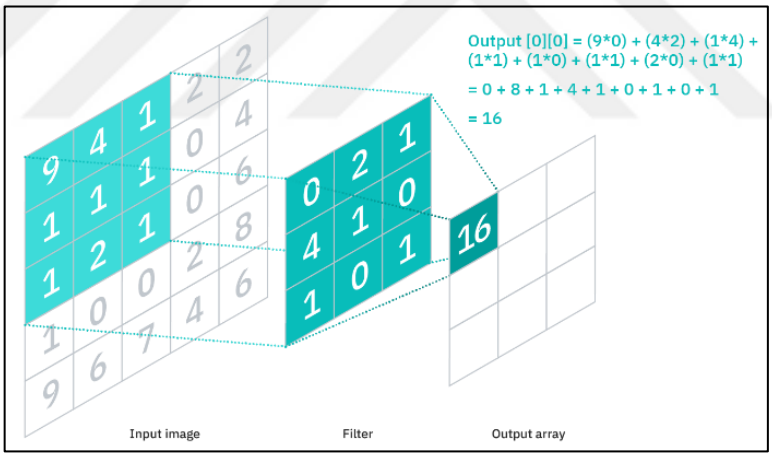
Görüntü ve ses sinyalleri üzerinde yüksek performans gösteren yapay sinir ağlarının bir üyesi olan evrişimli sinir ağları (*Convolutional Neural Networks*), yapısı itibariyle üç farklı katman yapısını barındırmaktadır:

- Evrişimsel katman (*Convolutional layer*)
- Havuzlama katmanı (*Pooling layer*)
- Tam bağlı katman (*Fully-Connected layer*)

Evrişimsel sinir ağları, ilk katmanlarını evrişimsel ve havuzlama katmanlarının oluşturduğu ardından tam bağlı katmanın takip ettiği bir yapı olarak tanımlanmaktadır. Evrişimsel sinir ağları, ilk katmanlarda renk, kenar, köşe gibi basit özellikleri çıkartırken sonraki katmanlarda bulunmak istenen nesneyi tanımlayabilecek nihai özellikleri çıkartmaktadır (IBM, b.t.).

2.2.1.1 *Evrişimsel Katman*

Evrişimsel katman, bir evrişimsel sinir ağının temel yapı taşıdır. Görüntü veya ses gibi girdiler, belli filtrelerden geçirilerek özellik haritalarının (*feature maps*) çıkartılmasına ihtiyaç duymaktadır. Evrişimsel katmanlardaki eğitilen filtreler veya diğer adıyla çekirdekler (*kernels*), görüntü veya analizi istenen görüntü üzerinde belli aralıklarda hareketi ile çıktılar oluşturmaktadır. Her evrişimsel katmanda gerçekleşen bu süreç konvolüsyon (evrişim) olarak adlandırılmaktadır. Görüntüye ait özellikleri çıkartan filtre, iki boyutlu (2D) bir ağırlık dizisidir. Boyutlarında değişiklikler olsa da bu filtrelerin yaygın kullanım boyutu 3x3'lük bir matristir. Filtre boyutu, katman çıktısının boyutunu da belirlemektedir. Filtre görüntünün bir bölgesine uygulanmaktadır ve giriş pikselleri ile filtre arasında bir nokta çarpımı hesaplanmaktadır. Bu nokta çarpımı daha sonra bir çıktı dizisine aktarılmaktadır. Filtre, bu işlemi tüm görüntü boyunca tekrarlamakta ve katmanın son çıktısı özellik haritası olarak adlandırılmaktadır (IBM, b.t.).



Şekil 2.8 Havuzlama (*pooling*) işlemi (IBM, b.t.)

Şekil 2.8'de, özellik haritasındaki her bir çıktı değeri, giriş görüntüsündeki her bir piksel değerinden beslenmemektedir. Bu çıktı değeri, filtrenin uygulandığı görüntü bölgesinin değerlerine bağlı olmaktadır. Görüntü boyunca hareket eden filtrenin ağırlıkları her bir hareket süreci boyunca sabit kalmaktadır ve bu parametre paylaşımı olarak adlandırılmaktadır. Ağırlık değerleri, evrişimli sinir ağının eğitimi sırasında geri yayılım algoritması ve gradyan inişi ile ayarlanmaktadır. Bunun yanı sıra çıktı

boyutlarını etkileyen üç farklı hiper parametre bulunmaktadır. Bu hiper parametreler aşağıda listelenmiştir:

1. Filtre sayısı, katman çıktı sayısını etkilemektedir. Her filtre için bir adet özellik haritası oluşmaktadır.
2. Adım büyüklüğü (*stride*), çekirdeğin görüntü üzerinde bir adım için atlanacak piksel sayısıdır. Adım büyüklüğü arttıkça çıktıdaki her bir özellik haritasının boyutu küçülmektedir.
3. Sıfır dolgulama (*zero padding*), filtrelerin görüntü girdi görüntüsüne uymadığı durumlarda görüntü matrisinin uç koordinatlarına sıfır atanarak boyut uyumunun sağlanması işlemidir. Üç tip dolgulama çeşidi bulunmaktadır:
    - Geçerli Dolgu (*Valid Padding*): Dolgu olmaması (*no padding*) olarak da adlandırılmaktadır. Boyutların eşleşmemesi durumunda son konvolüsyon işlemi atlanmaktadır.
    - Aynı Dolgu (*Same Padding*): Bu dolgu katman çıktısının girdi ile aynı boyutta olmasını sağlamaktadır.
    - Tam Dolgu (*Full Padding*): Bu dolgu, girdinin uç koordinatlarına sıfır ekleyerek çıktının boyutunu arttırmaktadır.

Bir evrişimsel sinir ağı, evrişim işleminden sonra özellik haritasına bir çeşit aktivasyon fonksiyonu uygulayarak modele doğrusal olmama durumu kazandırmaktadır (IBM, b.t.).

2.2.1.2 Havuzlama Katmanı

Bir tür alt örnekleme olarak tanımlanan havuzlama katmanları, havuz katmanına girdi olan özellik matrislerinin boyutlarını indirgeme işlemini gerçekleştirmektedir. Evrişim katmanındaki filtreler gibi gelen girdi matrislerini taramaktadır. Bu katmanın kendine ait ağırlığı bulunmamaktadır. Yaygın iki tür havuzlama türü bulunmaktadır:

- Maksimum Havuzlama (*MaxPooling*): Çekirdek girdi matrisini tararken, çıkış matrisine aktarılacak maksimum değere sahip pikseli seçmektedir.

- Ortalama Havuzlama (*AveragePooling*): Çekirdek girdi matrisini tararken çıkış matrisine çekirdeğe karşılık gelen alandaki değerlerin ortalama değeri aktarmaktadır.

Bu katmanda çekirdek, bir çeşit seçim veya toplama işlemi uygulayarak çıktı matrisi oluşturmaktadır (IBM, b.t.).

2.2.1.3 Tam Bağlı Katman

Bu katman, temel bir yapay sinir ağı örneğidir. Evrişimli sinir ağlarında bu katman, çıkartılan özellik haritalarının sınıflandırma veya yakınsama işlemini yapmaktadır. Bu katmandaki her bir gizli katman birbiri ile bağlantılıdır. Bu özelliği sebebiyle bu katmana tam bağlı katman adı verilmektedir (IBM, b.t.).

2.2.1.4 Evrişimli Sinir Ağları ve Bilgisayarlı Görü

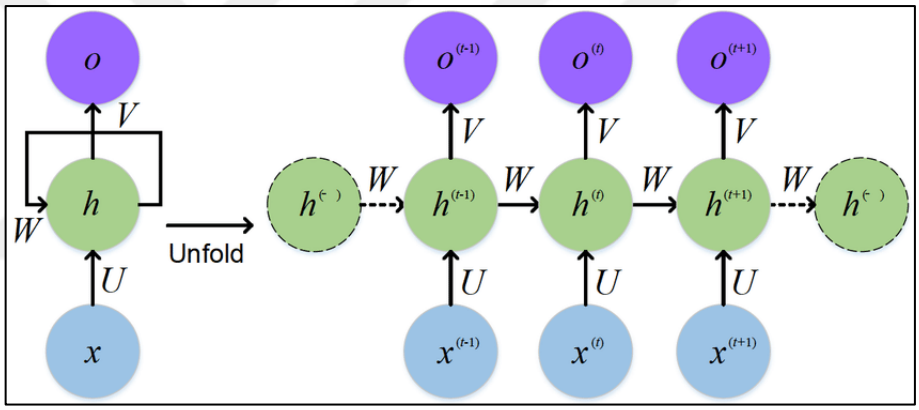
Evrişimli sinir ağları, görüntü tanıma ve bilgisayarlı görme problemlerinin çözümünde yaygın kullanılan yöntemlerden biri olarak yerini almaktadır. Günümüzde bilgisayarlı görü alanında bazı uygulamalar aşağıdaki gibidir:

- Pazarlama: Sosyal medya paylaşımlarındaki nesnelere yönelik reklam önerisinde bulunan uygulamalar
- Sağlık Hizmetleri: Kanserli hücrelerin tespiti
- Perakende: Görsel benzerliğinden yola çıkarak ürün araması ile e-ticaret platformları için kullanıcı deneyiminin iyileştirilmesi
- Otomotiv: Otonom araçlarda yol, şerit, yaya gibi nesnelerin tespiti ve ayrıştırılması

Evrişimli sinir ağları, bu örnek uygulamalar dışında tarımdan sanayiye daha birçok alanda uygulama alanına sahiptir (IBM, b.t.).

2.2.2 *Uzun-Kısa Dönemli Hafıza Ağı*

Özyineli sinir ağı (*Recurrent Neural Network*), sıralı, ardışık veya zaman serisi verilerini girdi olarak alan bir tür yapay sinir ağıdır. Bu derin öğrenme algoritmaları, dil çevirisi, anlık çeviri, doğal dil işleme (*natural language processing*), konuşma tanıma ve resim yazısı gibi sıralı veya zamansal problemler için kullanılmaktadır. Bu sinir ağı türünün kullanımına örnek olarak Siri ve Google Çeviri uygulamaları verilebilir. Özyineli sinir ağlarında katmana gelen girdiler diğer sinir ağlarında olduğu gibi birbirinden bağımsız değildir. Bir sayı dizisi için bir sonraki çıktıyı etkileyebilecek bir hafıza mekanizması bulunmaktadır. Her çıktı kendinden önceki girdilerin çıktılarından etkilenmektedir (IBM, b.t.). Şekil 2.9'da özyineli sinir ağlarının katman yapısı görülmektedir.



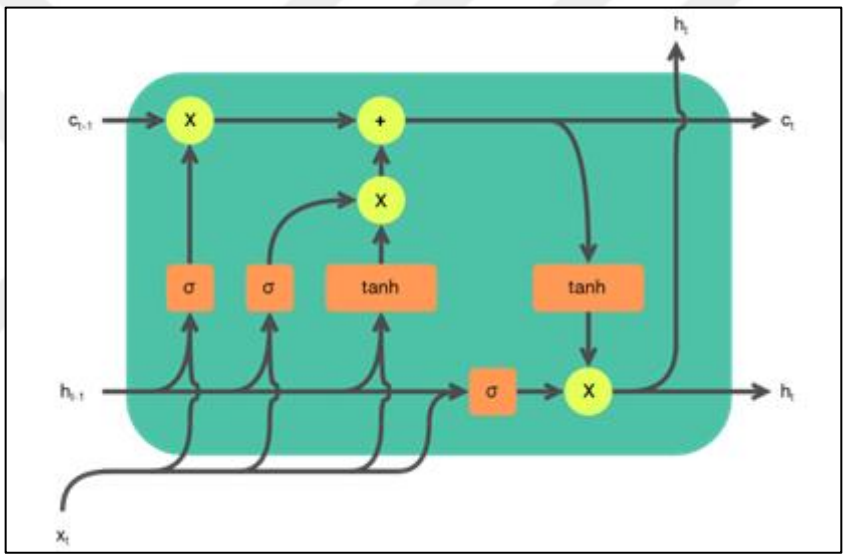
Şekil 2.9 Özyineli sinir ağı katmanı yapısı (Feng vd., 2017)

Özyineli sinir ağlarının çeşitleri aşağıdaki gibidir.

- Çift Yönlü Özyineli Sinir Ağları (BRNN): Tek yönlü sinir ağları, mevcut durum hakkında yalnızca kendisinden önceki girdilerden bilgi aktarımı yaparken, çift yönlü özyineli sinir ağları dizinin sonrasındaki verilerden de yararlanmaktadır.
- Gated Recurrent Units (GRU): Bu özyineli sinir ağı çeşidi LSTM'ler gibi özyineli sinir ağı modellerinde yaşanan kısa süreli bellek probleminin çözümüne odaklanmaktadır. Tek bir hücre durumu (*cell state*) kullanmamaktadır. GRU, iki adet gizli durum kullanmaktadır; bir sıfırlama ve

bir de güncelleme kapısı. LSTM hücrelerindeki gibi aktarılacak bilgiyi ve miktarını belirlemektedir.

- Long Short-Term Memory (LSTM): Uzun-kısa dönemli hafıza kaybolan gradyan problemine Sepp Hochreiter ve Jurgen Schmidhuber tarafından önerilen bir özyineli sinir ağı mimarisidir. Özyineli sinir ağlarında tahmini etkileyen durum yakın geçmişte değil ise mevcut durumun tahmin başarısını olumsuz yönde etkileme ihtimali bulunmaktadır. Uzun-kısa dönemli hafıza, sadece yakın geçmişteki değil daha uzak geçmişin bilgi aktarımına da olanak sağlayan bir yöntemdir (IBM, b.t.). Şekil 2.10'da bir LSTM hücresinin yapısının temsili görülmektedir.



Şekil 2.10 LSTM hücresi yapısı (Wikimedia Commons, b.t.)

LSTM, bir dizi verisindeki bilgilerin ağa girişini, depolanmasını ve ağdan çıkışını kontrol eden geçitler (*gate*) kullanmaktadır. Buradaki her bir geçit kendi başına birer sinir ağıdır. Geçitlerin ve diğer kavramlar aşağıdaki gibidir.

- Unutma geçidi (*Forget Gate*): Bu geçit hangi bilginin muhafaza edilip hangi bilginin unutulacağını belirlemektedir.
- Girdi geçidi (*Input Gate*): Girdi geçidi, hangi değerlerin güncelleneceğini belirlemektedir.
- Hücre durumu (*Cell State*): LSTM hücresinin uzun dönemli hafızasıdır.

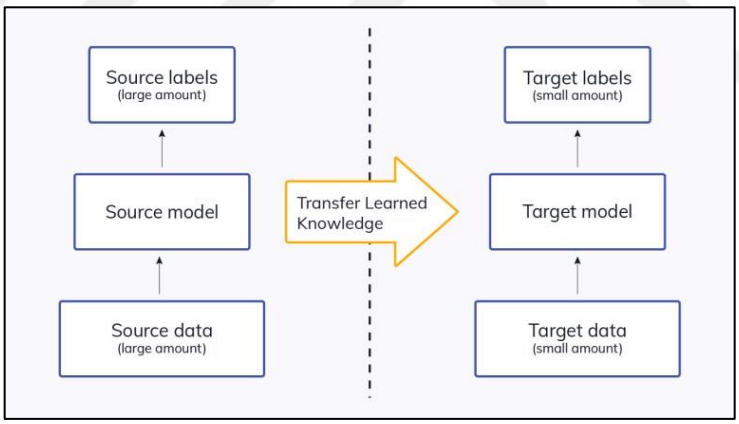
- Gizli durum (*Hidden State*): LSTM hücresinin kısa dönemli hafızasıdır.
- Çıktı geçidi (*Output Gate*): Bir sonraki gizli durumu belirleyen geçittir. Bir sonraki gizli duruma hangi çıktıların taşınacağını belirlemektedir.

LSTM hücresinin çalışma sürecinde ilk adım unutma (*forget gate*) geçididir. Bu geçitte hem önceki gizli durum hem de girdi verileri dikkate alınarak hangi bilginin daha faydalı olduğu ve hangi bilginin ne kadar unutulacağı konusunda sigmoid aktivasyonu kullanılarak 0-1 aralığında çıktı üretilmektedir. Bir bilgi ilişkisiz bulunduğu durumda bu ağ 0'a yakın, yüksek ilişki düzeyi bulunduğu durumda ise 1'e yakın değer üretilmektedir. Çıktı alınan bu değerlere önceki hücre durumu ile nokta çarpımı işlemi uygulanmaktadır. Unutma geçidinde gizli durum ve yeni veri noktası göz önüne alınarak hangi bilgilerin unutulacağına karar verilmektedir. Bir sonraki adım, yeni bellek ağını ve giriş geçidini içermektedir. Bu adımda, eski gizli durum ve yeni girdiler değerlendirilerek uzun süreli belleğe hangi bilgilerin geçirileceğine karar verilmektedir. Bu adımda hem bellek ağı hem de girdi geçidi ağı eski gizli durum ile yeni girdi verilerini girdi olarak almaktadır. Buradaki girdiler unutma geçidi girdileri ile aynı olmaktadır. Yeni bellek ağı, bir güncelleme vektörü oluşturmak amacıyla eski gizli durum ile yeni girdi verilerinin nasıl birleştirileceğini öğrenen *Tanh* aktivasyon fonksiyonuna sahip bir sinir ağıdır. Bu ağda *Tanh* aktivasyon fonksiyonu kullanıldığı için çıktı değerleri -1 ile +1 aralığında olmaktadır. Burada yeni bellek ağı tarafından oluşturulan bileşenlerden hangilerinin muhafaza edilmesi gerekliliğini belirleyen bir filtre görevi gören sigmoid aktivasyon fonksiyonuna sahip giriş geçididir. Ağ üzerindeki uzun süreli bellek güncellendikten ve gizli duruma karar verildikten sonraki son adım çıkış kapısıdır. Bu çıkış kapısında yeni hücre durumu, eski gizli durum ve yeni girdi verileri kullanılmaktadır (IBM, b.t.).

2.2.3 Aktarımlı Öğrenme (Transfer Learning)

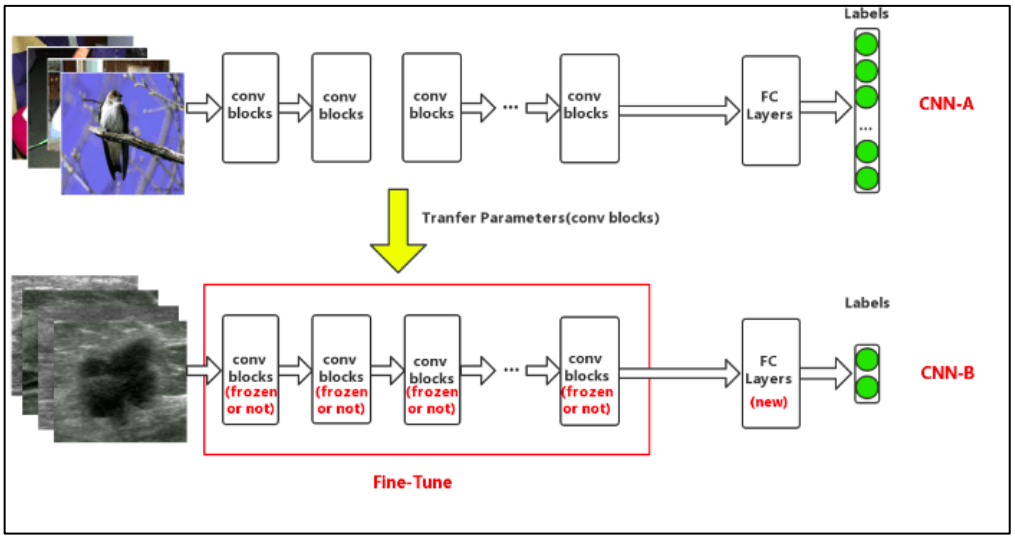
Aktarım öğrenimi, başka bir veri seti üzerinde eğitilmiş bir derin sinir ağı modelinin özellik temsillerinden yararlanma amacıyla oluşturulmuş bir yaklaşımdır. Bu yaklaşım, bazı hedef problemler için sıfırdan yeni bir derin sinir ağı modeli oluşturulması gerekliliğini ortadan kaldırmaktadır. Önceden eğitilmiş olan derin sinir ağı modelleri daha çok bilgisayarlı görü alanında standart karşılaştırma ve ölçüt olarak

kabul edilen büyük veri kümeleri (ImageNet, Coco vb. gibi) üzerinde eğitilmektedir. Bu eğitimler sonucunda modellerdeki hesaplanmış ağırlık değerleri farklı bilgisayarlı görme problemlerinin çözümünde kullanılabilmektedir. Eğitilmiş bu modeller, yeni hedef problemlerin çözümünde doğrudan tahmin oluşturmak için kullanılabilirken bunun yanı sıra yeni bir modelin eğitim sürecine de entegre edilebilmektedir. Önceden eğitilmiş modellerin veya modellerin bir kısmının yeni modele eklenmesi, eğitim süresinin düşmesini ve genelleme hatasının azalmasını sağlamaktadır. Aktarım öğrenimi, çözümü aranılan problem ile ilgili küçük bir eğitim veri kümesi olduğunda, görece sadece küçük veri kümesi üzerinde eğitimden daha iyi sonuçlar vermektedir. Önceden eğitilmiş bu modelin ağırlıkları, yeni oluşturulan derin sinir ağı modeli için bir başlangıç noktası olarak da kullanılabilmektedir. Aktarım öğreniminden bilgisayarlı görü alanı dışında doğal dil işleme alanında da faydalanılmaktadır. Aktarım öğrenimi konsepti Şekil 2.11'deki gibidir. Önceden eğitilmiş yapay sinir ağlarının bir diğer avantajı da gerçek dünya uygulamalarında kullanım için genelleşmiş olmalarıdır (Neptune.Ai, b.t.).



Şekil 2.11 Aktarım öğrenimi konsepti (Neptune.Ai, b.t.)

İnce ayarlama (*Fine-Tuning*), aktarımlı öğrenme kapsamında opsiyonel bir durumdur. İnce ayarlama model performansının iyileştirilmesi için gerçekleştirilen bir adımdır. Bu adımda modelin yeniden eğitilmesi gerekmesi nedeniyle aşırı uyum (*Overfitting*) problemi ile karşılaşılması ihtimal dahilinde olacaktır. Şekil 2.12'de ince ayarlamanın nasıl gerçekleştirilebileceği yer almaktadır (Neptune.Ai, b.t.).



Şekil 2.12 Aktarım öğrenimi ince ayarlama (*fine tuning*) yöntemi (Neptune.Ai, b.t.)

Aşırı uyum önlenebilecek bir problemdir. Modelin bir kısmı dondurulup öğrenme oranı düşürülerek aşırı uyum problemi aşılabilmektedir. Yapılan bu güncellemeler model performansında değişiklikler gözlemlenebilmektedir. Hedeflenen başarı metriğine ulaşıldığında veya maksimize (veya minimize) edilmek istenen metrikte daha fazla gelişme olmadığı görüldüğünde eğitim sonlandırılmaktadır. Aktarım öğreniminin başlıca kullanılma sebepleri aşağıdaki gibidir:

- Yüksek doğruluk oranına sahip yapay sinir ağı modelleri için veri sayısının çok fazla olması gerekmektedir. Birçok aktarım öğrenimi için referans olarak alınan modeller *ImageNet* veri kümesi ile eğitilmiştir. *ImageNet* 1 milyonu aşkın veri sayına sahiptir. Aynı veri sayılarına belirli bir probleme yönelik oluşturulmuş veri kümelerinin ulaşması mümkün değildir.
- Aktarımlı öğrenmede referans olarak alınan modeller, *ImageNet* gibi büyük bir veri kümesi üzerinde eğitilmiştir. Referans olarak alınan modeller gibi büyük yapay sinir ağı modellerini 1 milyonun üzerinde veriye sahip bir veri kümesi ile eğitmek, yüksek bir bilgi işlem gücü kapasitesine ihtiyacı beraberinde getirmektedir.
- Referans olarak alınan modeller gibi büyük modellerin *ImageNet* gibi büyük bir veri kümeleri ile eğitilmesinin yüksek bilgi işlem gücü gerekliliğinin yanı

sıra ayrıca bu eğitim süresi günler veya haftalar gerektirebilmektedir. Aktarımlı öğrenme bu süreleri ortadan kaldırabilmektedir (Neptune.Ai, b.t.).

2.2.3.1 VGG

Simonyan ve Zisserman'ın (2014) ortaya koyduğu VGG yapay sinir ağı mimarisi, *ImageNet Challenge 2014*'te yerelleştirme (*localisation*) ve sınıflandırma (*classification*) alanlarında birinci ve ikinci sırayı almıştır. VGG yapay sinir ağı, büyük ölçekli görüntü tanıma problemi üzerinde evrişimli sinir ağ derinliğinin doğru tespit başarısındaki etkisi araştırılarak ortaya konmuştur. Bu yapay sinir ağı yönteminde çok küçük (3 x 3) evrişim filtreleri kullanılarak derinliği arttırılan bir yapı ortaya konmuştur. VGG'de evrişimli katman derinliği 16 ve 19 katmanlı olarak konfigüre edilmiştir.

Yine Simonyan ve Zisserman'ın (2014) yaptığı çalışmaya göre VGG evrişimli sinir ağında sabit 224x224 üç kanallı RGB görüntüler girdi olarak kullanılmaktadır. Bu girdilerde eğitim verisinden hesaplatılmış ortalama RGB değeri, her bir pikselden çıkarılarak bir ön işleme gerçekleştirilmiştir. Görüntü, 3x3 boyutta küçük filtreler kullanan evrişimli sinir ağı katmanlarından geçirilmektedir ve 3x3'lük filtre boyutu sol-sağ, yukarı-aşağı ve merkez kavramlarının belirlenebilmesi için en küçük boyuttur. VGG konfigürasyonlarının birinde, girdi kanallarının doğrusal dönüşümü için 1x1 evrişim filtreleri de kullanılmaktadır. Evrişimdeki adım 1 piksel olarak sabitlenmiştir. Evrişimli katmanlarda yapılan dolgular ile çıktı çözünürlüğünü korunmaktadır. Tüm evrişimsel katmanlarını maksimum havuzlama katmanlarını izlememektedir. Maksimum havuzlama katmanının olduğu aşamalarda 5 havuzlama katmanı kullanılmaktadır. Maksimum havuzlama, 2 piksel boyutundaki adımlar ile 2x2'lik bir pencere ile gerçekleştirilmektedir. Evrişimsel katmanları 3 tam bağlantılı katman (*fully connected layer*) takip etmektedir. Bu katmanlardan ikisi 4096 hesaplama birimine sahiptir, son ve üçüncü katman ise her biri bir sınıfa ait olmak üzere 1000 hesaplama biriminden oluşmaktadır ve *softmax* katmanı ile yapay sinir ağı tamamlanmaktadır.

Simonyan ve Zisserman (2014) tüm gizli katmanlarda, doğrusallığı bozması için ReLU aktivasyon fonksiyonu kullanmışlardır. VGG çalışması kapsamında

hazırlanmış sinir ağlarının yalnızca birinde Yerel Tepki Normalleştirme (LRN) kullanılmıştır. Yerel tepki normalleştirmesinin ILSVRC veri seti kümesinde performansı iyileştirmediği, bunun yanı sıra bellek tüketiminin ve hesaplama süresinin artmasına sebep olduğu görülmüştür. Şekil 2.13'te VGG'nin farklı konfigürasyonları yer almaktadır. Şekil 2.14'te konfigürasyonlara ait milyonluk ölçekte parametre sayıları verilmiştir.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Şekil 2.13 VGG ağı konfigürasyonları (Simonyan ve Zisserman, 2014)

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

Şekil 2.14 Sinir ağlarının içerdiği parametre sayıları (Simonyan ve Zisserman, 2014)

Evrişimsel sinir ağının eğitimi, *minibatch* şekli ile girdiler alınmakta ve momentum kullanan gradyan inişi ile çok terimli lojistik regresyon hedefini gerçekleştirmek amacıyla optimize edilerek gerçekleştirilmektedir. *Batch* büyüklüğü 256 görüntüye ve momentum da 0.9'a ayarlanmıştır. Eğitim, ağırlık değişimi için L2 ceza çarpanı $5x10^{-4}$'e ayarlanmıştır. İlk iki tam bağlı katmanlar için *dropout* oranı 0,5'e ayarlanmıştır. Öğrenme oranı (*learning rate*) 10^{-2} olarak belirlenmiştir. Validasyon

seti üzerindeki doğru tahmin oranında iyileşme durduğunda 10^{-3}'e ayarlanmıştır. 74 *epoch* sonunda öğrenme durdurulmuştur.

ILSVRC-2012 veri seti, 1000 farklı sınıfın görüntülerini barındırmaktadır ve kullanıma göre üç veri kümesine bölünmektedir; 1,3 milyon görüntü içeren eğitim seti, 50 bin görüntü içeren validasyon seti ve 100 bin görüntü içeren sınıf etiketli test seti. Sınıflandırma başarısı iki metriğe dayanarak değerlendirilmektedir. *Top-1* hatası ve *Top-5* hatası olan bu hata metriklerinden ilki çok sınıflı sınıflandırma hatası olarak belirtilmektedir ve yanlış sınıflandırılmış görüntülerin oranını ifade etmektedir. İkinci hata metriği ise *ILSVRC*'de kullanılan ana değerlendirme metriğidir ve alınan en yüksek 5 olasılık değerinde doğru etiketi bulundurmayan tahminlerin oranıdır. Şekil 2.15'te VGG'nin diğer sinir ağları ile skorları karşılaştırılmıştır.

Method	top-1 val. error (%)	top-5 val. error (%)	top-5 test error (%)
VGG (2 nets, multi-crop & dense eval.)	**23.7**	**6.8**	**6.8**
VGG (1 net, multi-crop & dense eval.)	24.4	7.1	7.0
VGG (ILSVRC submission, 7 nets, dense eval.)	24.7	7.5	7.3
GoogLeNet (Szegedy et al., 2014) (1 net)	-	7.9	
GoogLeNet (Szegedy et al., 2014) (7 nets)	-	**6.7**	
MSRA (He et al., 2014) (11 nets)	-	-	8.1
MSRA (He et al., 2014) (1 net)	27.9	9.1	9.1
Clarifai (Russakovsky et al., 2014) (multiple nets)	-	-	11.7
Clarifai (Russakovsky et al., 2014) (1 net)	-	-	12.5
Zeiler & Fergus (Zeiler & Fergus, 2013) (6 nets)	36.0	14.7	14.8
Zeiler & Fergus (Zeiler & Fergus, 2013) (1 net)	37.5	16.0	16.1
OverFeat (Sermanet et al., 2014) (7 nets)	34.0	13.2	13.6
OverFeat (Sermanet et al., 2014) (1 net)	35.7	14.2	-
Krizhevsky et al. (Krizhevsky et al., 2012) (5 nets)	38.1	16.4	16.4
Krizhevsky et al. (Krizhevsky et al., 2012) (1 net)	40.7	18.2	-

Şekil 2.15 VGG'nin diğer yapay sinir ağı mimarileriyle karşılaştırılması (Simonyan ve Zisserman, 2014)

2.2.3.2 EfficientNet

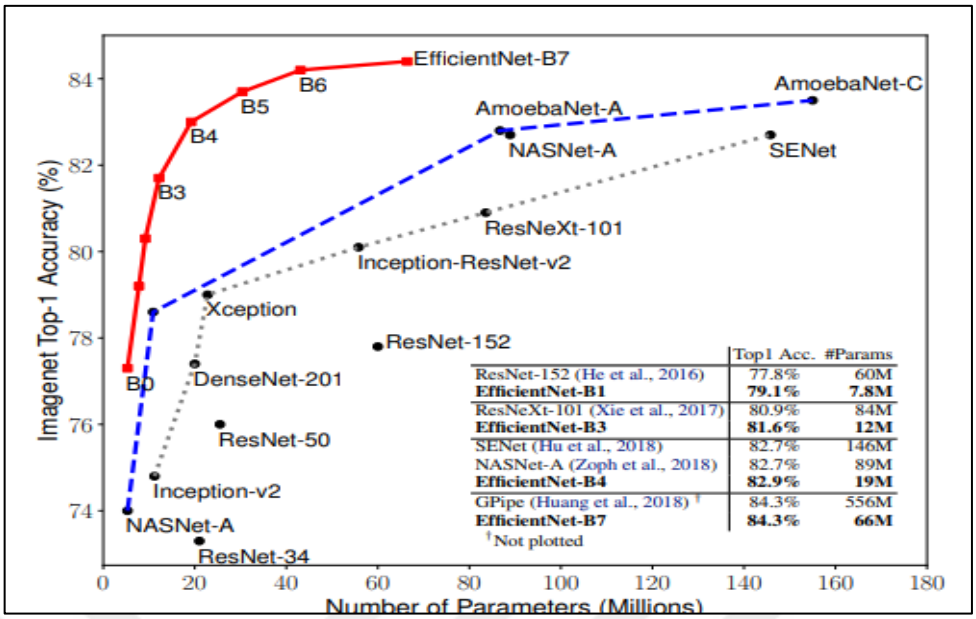
Tan ve Le'nin (2019) yaptığı çalışmaya göre evrişimli sinir ağları belli bir bilgi işlem gücü kullanılarak geliştirilmektedir ve bundan sonraki süreçte de doğruluk oranının veya maksimize edilmek istenen metrik doğrultusunda ölçeklendirilmektedir. *EfficientNet* çalışmasının temel amaçlarından biri de modelin ağ derinliği, genişliği ve çözünürlüğü üzerine doğruluk oranının yükseltilmesi ile birlikte modelin az bilgi işlem

kaynağı kullanılarak ölçeklendirilmesidir. Ağ derinliği, genişliği ve çözünürlük dengesinin sağlanması olarak da ifade edilebilir. Yapılan çalışmada bir bileşik katsayı ile ağ derinliğini, genişliği ve çözünürlüğü ölçekleyen bir yöntem ortaya konmuştur. Bu yöntem ile birlikte ortaya *EfficientNet* adlı bir model oluşturulmuştur. Şekil 2.16'da *EfficientNet-B0*'ın mimarisi görülmektedir.

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

Şekil 2.16 EfficientNet-B0 ağ mimarisi (Tan ve Le, 2019)

Kurulan yeni sinir ağı yapılarından *EfficientNet-B7*, mevcut en iyi evrişimli sinir ağından 8,4 kat daha küçük ve 6,1 kat daha hızlı tahminleme yapmakta ve *ImageNet* veri kümesi üzerinde %84,3 *Top-1* doğruluk oranına ulaşmıştır. *EfficientNet*, aktarımlı öğrenme aracı olarak *CIFAR-100* veri kümesinde %91,7 doğruluk oranına ulaşırken *Flowers* veri kümesinde %98,8 doğruluk oranına ulaşmaktadır. Bununla birlikte diğer evrişimli sinir ağlarına göre daha az parametre içermektedir. *ImageNet* veri seti üzerindeki sonuçlar Şekil 2.17'de görülmektedir.



Şekil 2.17 Model büyüklüğüne karşılık doğruluk oranı grafiği (Tan ve Le, 2019)

BÖLÜM 3
DENEY VE BULGULAR

Bu bölümde, yapılan şiddet sınıflandırma çalışması ile fiziksel şiddet sistemleri için geliştirilmiş öncül fonksiyon anlatılmaktadır. Tezin ana odak noktası olması sebebiyle öncelikli olarak videolardaki fiziksel şiddetin sınıflandırılması ele alınmıştır. Sonra da bu sınıflandırıcı benzeri yüksek işlem kapasitesi gerektirecek sistemlerin sürekli çalışması gerekliliğinin ortadan kaldırılması amacıyla geliştirilmiş olan öncül fonksiyon anlatılacaktır.

3.1 Evrişimli Ağlar ve Uzun-Kısa Süreli Hafıza Ağları ile Sınıflandırma

Veri kümesi içerisindeki videoların eğitime hazırlanması için video görüntülerinin sıkıştırılmış formattan matris formatında hafızaya alınması gerekmektedir. VGG-19 ile yapılan testler önce 100 video kare sayısı ile gerçekleştirilmiştir. Bu video sayısı, sonraki deneylerde bilgi işlem gücü kısıtları nedeniyle 34 video kare sayısına düşürülmüştür. Önceden yüklenmiş olan 100 video karelik paketlerden her üçüncü karenin alınması ile toplam kare sayısı her bir video için yaklaşık olarak 1/3 oranına düşürülmüştür. Sonraki bölümlerde ifade edilecek olan deney sonuçları 34 video karesi içeren eğitim ve testlerin sonuçlarından oluşmaktadır.

Video görüntüleri, fiziksel şiddet içeren ve fiziksel şiddet içermeyen olmak üzere iki sınıftan oluşmaktadır. Bu ikili sınıflandırmada etiket olarak fiziksel şiddet içermeyen görüntüler için [1, 0], fiziksel şiddet içeren görüntüler için [0, 1] matrisleri kullanılmıştır.

Video görüntülerinde 100 karelik görüntü dizisinden eşit zaman aralıklı 34 adet kare belirlendiğinden uygun kare sayısına sahip olmayan videolar, eğitim kümesi içerisinden çıkartılmıştır. Burada 2000 adet olan toplam veri sayısı 1857'ye düşmüştür. Ancak bu düşüş etiket dağılım dengesini büyük ölçüde bozmaması sebebiyle etiket sayısının dengelenmesi konusunda bir değişiklik yapılmamıştır. Son

durumda fiziksel şiddet içerikli videoların sayısı 975, fiziksel şiddet içermeyen videoların sayısı 882 olmuştur.

Bu çalışmada video görüntülerinden özellik çıkarımı yapılabilmesi için iki farklı evrişimli sinir ağı mimarisi ile aktarımlı öğrenme gerçekleştirilmiştir. Bunlar VGG-19 ve EfficientNet-B0'dır. Eğitimin toplam süresinin kısaltılması amacıyla evrişimli sinir ağı çıktıları kaydedilmiştir. Uygulanan bu yöntem ile geliştirme sürecinde karşılaşılan hatalar nedeniyle veri kaybıyla yaşanabilecek süre kayıplarının önüne geçilmiştir.

İki ayrı evrişimli sinir ağından çıkartılan özellik haritaları, ağların yapıca birbirinden farklı olması sebebiyle tensör boyutlarında farklılık göstermektedir. Bu durum, iki farklı özellik haritasına uygun giriş yapısına sahip sınıflandırma sinir ağını gerekli kılmıştır. Bunun yanı sıra kullanılan sınıflandırma sinir ağının belli boyutsal kısıtlarının olması bu çıkılarda yeniden şekillendirme (*reshape*) işlemini zorunlu kılmıştır. Yeniden şekillendirme ile sınıflandırma sinir ağı girdileri Tablo 3.1'deki gibi olmuştur.

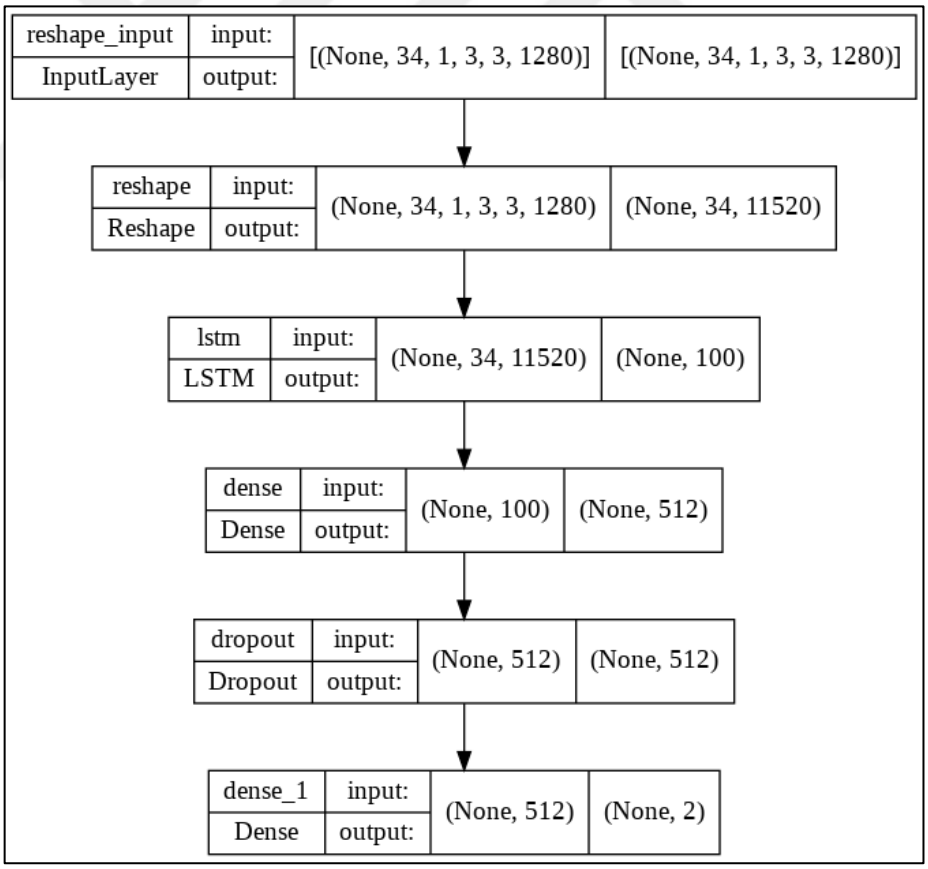
Tablo 3.1 Evrişimli sinir ağlarının çıktı boyutları

Sinir Ağı Adı	**Çıktı Şekli**
VGG-19	(34, 2048)
EfficientNet-B0	(34, 1, 3, 3, 1280)

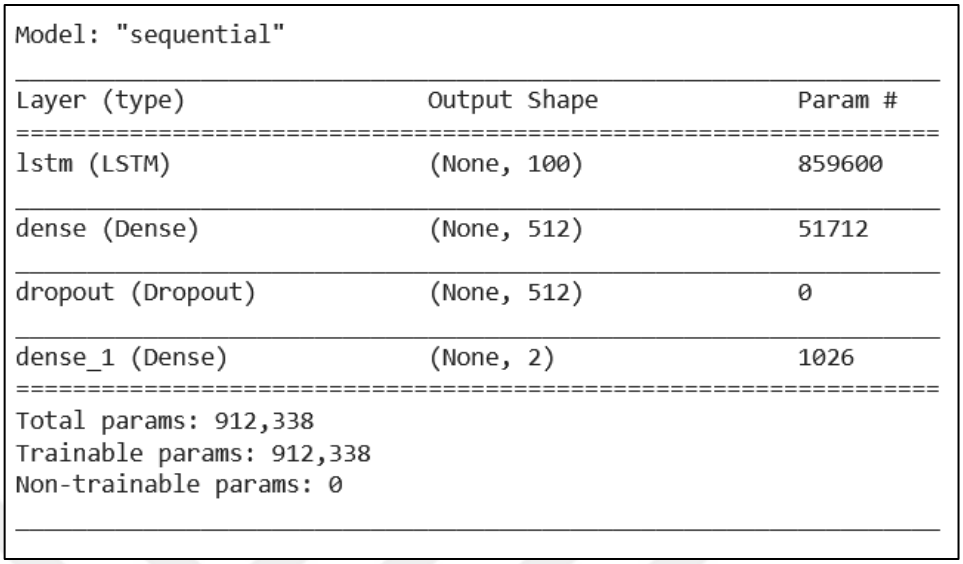
Özellik haritalarının çıkartılıp dosyalar halinde kaydedilmesinin ardından kaydedilen bu dosyalar tekrar okunup etiket değerleri ile eşleştirilerek sınıflandırma ile görevli sinir ağı için veri kümesi oluşturulmuştur. Burada veri kümesinin %25'i test kümesini oluşturması için rastgele seçilerek ayrıştırılmıştır. Deneyin aynı sonuçları üretmesi için rastgele durum sayısı 42 olarak seçilmiştir.

Bu çalışma kapsamında kullanılacak olan sınıflandırma sinir ağı, uzun-kısa dönemli hafıza (*LSTM*) ve tam bağlı katmanlar (*Fully connected layers*) ile oluşturulmuştur. Model aşağıdaki şekilde kurgulanmıştır;

- Modelde ilk katman girdi şeklinin sıralı tek bir diziye dönüştürülmesidir.
- İkinci katman ardışık ilişkinin çıkartılmasını sağlayacak olan uzun-kısa dönemli hafıza katmanıdır. Bu katmanda 100 yapay sinir ağı hücresi kullanılmıştır. Seriye ait aşırı öğrenme oluşmaması için özyineli düşürme (*recurrent dropout*) kullanılmıştır. Yapılan farklı denemeler sonucunda özyineli düşürme %10 ve aktivasyon fonksiyonu olarak da *Tanh* seçilmiştir.
- Üçüncü katmanda 512 yapay sinir hücresi bulunduran tam bağlı katman bulunmaktadır. Aktivasyon fonksiyonu olarak *ReLU* kullanılmıştır.
- Dördüncü katman, düşürme katmanıdır. Aşırı öğrenme probleminin aşılması için %30 oranında düşürme uygulanmıştır.
- Beşinci ve son katman *sigmoid* aktivasyon fonksiyonuna sahip tam bağlı katmandır. Burada hedef etiket değerlerine uygun olarak 2 yapay sinir ağı hücresi bulunmaktadır.



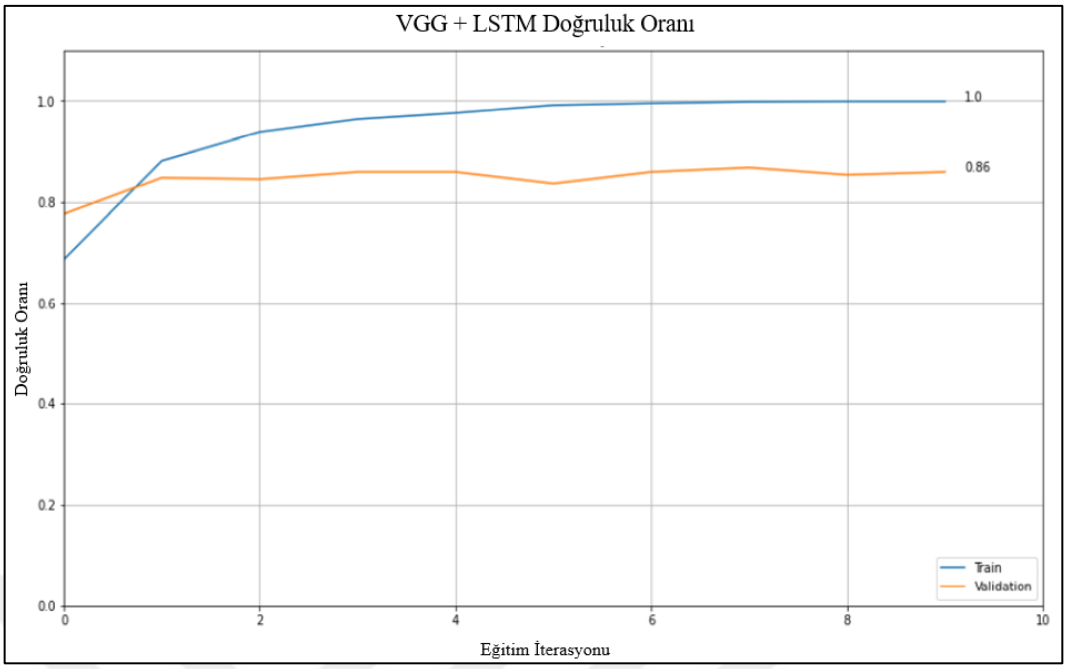
Şekil 3.1 Sınıflandırma ağının yapısı



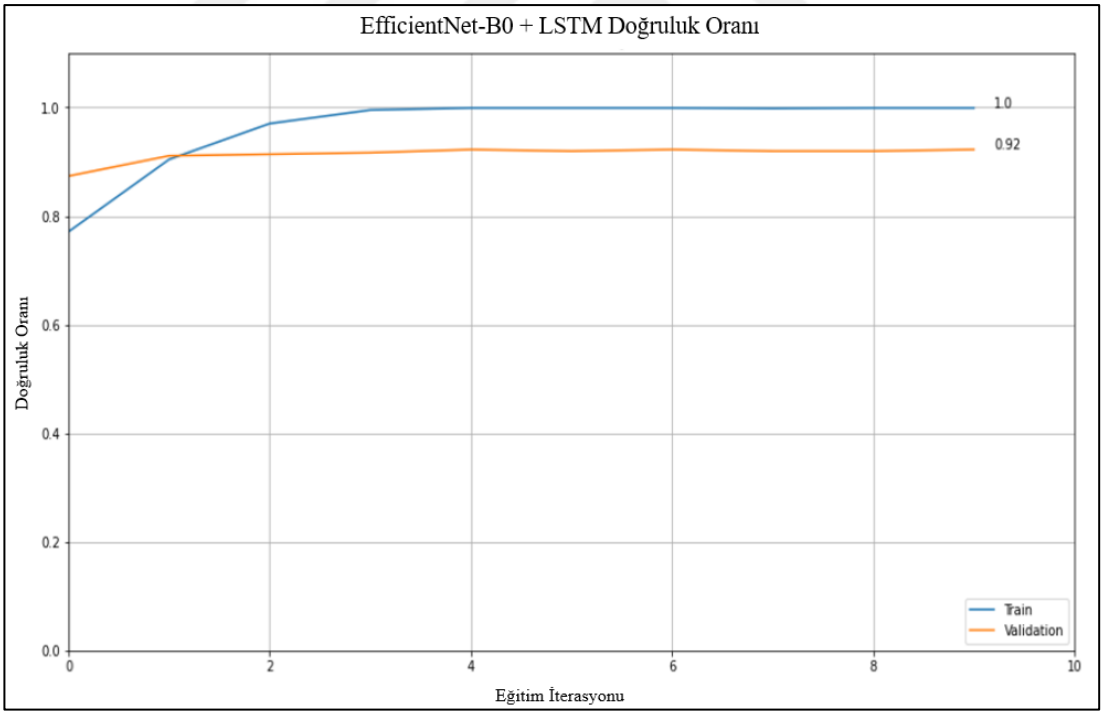
```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 100)               859600
_________________________________________________________________
dense (Dense)                (None, 512)               51712
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 1026
=================================================================
Total params: 912,338
Trainable params: 912,338
Non-trainable params: 0
_________________________________________________________________
```

Şekil 3.2 LSTM ağı katman ve parametre sayıları

Şekil 3.1 ve Şekil 3.2'de sınıflandırma ağının yapısı, katman ve parametre sayıları verilmiştir. Optimizasyon için *Adam* algoritması kullanılmıştır, kayıp (*loss*) fonksiyonu için *binary cross entropy* kullanılmıştır. Eğitim sürecinde doğruluk oranı üzerinden takip yapılmıştır. Kurulmuş olan bu sınıflandırıcı sinir ağı toplamda 4.701.138 eğitilebilir parametre içermektedir. Eğitilmeyen ya da dondurulan katman ve hücre barındırmamaktadır. Eğitimde, eğitim verisinin %25'i validasyon için kullanılmıştır. Eğitim toplamda 10 *epoch* (eğitim döngüsü) sürmüştür. Eğitim döngüsünün sayısının arttırılması validasyonda bir iyileşmeyi beraberinde getirmemiştir. Şekil 3.3 ve Şekil 3.4'te eğitim süresince doğruluk oranının eğitim ve validasyon verilerindeki değişimi görülmektedir.



Şekil 3.3 VGG + LSTM eğitim ve validasyon doğruluk oranı grafiği



Şekil 3.4 EfficientNet-B0 + LSTM eğitim ve validasyon doğruluk oranı grafiği

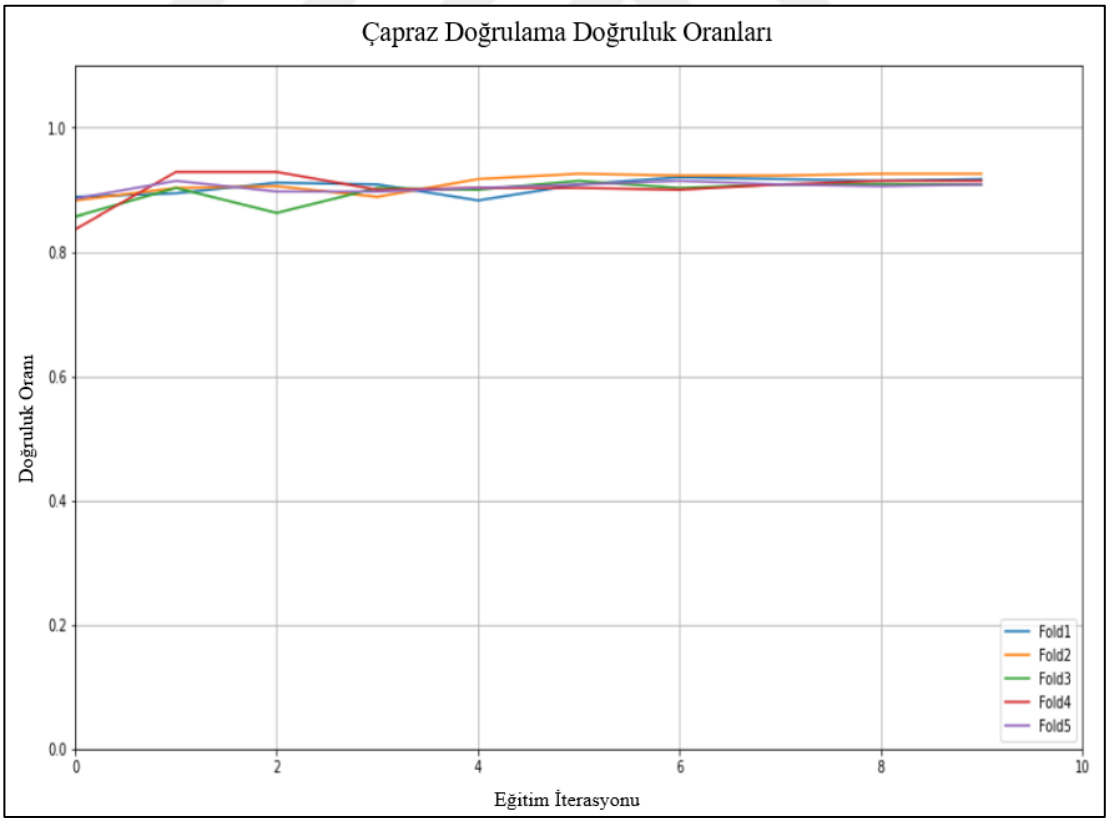
İki farklı evrişimli sinir ağına sahip sistemler için test sonuçları Tablo 3.2'deki gibidir.

Tablo 3.2 ConvLSTM test verisi üzerindeki doğruluk oranları

Sinir Ağı Adı	**Doğruluk Oranı**
VGG-19 + LSTM	%89
EfficientNet-B0 + LSTM	%90

Tablo 3.2'de görüldüğü üzere EfficientNet-B0 ile kurulan ConvLSTM sistemi hem validasyon doğruluk oranı ile hem de test doğruluk oranı ile daha iyi bir performans sergilemiştir.

EfficientNet-B0 ile kurulmuş olan ConvLSTM sistemleri için yapılan çapraz doğrulamaya ait doğruluk oranı grafiği Şekil 3.5'teki gibidir.



Şekil 3.5 *EfficientNet-B0* + LSTM ağı çapraz validasyon doğruluk oranı grafiği

Yapılan sınıflandırma sonucunda test verisi üzerindeki diğer sınıflandırma metriği Tablo 3.3'te verilmiştir.

Tablo 3.3 ConvLSTM test sonucunda çıkan metrikler

Sınıf	**Precision**	**Recall**	**F1**
[1, 0]	0,91	0,88	0,90
[0, 1]	0,89	0,92	0,90

Tablo 3.3'teki değerlere bakıldığında diğer metriklerde ekstrem bir değişiklik veya tek yönlü karar verme gözlemlenmemiştir.

3.2 Fiziksel Şiddet Tespiti Sistemleri için Öncül Fonksiyon

Polat ve Alın'ın (2021) yaptığı çalışmaya göre çeşitli alanlarda kullanılan yapay sinir ağları, donanıma, ağ derinliğine, ağ genişliğine ve kullanılan algoritmaya bağlı olarak yüksek işlem gücü birimlerine ihtiyaç duymaktadır. Artan işlem gücü ihtiyacının karşılanması cihazların daha yüksek güç tüketimine ihtiyaç duymasını beraberinde getirecektir.

Videolardaki fiziksel şiddetin tespit edilmesinde, bu çalışma kapsamında ortaya konmuş olan yöntem ve benzeri türevleri ile oluşturulmuş yapay sinir ağı yapılarının sürekli olarak aktif tutulması, yapılan işlemlerin karmaşıklığı nedeniyle tahmin çıkarımı vakit almakta, tahmin çıkarımındaki bekleme süresi nedeniyle gerçek zamanlı olma özelliğini yitirebilmekte ve yüksek enerji gerektirmektedir.

Yukarıda belirtilen sebepler ile bu çalışmada bir fiziksel şiddet tespiti öncül fonksiyonu önerilmiştir. Burada önerilen yöntem "zamana bağlı mod değişkenliği" olarak tanımlanmaktadır.

Mod, bir sayı dizisi içerisinde en çok tekrar eden sayı olarak ifade edilmektedir. Mod işlemi, zamana bağlı olarak bir video üzerine uygulandığı takdirde arka plana ait

görüntü matrisine ulaşabilmektedir. Zheng ve arkadaşlarının (2006) mod işlemi ile yaptığı çalışmanın sonuçları Şekil 3.6'da görülmektedir.

Şekil 3.6 Otoyol videosuna yapılan arka plan görüntüsünün bulunması (Zheng vd., 2006)

Bu çalışmada önerilmiş olan yöntem, kısa süreli video kliplerinde en fazla tekrar eden değeri, o klibin arka planı olarak kabul ederek o lokasyon için arka plan değerinden ne kadar değişkenlik gösterdiğini bulmaya dayanmaktadır. Önerilen bu yöntem, diğer hareket tespiti algoritmalarından temelde tek yönlü ve zamana bağlı az değişim gösteren eylemlerin filtrelenmesini sağlamaktadır.

Öncül fonksiyon ile oluşturulan mod matrisi, her x-y piksel lokasyonu için referans değer olarak kabul edilmektedir. Bu referans değerler, zamana bağlı piksel dizilerinde varyans hesaplamasında ortalama yerine kullanılmaktadır. Yapılan bu varyans işlemi sonucunda x-y piksel lokasyonları için aşırı hareketlilik matrisi oluşturulmaktadır. Yapılan işlemler sırasıyla aşağıdaki eşitlikler (eşitlik 3.1, eşitlik 3.2, eşitlik 3.3, eşitlik 3.4) ile gerçekleştirilmektedir. *M*, 3 boyutlu video matrisi olarak kabul edilirse (M_{mnz});

$$M_z = \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mn} \end{bmatrix} \tag{3.1}$$

$$mod(M_{zaman}) = Y \tag{3.2}$$

$$M_{aşırıhareket} = \frac{\sum(M_{mnz} - Y)^2}{N} \tag{3.3}$$

$$M_{aşırıhareket} = \begin{bmatrix} \hat{x}_{11} & \cdots & \hat{x}_{1n} \\ \vdots & \ddots & \vdots \\ \hat{x}_{m1} & \cdots & \hat{x}_{mn} \end{bmatrix} \quad (3.4)$$

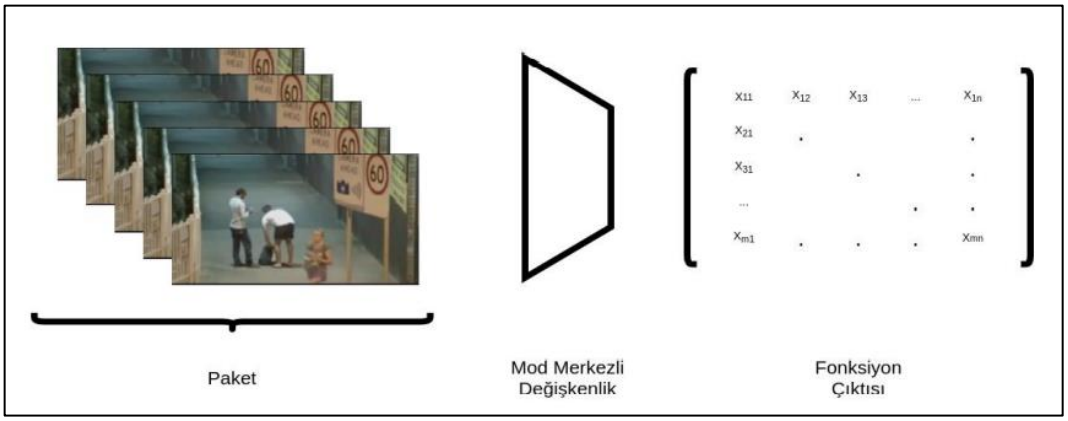
Bu eşitliklerde M_z video için temsili kare (*frame*) matrisini, m, n ve z matris boyutlarını, Y mod matrisini ve $M_{aşırıhareket}$ de aşırı hareket matrisini ifade etmektedir. Önerilen aşırı hareket öncül fonksiyonuna ait algoritma Şekil 3.7'deki gibidir.

Algoritma

Boş bir 3 boyutlu dizi (paket) oluştur.
Video kareleri için döngü başlat:
    Video karesini gri ölçeğe çevir.
    Video karesini pakete ekle.
    Eğer paketteki kare sayısı belirlenen sayıya ulaştı ise:
        Piksel lokasyonları için mod bul.
        Zaman ekseninde mod ile piksel farklarını al.
        Karelerini topla.
        Toplam kare sayısına böl.

Şekil 3.7 Aşırı hareket öncül fonksiyonu algoritması

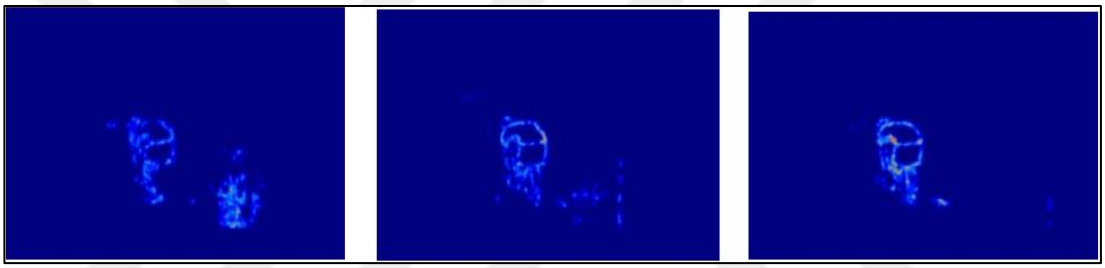
Fonksiyona ait dönüşüm işleminin konsept çıktısı Şekil 3.8'deki gibidir.



Şekil 3.8 Öncül fonksiyon temsili çıktısı

Önerilen yöntem, *UCF-Crime* veri setinden sabit kamera ile kaydedilmiş bazı video görüntüleri üzerinde test edilmiştir. Testlerde paket büyüklükleri 30, 60 ve 90 kare olarak belirlenmiştir.

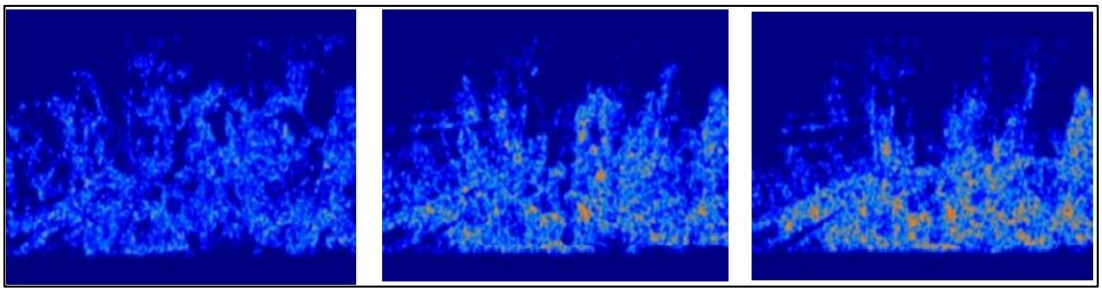
Şekil 3.9 Orijinal video karesi (Sultani vd., 2018)



Şekil 3.10 30, 60 ve 90'lık paketlerde öncül fonksiyon çıktıları (Polat ve Alın, 2021)

Bu videoda (Şekil 3.9) 30 ve 60 kare büyüklüğündeki paketlerde tek yönlü hareket gerçekleştiren bireye ait pikseller de tespit edildiği görülmüştür. Ancak 90 kare büyüklüğündeki pakette alınan çıktıda bu bireye ait hareketin filtrelendiği görülmüştür. Çıktılar Şekil 3.10'daki gibidir.

Şekil 3.11 Orijinal video karesi (Sultani vd., 2018)



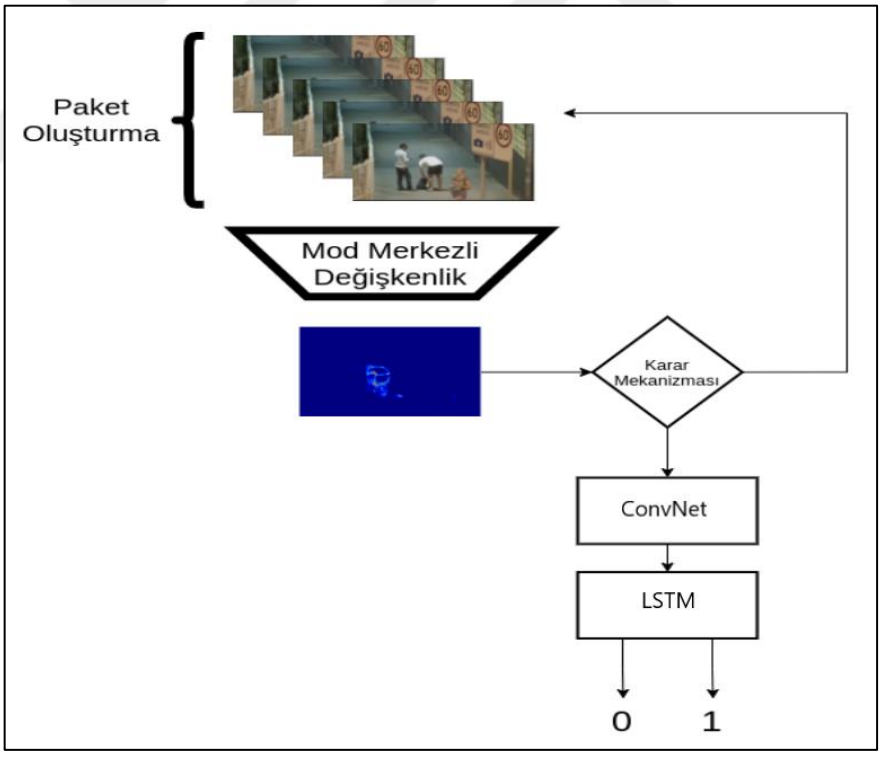
Şekil 3.12 30, 60 ve 90'lık paketlerde öncül fonksiyon çıktıları (Polat ve Alın, 2021)

Bu videodaki (Şekil 3.11) insanlar kalabalık bir grup halinde sahne boyunca dağılmıştır ve aşırı hareket göstermektedirler. Burada paket büyüklüğünün artışı ile birlikte aşırı hareket olarak algılanan bölgelerin belirginliğinin arttığı görülmüştür. Çıktılar Şekil 3.12'deki gibidir.

BÖLÜM 4
SONUÇLAR

Oluşturulan yapay sinir ağı yapısı (ConvLSTM) ile video kliplerinde %90 doğruluk oranı ile fiziksel şiddetin varlığı ve yokluğu tespit edilmiştir. Daha optimize ve daha yüksek işlem gücüne sahip yapay sinir ağları ile doğruluk oranı daha da yükseltilebilir.

Oluşturulan ConvLSTM sistemi ve bu sistem gibi yapay sinir ağı mimarileri ile fiziksel şiddetin tespit edilebilirliği ortaya konmuştur. Bu çalışma kapsamında önerilen aşırı hareket öncül fonksiyonu gibi yöntemler ile fiziksel şiddet tespiti sistemlerinin kullanımı, bu sistemlerin kullanım maliyetlerini düşürebilir ve kullanım yaygınlığını arttırabilir.



Şekil 4.1 Öncül fonksiyonlu ConvLSTM uygulama akışı

Şekil 4.1'de öncül fonksiyon ile fiziksel şiddet tespit sistemi için örnek uygulama şeması verilmiştir. Önerilen aşırı hareket öncül fonksiyonu, bu çalışmada belirtildiği

şekli ile her video üzerinde başarılı sonuçlar üretememektedir. Bu fonksiyonun daha da geliştirilebilmesi ve başarısının gösterilebilmesi için bir değerlendirme metriğine ihtiyaç duymaktadır. Belirlenecek bu metrik ile öncül fonksiyonun veya türevlerinin başarısı ölçülebilir ve fonksiyonlar üzerinde parametrelerin genelleştirilmesi üzerine çalışılabilir.

KAYNAKLAR

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., vd. (23 Ağustos 2022). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*. www.tensorflow.org.

Abdali, A. M. R., ve Al-Tuma, R. F. (2019). Robust Real-Time Violence Detection in Video Using CNN and LSTM. *SCCS 2019 - 2019 2nd Scientific Conference of Computer Sciences*, 104-108. https://doi.org/10.1109/SCCS.2019.8852616

Analyticsvidhya. (03 Eylül 2022). *Artificial Neural Network | How does Artificial Neural Network Work*. https://www.analyticsvidhya.com/blog/2021/04/artificial-neural-network-its-inspiration-and-the-working-mechanism/

Arceda, V. M., Fabián, K. F., ve Gutíerrez, J. C. (2016). Real time violence detection in video. *IET Seminar Digest*, *2016*(2). https://doi.org/10.1049/ic.2016.0030

Arm®. (23 Ağustos 2022). *What is pattern recognition.* https://www.arm.com/glossary/pattern-recognition

Bilinski, P., ve Bremond, F. (2016). Human violence recognition and detection in surveillance videos. *2016 13th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2016*, *August*, 30-36. https://doi.org/10.1109/AVSS.2016.7738019

Chahal, K. S., ve Dey, K. (2018). *A Survey of Modern Object Detection Literature using Deep Learning*. http://arxiv.org/abs/1808.07256

Cheng, F. C., Huang, S. C., ve Ruan, S. J. (2010). Advanced motion detection for intelligent video surveillance systems. *Proceedings of the ACM Symposium on Applied Computing*, 983-984. https://doi.org/10.1145/1774088.1774295

Chollet, F. (2017). *Deep Learning with Python*. www.manning.com

Deniz, O., Serrano, I., Bueno, G., ve Kim, T. K. (2014). Fast Violence Detection in Video. *Proceedings of the 9th International Conference on Computer Vision Theory and Applications*, *2*, 478-485. https://doi.org/10.5220/0004695104780485

Ditsanthia, E., Pipanmaekaporn, L., ve Kamonsantiroj, S. (2019). Video Representation Learning for CCTV-Based Violence Detection. *TIMES-ICON 2018 - 3rd Technology Innovation Management and Engineering Science International Conference*, 1-5. https://doi.org/10.1109/TIMES-iCON.2018.8621751

Dorogyy, Y., Kolisnichenko, V., ve Levchenko, K. (2018). Violent crime detection system. *2018 IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies, CSIT 2018 - Proceedings*, *1*, 352-355. https://doi.org/10.1109/STC-CSIT.2018.8526596

Du, Q., Liu, K., Yang, H., ve Ma, B. (2010). Optical flow and principal component analysis-based motion detection in outdoor videos. *Eurasip Journal on Advances in Signal Processing*, *2010*. https://doi.org/10.1155/2010/680623

Feng, W., Guan, N., Li, Y., Zhang, X., ve Luo, Z. (2017). *Audio visual speech recognition with multimodal recurrent neural networks*. 681–688. https://doi.org/10.1109/IJCNN.2017.7965918

GeeksforGeeks. (03 Eylül 2022). *Intuition of Adam Optimizer.* https://www.geeksforgeeks.org/intuition-of-adam-optimizer/

Google Colab. (23 Ağustos 2022). *Frequently Asked Questions* https://research.google.com/colaboratory/faq.html

Goto, S., ve Aoki, T. (2014). *Violent Scenes Detection Using Mid-Level Violence Clustering*. 283-296. https://doi.org/10.5121/csit.2014.4224

Ha, J., Park, J., Kim, H., Park, H., ve Paik, J. (2018). Violence detection for video surveillance system using irregular motion information. *International Conference on Electronics, Information and Communication, ICEIC 2018*, *2018-Janua*, 1-3. https://doi.org/10.23919/ELINFOCOM.2018.8330609

Hanson, A., Pnvr, K., Krishnagopal, S., ve Davis, L. (2019). Bidirectional convolutional LSTM for the detection of violence in videos. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11130 LNCS*, 280-295. https://doi.org/10.1007/978-3-030-11012-3_24

Hassner, T., Itcher, Y., ve Kliper-Gross, O. (2012). Violent flows: Real-time detection of violent crowd behavior. *IEEE Computer Society Conference on Computer*

Vision and Pattern Recognition Workshops, 1–6. https://doi.org/10.1109/CVPRW.2012.6239348

Horhan, M., ve Eidenberger, H. (2013). New content-based features for the distinction of violent videos and martial arts. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 1836-1839. https://doi.org/10.1109/ICASSP.2013.6637970

Huang, S. C. (2011). An advanced motion detection algorithm with video quality analysis for video surveillance systems. *IEEE Transactions on Circuits and Systems for Video Technology*, *21*(1), 1-14. https://doi.org/10.1109/TCSVT.2010.2087812

Huang, T. S., Schreiber, W. F., ve Tretiak, O. J. (1971). Image Processing. *Proceedings of the IEEE*, *59*(11), 1586-1609. https://doi.org/10.1109/PROC.1971.8491

IBM. (23 Ağustos 2022). *What are Convolutional Neural Networks.* https://www.ibm.com/cloud/learn/convolutional-neural-networks

IBM. (03 Eylül 2022). *What are Recurrent Neural Networks*. https://www.ibm.com/cloud/learn/recurrent-neural-networks

IBM. (23 Ağustos 2022). *What is Computer Vision*. https://www.ibm.com/topics/computer-vision

Janai, J., Güney, F., Behl, A., ve Geiger, A. (2017). *Computer Vision for Autonomous Vehicles: Problems, Datasets and State of the Art*. http://arxiv.org/abs/1704.05519

Keras. (03 Eylül 2022). *Losses.* https://keras.io/api/losses/

Kurylyak, Y. (2009). A real-time motion detection for video surveillance system. *Proceedings of the 5th IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS'2009*, 386-389. https://doi.org/10.1109/IDAACS.2009.5342954

Lee, J., ve Bovik, A. (2009). Video Surveillance. Içinde *The Essential Guide to Video Processing* (ss. 619-651). Elsevier Inc. https://doi.org/10.1016/B978-0-12-374456-2.00022-0

Li, J., Jiang, X., Sun, T., ve Xu, K. (2019). Efficient violence detection using 3D convolutional neural networks. *2019 16th IEEE International Conference on*

Advanced Video and Signal Based Surveillance, AVSS 2019. https://doi.org/10.1109/AVSS.2019.8909883

Li, X., Ng, M. K., ve Yuan, X. (2015). Median filtering-based methods for static background extraction from surveillance video. *Numerical Linear Algebra with Applications*, *22*(5), 845-865. https://doi.org/10.1002/nla.1981

MATLAB & Simulink. (23 Ağustos 2022). *Image Transform.* https://ch.mathworks.com/discovery/image-transform.html

Mishra, R., Mittal, N., ve Khatri, S. K. (2019). Digital Image Restoration using Image Filtering Techniques. *2019 International Conference on Automation, Computational and Technology Management (ICACTM)*, 268–272. https://doi.org/10.1109/ICACTM.2019.8776813

OpenCV. (23 Ağustos 2022), *About.* https://opencv.org/about/

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., ve Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*(85), 2825–2830. http://jmlr.org/papers/v12/pedregosa11a.html

Polat, M.F., ve Alın, A. Sabit Kameralı Derin Öğrenme Fiziksel Şiddet Tespiti Sistemi için Aşırı Hareket Öncül Fonksiyonu, Second International Applied Statistics Conference (UYIK-2021), (179-184), Tokat, Türkiye. ISBN: 978-975-7328-80-3. https://www.uyik.org/uploads/proceedings-book-uyik-2021.pdf

Python. (23 Ağustos 2022). *History and License — Python 3.10.6 documentation* https://docs.python.org/3/license.html

Ren, S., Chen, G., Li, T., Chen, Q., ve Li, S. (2018). A deep learning-based computational algorithm for identifying damage load condition: An artificial intelligence inverse problem solution for failure analysis. *CMES - Computer Modeling in Engineering and Sciences*, *117*(3), 287-307. https://doi.org/10.31614/cmes.2018.04697

Schedi, M., Sjöberg, M., Mironicə, I., Ionescu, B., Quang, V. L., Jiang, Y. G., ve Demarty, C. H. (2015). VSD2014: A dataset for violent scenes detection in Hollywood movies and web videos. *Proceedings - International Workshop on*

Content-Based Multimedia Indexing, *2015-July*. https://doi.org/10.1109/CBMI.2015.7153604

Scikit-Learn. (03 Eylül 2022). *sklearn.metrics.log_loss — scikit-learn 1.1.2 documentation*. gönderen https://scikit-learn.org/stable/modules/generated/sklearn.metrics.log_loss.html

Simonyan, K., ve Zisserman, A. (2014). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. http://arxiv.org/abs/1409.1556

Soliman, M. M., Kamal, M. H., El-Massih Nashed, M. A., Mostafa, Y. M., Chawky, B. S., ve Khattab, D. (2019). Violence Recognition from Videos using Deep Learning Techniques. *2019 Ninth International Conference on Intelligent Computing and Information Systems (ICICIS)*, 80–85. https://doi.org/10.1109/ICICIS46948.2019.9014714

Sudhakaran, S., ve Lanz, O. (2017). Learning to detect violent videos using convolutional long short-term memory. *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2017*. https://doi.org/10.1109/AVSS.2017.8078468

Sultani, W., Chen, C., ve Shah, M. (2018). Real-World Anomaly Detection in Surveillance Videos. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6479–6488. https://doi.org/10.1109/CVPR.2018.00678

Tan, M., ve Le, Q. v. (2019). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. http://arxiv.org/abs/1905.11946

Tian, Y. L., ve Hampapur, A. (2005). Robust salient motion detection with complex background for real-time video surveillance. *Proceedings - IEEE Workshop on Motion and Video Computing, MOTION 2005*, 30-35. https://doi.org/10.1109/ACVMOT.2005.106

Neptune.ai. (03 Eylül 2022). *Transfer Learning Guide: A Practical Tutorial With Examples for Images and Text in Keras*. https://neptune.ai/blog/transfer-learning-guide-examples-for-images-and-text-in-keras

Turaga, P., Chellappa, R., ve Veeraraghavan, A. (2010). Advances in Video-Based Human Activity Analysis: Challenges and Approaches. Içinde *Advances in Computers* (C. 80, Issue C). https://doi.org/10.1016/S0065-2458(10)80007-5

Ullah, F. U. M., Ullah, A., Muhammad, K., Haq, I. U., ve Baik, S. W. (2019). Violence detection using spatiotemporal features with 3D convolutional neural network. *Sensors (Switzerland)*, *19*(11), 1-15. https://doi.org/10.3390/s19112472

University of Queensland - Queensland Brain Institute. (23 Ağustos 2022). *Central Nervous System: brain and spinal cord.* https://qbi.uq.edu.au/brain/brain-anatomy/central-nervous-system-brain-and-spinal-cord

Wikimedia Commons. (6 Eylül 2022). *File:LSTM_cell.svg.* https://commons.wikimedia.org/wiki/File:LSTM_cell.svg

Wikimedia Commons. (6 Eylül 2022). *File:TensorFlowLogo.svg.* https://commons.wikimedia.org/wiki/File:TensorFlowLogo.svg

Zhan, C., Duan, X., Xu, S., Song, Z., ve Luo, M. (2007). An improved moving object detection algorithm based on frame difference and edge detection. *Proceedings of the 4th International Conference on Image and Graphics, ICIG 2007*, 519-523. https://doi.org/10.1109/ICIG.2007.153

Zhou, P., Ding, Q., Luo, H., ve Hou, X. (2018). Violence detection in surveillance video using low-level features. *PLoS ONE*, *13*(10), 1-15. https://doi.org/10.1371/journal.pone.0203668

Zou, B., Nurudeen, M., Zhu, C., Zhang, Z., Zhao, R., ve Wang, L. (2018). A neuro-fuzzy crime prediction model based on video analysis. *Chinese Journal of Electronics*, *27*(5), 968-975. https://doi.org/10.1049/cje.2018.02.019