T.C.

TRAKYA ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

İNSAN AKTİVİTELERİNİN DERİN ÖĞRENME TEKNİĞİ İLE TESPİT EDİLMESİ

ASIM ERKUT

YÜKSEK LİSANS TEZİ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

Tez Danışmanı: Doç. Dr. İlhan UMUT

EDİRNE-2019

ASIM ERKUT 'un hazırladığı "İnsan Aktivitelerinin Derin Öğrenme Tekniği İle Tespit Edilmesi" başlıklı bu tez, tarafımızca okunmuş, kapsam ve niteliği açısından Bilgisayar Mühendisliği Anabilim Dalında bir Yüksek lisans tezi olarak kabul edilmiştir.

Jüri Üyeleri (Ünvan, Ad, Soyad): İmza

Doç. Dr. İlhan UMUT .....

Dr.Öğr.Üyesi Özlem UÇAR .....

Dr.Öğr.Üyesi Edip Serdar GÜNER .....

Tez Savunma Tarihi:09/07/2019

Bu tezin Yüksek Lisans tezi olarak gerekli şartları sağladığını onaylarım.

İmza

Doç. Dr. İlhan UMUT .....

Tez Danışmanı

Trakya Üniversitesi Fen Bilimleri Enstitüsü onayı

Prof. Dr. Murat YURTCAN

Fen Bilimleri Enstitüsü Müdürü

Fen Bilimleri Enstitüsü Müdürü

T.Ü.FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ YÜKSEK LİSANS PROGRAMI

DOĞRULUK BEYANI

Trakya Üniversitesi Fen Bilimleri Enstitüsü, tez yazım kurallarına uygun olarak hazırladığım bu tez çalışmasında, tüm verilerin bilimsel ve akademik kurallar çerçevesinde elde edildiğini, kullanılan verilerde tahrifat yapılmadığını, tezin akademik ve etik kurallara uygun olarak yazıldığını, kullanılan tüm literatür bilgilerinin bilimsel normlara uygun bir şekilde kaynak gösterilerek ilgili tezde yer aldığını ve bu tezin tamamı ya da herhangi bir bölümünün daha önceden Trakya Üniversitesi ya da farklı bir üniversitede tez çalışması olarak sunulmadığını beyan ederim.

09 / 07 / 2019

Asım ERKUT

İmza



Yüksek Lisans Tezi

İnsan Aktivitelerinin Derin Öğrenme Tekniği İle Tespit Edilmesi

T.Ü. Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

ÖZET

Yapılan bu tez çalışmasında video ya da kamera görüntülerinden insanları tespit ederek, insanların aktivitelerini(ayakta, koşuyor, zıplıyor, oturuyor ve yatıyor) derin öğrenme tekniklerini kullanarak tahmin edilmesi amaçlanmaktadır.

Çalışmada veri seti olarak 10000 den fazla farklı internet kaynağından elde edilen resim ve video görüntüleri kullanılmıştır. Çalışmada faklı yazılım ve donanım teknolojileri kullanılmıştır. Verileri düzenleme ve etiketlemede java dili, eğitim ve model oluşturmada DIGITS, framework olarak caffe, testlerde ise Python dili kullanılmıştır.

Sonuç olarak 8458 adet resim kullanılarak yapılan çalışmada farklı insan aktiviteleri derin öğrenme teknikleri kullanılarak ortalama %97 gibi büyük bir doğrulukla sınıflandırılmıştır. Tek bir resim ile yapılan DIGIT testinde %90.88, 1341 resim ile yapılan Python testinde ortalama % 97.4, video testlerinde ise ortalama 4 fps ve %92 doğruluk ile insan aktivitesi sınıflandırılmıştır.

Yıl : 2019

Sayfa Sayısı : 49

Anahtar Kelimeler : Derin Öğrenme, Bilgisayar Yazılımı, Görüntü İşleme.

Master's Thesis

Determination of Human Activities with Deep Learning Technique.

Trakya University Institute of Natural Sciences

Computer Engineering

ABSTRACT

In this thesis, it is aimed to detect people from video or camera images and to predict people's activities (standing, running, jumping, sitting and lying) by using deep learning techniques.

In the study, pictures and video images obtained from more than 10000 different internet sources were used as data set. Different software and hardware technologies were used in the study. Java language was used for data editing and labelling, DIGITS was used for training and modelling, caffe was used as framework and Python language was used for tests.

As a result, different human activities were classified with an average accuracy of 97% by using deep learning techniques in the study, which was conducted using 8458 images. In DIGIT test with a single image, human activity was classified as 90.88%, Python test with 1341 images, and 97.4% on average, 4 fps and 92% accuracy on video tests.

Year : 2019

Number of Pages : 49

Keywords : Deep Learning, Computer Software, Image Processing

TEŞEKKÜR

Tez çalışmam sırasında bilgi ve tecrübeleri ile destek olan tez danışmanım sayın Doç. Dr. İlhan Umut'a, sonsuz teşekkürlerimi ve saygılarımı sunarım. Ayrıca gösterdikleri sabır ve destek için aileme sonsuz sevgilerimle

İÇİNDEKİLER

ÖZET iv

ABSTRACT v

TEŞEKKÜR vi

İÇİNDEKİLER vii

SİMGELER VE KISALTMALAR ix

ŞEKİLLER LİSTESİ x

ÇİZELGELER LİSTESİ xii

1 GİRİŞ 1

1.1 Problem Tanımı 1

1.2 Tezin Amacı 1

1.3 Tez Organizasyonu 1

1.4 Tezin Katkıları 2

1.5 İlişkili Çalışmalar 2

2 DERİN ÖĞRENME 4

2.1 CNN (Convolutional Neural Networks) 6

2.1.1 Convolution Katmanı 6

2.1.2 Pooling Katmanı 7

2.1.3 Aktivasyon Katmanı 7

2.1.4 Fully Connected Layer 8

2.2 RNN: Recurrent Neural Network. 9

2.3 R-CNN: Regional (Bölgesel) Convolutional Neural Network 11

2.3.1 Fast-R-CNN 12

2.3.2 Faster-R-CNN 13

2.4 SSD: Single Shot MultiBox Detector 14
3 MATERYAL METOD 17
3.1 Çalışmada kullanılan yazılım teknolojileri ve donanım 17
3.1.1 Java 17
3.1.2 Digits 17
3.1.3 Python 18
3.1.4 Donanım ve yazılım versiyonları 18
3.2 Veri Seti 18
3.3 Model oluşturma 23
3.4 Model eğitim sonuçları 26
3.5 Test 29
3.5.1 DIGIT testi 29
3.5.2 Python Test1 31
3.5.3 Python Test2 31
4 SONUÇLAR 33
KAYNAKÇA 35
ÖZGEÇMİŞ 37
EKLER 38

SİMGELER VE KISALTMALAR

ADAM	Adaptive Moment Estimation
CNN	Convolutional Neural Network (Evrişimsel Sinir ağları)
CPU	Central Processing Unit (Merkezi İşlemci)
CUDA	Compute Unified Device Architecture
DIGITS	The NVIDIA Deep Learning GPU Training System
FC	Fully Connected Layer (Tam Bağlantı Katmanı)
GPS	Global Positioning System (Küresel Konumlama Sistemi)
GPU	Graphics processing unit (Grafik İşlemci)
HDF	Hierarchical data format
HMM	Hidden Markov Model
ILSVRC	Imagenet Large Scale Recognition Challenge
JVM	Java Virtual Machine (Java Sanal Makinesi)
LMBD	Lightning Memory-Mapped Database
LSTM	Long short-term memory (Uzun-Kısa Süreli Hafıza)
NMC	Nearest Mean Classsifier (En Yakın Ortalama Sınıflandırıcı)
R-CNN	Regional Convolutional Neural Network (Bölgesel YSA)
ReLU	Rectified Linear Unit (Doğrultulmuş Doğrusal Birim)
RNN	Recurrent Neural Network (Tekrarlayan Sinir Ağı)
SGD	Stochastic Gradient Descent (Olasılıksal dereceli azaltma)
SSD	Single Shot MultiBox Detector
SVM	Support Vector Machine
Tanh	Hyperbolic Tangent (Tanjant hiperbolik)
YSA	Yapay Sinir Ağları

ŞEKİLLER LİSTESİ

Şekil 2.1 Filtreleme işlemi 6

Şekil 2.2 Örnek bir pooling gösterimi 7

Şekil 2.3 Sigmoid, Tanh ve ReLU fonksiyonlarının grafiksel gösterimi 8

Şekil 2.4 DropOut gösterimi 9

Şekil 2.5 RNN yapısı 9

Şekil 2.6 LSTM 10

Şekil 2.7 Örnek bir R-CNN yapısı 11

Şekil 2.8 Örnek bir Fast R-CNN yapısı 12

Şekil 2.9 Selective seach yapılması için işlenmiş resim örneği 13

Şekil 2.10 Örnek bir Faster R-CNN yapısı 14

Şekil 2.11 SSD 15

Şekil 2.12 SSD Mimari yapısı 15

Şekil 3.1 Gerdirme(Squash) ve dolgu(Fill) veri setlerinden örnek bir görüntü 19

Şekil 3.2 Resim düzenleme ve etiketleme uygulaması. 20

Şekil 3.3 Çalışmada kullanılan veri miktarı ve sınıf dağılımları 21

Şekil 3.4 DIGIT yeni bir dataset oluşturma ekranı 22

Şekil 3.5 DIGITS dataset listesi 23

Şekil 3.6 DIGIT yeni bir model oluşturma ekranı 24

Şekil 3.7 Ayrıntılı öğrenme katsayısı ayarı 25

Şekil 3.8 GoogLeNet Convolutional Deep Neural Network 26

Şekil 3.9 Oluşturulan modellerin isimleri ve zaman bilgileri 26

Şekil 3.10 Modellerin doğruluk yüzdeleri 27

Şekil 3.11 En başarılı model(squash_full_20) ile en kötü modelin (fill_normal_50) eğitim değerlerinin epoklara göre değişimi. 28

Şekil 3.12 Karmaşıklık matrisi ve ölçüt formülleri..... 29

Şekil 3.13 DIGIT testi sonuçları 30

Şekil 3.14 Python Test 2 görüntüsü 32

ÇİZELGELER LİSTESİ

Çizelge 3-1 Farklı veri setlerin veri sayıları 23

Çizelge 3-2 Tüm modellerin 5. ve 30. epoklarının loss ve accuracy yüzdeleri 27

Çizelge 3-3 Python Test 1 in karmaşıklık matrisi 31

Çizelge 3-4 Python Test 1 in sonuçları 31

BÖLÜM 1

1 GİRİŞ

1.1 Problem Tanımı

Bilgisayarlı görü göz ile yapılan algılamanın bilgisayara yaptırılması amacını taşır. Resimde yer alan nesnenin ne olduğu ya da birçok nesne olması durumunda koordinatlarının neler olduğu gibi tespitlerde bulunabilir. Görüntü işlemenin bir üst kategorisidir. Çeşitli alanlarda kullanılmaktadır. Yüz tanıma, plaka tanıma gibi günlük hayatta karşılaştığımız birçok uygulaması mevcuttur. Görüntü işlemede sıkıştırma, iyileştirme, kırpa, yakınlaştırma gibi işlemler yapılırken burada verinin ne ve nerede olduğuna dair cevap aranmaktadır. İnsanların yapabileceği görme ile ilgili görevleri otomatikleştirmeyi hedefler (Sonka, Hlavac, & Boyle, 2008).

Günümüzde bir resimdeki insanların makine tarafından tespiti mümkündür. Güvenlik kontrolü amacıyla toplulukta yer alan insanların ayakta durma, koşma, oturma, uzanma, yürüme, zıplama gibi yapılan aktivitelerin de tahmin edilmesi önemli bir problemdir. Çok sayıda insanın olduğu görüntülerde izlemenin insan tarafından yapılması uzun süreli ve zor olacaktır. Çalışmada bu problemin derin öğrenme yöntemleri kullanılarak hızlı şekilde çözümlenmesi hedeflenmektedir.

1.2 Tezin Amacı

Yapılan bu tez çalışmasında video ya da kamera görüntülerinden insanları tespit ederek, insanların aktivitelerini(ayakta, koşuyor, zıplıyor, oturuyor ve yatıyor) derin öğrenme tekniklerini kullanarak tahmin etmeyi amaçlamaktayız.

1.3 Tez Organizasyonu

Birinci bölümde konuya genel bir giriş, ikinci bölümde ise derin öğrenme yönteminden bahsedilmiştir. Bölüm üçte kurulan derin öğrenme mimarisi ve uygulaması son bölümde de çalışma sonuçları verilmiştir.

1.4 Tezin Katkıları

İnsan aktivitelerinin tanımlanmasında sensör tabanlı algılayıcılar oldukça fazla kullanılmaktadır. Örneğin mobil cihazlarda bulunan GPS, pusula, ivmeölçer gibi algılayıcılar ile yapılan aktivite tespit edilebilmektedir. Ancak böyle algılayıcıların olmadığı durumlarda resim ve video üzerinden de bezer verilere sahip olmak isteyebiliriz. Tezde sadece görseller kullanılarak bu hedefe ulaşılmaya çalışılmıştır. Çalışma özellikle psikiyatri hastalarının otomatik aktivite takibi için geliştirilmiş olsa da daha farklı araştırmalarda kullanılabilir; Normal yetişkin insanların otomatik takibi ya da ufak çocukların gözlenmesi için de kullanılabilir. Psikolojik rahatsızlığı olan kişiler üzerinde herhangi bir sensör barındırmak istemeyebilir. Bu durumlarda temassız olan bu yöntemle kişinin aktiviteleri etkili olarak takip edilebilir.

1.5 İlişkili Çalışmalar

İnsan pozu tahmini, bilgisayar vizyon topluluğunda da popüler bir konudur. İnsan pozlarının sonuçlarına dayanarak, insan etkinliği etkin bir şekilde tanınabilir. Huo ve arkadaşları ilk önce gövde tespiti ve takibi için 2D model, eller takibi için ten rengi modeli kullanılmıştır (Huo, Hendriks, Paclik, & Oomes, 2009). 3D rekonstrüksiyon, senkronize çoklu kameralardan çoklu görünüm ile yapılır. 2B ve 3B koordinatlar daha sonra normalleştirilmiş bir özellik alanına dönüştürülür ve anahtar pozları tanımak için en yakın ortalama sınıflandırıcı (NMC) ile sınıflandırılır. Bununla birlikte, önceden tanımlanmış tüm kilit pozları biriktirmek çok büyük bir görev olabilir, çünkü insan pozları çok büyük farklılıklar gösterir. Lee ve Nevatia veri odaklı Markov zinciri Monte Carlo (DD-MCMC) kullanarak ön plan bloğu takibi, 2D eklemler ve vücut parçaları takibi ve 3D poz tahminini içeren üç aşamalı bir yöntem önermektedir (Lee & Nevatia, 2009). Toplantı odası sahneleri 3B insan pozları ile analiz edilir. Bununla birlikte, hesaplama maliyeti çok yüksektir. Ke ve arkadaşları (Ke, Hwang, Lan, & Wang, 2011), çok daha düşük zaman karmaşıklığı ile 3B poz tahmini için başka bir yöntem önermektedir. Şekil, renk ve zaman sürekliliği bilgisi ortak olarak 2B gövde parçalarını izlemek için kabul edilir ve 3B poz

tahmini, sentez analiz metodolojisine dayanan yokuş aşağı tek yönlü algoritma ile elde edilir. Thuc ve arkadaşları, tahmini 3B pozları bir özellik alanına dönüştürür ve insan hareketleri HMM'ler kullanılarak tanınabilir (Hoang, Ke, Hwang, Yoo, & Choi, 2012).

BÖLÜM 2

2 DERİN ÖĞRENME

Derin öğrenme konusunda geçmeden önce tarihçesinden bahsetmek gerekirse ilk olarak Alan Turing bilgisayarların zeki olup olmadıklarını test edebilmek için Turing Test'ini oluşturdu. Bilgisayar testi başarıyla geçebilmek için kullanıcısına insan olduğunu inandırtmak zorundaydı (Turing, 1950).

1952- Arthur Samuel makine öğrenme tekniğini kullanan ilk uygulama olan dama oyununu yazdı. Uygulama IBM'e ait başka bir oyunu oynayarak onun kazanma yöntemlerini öğrendi ve bu yöntemleri kendisi de kullanmaya başladı.

1958-Frank Rosenblatt perceptron'u geliştirdi. Bu yapay sinir ağlarının en basit formudur, beynindeki nöronların çalışma şeklini taklit etmeye çalışır.

1967- The Nearest Neighbor, En yakın komşu algoritması yazıldı. Yazılımla basit kalıpların öğrenilerek satış temsilcilerine harita üzerinde tüm şehirleri en etkin ve kısa şekilde gezebilme imkanı sağladı (Altman, 1992).

1981- Gerald Deijong Açıklama tabanlı öğrenme (Explanation Based Learning) konseptini geliştirdi. Yöntem eğitim verileri analiz ederek genelleme yapar ve önemsiz verileriler dışlar, alan teorisi tabanlı bir makine öğrenim yöntemidir.

1986 - David Rumelhart, Geoffrey Hinton ve Ronald J. Williams geri beslemenin, sinir ağlarında bulunan gizli katmanda ki verilerin temsillerini üretebildiğini deneysel olarak gösterdi (David E. Rumelhart, 1987).

2006- Geoffrey Hinton, Simon Osindero ve Yee-Whye Teh "deep learning" derin öğrenme kavramını geliştirdi. Burada yazılımın veriyi tanıyıp sınıflandırabilmesi için gereken algoritmalarının açıklamasını yaptı (Hinton G. E., 2007; Hinton, Osindero, & Teh, 2006).

"Makine öğrenmesi, yazılım kullanılarak doğruluğu yüksek tahmin yapılabilmesini sağlayan yapay zeka" ilk olarak 1959 da Arthur L.Samuel tarafından

isimlendirilmiştir (Samuel, 1959). Geliştirdiği dama oyunu kendi kendine öğrenen ilk uygulama olmuştur. Uygulama her yolu aramak yerine bulunduğu pozisyona göre skorlama ve oyuncu her iki taraf için kazanma şansını hesaplama, önceki durumlar hatırlama gibi özellikleri vardı.

Makine öğrenimi geri besleme olup olmadığı durumuna göre denetimli ve denetimsiz olmak üzere 2 kategoriye ayrılır.

Denetimli (Gözetimli) öğrenmede öğretici tarafından eğitim veri seti ile örnek girdi ve çıktılar sunulur, bu bilgilerin kullanıldığı bir makine öğrenimi yöntemdir (Stuart J. Russell, 2010). Burada örnek veriler üzerinden bir çıkarımda bulunulur. Öğrenme sonunda örnekler içerisinde bulunmayan veriler için de en doğru çıktı üretmesi beklenir. Öğrenmenin sağlıklı olması için verilerin çeşitliliğinin ve sayısının fazla olması önemlidir.

Denetimsiz (Gözetimsiz) öğrenmede ise etiketlenmemiş verilerden çıkarım yapılması gerekir. Burada geri besleme yapılmadığından doğruluğun değerlendirilmesi yapılamamaktadır. Yöntemin merkezinde kümeleme ve istatistiksel yoğunluk tahmini bulunmaktadır (Jordan & Bishop, 2004). Veri madenciliği ve istatistik hesaplamaları da kullanılır (Mannila, 1996). Veri analizi yaparak denetimsiz öğrenme gerçekleştirir.

Derin öğrenme algoritmaları, yapay sinir ağlarından esinlenilerek geliştirilmiş makine öğrenmesinin bir alt alanıdır. Konuya özgü algoritmalar yerine denetimli, yarı denetimli ya da denetimsiz şekilde çalışmaktadırlar. Başarılı sonuç almak için çok sayıda veriye ihtiyaç duyar (Bengio, LeCun, & Hinton, 2015)

Derin öğrenme tanımı 2000 yılında Igor Aizenberg ve arkadaşları tarafından tanıtılmıştır (I. N. Aizenberg, 2000). Mimari, birçok gizli katmandan oluşmaktadır. Burada, evrişim (convolution) algoritması verilerin sınıflara veya kümelere ayrılmasında Yapay Sinir Ağları ile birlikte kullanılır.

Öğrenme kalitesinin yüksek olması için gerçek örneklerin olabildiğince çok olması gerekmektedir. Ancak bu yapılamadığı zamanlarda ya da kalitenin daha da artırılması istendiğinde yapay olarak eğitim örneklerini çoğaltmak için kullanılır.

Görüntünün simetrisinin alınması, gürültü eklenmesi, boyut değiştirilmesi, kırpma, renk titretme, siyah beyaza çevirme, afin dönüşümleri gibi yöntemler

uygulanmaktadır (Bjerrum, 2017). Derin öğrenmede kullanılan tekniklerde en çok kullanılanları aşağıda anlatılmaktadır.

2.1 CNN (Convolutional Neural Networks)

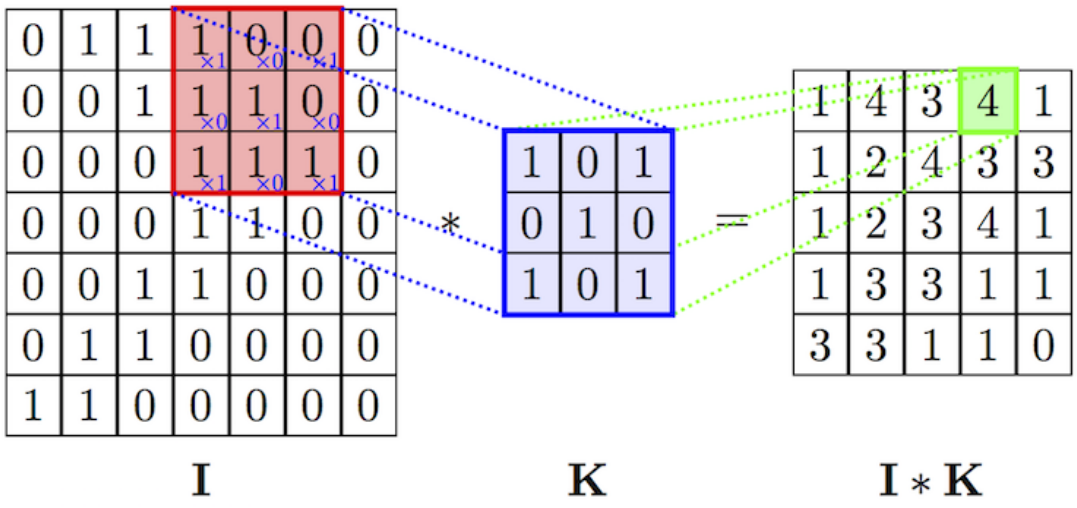
Evrişimsel Sinir ağları sadece yapay sinir ağları ile sınırlı değildir yapay sinir ağları kullanımının önceki bölümünde birçok katmandan oluşan evrişim (convolve) bölümüne sahiptir. Her katmanda filtre, doğrusal olmaya katman ve öznitelik havuzlama katmanı bulunur. Bu ağlar eğitilebilir çok katmanlı bir yapıdadır (Y. LeCun, 1990). Çalışmada da bu yöntem SSD(Single Shot MultiBox Detector) ile birlikte kullanılmıştır

2.1.1 Convolution Katmanı

Bu katmanda filtreler bulunur, filtre matris üzerinde kaydırılarak bire bir çarpılır ve sonuçlar toplanarak yeni matriste yerine yazılır. Burada her bir filtre bir nörona karşılık gelmektedir. Filtre içerisindeki değerler ise ağırlıklardır. Kenar hücrelerdeki veri kaybını önlemek için dış çeperine istenirse boşluklardan oluşan pad ler de eklenebilir. Filtre kaydırılması sırasındaki kaydırma hücre sayısı Stride olarak ifade edilir. Çıktı matrix boyutu için aşağıdaki hesaplama kullanılır

ConvoldedWith= ((MatrixWidth-FilterWidth+2*Pad)/StrideW) +1

ConvoldedHeight= ((MatrixHeight-FilterHeight+2*Pad)/StrideH) +1



Şekil 2.1 Filtreleme işlemi

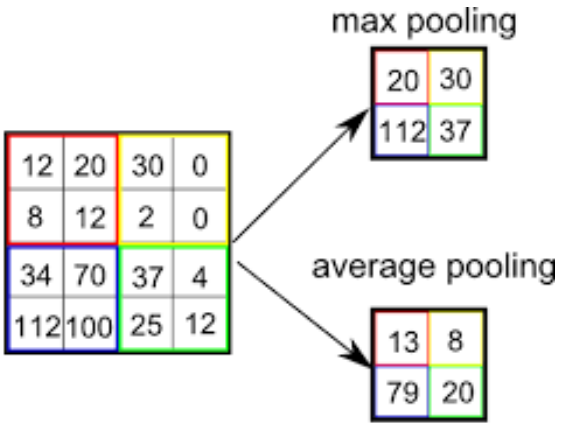
Şekil 2.1’deki örnekte (Pad 1 yapılarak düzenlenecek);

Örnekte Girdi: 7, Filtre: 3, Pad: 0, Stride: 1 dir, Formül uygulandığında ((7-3+2*0)/1)+1 = 5 pixellik çıktı sonucuna ulaşılır.

2.1.2 Pooling Katmanı

Ana amacı özellikleri kaybedilmeden boyutun küçültülmesidir. Temel özellikleri;

- Düşük hesaplama,
- Çeviri Değişmezliği,
- Dönüşüm Değişmezliği,
- Ölçeklendirme değişmezliği,



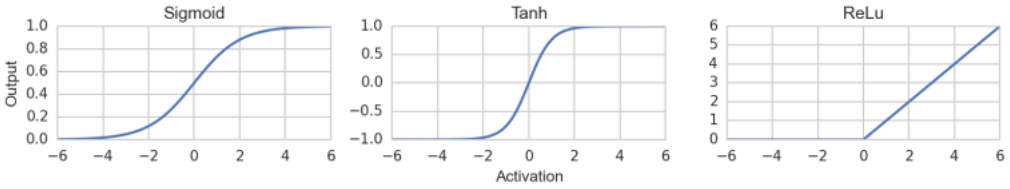
Şekil 2.2 Örnek bir pooling gösterimi

Örnekleme yapmak ve matrix boyutunu küçültmek için kullanılır. Girdi olarak alınan matrix değerleri üzerinde pooling çerçevesi gezdirilir ve seçilen metoda göre elde edilen değerler yeni matrix değerleri olarak yazılır

2.1.3 Aktivasyon Katmanı

Gizli katmanda matrix çarpımı yapılarak nöronların ağırlıkları hesaplandıktan sonra çıktı doğrusal olmayan (non-linear) bir değere dönüştürülür. Derin öğrenme yöntemleri de doğrusal olmayan problemlerin çözümünde kullanıldığı için aktivasyon fonksiyon kullanımı uygun olmaktadır.

Burada çeşitli fonksiyonlar bulunmaktadır(Sigmoid, Tanh ve ReLU (Rectified Linear Unit))



Şekil 2.3 Sigmoid, Tanh ve ReLU fonksiyonlarının grafiksel gösterimi

ReLU:İleri beslemeli (Feed Forward) beslemeli ağlarda genellikle ReLU kullanılmaktadır. Çünkü bu fonksiyonla öğrenme hızı sigmoid ve tanh ‘e göre daha hızlıdır. Gradyanın yok olma ihtimalini yok eder. Ancak bir dezavantaj olarak aktivasyonun patlamasına neden olabilir.

Sigmoid (Lojistik, Soft Step): Sigmoid fonksiyonunda 0 ile +1 arasında değerler elde edilir, Bu sayede yüksek girişler nedeniyle aktivasyon patlaması yaşanmaz.

$$s(x) = \frac{1}{1 + e^{-x}}$$

Tanh : bu fonksiyonda -1 ile +1 arasında değerler elde edilir

$$\tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$

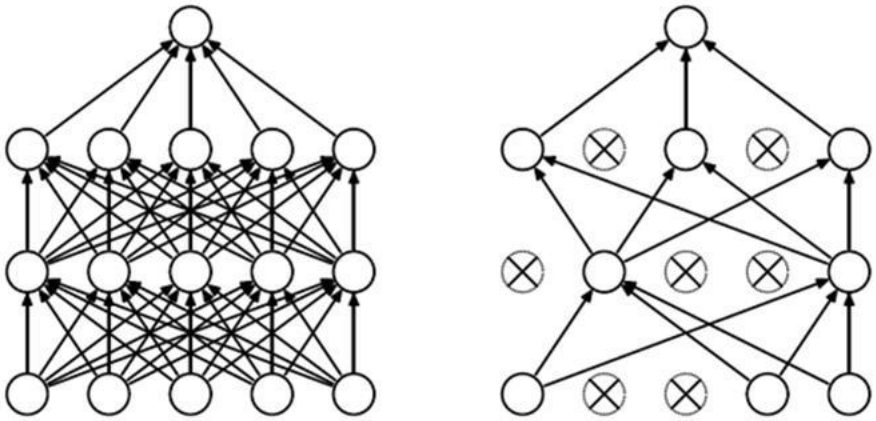
2.1.4 Fully Connected Layer

Konvolüsyon ve pooling katmanlarından sonra son katman olarak kullanılır. Bu katman tüm giriş ve çıkışları birbirine bağlıdır ve farklı ağırlıklara sahiptir. Farklı mimarilerde bu katman farklı sayılarda olabilir. Multi Layer Perceptron mimarisindedir. Birden fazla bu katmandan olması durumunda önceki katmanın çıkış sayısı ile sonraki katmanın giriş sayısı eşit olmalıdır.

Burada katmanda önceki katmanlarda daha çok tercih edilen ReLU aktivasyon fonksiyonu yerine daha çok tanh ya da sigmoid fonksiyonları tercih edilir.

Dropout- Seğreltme Yöntemi bu katmanda kullanılır, seçilen düğümlerdeki bilgilerin yok edilmesi ile sistemin ezberleme (overfitting) yapmasının önüne geçilir ve

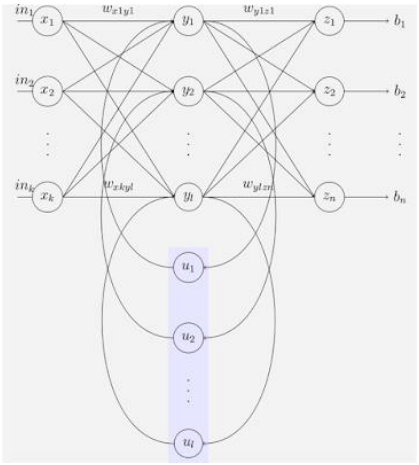
eğitim performansı artırılmış olur. Genellikle belli eşik değerlerin altındaki düğümler arasından unutma işlemi yapılır ancak eşik değere bakılmaksızın rasgele unutma yöntemi de kullanılmaktadır (Nitish Srivastava, 2014).



Şekil 2.4 DropOut gösterimi

2.2 RNN: Recurrent Neural Network.

Bir çeşit yapay sinir ağıdır. Geleneksel sinir ağlarında tüm girdilerin birbirinden ayrı olduğu varsayılır, ancak örneğin konuşma sırasında bir kelime içerisindeki harflerin sonraki harflere etkisi bulunmaktadır bu durumda YSA'daki saklı katmanında üretilen çıkış sinyali bir sonraki işlemde diğer giriş bilgilerine ek yeni bir giriş olarak kullanılır dolayısıyla bir hafıza meydana gelir. Bu sayede sıralı bir durum oluşur ve sadece giriş değil bir önceki durumda yeni çıktıya etki eder.

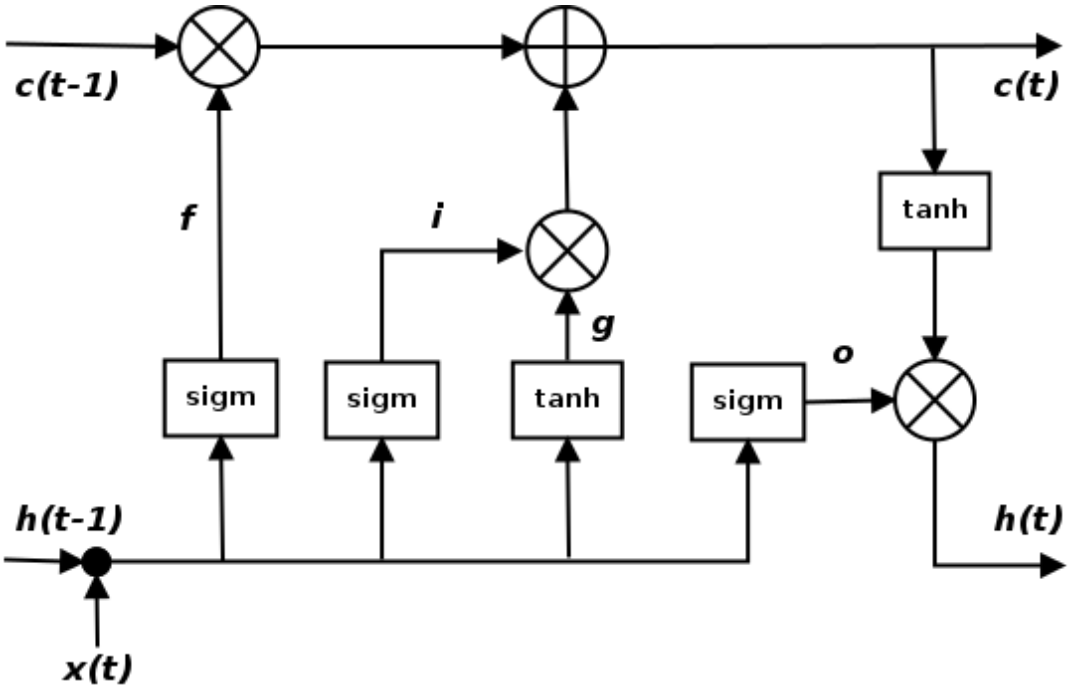


Şekil 2.5 RNN yapısı

RNN konuşma tanıma ve zaman serileri gibi problemlerde çok kullanışlıdır. İçerisinde hafıza barındırdığı için Turing makinesi simülasyonunda kullanılabilir (Sontag, 1991).

YSA'larda eğitim; hesaplanan çıktı ile gerçek çıktı arasındaki fark ile oluşan eğim (gradient) sonucunda ağırlık değerlerinin yeniden hesaplanması ile yapılır. Bu tür ağlarda eğim patlaması ya da sıfırlanması durumu oluşabilir eğer eğim sıfırlanırsa eğitim devam edemez, RNN de de önceki çıkışın yeni girişlere eklenmesi sırasında bu problem oluşabilir.

Bu sorunu çözebilmek için LSTM (Long short-term memory) (Uzun kısa dönem hafıza) yöntemi geliştirilmiştir. LSTM de önceki işlemde oluşan çıktı hafızaya alınır ve bu hafızaya hücre adı verilir. Burada hücrenin saklanacağına ya da silineceğine karar verilir, silinen hücreler bir sonraki işlemde kullanılmazlar. Hochreiter ve Schmidhubert tarafından ilk olarak geliştirilmiştir (S. Hochreiter, 1997).



Şekil 2.6 LSTM

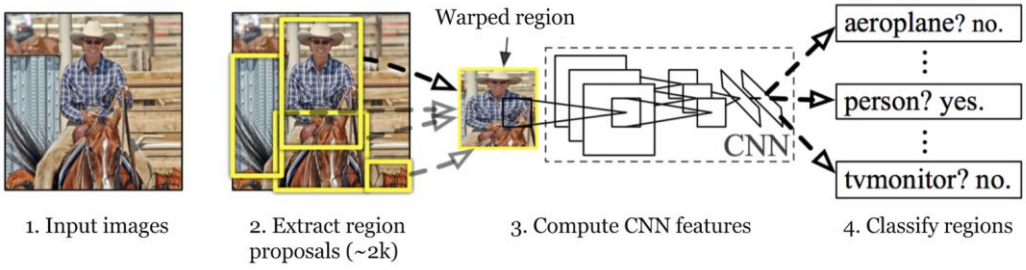
2.3 R-CNN: Regional (Bölgesel) Convolutional Neural Network

Bir resimde ne olduğunun tespiti için CNN kullanmak yeterlidir. Ancak resimde birden fazla obje bulunduğunda bu objelerin koordinatlarının ve ne olduklarının tespitine ihtiyaç bulunmaktadır. Bu durumda resmin her bölgesinin taranması gerekmektedir. 2014 yılında Ross Girshick tarafından geliştirilmiştir (R. Girshick, 2014).

R-CNN (Bölgesel CNN) ile görüntüdeki ana objeler koordinatları ile birlikte tanımlanırlar, R-CNN ler insanın sezgisel olarak yaptığı işlemi seçmeli arama işlemini kullanarak gerçekleştirir.

Seçici arama işleminde farklı koordinatlar seçilir ve bölgeler arasındaki benzerlikler tespit edilir ve herhangi bir nesneye karşılık gelip gelmediği incelenir (J.R. Uijlings, 2013).

Nesne tespit edildiğinde koordinatlar objenin gerçek boyutlarına uyacak şekilde daraltılır. R-CNN içerisinde 3 özellik barındırır bunlar görüntü özellikleri tespiti için CNN, sınıf tahmin eden sınıflandırıcı ve koordinatların netleştirilmesi bölümleridir.

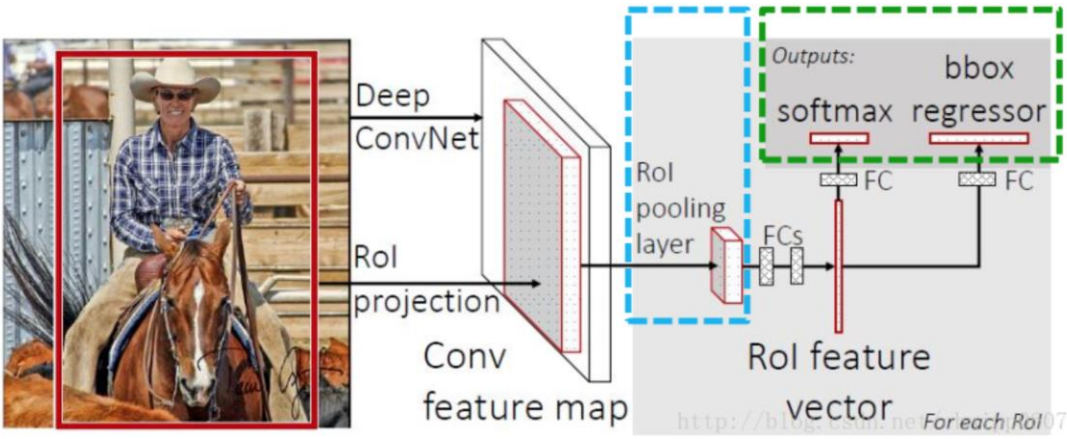


Şekil 2.7 Örnek bir R-CNN yapısı

R-CNN de bölge önerisinde bulunulur buna ilgi bölgesi denir. RoI (Region of Interest) şeklinde adlandırılır. Resim üzerinde farklı boyutlarda 2000 civarı bölge tespit edilir. Bu resimler CNN ağından geçirilmeden önce yeniden ölçeklendirilir. Lineer regresyon ile nesnenin bulunduğu tespit edilir ve resim üzerinde dikdörtgen tespiti yapılır. Resim de neslerin tanıma süresi 30 -60 sn arasında sürmektedir. Gerçek zamanlı nesne tanınması imkansızdır. Son katmanda resim tespiti için SVM (Support Vector Machine) kullanılmaktadır. SVM ile resim grupları arasına sınır çizilir.

2.3.1 Fast-R-CNN

Resmin tüm bölgelerini farklı büyüklükteki kutular ile taramak maliyetli olduğu için Fast R-CNN (Region of Interest Pool)-İlgi havuzu geliştirilmiştir. Burada havuzlama işlemi kullanılarak tek bir geçişte bölgeler tespit edilir. R-CNN deki 3 özelliği tek seferde hesaplar. Sınırlayıcı koordinatları için soft-max katmanına paralel doğrusal regresyon katmanı eklenir (Girshick, 2015).



Şekil 2.8 Örnek bir Fast R-CNN yapısı

R-CNN deki gibi direk bölge önerisinde bulunulmaz. Önce resim CNN den geçirilir. Daha sonra özellik haritası çıkarılarak seçici tarama (Selective Search) ile bölge önerisinde bulunulur. Bu şekilde R-CNN deki gibi her bölgenin ayrı ayrı CNN den geçirilmesi ihtiyacı ortadan kalkmış oluyor. Belirlenen bölgeler yine R-CNN de olduğu gibi yeniden boyutlandırılarak Fully Connected Layere bağlanır. Sınıflandırma işlemi sinir ağı üzerinde yapılmaktadır. R-CNN e göre yaklaşık 10 kat hızlı çalışmaktadır.

Seçici tarama (Selective Search)

Nesne tespiti için bir bölge tespit algoritmasıdır. Renk, doku, boyut ve şekil uyumluluğuna bağlı olarak benzer bölgelerin hiyerarşik gruplandırmasına dayanmaktadır. Grafik tabanlı bir yöntemdir.

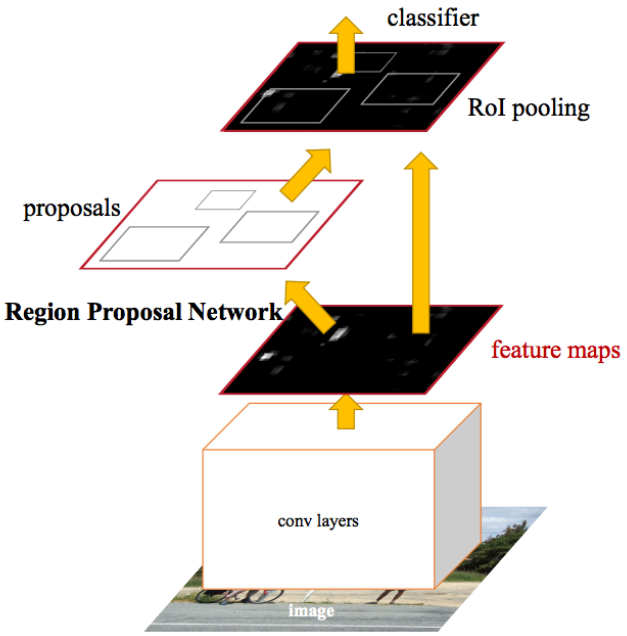


Şekil 2.9 Selective seach yapılması için işlenmiş resim örneği

2.3.2 Faster-R-CNN

Fast R-CNN de obje konumlarını tespit etmek için potansiyel ilgili bölgeleri ve sınırlayıcı kutu üretilir ancak bu bölgelerin hepsinde tanımlaması yapılan bir obje tespit edilemeyebilir. Buda bir darboğaza neden olmaktadır. Bölgesel Teklif Ağı (Region Proposal Network) öncesinde bir CNN eklenir.

Bölgesel teklif ağı Konvolusyonel network tarafından çıkarılan özellik haritası üzerinde kayan bir pencere den geçerek çalışır. Kutuların her birisi için sınırlayıcı kutu ve puanlar elde edilir. Sezgisel olarak pencere boyutları belirlenirken eğitimlerde belirlenen pencere boyutlarına yakın seçimler yapılır daha önce karşılaşılmamış sıradışı en ve boylardaki dikdörtgenler elenir. Belirlenen dikdörtgenlerin ortasına bir çapa eklenilerek en boy oranı ve ölçeğe göre bölgeler tespit edilir (Shaoqing Ren, 2015).

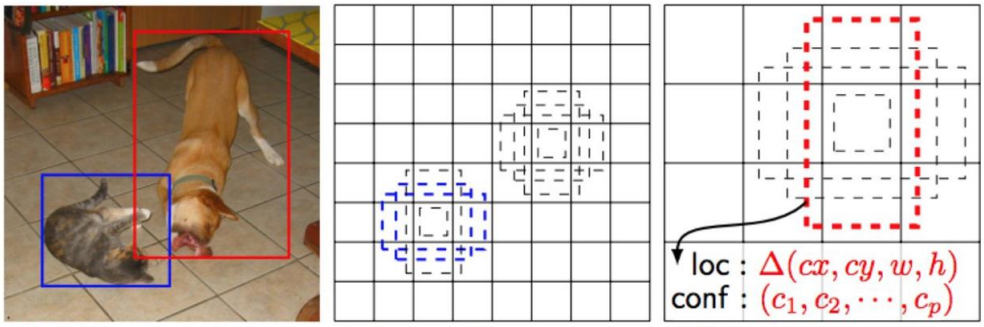


Şekil 2.10 Örnek bir Faster R-CNN yapısı

Fast R-CNN de zaman kaybının büyük bölümü seçici tarama bölge önerisi aşamasında oluşmaktadır. Faster R-CNN de seçici tarama işlemi yerine ayrı bir bölge önerisi ağı kullanılmaktadır. Bölge önerisi ağının 2 görevi bulunmaktadır. Her öneri için orada nesne olup olmadığına ve pencere büyüklüğüne karar verir.

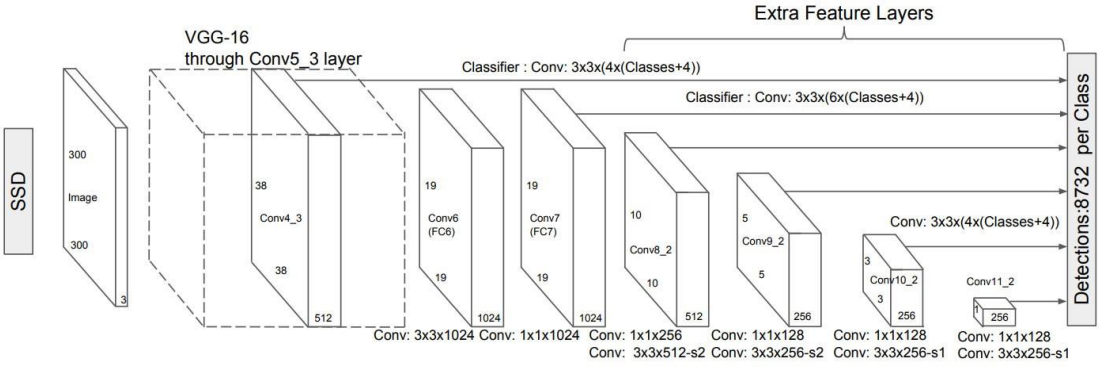
2.4 SSD: Single Shot MultiBox Detector

Bu yöntemde görüntüdeki nesneleri tespit etmek için tek bir derin öğrenme ağı kullanılır. Burada sınırlayıcı kutuların çıkış alanı bir dizi varsayılan kutuya ayırır. Tahmin sırasında ağ her kutudaki var olan nesne için kategorilerine göre puan üretir ve kutu nesne boyutlarına daha iyi uyum sağlayacak şekilde düzenlenir. Ayrıca ağ özellik haritalarını çıkarırken çözünürlüklerinin aynı olması koşulunu zorunlu tutmuyor. Böylece tek ağda, verileri kodlama ve çoğaltma işlemi tamamlanmıştır. Bu da SSD'nin eğitilmesi ve entegrasyonu kolaylaştırır. Faster-R-CNN ile karşılaştırıldığında daha küçük görüntü boyutlarında bile daha iyi doğruluk oranına sahiptir (W. Liu, 2016).



Şekil 2.11 SSD

Şekilde SSD sadece eğitim sırasında her nesne için bir giriş görüntüsü ve zemin doğru kutularına ihtiyaç duyar. Bir konvolüsyonel yöntemle farklı ölçekler içeren bir dizi farklı varsayılan kutular değerlendirilir. Tüm nesne kategorileri için varsayılan kutular farklı ölçeklerde belirlenir. her iki pozisyonda da (8 * 8) (4 * 4), her iki biçimdeki dezavantajları ve doğrulukları tahmin edilir. Eğitim zamanında, ilk önce bu varsayılan ile kedi ve bir köpekle eşleştirildi. Diğer bölgeler negatif olarak işaretlenir.



Şekil 2.12 SSD Mimari yapısı

SSD nin başlangıç katmanında Vgg-16 algoritması kullanılmaktadır. Oxford üniversitesi Görsel Geometri Grubu ILSVRC2014 yarışmasında kullanılmıştır. 13 konvolüsyon ve 3 tam bağlı katmanından oluşur. Maxpool, Fully connected layer, Relu layer, Dropout layer ve Softmax layer katmanları vardır. Toplamda 41 katman bulunur. Girdi görüntü çözünürlüğü 300x300x3 şeklindedir. Son katman ise sınıflandırma

yapılmaktadır. Vgg-16 katmanında (Conv4_3) ki her hücre için (38*38 pixel) 4 nesne tahmini yapılır.

Küçük ölçekli nesnelerde Faster R-CNN e göre performası daha düşük olmasına rağmen farklı ölçeklerle oluşturulan özellik haritaları sayesinde aynı resim üzerinde bulunan farklı boyutlardaki objeleri daha hızlı tespit edebilmektedir. 22 fps de tarama yapabilmektedir.

BÖLÜM 3

3 MATERYAL METOD

3.1 Çalışmada kullanılan yazılım teknolojileri ve donanım

3.1.1 Java

Nesne yönelimli bir programlama dilidir. Derlenen uygulamalar farklı işletim sistemleri üzerine kurulan JVM içerisinde değişiklik yapılmaya ihtiyaç duymadan çalıştırılabilmektedir.

Çalışmamızda videoların resimlere dönüştürülmesi, resimlerin içerisinden eğitilecek hareketin seçilmesi, seçilen hareketlerin etiketlenerek klasörlerde toplanması, resimlerin yatayda çevrilmesi, üzerlerine kirletme uygulanması gibi işlemler hazırlanan java uygulaması ile yapılmıştır.

3.1.2 Digits

NVIDIA tarafından geliştirilen bir uygulama olan DIGITS, Derin Öğrenme GPU Eğitim Sistemi (Deep Learning GPU Training System) dir.

Digits derin öğrenme frameworkleri için ortak bir uygulama çatısıdır, İçerisinde Tensorflow, Torch, Theano, Caffe gibi farklı frameworkler kullanılabilmektedir.

Digits CUDA(Compute Unified Device Architecture) mimarisini de desteklemektedir. NVIDIA tarafından üretilen GPU'nun donanımsal hesaplama gücünü kullanan paralel hesaplama mimarisidir. Birçok işletim sistemi ve yazılım ortamı tarafından desteklenmektedir. CPU'larda veri işlemesi seri olarak CUDA destekli GPU mimarisinde donanımsal olarak paralel işlem yapıldığından yüksek performans sağlanmaktadır.

DIGITS 3 temel bölümden oluşmaktadır: Dataset’lerin hazırlanması, modellerin hazırlanması ve mevcut modellerden yeni modellerin oluşturulması.

3.1.3 Python

Python günümüzde popüler bir script dilidir. Çalıştırma sırasında yorumlandığı için derlenmesine ihtiyaç bulunmamaktadır. Etkileşimli bil dildir, etkileşimli kabuk (interactive shell) üzerinde parça parça çalıştırılabilmektedir. Farklı işletim sistemlerinde çalıştırılabilmektedir.

Python kütüphanelerini yönetmek için Anaconda paket yöneticisi kullanılmıştır. Kütüphanelerinde sürüm yönetimini de üstlenmektedir. Modeli gerçek video ya da kamera görüntüleri ile test etmek ve istatistiksel sonuçları hesaplatmak için Python dili kullanılmıştır. OpenCV (DeepLearning Model kullanımı) ve Numpy (Matematik), ImUtils (Video işleme) kütüphaneleri kullanılmıştır. Python ile modelin confusion matrixi Scikit-Learn (Makine Öğrenme) kütüphanesi ile oluşturulmuştur

3.1.4 Donanım ve yazılım versiyonları

Ekran Kartı: NVidia GeForce GTX 1070 Ti, 2432 Cuda Core, 8 GB Ram

İşlemci: Intel i7 8700K, 3.7 GHz

Bellek: 16 GB Ram

Nvidia Digits (Deep Learning GPU Training System) 6.1.1

Deep Learning FrameWork'ü olarak Caffe 0.15.14

Python Editor: PyCharm, Python 3.7

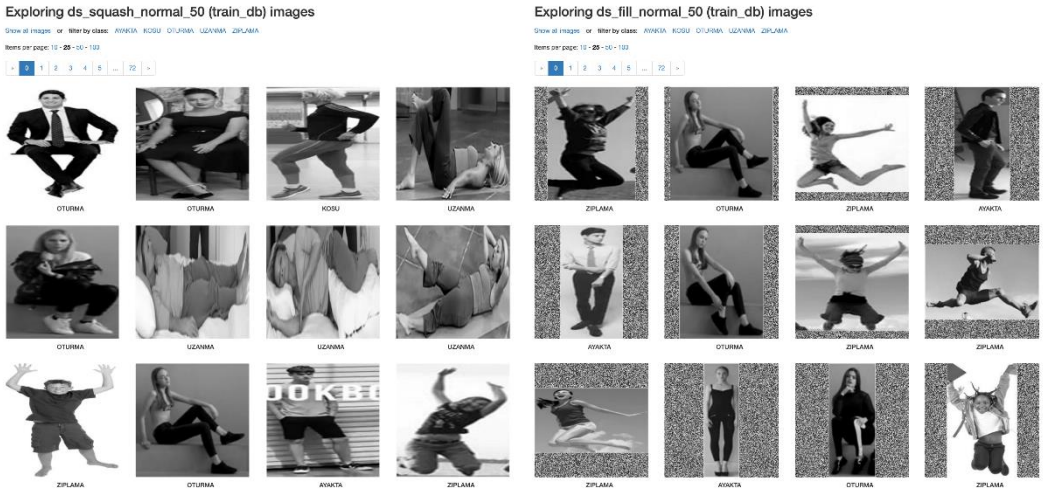
Java Editor: IntelliJ Idea, Java 8

Python Environment: Anaconda 5.3

3.2 Veri Seti

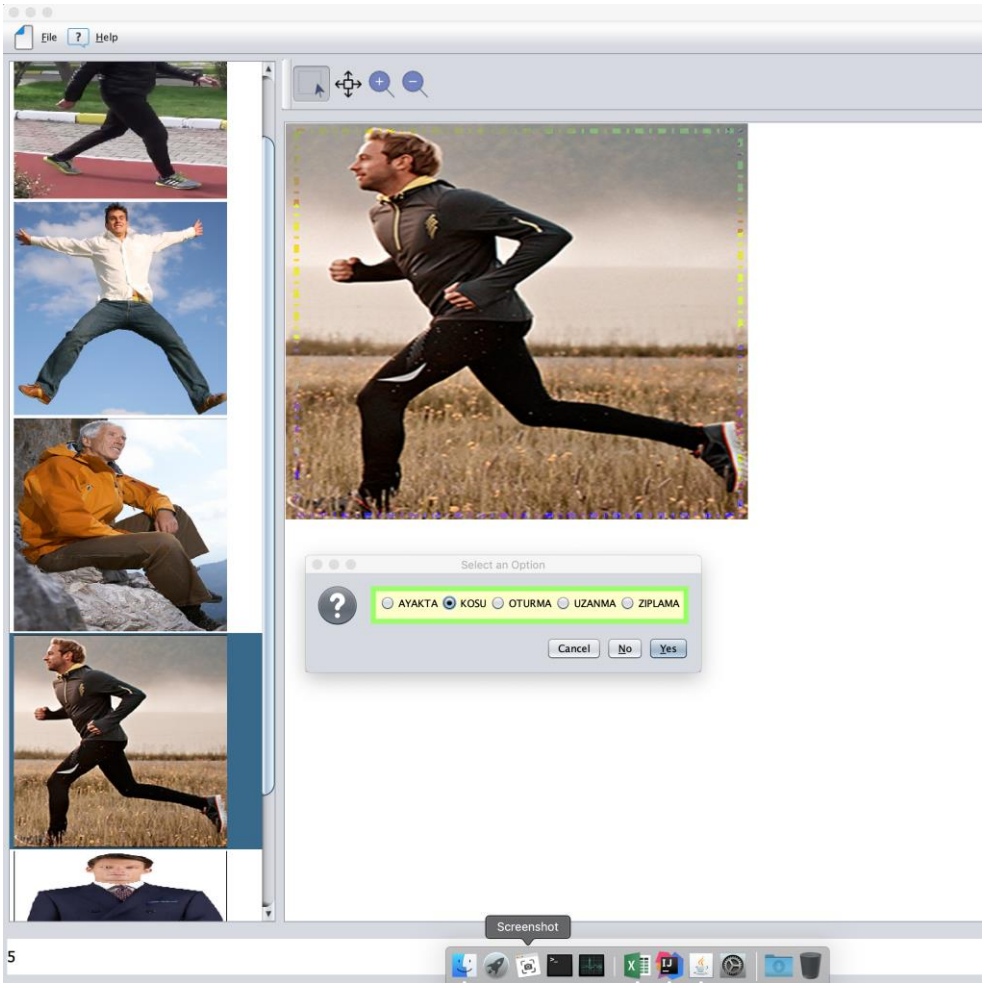
Çalışmada kullanılan veri seti çeşitli aktivitelerdeki insan görüntülerinden elde edilmiştir. Bular ayakta duran (AYAKTA), koşan (KOSU), oturan (OTURMA), zıplayan (ZIPLAMA), yerde uzanan (UZANMA) şeklinde 5 grupta toplanmıştır. Veriler bilinen internet arama motorları olan Google ve Yandex'ten ayrıca Youtube üzerindeki videolardan resimlere dönüştürme şeklinde toplanmıştır. Eğitim için 7117 test için

ise1341 adet resim kullanılmıştır. Veri setinin örnek bir görüntüsü Şekil 3.1'te görülmektedir.



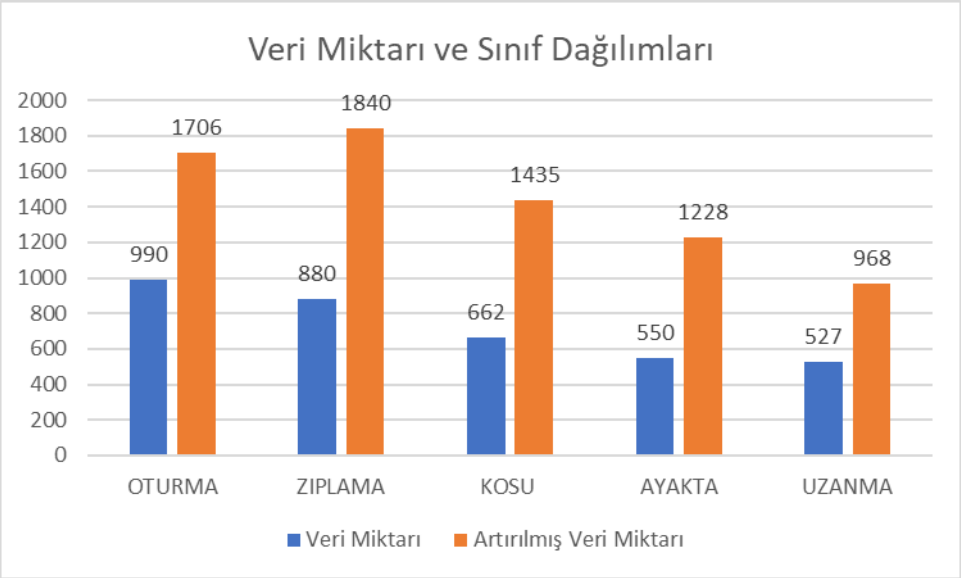
Şekil 3.1 Gerdirme(Squash) ve dolgu(Fill) veri setlerinden örnek bir görüntü

Toplanan görüntüler farklı çözünürlükte ve boyutlarda olması sebebiyle, gerçekleştirilen çalışmaya özel bir uygulama ile resim içerisinde tespit ettiğimiz eğitimde kullanılacak hareketler işaretlenerek kırpılmış ve etiket isimleri ile belirtilen klasörlerde biriktirilmiştir. Uygulamanın görsel bir ara yüzü Şekil 3.2'te gösterilmiştir.



Şekil 3.2 Resim düzenleme ve etiketleme uygulaması.

Seçilen resimler yatay düzlemde çevirme (ayna görüntüsü) yöntemiyle çoğaltılmıştır. Toplanan resimler renkli ya da siyah beyaz karışık toplanmıştır.



Şekil 3.3 Çalışmada kullanılan veri miktarı ve sınıf dağılımları

- Oturma Sandalyede, yerde bağdaş, çeşitlilik fazla olduğundan. Örnek sayısı fazladır
- Zıplama Eller ve kolların çapraz olmasına dikkat edilmiştir. Örnek sayısı fazladır
- Koşu Bacaklar ve kolların çapraz olması
- Ayakta Düz bir duruş ve kollarda çapraz lığın olmaması
- Uzanma Yerde yatış şeklinde, yan ya da düz

DIGITS Dataset modellerin eğitilebilmesi için gereken resim veri tabanıdır. Farklı parametrelerle bu veri tabanı üretilebilir (Şekil 3.4).

İmaj Tipi: Siyah-beyaz ya da renkli (Siyah-beyaz seçildi)

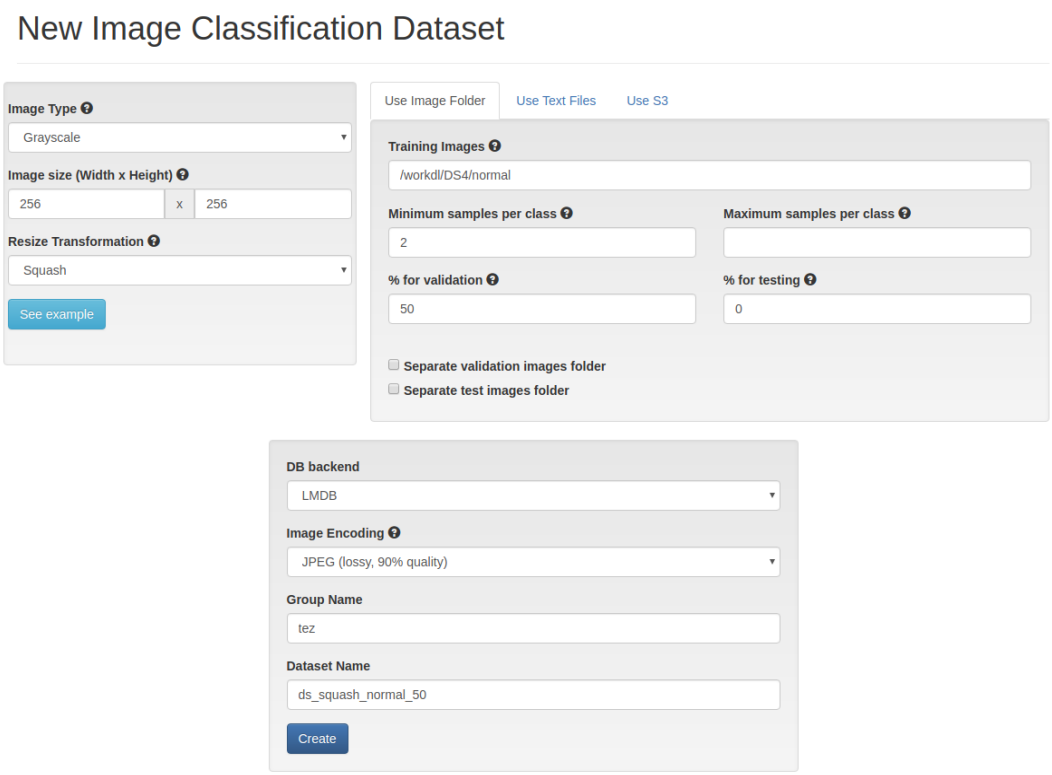
İmaj size: Çözünürlük (256*256 seçildi)

Çözünürlük Dönüşümü: Squash ve Fill dönüşüm yöntemi kullanılmıştır. Squash kaynak resim çözünürlüğü hedef çözünürlüğü ile aynı olacak şekilde yeniden boyutlandırılır, yatayda ya da dikeyde genleşme olabilir. Fill ise kaynak resim hedef çözünürlüğe göre en boy oranı değişmeyecek şekilde yeniden boyutlandırılır. Yanlarda ya da alt üst bölgede boşluk oluşabilir, resmin doğal ölçülerinde bozulma olmaz.

Model üretilirken mevcut resimlerimizden istenen miktardaki resim validasyon için ayrılabilmektedir. Çalışmada %20 ve %50 olacak şekilde iki farklı validasyon oranı seçilmiştir.

Veri yapısı olarak LMBD ya da HDF5 kullanmaktadır. HDF5 yapısında resimler diskte oldukları gibi saklanırlar ve okunduklarında resmin belleğe tamamen alınması gerekir, dolayısıyla yüksek çözünürlüklü resimlerde ekran kartı kapasitesine dikkat etmek gerekir. LBMD yapısında ise parça parça okuma işlemi yapılabilmektedir. Bellek aşımı problemleri yaşamamak için tercih edilmektedir. Çalışmada LMBD seçildi

Resim formatı olarak PNG (Lossless) resim formatında farklı formatlarda okunan resimler PNG resim formatına dönüştürülerek mevcut kaliteleri değiştirilmeden saklanırlar. JPG (Losly) resim formatında ise farklı formatlarda okunan resimler JPG resim formatına dönüştürülerek %90 kalite ile saklanırlar. Çalışmada JPG seçildi.



Şekil 3.4 DIGIT yeni bir dataset oluşturma ekranı

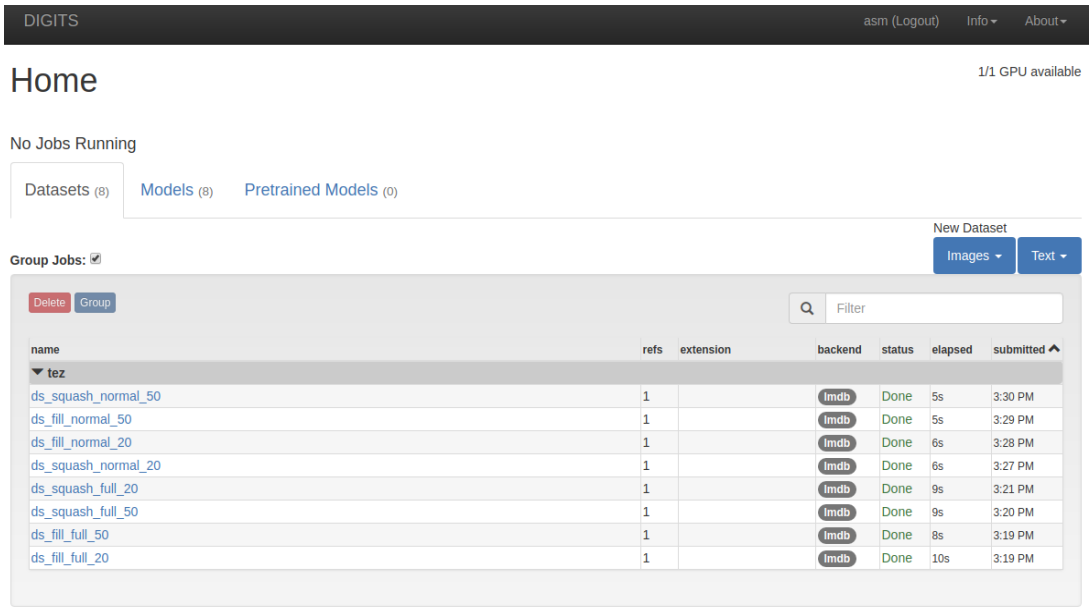
Çalışmada çözünürlük dönüşümü(Squash ve Fill), veri miktarı(normal ve arttırılmış) ve validasyon oranı(Toplam verinin %20 si ve %50 si) değerlerinin

değiştirilmesiyle toplam 8 adet farklı dataset oluşturulmuştur (Çizelge 3-1). Modeller her bir dataset için ayrı ayrı oluşturulup kıyaslanmıştır.

Çizelge 3-1 Farklı veri setlerin veri sayıları

Eğitim Data Setleri	Eğitim	Validasyon
fill_normal_20	2888	721
fill_full_20	5741	1436
squash_normal_20	2888	721
squash_full_20	5741	1436
fill_normal_50	1805	1804
fill_full_50	3589	3588
squash_normal_50	1805	1804
squash_full_50	3589	3588

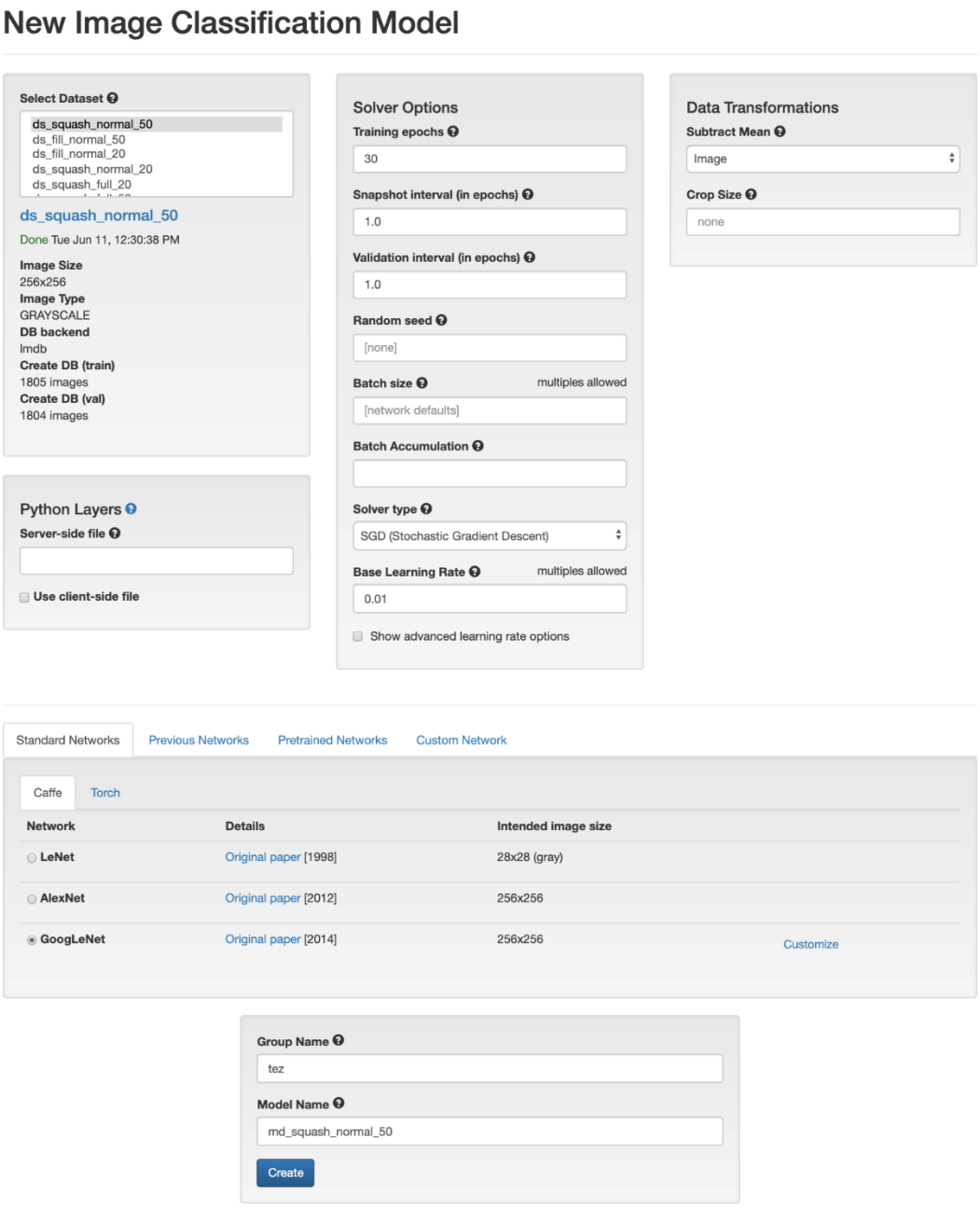
Farklı oluşturulan veri setlerinin DIGITS ekran görüntüsü Şekil 3.5' da görülmektedir.



Şekil 3.5 DIGITS dataset listesi

3.3 Model oluşturma

Modellerin hazırlanmasında önceden oluşturulan veri setleri kullanılmıştır. Dolayısıyla 8 farklı dataset için 8 farklı CNN modeli oluşturulmuştur. Model oluşturma işleminin çeşitli parametreleri bulunmaktadır. Bunlar Şekil 3.6'de görülmektedir.

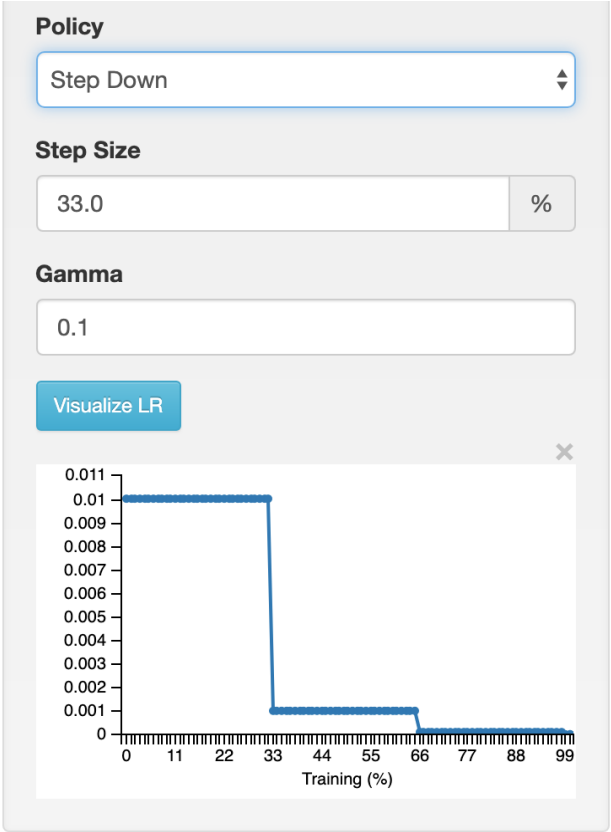


Şekil 3.6 DIGIT yeni bir model oluşturma ekranı

Dataset: Hazırlanan DIGITS veri seti seçilir.

Epoch: Eğitim tur sayısıdır. Model de ilk parça eğitildikten sonra başarımı test edilir, başarım durumuna göre geri yayılım ile (backpropagation) ağırlıklar yeniden hesaplanarak işlem yeniden tekrarlanır. Buradaki eğitim adımlarının her birine epoch denilmektedir. Belli bir eğitim sayısından sonra başarım artış çok azalmaktadır ya da modelin ezberlenmesine yol açılmaktadır. Bu nedenle bu değeri 30 ile sınırlandırdık.

Learning Rate: Öğrenme hızını etkileyen parametredir. Girdi ağırlıkları bu değerle çarpılarak çıktı oluşmaktadır. Değerin büyük olması öğrenmeyi hızlandırmasına rağmen doğru hedefe yaklaşıldığında altında ve üzerinde büyük salınım yaparak hedefe ulaşmakta güçlük çekilir, küçük olması durumunda ise hedefe hiç ulaşamama durumu oluşmaktadır. Yapılan testlerde bu değerin ilk 10 epok için 0.01, sonraki 10 epok için 0.001, son 10 epok için 0.0001 olacak şekilde steplere ayrılarak en iyi başarım sağlanmıştır. Bu seçeneklerin görüntüsü Şekil 3.7' gösterilmektedir.



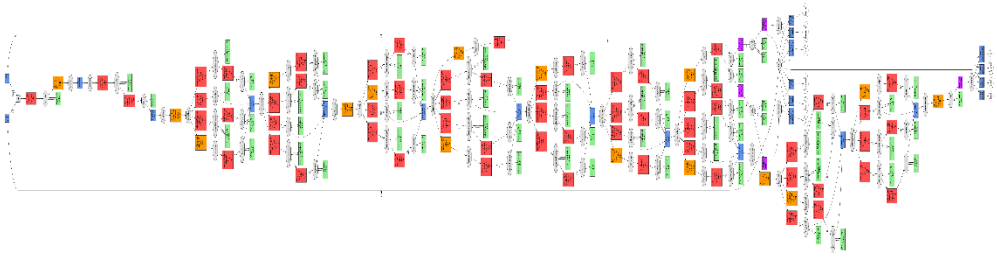
Şekil 3.7 Ayrıntılı öğrenme katsayısı ayarı

Solver Type: Çalışmada SGD seçilmiştir.

SGD: (Stochastic Gradient Descent) Olasılıksal dereceli azaltma, Verimli ve hızlı sonuç alınmaktadır.

ADAM: (Adaptive Moment Estimation) Öğrenme hızı yönü hazıfada saklanarak sonraki öğrenmelerde önceki öğrenme yönünde baskı yaparak farklı alternatif çözümlerin bulunmasında fayda sağlanması hedeflenmiştir.

Çalışmada framework olarak Caffe, network olarak ise GoogLeNet seçilmiştir. Bunun sebebi 2014 yılında yapılan bir yarışmada ILSVRC (Imagenet Large Scale Recognition Challenge) en düşük hata oranına (%6.67) sahip networktür. Ayrıca bir önceki yarışmanın birincisi olan AlexNetten 12 kat daha az parametre kullanmaktadır. Dolayısıyla çok daha hızlı ve doğrudur.



Şekil 3.8 GoogLeNet Convolutional Deep Neural Network

3.4 Model eğitim sonuçları

Bu bölümde farklı modellerle yani farklı veri setleri ile yapılan eğitim sonuçları incelenmektedir. Şekil 3.9 de tüm modellerin eğitim süreleri verilmektedir. En büyük veri setine sahip fill_full_20 ve squash_full_20 eğitimleri 11 dakika sürmüştür.

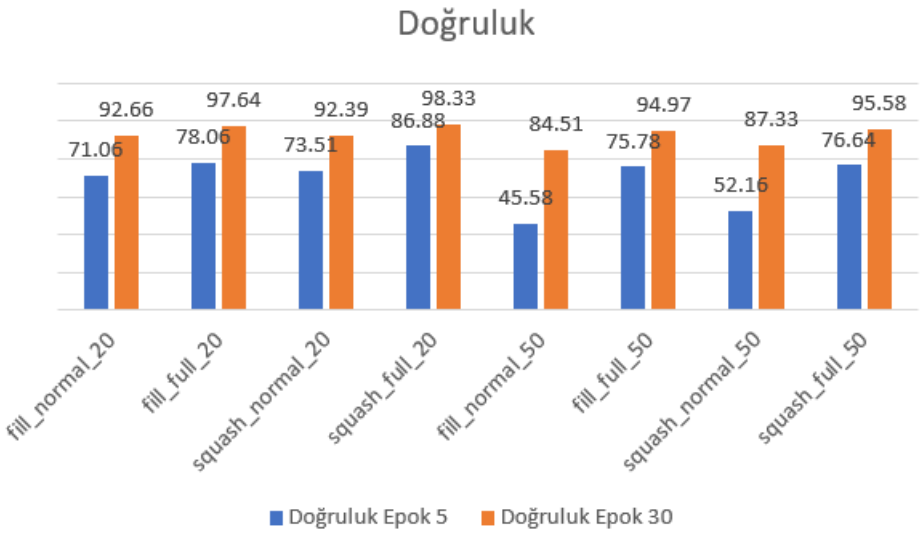
md_fill_normal_20	caffe	Done	5m	6:49 PM
md_fill_full_20	caffe	Done	11m	5:20 PM
md_squash_full_20	caffe	Done	11m	5:05 PM
md_squash_normal_20	caffe	Done	5m	4:56 PM
md_fill_full_50	caffe	Done	9m	4:44 PM
md_squash_full_50	caffe	Done	9m	4:34 PM
md_fill_normal_50	caffe	Done	4m	3:56 PM
md_squash_normal_50	caffe	Done	4m	3:46 PM

Şekil 3.9 Oluşturulan modellerin isimleri ve zaman bilgileri

Gerçekleştirilen 30 epokluk eğitim sonucunda %98.33 doğruluk ve %0.05 loss değeriyle en başarılı model squash_full_20 iken, %84.51 doğruluk ve %0.44 loss değeriyle en kötü model ise fill_normal_50 olduğu Çizelge 3-2'de görülmektedir. Bunun sebebi başarılı olan modeldeki veri sayısının yüksek olması ve resmin gerdirilmesidir. Resmin gerdirilmediği veri setlerinde resimde var olmayan gürültülerin resme doldurma yoluyla eklenmesi veriyi bozmuştur.

Çizelge 3-2 Tüm modellerin 5. ve 30. epoklarının loss ve accuracy yüzdeleri

	Loss		Accuracy	
Model	Epok 5	Epok 30	Epok 5	Epok 30
fill_normal_20	0.786	0.230	71.059	92.663
fill_full_20	0.640	0.108	78.055	97.638
squash_normal_20	0.730	0.228	73.505	92.391
squash_full_20	**0.352**	**0.058**	**86.875**	**98.333**
fill_normal_50	1.181	**0.444**	**45.577**	**84.513**
fill_full_50	0.683	0.151	75.778	94.972
squash_normal_50	**1.285**	0.362	52.157	87.334
squash_full_50	0.650	0.142	76.638	95.583



Şekil 3.10 Modellerin doğruluk yüzdeleri

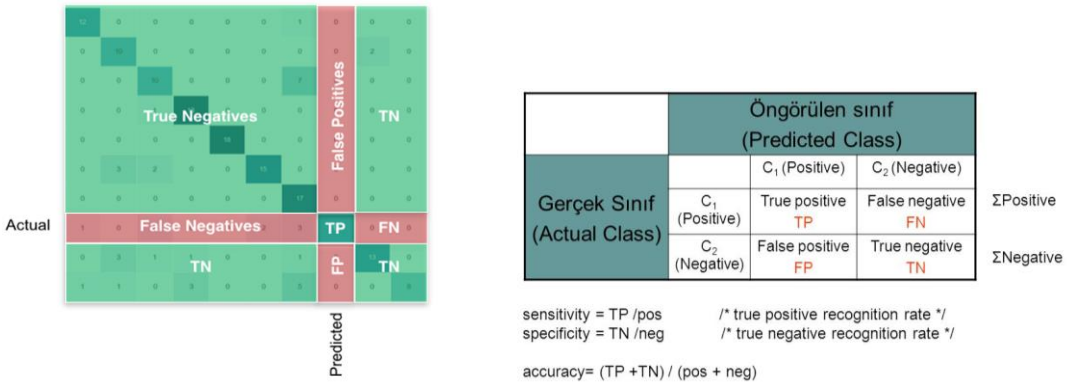


Şekil 3.11 En başarılı model(squash_full_20) ile en kötü modelin (fill_normal_50) eğitim değerlerinin epoklara göre değişimi.

Şekil 3.11'te görüldüğü gibi eğer veri seti iyi değilse (yetersiz sayıda, farklı çözünürlüklerde, farklı ışık ortamlarında vb. ve gürültülü) epoklara göre doğruluk ve loss değerlerinin değişimi daha düzensizdir. Eğer tüm görüntüler arama motorları yerine laboratuvar ortamında ve aynı kamera ile çekilseydi çok daha başarılı olurdu fakat gerçek dünyada başarı düşerdi, farklı ortamlarında da modelin kullanılabilmesi için çalışmada arama motorlarındaki resimleri tercih ettik.

3.5 Test

Eğitilen modeller test edilirken doğruluk değerinin yanında karmaşıklık matrisleri kullanılarak Sensitivity (Duyarlılık), Specifite (Özgüllük) değerleri hesaplanmıştır. Şekil 3.12'te görüldüğü üzere iki ve çoklu sınıfa ait karmaşıklık matrisi yer almaktadır.



Şekil 3.12 Karmaşıklık matrisi ve ölçüt formülleri

Temelde 3 farklı test yapılmıştır.

- DIGIT testi: Tek bir resmi ile detaylı model testi
- Python test1: 1341 test resmi kullanılarak yapılan model testi
- Python test2: modelin gerçek zamanlı kamera ya da video ile testi

Tüm deney modelleri CNN tipindedir ve çok sınıflı sınıflandırma problemini çözmek için tasarlanmışlardır. Çıkış katmanı hariç tüm katmanlarda Relu aktivasyon fonksiyonu kullanılmıştır. Son katmanda olasılıkların elde edilmesini sağlamak için Softmax fonksiyonu kullanılmıştır.

3.5.1 DIGIT testi

Gerçekleştirilen bu test Şekil 3.13'de görüldüğü gibi rastgele seçilen kız çocuğu resmi kullanılarak yapılmıştır. %90.88 gibi yüksek bir ihtimalle ayakta olduğunu doğru bir şekilde bilmiştir. Model çocuğun ellerinin biraz açık olması sebebiyle %7.98 bir ihtimalle zıplıyor olabilir sonucuna varmıştır.



Şekil 3.13 DIGIT testi sonuçları

Şekilde ayrıca gerçekleştirilen filtreleme işlemlerinin bir bölümü ve en son softmax sonucunu da görsel olarak göstermektedir.

3.5.2 Python Test1

Bu test te gerçekleştirilirken en başarılı model olan squash_full_20 seçilmiştir. Eğitimde kullanılmayan 1341 adet resim python dili ile yazılan bir uygulama ile test edilmiştir. Testin karmaşıklık matrisi Çizelge 3-3'te verilmiştir.

Çizelge 3-3 Python Test 1 in karmaşıklık matrisi

		Tahmin Edilen				
		Ayakta	Koşu	Oturma	Zıplama	Uzanma
Gerçek	Ayakta	204	7	1	7	0
	Koşu	11	218	4	14	0
	Oturma	3	1	266	6	4
	Zıplama	0	11	0	287	0
	Uzanma	0	0	11	6	280

Yapılan test sonucunda bütün sınıflar ortalama %97 gibi büyük bir doğrulukla sınıflandırılmıştır.

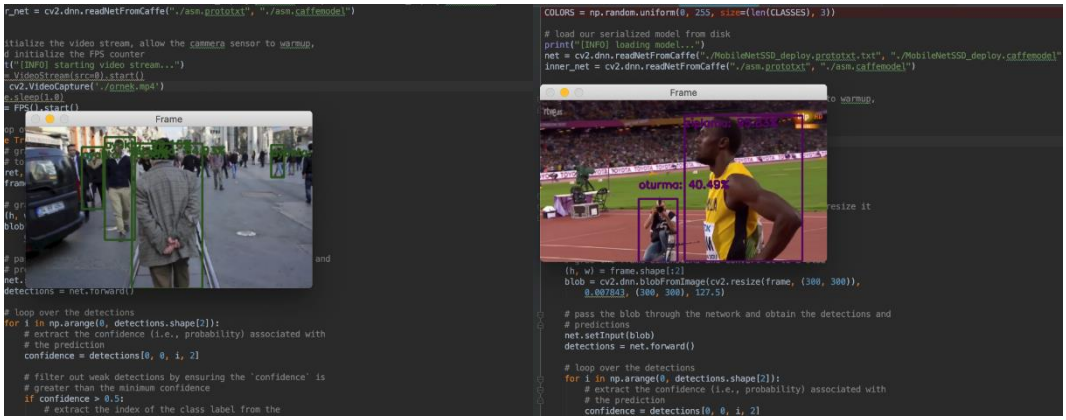
Çizelge 3-4 Python Test 1 in sonuçları

Sınıf	Doğruluk	Duyarlılık	Özgüllük
Ayakta	0.98	0.93	0.99
Koşu	0.96	0.88	0.98
Oturma	0.98	0.95	0.98
Zıplama	0.97	0.96	0.97
Uzanma	0.98	0.94	1.00

3.5.3 Python Test2

Oluşturulan en başarılı CNN modeli (squash_full_20) ile var olan SSD modeli ortak kullanılarak gerçek zamanlı video ya da kamera görüntülerindeki insanlar işaretlenerek aktiviteleri tahmin edilmiştir. SSD modeli çalışmaya özel Python dili ile yazılmış programa yüklenerek video sahnesindeki var olan insanlar tespit edilip çerçeve içine alınmıştır. Tezin amacı olan CNN modeli ise insan aktivitesinin ne olduğu ve olma ihtimali yüzdesi ile birlikte her insan için ayrı olarak tahmin edilerek çerçeve içine yazılmaktadır. Program merkezi işlemci hızına bağlı olarak çalışmaktadır. Testin yapıldığı bilgisayarın işlemcisi Intel i7 bir işlemcidir. Resimdeki insan sayısına bağlı

olarak saniyede 1-5 frame arası hızla ve yüksek bir doğrulukla çalışmaktadır. Şekil 3.14'te programın örnek bir çıktı sonuç görülmektedir. Şekil 3.14' te soldaki çerçevenin kalabalık olması sebebiyle SSD modeli tüm insanları tespit edemese de oluşturulan CNN modeli tespit edilen insanların aktivitelerini başarı ile tespit etmektedir. Eğer yazılımda grafik işlemci kullanılması durumunda performansın daha başarılı ve gerçek zamanlı olacağı düşünülmektedir.



Şekil 3.14 Python Test 2 görüntüsü

BÖLÜM 4

4 SONUÇLAR

Çalışma sonucunda yapılan testlerde insan aktivitesi büyük bir doğrulukla sınıflandırılmıştır. Çalışmanın bazı eksikliklerinin olmasının yanında güçlü yönleri fazladır. En önemlisi çalışmaya özel farklı programlama dillerinde faklı yazılımlar oluşturulmuştur. Diğer bir üstün yanı ise var olan hazır bir veri tabanı yerine çalışmaya özel bir resim veri tabanı oluşturulmuştur. Çalışmanın gerçek dünya testleri grafik işlemci yerine merkezi işlemcide gerçekleştirilmiştir. Bu yüzden performansı düşüktür.

Resimlerin düzenlenmesi ve etiketlenmesi için çalışmaya özel java dilinde bir program geliştirilmiştir. Program sayesinde tüm resimler manuel olarak tek tek seçilerek etiketlenmiştir. Tez için bilgisayar donanımı temin edilip, buna ubuntu işletim sistemi ve tüm DIGIT kütüphaneleri yüklenip, eğitimler bulut yerine lokal bir bilgisayarda yapılmıştır. Testlerde DIGIT sistemi dışında, macOS işletim sistemi kullanan bir dizüstü bilgisayar ve python dili ile yazılan programlar kullanılmıştır. 1. Program verilen resimleri test ederek karmaşıklık matrisi ve istatistiksel sonuçları hesaplamaktadır. 2. Program ise video ya da kamera görüntülerini analiz ederek devam eden görüntü üzerinde etkilemeler yaparak gerçek zamanlı olarak aktiviteleri tespit etmektedir. Böylece geliştirilen model sadece model olarak kalmayıp gerçek dünyada da kullanılmaktadır.

Yapılan çalışmaların çoğunda laboratuvar ortamında, aynı kamera ve ışık ortamında elde edilen resimler ya da hazır veri tabanları kullanılmıştır. Çalışmamızda ise farklı arama ve video sitelerinden alınan çok çeşitli veriler elde edilerek daha gerçekçi bir model oluşurmuştur. Bu işlem için ciddi bir zaman ve emek harcanmıştır. 5 faklı sınıf için ayrı olarak aramalar yapılmış toplam 10000 den fazla resim incelenmiştir. Çeşitli elemeler sonunda eğitim için 7117 test için ise1341 adet toplamda 8458 resim çalışmaya dahil edilmiştir.

Sonuç olarak 8458 adet resim kullanılarak yapılan çalışmada farklı insan aktiviteleri (ayakta, koşuyor, oturuyor, zıplıyor ve yatıyor) derin öğrenme teknikleri

kullanılarak ortalama %97 gibi büyük bir doğrulukla sınıflandırılmıştır. Tek bir resim ile yapılan DIGIT testinde %90.88, 1341 resim ile yapılan Python testinde ortalama % 97.4, video testlerinde ise ortalama 4 fps ve %92 doğruluk ile insan aktivitesi sınıflandırılmıştır.

Gelecekte insanların kim olduklarını tanıyan, daha fazla aktiviteyi tespit eden iki farklı model oluşturmayı, ayrıca grafik işlemciyi kullanan, mobil ve otomasyon uygulaması yapılarak insanların aktivitelerini takip edip analiz eden bir yazılım geliştirmeyi hedeflemekteyiz.

KAYNAKÇA

Altman, N. S. (1992). *An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician. 46 (3): 175–185.*

Bengio, Y., LeCun, Y., & Hinton, G. (2015). *Deep Learning. Nature .521 (7553): 436–444.].*

Bjerrum, E. J. (2017). *SMILES Enumeration as Data Augmentation for Neural Network Modeling of Molecules. ArXiv e-prints.*

David E. Rumelhart, J. L. (1987). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition.*

Girshick, R. (2015). *Fast R-CNN, ICCV '15 Proceedings of the IEEE International Conference on Computer Vision (ICCV), sy. 1440-1448.*

Hinton, G. ,., Osindero, S., & Teh, Y. W. (2006). *A Fast Learning Algorithm for Deep Belief Nets . Neural Computation. 18 (7): 1527–1554.*

Hinton, G. E. (2007). *Learning multiple layers of representation. Trends in Cognitive Sciences. 11 (10): 428–434.*

Hoang, L., Ke, S., Hwang, J., Yoo, J., & Choi, K. (2012). *Human Action Recognition based on 3D Body Modeling from Monocular Videos. In Proceedings of Frontiers of Computer Vision Workshop, Tokyo, Japan, 2–4 February ; pp. 6–13.*

Huo, F., Hendriks, E., Paclik, P., & Oomes, A. M. (2009). *Human Motion Capture and Pose Recognition. In Proceedings of the 10th IEEE Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS), London, UK, 6–8 May ; pp. 13–16.*

I. N. Aizenberg, N. N. (2000). *Multiple-Valued Threshold Logic and Multi-Valued Neurons, in Multi-Valued and Universal Binary Neurons, Boston, MA: Springer US, pp. 25–80.*

J.R. Uijlings, K. T. (2013). *Selective Search for Object Recognition, International Journal of Computer Vision, sy. 154-171.*

Jordan, M. I., & Bishop, C. M. (2004). *Neural Networks. In Allen B. Tucker. Computer Science Handbook, Second Edition (Section VII: Intelligent Systems). Boca Raton, Florida: Chapman & Hall/CRC Press LLC.*

Ke, S., Hwang, J., Lan, K., & Wang, S. (2011). *View-invariant 3D Human Body Pose Reconstruction Using a Monocular Video Camera. In Proceedings of Fifth ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC), Ghent, Belgium, 23–26 August.*

Lee, M., & Nevatia, R. (2009). *Human pose tracking in monocular sequence using multilevel structured models. IEEE Trans. Pattern Anal. Mach. Intell. 31, 27–38.*

Mannila, H. (1996). *Data mining: machine learning, statistics, and databases. Int'l Conf. Scientific and Statistical Database Management. IEEE Computer Society.*

Nitish Srivastava, G. H. (2014). *Dropout: A Simple Way to Prevent Neural Networks from Overfitting, Journal of Machine Learning Research 15.*

R. Girshick, J. D. (2014). *Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation, in The IEEE Conference on Computer Vision and Pattern Recognition, pp. 580–587.*

S. Hochreiter, J. S. (1997). *, «Long Short-Term Memory,» Neural Computation, pp. 1735 - 1780.*

Samuel, A. L. (1959). *Some Studies in Machine Learning Using the Game of Checkers. IBM Journal of Research and Development.*

Shaoqing Ren, K. H. (2015). *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, Advances in Neural Information Processing Systems 28 (NIPS).*

Sonka, M., Hlavac, V., & Boyle, R. (2008). *Image Processing, Analysis, and Machine Vision.* Newyork: Thomson.

Sontag, H. T. (1991). *Turing Computability With Neural, Applied Mathematics Letters, pp. 77-80.*

Stuart J. Russell, P. N. (2010). *Artificial Intelligence: A Modern Approach,.*

Turing, A. M. (1950). *Computing Machinery and Intelligence. Mind 49: 433-460.*

W. Liu, D. A. (2016). *SSD: Single Shot MultiBox Detector. European Conference on Computer Vision (pp. 21-37). Springer International Publishing.*

Y. LeCun, B. B. (1990). *Handwritten digit recognition with a backpropagation network, in NIPS'89.*

ÖZGEÇMİŞ

1976 yılında doğdu. İlk ve orta öğrenimini Simav'da tamamladı. 1993 yılında Çınarlı Endüstri Meslek Lisesi'nden mezun oldu. 1998 yılında Marmara Üniversitesi Teknik Eğitim Fakültesi Elektronik Öğretmenliği bölümünden mezun oldu. 2003 yılında Trakya Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Ana Bilim Dalı Yüksek Lisans Programına başladı. 2 yıl Mili Eğitim Bakanlığı'nda Elektronik Öğretmeni olarak görev yaptı. Daha sonra özel sektörde çalışmaktadır. Evli ve iki çocuk babasıdır.

EKLER

Testlerde kullanılan insan aktivitelerine ait tüm resimlere aşağıdaki adresten erişilebilir. https://www.kaggle.com/asimerkut/humanactivity