TÜRKİYE CUMHURİYETİ
YILDIZ TEKNİK ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

YÜZ FİLTRESİ OLUŞTURMA

19011602 — Cafer YAHŞİ
18011616 — Mehmet Onur AYSEL

BİLGİSAYAR PROJESİ

Danışman
Doç. Dr. Öğretim Üyesi Ayşe Betül OKTAY

Haziran, 2022

TEŞEKKÜR

Yararlandığımız ders notlarından ve slaytlardan dolayı hocalarımıza, Towards Data Science sitesinde emek verenlere, ders videolarını paylaştıklarından dolayı Stanford Üniversitesi'ne ve Machine Learning Mastery kitabının yazarına emeklerinden dolayı çok teşekkür ediyoruz. Kendilerinden çok şey öğrendik.

Cafer YAHŞİ
Mehmet Onur AYSEL

İÇİNDEKİLER

KISALTMA LİSTESİ

ANN	Artificial Neural Networks
CNN	Convolutional Neural Networks
HoG	Histogram of Oriented Gradients
ReLu	Rectifier Linear Units
SPM	Spacial Pyramid Matching

ŞEKİL LİSTESİ

ÖZET

YÜZ FİLTRESİ OLUŞTURMA

Cafer YAHŞİ
Mehmet Onur AYSEL

Bilgisayar Mühendisliği Bölümü
Bilgisayar Projesi

Danışman: Doç. Dr. Öğretim Üyesi Ayşe Betül OKTAY

Bu proje, herhangi bir sosyal medya platformunda bulunan bir filtrenin nasıl çalıştığını, bu filtrelerin, fotoğrafta ağız, burnun ve kulağın yerini nasıl tespit etiğini ve bu uzuvlara nasıl filtre uyguladığını anlatmakta ve filtre işlemini, hem evrişimsel sinir ağı modelini kullanarak hem de kameradan gelen görüntüyü gerçek zamanlı olarak uygulamaktadır.

Anahtar Kelimeler: evrişimsel sinir ağları, yapay sinir ağları, makine öğrenmesi, aktivasyon fonksiyonu, kayıp fonksiyonu, katman, çok katmanlı yapı, nöron, batch normalizasyonu, öznitelik tanımlama, ReLU, sigmoid fonksiyon, tanh fonksiyonu, softmax fonksiyonu, ortaklama, maksimum ortaklama, düzenleştirme, düzleştirme çıkarma(atma), padding işlemi

ABSTRACT

CREATING FACE FILTER

Cafer YAHŞİ
Mehmet Onur AYSEL

Department of Computer Engineering
Computer Project

Advisor: Assoc. Prof. Dr. Ayse Betul OKTAY

This project includes such informations about how filtering process works which are in social media, how these filters can automatically detect face landmarks at works on these face parts such as mouth, nose and ear. It also applies face filtering process witj simple convolutional neural network model and by real time on image which comes from camera.

Keywords: convolutional neural networks, artificial neural networks, activation function, loss function, error rate, layer,neuron, batch normalization, feature description, relu, sigmoid function, tanh function, softmax function, pooling, max pooling, regularization, flattening, dropout, padding

1
Giriş

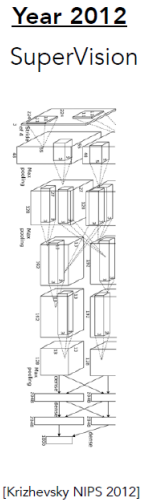
Bilgisayar biliminin son 20 yılda ivmelenerek gelişmesi, beraberinde birçok yeni alanı da ortaya çıkardı.Bu alanlardan son yıllarda en çok adını duyduğumuz iki tanesi: Makine Öğrenmesi ve Bilgisayarlı Görü. Peki bu iki kavram hayatımıza nasıl girdi sorusuna cevap verecek olursak genelde diğer buluş ve icatlarda da olduğu gibi kaynak: Nasıl insanlar daha hızlı seyahat edebiliriz sorusuna cevap araken doğaya bakıp uçan nesneleri örnek alıp uçaklar yaptıysa burada da insan beyni ,beynin öğrenme ve bilgi çıkarma yapısı ve aynı zamanda insanlardaki görme ve görülen karelerin beyinde işlenme mekanizması acaba bir makine tarafından yapılabilir mi sorusuna cevap aranmıştır. Tabi ki son yıllarda özellikle sosyal medya platformlarının hayatımıza girmesi,yazılı medya yerine görsel medyanın daha fazla kullanılması ve canlı yayın platformlarının da hayatımıza girmesiyle fotoğraf ve video gibi verilerle daha fazla haşır neşir olmamız, bilgisayarlı görü alanının daha da fazla öne çıkmasına neden olmuştur. Bunlara ek olarak son yıllarda gelişen kamera ve lens teknolojleri ile mobil cihazlardaki kamera kalitesinin artması ve ayrıca mobil cihazların da halk arasında kullanımının logaritmik olarak yaygınlaşması yüksek miktarda görsel veri elde etmemizi sağlıyor fakat bu da yeni bir sorun ortaya çıkarıyor: bu kadar yüksek hacme sahip veriyi nasıl kullanırız? Burada da karşımıza makine öğrenmesi çıkıyor. Makine öğrenmesi için, temelde elimizde bulunan ve işlenmemiş, karmaşık veriden bazı metotlar kullanarak elimizde olmayan bilgiyi çıkarma işlemi denebilir. Makine öğrenmesi yaparken görsel veri kümeleri kullandığımızda bu iki alan biribiri içine çok girmekte. Özellikle sosyal medyanın yaygınlaşması ve bu tarz platformlarda gerek insanların güzel gözükmek istemesi, gerek eğlence amaçlı bir çok fotoğraf filtresi bulunmakta. Bu belgede, herhangi bir sosyal medya platformunda bulunan bir filtre nasıl çalışıyor, bu filtreler fotoğrafta ağzın veya burnun yerini nasıl tespit ediyor ve bu uzuvlara nasıl filtre uyguluyor bunları anlatmaktadır.

2
Makine Öğrenmesi ve Derin Öğrenme

Makine öğrenmesi, genellikle büyük çapta projeler için kullanıldığından veri kümesi olarak şirketlere ait veya herhangi bir problem üzerine oluşturulmuş veritabanlarında tutulan veriler kullanıldığı için veriseti her bir veri için o veriye ait özellikleri tutan kolonlara bölünmüştür. Bizler de bu hazır özellikleri kullanırız fakat fotoğraflar da bildiğiniz gibi piksellerden oluşur ve her piksel için eğer fotoğraf renkliyse 3 adet, siyah - beyaz ise 1 adet 0-255 arasında sayı tutulur. Bu yüzden fotoğraflarda hazır öz nitelikler bulunmaz. Eğer makine öğrenmesi yapacağımız verisetinde hazır öz nitelikler yoksa genelde böyle veri kümeleri için derin öğrenme kullanılır. Derin öğrenme, etiketi olmayan verileri kullanarak bu veriler arasındaki bağları veya örüntüleri bulan yapay sinir ağlarını kullanan makine öğrenmesinin bir alt dalıdır. Amaç, makine öğrenmesiyle aynıdır, sadece kullandığımız veriler etiket içermezler. Biz de projemizde görsellere dayalı bir veri kümesi kullandığımız için derin öğrenme algoritmalarından CNN algoritmasını kullandık.

3
Evrişimli Sinir Ağları

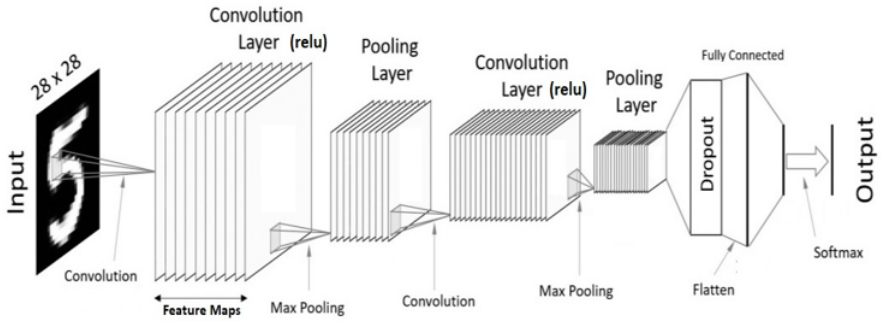
CNN, Stanford Üniversitesi öncülüğünde yapılan, Stanford ve Princeton Üniversitelerinin katkılarıyla hazırlanmış, 22 bin kategori ve 14 bin fotoğraftan oluşan ImageNet veri kümesi kullanılarak yapılan ImageNet yarışması için 2012'de Alex Krizhevsky, Ilya Sutskever ve Geoffrey Hinton tarafından Supervision ismiyle geliştirildi -AlexNet olarak da anılır- ama aslına temeli 1990'lara dayanmaktadır. 1998'de Jan LeCun ve ortakları tarafından Bell Labaratuvarında, elle yazılan çeklerdeki sayıların, adreslerin ve postane tespiti için CNN geliştirmişlerdi. Fotoğraflardan alınan pixellerin sayıya mı yoksa harfe mi ait olduğuna göre sınıflandırma yapıyordu. 2012'de geliştirilen SuperVision, transistörlerin sayısının artması ve ekran kartlarının oldukça gelişmesinden yararlanıp modeli oldukça ilerletti. 1998'de etiketli verilerin sayısı ve bu verilere erişebileceğiniz veri kümeleri oldukça azdı.



Şekil 3.1 Supervision

Convolutional Neural Network, Türkçe ismiyle evrişimsel sinir ağları girdi olarak verilen fotoğrafların üzerindeki sadece o fotoğrafa ait özellikleri keşfeden bir derin

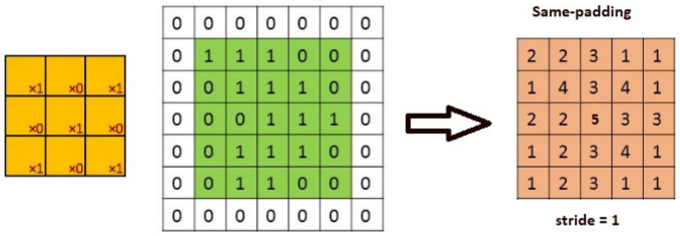
öğrenme algoritmasıdır. Size bir köpek fotoğrafı gösterildiğinde hemen aklınızdaki köpek imajıyla o fotoğrafı karşılaştırırsınız ve spesifik özellikleriyle onun köpek olduğunu algılarsınız. Aslında evrişimsel sinir ağlarıda böyle çalışır ama bilgisayar bilimlerinde her şey nihayetinde sayılarla ilişkilidir. Sisteme yüklediğiniz köpek fotoğraflarıyla sinir ağı bir köpek imajı yaratır, tabiki bizde olduğu gibi değil daha çok fotoğrafın bizim göremediğimiz düz veya convex alt düzey şekilleriyle belirler. Bu işlemleri kademeli olarak yapabilmek ve ayrıca bazı makine öğrenmesi problemlerini aşabilmek için katmanlara ayrılır.



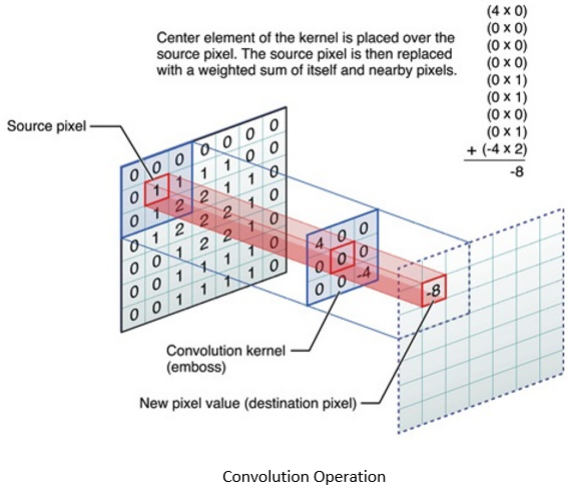
Şekil 3.2 Evrişimli Sinir Ağları Katman Yapısı

3.0.1 Evrişimsel Katman

Convolutional Layer ya da Evrişimsel Katman, algoritmanında adını aldığı ana katmandır aynı zamanda giriş katmanıdır. Verinin hangi boyutta gelceği bu katmanda belitilmelidir.Bu katmanın ana işlevi fotoğrafta bulunan göremediğimiz düz veya convex alt düzey sekilleri keşfetmek.Bu işleme katmanada adını veren Convolution Operation denir.Bu işlem fotoğrafa ait pixel değerleri üzerinde sayısı değişen genellikle tek sayıdan oluşan 3x3,5x5,7x7 matrislerin gezdirilmesiyle elde edilir.Burada sorun filtre uygulandıktan sonra input değişkeninin boyutunun azalmasıdır bu problemi çözmek içinde padding işlemi yapılır.Padding işlemi convolution işlemi uygulanmadan önce fotoğrafın pixel değerlerinin etrafının bir çerçeve gibi sarılması işlemidir.En çok kullanılan padding yöntemi same paddingtir ve 0 lardan oluşan bir çerçeve kullanılır.



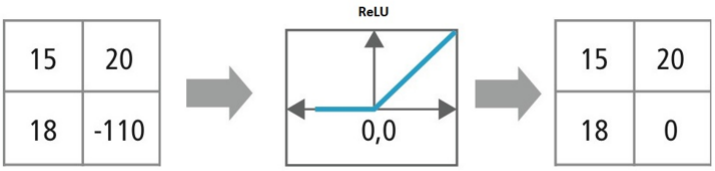
Şekil 3.3 Same Padding İşlemi



Şekil 3.4 Evrişim (Convolution) İşlemi

3.0.2 Doğrusal Olmayan Katman

Doğrusalsızlık katmanı ya da **Non-Linearity Layer** Sinir ağlarındaki bir diğer problem de doğrusallıktır. Bu problemi çözmek için genelde aktivayson fonksiyonları kullanılır. Bu fonksiyonlar aldığı değerleri belirli değerler arasına sıkıştıran fonksiyonlardır. **Relu, tanh, sigmoid, softmax** gibi bir çok aktivayson fonksiyonu bulunur. Genelde fotoğraflarda karmaşık yapılar tespit edildiği ve daha optimize çalıştığı için **Rectifier(ReLu)** fonksiyonu kullanılır.

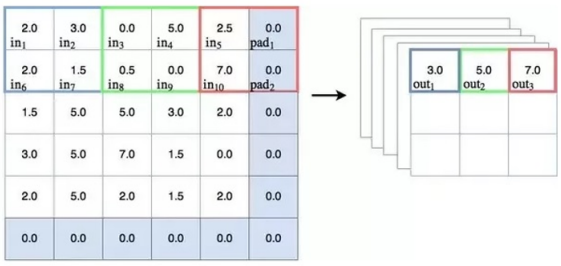


Şekil 3.5 ReLu Fonksiyonu

3.0.3 Ortaklama (Aşağı Örnekleme) Katmanı

Pooling ya da ortaklama katmanın temel amacı, parametre sayısını dengelemek ve aşırı uyum, yani modelimizin verisetini ezberleme sorununu çözmektir. Genelde her bir evrişim katmanından sonra modele bir de ortaklama katmanı eklenir. ortaklama katmanı, bu sorunları çözerken genelde aşağı örnekleme(downsampling) ya da sub-sampling kullanır. En çok kullanılan çeşidi max pooling olsa da average, L2-norm pooling gibi farklı pooling çeşitleri de vardır. Genelde 2x2 filtre kullanılsa da veri setinin büyüklüğüne göre bu değişebilir. Max poolingde aşağıda görüldüğü gibi seçilen

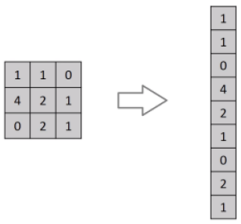
pencereye göre o penceredeki max değer bulunur ve o dört kare için sadece o sayı alınır. Böylece örnek sayısı azalmış, yani sub-sampling yapılmış olunur.



Şekil 3.6 Maksimum Ortaklama(Max Pooling)

3.0.4 Düzleştirme Katmanı

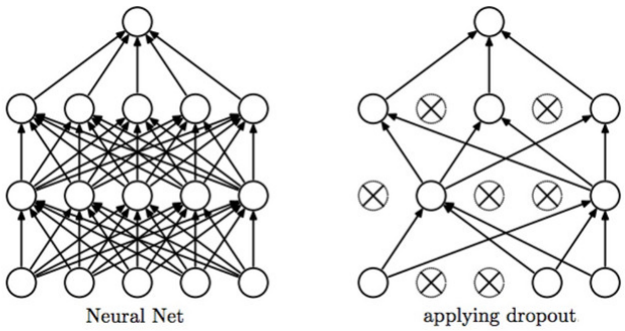
Aslında her evrişimsel sinir ağları yapısı modelinin en sonunda bir yapay sinir ağları yapısı bulunur. Yapay sinir ağları yapısı da girdi olarak sadece tek boyutlu matrisleri kabul eder. Bu yüzden yeterli sayıda evrişim, ortaklama, doğrusal olmayan katman eklendikten sonra yapay sinir ağı yapısına girmeden düzleştirme (flattening) işlemi yapılması gerekir. Düzleştirme, herhangi bir boyuta sahip bir matrisin 1xn şekline çevrilmesi işlemidir.



Şekil 3.7 Düzleştirme

3.0.5 Çıkarma (Atma)

Çıkarma işlemi, katman olarak sayılmasa da her katman işleminden sonra yapılır. Çıkarma, belirlenen değer altında kalan nöronları sistemden çıkarma işlemidir. Yapılma amacı, negatif değer üreten nöronları sistemden atarak düzenleştirmeyi (regularization) arttırmak ve dolayısıyla modelin karmaşıklığını engelleyerek sistemdeki aşırı uyumu azaltmaktır.



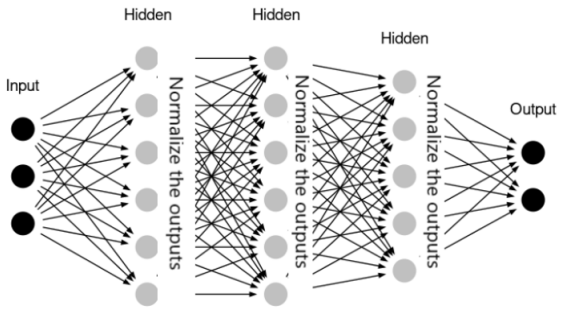
Şekil 3.8 Nöronları Çıkarma(Atma)

3.0.6 Batch Normalizasyonu

Batch normalizasyonu; katmanların, input olarak aldıkları parametreleri her seferinde değiştirme işlemini oldukça hızlandırır çünkü birden fazla katmanın aynı anda, kayıp fonksiyonunun değerine göre parametre değiştirmesine olanak tanır, dolayısıyla katmanların sırasıyla öğrenme yapmasından ziyade her katmanın aynı anda öğrenme yapmasına olanak sağlar. Bu sayede modelin eğitim hızını yükseltir ayrıca öğrenme oranının yüksek olduğu modellerde gradyantların (gradients) yok olmasını engeller. Modelin daha kararlı olmasını sağlar. Modeldeki dağılımın oratalamasını 0, standard sapmasını 1 olacak şekilde ayarlar ve tüm değerler -1 ile 1 arasına şıkıştırılır, Standard Gaussian'da olduğu gibi.

*Batch normalizasyonunun kullanılması gereken yer aktivasyon fonksiyonundan hemen öncedir. Böylece sıkıştırılmış değerlerin aktivasyon fonksiyonu ile belirli bir değer aralığına yakınsaması daha kolay sağlanmış olur.

*Ayrıca batch normalizasyonu, veri karmaşıklığını azalttığı için (veriyi sıkıştırdığı için) modelin düzenleştirilmesini(regularization) arttırmak için çıkarma katmanının kullanılmasına gerek yoktur.

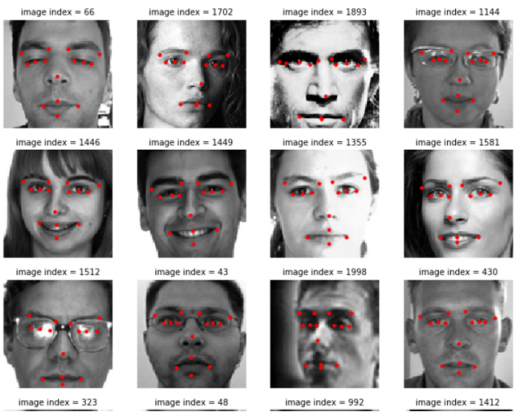


Şekil 3.9 Batch Normalizasyonu

4
CNN Uygulaması

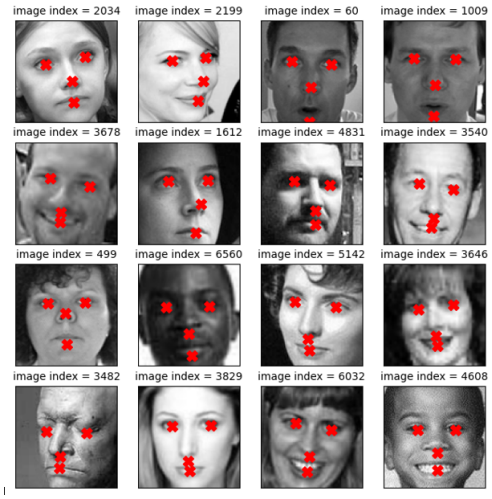
Projede amacımız, bilgisayarda kameramız açıkken algoritmamız sizin tarfınızdan belirlenen 8 adet yüz filtresinden birtanesini yüzünüze uygulamak ve bunu siz hareket ettiğinizdede yüzünüzle birlikte hareket etmesini sağlamak. Bu işlemi gerçekleştirmek için Convolutional Neural Network algoritmasını kullanacağız. Öncelikle sinir ağımız eğitmek için kullanacağımız dataseti tanıyalım .Data setimiz 7000 adet siyah beyaz insan yüzü fotoğrafından oluşuyor.Ayrıca amacımız insan yüzündeki landmarkları belirlemek olduğu için her insan yüzü için işaretlenmiş 15 adet nokta mevcut. Her nokta x ve y ekseninden oluştuğu için totalde 30 adet veri tutulmakta her fotoğraf için. Fakat totalde tüm fotoğrafların 30 noktası işaretlenmemiş, bu yüzden hem bazı noktalar bizim işimize yaramadığı için hem de bazı fotoğraftaki verilerin eksik olmasından kaynaklı, bizde işimize yaracak filtre uyguluyabiliceğimiz sağ göz merkezi,sol göz merkezi,burun ucu ve ağız ortası işaretlenmiş verileri kullanacağız.

Kullanacağımız verileri seçtikten sonra bir değişkende fotoğrafa ait değerler diğer değişkende o fotoğrafa ait sağ göz merkezi, sol göz merkezi, burun ucu ve ağız ortası gibi her değişken için 2 adet 0-96 arası kordinat tutan 8 kolonlu bir değişken oluştu. Datasetimizi %20'si test %80'i eğitim omak üzere 2 parçaya bölüyoruz. Modelimizi 4 defa tekrar edecek şekilde sırasıyla evrişim katmanı, Batch



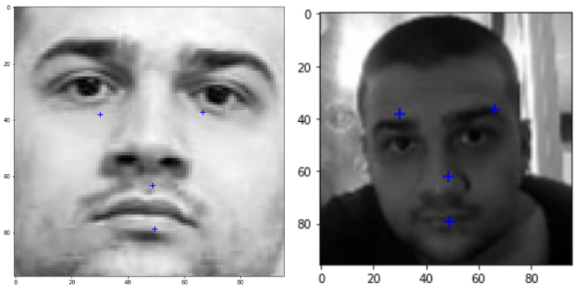
Şekil 4.1 Veri Setini Oluşturan Birkaç Fotoğraf

Normalizasyonu, ortaklama ve Atma katman ve işlemleri olucak şekilde ayarlıyoruz. Convolutional katmanında filtre sayımız genelde 32 veya 64 olarak değişiyor, padding algoritması olarak en çok kullanılan maximum ortaklama(max pooling) algoritmasını kullanıyoruz. Fotoğraflarımız 96x96 olduğu için input boyutumuz da 96x96. Aktivasyon fonksiyonu olarak, ise karmaşık sistemlerde daha çok kullanıldığı için ReLU fonksiyonunu kullanıyoruz.Sistemde eş zamanlı öğrenme yapmak, sistemin öğrenme hızını arttırtmak ve aynı anda ezberleme(overfitting) sorununu çözdüğü için Batch Normalization kullandık. Optimizer olarak ADAM yani Adaptif Moment Tahmini(Adaptive Moment Estimation) kullandık. Adam optimizer kullanama sebebimiz öğrenme hızını modelin gidişatına göre arttırıp azaltabilmesi.

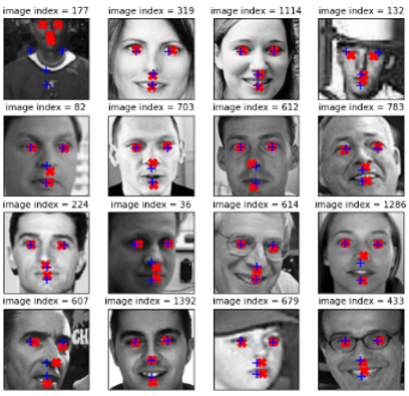


Şekil 4.2 Doğru Yüz Konumları

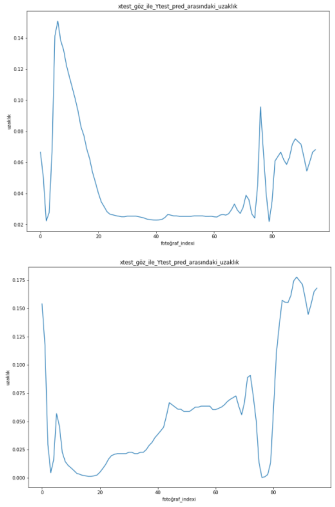
Modelimizi oluşturduktan sonra test etmek için ayırdığımız veriyi test için kullanıyoruz. Sonuçlarda net bir sınıf olmadığı için doğruluk yüzdesi veremiyoruz. Bu yüzden bulduğumuz koordinatlarla sistemde işaretlenmiş koordinatlar arasındaki uzaklığı öklid uzaklık metriğini kullanarak hesapladık. Herbir değişken için toplamda 4 adet grafik oluşturduk. Enson gerçek hayatta ne kadar optimize çalıştığını görebilmek için 2 adet fotoğraf için modelimize tahmin yaptırdık. Çok yüksek oranda optimize çalışmasa da sonuçlar iyi denebilecek seviyede.



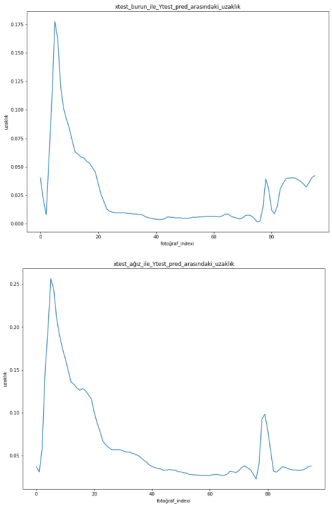
Şekil 4.3 Model'in, Cafer'in yüzünün hatlarını bulması



Şekil 4.4 Caption



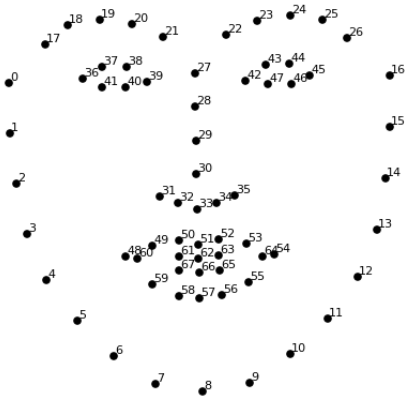
Şekil 4.5 Performans 1



Şekil 4.6 Performans 2

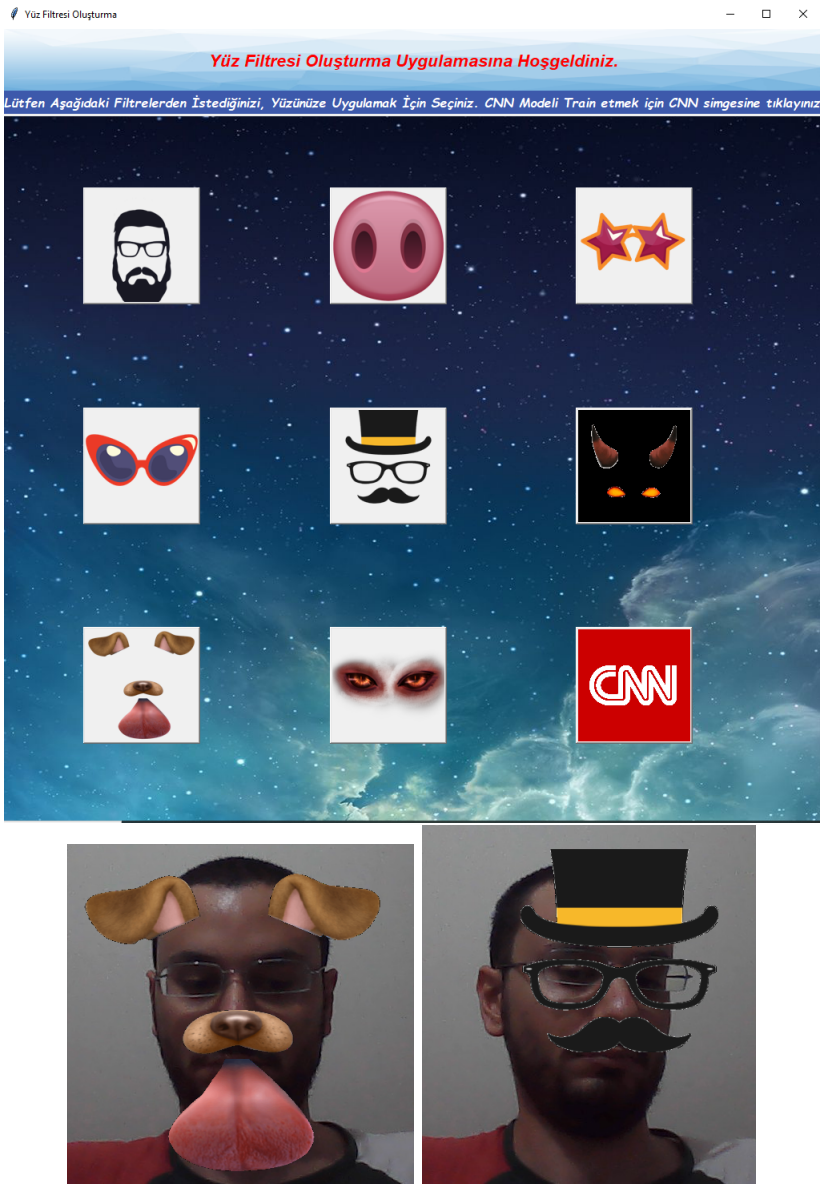
5
Gerçek Zamanlı Uygulama

Bilgisayarın kamerasıdan alınan görüntünün gerçek zamanlı olarak filtrelenmesi için açık kaynak kodlu CMake yazılımı yüklendi ve bu yazılımın dlib kütüphanesinin yüz tanıma fonksiyonu kullanıldı. Bu fonksiyonun çalışabilmesi için ayrıca dlib kütüphanesinin shape predictor fonksiyonu kullanıldı. Bu fonksiyon, aşağıdaki şekli .dat dosyası olarak alıyor ve kamerada gördüğü yüzün hatlarını, bu şeklin üzerindeki noktalara göre buluyor:



Şekil 5.1 Tahmin unsuru (predictor)

Bu tahmin unsuru olarak adlandırdığımız .dat dosyasının üzerindeki noktaları kullanarak göz, burun, ağız organlarının yüksekliğini ve genişliğini, basit bir işlem ile hesaplayoruz. Daha sonra bunların üzerine filtre olarak vereceğimiz fotoğrafın uygun olarak yerleşmesi için fotoğrafı, hesapladığımız organın boyutu ile aynı olacak şekilde boyutunu değiştiriyoruz. Daha sonra bu fotoğrafın gri halini yaratıp bunun üzerine treshold uyguluyoruz, bu alan bizim maskemiz olmuş oluyor. Bu maskeyi kullanarak basit bir bitwise and işlemi yaparak üzerine filtre yapacağımız yüz hattını karartmış olduk, yani filtreledik. Bu alanın üstüne de boyutu değiştirilmiş filtre fotoğrafımızı koyduk mu ve bu işlemlerin sürekli olmasına dikkat ediyoruz, kameradaki konumumuz değiştikçe filtrenin boyutunun küçülüp büyümesi ve konum değiştirmesi için.



Şekil 5.2 Gerçek Zamanlı Yüz Filtresi

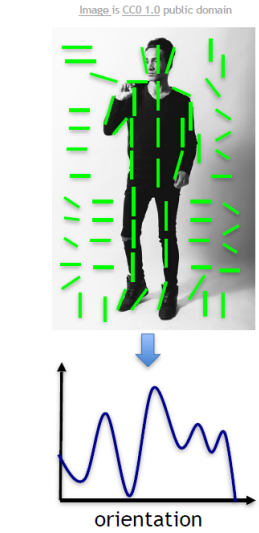
5.0.1 Yüz Dedektörünün Çalışması

Dlib kütüphanesinin frontal face detector fonksiyonu aslında evrişimsel sinir ağlarını(CNN) ve **HOG(Histogram of Gradients)** kullanıyor.

5.0.1.1 HOG Nedir?

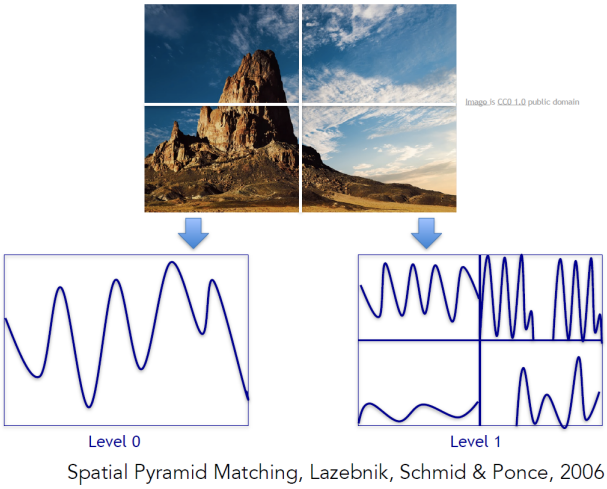
HOG, 2005 yılında Dalal ve Triggs tarafından geliştirildi. Aslında Canny Kenar Saptayıcı(Canny Edge Detector) ve David Lowe'nin 1999'da geliştirdiği obje eşleştirme uygulaması olan SIFT gibi bir öznitelik açıklayıcı olarak çalışıyor(feature descriptor). İnsan vücuduna ait bir fotoğraftan insan vücudunun bölgeleri ayrıştırılabiliyor ve fotoğraftaki insanın nasıl bir pozisyonda olduğu anlaşılabiliyordu.

Görüntü işleme uygulamalarında, obje tespiti(object detection) için sıklıkla kullanılan bir yöntem. HOG, bir fotoğrafta belirlenmiş bir alanda uyum gösteren gradyent oluşumlarını sayıyor diye özetlenebilir.



Şekil 5.3 HoG

2006'da geliştirilen **Uzaysal Piramid Eşleme(Spatial Pyramid Matching)**: Algoritmanın ana fikri, fotoğrafın featurelarının bize fotoğrafın nerede çekildiğini belirtebileceği fikrine dayanıyordu. (Tepe, orman, deniz, otoban...) Bu featurelar, fotoğrafın farklı yerlerinden farklı çözünürlükler ile alınıyor ve feature descriptora konuluyor ve nihayetinde SVM algoritması çalıştırılıyordu.



Şekil 5.4 Uzaysal Piramid Eşleme

Özgeçmiş

BİRİNCİ ÜYE

İsim-Soyisim: Cafer YAHŞİ
Doğum Tarihi ve Yeri: 13.10.1999, Bursa
E-mail: caferyi33@gmail.com
Telefon: 0538 810 07 24
Staj Tecrübeleri: -

İKİNCİ ÜYE

İsim-Soyisim: Mehmet Onur AYSEL
Doğum Tarihi ve Yeri: 24.03.1998, İzmir
E-mail: onur.aysel@std.yildiz.edu.tr
Telefon: 0531 254 02 98
Staj Tecrübeleri: -

Proje Sistem Bilgileri

Sistem ve Yazılım: Windows İşletim Sistemi, Python, CMake
Gerekli RAM: 2GB
Gerekli Disk: 800MB (519 MB for data, 100 MB for CMake, 90 MB for python and other for applications(Spyder, anaconda,...)