Konveks Sınıf Modelleri Kullanarak Dijital İmgelerdeki Nesne Görüntülerinin Konumlarının Bulunması

Proje No: 109E279

Doç. Dr. Hakan Çevikalp
Hüseyin Gündüz
Musa Aydın
Güvenç Usanmaz
Onur Akyüz

ŞUBAT 2013
ESKİŞEHİR

ÖNSÖZ

Eskişehir Osmangazi Üniversitesi, Mühendislik Mimarlık Fakültesi, Elektrik – Elektronik Mühendisliği Bölümü, Makine ile Öğrenme ve Bilgisayarlı Görü laboratuvarlarında, Eskişehir'de 1 Mart 2010 – 1 Mart 2013 tarihleri arasında gerçekleştirilen ve Türkiye Bilimsel ve Teknolojik Araştırma Kurumu (TÜBİTAK) tarafından 109E279 proje numarasıyla desteklenen "Konveks Sınıf Modelleri Kullanarak Dijital İmgelerdeki Nesne Görüntülerinin Konumlarının Bulunması" başlıklı projenin sonuç raporu burada sunulmuştur. Projede görsel nesne konum bulma için konveks sınıf modellemeye dayalı yeni tek sınıflı sınıflandırıcılar geliştirilmiş ve bu sınıflandırıcılar ardışıl bir yapıda kullanılmıştır. Projemiz kapsamı ve amaçları doğrultusunda başarıyla sonuçlandırılmıştır. Tüm proje çalışanları olarak TÜBİTAK' a desteği için teşekkür ederiz.

Proje kapsamında 2.5 ay süreyle Fransa'nın Grenoble kentindeki CNRS laboratuvarında çalışmalar yaptım. Bu zaman süresince laboratuvarda beni ağırlayan ve çalışmalarıma yardımcı olan Bill Triggs'e ve bir çok konuda hiç bir yardımını esirgemeyen Sibt Ul Hussain'e teşekkür ederim.

Doç. Dr. Hakan Çevikalp

Proje Yürütücüsü

İÇİNDEKİLER

ŞEKİLLER

TABLOLAR

ÖZET

Bu projede nesne konum bulma için nesnelere ait sınıfları konveks sınıf modelleri ile yakınsayarak ardışıl tek sınıflı sınıflandırıcılar kullanan bir sistem geliştirilmiştir. Bir nesne konum bulma sistemi sayısal imgeler içindeki nesne kategorisine ait herhangi bir örneğin varlığını tespit ederek konumunu döndürebilmelidir. Son zamanlarda geliştirilen bir çok konum bulma yöntemi, kayan pencereler yaklaşımını kullanarak imgeden gelen bir pencereyi nesneye ait örnek yada arka-plan olarak sınıflardırmaktadır. Bu amaçla makine ile öğrenme temelli Destek Vektör Makineleri ve hızlandırmalı (boosting) sınıflandırıcılar gibi iki sınıflı sınıflandırıcılar arka-plana ait kolay örneklerin hızlı bir şekilde elenebilmesi için genellikle ardışıl bir şekilde kullanılmıştır. Biz bu projede nesneye ait sınıfa odaklanarak, bu sınıfı modelleyerek oluşturulan tek sınıflı sınıflandırıcıların nesne konum bulmada ikili sınıflara oranla daha iyi bir alternatif olduğunu göstermeyi amaçladık. Bunun temel nedeni özellikle ardışıl sınıflandırıcının ilk katmanlarında tek sınıflı sınıflandırıcıların oldukça basit ve hızlı bir şekilde oluşturulması ve arka-plana ait örnekleri daha hızlı bir şekilde reddetmeleridir. Bu amaçla nesne sınıflarını doğrusal affine bir hiper-düzlem, doğrusal ve doğrusal olmayan hiper-küreler ile betimleyerek, bu konveks modellere olan uzaklıkları kullanan ardışıl bir sınıflandırıcı sistemi geliştirdik. Yapılan deneysel çalışmalarda “Faces in the Wild” ve ESOGU yüz sezme veritabanlarında literatürdeki en iyi sonuçlara benzer yada daha iyi sonuçlar alınırken INRIA kişi sezme veri tabanında literatürdeki en iyi sonuçlar elde edilmiştir. “PASCAL VOC 2007 challenge” nesne veritabanları ile yapılan çalışmalarda ise nesne sınıflarına ait örnek sayısının az olması sebebiyle bir çok nesne kategorisinde literatürdeki sonuçlardan çok daha iyi sonuçlar alınamamasına karşın bazı nesne gruplarında literatürdeki en iyi sonuçlar elde edilmiştir. Bunun yanında, beklenildiği üzere tek sınıflı sınıflandırıcılar iki sınıflı sınıflandırıcılara oranla işlem yükü açısından büyük yararlılıklar sağlamıştır.

Anahtar Kelimeler – Nesne konum bulma, ardışıl sınıflandırıcı, konveks sınıf modeli, tek sınıflı sınıflandırıcı, iki sınıflı sınıflandırıcı.

ABSTRACT

In this project we developed an efficient object detection system based on cascade of one-class type classifiers using convex class models for approximating object classes. An object detector must detect and localize each instance of the object class of interest in the image. Many recent detectors adopt a sliding window approach, reducing the problem to one of deciding whether the detection window currently contains a valid object instance or background. Machine learning based discriminants such as SVM (Support Vector Machine) and boosting are typically used for this, often in the form of classifier cascades to allow more rapid rejection of easy negatives. We argue that "one class" methods – ones that focus mainly on modeling the range of the positive class – are a useful alternative to binary discriminants in such applications, particularly in the early stages of the cascade where one-class approaches may allow simpler classifiers and faster rejection. We implement this in the form of a short cascade of efficient nearest-convex-model one-class classifiers, starting with linear distance-to-affine-hyperplane and interior-of-hypersphere classifiers and finishing with kernelized hypersphere classifiers. We show that our methods have very competitive performance on the Faces in the Wild and ESOGU face detection datasets and state-of-the art performance on the INRIA Person dataset. Although we could not obtain the best results on PASCAL VOC 2007 challenge datasets on many object categories, we still managed to achieve the best results for some object categories despite the scarcity of the object samples. As predicted, the one-class formulations provide significant reductions in classifier complexity relative to the corresponding two-class ones.

Keywords – Object detection, cascade classifier, convex class model, one-class classification, binary classification.

1. GİRİŞ

Nesne konum bulma bir sayısal imgede genel nesne sınıflarına ait herhangi bir örneğin varlığının tespit edilip, imgedeki konumunun ve ölçeğinin bulunmasını gerektiren bir bilgisayarlı görü uygulamasıdır. Özellikle son zamanlarda güvenlik, robotik, askeri ve ticari uygulama alanlarının artmasıyla birlikte oldukça popüler bir konu haline gelmiştir. Bununla birlikte son on yılda yapılan bir çok önemli gelişmeye rağmen sayısal imgelerdeki nesne konumlarının bulunması işi oldukça güçtür. Bunun en önemli sebebi aynı sınıfa ait veri örneklerinin görünüş, renk, doku, poz gibi yönlerden farklılıklar göstermesidir. İnsanlar, kediler, sandalyeler gibi bir çok doğal nesne grupları esnek deformasyonlar içerir ve farklı görüş açısından çekilen imgelerde benzer nesneler oldukça farklı görünürler. Ayrıca bu zorluklara ilave olarak ölçek ve ışık farklılıkları, karmaşık arka-plan, örtüşme ve kenarlardan kesime uğramış nesne görüntüleri konum bulma problemini daha da zorlaştıran önemli faktörlerdendir.

Nesne konum bulma sistemlerinin başarısını etkileyen başlıca iki önemli faktör vardır: Örnekleri betimlemek için kullanılan öznitelikler ve konum bulma işlemini gerçekleştiren öğrenme algoritması. Betimleme ile ilgili olarak, iyi bir nesne konum bulma algoritmasının karmaşık sahneleri içeren imgelerde sınıf-içi değişimlerden az etkilenecek ve sınıflar arasındaki farklılıkları öne çıkaracak betimleme teknikleri kullanması gerekmektedir. Bu şartları sağlayan imge betimlerini kullanan sınıflandırıcıların işleri daha da kolaylaşacaktır. Bu amaçla ilk önerilen konum bulma algoritmaları gri-seviye değerlerini [1], dalgacık dönüşümünü [2], kenarları [3], yada Gabor filtrelerinin çıktılarını [4] kullanmışlardır. Son zamanlarda ise gerek nesne sınıflandırmada gerekse nesne konum bulma yöntemlerinde histogram tabanlı öznitelikler gerek başarıları gerekse de verimliliği ile ön plana çıkmışlardır. Bunların çoğu imge gradyanlarının açılarını kullanmaktadırlar. Örnek olarak Scale Invariant Feature Transform (SIFT) [5], SURF [6], Histograms of Oriented Gradients (HOG) [7], PHOG [8], genel şekil bağlamı (Generalized Shape Context) [9] ve bölgesel kenar açı histogramları (Local Edge Orientation Histograms) [10] sayılabilir. Diğer öznitelikler ise bölgesel gri-seviye farklıklarını kullanırlar ve bu sebeple genel şekil yerine daha çok nesnelerin dokusal özelliklerini ön plana çıkarırlar. Bunların arasında Yerel İkili Örüntüler (Local Binary Patterns - LBP) [11,12] ve Yerel Üçlü Örüntüler (Local Ternary Patterns - LTP) [13] ön plana çıkmaktadır. Yerel betimleme teknikleri ile elde edilen öznitelikler, nesnelerin karakteristik özelliklerini oluşturan yerel biçimleri iyi bir şekilde

modelleyebilmekte ve ayrıca belirli bir dereceye kadar dönme ve öteleme etkilerini bastırabilmektedirler. En iyi öznitelikler genellikle eldeki uygulamaya bağlı olup yeni öznitelik çıkarma yöntemleri sıklıkla önerilmektedir. Literatürde en iyi sonuçları veren konum bulma algoritmaları genellikle birden fazla özniteliği bir arada kullanmaktadır. Bu amaçla [12, 14, 15] gibi çalışmalarda farklı öznitelikler uygun bir normalizasyondan sonra arka-arkaya bağlanmak suretiyle birleştirilirken, [8, 16] gibi çalışmalarda yazarlar en iyi kombinasyon katsayılarını, öğrenme aşamasında bir en iyileme algoritması kullanarak öğrenmişlerdir. Bazı yazarlar ise bu öznitelikler arasından daha küçük alt kümeleri boosting gibi öğrenme teknikleri kullanarak seçmişlerdir [17,18].

Birçok nesne konum bulma sistemi, imgelere ait özniteliklerin belirlenmesinden sonra, probleme nesnelere ait örnekleri içeren veri sınıfını, söz konusu nesne haricinde imgelerden oluşan arka-plan veri sınıfından ayırt etmeyi amaçlayan iki-sınıflı bir sınıflandırıcı tasarlama problemi olarak yaklaşır. Bu amaçla en yakın komşu sınıflandırıcılarından, yapay sinir ağlarına, olasılıksal yöntemlerden sınıflandırıcı ağaçlarına kadar bir çok sınıflandırıcı kullanılmıştır. Fakat özellikle iki yöntem son zamanlarda yüksek başarımı ve verimiliği ile ön plana çıkmıştır: Destek Vektör Makineleri (DVM) ve Boosting yaklaşımını kullanan ardışıl sınıflandırıcılar. Viola ve Jones [17] yüz sezme için dalgacık dönüşümüne benzer öznitelikler ve AdaBoost tekniğini kullanarak ardışıl bir sınıflandırıcı dizayn etmiştir. Bu sınıflandırıcının ön katmanlarındaki basit sınıflandırıcılar etkili bir şekilde arka-plana ait örnekleri reddederek sadece yüzleri içeren örnekleri ve zor arka-plan pencerelerini geçirmişlerdir. Son katmanlarda ise daha güçlü sınıflandırıcılar kullanılarak yüz resimleri karmaşık arka-plan resimlerinden ayırt edilmiştir. Böylece gerçek zamanlı çalışan bir yüz sezme algoritması geliştirilmiştir. AdaBoost tabanlı ardışıl sınıflandırıcı yüz sezme için çok iyi sonuçlar versede, daha genel nesne kategorileri için Destek Vektör Makineleri daha çok tercih edilmektedir [7, 8, 14, 19, 20]. Doğrusal Destek Vektör Makineleri genellikle hızı ve kolaylığı yönünden tercih edilmekle birlikte birçok çalışmada gösterildiği üzere doğrusal olmayan Destek Vektör Makineleri çok daha başarılı sonuçlar vermektedir [8], fakat bu sınıflandırıcıların hızları doğrusal olanlarına göre oldukça düşüktür. Bu sebeple, nesne konum bulma uygulamalarında Destek Vektör Makineleri de ilk katmanında doğrusal DVM ve bir sonraki katmanında ise doğrusal olmayan DVM'nin olduğu ardışıl bir sınıflandırıcı olarak kullanılmaktadır [8,14].

İki sınıflı sınıflandırıcılara ek olarak bizler [22] ve Jin ve diğerleri [21] sayısal imgelerde nesne konum bulma için iki sınıflı sınıflandırıcılar yerine tek sınıflı sınıflandırıcılar kullanmaya başlamışlardır. Burda temel amaç nesne sınıfına odaklanmak ve nesne sınıfına ait

örneklerin giriş uzayında kapladığı bölgeyi doğru bir şekilde yakınsamaktır. Konum bulma sırasında her bir imge penceresi nesne sınıfını yakınsamak için kullanılan modele olan uzaklığa göre nesne yada arka-plan sınıfına atanır. Bu sebeple Jin ve diğerleri [21] doğrusal olmayan hiper-küreleri nesne sınıflarını modellemek için kullanmışlardır. Bu yaklaşım kullanılarak nesne sınıfları doğru bir şekilde modellenmesine karşın, bu yaklaşım hesap yükünün fazla olması sebebiyle gerçek zamanlı uygulamalar için uygun değildir (Doğrusal olmayan yaklaşımlarda örneklerin modele olan uzaklığını hesaplamak için sınıflandırıcının döndürdüğü destek vektörleri ile kernel fonksiyonların gerçeklenmesi gerekir, bu işlem destek vektör sayısının fazla olduğu durumlarda oldukça yavaş olacaktır) Bu işlemi hızlandırmak için yazarlar her bir imge penceresini 9 bloğa bölmüşler ve gözlerin olduğu bölgelerin yanaklardan yada burun bölgesinden daha koyu olması gibi pratik kurallar kullanılarak bir dizi test uygulamışlar ve doğrusal olmayan sınıflandırıcı tüm bu testleri geçen pencerelere uygulanmıştır. Bu sebeple bu yöntem sadece yüz sezme için kullanılabilmektedir. Son olarak biz bu proje kapsamında [22]'deki çalışmamızda, nesne konum bulma amacıyla doğrusal hiper-düzlemleri ve doğrusal/kernel hiper-küreleri kullanan ardışıl bir sınıflandırıcı sistemi geliştirdik.

Test imgelerindeki nesnelerin konumlarınının belirlenmesi sırasında, eğitilen sınıflandırıcılar imge üzerinden sistematik olarak seçilen farklı boyutlardaki pencerelere uygulanır ve sınıflandırıcının verdiği çıktıya göre nesnelerin imgede olabileceği yerler tespit edilmeye çalışılır. İmge üzerinde sınıflandırıcının uygulanacağı bölgesel pencereler seçilirken genellikle üç yol izlenir: İlkinde imge üzerinde değişik ölçeklerde sabit boyutlu bir pencere kaydırılır (kayan pencereler yöntemi) ve bu şekilde tüm imge taranır. Her bir kayan pencere için bu pencereye ait öznitelik vektörü oluşturularak sınıflandırıcıya verilir, ve sınıflandırıcı da pencereyi nesne yada arka-plan sınıflarından birine atar. Bu yöntemde sınıflandırıcıya gönderilen pencere sayısı onbinlerle ifade edilmektedir ve bu sebeple kayan pencereler yöntemini kullanan konum bulma algoritması oldukça yavaş olabilir. Sistemin hızlı olabilmesi için hem öznitelik çıkarımının hem de sınıflandırıcının çok hızlı olması gerekmektedir. Bu sebeple literatürdeki en başarılı yöntemler öznitelik çıkarımı için integral images [17], integral histograms [23], distributive histograms [24], yada diğer hızlı histogram tabanlı kayan pencere yöntemi [25] gibi son derece etkili ve hızlı yöntemler kullanmaktadırlar. Bölgesel pencereler seçilirken ikinci bir yöntem olarak, ilk önce imge üzerinde belirgin noktaları bulmak için Harris Laplace ve Difference of Gaussians gibi imgelerdeki belirgin noktaları bulan detektör algoritmaları uygulanır ve sınıflandırıcılar bu

algoritmaların döndürdüğü noktalar etrafından seçilen pencerelere uygulanır. Son olarak [26]'da yazarlar imge içinde nesneleri arama işini hızlandırmak için "branch and bound" yöntemini adapte etmişlerdir. Tüm bu yöntemler arasında kayan pencereler yöntemi diğer yöntemlere göre çok daha iyi sonuçlar verdiğinden, son zamanlarda yüksek başarımları ile ön plana çıkan tüm nesne konum bulma sistemleri kayan pencereleri kullanmaktadır. Kayan pencerelerde onbinlerce pencerenin sınıflandırılması gerektiğinden bu işlem, sınıflandırıcıların ardışıl (cascade structure) olarak kullanılması gerekliliğini ortaya çıkarmıştır. Bu sebeple ardışıl olarak kullanılan sınıflandırıcıların ilk aşamalarında daha çok fazla hesap yükü gerektirmeyen hızlı sınıflandırıcılar seçilir. Bu amaçla seçilen sınıflandırıcıların amacı nesne örneklerine ait olan tüm pencereleri geçirerek, arka-plana ait görüntülerin birçoğunu elemektir. Bu sınıflandırıcıları geçen pencereler daha sonraki aşamada sınıflandırma başarımı çok daha yüksek, fakat daha fazla hesap yükü gerektiren sınıflandırıcılar tarafından sınıflandırılarak nesne sınıfına ait olup olmadığı kesin olarak belirlenir. Örnek olarak, nesne konum bulmada en iyi yöntemler olarak kabul edilen Harzallah ve arkadaşları [14] ve Vedaldi ve arkadaşları [8] tarafından geliştirilen sistemlerde ardışıl sınıflandırıcıların ilk aşamasında hızı sebebiyle doğrusal Destek Vektör Makineleri tercih edilmiş, ardışıl sınıflandırıcının son aşamasında ise başarımı yüksek fakat oldukça yavaş doğrusal olmayan Destek Vektör Makineleri kullanılmıştır. Son aşamadaki sınıflandırıcı sadece ilk aşamaları geçen az sayıda pencereye uygulandığından bu tür konum bulma sistemlerinin hızı çok kötü etkilenmemektedir.

Son zamanlarda yapılan çalışmalarda nesne konum bulmada nesnelere ait parçaların kullanımının başarımı önemli ölçüde arttırdığı gözlenmiştir. Bunun başlıca nedeni ise parça-temelli yöntemlerin örtüşmeye yada esnek deformasyonlara karşı daha gürbüz olmasıdır. Nesne paraçalarını kullanan yöntemler genel itibariyle iki kategori altında toplanabilir: "bottom-up" ve "top-down" yaklaşımlar [27]. "Bottom-up" yaklaşımlarda nesne parçalardan ve görsel fragmanlardan oluşan bir bütün olarak algılanır. İlk olarak nesneye ait olabilecek parçalar bulunur ve daha sonra parçaların geometrik ve istatiksel bilgileri kullanılarak nesnenin imgedeki konumu kestirilmeye çalışılır. Parçaları anlamlı bir şekilde birleştirmek amacıyla bir çok yöntem ve yapı kullanılmaktadır. Bunların arasında birleşik Gaussian dağılımı (joint Gaussian distribution) [28], ağaç yapısı (tree structure) [29], yıldız yapısı (star structure) [30] ve parçalar arası ikili ilişkiler [28] öne çıkan yöntemlerdir. "Top-down" yaklaşımlarda ise nesne (bir bütün olarak) ve parçaları ayrı ayrı aranır ve bu iki bilgi birleştirilerek nesnenin konumu belirlenir [8, 31, 32]. Bu tür yöntemlerde nesneler genellikle

sıradüzensel parçaların birleşimi olarak algılanır ve konum belirlemede hem parçaların hem de nesneyi içeren pencelerin öznitelikleri kullanılır. Örnek olarak Felzenszwalb ve diğerleri [31] konum bulma amacıyla kök (roots) ve parça (parts) seziciler eğiterek bunları imgeye ayrı ayrı ugulamışlar ve son olarak bu iki sezicinin çıktılarını birleştirerek nesnenin konumunu belirlemişlerdir. Bu yaklaşımın 3-aşamalı değişik bir versiyonu da [32]'de önerilmiştir.

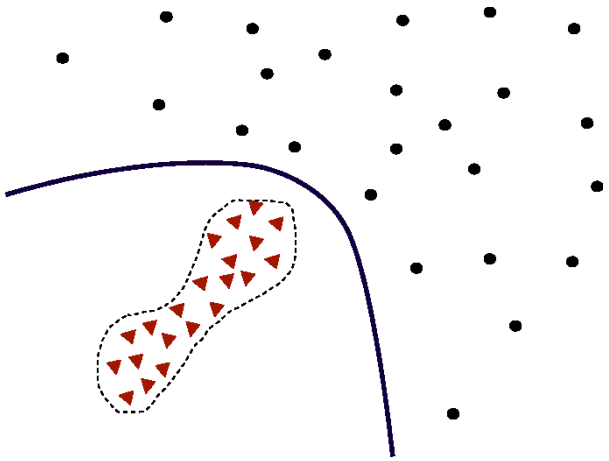
Konum bulma sistemlerinin performansını büyük ölçüde etkileyen diğer bir faktör ise sistemin eğitilme şeklidir. Buna genellikle literatürde çok fazla önem verilmemişse de yaptığımız çalışmalarda sistemin eğitilme şeklinin performansı çok önemli ölçüde etkilediği görülmüştür. Bu amaçla Felzenszwalb ve diğerleri [31] saklı eğitim (latent training) adı verilen yöntemi geliştirmişlerdir. Bu yöntemde nesne örneklerinin imgedeki gerçek konumlarını gösteren kutuların pozisyonları saklı değişkenler olarak ele alınmıştır. İlk olarak sınıflandırıcılar elle işaretlenen nesne konumları kullanılarak eğitilirler. Daha sonra eğitilen konum bulma sistemi her bir nesne örneği etrafında farklı pozisyonlarda ve ölçeklerde nesneye ait örneği arar. En yüksek skoru veren pozisyon nesne örneğinin gerçek pozisyonu olarak kaydedilir ve sistemi tekrar eğitmek için bu pozisyonlar kullanılır. Daha sonra içinde nesneye ait örneklerin olmadığı resimler kullanılarak zor arka-plan görüntüleri olarak adlandırılan ve sistemin yanlışlıkla nesne olarak döndürdüğü örnekler toplanır ve bu örnekler sistemin eğitilmesi için kullanılır. Bu işlem bir kaç defa tekrarlanır ve konum bulma algoritmasının son şekli verilir. Bu yaklaşımın bir çok yararı vardır: İlk olarak bu yaklaşım farklı insanlar tarafından yanlış yada tutarsız olarak etiketlenen nesne konumlarına karşı daha gürbüzdür. İkinci olarak örnekler arasındaki farklılıkları en aza indirgeyerek daha keskin nesne şablonlarının elde edilmesine olanak sağlar. Son olarak, nesneye ait parçalarda kullanıldığında bu yöntemde parçaların konumlarının etiketlenmesine gerek duyulmaz.

2. YÖNTEM

Görsel nesne konum bulmada nesne sınıfına ait olmayan tüm örnekler arka-plan sınıfına dahildir ve bu sebeple giriş öznitelik uzayında nesneye ait öznitelik vektörleri arka-plan örnekleri arasında uzanan bazı belirli bölgelerde bulunurlar. Günlük hayattan kareleri yansıtan sayısal imgelerde arka-plan oldukça karmaşıktır ve büyük farklılıklar gösterebilir. Sayısal imgelerde nesneyi düzgün bir şekilde içermeyen herhangi bir pencere arka-plan sınıfına dahil edildiğinden, ikili sınıflandırıcıların başarılı bir şekilde eğitilmesi için arka-planı iyi bir şekilde temsil eden çok sayıda örneğe ihtiyaç duyulur. Bu da yöntemlerin eğitimi sırasında neden arka-plandan bir kaç defa zor örnekler toplanılması gereğini açıklamaktadır. Fakat bu şekilde oluşturulan eğitim setindeki arka-plan örnek sayısı nesneye ait örneklerin sayısından çok fazla olduğu için (unbalanced training set) ikili sınıflar birçok örneği arka-plan sınıfı olarak sınıflandırma eğilimi gösterecektir. Bu sebeple özellikle ardışıl sınıflandırıcıların ilk katmanlarında bu yaklaşımı kullanmak yerine tek sınıflı sınıflandırıcıları kullanmak daha verimli ve doğru bir tercih olacaktır. Bu durum Şekil 2.1'de gösterilmiştir. Şekilde görüldüğü üzere, söz konusu nesneyi temsil eden örneklerin giriş uzayı içinde gömülü olduğu bölgeler arka-plandan gelen örneklerin azlığından ve bu örneklerin doğru bölgelerde seçilememesinden dolayı iki sınıflı sınıflandırıcı yaklaşımıyla doğru olarak bulunamamıştır. Bu durumda arka-plana ait birçok örnek yanlış olarak nesneye ait görüntüler olarak sınıflandırılacak (false positives) bu da sistemin güvenilirliğini azaltacaktır. Bizce bu sorunu çözmek için araştırmacılar arka-plan verilerini modellemeden ziyade ilginin büyük bölümünü nesne örneklerinin giriş uzayında gömülü olduğu bölgelerin kestirimi, yani nesne sınıfının topolojik yapısını modelleme üzerinde toplamalıdırlar. Başka bir deyişle sınıflandırma problemi iki sınıflı sınıflandırma yerine tek sınıflı sınıflandırıcı dizayn etme problemi olarak ele alınmalıdır. Nesnelerin topolojik yapısı uygun modellerle modellendikten sonra, bir test örneğinin bu geometrik modellere olan uzaklığına bakılarak o örneğin nesne sınıfına ait olup olmadığına karar verilebilir. Şekil 2.1'de görüldüğü gibi nesnelere ait örneklerin gömülü olduğu bölgeler doğru bir şekilde modellenebilirse, nesnelerin konumlarının belirlenmesi için daha doğru karar sınırları bulunabilir.

Biz bu çalışmada nesne konum bulma için doğrusal DVM sınıflandırıcısı ile birlikte 4 katmandan oluşan ardışıl bir sınıflandırıcı sistemi geliştirdik. Ardışıl sınıflandırıcının ilk katmanında iki sınıflı bir sınıflandırıcı (doğrusal DVM sınıflandırıcısı) kullanılırken diğer 3 katman tek sınıflı sınıflandırıcılardan oluşmaktadır. Ardışıl tek sınıflı sınıflandırıcının ilk katmanında hızlı bir sınıflandırıcı olan ve nesneye ait sınıfı bir hiper-düzlem ile modelleyen

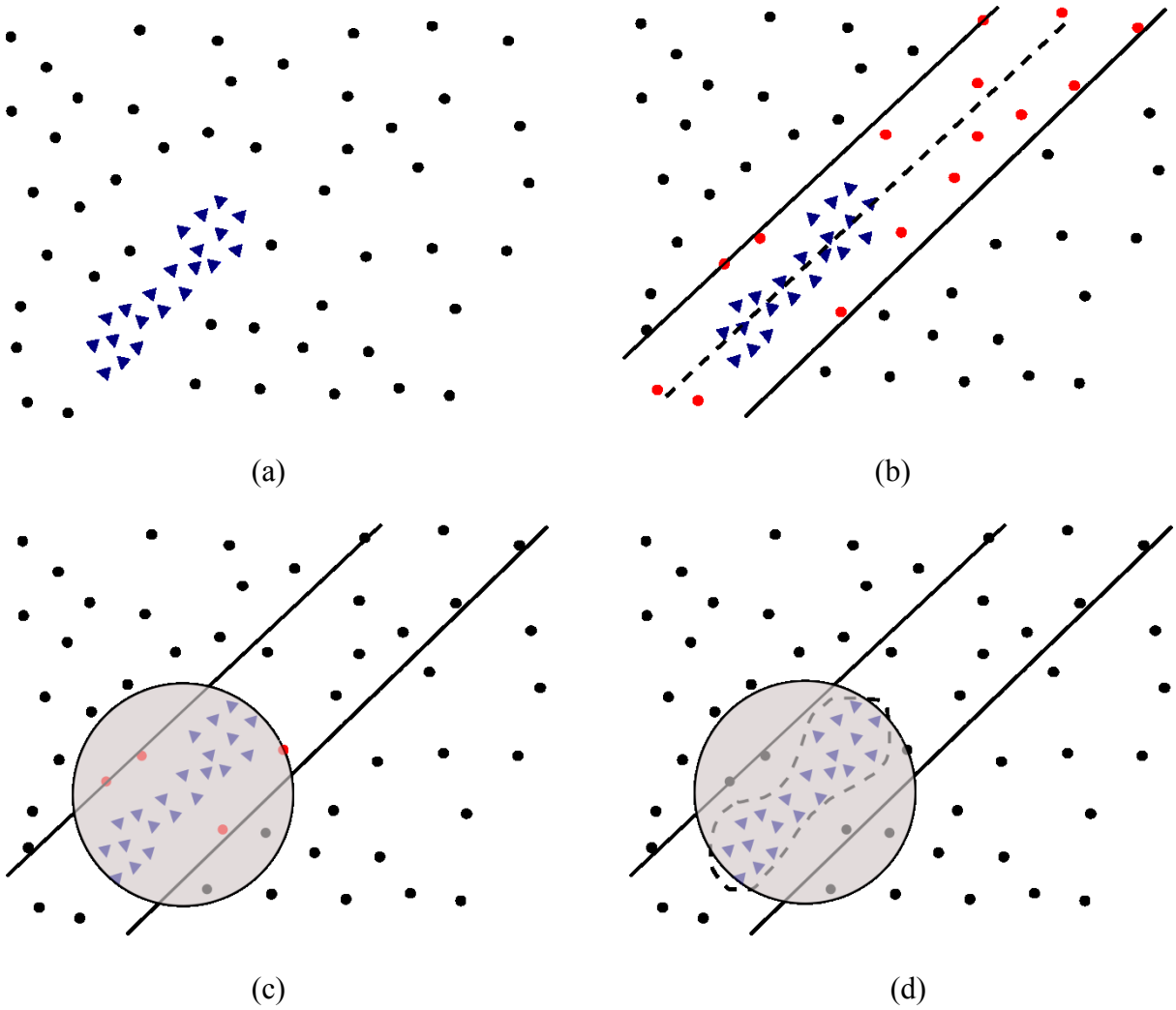
bir sınıflandırıcı kullandık. Bu katmanın amacı nesneye ait hemen hemen tüm örnekleri geçirerek, arka-plana ait örneklerin bir çoğunu elemektir. İkinci aşamasında ise nesne sınıfını doğrusal bir hiper-küre ile modelleyen sınıflandırıcı kullanılmıştır. Bu sınıflandırıcı da oldukça hızlı olup hiper-düzlem sınıflandırıcısı ile uyumlu bir şekilde çalışarak ilk katmandan geçen bir çok arka-plan görüntüsünü Şekil 2.2'de görüldüğü gibi elemektedir. Son katmanda ise doğrusal olmayan hiper-küre sınıflandırıcısı kullanılmıştır. Bu sınıflandırıcı hız bakımından diğer iki katmandaki sınıflandırıcılara oranla çok daha yavaştır. Fakat her iki katmandaki sınıflandırıcıyı da geçen az sayıda örneğe uygulanmaktadır.



Şekil 2.1. Kesik çizgiyle gösterilen sınırlar üçgen sembolle ifade edilen nesne sınıfının topolojisini belirlemektedir. Bu bölgelerin yakınlarına düşen örneklerin nesne sınıfına ait olması beklenir. Fakat, iki sınıflı bir yaklaşım kullanıldığında arka-plan sınıfına ait örneklerin azlığı ve uygun yerlerden seçilmemiş olması sebebiyle iki sınıflı sınıflandırıcının bulduğu karar sınırları (koyu renkle belirtilen karar sınırı) doğru olmayabilir.

Sınıfları modellemek için bu çalışmada hiper-düzlemler ve hiper-küreler kullanılmıştır. Bununla beraber proje başvurumuzda belirttiğimiz üzere sınıfları yakınsamak için affine altuzaylar, konveks zarflar, hiper-küreler ve hiper elipsler gibi konveks sınıf modelleri de kullanılabilir. Fakat tüm bu modellerin hesap karmaşası daha yüksek olduğu için gerçek zamanlı konum bulma uygulamaları için çok uygun olmadıkları gözlenmiştir.

Önerdiğimiz ardışıl sınıflandırıcının performansını ve verimliliğini karşılaştırmak için ayrıca iki sınıflı bir sınıflandırıcı olan Destek Vektör Makineleri de kullanılmıştır. Doğrusal ve doğrusal olmayan Destek Vektör Makineleri ile önerdiğimiz tek sınıflı sınıflandırıcılar kullanılarak değişik ardışıl sınıflandırıcılar oluşturularak bu sistemler hız ve doğruluk bakımından karşılaştırılmıştır.



Şekil 2.2. Önerilen ardışıl tek sınıflı sınıflandırıcılar: (a) Nesne sınıfına ve arka-plana ait temsili örnekler. Nesne sınıfına ait örnekler mavi renkte üçgenlerle arka-plan ise siyah renkli çemberlerle gösterilmiştir. (b) Birinci katmanda uygulanan doğrusal hiper-düzlem sınıflandırıcısı çıktısı. Yanlış olarak nesne sınıfına atanan arka-plan görüntüleri (false positives) kırmızı renk ile gösterilmiştir. (c) İkinci katmandaki doğrusal hiper-küre sınıflandırıcısı birinci katmanı geçen bir çok arka-plan örneğini elemiştir. (d) Son katmanda kullanılan doğrusal olmayan hiper-küre sınıflandırıcısı nesne sınıfını çok doğru bir şekilde modeller ve her iki katmandan geçen yanlış örnekleri eleyerek en son kararları verir.

Aşağıda ilk olarak görüntüleri betimlemek için kullanıldığımız öznitelikler açıklanmış ve ardından ardışıl sınıflandırıcılarda kullandığımız tek ve iki sınıflı sınıflandırıcılar ayrıntıları ile açıklanmıştır.

2.1 İMGE BETİMLEME

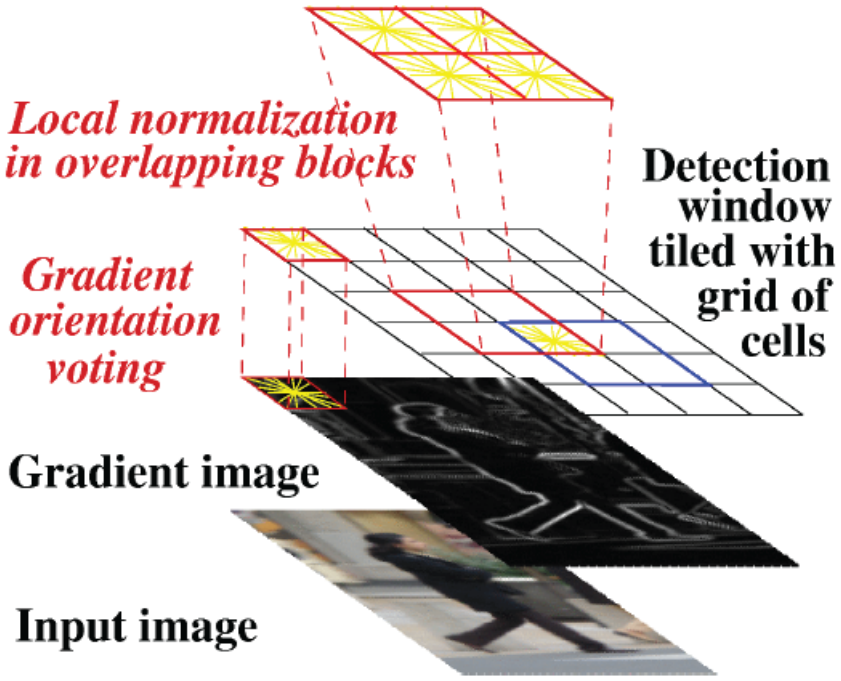
İmge betimleme amacıyla Yönlü Gradyan Histogramları - YGH (Histograms of Oriented Gradients) ile beraber Yerel İkili Örüntü - YİÖ (Local Binary Patterns) öznitelikleri kullanılmıştır. YGH öznitelikleri nesnelerin genel görünüşlerini betimlerken, YİÖ ise

nesnelerin doku (texture) bilgilerini betimlediğinden bu iki öznitelik birbirilerini tamamlar niteliktedir. YGH ve YİÖ öznitelikleri birlikte kullanılırken, bu öznitelikler normalizasyon uygulanarak arka-arkaya bağlanmışlardır.

2.1.1 Yönlü Gradyan Histogramları

Yönlü Gradyan Histogramları (YGH) son zamanlarda nesne konum bulmada en fazla kullanılan çok başarılı bir betimleme biçimidir [7]. SIFT gibi YGH'da yerel imge görüntülerinin, bu bölgelere ait kenar yönlerinin yada gri-seviye gradyanlarının yerel dağılımları kullanılarak etkili bir şekilde kodlanabildiği varsayımı üzerine kuruludur. Çalışmalarımızda [7]'de verilen temel yöntemin üzerine [31]'de yapılan değişiklikler kullanılarak elde edilen YGH öznitelikleri kullanılmıştır.

YGH öznitelikleri bulunurken izlenen temel yöntem Şekil 2.3'de gösterilmiştir. ilk olarak imgeye ait gradyanlar hesaplanmaktadır. Daha sonra imge hücre (cell) adı verilen ve birbiriyle örtüşmeyen dikdörtgensel bölgelere bölünmüş ve herbir hücreye ait gradyan yönlerinin histogramları hesaplanmıştır. Son olarak belli sayıda hücreler bir araya getirilerek blok adı verilen yapılar oluşturulmuş ve her bir bloktaki hücreye ait histogramlar yerel olarak normalize edilmiştir. Daha sonra bu histogramlar arka-arkaya bağlanmak suretiyle bir imge penceresine ait YGH öznitelikleri elde edilmiştir.

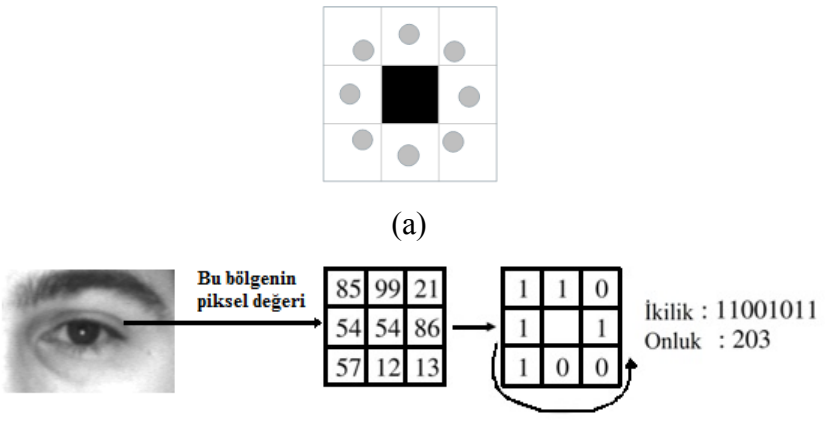


Şekil 2.3. Yönlü gradyan histogramlarının çıkarılması (bu şekil Bill Triggs'e ait sunumdan alınmıştır).

İmge gradyanlarını bulmak için her iki yönde $[-1,0,1]$ filtresi kullanılmıştır. Hücreler insan ve PASCAL VOC veritabanlarındaki nesnelerin konumunu bulma deneylerinde 8×8 pikselden oluşacak şekilde seçilirken yüz sezme deneylerinde 6×6 piksel olarak seçilmişlerdir. Bloklar Şekil 2.3'de görülüdüğü gibi 2×2'lik hücreler kullanılarak oluşturulmuştur. Yön histogramlarının binlerini oluşturmak için $[0^\circ-180^\circ]$ dereceleri arası 9 eşit parçaya bölünmüştür. Bu yol izlenerek her bir hücreden elde edilecek öznitelik sayısı 36 olmaktadır. Felzenszwalb ve diğerleri [31]'de yön histogramalarını elde etmek için 9 binlik işaretsiz (unsigned) yönlerin yanında 18 binlik işaretli ($[0^\circ-360^\circ]$) yönleride kullanarak daha büyük boyutlu histogramlar elde etmişler vu bunlara temel bileşen analizi uygulayarak boyutu 27'ye düşürmüşlerdir. Bu özniteliklere son olarak 4-boyutlu normalizasyon katsayıları da eklenerek her bir hücre için 31-boyutlu öznitelikler elde edilmiştir. Bu çalışmada Felzenszwalb tarafından paylaşıma açılan YGH öznitelik çıkarma kodları kullanılmıştır.

2.1.2 Yerel İkili Örüntüler

Yerel ikili örüntü (YİÖ) herhangi bir piksel için bu noktanın komşuluğunda kurallı ikili gri-seviye değerleri karşılaştırmaları yapmaya dayanan bir yöntemdir [11]. Bu karşılaştırmalardan ikili bir kod elde edilir ve bu kodun onluk değeri o piksele ait öznitelik değeri olarak kullanılır. Merkez piksele komşu piksellerin seçimi farklı ölçeklerde değişiklik göstermesine rağmen en tercih edilen yöntem, komşu pikselleri yarıçapı 1 piksel olan çember üzerinden örneklemektir.



Şekil 2.4. YİÖ'nün elde edilişi.

Şekil 2.4'de yarıçapı 1 ve dairesel komşuluğu 8 nokta olan YİÖ'nün elde edilişi gösterilmektedir. Orta noktadaki referans pikselinin çevresindeki gri-seviye değeri bu

noktanın gri-seviye değerinden büyük veya bu noktanın değerine eşitse 1, bu noktanın değerinden küçükse 0 değeri verilir. Böylelikle piksel gri-seviyeleri farkından oluşan ikili bir kod elde edilir. Daha sonra bu kod onluk sayı sistemine çevrilir. Elde edilen bu değer ışık şiddeti değişimlerinden daha az etkilenmektedir. Bu çalışmada yarıçapı 1 olan 8 nokta komşuluklu (Şekil 2.4. (a)) YİÖ kullanılmıştır. YİÖ öznitelikleri nesne konum bulmada yerel olarak kullanılmıştır. Bu amaçla imge pencereleri birbiriyle örtüşmeyen $m \times n$ alt bölgeye bölünmüş ve her bir bölge için 59 boyutlu YİÖ histogramları elde edilmiştir (59 binden oluşan histogram kodlamasında 58 bin birörnek (uniform) değerler için geriye kalan tek bin ise birörnek olmayan (non-uniform) değerler için kullanılmıştır). Daha sonra bu histogramlar L1 normu kullanılarak normalize edilip arka-arkaya bağlanmıştır. Bu sayede her bir imge penceresi $d = m \times n \times 59$ boyutlu öznitelik vektörü ile betimlenmiştir.

2.2 İKİ SINIFLI SINIFLANDIRICILAR

2.2.1 Destek Vektör Makineleri

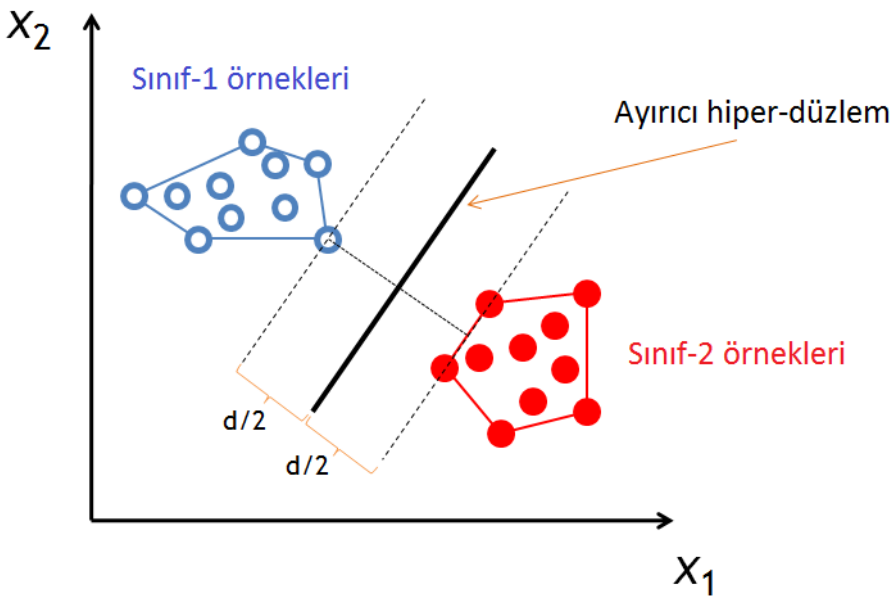
Destek Vektör Makineleri (DVM) ses tanıma, görsel nesne tanıma, teks sınıflandırma gibi zorlu örüntü tanıma problemlerinde oldukça sık kullanılan başarılı bir yöntemdir [38, 39]. Destek Vektör Makineleri iki sınıfla çalışan bir sınıflandırıcı olup, yöntem bu iki sınıfı en iyi ayıran doğrusal bir hiper-düzlem bulur. Ayırıcı hiper-düzlem bulunurken iki sınıf arasında maksimum boşluk (margin) kalıcak şekilde bir hiper-düzlem döndürülür. Şekil 2.5'de doğrusal olarak ayrıştırılabilen iki sınıfı en iyi ayıran hiper-düzlem gösterilmiştir.

Elimizde iki sınıfa ait *n* örnek olduğunu varsayalım,

$$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n), \quad \mathbf{x}_i \in \Re^d, \quad y_i \in \{-1, 1\}. \tag{1}$$

Yukarıdaki eşitlikte $\mathbf{x}_i$ veri örneklerini y_i ise örneklere ait etiket bilgilerini göstermektedir. Sınıflar "positive" (+1) ve "negative" (-1) sınıflar olarak isimlendirilmektedir. Destek Vektör Makinelerinde ayırıcı hiper-düzlemi karakterize eden iki parametre vardır: hiper-düzlemin normali **w** ve ofset parametresi *b*. Ayırıcı hiper-düzlemin üzerinde bulunan veriler $\langle \mathbf{w}, \mathbf{x} \rangle + b = 0$ eşitliğini sağlarlar. Aynı şekilde (+1) sınıfına ait tüm veriler için $\langle \mathbf{w}, \mathbf{x} \rangle + b > 0$, (-1) sınıfına ait veriler içinde $\langle \mathbf{w}, \mathbf{x} \rangle + b < 0$ eşitlikleri sağlanır. Bu durumda yeni bir test örneği aşağıdaki fonksiyonun işaretine bakılarak (+1) yada (-1) sınıflarından birine atanır

$$f(\mathbf{x}_{test}) = \langle \mathbf{w}, \mathbf{x}_{test} \rangle + b. \tag{2}$$



Şekil 2.5. Destek Vektör Makineleri ile iki sınıflı sınıflandırma. Hiper-düzlemin üst kısmında kalan örnekler bir sınıfa ve altında kalan örnekler ise diğer sınıfa atanır.

Doğrusal olarak iki sınıfın ayrıldığı durumlarda bu iki sınıfı en iyi ayıran hiper-düzlemi bulma problemi, aşağıdaki ikilenik programlama (quadratic programming) problemine dönüştürülür

$$\begin{aligned} &\min_{\mathbf{w},b} \frac{1}{2} \| \mathbf{w} \|^2 + C \sum_{i=1}^{n} \xi_i \\ &\text{s.t.} \quad y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1 + \xi_i \geq 0, \qquad \xi_i \geq 0. \end{aligned} \tag{3}$$

Yukarıdaki eşitlikte ξ_i'ler kısıtları sağlamayan örneklerle ilişkilendirilmiş arttıran yapay değişkenler (slack variables) olup, C terimi ise kullanıcının girdiği hata ceza terimidir (error penalty term). Bu problemi çözmek yerine genellikle problemin aşağıda verilen dual eşleniği çözülür.

$$\begin{aligned} &\min_{\alpha} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_{i=1}^{n} \alpha_i \\ &\text{s.t.} \quad \sum_{i=1}^{n} y_i \alpha_i = 0, \qquad 0 \leq \alpha_i \leq C \end{aligned} \tag{4}$$

Yukarıdaki eşitlikte α_i'ler bulunması gereken Lagrange katsayılarını ifade etmektedir. Bu optimizasyon problemi konveks olup, global en iyi noktası (global minumum) vardır. En iyi Lagrange katsayılarının belirlenmesinden sonra ayırıcı hiper-düzlemin normali $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$ eşitliği ile bulunur. Lagrange katsayılarının çoğu sıfırdır ve bu sebeple bu katsayılara karşılık gelen örneklerin hiper-düzlem üzerinde etkisi yoktur. Fakat sıfırdan farklı katsayılara karşılık

gelen örnekler hiper-düzlemi oluştururlar ve bu elemanlar destek vektörleri olarak isimlendirilirler. Bu vektörler genellikle iki sınıfın birbirine yakın olduğu bölgelerden gelirler ve iki sınıf arasındaki karar sınırlarını (decision boundaries) karakterize ederler. Hiper-düzleme ait normal belirlendikten sonra ofset parametresi *b* eşitlik (3)'de verilen kısıtlar ve Lagrange katsayısı $0 < \alpha_i < C$ şartını sağlayan herhangi bir destek vektör kullanılarak bulunabilir.

Doğrusal olarak ayrıştırılamayan veriler için doğrusal olmayan Destek Vektör Makineleri, verileri öncelikle farklı ve çok büyük boyutlu bir uzaya eşlemlemekte ve sınıflandırma işlemini bu yeni öznitelik uzayında gerçekleştirmektedir. Genellikle bu öznitelik uzayı çok yüksek boyutludur ve bu nedenle teknik olarak hesaplamalar oldukça zordur. Bu problemi çözmek için çekirdek hilesi kullanılır. Bu hilede kısaca eşitlik (4)'de verilen iki örnek arasındaki iç çarpım $\left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle = \mathbf{x}_i^T \mathbf{x}_j$ kernel fonksiyonu ile değiştirilir, $k(\mathbf{x}_i, \mathbf{x}_j) = \left\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \right\rangle$. Burada $\phi : \Re^d \to \Im$ fonksiyonu her bir veriyi çok boyutlu uzaya haritalayan fonksiyonu ifade etmektedir. Bu şekilde veri hiç bir zaman gerçekte çok büyük boyutlu uzaya haritalanmadan her işlem kernel fonksiyonları üzerinden gerçekleştirilir. Bu durumda doğrusal olmayan Destek Vektör Makinelerinin karar fonksiyonu çekirdek ile birlikte şu şekilde ifade edilir:

$$f(\mathbf{x}_{test}) = \sum_{i=1}^{n_s} \alpha_{si} y_{si} k(\mathbf{x}_{si}, \mathbf{x}_{test}) + b \tag{5}$$

Bu çalışmada iki çeşit çekirdek fonksiyonu kullanılmıştır:

- Doğrusal çekirdek,

  $k(\mathbf{x}_i, \mathbf{x}_j) = \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle$

- Gaussian tabanlı çekirdek,

  $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \parallel \mathbf{x}_i, \mathbf{x}_j \parallel^2)$.

Doğrusal Destek Vektör Makinelerinde bir örneği sınıflandırmak için sadece basit bir iç çarpım gerektiğinden hız yönünden bu yöntem gerçek zamanlı nesne konum bulma sistemleri için son derece uygundur. Fakat daha öncede belirttiğimiz üzere nesneye ait örnekler arka-plan görüntüleri arasına gömülmüş bir durumdadır. Bu sebeple bu yaklaşım veri sayısının çok fazla olduğu uygulamalar için pek uygun değildir.

2.3 TEK SINIFLI SINIFLANDIRICILAR

2.3.1 Doğrusal Hiper-Düzlem Sınıflandırıcısı

Ardışıl tek sınıflı sınıflandırıcıların ilk katmanında nesneye ait sınıfları modellemek için doğrusal bir hiper-düzlem kullandık. Bunun öncesinde ilk olarak affine zarflar (affine hulls) denenmiştir. Affine zarflar doğruluk açısından olumlu sonuçlar verse de, gerçek zamanlı konum bulmada doğrusal Destek Vektör Makinelerine oranla hız bakımından verimliliği düşük kalmıştır. Bunun sebebi her iki yöntemdeki örneklerin izdüşümlerinin alındığı vektör sayılarının çok farklı olmasıdır. Doğrusal DVM'de test sırasında öznitelik vektörünün tek bir vektör üzerinde izdüşümü alınır (İzdüşüm alınan vektör hiper-düzlemin normalidir). Oysa affine zarflarda izdüşüm vektörlerinin sayıları 50 veya daha fazla olarak bulunduğu için, sistemin gerçek zaman performansı düşmüştür.

Sistemi hızlandırmak amacıyla nesne sınıfına ait örneklerin oluşturduğu affine zarfa en yakın, arka-plan sınıfına ait verilere ise en uzak hiper-düzlemi bulma problemi formüle edilerek çözülmüştür. Örneklerin bu hiper-düzleme olan uzaklığına bakılarak nesne veya arka-plan sınıfına ait olduğu tespit edilebilir. Bu işlem için sadece hiper-düzlemin normali kullanılacağından sistem DVM kadar hızlı olmaktadır.

Nesne sınıfını modellemek için kullanılan hiper-düzlem $\mathbf{w}^T\mathbf{x}+b=0$ eşitliği ile verilsin. Hiper-düzlemi bulurken amacımız nesneye ait sınıfı en iyi şekilde yakınsayan aynı zamanda da arka-plana ait veri örneklerinden olabildiğince uzak bir hiper-düzlem bulmaktır. Hiper-düzlem bulunduktan sonra Şekil 2.2.(b)'de görüldüğü üzere, bir test örneğinin hiper-düzleme olan uzaklığı belirli bir eşik değerinin, τ, üzerinde ise örnek arka-plan sınıfına atanacaktır. Bu durum matematiksel bir ifade ile aşağıdaki gibi yazılabilir.

$$\begin{cases} |\mathbf{w}^T\mathbf{x}_{test}+b| \le \tau, & \text{nesne sinifina } (+1) \text{ ait örnek} \\ |\mathbf{w}^T\mathbf{x}_{test}+b| > \tau, & \text{arka - plana } (-1) \text{ ait örnek} \end{cases}$$

Eşik değeri çapraz sınama ile belirlenebilir.

$\mathbf{X}_+$ ve $\mathbf{X}_-$ matrisleri satırları sırasıyla nesne ve arka-plan sınıflarına ait örnekleri içeren matrisler olsun. Aynı şekilde $\mathbf{e}_+$ ve $\mathbf{e}_-$ 1 sayısını içeren uygun boyutlu sütun vektörleri olsun. Bu durumda $\overline{\mathbf{X}}_+=[\mathbf{X}_+ \quad \mathbf{e}_+]$ ile $\overline{\mathbf{X}}_-=[\mathbf{X}_- \quad \mathbf{e}_-]$ matrisleri tanımlanabilir. Nesne sınıfını en iyi yakınsayan hiper-düzlem en küçük kareler yöntemi kullanılarak aşağıdaki en iyileme problemi ile çözülebilir

$$\min_{\mathbf{w},b,\|\mathbf{w}\|=1} \|\mathbf{X}_+\mathbf{w}+\mathbf{e}_+b\|^2 = \min_{\mathbf{z}} \frac{\mathbf{z}^T\mathbf{G}\mathbf{z}}{\|\mathbf{w}\|^2}. \qquad (6)$$

Bu eşitlikte $\mathbf{z} = \begin{pmatrix} \mathbf{w} \\ b \end{pmatrix}$ ve $\mathbf{G} = \overline{\mathbf{X}}_+^T \overline{\mathbf{X}}_+$ ifade etmektedir. Biz bu yaklaşımı kullanmak yerine nesne sınıfını modelleyen hiper-düzlemi aynı zamanda arka-plan örneklerinden uzaklaştırma amacı güden aşağıdaki en iyileme yöntemini kullandık [34]

$$\min_{\mathbf{w},b,\|\mathbf{w}\|=1} \frac{\| \mathbf{X}_+ \mathbf{w} + \mathbf{e}_+ b \|^2}{\| \mathbf{X}_- \mathbf{w} + \mathbf{e}_- b \|^2 + \delta(\| \mathbf{w} \|^2 + b)} = \min_{\mathbf{z}} \frac{\mathbf{z}^T \mathbf{G} \mathbf{z}}{\mathbf{z}^T \mathbf{H} \mathbf{z}}. \tag{7}$$

Yukarıdaki eşitlikte $\mathbf{H} = \overline{\mathbf{X}}_-^T \overline{\mathbf{X}}_- + \delta \mathbf{I}$ ile gösterilmekte olup, δ parametresi ise kullanıcının belirlediği ve matrisin tersinin alınabilmesi için kullanılan küçük bir tolerans değeridir. Bu eniyileme probleminin çözümü $\mathbf{G}\mathbf{z} = \lambda \mathbf{H}\mathbf{z}$ özdeğer özvektör ayrıştırmasında en küçük özdeğere karşılık gelen özvektördür.

Bu yöntem yüz ve insan konum bulmada iyi çalışmasına rağmen aykırı değerlerin çok olduğu PASCAL VOC veritabanları için iyi sonuçlar vermemiştir. Bu sebeple aykırı değerlerden daha az etkilenen daha gürbüz bir yöntem önerdik. Bu yöntemde nesne sınıfını en iyi yakınsayan ve aynı zamanda arka-plan örneklerinden uzak olan hiper-düzlemi bulma problemi aşağıdaki şekilde formüle edilmiştir

$$\begin{aligned} \min_{\mathbf{w},\xi \geq \mathbf{0}} \ & \frac{1}{2} \| \mathbf{w} \|^2 + C_+ \sum_i (\xi_i + \xi_i^*) + C_- \sum_j \xi_j \\ \text{s.t.} \ & \mathbf{w}^T \mathbf{x}_i + b \leq \Delta + \xi_i, \\ & \mathbf{w}^T \mathbf{x}_i + b \geq -\Delta - \xi_i^*, \\ & \mathbf{w}^T \mathbf{x}_j + b \geq \Delta + 1 - \xi_j, \ i \in I_+, \ j \in I_-. \end{aligned} \tag{8}$$

Bu eşitlikte $C_+ (C_-)$ kullanıcı tarafından girilmesi gereken ve kısıtları bozan örneklerle ilişkilendirilen hata katsayılarını belirleyen parametreleri, $I_+ (I_-)$ sınıflara ait indisleri gösteren kümeleri, Δ ise kullanıcı tarafından 0 ve 1 arasında bir değere atanması gereken parametreyi ifade etmektedir. Bu eniyileme probleminde nesne sınıfına ait örnekler $\mathbf{w}^T \mathbf{x} + b = \Delta$ ve $\mathbf{w}^T \mathbf{x} + b = -\Delta$ ile karakterize edilen iki paralel hiper-düzlem arasında kalırken, arka-plana ait örnekler ise $\mathbf{w}^T \mathbf{x} + b = 1 + \Delta$ hiper-düzleminin üstünde kalmaktadır. Nesne ve arka-plana ait örnekler arasındaki uzaklık en azından $1/\| \mathbf{w} \|$ kadar olacaktır. Kısıtların sağlanamadığı durumlar için ξ_i, ξ_i^*, ξ_j arttıran yapay değişkenleri tanımlanmıştır. Bu problem de konveks bir problem olup global en iyi noktası vardır. Bu yaklaşımdan daha iyi bir yaklaşım arka-plana ait örneklerin ayırıcı hiper-düzlemin her iki tarafında da kalabilecek şekilde yayılmasına izin vermektir (Bu da (8)'de verilen eniyileme problemindeki en son

konveks kısıtın, $\mathbf{w}^T\mathbf{x}_j + b \geq \Delta + 1 - \xi_j$, konveks olmayan $|\mathbf{w}^T\mathbf{x}_j + b| \geq \Delta + 1 - \xi_j$ kısıtı ile değiştirilmesi ile elde edilebilir). Fakat bu problem konveks olmadığından özellikle veri sayısının çok fazla olduğu durumlarda çözümü çok zor olacaktır. Önerdiğimiz yöntem (8), bu eksikliğe rağmen veri uzay boyutunun da büyük olması sebebiyle (7)'de verilen yöntemden çok daha başarılı sonuçlar vermiştir.

İlk olarak [22]'deki çalışmamızda konum bulma amacıyla önerdiğimiz nesne sınıfını en iyi şekilde yakınsayan aynı zamanda arka-plan örneklerinden olabildiğince uzak hiper-düzlemi bulma fikri, [40]'da yapılan çalışmada da benimsenmiş ve bu fikir eğitim setindeki tüm sınıfların verilmediği ve açık küme (open set) olarak adlandırılan daha genel örüntü tanıma problemleri için adapte edilmiştir. Yazarlar ayrıca bizim bu projede savunduğumuz gibi nesne konum bulma problemi için iki sınıf sınıflandırıcılar yerine tek sınıflı sınıflandırıcıların daha uygun olduğu tezini savunmuşlardır. Yaptıkları çalışmada bizim yöntemimizden farklı olarak [41]'de önerilen tek sınıflı sınıflandırıcı yada DVM sınıflandırıcısının döndürdüğü hiper-düzleme paralel bir hiper-düzlem alarak, sınama amaçlı ayrılmış örnekleri kullanarak bu hiper-düzlemleri aşağı yada yukarı hareket ettirerek Şekil 2.2. (b)'de gösterildiği gibi söz konusu sınıfın örneklerini bu iki paralel hiper-düzlem arasında arka-plana ait örnekleri ise bu hiper-düzlemlerin üstünde yada altında kalacak şekilde ayarlamaya çalışmışlardır.

Özetlenecek olursa nesneya ait örnekleri en iyi yakınsayan ve aynı zamanda da arka-plan örneklerinden olabildiğince uzak olan hiper-düzlemi bulmak için çeşitli yöntemler kullanılabilir. Fakat bu yöntemlerin içinde en başarılı hiper-düzlem kusursuz (8)'de verilen problemde konveks olmayan kısıtın eklenmesiyle elde edilecek problemin çözümünden elde edilen hiper-düzlem olacaktır. Konveks olmayan bu problemi büyük veri tabanları için sorunsuz bir şekilde çözecek algoritma literatürde büyük ses getirebilme potansiyeline sahiptir.

2.3.2 Doğrusal Hiper-küre Sınıflandırıcısı

Ardışıl tek sınıflı sınıflandırıcıların ikinci katmanında nesne sınıfını yakınsamak amacıyla doğrusal bir hiper-küre kullanılmıştır. Bir sınıfa ait tüm örnekleri içeren en küçük ve kompakt hiper-küre kavramı Tax ve Duin [33] tarafından önerilmiş ve aykırı değer sezimi için başarı ile kullanılmıştır. Şekil 2.2'de gösterildiği üzere hiper-küre sınıflandırıcısı kendinden önce gelen hiper-düzlem sınıflandırıcısı ile birbirlerini tamamlayacak şekilde çalışarak, hiper-düzlemin yanlışlıkla nesne sınıfına atadığı arka-palana ait örnekleri başarılı bir şekilde elemektedir.

Bir sınıfa ait *d*-boyutlu uzaydaki veri örneklerini $\{\mathbf{x}_i \in R^d\}_{i=1,\ldots,n}$ ile gösterelim. Bu sınıfa ait örnekleri içeren en küçük ve kompakt hiper-küreyi temsil eden iki parametre vardır: kürenin merkezi **c** ve kürenin yarıçapı *r*. Hiper-kürenin merkezi ve yarıçapı veri örneklerinin giriş uzayındaki konumlarına göre değişir ve aşağıdaki ikilenik programlama (quadratic programming) probleminin çözülmesiyle bulunurlar.

$$\begin{aligned} &\min_{\mathbf{c}, r\geq 0, \xi\geq 0} && r^2 + \gamma \sum_{i=1}^{n} \xi_i \\ &s.t. && \| \mathbf{x}_i - \mathbf{c} \|^2 \leq r^2 + \xi_i. \end{aligned} \tag{9}$$

Genellikle bu problemi çözmek yerine bu problemin aşağıda verilen çifteşleniği (dual) çözülür

$$\begin{aligned} &\min_{\alpha} && \sum_{i,j} \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_i \alpha_i \| \mathbf{x}_i \|^2 \\ &s.t. && \sum_{i=1}^{n} \alpha_i = 1, \quad 0 \leq \alpha_i \leq \gamma, \quad i = 1,\ldots,n. \end{aligned} \tag{10}$$

Yukarıdaki formülde α_i'ler Lagrange katsayıları olup $\gamma \in [0,1]$ ise kullanıcının belirlediği ve kompakt modele çok uzak olan aykırı değerleri belirlemede kullanılan bir katsayıdır. Bu ikilenik eniyileme problemi de konveks bir problem olup, global bir en iyi noktası vardır. Problemin çözümünü veren optimal α_i katsayıları belirlendikten sonra hiper-kürenin merkezi $\mathbf{c} = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i$ formülü ile yarıçapı ise $0 < \alpha_i < \gamma$ şartını sağlayan Lagrange katsayısına karşılık gelen herhangi bir $\mathbf{x}_i$ örneğinin hiper-kürenin merkezine olan uzaklığına bakılarak bulunur, $r = \| \mathbf{x}_i - \mathbf{c} \|$.

Eğer elimizde nesne sınıfına ait olmayan örnekler (bizim problemimizde arka-plana ait örnekler) varsa bu örneklerden yararlanılarak daha iyi hiper-küre modelleri tanımlanabilir. Bu durumda amaç nesne sınıfına ait örnekleri içeren ve nesne sınıfına ait olmayan örnekleri dışında bırakan bir hiper-küre bulmaktır. Elimizde nesne sınıfına ait n_1 adet i ve j indisleriyle gösterilen örneklerle birlikte arka-plana ait n_2 adet l ve m indisleriyle belirtilen örnekler olduğunu varsayalım. Nesneye ait örnekleri içeren ve arka-plana ait örnekleri dışarıda bırakan en kompakt hiper-küre bulma problemi aşağıdaki ikilenik en iyileme problemi şeklinde ifade edilebilir

$$\begin{aligned} \min_{\mathbf{c}, r\geq 0, \xi\geq 0} \quad & r^2 + \gamma_1 \sum_i \xi_i + \gamma_2 \sum_l \xi_l \\ s.t. \quad & \| \mathbf{x}_i - \mathbf{c} \|^2 \leq r^2 + \xi_i, \quad i = 1,...,n_1 \\ & \| \mathbf{x}_l - \mathbf{c} \|^2 \geq r^2 - \xi_l, \quad l = 1,...,n_2. \end{aligned} \tag{11}$$

Daha önceki durumda olduğu gibi bu problemi çözmek yerine genellikle problemin aşağıda verilen çifteşleniği çözülür.

$$\begin{aligned} \min_{\alpha} \quad & \sum_{i,j} \alpha_i \alpha_j \langle \mathbf{x}, \mathbf{x} \rangle + \sum_{l,m} \alpha_l \alpha_m \langle \mathbf{x}_l, \mathbf{x}_m \rangle - 2 \sum_{l,j} \alpha_l \alpha_j \langle \mathbf{x}_l, \mathbf{x}_j \rangle + (\sum_l \alpha_l \| \mathbf{x}_l \|^2 - \sum_i \alpha_i \| \mathbf{x}_i \|^2) \\ s.t. \quad & \sum_i \alpha_i - \sum_l \alpha_l = 1, \quad \forall i, j \;\; 0 \leq \alpha_i \leq \gamma_1, 0 \leq \alpha_l \leq \gamma_2 \end{aligned} \tag{12}$$

Bu sınıflandırıcı iki sınıfa ait örnekleri de kullanmasına rağmen DVM'lerinden oldukça farklı prensipte çalışır. DVM iki sınıfı birbirinden ayırcak hiper-düzlem döndürürken, hiper-küre sınıflandırıcısı ise nesneye ait sınıfın örneklerini içeren ve arka-plan sınıfına ait örnekleri içermeyen kompakt bir küre döndürür. Bu durumda iki sınıfı ayıran karar sınırları DVM'lerinden farklı olarak doğrusal değildir. En iyi Lagrange katsayıları bulunduktan sonra hiper-kürenin merkezi aşağıdaki eşitlikle bulunur

$$\mathbf{c} = \sum_{i=1}^{n_1} \alpha_i \mathbf{x}_i - \sum_{l=1}^{n_2} \alpha_l \mathbf{x}_l . \tag{13}$$

Yaptığımız deneysel çalışmalarda arka-plana ait örneklerin dahil edilmesi sınıflandırıcının performansını arttırdığından hiper-küre sınıflandırıcısı için (12)'de verilen formülasyon kullanılmıştır. Bu ikilenik eniyileme problemi de konvekstir ve global en iyi noktası vardır. Eğitim setindeki örneklerin sayısının fazla olduğu durumlarda bu ikilenik problemde DVM'nde olduğu gibi "Sequential Minimal Optimization" tekniği kullanılarak çözülebilir. Bu durumda çok boyutlu Hessian matrisini oluşturmaya gerek kalmadan, iteratif olarak sadece kısıtları bozan iki örneğin Hessian matrisleri kullanılır. Biz bu amaçla http://cmp.felk.cvut.cz/ sayfasında paylaşıma açılmış ikilenik program çözücüyü (bu program çözücü sadece girdi olarak tam boyutlu Hessian matrisi kabul ediyordu) geliştirerek milyonlarca veriyle çalışacak hale getirdik ve hiper-küreleri bulmak için geliştirdiğimiz bu algoritmayı kullandık.

Hiper-küreyi karakterize eden parametreler (kürenin merkezi **c** ve kürenin yarıçapı *r*) bulunduktan sonra bir test örneğinin, $\mathbf{x}_{test}$, kürenin merkezine olan uzaklığı bulunur ve bu uzaklık yarıçapla karşılaştırılarak o örneğin nesne sınıfına ait olup olmadığı belirlenir. Matematiksel olarak ifade edilirse

$$|(\| \mathbf{x}_{test} - \mathbf{c} \|) - r| \leq \theta \Rightarrow \mathbf{x}_{test} \text{ nesne sinifina ait;}$$
$$|(\| \mathbf{x}_{test} - \mathbf{c} \|) - r| > \theta \Rightarrow \mathbf{x}_{test} \text{ arka - plana ait}$$

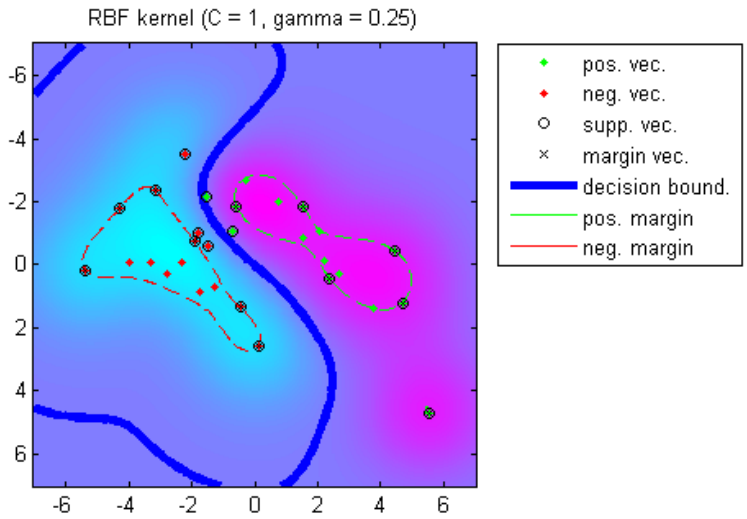
2.3.3 Doğrusal Olmayan Hiper-küre Sınıflandırıcısı

Ardışıl sınıflandırıcının son katmanında doğrusal olmayan hiper-küre sınıflandırıcısı kullanılmıştır. Hiç şüphesiz sınıfların topolojisi çok daha karmaşık ve konveks olmayan yapılar olabilir. Bu durumda sınıfları doğrusal hiper-kürelerle modellemek uygun değildir. Bu problemin çözümü için izlenen yol, örnekleri kernel hilesini kullanarak bu modelin uygun olduğu çok daha büyük boyutlu bir uzaya çıkarmak ve bu yeni uzayda kürenin parametrelerini bulmaktır. Bu işlem Destek Vektör Makinelerinde olduğu gibi eşitlik (12)'deki iç çarpımların kernel fonksiyonlarla değiştirilmesi ile kolayca gerçekleştirilir. Daha açık bir şekilde ifade edilirse her bir iç çarpım $< \mathbf{x}_i, \mathbf{x}_j >$, kernel fonksiyonu $k < \mathbf{x}_i, \mathbf{x}_j >=< \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) >$ ile değiştirilmelidir. Burada $\phi : R^d \rightarrow \Im$ giriş uzayındaki örnekleri büyük boyutlu yeni uzaya aktaran fonksiyonu ifade etmektedir. Kullanılan kernel fonksiyonlar arasında *m*. dereceden polinom fonksiyonu $k(\mathbf{x}_i, \mathbf{x}_j) = (< \mathbf{x}_i, \mathbf{x}_j >)^m$, Gaussian kerneli $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \| \mathbf{x}_i, \mathbf{x}_j \|^2)$ ve sigmoid kerneli $k(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\kappa(< \mathbf{x}_i, \mathbf{x}_j >) + \Theta)$ sayılabilir. Bu sayede çok daha karmaşık sınıfların topolojileri doğru olarak modellenebilir.

Doğrusal olmayan durumda bir örneğin hiper-küreye olan uzaklığının hesaplanabilmesi için sıfırdan farklı Lagrange katsayılarına karşılık gelen destek vektörleri kullanılarak kernel fonksiyonların hesaplanması gerekir. Bu sebeple doğrusal olmayan hiper-küre sınıflandırıcısı doğrusal hiper-küre sınıflandırıcısına oranla oldukça yavaştır. Fakat bu sınıflandırıcı son katmana gelen az sayıda örneğe uygulandığı için konum bulma sisteminin genel hızını çok kötü etkilememektedir.

Bunun yanında deneysel çalışmalar doğrusal olmayan hiper-küre sınıflandırıcısının doğrusal olmayan Destek Vektör Makinelerine oranla çok daha az destek vektörü döndürdüğünü ortaya çıkarmıştır. Deneysel olarak gözlemlenen bu durumu açıklamak için Şekil 2.6'dan yararlanılabilir. Doğrusal DVM sınıflandırıcısında destek vektörleri iki sınıfın birbirine yaklaştığı bölgelerden gelmektedir. Oysa doğrusal olmayan DVM sınıflandırıcısı kullanıldığında şekilde görüldüğü gibi destek vektörleri sınıfları sınırlayan tüm bölgelerden gelmektedir. Buna karşı tek sınıflı hiper-küre sınıflandırıcısında destek vektörler sadece nesne sınıfını (positive class) belirleyen sınırlardan geldiğinden, bu sınıflandırıcı DVM'ne oranla çok daha az destek vektör döndürmektedir. Bu da konum bulma deneylerimizde

önerilen ardışıl sınıflandırıcıyı kullanan konum bulma sisteminin hızını doğrusal olmayan Destek Vektör Makineleri kullanan konum bulma sisteminin hızına oranla 20 kata çıkan oranlarda arttırmıştır. Bu sebeple önerilen konum bulma sistemi, literatürdeki doğrusal olmayan sınıflandırıcı kullanan konum bulma sistemlerine oranla hız yönünden çok daha avantajlıdır. Doğrusal olmayan hiper-küre sınıflandırıcısı sınıflandırma başarımı yönünden DVM sınıflandırıcıları ile karşılaştırıldığında iki durum ortaya çıkmakatdır: Nesne grubuna ait örnek sayısı çok fazla ise başarım yönünden bu iki sınıflandırıcısı arasında pek bir fark gözlenmemiştir (deneysel çalışmalar kısmında verilen yüz sezme deneyleri bu seneryoya girmektedir). Fakat örnek sayısı az ise doğrusal olmayan DVM sınıflandırıcısı çok daha başarılı sonuçlar vermiştir (deneysel çalışmalarda verilen PASCAL VOC veri tabanı üzerindeki çalışmalar bu duruma girmektedir ve bu farkın nedenleri "Vargılar ve Tartışma" kısmında ayrıntılı olarak açıklanmıştır)



Şekil 2.6. DVM sınıflandırıcısı ile elde edilen karar sınırları. Sınıflandırıcının döndürdüğü destek vektörler her iki sınıfı belirleyen sınırlardan gelmektedir.

2.4 KONUM BULMA SİSTEMİNİN AZALTILMIŞ KÜME YÖNTEMLERİ KULLANILARAK HIZLANDIRILMASI

Proje kapsamında yaptığımız çalışmalarda doğrusal olmayan hiper-küre sınıflandırıcılarının yanında karşılaştırma amaçlı doğrusal olmayan DVM sınıflandırıcısı da kullanılmıştır. Özellikle doğrusal olmayan DVM algoritmasının döndürdüğü destek vektör sayısı çok fazla olduğu için konum bulma sisteminin hızı oldukça yavaşlamıştır. Örnek olarak doğrusal olmayan hiper-küre sınıflandırıcısının kullanıldığı ardışıl sınıflandırıcı bir imgedeki

nesnenin konumunu 4-5 saniye gibi bir zamanda döndürürken, doğrusal olmayan DVM'ni kullanan ardışıl sınıflandırıcı için bu zaman 2 dakikaya kadar çıkabiliyordu. Bu durumu iyileştirmek adına doğrusal olmayan sınıflandırıcıların sınıflandırma başarısını fazla düşürmeden, hızlarını arttırma yoluna gidilmiştir. Bu da doğrusal olmayan sınıflandırıcıların döndürdüğü Destek Vektör Makinelerinin sayısının gelişmiş teknikler kullanılarak azaltılması ile mümkündür. Bu yöntemler literatürde azaltılmış küme (reduced set) yöntemi olarak adlandırılmaktadırlar. DVM ve Kernel Principal Component Analysis (KPCA) gibi yöntemlerin döndürdüğü Destek Vektör Makinelerinin sayısını azaltmak için birkaç yöntem önerilmiştir [36, 37, 38], ve tüm bu yöntemler bizim amaçlarımız doğrultusunda kullanılabilir. Biz bu projede [37]'de verilen ve iteratif altuzay kestirimine dayalı yöntemi kullandık.

Azaltılmış küme yöntemlerinde amaç test örneklerinin sınıflandırılması sırasında kullanılan Lagrange katsayıları ve bunlara karşılık gelen destek vektörleri, daha az sayıda destek vektörleri ve bunlara uygun katsayılar ile ifade etmektir. Matematiksel olarak amaç $\Psi = \sum_{i=1}^{n_s} \alpha_i \phi(\mathbf{x}_i)$ terimini (bu terim doğrusal olmayan hiper-küre sınıflandırıcısında hiper-küreye olan uzaklığın bulunmasında, doğrusal olmayan DVM sınıflandırıcısında ise ayırıcı hiper-düzleme olan uzaklığın bulunmasında kullanılır) daha az sayıda destek vektör kullanarak yakınsamaktır

$$\sum_{i=1}^{n_s} \alpha_i \phi(\mathbf{x}_i) \approx \sum_{i=1}^{n_z} \beta_i \phi(\mathbf{z}_i) \qquad (14)$$

Yukarıdaki eşitlikte $\mathbf{z}_i$ vektörleri azaltılmış destek vektörleri kümesini oluşturan yeni destek vektörlerini, β_i'ler ise bu vektörlere karşılık gelen uygun kombinasyon katsayılarını ifade etmektedir. Azaltılmış kümedeki yeni destek vektörlerinin sayısı n_z, orjinal destek vektörlerinin sayısından n_s çok daha az olacak şekilde seçilir, $n_z < n_s$. Bu işlem iki aşamada gerçekleştirilir: İlk olarak azaltılmış kümeye ait $\mathbf{z}_i$ vektörleri bulnur ve daha sonra (14)'deki eşitliği sağlayacak uygun β_i katsayıları bulunur. Elimizde azaltılmış kümeye ait tüm $\mathbf{z}_i$ vektörleri varsa, $\boldsymbol{\beta}$ vektörü aşağıdaki formül kullanılarak elde edilir

$$\boldsymbol{\beta} = (\mathbf{K}^z)^{-1} \mathbf{K}^{zx} \boldsymbol{\alpha}. \qquad (15)$$

Yukarıdaki eşilikte $\mathbf{K}^z = \left[\left\langle \phi(\mathbf{z}_i), \phi(\mathbf{z}_j) \right\rangle\right]$ ve $\mathbf{K}^{zx} = \left[\left\langle \phi(\mathbf{z}_i), \phi(\mathbf{x}_j) \right\rangle\right]$ ile ifade edilmektedir.

Kernel olarak Gaussian kernel seçilmesi durumunda $\Psi = \sum_{i=1}^{n_s} \alpha_i \phi(\mathbf{x}_i)$ terimini en iyi yakınsayan tek bir $\mathbf{z}$ vektörü aşağıdaki iteratif işlemle elde edilebilir [37]

$$\mathbf{z}_{t+1} = \frac{\sum_{i=1}^{n_s} \alpha_i \exp(-\| \mathbf{z} - \mathbf{x} \|^2 / \gamma) \mathbf{x}_i}{\sum_{i=1}^{n_s} \alpha_i \exp(-\| \mathbf{z} - \mathbf{x} \|^2 / \gamma)}. \tag{16}$$

Fakat bizim amacımız birden fazla $\mathbf{z}_i$ vektörü bulmak olduğu için, çalışmalarımızda Şekil 2.7'de verilen algoritma kullanılmıştır.

Bu yaklaşım kullanılarak nesne konum bulma sisteminin doğruluk performansında küçük kayıplara rağmen çok büyük hızlandırmalar elde edilmiştir.

Algorithm 1 Algorithm for Obtaining Reduced Set of Support Vectors

Input:

$\mathbf{X} = [\mathbf{x}_1 ... \mathbf{x}_{n_s}]$: initial set of support vectors

$\alpha = [\alpha_1 ... \alpha_{n_s}]$: initial set of expansion coefficients

n_z: number of reduced support vectors

Output: $\mathbf{Z}$ and β

Initialization:

$\mathbf{X}_{initial} = \mathbf{X}$, $\alpha_{initial} = \alpha$, $\mathbf{Z} = []$.

Description:

for $i = 1 : n_z$ **do**

$\mathbf{z}_0 = randn(d, 1)$;

while $||\mathbf{z}_{t+1} - \mathbf{z}_t|| \geq tol$ **do**

$\mathbf{z}_{t+1} = \frac{\sum_{i=1}^{n_s} \alpha_i \exp(-||\mathbf{z}_t - \mathbf{x}_i||^2/\gamma)\mathbf{x}_i}{\sum_{i=1}^{n_s} \alpha_i \exp(-||\mathbf{z}_t - \mathbf{x}_i||^2/\gamma)}$;

end while

$\mathbf{z} = \mathbf{z}_{t+1}$;

$\mathbf{Z} = [\mathbf{Z}\ \mathbf{z}]$;

$\beta = (\mathbf{K}^z)^{-1} \mathbf{K}^{zx_{initial}} \alpha_{initial}$

$\alpha = [\alpha\ \ -\beta]$;

$\mathbf{X} = [\mathbf{X}\ \mathbf{Z}]$;

end for

Şekil 2.7. Azaltılmış destek vektör kümesini bulmak için kullanılan algoritma.

2.5 NESNE KONUM BULMADA NESNE PARÇALARINDAN FAYDALANMA TEKNİKLERİ

Son zamanlarda yapılan çalışmalarda nesne konum bulmada nesnelere ait parçaların kullanımının başarımı önemli ölçüde arttırdığı gözlenmiştir. Bunun başlıca nedeni ise parça-temelli yöntemlerin örtüşmeye yada esnek deformasyonlara karşı daha gürbüz olmasıdır. Bizde bu sebeple [31]'de önerilen yöntemi izleyerek kök (roots) ve parça (parts) seziciler olmak üzere iki ayrı konum bulma algoritması eğittik. Kök sezicilerin amacı imgede nesnenin tüm görüntüsünü içeren olası bölgeleri döndürmektir ve bu amaçlada önceki bölümlerde ayrıntılı bir şekilde açıkladığımız ardışıl sınıflandırıcı kullanılmıştır. Kök sezicilerin döndürdüğü bölgelere uygulanan parça seziciler, olası nesne görüntüsü içeren bölgelerde nesnelere ait parçaların konumlarını bulmada kullanılmışlardır. Bir imge penceresinin nesne görüntüsünü içerip içermediğine dair en son karar ve güven skoru (confidence score) kök ve parça sezicilerin her ikisinide geçen pencerenin aldığı skorların çarpımıyla elde edilmiştir. Kök ve parça seziciler [31]'de olduğu gibi farklı ölçeklerde uygulanmışlardır (parça seziciler kök sezicin döndürdüğü pencerenin iki katı büyüklükteki ölçekte uygulanmışlardır).

Parça sezicileri karaketerize ederken [31]'de verilen yöntem izlenmiş fakat kök ve parça seziciler farklı bir şekilde uygulanmıştır. Çalışmalarımızda *n*-parçadan oluşan bir nesne (*n*+1) elemandan oluşan, $P_1, P_2, ..., P_n, b$, parça modeli ile betimlenmiştir. [31]'de olduğu gibi her bir parça P_i'de 3 elemandan oluşan (f_i, v_i, d_i) modeli ile betimlenmiştir. Burada f_i parça i için kullanılan filtreyi, v_i iki boyutlu demir atma noktası (anchor position) olarak adlandırılan ve parçanın konumunu kök sezicin döndürdüğü pencerenin konumuna bağlı olarak ifade eden konumu göstermektedir. Demir atma noktasından farklı konumlara tekabül edilen deformasyonların ağırlıklandırılması içinde 4-boyutlu d_i katsayı vektörü kullanılmıştır. Bir parça sezicinin H ile gösterilen bir imge öznitelik uzayında $l_i = (x_i, y_i)$ pozisyonundaki güven skoru aşağıdaki formülle bulunmuştur

$$\text{score}(P_1, ..., P_n) = \sum_{i=1}^{n} f_i^T \Phi(H, P_i) - \sum_{i=1}^{n} d_i^T \Phi(dx_i, dy_i) + b. \qquad (17)$$

Yukarıdaki eşitlikte (dx_i, dy_i) parça i'nin demir atma noktasına bağlı konumunu, $\Phi(dx_i, dy_i) = (dx_i, dy_i, dx_i^2, dy_i^2)$ deformasyon özniteliklerini ve $\Phi(H, P_i)$ parça i için kullanılan imge öznitelik vektörünü ifade etmektedir. Tüm bu parametreler tek bir $\boldsymbol{\beta}$ vektörü ile ifade edilirse, güven skoru $\boldsymbol{\beta}^T \Phi(H, z)$ eşitliği ile bulunabilir.

Parça filtrelerinin ilk değerleri [31]'de olduğu gibi Destek Vektör Makinesi sınıflandırıcısının döndürdüğü hiper-düzlemin normali kullanılarak belirlenmiştir, daha sonra bu değerler iteratif olarak eniyilenmiştir. Deformasyon katsayılarının başlangıç değerleri $d_i = (0,0,0.05,0.05)$ olarak seçilmiş, daha sonra bunlarda filtrelerle birlikte iteratif olarak güncellenmiştir. Filtreler ve katsayıların başlangıç değerleri kullanılarak, nesneye ve arka-plana ait örneklerin öznitelik vektörleri, $\Phi(H,z)$, belirlenmiş ve daha sonra parça modelleri aşağıda verilen ikilenik eniyileme problemi çözülerek güncellenmiştir,

$$\begin{aligned} &\min_{\boldsymbol{\beta},\xi\geq 0} \ \frac{1}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} + C\sum_i \xi_i \\ &s.t. \ \ \boldsymbol{\beta}^{\mathrm{T}}\Phi(H,z_i) + b \geq 1-\xi_i, \qquad \forall i \in I_+, \\ &\qquad \boldsymbol{\beta}^T\Phi(H,z_i) + b \leq -1+\xi_i, \qquad \forall i \in I_-, \\ &\qquad \beta_k \leq 0, \ \forall k \in D. \end{aligned} \tag{18}$$

Yukarıdaki eşitlikte D ikilenik deformasyon katsayılarına, dx^2, dy^2, karşılık gelen indisleri içeren kümeyi ifade etmektedir. Bu ikilenik programlama DVM formülüne benzemekle birlikte çok büyük bir farklılık vardır. Son kısıtta ikilenik deformasyon katsayılarına karşılık gelen Lagrange katsayılarının negatif olma zorunluluğu vardır. Bu şekilde eşitlik (17)'de verilen skor hesaplanırken, ikilenik deformasyon katsayılarının toplam skordan daima çıkartılması sağlanmış olur. Bu sınıflandırıcıda da DVM sınıflandırıcısına benzer şekilde nesneye ait örneklerle arka-plana ait örnekler arasında en azından $2/\|\boldsymbol{\beta}\|$ genişliğinde bir boşluk (margin) kalması garantilenir.

Bu eniyileme problemi DVM'den farklıdır ve literatürdeki standard ikilenik programlama çözme yöntemleriyle çözümü mümkün değildir. [31]'de yapılan çalışmada da bu yaklaşım kullanılmakla beraber paylaşıma açtıkları kod karmaşık bir yapıda verildiği için bu kodu kullanmak mümkün olmamıştır. Bunun yerine Çek Teknik Üniversitesi'nden Franc Vojtech ile işbirliğne giderek kendi kodumuzu geliştirdik ve http://cmp.felk.cvut.cz/~xfrancv/ocas/html/index.html adresinde araştırmacıların kullanımına açtık. Problemin çözümü için eniyilenmiş kesen düzlemler algoritması (optimized cutting plane) kullanılmıştır. Bu algoritma son yıllarda çok fazla verinin kullanıldığı ikilenik problemlerin çözümü için başarı ile uygulanmıştır. Bu algoritmada kesen düzlemler modeli parametre uzayında en iyi yönü bulmak için kullanılmaktadır. Bizim problemimizde bazı katsayıların negatif olması gerektiğinden bu aşamada algoritmada bu kısıtların gerçeklenmesi için iyileştirmeler yapılmıştır. (18)'de verilen problemin çözümü için ayrıca [42]'de bir

çalışma yapılmış ve geliştirilen Matlab kodu yakın bir zamanda kullanıcıların paylaşımına açılmıştır.

3. DENEYSEL ÇALIŞMALAR

Proje kapsamında geliştirdiğimiz konum bulma yöntemini sayısal imgelerde yüz, insan, ve PASCAL VOC veritabanındaki nesnelerin konumlarını bulmada kullandık (Geliştirilen sistemin kodlarına http://www2.ogu.edu.tr/~mlcv/softwarecvpr2012.html adresinden ulaşılabilir). Konum bulma yöntemlerinin başarımlarını ölçmek için PASCAL VOC ölçütü kullanılmıştır. Bu ölçüte göre konum bulma sisteminin döndürdüğü pencere R ile elle işaretlenen nesne konumlarını içeren pencereler Q arasındaki örtüşmenin tüm alana olan oranı, $\frac{\text{alan}|Q \cap R|}{\text{alan}|Q \cup R|}$, %50'den büyükse bu doğru bir konum olarak sayılmıştır. Aksi durumda bulanan pencere yanlış (false positive – FP) olarak kabul edilmiştir. (Örtüşme oranı yüz sezme deneyleri için %45 olarak alınmıştır. Bunun bir sebebi farklı kişiler tarafından işaretlenen yüz bölgesinin kişilere bağlı olarak farklı olarak yorumlanmasıdır. Diğer sebebi ise literatürdeki mevcut yüz sezme algoritmalarının yüz bölgelerini bizim işaretlediğmiz yüz bölgelerinden farklı olarak döndürmesidir. Bu algoritmalar üzerinde çalışarak yüz konumlarını bizim istediğimize benzer şekilde döndürecek değişiklikler yapılmıştır. Aradaki farklılıkların sonuca etkilememesi içinde örtüşme oranı %50 yerine %45 olarak alınmıştır.) Daha sonra PASCAL VOC ölçütü kullanılarak doğru ve yanlış konumlar olarak belirlenen pencerelerin skorları kullanılarak Precision-Recall eğrileri elde edilmiş ve başarım performansı olarak bu eğrilerden elde edilen ortalama kesinlik skorları (average precision – AP) kullanılmıştır.

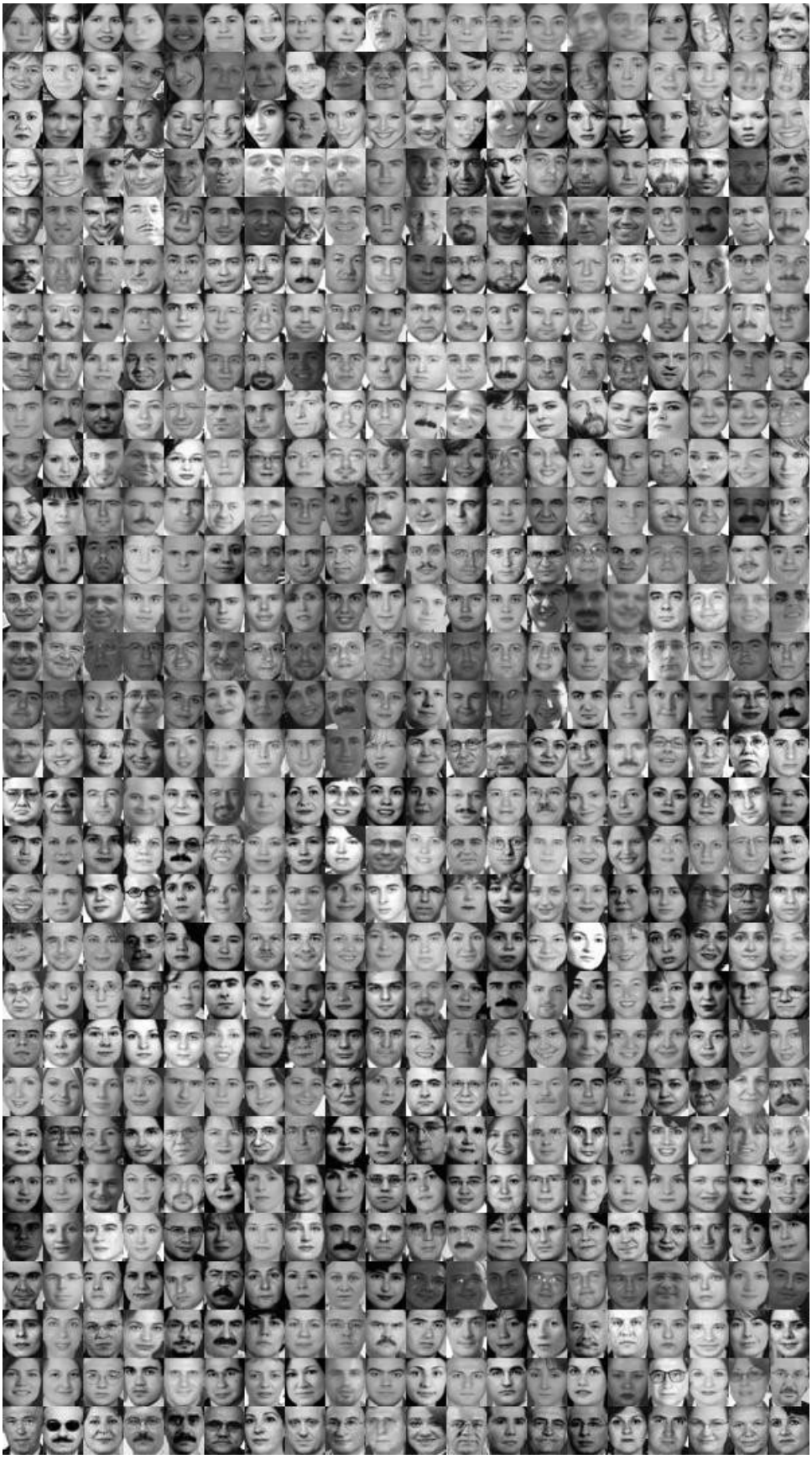
3.1 YÜZ SEZME

Yüz sezme deneyleri için literatürde en çok kullanılan veritabanı MIT-CMU [1] veri tabanıdır. Fakat bu veri tabanındaki resimler gri-seviyelerde çekilmiş resimler olup çözünürlükleri çok düşüktür. Bu sebeple renkli imgelerle çalışabilen HOG gibi betimleyicilerin tüm avantajlarını kullanmak mümkün deeğildir. Ayrıca veri tabanındaki resim sayısı azdır. Daha yakın bir zamanda oluşturulan "Faces in the Wild" (http://vis-www.cs.umass.edu/lfw adresinden indirilebilir) veri tabanındaki resimlerin çoğunda yüzler resmin tam ortasındadır ve benzer boyutlardadırlar. Bu sebeple yüz sezme algortimalarının farklı ölçeklerdeki konum bulma başarısını ölçmek mümkün değildir. Ayrıca bu veri tabanı yüz sezme değil yüz tanıma amaçlı toplanmış olup, imgelerdeki yüzler Viola ve Jones [17] tarafından önerilen AdaBoost ardışıl sınıflandırıcı kullanan yüz sezme algoritması kullanılarak toplanmıştır. Buna bağlı olarak bu veri tabanı bu algoritmaya bir avantaj sağlamaktadır. Bu sebeple literatürdeki bu alandaki boşluğu doldurmak için ESOGU

(Eskişehir OsmanGazi University) adını verdiğimiz yeni bir yüz sezme veri tabanı oluşturduk. Yüz sezme algoritmalarının başarımları "Faces in the Wild" ve ESOGU yüz sezme veri tabanları üzerinde test edilmiştir.

3.1.1 Yüz Konum Bulma Sisteminin Eğitilmesi

Yüz konum bulma sistemini eğitmek amacıyla ilk olarak web ortamından 12500 civarında ön cepheden çekilmiş yüz resmi topladık. Bu amaçla toplanan bazı örnekler Şekil 3.1'de gösterilmektedir. Bunların arasında yüz tanıma veritabanlarından alınan örnekler olmakla birlikte günlük hayatta çekilen gerçekçi resimler çoğunluktadır. Bu sebeple yüz örnekleri arasında büyük görünüş ve aydınlatma farklılıkları vardır. Daha sonra tüm örnekler 35×28 boyutlarına sığacak şekilde örneklenmişler ve bu örnekler kullanılarak Yerel İkili Örüntü (YİÖ) ve Yönlü Gradyan Histogramları (YGH) öznitelikleri elde edilmiştir. Arka-plan sınıfı içinse içinde yüz içermeyen imgelerden rastgele 10000 adet örnek alınmış ve öznitelik vektörleri çıkarılmıştır. YİÖ özniteliklerini çıkarmak için her örnek 2×2 eşit parçaya bölünmüş ve her parçadan yarıçapı 1 ve komşuluğu 8 olan YİÖ öznitelikleri çıkarılmıştır. Bu histogramlar arka-arkaya bağlanmak suretiyle tüm örneğe ait YİÖ öznitelik vektörü elde edilmiştir. YGH özniteliklerini çıkarmak için 6×6 piksel boyutunda hücreler (cell) oluşturularak herbir hücreden Felzenszwalb'ın [31] yaklaşımı kullanılarak 32 boyutlu YGH öznitelikleri çıkarılmıştır. Bunlarda arka-arkaya bağlanarak tüm resme ait YGH öznitelik vektörü elde edilmiştir. Bu özniteliklerin yanında Yerel Üçlü Örüntü (Local Ternary Patterns) öznitekleri de denenmiş, ve tüm bu öznitelikler ve onların kombinasyonlarından oluşan öznitelikler başarım açısından test edilmiştir. Son olarak hem başarım hem de boyut kıstasları göz önüne alınarak YİÖ ve YGH özniteliklerinin birlikte kullanılmasına karar verilmiştir. İlk veriler kullanılarak yüz konum bulma sistemindeki ardışıl sınıflandırıcılar eğitilmiş, ve sistem binlerce yeni imge üzerinde test edilerek sistemin konumlarını doğru olarak döndüremediği yüz resimleri (hard positives) ve sistemin yanlışlıkla yüz olarak döndürdüğü arka-plan örnekleri (hard negatives-false positives) toplanmıştır. Sistem daha sonra bu yeni zor örneklerde eklenerek tekrar eğitilmiş ve yeni imgeler üzerinde test edilmiştir. Bu işlem bir kaç kere tekrarlanmış ve konum bulma sistemine son şekli verilmiştir. Eğitim setindeki yüz sınıfına ait son örnek sayısı 20K olup arka-plan örnek sayısı ise 112275'dir. Böylece eğitim setindeki toplam örnek sayısı 132K civarına yükselmiştir. Eğittiğimiz yüz konum bulma sisteminde 4 katmandan oluşan bir ardışıl sınıflandırıcı kullanılmıştır. İlk katmanda doğrusal olmayan bir DVM sınıflandırıcısı kullanılmış olup, diğer üç katman tek sınıflı



Şekil 3.1. Yüz konum bulma sistemini eğitmek için kullanılan yüz resimlerinden örnekler.

sınıflandırıcıları içermektedir. Yani ikinci katmanda doğrusal bir hiper-düzlem sınıflandırıcısı, üçüncü katmanda doğrusal bir hiper-küre sınıflandırıcısı ve son katmanda ise doğrusal olmayan kernel hiper-küre sınıflandırıcısı kullanılmıştır. Bu ardışıl sınıflandırıcıya ek olarak 2.5'de anlatılan yaklaşım kullanılarak sisteme parça seziciler de eklenerek başarımın üzerinde etkileri araştırılmıştır (ayrıntılar için [44]'de verilen çalışmamıza bakınız).

3.1.2 Faces in the Wild Yüz Veri Tabanı Kullanılarak Elde Edilen Sonuçlar

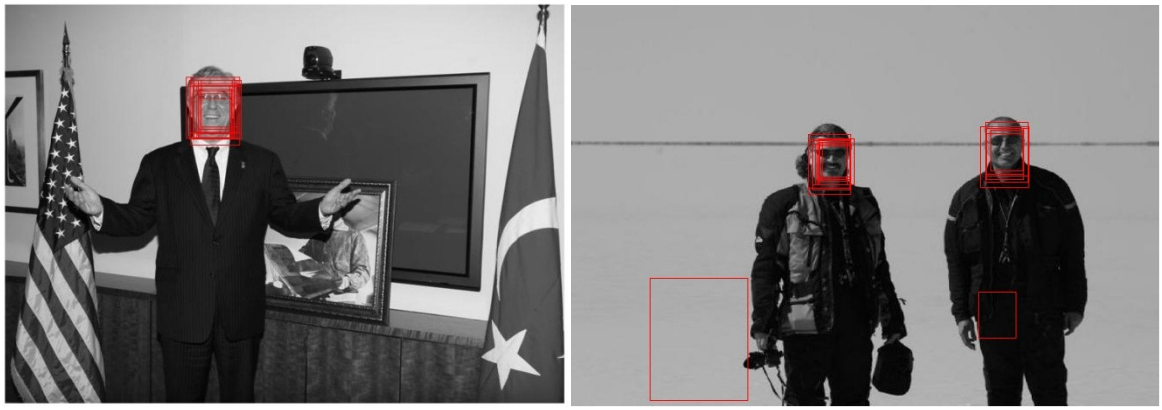
Konum bulma algoritmalarını test etmek için bu veri tabanından ön cepheden çekilmiş yüzleri içeren 13127 adet resim kullandık. Veri tabanındaki yüz resimlerinin büyük bir bölümünde yüzler resmin tam ortasında olup aynı boyutlarda olacak şekilde ölçeklendirilmişlerdir. Bu sebeple bu veri tabanı çok farklı ölçeklerde çalışması gereken yüz sezme algoritmalarının başarımlarını ölçmek için uygun değildir.

Yöntemimizi literatürde yakın zamanlarda önerilen başarılı yüz sezme algoritmalarıyla karşılaştırdık. Karşılaştırma amacıyla Viola ve Jones [17] tarafından önerilen AdaBoost tabanlı ardışıl sınıflandırıcıları kullanan yüz sezme algoritması, Zhu ve Ramanan [42] tarafından önerilen parça tabanlı yüz sezme algoritması ve Kalal ve diğerleri [43] tarafından önerilen yüz sezme yöntemi kullanılmıştır.

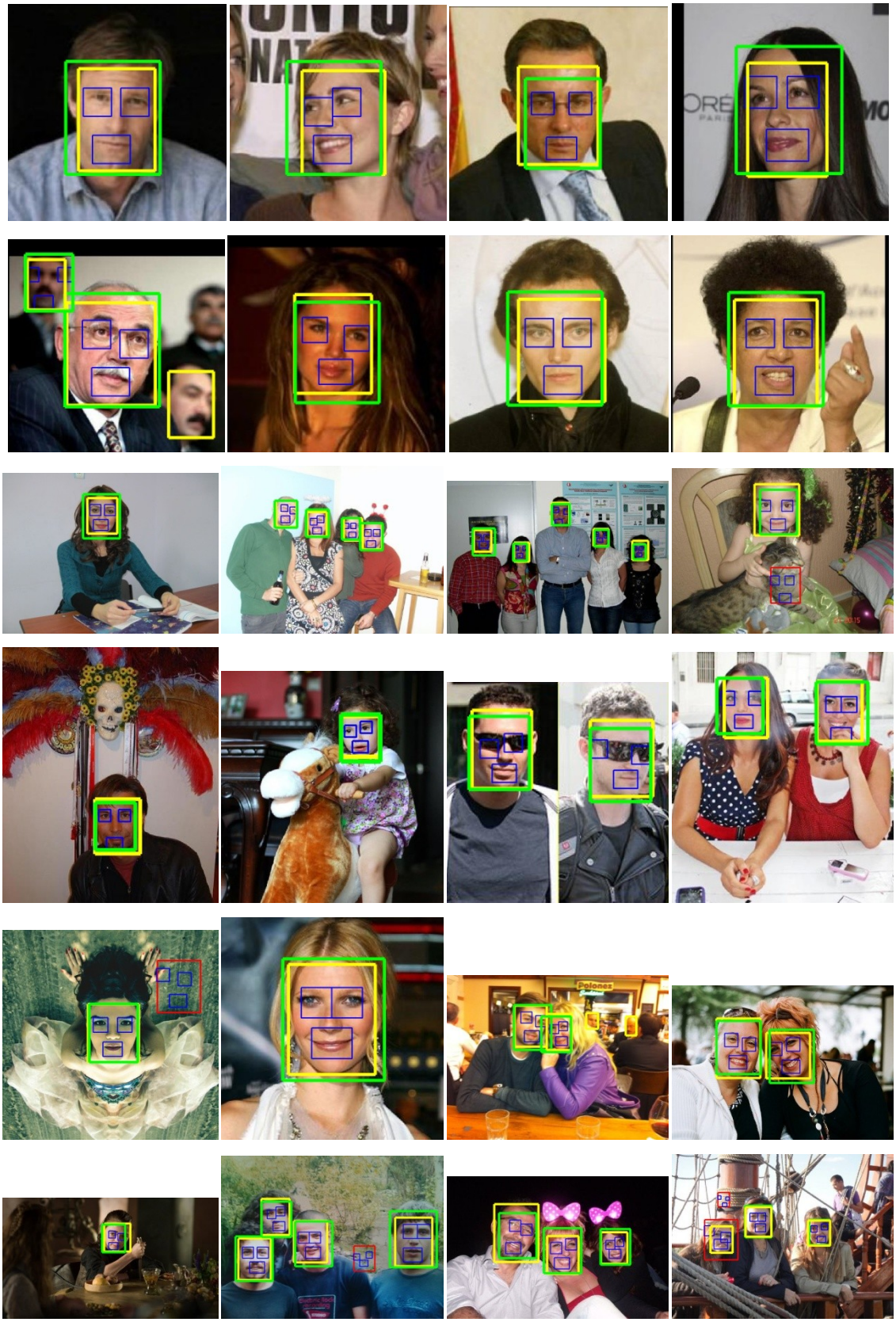
Test aşamasında kayan pencelereler yöntemi kullanılmıştır. Bu yöntemde ilk olarak 1.15 ölçek skalası kullanılarak imge altörnekleme işlemine tabi tutulmuş ve farklı ölçeklerden oluşan bir imge piramidi oluşturulmuştur. Sınıflandırıcılar uygulanırken kullanılan pencere yatay düzlemde 3 piksel düşey düzlemde 4 piksel kaydırılmıştır. Bu şekilde imgede yüzün bulunduğu bölgelerin etrafında bir çok pencere Şekil 3.2'de görüldüğü gibi yüz olarak sınıflandırılmaktadır. Bu penceler kullanılarak yüzün bulunduğu en iyi tek bir konum bulunmalıdır. Bu amaçla maksimum olmayanı bastırma (non-max surpression) tekniğini kullandık. Bu tekniğe göre ardışıl sınıflandırıcıların yüz olarak sınıflandırdığı tüm pencereler en yüksek skordan başlayarak sıralanmış ve iteratif olarak en yüksek skora sahip pencere yüz olarak kabul edilmiş ve onunla örtüşen diğer pencereler elenmiştir. Şekil 3.2'nin sağ tarafında görüldüğü gibi arka-plana ait başka pencerelerde ardışıl sınıflandırıcılar tarafından yüz olarak döndürülmesine rağmen yüze benzemeyen bölgelerde döndürülen pencere sayısı oldukça azdır. Dikkat edilirse gerçekten yüzün olduğu bölgelerde genel olarak fazla sayıda pencere yüz olarak döndürüldüğü için bu bilgiler kullanılarak, arka-plandan gelen bazı yanlış sınıflandırılmalar elenebilir. Biz bu sebeple ardışıl sınıflandırıcının döndürdüğü bir pencereyle örtüşen en az 3 pencere daha yoksa bu pencereyi de eledik. Bu da bizim yüz (yada nesne)

olmadığı halde yüz olarak döndürülen bir çok pencereyi (false positives) elememizi sağlamıştır.

Elde edilen başarım oranları Tablo 3.1'de verilmiştir. Ayrıca önerdiğimiz konum bulma sistemlerinin bu veri tabanından seçilen bazı örnekler için döndürdüğü konumlar Şekil 3.3'de gösterilmiştir. Tablo 3.1'de Ardışıl Sınıflandırıcılar önerdiğimiz temel yöntemi ifade etmektedir ve Ardışıl Sınıflandırıcılar + Parça Seziciler hem kök hem de parça sezicileri kullanan yöntemimizi ifade etmektedir. Bu veri tabanı için en yüksek başarım, %96.90, Zhu ve Ramanan [42] tarafından önerilen yüz sezme algoritması ile elde edilmiştir. Bu yöntemi önerdiğimiz yöntemler takip etmişlerdir. Parça sezicileri kullanan yöntem sadece kök sezicilerden oluşan temel yönteme oranla çok küçük bir iyileşme sağlamıştır. En kötü sonuçları Viola ve Jones tarafından önerilen yöntem vermiştir. Faces in the Wild veri tabanındaki resimler Viola ve Jones ardışıl sınıflandırıcısı kullanılarak toplanmasına rağmen bu yöntemin kötü sonuçlar vermesinin temel nedeni resmin ortasındaki büyük yüz resimlerinin yanındaki daha küçük ve zor olan yüzlerin konumlarını da etiketleyerek başarım ölçümünde kullanmamızdır. Bu algoritma bu tür zor yüz resimlerini büyük ölçüde kaçırmıştır.



Şekil 3.2. Maksimum olmayanı bastırma uygulanmamış konum bulma algoritması çıktıları.



Şekil 3.3. Ardışıl sınıflandırıcılar ve parça seziciler kullanarak eğittiğimiz yüz sezme algoritmalarımızın Faces in the Wild ve ESOGU Yüz Sezme veritabanlarından seçilen bazı imgeler üzerindeki çıktıları. Yeşil kutular doğru olarak bulunan konumları, kırmızı kutular yanlış konumları, mavi kutular ise yüz parçaları olarak bulunan konumları göstermektedir.

Tablo 3.1. Yüz sezme yöntemlerinin Faces in the Wild veri tabanındaki başarı oranları. Değerler ortalama kesinlik skorları olup Precision-Recall eğrilerinin altında kalan alanların hesaplanmasıyla bulunmuşlardır.

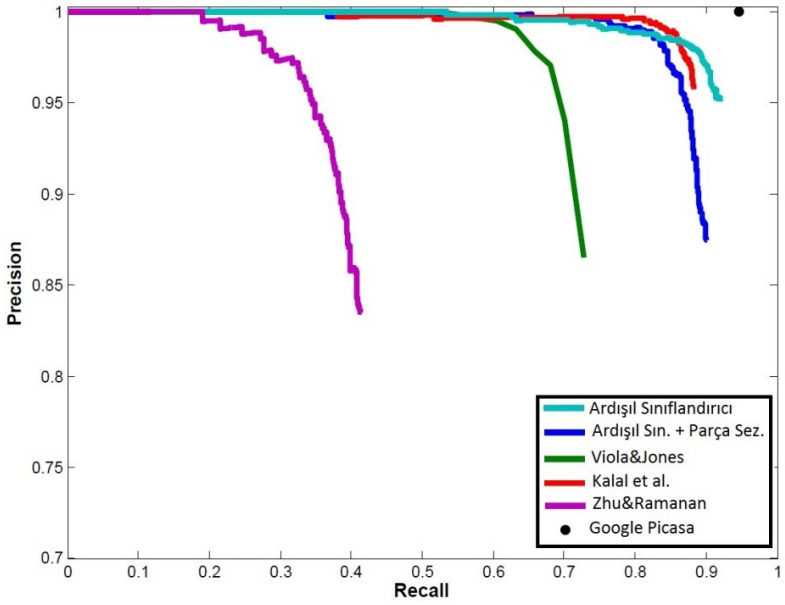
Yöntemler	**Ortalama Kesinlik Skorları - Average Precision (%)**
Ardışıl Sınıflandırıcı	94.12
Ardışıl Sınıflandırıcı+Parça Seziciler	94.39
Kalal et al. [43]	87.89
Zhu&Ramanan [42]	96.90
Viola&Jones [17]	87.89

3.1.3 ESOGU Yüz Sezme Veri Tabanı Kullanılarak Elde Edilen Sonuçlar

Konum bulma yöntemleri ayrıca bizim tarafımızdan toplanan "ESkisehir OsmanGazi University" (ESOGU) yüz sezme veri tabanı üzerinde test edilmiştir. Literatürdeki yüz sezme veri tabanlarındaki boşluğu doldurmak amacıyla oluşturduğumuz bu veri tabanındaki tüm imgeler günlük hayatı yansıtan kareler olup, arka-planları oldukça karmaşıktır. Yüzler arasında çok farklı aydınlatma ve ölçek farklılıkları olduğu gibi bazı resimlerde yüzün olduğu bölgeler farklı nesnelerle kapatılmıştır. Bu veri tabanı iki parçadan oluşmaktadır. İlk toplanan veri tabanında 285 adet renkli imge bulunmaktadır. Bu imgeler içinde 970 adet ön cepheden gözüken yüz resmi bulunmaktadır. Bu ilk veri tabanı [22]'de yaptığımız çalışmalarda kullanılmıştır. Daha sonra hakemlerin önerileri de dikkate alınarak bu veri tabanına 382 yeni imge eklenerek son şekli verilmiştir. Veri tabanında toplam 667 imge bulunmaktadır ve bu imgelerde 2042 adet ön cepheden çekilen yüz bulunmaktadır. Veri tabanından bazı resimler Şekil 3.5'de gösterilmiştir.

ESOGU yüz sezme veri tabanında elde edilen Precision-Recall eğrileri Şekil 3.4'de, bu eğrilerden elde edilen kesinlik skorları ise Tablo 3.2'de gösterilmiştir. Tabloda görüldüğü üzere bu veri tabanında en iyi sonuçlar önerdiğimiz yöntemlerle elde edilmiştir. Önerilen yöntemlerden sonra Kalal ve diğerleri [43] tarafından önerilen yöntem üçüncü sırada olup en kötü sonuçları ise Zhu ve Ramanan'ın [42] yöntemi vermiştir. Faces in the Wild veri tabanında en iyi sonuçları veren yöntemin bu veri tabanında kötü sonuçlar vermesinin temel nedeni yöntemin sadece parça tabanlı olmasından kaynaklanmıştır (yöntem kök seziciler kullanmamaktadır). Yöntem, parçaların iyi olarak ayırt edilebilmesi için yüz resimlerinin boyutunun 80×80'den büyük olmasını gerektirmektedir. Bu şart Faces in the Wild veritabanındaki yüz resimleri için sağlanmakla birlikte ESOGU veri tabanında bu boyutlardan

küçük bir çok yüz bulunmaktadır ve yöntem bu yüzlerin konumlarını bulamamıştır. Bu sebeple performans çok büyük ölçüde düşmüştür. Konum bulma yöntemlerini bu veri tabanı üzerinde ayrıca ticari bir ürün olan Google Picasa ile karşılaştırdık. Google Picasa konumları verme yerine imge üzerinde yüzleri gösterdiğinden, bu programın döndürdüğü tüm yüz imgelerini görsel olarak saydık ve bunların hepsini doğru olarak kabul ettik (Bazı yüzler PASCAL VOC ölçütü göze alındığında yanlış görünmekle birlikte bunlar da doğru olarak sayılmıştır). Ayrıca programın döndürdüğü ve yanlış olarak etiketlenen arka-plan görüntüleri de dikkate alınmamıştır. Bu da bize "Recall Rate" oranını vermektedir. Recall rate oranı %91.52 olarak bulunmuş ve Precision-Recall eğrisi üzerinde işaretlenmiştir. Önerdiğimiz yöntemlerde Recall rate oranları bu orana oldukça yakındır ve buda oldukça umut vericidir.



Şekil 3.4. ESOGU Yüz Sezme veri tabanı için elde edilen Precision-Recall eğrileri.

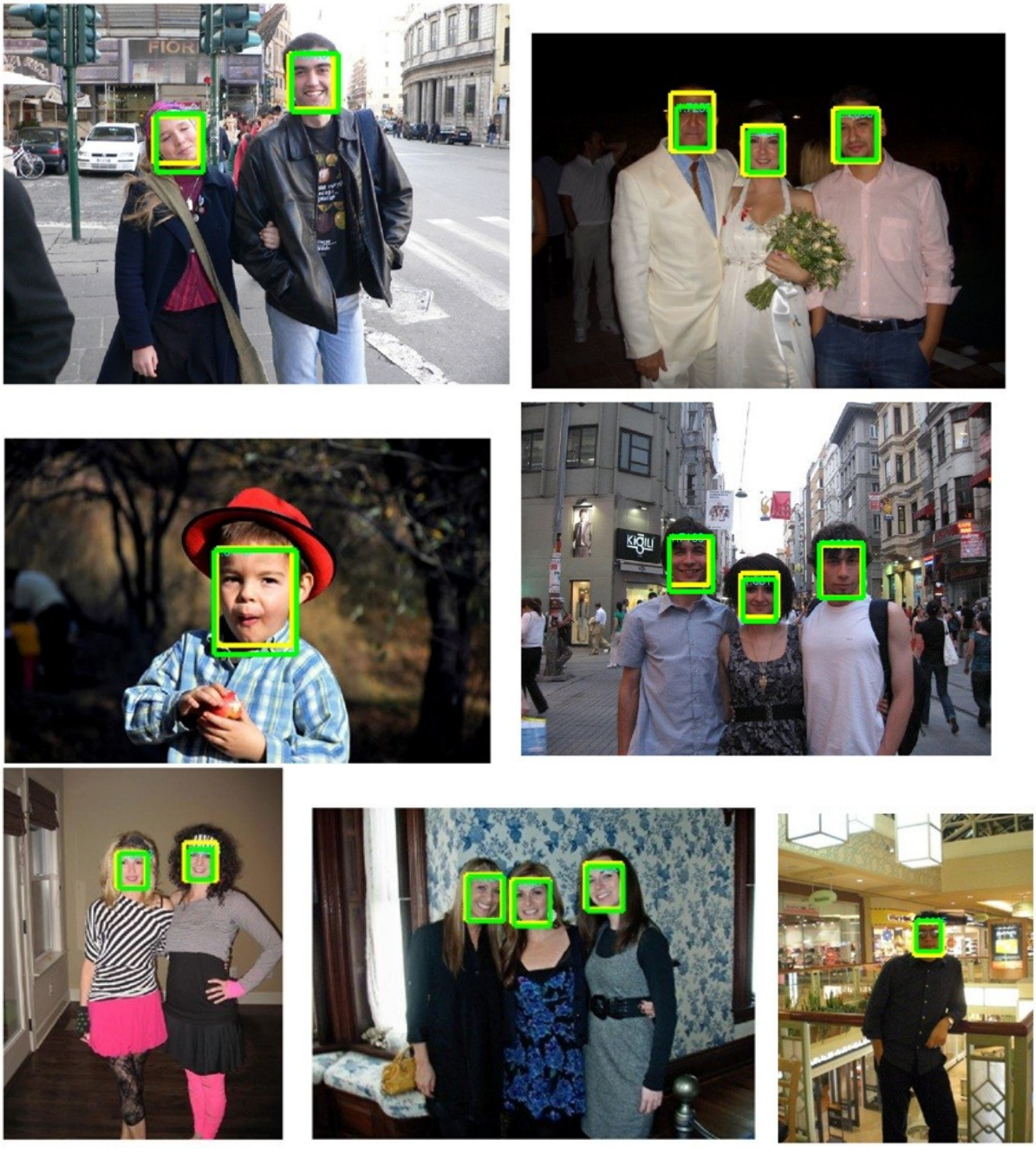
Tablo 3.2. Yüz sezme yöntemlerinin ESOGU Yüz Sezme veri tabanındaki başarı oranları. Değerler ortalama kesinlik skorları olup Precision-Recall eğrilerinin altında kalan alanların hesaplanmasıyla bulunmuşlardır.

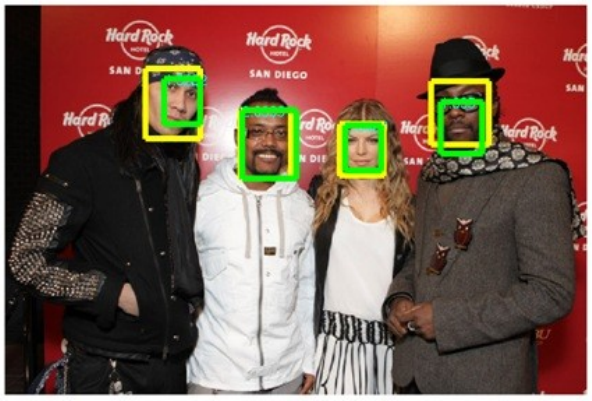
Yöntemler	**Ortalama Kesinlik Skorları - Average Precision (%)**
Ardışıl Sınıflandırıcı	87.05
Ardışıl Sınıflandırıcı+Parça Seziciler	83.86
Kalal et al. [43]	79.67
Zhu&Ramanan [42]	50.55
Viola&Jones [17]	73.61

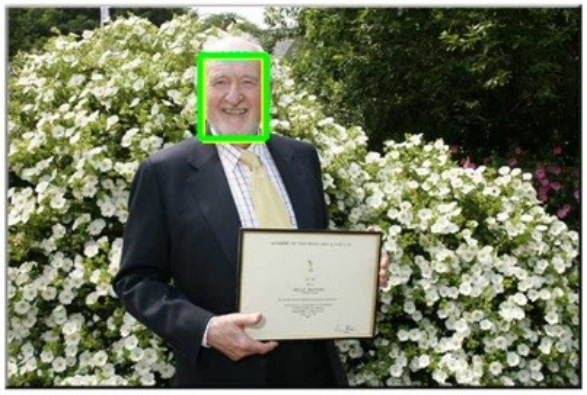
Şekil 3.5. ESOGU yüz sezme veri tabanından alınan bazı örnekler. İmgelerdeki yüz resimleri farklı ölçeklerde olup, arka-planlar oldukça karmaşıktır. Ayrıca imgeler farklı aydınlatma koşulları altında çekilmiş olup, bazı yüz resimleri farklı nesneler tarafından kısmi olarak kapatılmıştır.

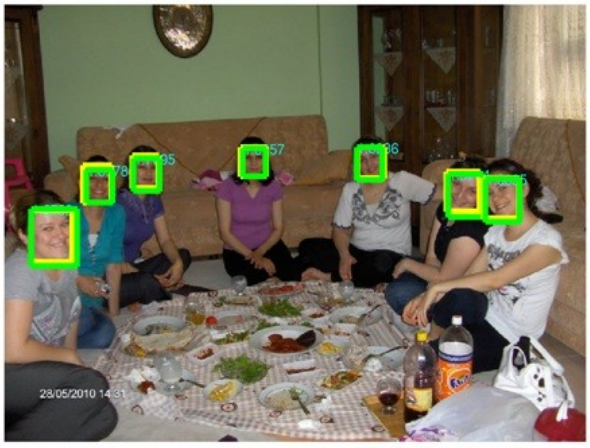
3.1.4 ESOGU Yüz Sezme Veri Tabanında ve Diğer İmgelerde Elde Edilen Görsel Sonuçlar

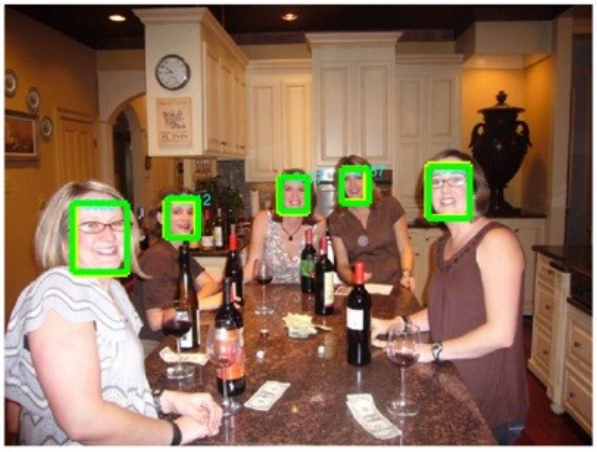
Burada geliştirilen ardışıl sınıflandırıcının ESOGU veri tabanından seçilen bazı imgeler üzerindeki çıktıları görsel olarak verilerek sistemin başarılı ve başarısız olduğu durumlar hakkında fikir edinilmesi amaçlanmıştır. İlk olarak sistemin tüm yüzleri başarı ile yakaladığı örnekler Şekil 3.6'da verilmiş daha sonra da sistemin yüzleri bulamadığı yada yanlış bulduğu durumlardan örnekler Şekil 3.7'de verilmiştir. Son olarak veri tabanında bulunmayan bir kaç örnek imge üzerinde elde edilen sonuçlar Şekil 3.8'de verilmiştir.

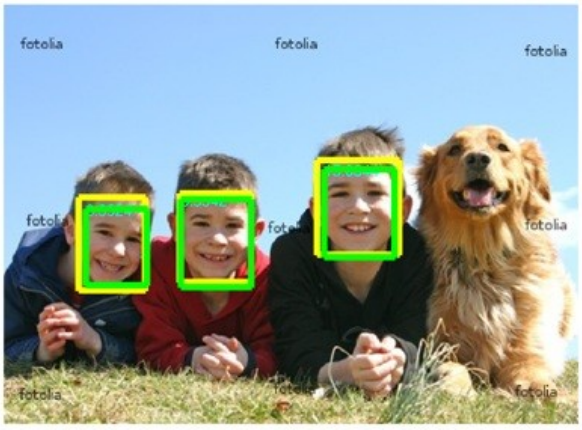



Hard Rock
SAN DIEGO

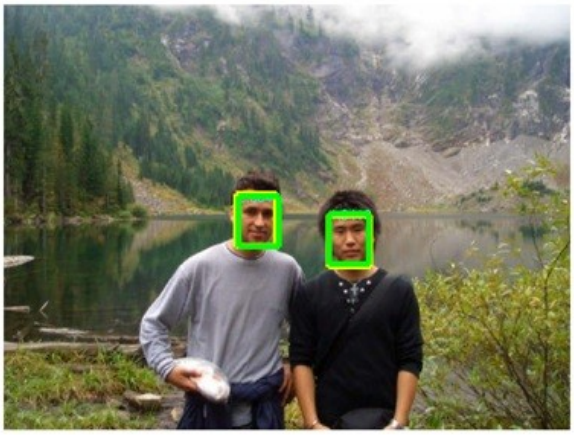


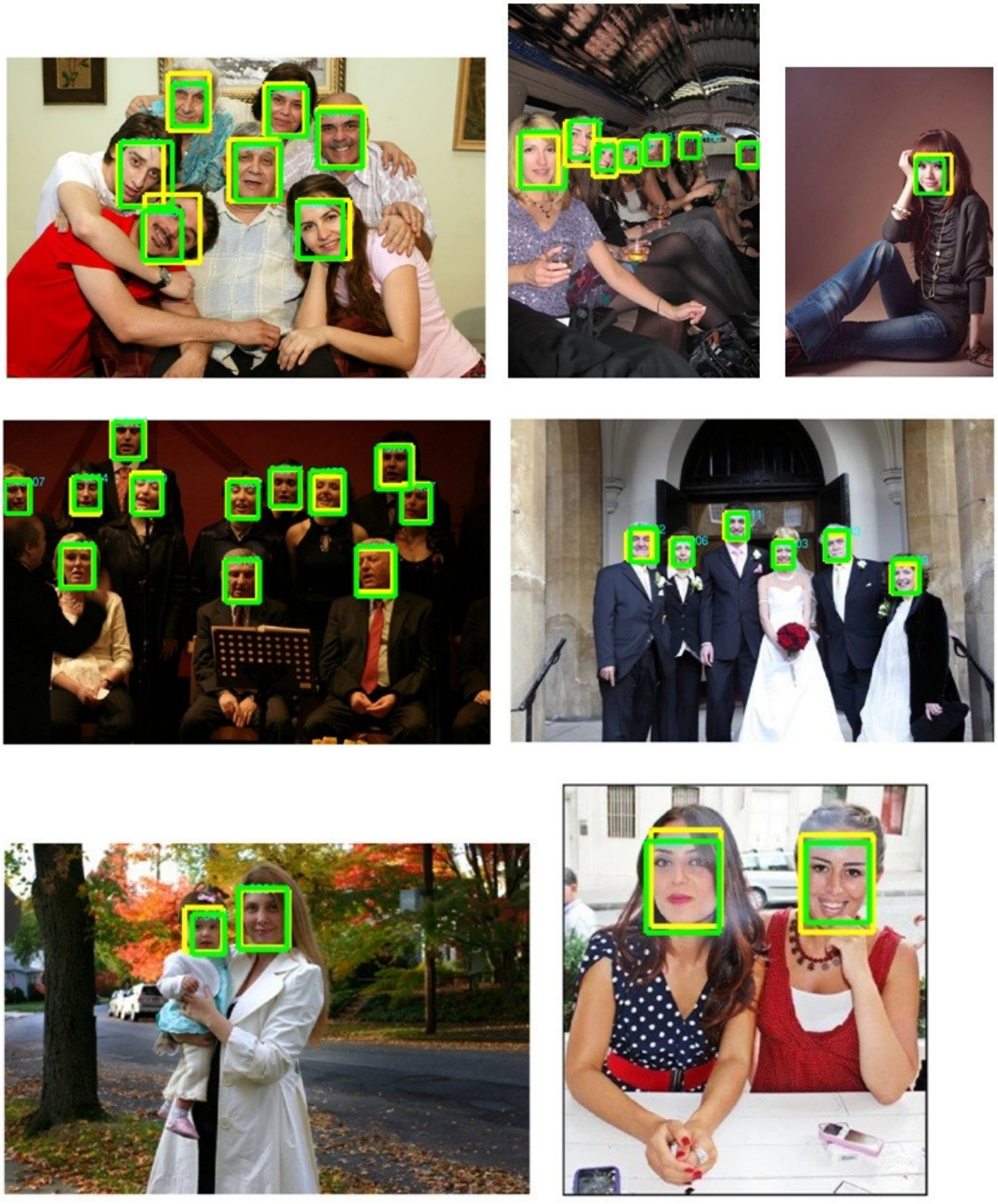

28/05/2010 14:31




fotolia
fotolia
fotolia
fotolia
fotolia
fotolia
fotolia

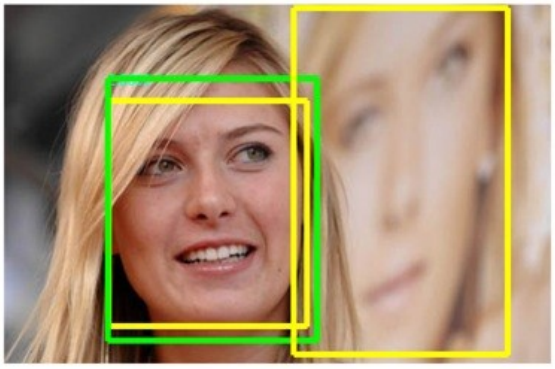

21/12/2008 10:12

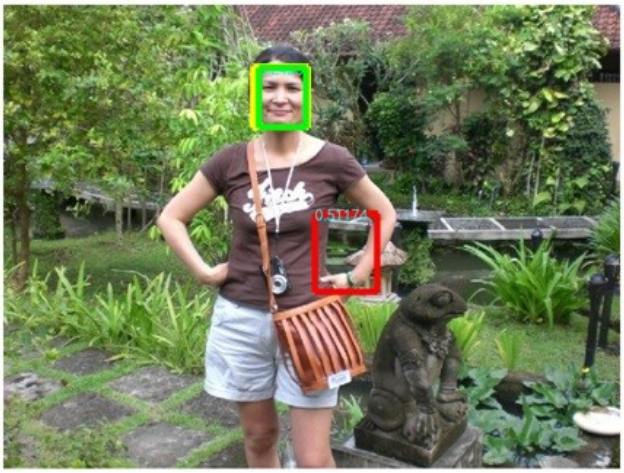


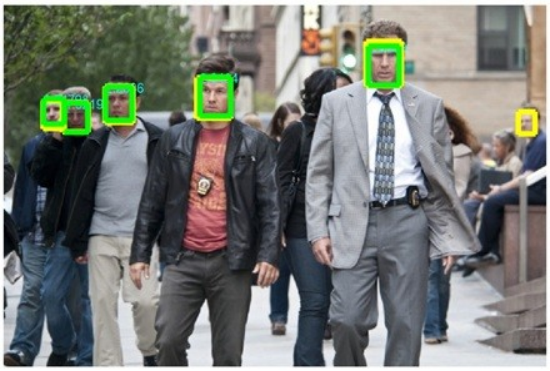


Şekil 3.6. Tüm yüzlerin geliştirdiğimiz ardışıl sınıflandırıcıları kullanan konum bulma sistemi tarafından başarı ile bulunduğu örnekler. Sarı kutular elle işaretlenen doğru konumları yeşil kutular ise sistem tarafından döndürülen ve PASCAL VOC ölçütü kullanılarak doğru olarak kabul edilen konumları göstermektedir.

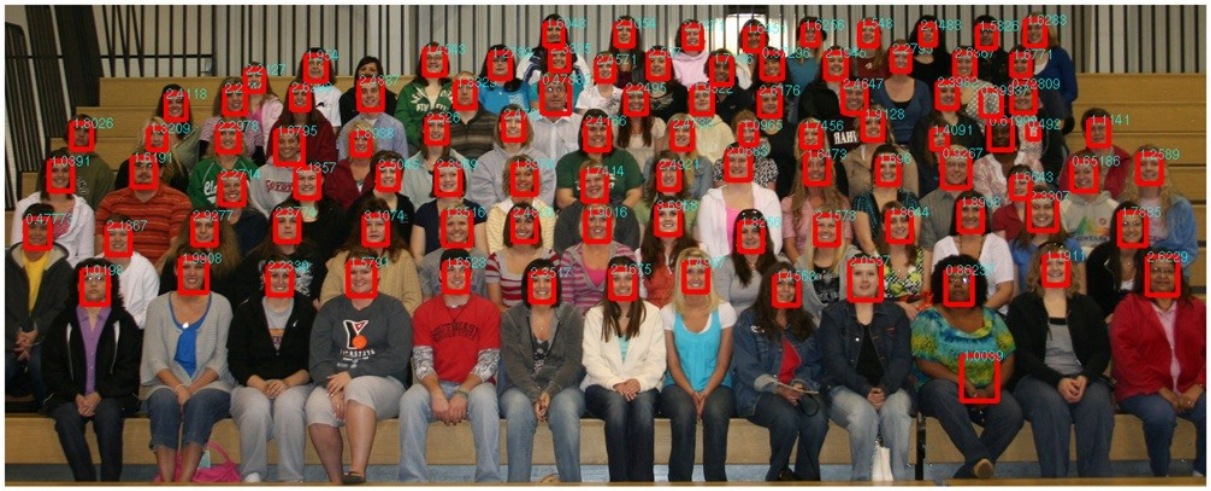

billard
SNOOKER
BUKANA PUB



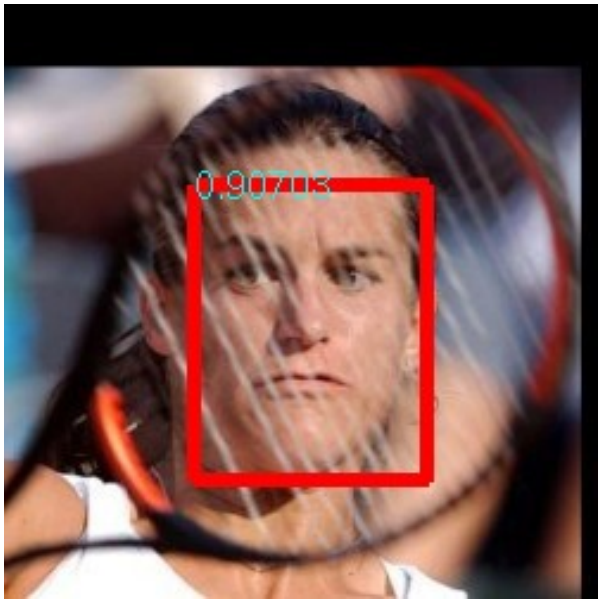


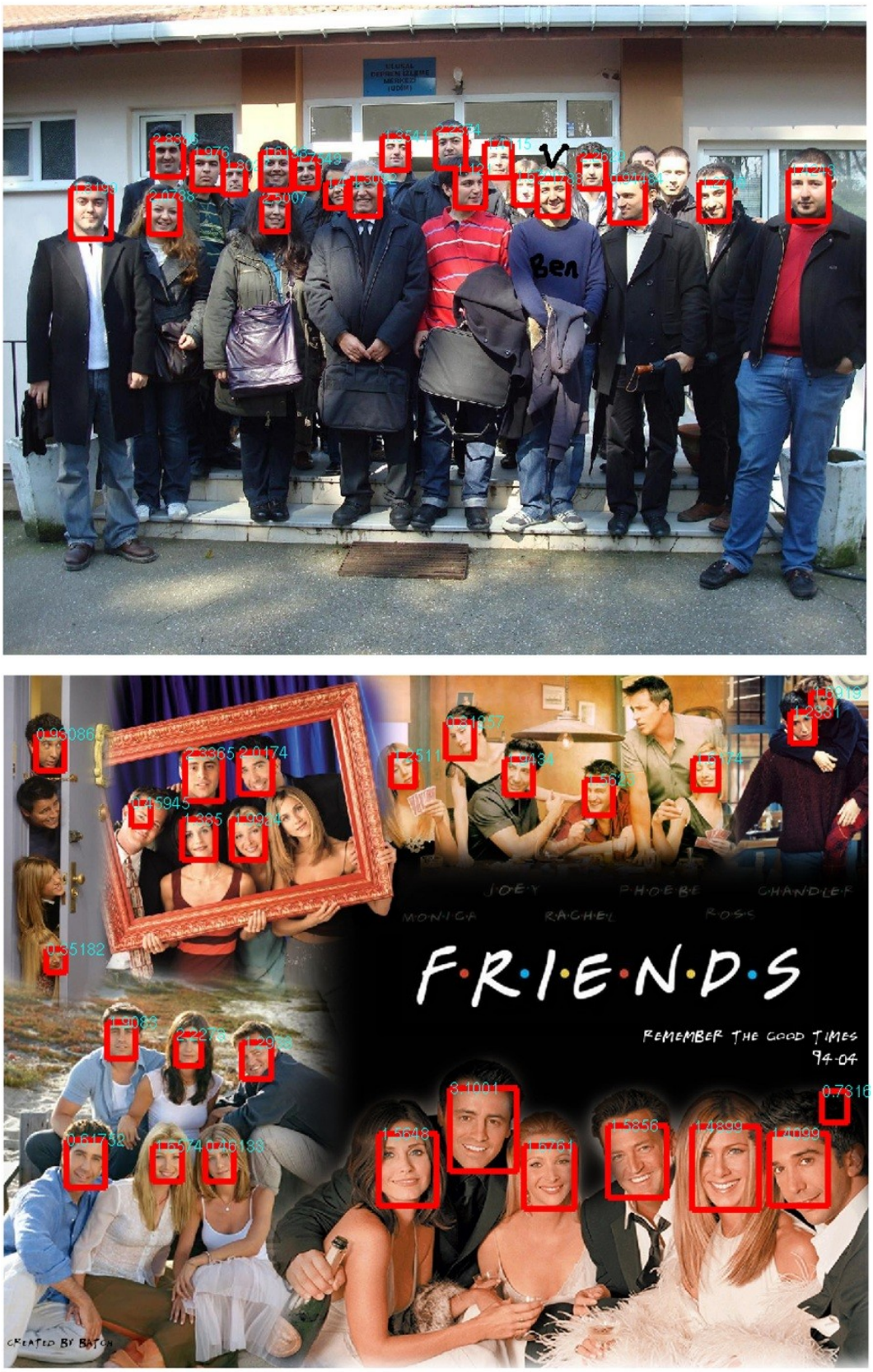


Şekil 3.7. Konum bulma sisteminin hata yaptığı bazı örnekler. Sarı kutular elle işaretlenen doğru konumları yeşil kutular sistem tarafından döndürülen ve PASCAL VOC ölçütü kullanılarak doğru olarak kabul edilen konumları, kırmızı kutular ise PASCAL VOC ölçütüne göre yanlış olarak değerlendirilen konumları göstermektedir. Görüldüğü gibi sistem bazı çok fazla dönmüş yüzleri, çok küçük yüzleri veya diğer nesneler tarafından kapatılan yüzleri kaçırabilmektedir. Ayrıca bazı örneklerde de arka-planda yüzü andıran bölgeler yanlış olarak yüz resmi olarak döndürülmüştür (kırmızı kutularla belirtilen bölgeler).




11





Şekil 3.8. Konum bulma sisteminin veri tabanlarında olmayan imgeler üzerinde çıktıları. Kırmızı kutular sistemin yüz resmi olarak döndürdüğü konumları göstermektedir.

3.2 İNSAN SEZME

3.2.1 İnsan Konum Bulma Sisteminin Eğitilmesi

İnsan sezimi için Inria Person veritabanı [7] kullanılmıştır. Yüz sezmede olduğu gibi imgeleri betimlemek için YİÖ+YGH öznitelikleri kullanılmıştır. Fakat yüz sezmeden farklı olarak YİÖ özniteliklerini çıkarmak için her örnek 5×3 eşit parçaya bölünmüş ve her parçadan yarıçapı 1 ve komşuluğu 8 olan YİÖ öznitelikleri çıkarılmıştır. Bu histogramlar arka-arkaya bağlanmak suretiyle tüm örneğe ait YİÖ öznitelik vektörü elde edilmiştir. YGH özniteliklerini çıkarmak içinse 8×8 piksel boyutunda hücreler (cell) oluşturularak herbir hücreden Felzenszwalb'ın [31] yaklaşımı kullanılarak 32 boyutlu YGH öznitelikleri çıkarılmıştır. Bunlarda arka-arkaya bağlanarak tüm resme ait YGH öznitelik vektörü elde edilmiştir. Eğitim setindeki insanlara ait kısıtlı örnek sayısını arttırmak için, elle belirlenen doğru koordinatlar rastgele küçük miktarlarda ötelenerek örnek sayısının arttırımı sağlanmıştır. Daha sonra test setindeki imgeler kullanılarak rastgele 12K civarında arka-plan örneği oluşturulmuştur. Bu verilerle eğitilen sınıflandırıcılar eğitim setindeki tüm imgelere uygulanarak zor örnekler toplanmış ve bunlar da eğitim setindeki verilere eklenmiştir. Zor örnekler toplanırken özellikle eğitilen sistem elle işaretlenen ve insanların olduğu bilinen pozisyonların civarında uygulanmış ve PASCAL VOC metriğine göre örtüşme oranı %30'un altında kalan ve insanın bulunduğu konumdan gelen pencereler arka-plan sınıfına eklenmiştir. Bu şekilde deneyler sırasında performansta bir artış sağlanmıştır.

3.2.2 Deneysel Sonuçlar

Test aşamasında yüz sezme deneylerinde olduğu gibi kayan pencelereler yöntemi kullanılmıştır. Benzer şekilde ilk olarak 1.15 ölçek skalası kullanılarak imge altörnekleme işlemine tabi tutulmuş ve farklı ölçeklerden oluşan bir imge piramidi oluşturulmuştur. Sınıflandırıcılar uygulanırken kullanılan pencere yatay düzlemde 4 piksel düşey düzlemde 6 piksel kaydırılmıştır. Bu deneylerde parça seziciler kullanılmamıştır. Elde edilen başarımlar Tablo 3.3'de verilmiştir. Önerilen yöntemlerin başarısı şu an en iyi konum bulma yöntemleri arasında gösterilen Hussain&Triggs [15] ve Felzenszwalb ve arkadaşları [31] tarafından önerilen yöntemlerle karşılaştırılmıştır. Önerilen tüm ardışıl sınıflandırıcılar bu veri tabanında literatürde en iyi kabul edilen yöntemlerden çok daha başarılı sonuçlar vermiştir.

Tablo 3.3. İnsan sezme yöntemlerinin INRIA Person veri tabanındaki başarı oranları. Değerler ortalama kesinlik skorları olup Precision-Recall eğrilerinin altında kalan alanların hesaplanmasıyla bulunmuşlardır.

Yöntemler	**Ortalama Kesinlik Skorları - Average Precision (%)**
Ardışıl Sınıflandırıcı	93.46
Dalal&Triggs [7]	75.00
Hussain&Triggs [15]	84.10
Felzenszwalb et al. [31]	86.90

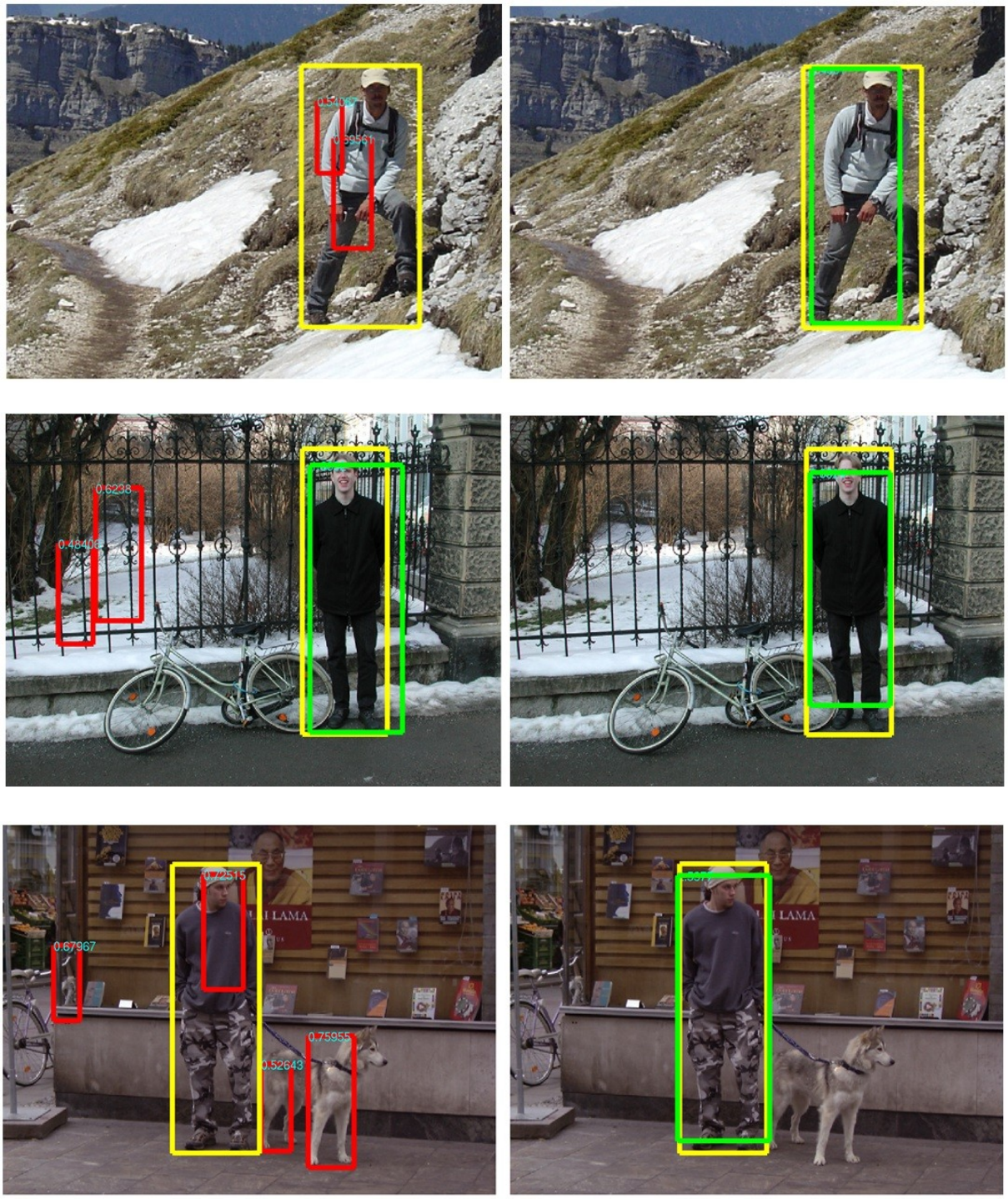
Geliştirdiğimiz yöntem ayrıca Dalal&Triggs [7] tarafından önerilen ve sadece doğrusal DVM sınıflandırıcısını kullanan sistem ile de kapsamlı bir şekilde görsel olarak karşılaştırılmış ve ne tür iyileştirmeler kazandırdığı gözlemlenmiştir. Bunun yanında geliştirdiğimiz sistemde sadece doğrusal sınıflandırıcılar kullanılarak da sonuçlar elde edilmiş ve bunlar doğrusal olmayan hiper-küreleri de içeren klasik sistemle karşılaştırılmış ve doğrusal olmayan sınıflandırıcıların performans üzerindeki etkileri gözlemlenmiştir.

İlk olarak Şekil 3.9'da doğrusal DVM sınıflandırıcısı kullanan konum bulma sistemi ile doğrusal DVM ve doğrusal hiper-küreleri birlikte kullanan konum bulma sistemi karşılaştırılmıştır. Doğrusal hiper-küre sınıflandırıcısının sisteme eklenmesi iki önemli katkıyı sağlamıştır. İlk olarak Şekil 3.9'da görüldüğü gibi şeklin sağ tarafında hiper-küre sınıflandırıcısını da kullanan sistem çok daha az sayıda kırmızı kutularla gösterilen yanlış konum (false positives) döndürmektedir. İkinci bir katkısı ise maksimum olmayanı bastırma algoritması üzerindeki iyileştirmedir. Bu algoritma tamamıyla sınıflandırıcıların döndürdüğü skorlara bağlıdır. Doğrusal DVM sınıflandırıcısını kullanan konum bulma sisteminde skor olarak ayırıcı hiper-düzleme olan uzaklıklar kullanılmıştır. Fakat bu skor bir örneğin bir sınıftaki diğer örneklere olan benzerliğini ölçme açısından uygun olamayabilir. Fakat bir sınıfı en iyi yakınsayan düzleme olan uzaklıklar yada bir sınıfı temsil eden hiper-kürenin merkezine olan uzaklıklar bu benzerliği ölçme açısından daha uygundur. Bu durum Şekil 3.9'un en üstündeki resimlerde açıkça görülmektedir. Doğrusal DVM skorları kullanan maksimum olmayanı bastırma algoritması insan resmini içeren bölgeden gelen çok daha küçük ve PASCAL VOC ölçütüne göre yanlış olan bölgeleri döndürürken doğrusal hiper-küre skorlarınında hesaba katılmasıyla sağdaki imgede görüldüğü gibi doğru konumlar bulunmuştur.

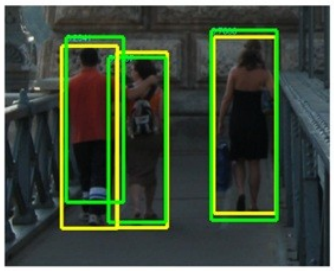
Şekil 3.9. Sadece doğrusal DVM sınıflandırıcısı kullanılarak elde edilen konum bulma sistemi çıktıları (sol taraftaki resimler) ile doğrusal DVM ve doğrusal hiper-küre sınıflandırıcısı kullanılarak elde edilen konum bulma sisteminin çıktılarının (sağ taraftaki resimler) karşılaştırılması.

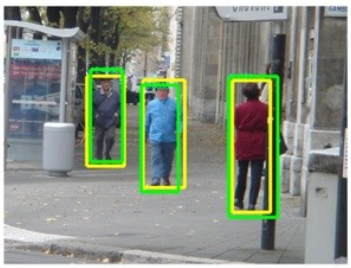
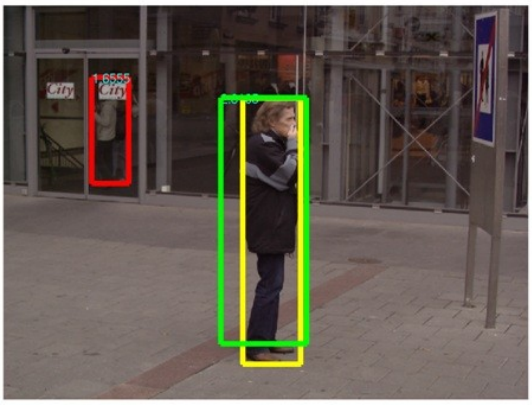
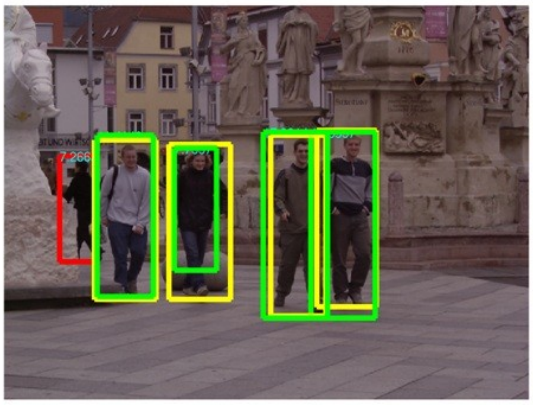
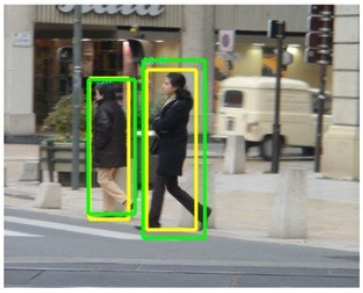
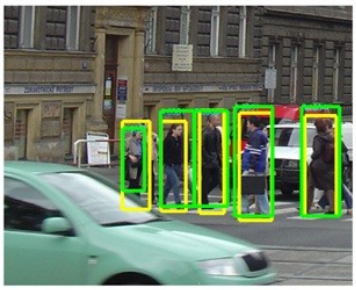
Şekil 3.10'da ise doğrusal sınıflandırıcıları kullanan konum bulma sistemi ile hem doğrusal sınıflandırıcıları hem de doğrusal olmayan kernel hiper-küre sınıflandırıcısını birlikte kullanan konum bulma sisteminin çıktıları görsel olarak karşılaştırılmıştır. Şekil 3.10'un sol tarafındaki resimler sadece doğrusal sınıflandırıcıları (doğrusal DVM + doğrusal hiper-düzlem + doğrusal hiper-küre) kullanan konum bulma sisteminin çıktılarını, sağ taraftaki resimler ise doğrusal sınıflandırıcılara ek olarak doğrusal olmayan hiper-küre sınıflandırıcını da içeren konum bulma sisteminin çıktılarını göstermektedir. İmgelerde görüldüğü üzere doğrusal olmayan sınıflandırıcının kullanılması hem yanlış döndürülen konumları azaltmış hem de çok daha güvenilir skorlar üreterek maksimum olmayanı bastırma algoritmasının çıktılarını iyileştirmiştir.

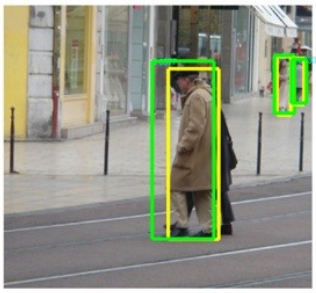
Şekil 3.11'de önerilen doğrusal ve doğrusal olmayan sınıflandırıcıların brilikte kullanıldığı konum bulma sisteminin Inria People test imgeleri üzerinde tüm insanları doğru bir şekilde bulduğu başarılı sonuçlar verilmiştir. Daha önceki gibi sarı kutular elle işaretlenen konumları, yeşil kutular konum bulma sisteminin döndürdüğü doğru konumları, kırmızı kutular ise yanlış konumları göstermektedir. Bu şekilde imgeler üzerinde yanlış olarak değerlendirilen kırmızı kutular olmakla birlikte o konumlarda gerçekten insanlar bulunmaktadır ve bunlar doğru çıktılardır. Bu yanlışlıklar veri tabanındaki etiketleme hatalarından kaynaklanmaktadır. Şekil 3.12'de ise sistemin hata yaptığı örnekler verilmiştir. Şekilde verilen imgelerde görüldüğü üzere sistem bazı durumlarda arka-plandan gelen bölgeleri yanlışlıkla insan olarak döndürmüş, bazı durumlarda insanların bulunduğu konumları tamamıyla kaçırmış, bazende insanların bulunduğu konumum tamamı yerine PASCAL VOC metriğine göre yanlış olarak değerlendirilen daha küçük yada daha büyük konumlar döndürmüştür.

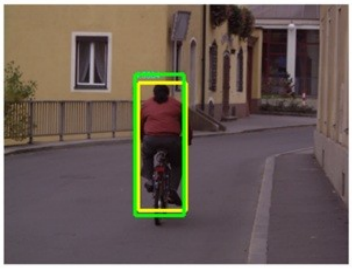
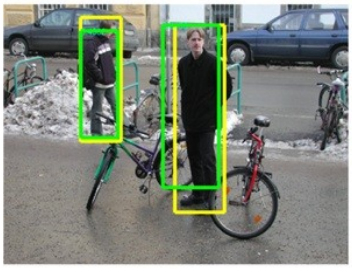


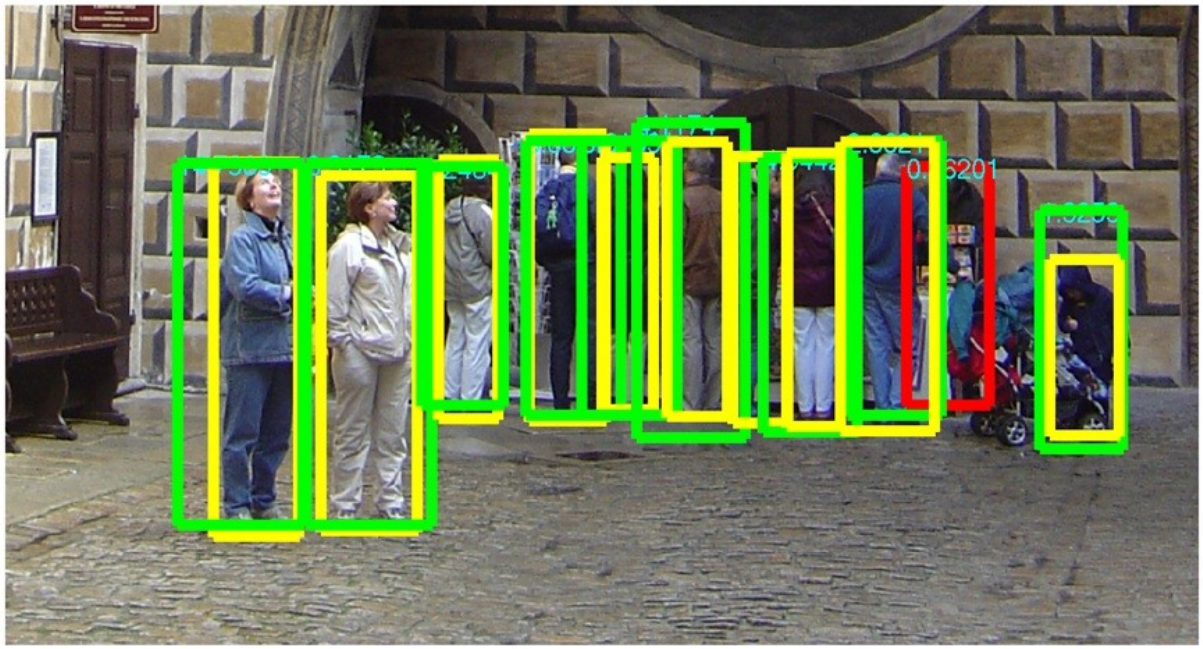
Şekil 3.10. Doğrusal sınıflandırıcılar kullanılarak elde edilen konum bulma sisteminin çıktıları (sol taraftaki resimler) ile doğrusal sınıflandırıcılar ve doğrusal olmayan kernel hiper-küre sınıflandırıcısı kullanılarak elde edilen konum bulma sisteminin çıktılarının (sağ taraftaki resimler) karşılaştırılması.

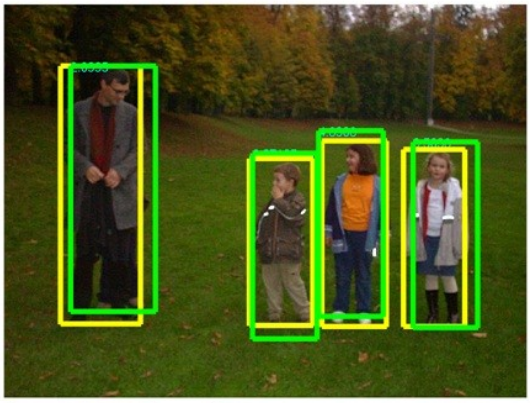


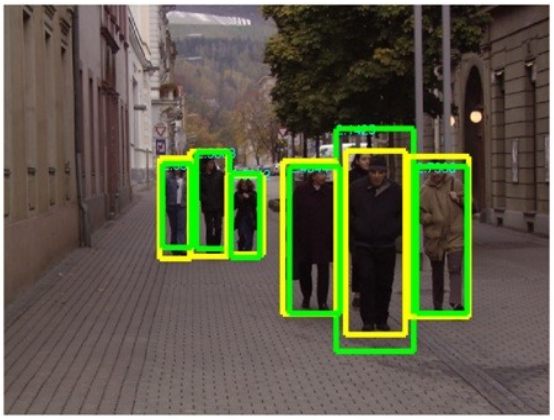


City
City




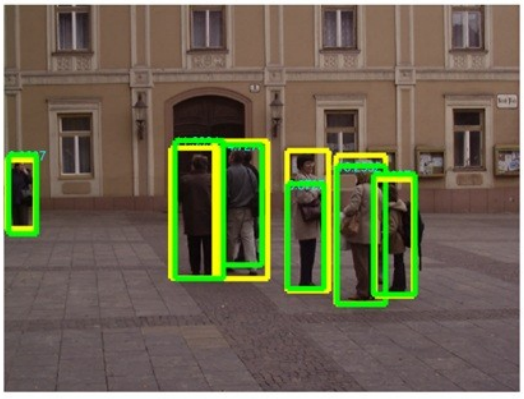


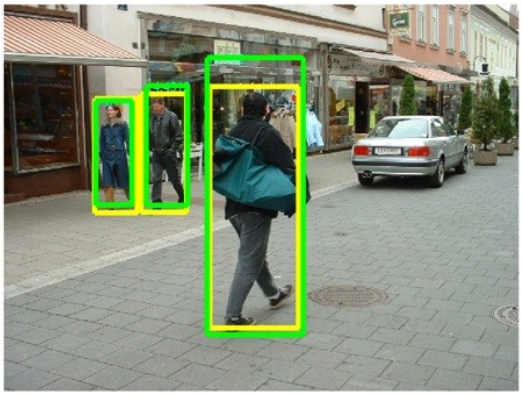



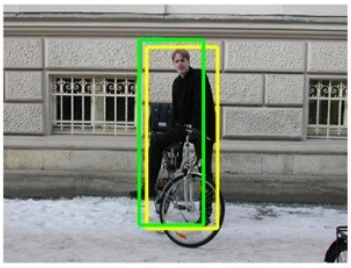


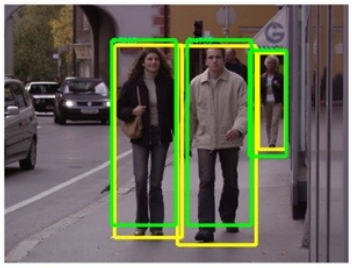


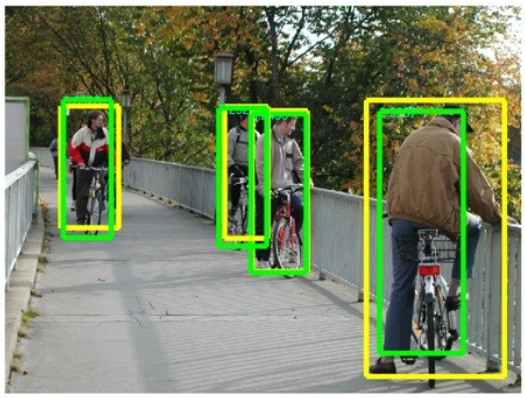


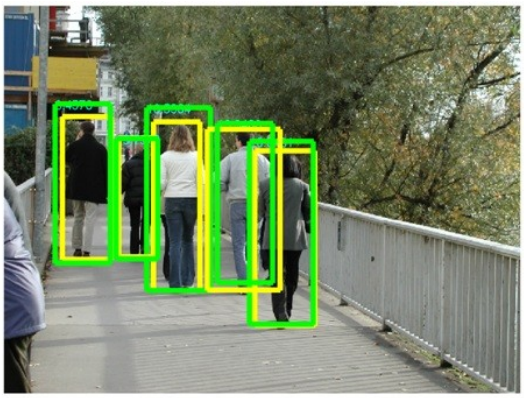

GAZELLE

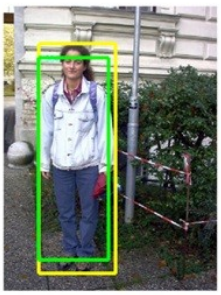


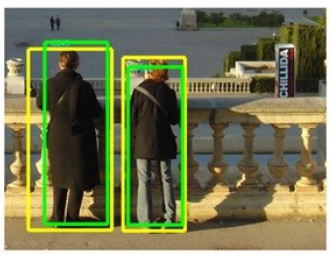


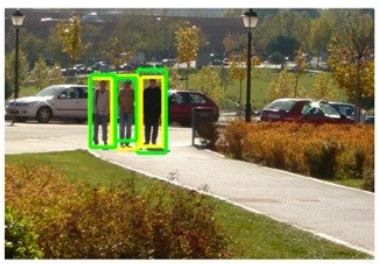


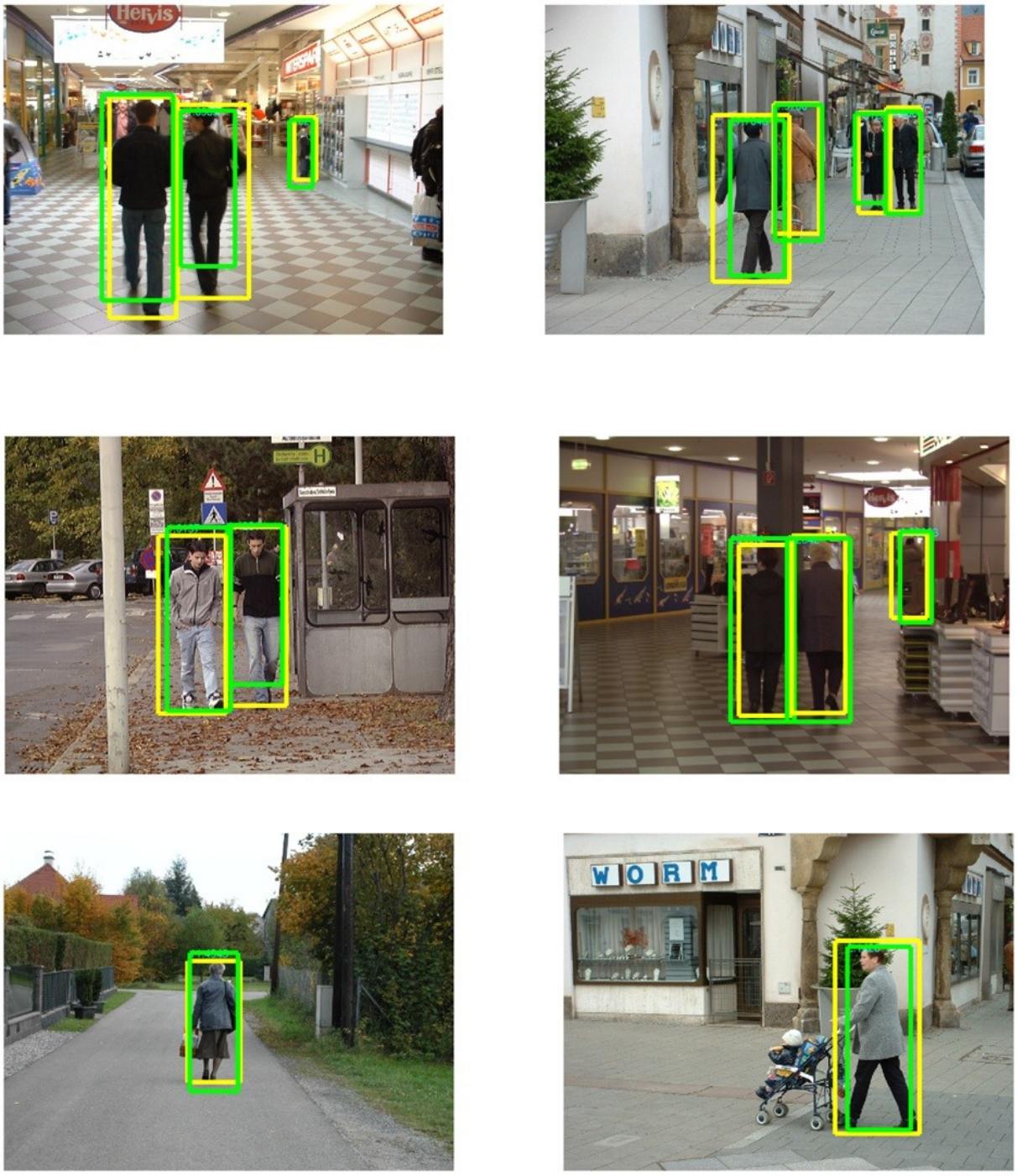




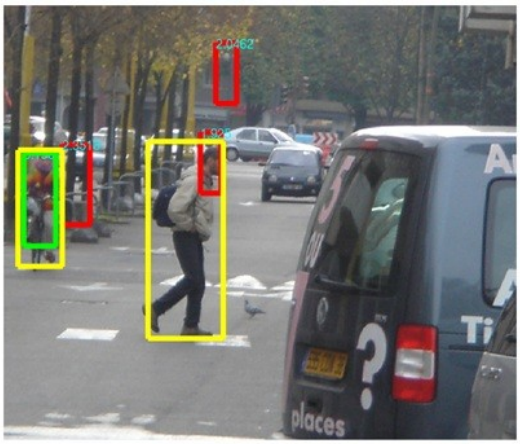




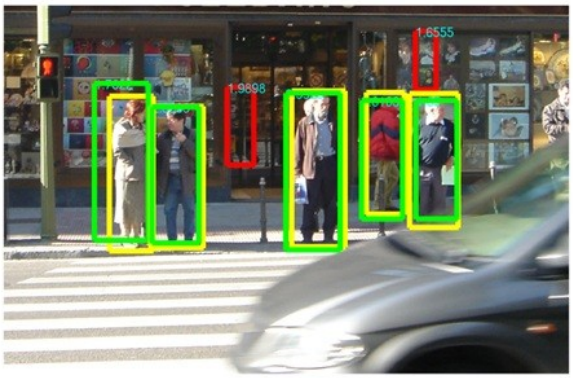


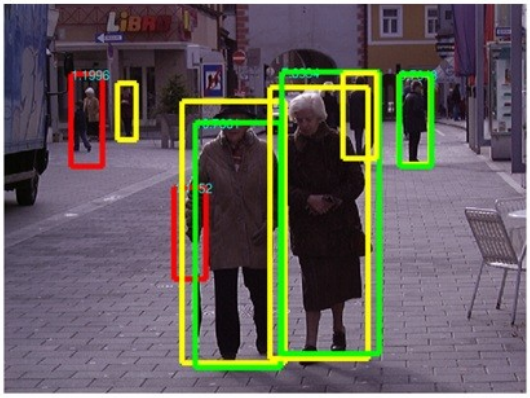


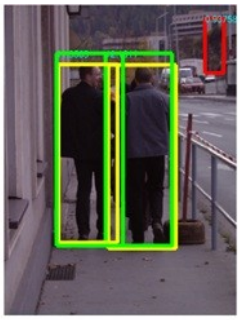
Şekil 3.11. Tüm insanların geliştirdiğimiz ardışıl sınıflandırıcıları kullanan konum bulma sistemi tarafından başarı ile bulunduğu örnekler. Sarı kutular elle işaretlenen doğru konumları yeşil kutular ise sistem tarafından döndürülen ve PASCAL VOC ölçütü kullanılarak doğru olarak kabul edilen konumları göstermektedir. Kırmızı kutular yanlış olarak değerlendirilen konumları göstermekle birlikte bu bölgelerde gerçekten insan resimleri bulunmaktadır ve bunlar veri tabanındaki etiketleme hatalarından ötürüdür.

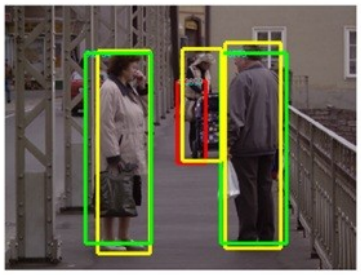

places

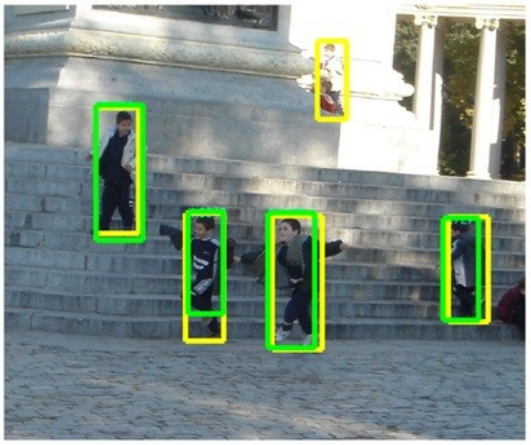

DELTA MOTOR
BITTER
SBARRO
CASTAGNA
2
ALLEE
2100

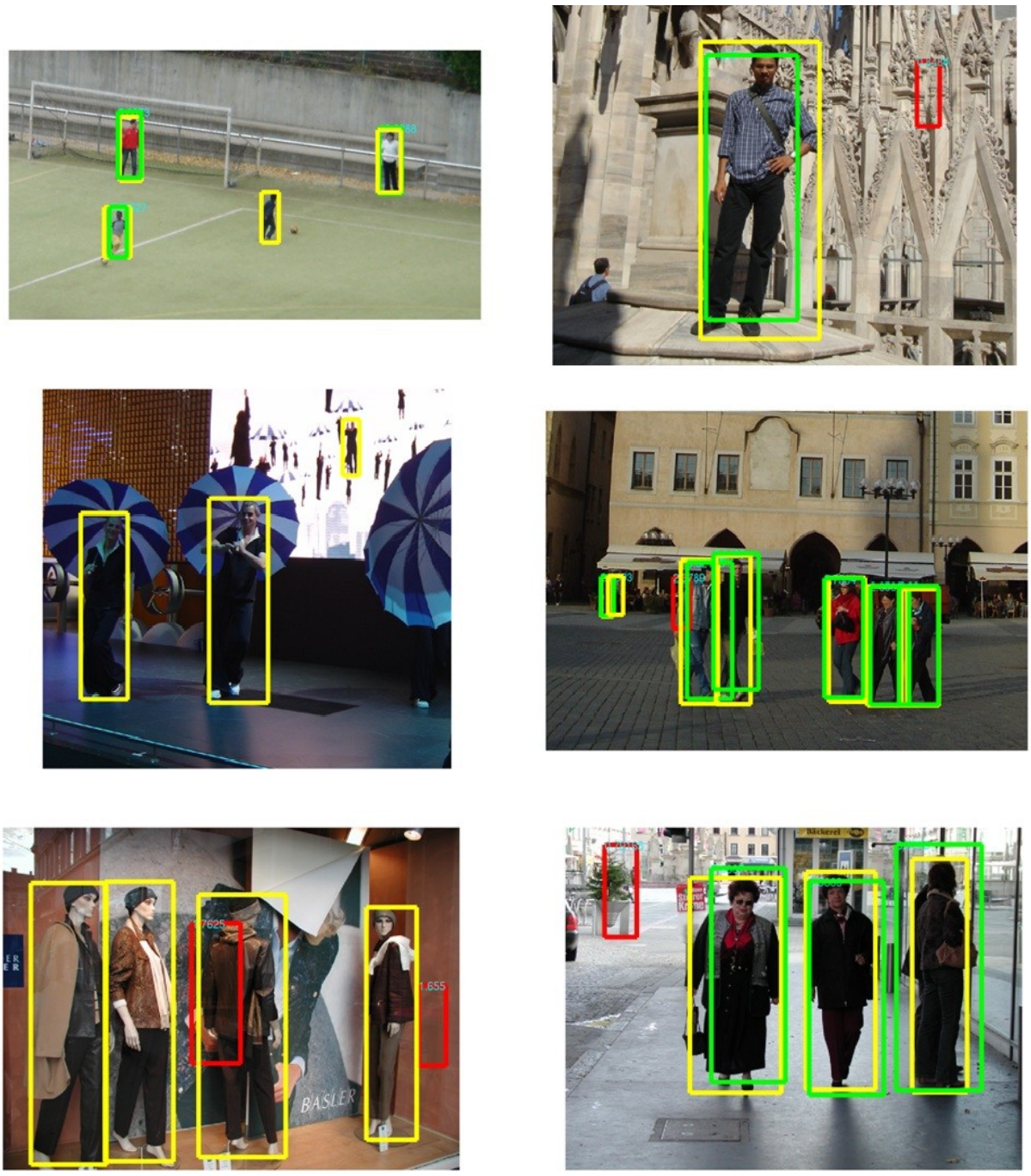












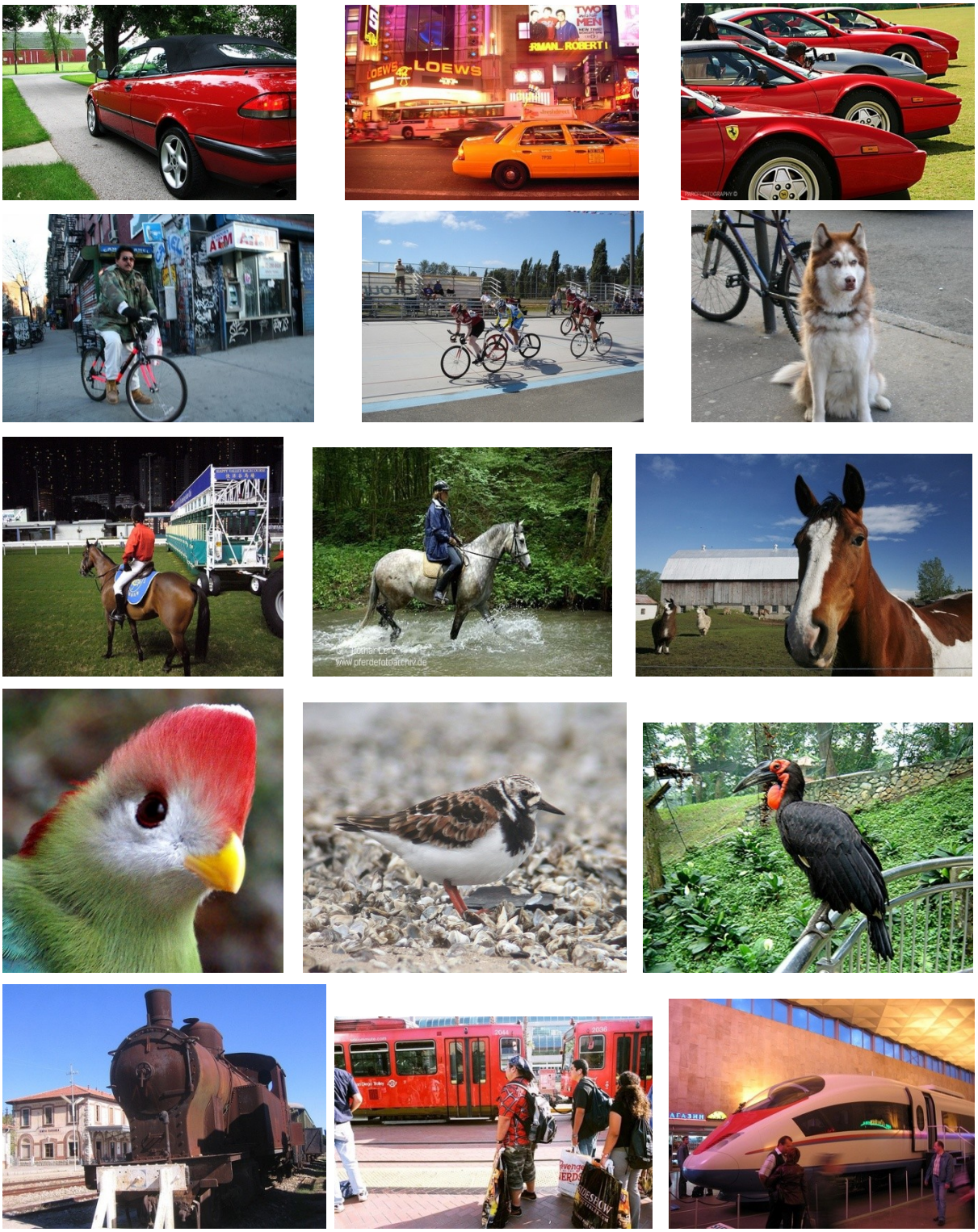
Şekil 3.12. Konum bulma sisteminin hata yaptığı bazı örnekler.

3.3 PASCAL VOC VERİTABANI ÜZERİNDEKİ ÇALIŞMALAR

Geliştirdiğimiz konum bulma sistemini ayrıca 2007 yılındaki PASCAL VOC yarışmasında kullanılan (*http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2007/*) görsel nesne sezimi veri tabanı üzerinde test ettik. Bu veri tabanında 20 adet nesne sınıfı bulunmaktadır ve veritabanında binlerce günlük yaşamdan kareleri yansıtan gerçekçi ve zor imgeler vardır. Veri tabanı eğitim imgeleri ve test imgeleri olmak üzere ikiye ayrılmıştır. Eğitim imgelerinde her bir imgedeki nesnenin koordinatları verilmiştir. Konum bulma sisteminin paremetrelerinin seçimi için eğitim setinde ayrıca geçerleme (validation) örnekleri de ayrılmıştır.

3.3.1 Görsel Nesne Konum Bulma Sisteminin Eğitilmesi

PASCAL VOC veritabanında bir nesneye ait Şekil 3.13'de görüldüğü gibi çok farklı pozlar bulunmaktadır. Bu sebeple bu veri tabanlarındaki nesnelerin konumlarını başarılı bir şekilde bulabilmek için kısmen rigid görünüşe sahip yüz ve ayakta duran insanları bulmak için kullandığımız yaklaşımda yenilikler ve iyileştirmeler yapma zorunluluğu doğmuştur. Farklı pozlara sahip nesnelerin konumlarını bulmak için kullanılan sistem genel itibariyle kaç değişik poz kullanmaya karar verip her değişik poz için rigid nesne bulmak için geliştirilmiş sistemi uygulamadan ibarettir. Bu bağlamda Felzenszwalb'ın [31] önerdiği tekniği kullandık. Buna göre kullanıcı önceden kaç poz kullanılacağını sisteme girmektedir. Bir sonraki aşama ise her bir poz için kayan pencereler yönteminde kullanılacak kök arama penceresinin en-boy (aspect ratio) oranı ile boyutunun belirlenmesidir. Bu amaçla eğitim setinde elle işaretlenen nesne görüntülerini içeren pencerelerin (bounding box) en-boy oranı küçükten büyüğe sıralanıp, bunlar her biri aynı sayı içerecek şekilde seçilen poz sayısına eşit bölgeye ayrılmaktadır (mesela elimizde 300 adet işaretlenmiş nesne görüntüsü varsa ve 3 poz kullanılması istenirse her bir bölge 100 nesne örneğine karşılık gelen en-boy oranlarını içerir). Bu bölgelerin daha sonra histogramları alınarak exponensiyel bir filtre ile yumuşatılmış ve maksimum noktaları bulunmuştur. Bulunan her bir tepe noktasına karşılık gelen değer, o poza ait eğitilecek arama penceresinin en-boy oranı olarak kabul edilmiştir. Bu arama penceresinin boyutu da bu pozla ilişkilendirilen tüm işaretlemelere ait bölgelerin kapladığı alanın ortalamasının 0.80 ile çarpılmış hali olarak kabul edilmiştir. Bu şekilde her bir farklı poza ait kayan penceler yönteminde kullanılacak araştırma penceresinin boyutu belirlenmiştir. Bu yaklaşımdaki en büyük sorun seçilen bölgelerde en-boy oranlarının histogramının birden fazla tepe noktası altında toplanması ihtimalidir. Böyle durumlarda her bir bölge için tek bir poz



Şekil 3.13. PASCAL VOC 2007 veritabanından alınan bazı örnek imgeler. Her bir satırda sırasıyla "car-araba", "bicycle-bisiklet", "horse-at", "bird-kuş", "train-tren" sınıfına ait örnekleri içeren 3 imge verilmiştir. Yukarıda görüldüğü üzere bazı imgelerde birden fazla sınıfa ait örnek olabilir.

almak yerine bu baskın noktalar sayısı kadar poz kabul etmek daha doğru olacaktır. Bu sebeple verilerin yoğunlaştığı noktaları bulmak için "mean-shift" topaklama yöntemi uygulanabilir. Fakat bizim amacımız temelde sınıflandırıcıları karşılaştırma olduğu için Felzenszwalb'ın tekniğinde herhangi bir değişiklik yapmadık. Araştırma pencerelerinin boyutunu belirlemek içinde Felzenszwalb'ın yaptığı gibi her bir pozla ilişkilendirilen pencelerin boyutlarının ortalaması alınmış ve 0.80 ile çarpılmıştır.

Nesne için kullanılacak poz sayısı ve bu pozlara karşılık gelen temel arama pencereleri belirlendikten sonraki aşamada her bir poza karşılık gelen örneklerin öznitelik vektörleri çıkartılarak örnekler sola ve sağa bakan örnekler olarak otomatik olarak ikiye ayrılmışlardır. Öznitelik vektörü olarak Felzenszwalb'ın yönteminde olduğu gibi sadece YGH öznitelikleri kullanılmıştır. Şekil 3.14'de bu yaklaşım kullanılarak ayrıştırılan araba resimleri gözükmektedir. Bu amaçla her bir nesne görüntüsünün y eksenine göre simetriği alınmış ve bu görüntüler arama penceresinin boyutlarına eşit olacak şekilde örneklendikten sonra öznitelikleri çıkartılarak ikili çiftlerin herbir elemanı ayrı topak sınıflarına gelecek şekilde bir topakalandırma yapılmıştır. Bu topaklama sırasında Felzenszwalb "bottom-up" yaklaşımını kullanarak her bir topağa düşen örneklerin ortalamasını bulmuş ve yeni gelen bir ikilide her örnek çiftindeki elemanları en yakın topak merkezine atamıştır ve bu iteratif olarak devam ettirilmiştir. Fakat bu yaklaşım oldukça ilkel olup, özellikle yüksek boyutlu verilerde örneklerin ortalamasını kullanmak iyi bir fikir değildir ([45]'de verilen çalışmamızda bunun nedenleri verilmiştir). Bu sebeple bu algoritmaya alternatif olarak daha sofistifike olan fakat daha fazla işlem yükü gerektiren ve her bir sınıfı konveks zarf ile modellemeye dayalı bir topaklama algoritması önerilmiştir. Bu topaklama algoritmasının detayları "IEEE Conference on Automatic Face and Gesture Recognition" için hazırladığımız bildiride [44] bulunabilir. Fakat burdaki deneylerde yine Felzenszwalb'ın topaklama algoritması kullanılmıştır.

Bu işlemlerden sonra her bir poz için iki farklı topağa ait olan örnekler kullanılarak daha önceden açıkladığımız ardışıl sınıflandırıcılar bazı değişiklikler yapılarak eğitilmiştir. Daha önceki deneylerde olduğu gibi ardışıl sınıflandırıcıda 4 katman kullanılmıştır. İlk katmanında hızlı bir sınıflandırıcı olan doğrusal Destek Vektör Makineleri kullanıldı. Bu katmanın amacı nesneye ait hemen hemen tüm örnekleri geçirerek, arka-plana ait örneklerin bir çoğunu elemektir. İkinci katmanda ise nesne sınıfını doğrusal bir hiper-düzlem ile modelleyen sınıflandırıcı kullanılmıştır. Bu sınıflandırıcı da oldukça hızlı olup Destek Vektör Makineleri ile uyumlu bir şekilde çalışarak ilk katmandan geçen bir çok arka-plan görüntüsünü elemektedir. Üçüncü katmanda ise nesne sınıfına ait örnekler doğrusal bir hiper-

küre ile yakınsanmıştır. Son katmanda ise doğrusal olmayan DVM sınıflandırıcısı kullanılmıştır. Bu veri tabanında her poz için nesne sınıfına ait çok az sayıda örnek olması ve örneklere ait öznitelik vektörlerinin boyutunun örnek sayısına göre çok büyük olması sebebiyle doğrusal olmayan hiper-küre sınıflandırıcısı çok iyi sonuçlar vermemiştir. Bu durumun nedenleri tartışma bölümünde ayrıntılı bir şekilde açıklanmıştır. Doğrusal olmayan DVM sınıflandırıcısı çok fazla destek vektörü döndürdüğünden sistemi hızlandırmak için Şekil 2.6'da verilen algoritma kullanılarak destek vektör sayısı her farklı poz için 400'e indirilmiştir.

Şekil 3.14. Önerdiğimiz topaklama algoritması ile otomatik olarak ayrıştırılan sağa bakan ve sola bakan araba örnekleri.

Yukarıda ayrıntıları verilen sistemi eğitmek için "latent training" olarak adlandırılan ve iteratif olarak kendini en iyileyen bir yaklaşım kullanılmıştır. Bu yaklaşıma göre ilk olarak sınıflandırıcılar eğitim setindeki elle işaretlenmiş nesne görüntüleri ve rastgele seçilen arka-plan görüntüleri kullanılarak eğitilmiştir. Daha sonra eğitilen konum bulma sistemi nesne görüntülerini içeren imgelerde nesnenin görüntüsünün olduğu bölgelerin etrafına uygulanmış ve konum bulma algoritmasının nesne görüntüsü olarak döndürdüğü pencerelerin elle işaretlenen bölgelerle kesişmesi kontrol edilmiştir. Kesişme oranları %70'in üzerinde olanlar içinde en yüksek skora sahip bölge nesneye ait örnek olarak etiketlenmiş ve bu işlem tüm görüntüler için tekrarlanmıştır. Kesişme oranı %70'in altında kalanlar ise aykırı değer olarak kabul edilmiştir ve bu görüntüler nesneye ait örnekler içinde yer almamışlardır. Daha sonra konum bulma algoritması içinde nesne olmayan görüntülere de uygulanarak literatürde "hard negatives" olarak adlandırılan ve nesne sınıfına ait olmadığı halde nesne sınıfına atanan örnekler toplanmıştır. Daha sonra sınıflandırıcılar konum bulma algoritmasının seçtiği örnekler kullanılarak tekrar eğitilmişlerdir. Bu işlem iteratif olarak belirlenen bir sayıda tekrarlanarak sistemin her iterasyonda kendini iyileştirmesi sağlanmıştır. Bu strateji nesne görüntülerinin farklı kişiler tarafından ve bazen yanlış olarak işaretlenmesinden kaynaklanan sorunları giderdiği gibi, aykırı değerleride ayıkladığından dolayı çok iyi sonuçlar vermiştir.

3.3.2 Deneysel Sonuçlar

PASCAL VOC 2007 yılına ait veritabanında elde edilen sonuçlar Tablo 3.4'de verilmiştir. Yöntemimizi o yılki yarışmada alınan en iyi sonuçlar ve Felzenszwalb'ın geliştirdiği yöntemlerin sonuçlarıyla karşılaştırdık. 2007 yılki yarışmada farklı bilgisayarlı görü gruplarının geliştirdiği 9 farklı konum sistemi yarışmıştır. Tabloda farklı kategoriler için en iyi sonucu veren yöntemin elde ettiği başarım verilmiştir. Tabloda Felzenszwalb I olarak gösterilen yöntemin sonuçları [31]'de yapılan çalışmadan alınmış olup bu çalışmada 2 farklı poz kullanılmış olup yöntemde hem kök seziciler hem de parça seziciler kullanılmıştır. Biz yöntemimizde poz sayısını 3 aldığımızdan daha iyi bir karşılaştırma için Felzenszwalb tarafından paylaşıma açılmış kodları kullanarak Felzenszwalb'ın yöntemin sonuçlarını 3 poz kullanarak elde ettik. 3 poz kullanarak ve sadece kök seziciler kullanarak elde edilen başarım tabloda Felzenszwalb II'nin altında verilmiştir. Hem kök seziciler hem de parça seziciler kullanarak elde edilen sonuçlar ise Felzenszwalb II+Parça Seziciler'in altında verilmiştir. Biz önerdiğimiz yöntemde iş istasyonumuzun hafızasının yetersiz kalması sebebiyle sadece kök sezicileri kullandık ve bu sebeple sınıflandırıcı başarımlarını doğru bir şekilde kıyaslamak için aldığımız sonuçların Felzenszwalb II sütunu ile karşılaştırılması gerekir.

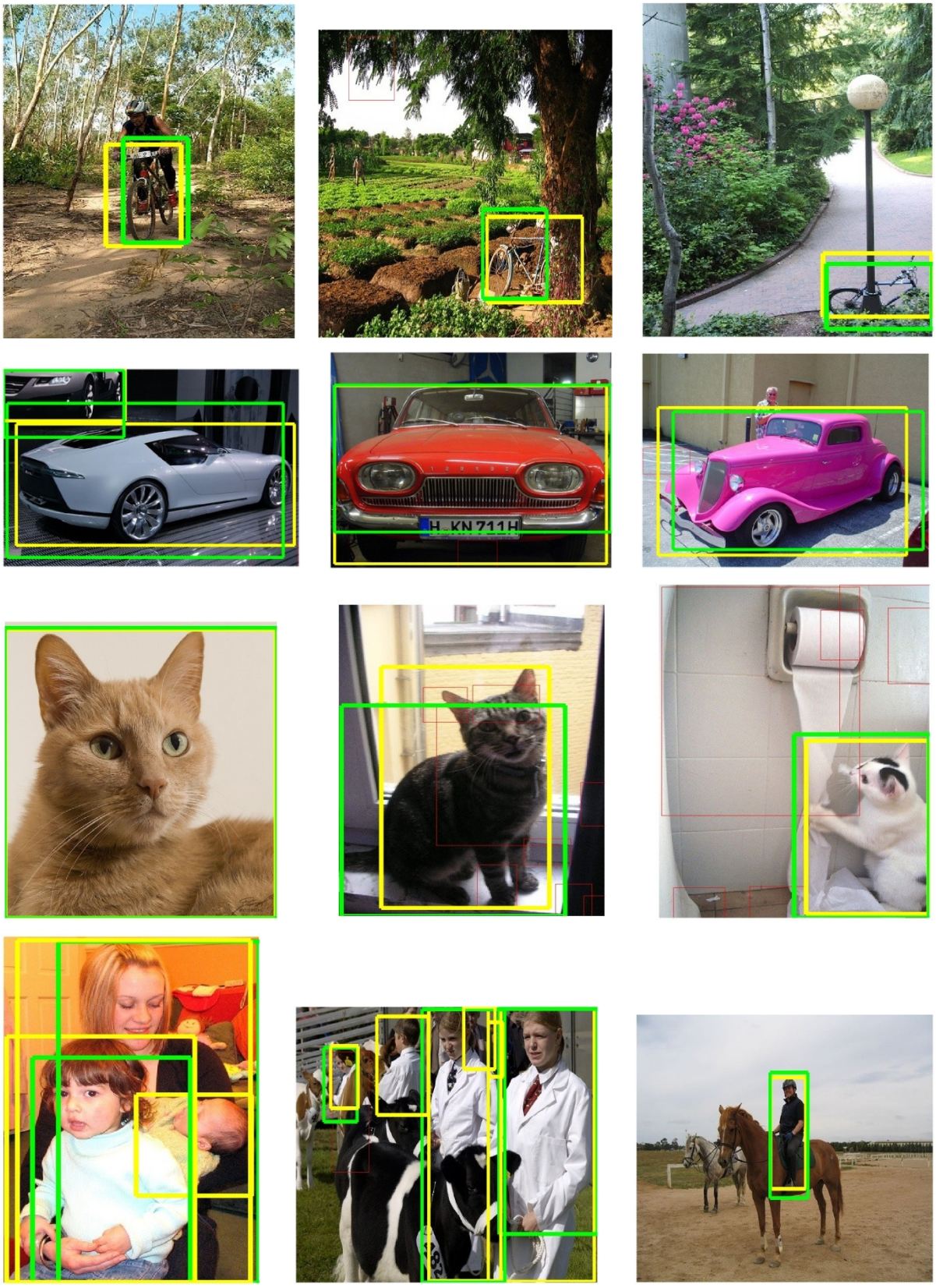
Konum bulma yöntemlerinin başarımlarının düşük çıkmasındaki en önemli nedenlerden biri de görsel olarak benzer nesne kategorilerinin birbiriyle karıştırılmasından kaynaklanmaktdır. Örnek olarak "horse" sınıfındaki örnekler genellikle "cow" ve "sheep" sınıfındaki örneklerle karıştrılırken "car", "bus", ve "train" kategorilerindeki örnekler birbiriyle karıştırılmıştır. "Bicycle" ve "motorbike" sınıflarının örnekleri de en fazla karıştırılan örneklerdendir. Bu sebeple 2007 yılında önerilen yöntemler benzer sınıflar arasında daha iyi ayırım yapmak için imgenin tamamınından gelen bilgileri de (context bilgisi) kullanmışlardır. Buna benzer bir yöntem de [31]'de Felzenszwalb ve diğerleri tarafından önerilmiş ve performansta artış elde edilmiştir. Biz bu tür bir yöntem kullanmadığımız için Felzenszwalb'ın geliştirdiğ yöntemlerde bu bilgi kullanılarak elde edilen sonuçları vermedik, fakat bu sonuçlar [31]'de verilmiştir.

Önerdiğimiz yöntem 2007 yılındaki yarışmada elde edilen en iyi sonuçlarla karşılaştırıldığında, yöntemimiz 20 nesne kategorisi arasında 13 kategoride daha iyi sonuçlar vermiştir. Aynı şekilde 3 poz kullanarak eğitilen sistemiz ile 3 poz kullanılarak eğitilen ve sadece kök sezicileri kullanan Felzenszwalb yöntemi (Felzenszwalb II) karşılaştırıldığında

Tablo 3.4. Önerilen Yöntemin PASCAL VOC 2007 veri tabanındaki başarı oranları. Değerler ortalama kesinlik skorları (%) olup Precision-Recall eğrilerinin altında kalan alanların hesaplanmasıyla bulunmuşlardır.

Nesne Kategorileri	**Önerilen Yöntem**	**2007 Yılı En İyi Sonuçlar**	**Felzenszwalb I [31]**	**Felzenszwalb II**	**Felzenszwalb II+Parça Sez.**
Aeroplane	23.90	26.20	29.00	19.10	**28.50**
Bicycle	46.40	40.90	54.00	47.50	**57.20**
Bird	**9.90**	9.80	0.60	2.30	3.00
Boat	11.60	9.40	13.40	13.30	**17.10**
Bottle	18.80	21.40	**26.20**	14.90	24.90
Bus	39.70	39.30	39.40	40.40	**47.50**
Car	46.20	43.20	46.40	41.80	**53.90**
Cat	11.40	**24.00**	16.10	6.70	13.60
Chair	17.90	12.80	16.30	16.20	**22.10**
Cow	**25.10**	14.00	16.50	21.00	**25.10**
Dining Table	17.10	9.80	**24.50**	15.30	20.40
Dog	4.90	**16.20**	5.00	10.60	3.90
Horse	40.40	33.50	43.60	40.10	**57.50**
Motorbike	36.50	37.50	37.80	34.10	**47.70**
Person	33.10	22.10	35.00	30.10	**42.40**
Potted Plant	9.50	12.00	8.80	11.30	**12.20**
Sheep	13.50	**17.50**	17.30	17.00	17.40
Sofa	16.80	14.70	21.60	21.90	**31.50**
Train	36.70	33.40	34.00	34.10	**44.60**
TV Monitor	39.40	28.90	39.00	30.00	**40.60**

yöntemimiz “bicycle”, “boat”, “bus”, “dog”, “potted-plant”, “sheep” ve “sofa” nesne kategorileri dışında diğer 13 nesne kategorisinde daha iyi sonuçlar vermiştir. Tabloda görüldüğü üzere parça sezicilerin kullanımı başarımı bir çok nesne kategorisinde önemli ölçüde arttırmış ve en iyi sonuçlar genellikle parça sezicileri kullanan Felzenszwalb’ın yöntemiyle elde edilmiştir. Buna rağmen geliştirdiğimiz yöntem iki kategoride (“bird” ve “cow” kategorileri) tüm yöntemler arasında en iyi sonuçları vermiştir.



Şekil 3.15. Konum bulma sisteminin PASCAL VOC 2007 veritabanındaki “bicycle”, “car”, “cat” ve “people” kategorileri için çıktıları.

Şekil 3.15'de geliştirdiğimiz konum bulma sistemimizin PASCAL VOC 2007 veritabanından gelen bazı imgeler üzerindeki çıktıları görülmektedir. Bir çok kategorideki düşük başarımların en büyük nedeni sistemin döndürdüğü pencere ile elle işaretlenen pencere arasındaki örtüşmenin PASCAL VOC metriğine göre yetersiz kalmasıdır. Ayrıca sistem imgenin kenarlarında kalan kesilmiş nesne görüntüleri ile başka nesneler tarafından kapatılmış nesneleri başarılı bir şekilde bulamamaktadır. Bunlara ek olarak sistem, belirlenen arama penceresinden daha küçük nesneleri bulamamaktadır. Başarımın düşük olma nedenlerinden biri de görsel olarak birbirine benzer nesne kategorileri örneklerinin karıştırılmasıdır.

3.4 TARTIŞMA VE VARGILAR

Geliştirdiğimiz yöntem yüz ve insan (ayakta duran) sezme deneylerinde çok başarılı sonuçlar vererek literatürdeki en başarılı yöntemlerden daha iyi sonuçlar verirken PASCAL VOC 2007 veritabanı üzerinde çok başarılı sayılacak sonuçlar alınamamıştır (Bu veri tabanında başarılı sayılacak sonuçlar doğrusal olmayan DVM kullanılarak elde edilmiştir). Bunun sebebi kullandığımız tek sınıflı sınıflandırıcıların karakteristiği ile ilgilidir. Kullandığımız ardışıl sınıflandırıcıda başarımı en çok etkileyen sınıflandırıcı doğrusal olmayan hiper-küre sınıflandırıcısıdır. Nesne sınıflarına ait örnekler öznitelik uzayında doğrusal olmayan bir manifold oluştururlar (Bu manifold kesinlikle doğrusal değildir. Rigid nesne olarak kabul ettiğimiz yüz sınıfında bile poz değişklikleri, dönme ve diğer etkilerden dolayı örneklerin giriş uzayında uzandığı sınırlar doğrusal değildir). Nesne sınıflarına ait bu manifoldların sınırlarının doğru olarak yakınsanabilmesi için özellikle sınırların olduğu bölgelerde çok sayıda biribirine yakın örneğe ihtiyaç vardır (manifold must be well-sampled and smooth). Jain ve diğerleri [46] bu tür manifoldları doğru bir şekilde yakınsayabilmek için örnek sayısının örnek uzay boyutundan en az 10 kat fazla olması gerektiğini bildirmişlerdir. Yüz sezme deneylerinde öznitelik vektörlerine ait uzay boyutu her zaman 1000'in altındadır ve yüz sınıfına ait 20K'dan fazla örnek kullanılmıştır. Yani nesne grubuna ait örnek sayısı örnek uzay boyutundan en az 20 kat daha fazladır. Buna bağlı olarak bu veri tabanı üzerinde çok iyi sonuçlar elde ettik. Inria Person veri tabanında ise çok fazla örnek olmamasına rağmen (örnek sayısı 1000 civarındadır) nesne grubuna ait resimlerin koordinatları rassal olarak küçük miktarlarda kaydırılarak bir örnekten çok sayıda yapay örnek elde edilmiş ve eğitim sırasında bu örneklerde kullanılmıştır. Böylece örnek sayısının örnek uzay boyutundan daha fazla olması sağlanmıştır. Fakat bu yaklaşımlar PASCAL VOC 2007 veritabanı üzerinde işe yaramamıştır. Öncelikle bu veri tabanında nesnelere ait örnek sayısı genel itibariyle 1000'den

düşüktür ve bu örnekler 3 farklı poza bölüştürüldükten sonra her bir poza düşen örnek sayısı 300'den çok daha az olmuştur. Ayrıca bu örnekler arasında çok fazla sayıda en-boy oranı olarak aynı gruba düşen fakat poz olarak birbirine benzemeyen ve aykırı değer olarak değerlendirilmesi gereken örnek vardır. Tüm bu sebeplerden dolayı PASCAL VOC 2007 veritabanındaki nesne gruplarını yakınsamada doğrusal olmayan hiper-küre sınıflandırıcısı pek iyi sonuçlar vermezken doğrusal olmayan DVM sınıflandırıcısı çok daha iyi sonuçlar vermiştir. Tüm bu sonuçlar geliştirdiğmiz konum bulma sisteminin veri sayısının fazla olduğu uygulamalar için uygun olduğunu göstermektedir. Bu tür uygulamalarda daha az sayıda destek vektör döndüren doğrusal olmayan hiper-küre sınıflandırıcısı önerdiğimiz konum bulma sisteminin diğer doğrusal olmayan sınıflandırıcıların kullanıldığı diğer konum bulma sistemlerinden çok daha hızlı çalışmasına olanak vermektedir.

4. SONUÇ VE ÖNERİLER

Bu çalışmada dijital imgelerdeki nesnelerin konumlarını bulmak için tek sınıflı ve iki sınıflı sınıflandırıcılardan oluşan ardışıl bir sınıflandırıcı sistemi geliştirdik. Ardışıl sınıflandırıcı genel itibariyle 4 katmandan oluşmaktadır. İlk katmanda iki-sınıflı doğrusal DVM sınıflandırıcısı kullanılırken, ikinci ve üçüncü katmanda sırasıyla tek sınıflı sınıflandırıcılar olan doğrusal hiper-düzlem ile hiper-küre sınıflandırıcıları kullanılmıştır. Son katmanda ise doğrusal sınıflandırıcılara oranla daha yavaş fakat başarımı yüksek olan doğrusal olmayan hiper-küre sınıflandırıcısı kullanılmıştır. Doğrusal olmayan hiper-küre sınıflandırıcısı doğrusal olmayan DVM sınıflandırıcısından çok daha az sayıda destek vektör döndürdüğünden oldukça hızlıdır. Fakat bu yöntemin başarılı bir şekilde çalışması için örnek uzay boyutuna oranla çok daha fazla sayıda örneğe ihtiyaç vardır. Örnek sayısının az olduğu durumlarda DVM sınıflandırıcısı daha iyi sonuçlar vermiştir. Günümüzde internet ortamındaki resim sayısı dikkate alındığında görsel nesne gruplarına ait çok sayıda örnek toplamak mümkündür ve geliştirdiğimiz yöntem ticari ve ciddi konum bulma uygulamaları için son derece elverişlidir.

İlk olarak tek pozla çalışacak şekilde geliştirilen yöntem daha sonra çok sayıda pozla çalışacak şekilde de geliştirilmiştir. Yöntemimiz Felzenszwalb ve diğerleri [31] tarafından önerilen yaklaşım kullanılarak nesne parçalarınıda kullanacak şekilde geliştirilmesine rağmen parça seziciler hafıza kısıtları yüzünden tam olarak test edilememiştir. Fakat özellikle parça sezicileri eğitmek için çok etkili bir sınıflandırıcı geliştirilmiş ve kullanıcıların paylaşımına açılmıştır.

Geliştirdiğimiz konum bulma yönteminde iyileştirmek yapmak ve bu çalışmada test edilen veri tabanlarındaki başarım oranlarını arttırmak mümkündür. İlk olarak yöntem kısmında açıklandığı gibi nesne sınıflarını yakınsayan daha iyi hiper-düzlemler bulmak için (8)'de verilen eniyileme problemine konveks olmayan kısıtlar eklenerek bu problemi fazla veriye rağmen çözebilecek algoritmalar geliştirilebilir. Bu şekilde elde edilecek hiper-düzlem nesne sınıfını en iyi şekilde modellerken arka-plan ait örneklerden de oldukça uzakta olacaktır. Bunun yanında nesne sınıflarını modellemek için hiper-düzlem ve hiper-kürelerin dışında yeni modeller de geliştirilebilir. Nesne parçalarını etkili bir şekilde kullanmak da sistemin başarımını arttıracaktır. Özellikle PASCAL VOC 2007 nesne veri tabanında en iyi sonuçlar genellikle kök ve parça sezicileri birlikte kullanan Felzenszwalb yöntemiyle elde edilmiştir. Fakat bu yöntem özellikle veri sayısının çok fazla olduğu ticari uygulamalar için uygun değildir. Bu yöntemde hem kök sezici hem de parça sezicilere ait öznitelikler arka-

arkaya bağlanmış ve çok büyük boyutlu öznitelik vektörleri elde edilmiştir. Veri sayısının fazla olduğu durumlarda bu şekilde oluşturulan verileri bilgisayarın hafızasına sığdırmak mümkün olmayabilir. Bu yaklaşım yerine bu projede denendiği gibi öncelikle kök seziciler kullanılarak tüm nesneyi içeren olası bölgeler bulunup daha sonra bu bölgeler içinde nesne parçalarının varlığı araştırılabilir. Ayrıca parçaları modellemek için çok farklı yaklaşımlar kullanmak mümkündür. Özellikle parçalar seçilirken eğitim setindeki veriler kullanılarak arka-plana göre daha ayırt-edici parçalar seçilebilir. Son olarak başarımı etkileyen en önemli faktörlerden biri de benzer nesne kategorilerinin birbiriyle karıştırılmasıdır. Bu tür turumlarda imgenin tamamından gelen bağlam (context) bilgilerini kullanmak faydalı olacaktır. Mesela dış görünüş olarak birbirine benzeyen “sheep” ve “dog” kategorilerindeki örnekler bu hayvanların genellikle farklı ortamlarda bulunmasından dolayı bağlam bilgisi kullanılarak daha sağlıklı bir şekilde ayırt-edilebilir.

5. KAYNAKLAR

[1] Rowley H. A., Baluja S., Kanade T., Neural network-based face detection, *IEEE Transactions on PAMI*, vol. 20, pp. 23–38, 1998.

[2] Papageorgiou C., Poggio T., A trainable system for object detection, *International Journal of Computer Vision*, vol. 38, pp. 15–33, 2000.

[3] Amit Y., Geman D., A computational model for visual selection, *Neural Computation*, vol. 11, pp. 1691–1715, 1999.

[4] Shams L., Speslstra J., Learning Gabor-based features for face detection, *World Congress in Neural Networks*, 1996.

[5] Lowe D.G., Distinctive image features from scale invariant keypoints, *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[6] Bay H., Ess A., Tuytelaars T., Van Gool L., Surf: Speeded up robust features, *Computer Vision and Image Understanding*, vol. 110(3), pp. 346–359, 2008.

[7] Dalal N., Triggs B., Histograms of oriented gradients for human detection, *IEEE Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[8] Vedaldi A., Gulshan V., Varma M., Zisserman A., Multiple kernels for object detection, *International Conference on Computer Vision*, 2009.

[9] Belongie S., Malik J., Puzicha J., Shape matching and object recognition using shape contexts, *IEEE Transactions on PAMI*, vol. 24(24), pp. 509–521, 2002.

[10] Levi K., Weiss Y., Learning object detection from a small number of examples: the importance of good features, *IEEE Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[11] Ahonen T., Hadid A., Pietikainen M., Face description with local binary patterns: Application to face recognition, *IEEE Transactions on PAMI*, vol. 28(12), pp. 2037–2041, 2006.

[12] Wang X., Han T.X., Yan S., A HOG-LBP human detector with partial occlusion handling, *International Conference on Computer Vision*, 2009.

[13] Tan X., Triggs B., Enhanced local texture feature sets for face recognition under difficult lighting conditions, *IEEE Transactions on Image Processing*, vol. 19, pp. 1635–1650, 2010.

[14] Harzallah H., Jurie F., Schmid C., Combining efficient object localization and image classification, *International Conference on Computer Vision*, 2009.

[15] Hussain S., Triggs B., Feature sets and dimensionality reduction for visual object detection, *British Machine Vision Conference*, 2010.

[16] Varma M., Ray D., Learning the discriminative power-invariance trade-off, *International Conference on Computer Vision*, 2007.

[17] Viola P., Jones M.J., Robust real-time face detection, *International Journal of Computer Vision*, vol. 57(2), pp. 137–154, 2004.

[18] Perrotton X., Sturzel M., Roux M., Implicit hierarchical boosting for multi-view object detection, *IEEE Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[19] Felzenszwalb P., McAllester D., Ramanan D., A discriminatively trained, multiscale deformable part model, *IEEE Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[20] Aldavert D., Ramisa A., Mantaras R.L., Toledo R., Fast and robust object segmentation with the integral linear classifier, *IEEE Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[21] Jin H., Liu Q., Lu H., Face detection using one-class-based support vectors, *International Conference on Automatic Face and Gesture Recognition*, 2004.

[22] Cevikalp H., Triggs B., Efficient object detection using cascades of nearest convex model classifiers, *IEEE Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[23] Porikli F., Integral histogram: A fast way to extract histograms in Cartesian spaces, *IEEE Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[24] Sizintsev M., Derpanis K.G., Hogue A., Histogram-based search: A comparative study, *IEEE Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[25] Wei Y., Tao L., Efficient histogram-based sliding window, *IEEE Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[26] Lampert C.H., Blaschko M.B., Hofmann T., Beyond sliding windows: Object localization by efficient subwindow search, *IEEE Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[27] Hussain S., *Machine learning methods for visual object detection*, Doktora Tezi, Laboratoire Jean Kuntzmann, 2011.

[28] Schnitzspan P., Fritz M., Roth S., Schiele B., Discriminative structure learning of hierarchical representations for object detection, *IEEE Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[29] Gall J., Lempitsky V., Class-specific hough forests for object detection, *IEEE Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[30] Leibe B., Leonardis A, Schiele B., Robust object detection with interleaved categorization and segmentation, *International Journal of Computer Vision*, vol. 77, pp. 259–289, 2008.

[31] Felzenszwalb P., Girshick R.B., McAllester D., Ramanan D., Object detection with discriminatively trained part based models, *IEEE Transactions on PAMI*, vol. 32(9), pp. 1627-1645, 2010.

[32] Zhu L., Chen Y., Yuille A., Freeman W., Latent hierarchical structural learning for object detection, *IEEE Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[33] Tax D. M. J., Duin R. P. W., Support vector data description, *Machine Learning*, vol. 54, pp. 45–66, 2004.

[34] Mangasarian O.L., Wild E.W., Multisurface proximal support vector machine classification via generalized eigenvalues, *IEEE Transactions on PAMI*, vol. 28, pp. 69–74, 2006.

[35] Platt J.C., Fast training of support vector machines using sequential minimal optimization, Advances in Kernel Methods-Support Vector Learning, Cambridge, MA, MIT Press, 1998.

[36] Schölkopf B., Mika S., Burges C. J. C., Knirsch P., Müller K.R., Ratsch G., Smola A.J., Input space versus feature space in kernel-based methods, *IEEE Transactions on Neural Networks*, vol. 10, pp. 1000–1017, 1999.

[37] Mika S., Schölkopf B., Smola A., Müller K.R., Scholz M., Ratsch G., Kernel pca and de-noising in feature spaces, *Neural Information Processing Systems (NIPS)*, 1999.

[38] Burges C. J. C., Simplified support vector decisions, *International Conference on Machine Learning*, 1996.

[39] Cortes C., Vapnik V., Support vector networks, *Machine Learning*, vol. 20, pp. 273-297, 1995.

[40] Scherier W. J., Rocha A., Sapkota A., Boult T.E., Towards open set recognition, *IEEE Transactions on PAMI*, 2012 (online publishment).

[41] Schölkopf B., Platt J., Shawe-Taylor J., Smola A., Williamson R., Estimating the support of a high-dimensional distribution, Microsoft Research, Technical Report, 1999.

[42] Zhu X., Ramanan D., Face detection, pose estimation, and landmark localization in the wild, *IEEE Society Conference on Computer Vision and Pattern Recognition*, 2012.

[43] Kalal Z., Matas J., Mikolajczyk K., Weighted sampling for large scale boosting, *British Machine Vision Conference*, 2008.

[44] Cevikalp H., Triggs B., Franc V., Face and landmark detection by using cascade of classifiers, *International Conference on Automatic Face and Gesture Recognition*, 2013.

[45] Cevikalp H., Triggs B., Nearest hyperdisk methods for high-dimensional classification, *International Conference on Machine Learning*, 2008.

[46] Jain A. K., Duin R. P. W., Mao J., Statistical pattern recognition, *IEEE Transactions on PAMI*, vol. 22, pp. 4-37, 2000.