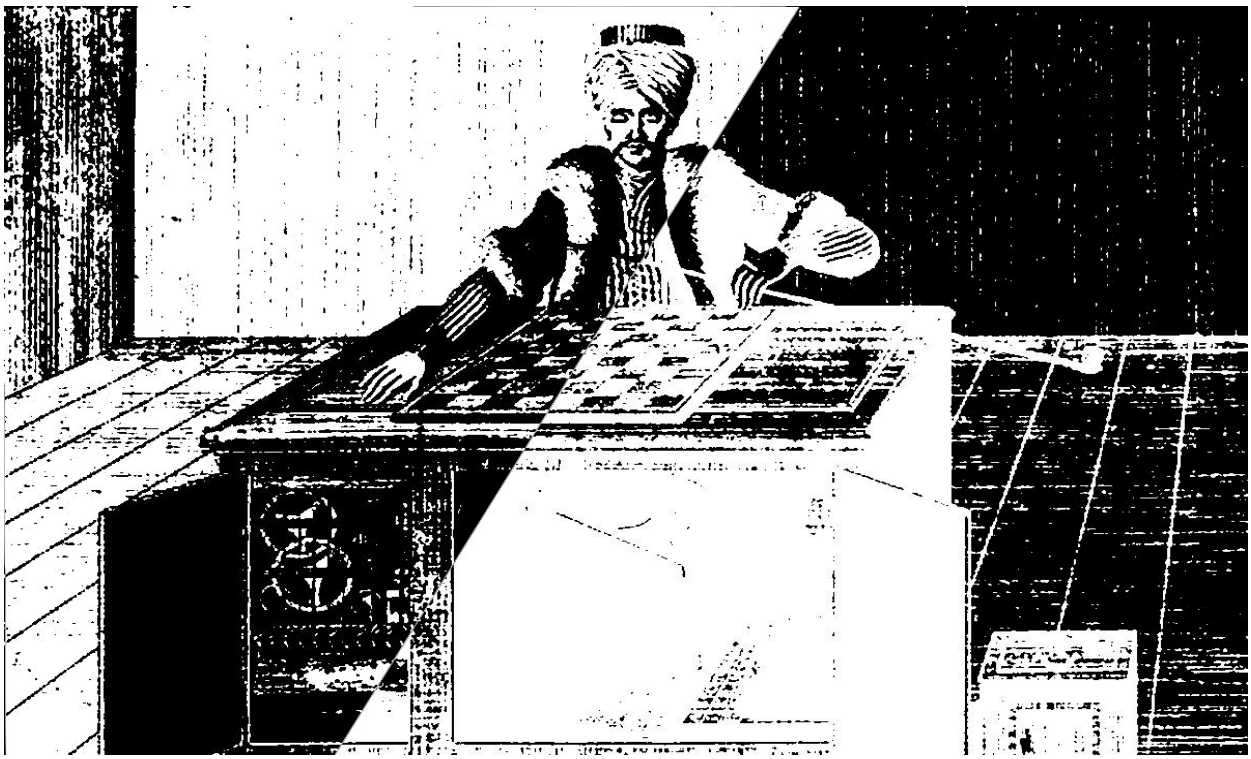


Real Time Nesne Tespiti

13.12.2018

Mehmet Çağrı Yonca
Karadeniz Teknik Üniversitesi
Bilgisayar Mühendisliği

Giriş

Projede resim görüntüsü üzerinde madeni paraların tespiti ve tanınması amaçlanmaktadır.

Proje Kapsamında Kullanılanlar

1. Ubuntu işletim sistemi üzerinde Vim text editörü.
2. Python ile birlikte görüntü işleyebilmek için gerekli donanımlara sahip olan OpenCV frameworku.
3. Yazılan kodların terminal üzerinde realtime test edilebilmesi için Ipython client.
4. Derin öğrenme için Google Cloud Engine.

Proje Aşamaları

Proje öncelikle planlandırıldı. Plana uygun olarak derin öğrenme ve görüntü işleme yöntemlerini kullanarak 2 farklı şekilde sonuca ulaşılabilir. İlk yöntem için yapılması gereken kolay fakat verimsiz bir yöntem olan derin öğrenme yöntemidir. Bu yöntemde izlenilmesi gereken 2 aşama bulunmaktadır:

- Göreve uygun derin bir mimari tespit edilmesi
- Mimarinin uygun verilerle beslenerek eğitilmesi

İkinci yöntem ise şu aşamaları içerir:

- Görüntü üzerinde çemberlerin tespit edilmesi
- Çemberlerin içerisindeki görüntülerin uygun yapay sinir ağı ile sınıflandırılması

Yukarıdaki yöntemlerden biri seçilip uygun veriler hazırlanarak uygulanması halinde istenilen sonuca ulaşılabilir.

Derin Öğrenme ile Nesne Tespiti

Derin öğrenme kısaca makinenin bir veri kümesinden belirli prosedürleri izleyerek anlam çıkartmasıdır. Bu cümlede “belirli prosedürler” olarak kastedilen derin öğrenme ağının mimarisi, veri kümesi ise yeni veri girildiğinde bilgisayarın önceki bildiklerinden çıkarım yapmasını istediğimiz “önceki bilgiler”dir.

Derin öğrenme için öncelikle veri kümesinin iyi hazırlanması gerekir. Veri makine için her şeydir. Eğer bir ağı eğitmek için yeterli veri yoksa o ağdan doğru bir sonuç beklenmemelidir. Örneğin bu proje için veri para resimleri olmaktadır. 1TL, 0.5TL, 0.25TL, 0.1TL, 0.01TL olmak üzere 5 sınıfımız olsun. Makine yeni veri geldiğinde hafızasındaki hangi veriye daha çok benzediğini bulmak için yeni gelen veriyi modele sokar, bu model veriyi belirli bir sınıfla etiketler, ayrıca bu model yeni eğitim verileri geldikçe kendisini geliştirmeye ve sınıflar için belirlediği sınırları değiştirmeye devam eder.

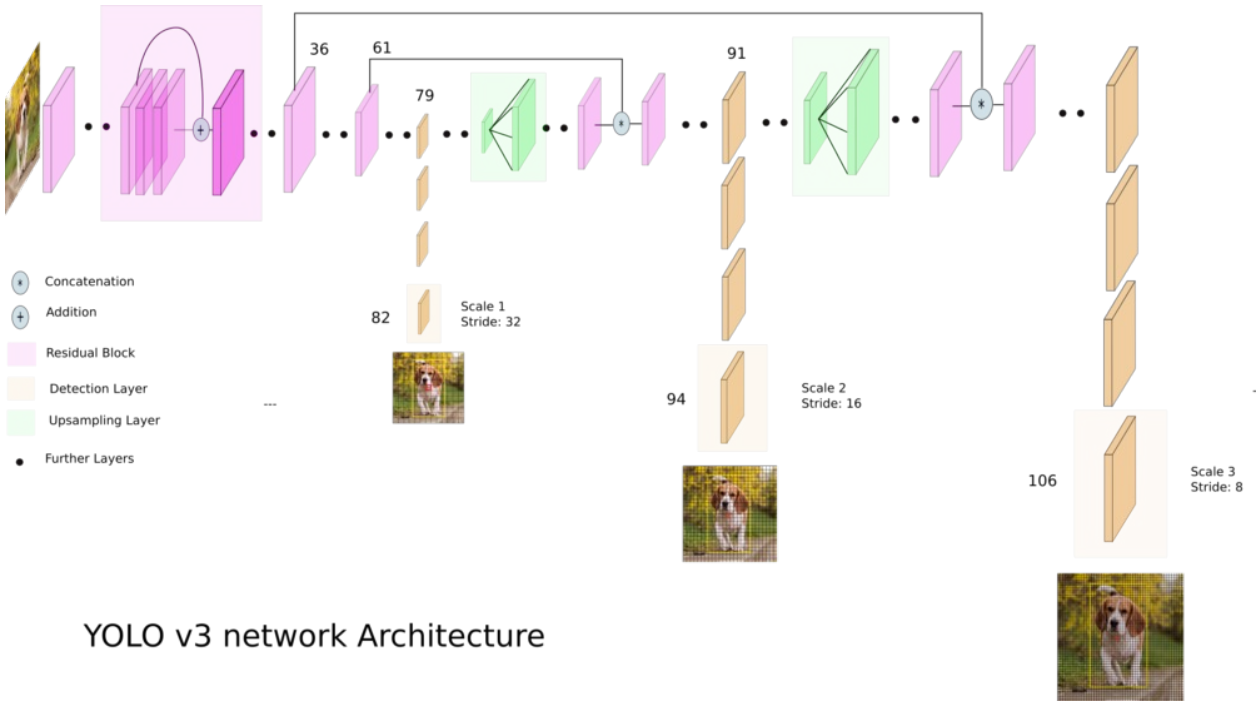
Ayrıca modele gelen bir görüntü bu sınıflardan hiçbirine ait olmayadabilir. Bu tür durumlarda ilgisi olmayan bir görüntüsü aranan sınıflardan biriyle etiketlememek için negatif görüntüler de kullanılır. Negatif görüntülerin da kullanıldığı durumlarda N istenilen sınıf, 1 ilgisiz sınıf olmak üzere toplam (N+1) sınıf oluşmaktadır.

Eğitilecek modelin gelen veriyi olması gereken sınıfla etiketleyebilmesi için istenilen sınıfla ilgili olabildiğince çok bilgiye sahip olmalıdır. Örneğin gerçek görüntüde arka plan, nesnenin yönü, yönelimi ve açısı, rengi ve parlaklığı gibi özellikler çok çeşitli olabilmektedir. Bu nedenle modelin bu tür çeşitliliklere duyarlı olabilmesi için her özellik türünden çeşitli verilerle çokça eğitilmelidir.

Ardından seçilmesi gereken bir mimari olmalıdır. Bu mimari eğitilecek veriler üzerindeki doku değişimlerinin nasıl hesaplanacağını belirtir. Bir derin öğrenme mimarisi giriş katmanı, gizli katman ve çıkış katmanı olmak üzere 3 ana katmandan oluşur. Normal sinir ağlarından farklı olmak üzere gizli katman kendi içerisinde farklı katmanlar taşıyabilmektedir. Derin öğrenme mimarilerinin farklılığını sağlayan kısım gizli katmanların farklılığıdır.

YOLO ile Nesne Tespiti

You only look once (YOLO) Joseph Chet Redmon tarafından gerçek zamanlı nesne tespitinde state of the art bir sistemdir. Aşağıda YOLO mimarisi gösterilmektedir.



YOLO v3 network Architecture

YOLO genel olarak gelen görüntüyü 32, 16 ve 8 kat küçülterek ayrı ayrı nesne tespitini yapar, uyuşan durumlarda etiketler. Mimari programcı tarafından önceden eğitilmiş modelin ağırlıklarını paylaşmaktadır fakat bu model proje içerisinde amaçlanan sınıfları içermediğinden hazırlanan verilerle sıfırdan eğitilebilmektedir.

Görüntü İşleme ve YSA ile Nesne Tespiti

Derin öğrenme ile uygulanan yöntemde bütün işlemler makine tarafından derin mimari üzerinde gösterilen adımlar işlenerek tespit edilir. Bu işlem gerekli bilgiden çok gereksiz bilgileri tespit ettiğinden çok verimli değildir, yavaştır fakat kolaydır, doğruluğu görüntü işleyerek çıkarımlarda bulunmaya göre çok daha düşüktür ve çok daha yüksek işlem yükü gerektirir. Fakat asıl amacın görüntüyü yüksek doğrulukta tespit edebilmek değil ortalama doğrulukta tespit edilen görüntüyü kullanabilmek isteyen kullanıcılar için idealdir.

Bu yöntemde görüntüde istenilen nesnenin temiz bir şekilde ön plana getirilmesi ve ön plana getirilen görüntünün YSA tarafından etiketlenmesi olmak üzere 2 aşama bulunmaktadır.

Nesnenin Ön Plana Getirilmesi

Bir görüntü üzerinde tespit edilmek istenen nesnenin arka veya ön planında bir yada birden fazla nesne olabilir. Aranan nesnenin farklı nesneler arasında ayırt edilebilmesi için featurelarına bakarak bir eleme yapılması gerekmektedir. Bu proje içerisinde madeni paraların tespit edilmesi istenildiğinden dairesel halkaların tespit edilmesi görüntü üzerinde yalnızca dairesel cisimleri ön plana getirecektir (üst üste gelen veya bir kısmı gözüken cisimlerin tespit edilmesi için renk tabanlı bir yaklaşım kullanılabilir).

İmaj üzerinde dairesel hatların ortaya çıkarılması için Hough Circle Transform (Hough Çember Dönüşümü) kullanılabilir. Hough Circle Transform her piksele bakar ve çevresinde 360 derece boyunca $x = a + R*\cos(ɵ)$, $y = b + R*\sin(ɵ)$ şartlarını sağlayan pikselleri arar. OpenCV bünyesinde yer alan HoughCircles fonksiyonu maximum ve minimum yarıçap değerleri ile işlem yapar, bu sayede gereksiz çemberlere threshold uygulanmış olur. Bu sayede aranan nesne adaylarının konumları görüntü üzerinde belirlenmiş olur.

Aday Noktalardan Özellik Çıkarımı

Hough Circle Transform ile tespit edilen bölgelerin üzerinde tespit edilmek istenen nesnenin izlerini aramak gerekir. Bir nesnenin karakteristik özelliklerini çıkartmaya özellik çıkarımı (feature extraction) denir. Bu işlemin uygulanabilmesi için şekil, doku veya renk gibi nitelikler incelenebilir.

Örneğin kenar bazlı bir şekil özelliği çıkartmak için Gaussian türevi kullanarak çok boyutlu bir Canny edge detector kullanabiliriz. Tespit edilen kenarlardan oluşan görüntü üzerinde kenar belirteçleri kullanarak karakteristiğini ortaya koyabiliriz. SURF, SIFT, ORB, BRIEF gibi belirtecler kenar goruntusu alınan bir görüntü üzerinde kullanılabilir. Bu sayede sinir ağı bu belirteçlerden tespit edilen karakteristiklerle eğitilir ve test için gelen veri aynı işleme tabi tutulup en yakın sınıfla etiketlendirilir. Bunun gibi çok farklı özellik belirteçleri kullanarak görüntüler üzerinden bilgi çıkarımı sağlanabilir.

Bu projede istatistik bazlı özellik çıkarıcı olarak bilinen Local Binary Pattern (LBP) kullanılmaktadır.

Local Binary Patterns

Local Binary Pattern operatörü bir görüntü üzerinde yerel renk karşıtlığındaki karakteristiği ortaya çıkartabilmek için geliştirilmiştir. 2005 yılında Lahdenoja tarafından istatistiksel ve yapısal doku analizinde kullanılmak üzere ortaya koyulmuştur. Görüntüdeki her piksel threshold uygulanarak binary değere atanır, merkezdeki pikselin çevresindeki pikseller yan yana yazılarak binary olarak karşılığı hesaplanır ve bu değer merkez pikselin yeni değeri olur (bu atama işlemi aynı görüntü üzerinde sonraki iterasyonları kapsamaz). Son olarak görüntünün histogramı oluşturulur. Bu histogram görüntünün karakterize özelliğini taşıyan bir histogramdır. Yapay sinir ağı bu eğitim verilerinin LBP histogramı ile eğitilir, test için gelen görüntüler dairesel bölgeleri tespit edildikten sonra histogramı çıkartılır ve en yakın sınıf ile etiketlenir.

Kaynakça

https://www.pyimagesearch.com/2015/12/07/local-binary-patterns-with-python-opencv/

https://www.robots.ox.ac.uk/~vgg/publications/2003/Mikolajczyk03a/mikolajczyk03a.pdf

https://docs.opencv.org/3.4/d4/d70/tutorial_hough_circle.html

https://arxiv.org/pdf/1611.07791.pdf

https://arxiv.org/pdf/1710.06232.pdf