Görüntü 101

Bölüm 1: Görüntüyü anlamak

İlk başta sayısal görüntü ve sayısal görüntü işleme nedirin felsefesini yapalım.

Sayısal bir görüntü adı üstünde dijital ortamda hazırlanmış ve bitlerden oluşan görsellerdir. Bu görsellerin verileri bilgisayarda matrisler halinde saklanır. Aslında ekranda gördüğümüz her bir pixel o görüntünün matris elemanlarıdır. Örneğin 100x100 boyutunda bir görüntü 100x100 boyutlu bir matrisi ifade eder.
Görüntü işleme ise temelinde bu matrisler üzerinde işlemler yapmaktır. Biz matris elemanlarını kontrol ederek görüntüye çeşitli teknikler uygulayabiliriz. Örneğin tüm görüntünün renklerini tersine çevirebiliriz, görüntüyü bulanıklaştırabiliriz, görüntüdeki kenarları algılayabilen bir filtre tasarlayabiliriz. Daha sonra kenarlarını ayırdığımız görsel ile derin öğrenme sayesinde nesne algılama gibi işlemler gerçekleştirebiliriz.

Şimdi kısaca çok kullanacağımız kavramlardan bahsedelim. Yeri geldiğinde hepsini detaylı inceleriz.

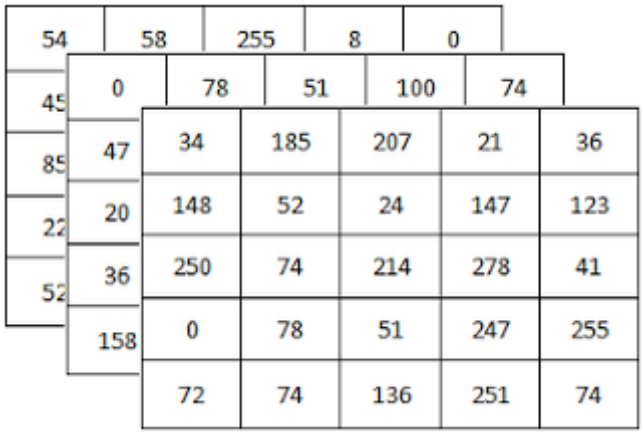
1. **Temel Kavramlar**
   - **Piksel**: Görüntünün en küçük birimi.
   - **Çözünürlük**: Görüntünün genişlik ve yükseklik boyutları (örneğin, 1920x1080).
   - **Renk Kanalları**: RGB (Red, Green, Blue) (Kırmızı, Yeşil, Mavi) veya RGBA (RGB + Alpha (şeffaflık)).
   - **Görüntü Sıkıştırma**: Çeşitli algoritmalar ile görüntüyü daha az yer kaplayacak şekilde küçültme işlemidir.
2. **Görüntü Formatları**
   - BMP, JPEG, PNG gibi farklı görüntü formatları vardır.
   - Örneğin BMP formatı, sıkıştırma olmadan ham piksel verilerini saklar ve işlemesi nispeten kolaydır.

Renk kanalları:

Görüntünün renkleri 3 temel rengin farklı oranlarda karıştırılması ile elde edilir(RGB). Bu temel renklere kanal denir ve her kanal 8 bit değer alır ve bu değerler 0-255 arasındadır. Örneğin saf kırmızıyı şu şekilde gösterebiliriz: (255, 0, 0) => görüldüğü üzere kırmızının değeri sonda, yeşil ve mavi ise 0 değerini almış. Diğer bir örnek (128, 0, 128) kanalları bize mor rengini verir.
Kanallar güzel ancak bu bize şu sorunu doğurur: Bildiğimiz gibi matrislerin her elemanı belli bir sayısal değerdir. Örneğin:

$$\begin{matrix} 21 & 3 \\ 67 & 215 \end{matrix}$$

şeklinde olur. Ama bizim 3 kanalımız vardı. O halde her bir kanalı temsil etmek için ayrı bir matrise ihtiyacımız var. Bu şu şekilde gözükür:



Böylece 1. matris kırmızı, 2.si yeşil ve 3.sü mavi renk değerlerini içerir.

Not: Elbette her uygulamada renkli bir görüntüye ihtiyacımız olmaz. Mesela çoğu derin öğrenme algoritmalarında resmin kenarlarını algılamak gibi teknikler kullanılır ki bunun için renk bilgilerine ihtiyaç yoktur. O zaman görüntüyü gray-scale(gri tonlama) şekline çevirerek tek bir matris ile daha kolay başa çıkabiliriz.
Gray-scale görüntüde tek matris yani tek kanal vardır, bu matrisin elemanları da grinin tonunu belirtir. 0=>saf siyahtır, 255=>saf beyazdır.

Bölüm 2: Alet edevat

Oki, görüntünün ne olduğunu anladık matrisin içindeki sayılardan başka bir şey değil. Ama bu matrisleri nasıl manipüle edeceğiz?
Pythonda veya c++'da çeşitli görüntü işleme kütüphaneleri var ancak hiçbir kütüphane kullanmadan saf c++ ile görüntü işleyebiliriz.
İşleri biraz daha ilerlettiğimizde video veya çok fazla görüntüyü işlemek isteriz. Ancak bu işlemler çok fazla memory harcar. Bunun için günümüzde büyük dil modellerinin(chatGPT, deepseek vs.)'de kullandığı ve bize paralel işlem yapabilme kabiliyeti sağlayan ekran kartları kullanılmakta. Bu işin öncüsü Nvidia'nın cuda çekirdekleri bize inanılmaz bir hız ve esneklik kazandırıyor.
Kısaca cuda çekirdekleri işlemci çekirdeklerine benzer ancak GPU içinde bu çekirdeklerden binlerce vardır.
Basit bir hesap yapalım: 2 adet 1000x1000 matrisimiz olsun(görüntü). Biz bu matrisleri birbiriyle çarpalım. Bu çarpımın her bir sonuç matrisi elemanı (Ci,j), A matrisinin i. satırı ile B matrisinin j. sütununun iç çarpımıdır. Bu iç çarpım, n adet çarpma işlemi gerektirir. Toplamda n^3'lük adım sayısı vardır. Bu O(n^3) ile gösterilir. (Litaratürde bir matematiksel ve algoritmik bir ifadenin işlem hacmi ve karmaşıklığının üst sınırı büyük O notasyonu -big O notation- ile ifade edilir.)
n= 1000 olduğuna göre bu çarpımın 1 milyar(10^9) adımdan oluştuğunu görürüz. Elimizde 2000 çekirdekli bir gpu varsa 10^9 / 2000 = 500.000 adımda bu işlem bitmiş olur. GPU optimizasyonunu da işin içine eklediğimizde cpu'ya kıyasla inanılmaz bir süre kazancı var.
Bu sadece 1 matris çarpımı için. Derin öğrenme girişine 100.000 görselden oluşan bir dataset işlediğimizi düşünürsek... Zamanın ne kadar önemli olduğu bu şekilde karşımıza çıkıyor.
Son bir gerçek örnek ile GPU önemini kapatalım. 4000x4000 bir matrisi kendim python ve cuda ile çarptım. Pythonda numpy adlı en verimli ve hızlı kütüphanelerden birini kullandım ve bu çarpmayı 56sn gibi bir sürede bitirdi. Çünkü python yavaş ve her ne kadar numpy kullansakta işlemler cpu üzerinden gerçekleşiyor. Ancak cuda ile aynı çarpımı yaptığımda o kadar hızlıydı ki, c++'ın kendi içindeki timer'ın minimum ölçüm hızı olan 1ms süresini gösterdi. Daha düşük bile olabilir ancak yerleşik olarak en düşük birim ms olduğundan bunu görebildim. 56sn ve <1ms... Fazla söze gerek yok.

Sonuç olarak nvidia ekran kartımızdaki cuda çekirdeklerinin farkına varıp bu güçlü donanımı kullanmamız şart.