Convolutional Neural Networks (CNN)

Resimleri sınıflandırmak ve objeleri tespit etmek için kullandığımız özelleşmiş sinir ağlarına denir.

2 ana kısımdan oluşuyor.

1- **Feature extraction:** Nitelik çıkarımı yapıyoruz
2- **Classification:** Bu çıkardığımız niteliklileri kullanarak sınıflandırma yaptığımız kısım

Bir tane resmi sınıflandırmak için gerekli olan CNN yapısı nasıl kuruluyor?

Amacımız: Bir tane resmi sınıflandırmak. Örneğin;

28x28'lik bir matrise sahip resmim olsun. Toplamda 784 tane değerden oluşan bir resme sahibim.

İnput_shape=784

Biz bu resmimizi CNN'ne sokabiliyoruz. Karşımıza 2. olarak bir layer çıkıyor. Convolutional Layer. Bu Convolutional Layer'in içerisinde bazı özellikler var. Bu özelliklerden ilki Convolutional Layer içerisinde **feature dedection** yapabilmemiz için yani nitelik tespiti yapabilmemiz için filtreler var. Filtrelerden kaç tane olduğu belli değil, kendimiz seçiyoruz. Filtre dediğimiz şey nxm'lik dediğimiz matris. Yani burda belirli boyutlarda matrisler tanımlayıp bunlara filtre diyebiliyoruz. Örneğin; 3x3'lük bir matris tanımlayabilirim.

Bu filtreler ne tespit ediyor?

Feature tespit ediyor. Low level'den high level'e doğru akan bir path var.

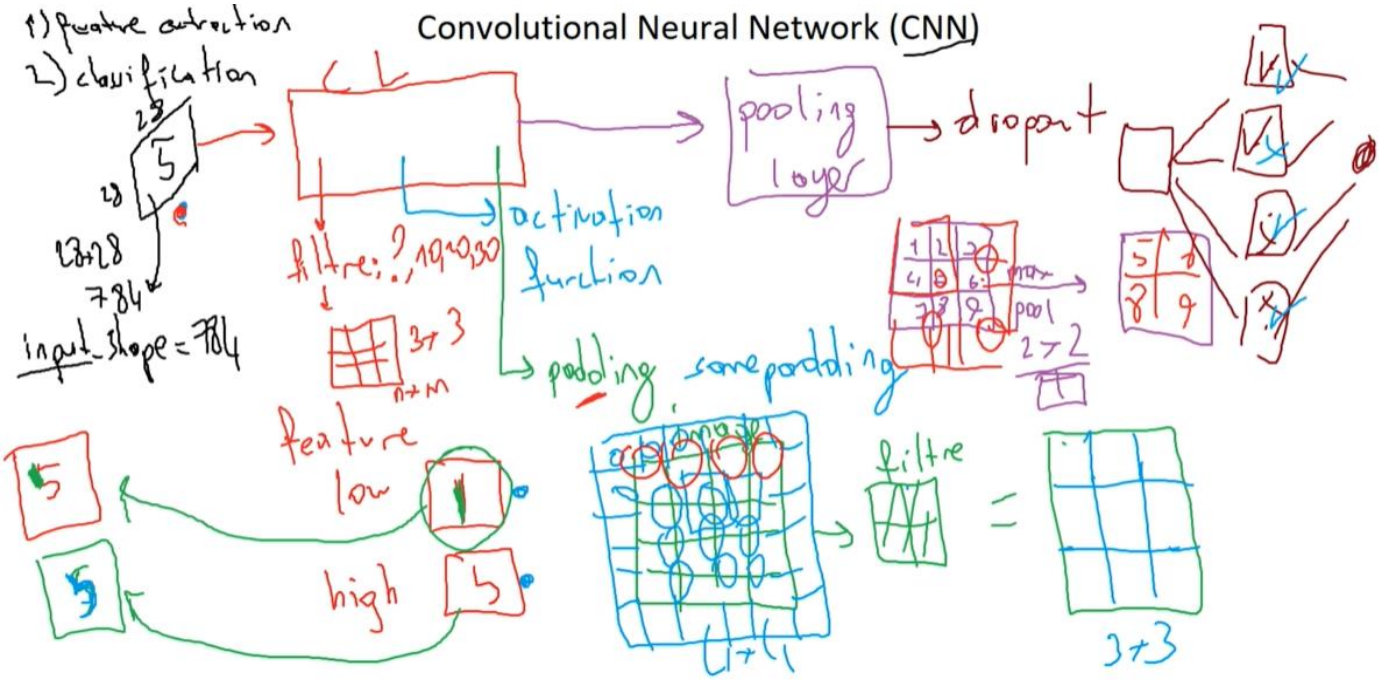
Low level filtre: 5 diye bir tane resmim var. Bunun üzerinde bu filtreyi koşturduğum zaman yani bu filtreyi dolaştırdığım zaman neyi elde ederim? Low levelde düz bir çizgim var, 5 olan resmimde de düz bir çizgim mevcut. Bu düz çizgi filtresi sayesinde resmimde bulunan düz çizgileri tespit edebilirim. Bu filtreler nitelik tespiti yapan yapılardır. **High level filtre**yle 5 resminin kıvrımlı kısmını tespit edebiliriz.

CNN içinde **activation function** da vardır. Aktivasyon fonksiyonlarının amacı: Doğrusallığı azaltmak yani nan lineer eklemektir. Aktivasyon fonksiyonları sayesinde daha karmaşık resimleri de öğrenebiliyoruz.

Bir tane de **padding** diye yapı da vardır. Diyelim ki elimizde 4x4'lük bir resim var. Buna 2x2'lik bir filtre uygulamak istiyoruz. Bu filtreyi resmin üzerinde dolaştırdığım zaman 3x3'lük bir matris elde ettim. Boyutu azaldı. Burda loss information gerçekleşti. Buraya gelinebilmenin yolu; **same padding** metodunu uyguluyoruz. Resmin etrafına bir tane daha satır ve sütun eklemiş oluyorum. Oraları 0 ile dolduruyoruz. Yani bir tane çerçeve oluşturuyoruz demektir. 2x2'lik bir filtreyi bu resim üzerinde dolaştırdığımız zaman gene 4x4'lük bir output ortaya çıkıcak. Bilgi kaybetmemiş olucaz. Padding işlemi bilgi kaybetmeyi engelleyen bir işlemdir.

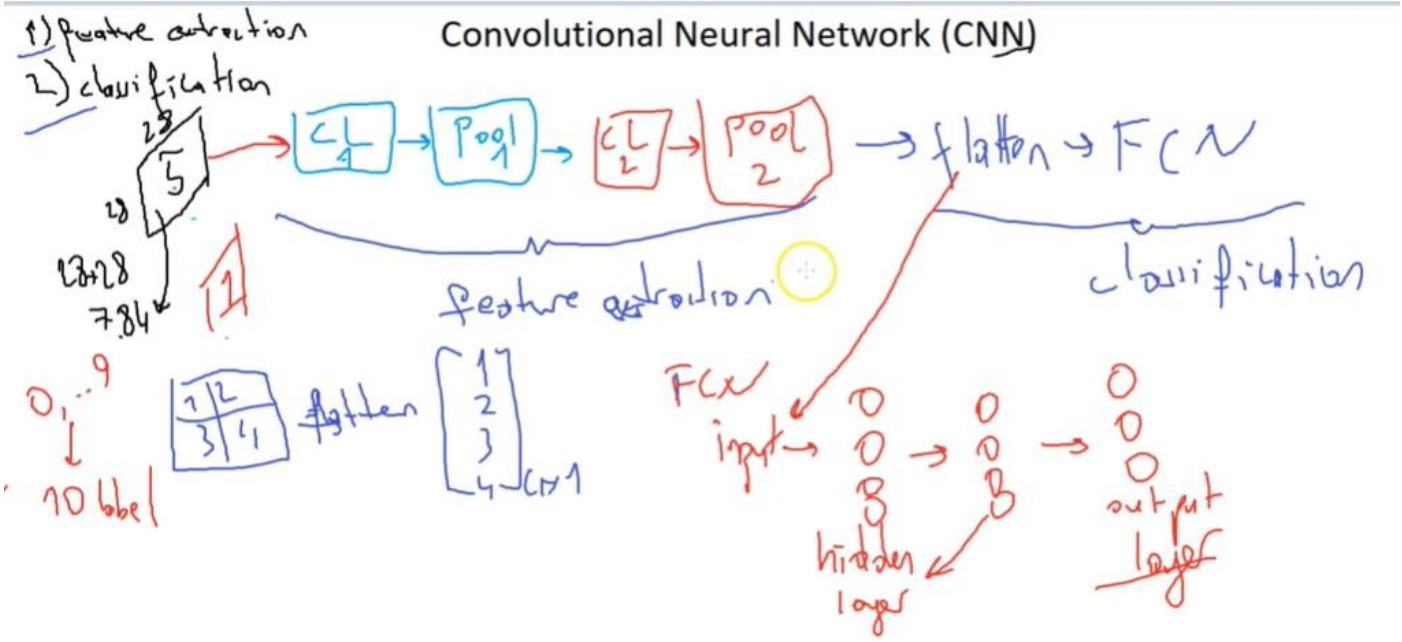
Daha sonra **pooling layer** var. Amacı: ezberlemeyi yani overfitting'i engellemek. Downsampling yapmaktır. Nasıl? Elimizde 3x3'lük bir matris var. Buna max pooling kullanıcaz. 2x2'lik bir max pool filtremiz olsun. Bu 2x2'lik yapıyı resmin üzerinde dolaştırıcaz. Sonucunda da dolaştırdığım yerlerin maksimumunu alıcaz. Boyutlarının daha küçük bir hale getirmiş olduk. CNN verilen inputu ezberlemek yerine öğreniyor.

Daha sonra **dropout** var. Aynı pooling gibi overfitting'i önleyen bir yapıdır. Nasıl kullanılıyor? Dropoutta bazı nöronları ya da bazı filtreleri kapatıyor. Örneğin; elimizde 4 tane filtre var ve bir tane resmim var. Bu resmin üzerine 4 tane filtre uyguluyacağım ve sonuçta da bir tane değerler falan çıkacak. Dropout diyor ki her seferinde her training sürecinde bu 4 tane filtrenin hepsini uygulama sadece mesala içerisinden 2 filtreyi uygula. Sonraki adımda farklı uygulanacak filtre seçtim yani burda her training sürecinde farklı farklı filtreler uygulayarak inputu yani resmimi ezberlememi engelliyor ve öğrenmeme neden oluyor.



Classification yapmadan önce flattening yapmak gerekiyor. Flattening: pooling layerden sonra 2x2'lik bir matrisim oluşmuştu. Bunu sınıflandırabilmek için düzleştirme yapmak zorundayım. Uygularsak 4x1'lik bir vektör elde etmiş olacağız.

Flattening işleminden sonra FCL kurmamız gerekiyor. FCL: bu gizli katmanlardan ve nöronlardan oluşan bir yapıdır.



Output layer: sınıflandırma yapmamı sağlayan yapay sinir ağı yani FCL kısmıdır.

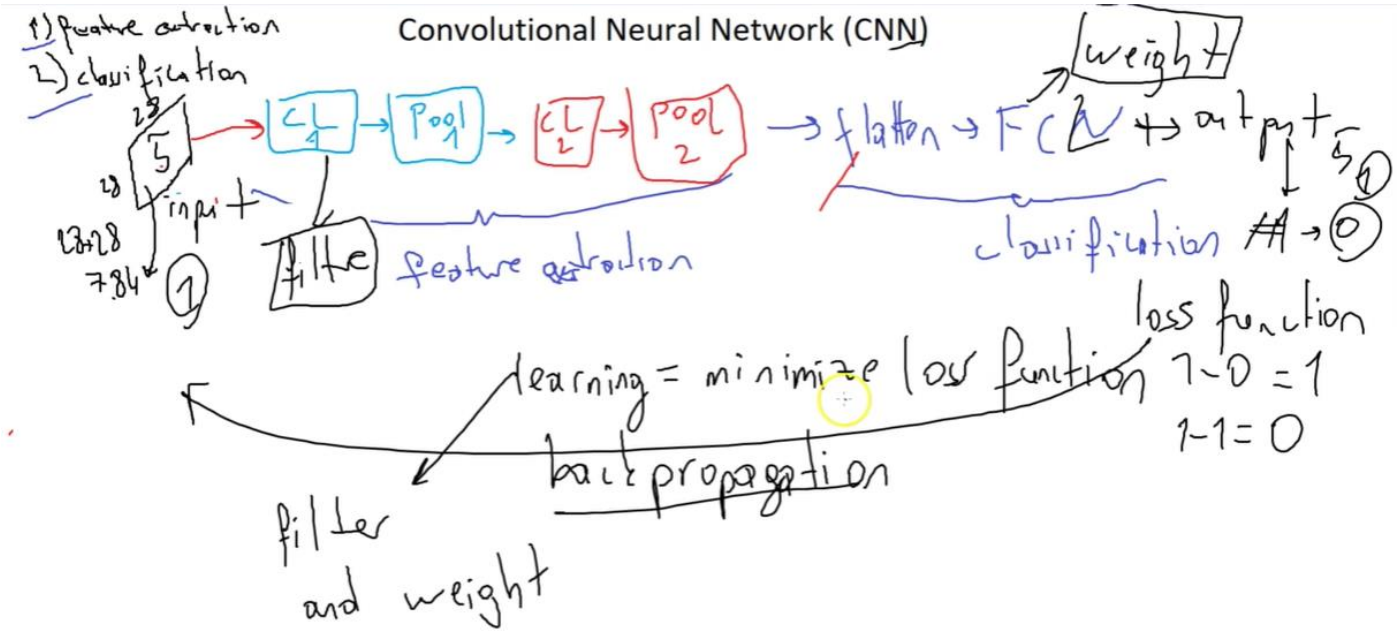
Output layer'deki nöron sayısı önemlidir. Çünkü bu nöron sayısını belirlememizdeki ana etken inputlarımızın label sayısıdır. Örneğin; 2 çeşit resmim var, bunlardan birisi 5 diğeri de 1 olsun. Sınıflandırmam gereken resim sayısı 2. Yani 2 tane labelim var. Bu yüzden output layer'in shape ya da size 2 olmak zorunda. Ya da örneğin, 0'dan 9'a kadar olan tüm rakamları sınıflandırmak istiyorum. Burda 10 tane labelim var demektir. Bu sefer output layer'in size 10 tane nörondan oluşmak zorunda.

Bu kurduğumuz yapı nasıl sınıflandırma yapacak?

Diyelim ki 5 sayısını içeriye yolladık. Output olarakta 1 sayısı çıktı. Burda bir tane loss function tanımlamam gerekiyor. Amacım, burdaki loss fonksiyonunu minilize etmek yani kaybı azaltmak. Eğer ben 5 sayısını yolluyorsam bunu label'i 1 ise yani etiketi 1 ise , bunun outputta 5 çıkmasını beklerim, label'inin etiketinin 1 olmasını beklerim. Böylece 1-1 den loss fonksiyonum 0 olucak. 5 sayısını öğrenmiş olacağım. Eğer öğrenemiyorsam loss fonksiyonunu minilize etmeye çalışacağım.

Minilize etmek? Learning demektir. Bunu backpropagation yöntemiyle yapıyoruz. Bizim yapımızda bulunan CNN'deki filtreleri ve FCL'deki weight leri güncelliyor.

Learning işleminde filter ve weight öğrenmiş oluyoruz.



Loss fonksiyonunu minilize etmekten kastımız optimize etmektir.