



Görüntü İşleme

Bilgisayar Mühendisliği, Yıldız Teknik Üniversitesi

Ödev 3

Şilan Fidan Vural

18011096

YÖNTEM

Konvolüsyonel sinir ağı(CNN) ile CIFAR-10 veri kümesi kullanılarak eğitilmiş bir öğrenme modelinin tasarımı yapılmıştır. Temel olarak sinir ağı tasarımı gerçekleştirilirken birçok durum göz önünde bulundurulmalıdır. Bu kısımda CIFAR-10 veri seti için yapılan ön işlemlerden ve tasarlanan sinir ağında kullanılan katmanlardan bahsedilmektedir.

CIFAR-10 veri seti 32x32 boyutundaki renkli resimlerden oluşmaktadır. 10 farklı sınıf mevcut olan veri setinde toplam 60000 adet resim bulunmaktadır.

İlk olarak veri setinde bulunan resimler train ve test olmak üzere iki gruba ayrılmış şekildedir. Burada train veri seti train ve validation olmak üzere bir kez daha ayrılmıştır. Bunun yapılmasının sebebi sinir ağlarında değişik parametreler ve katman sayılarının denenirken başarının validation üzerinden elde edilip en başarılı model seçildikten sonra onun test seti üzerinde denenmesinin amaçlanmasıdır. Test veri seti öncesinde model tarafından görülmediği için başarı ölçümü açısından yansız olunmaktadır. Veri setindeki resimler bu şekilde ayrıldıktan sonra normalize edilerek 0-1 aralığına getirilmektedir.

Bu aşamalardan sonra CNN yapısının oluşturulması kısmına gelinmektedir. Bu kısımda bir sinir ağı oluşturmak için denenmesi gereken birçok durum mevcuttur. Bu durumlardan bazıları konvolüsyon katman sayısının belirlenmesi, konvolüsyon katmanındaki filtre sayısı ve kernel büyüklüğünün belirlenmesi, dropout eklenmesi ve katmanların hangi sırayla dizilecekleri şeklinde özetlenebilir. Bu durumlar denerek en yüksek validation başarısının elde edildiği sinir ağının katman yapısı Figure 1’de gösterilmiştir.

Şekildeki resimde de görüldüğü gibi tasarlanan sinir ağında konvolüsyon katmanı sayısı 6 olarak belirlenmiştir.



Figure 1: CNN Yapısı

1) Convolution Katmanı

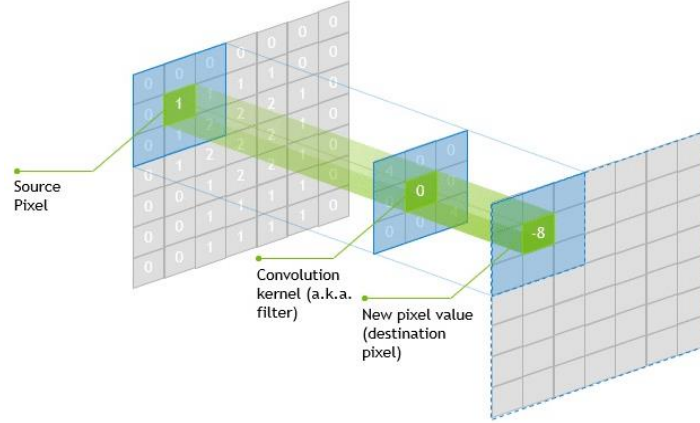
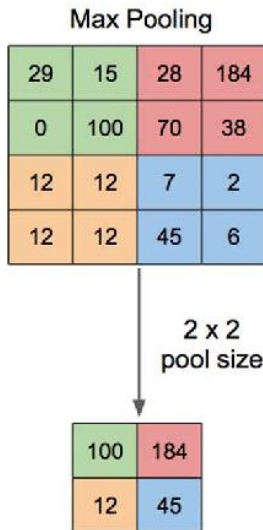


Figure 2: Convolution işlemi

Bu katmanda yapılan resimlere filtreler uygulanarak özelliklerin çıkartılması işlemidir. Bu işlem Figure 2’de gösterildiği gibidir. Kernel resim üzerinde gezdirilerek konvolüsyon işlemi uygulanmaktadır. Hangi filtrelerin(kernel) kullanılacağı kısmı ise katman tarafından belirlenmektedir. Bu katman içerisinde kullanıcı tarafından belirlenmesi gereken bazı parametreler mevcuttur. Bu parametreler filtre sayısı, kernel büyüklüğü, aktivasyon fonksiyonu ve padding gibi parametrelerdir. Kendi modelim için filtre sayıları sırasıyla 32, 32, 64, 64, 128 ve 128 olarak belirlenmiştir. Kernel büyüklüğü 3x3 seçilirken aktivasyon fonksiyonu olarak ReLu kullanılmıştır. Padding ise “same” alınmıştır. “same” padding ile girdi ile çıktı aynı boyuta sahip olmaktadır.

2) Max-Pooling Katmanı



Max-pooling katmanında, belirlenen pool büyüklüğü içerisindeki en büyük eleman alınarak hem boyut küçültülür hem de özellik haritasının en belirgin özellikleri elde edilir. Kendi modelim içerisindeki max-pooling katmanlarında pool büyüklüğü 2x2 ve stride da 2 seçilerek hem boyut yarıya düşürülür hem de en belirgin özellikler alınır. Birçok pooling yöntemi vardır ancak pratikte daha iyi başarılar elde edildiği için max-pooling kullanılmaktadır.

Figure 3: Max-pooling






3) Dense(Fully-connected) Katmanı

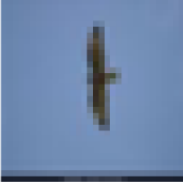

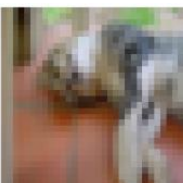


Bu katman kendinden önceki ve sonraki katmandaki tüm nöronlarla tam bağlantıya sahiptir. Yani girişteki her bir nöron çıkıştaki her bir nöronu etkilemektedir.

Bunlara ek olarak her Max-pooling katmanından sonra Dropout uygulanmıştır. Dropout eğitim sırasında içine verdiğimiz orana göre rastgele bir şekilde bazı nöronların göz ardı edilmesini sağlar. Overfitting problemine çözüm olarak kullanılan bir yöntemdir. Ayrıca tasarlanan sinir ağı içerisinde katmanlardan sonra batch normalization işlemi gerçekleştirilmiştir. Girdi olarak verilen resimleri normalize etmenin yanı sıra batch normalization ile eğitim sırasında her katmanın girdilerini mevcut batch'deki değerlerin ortalamasını ve varyansını kullanarak normalleştiririz. Batch normalization uygulanması kendi modelimde başarının artmasında etkili olmuştur.

UYGULAMA

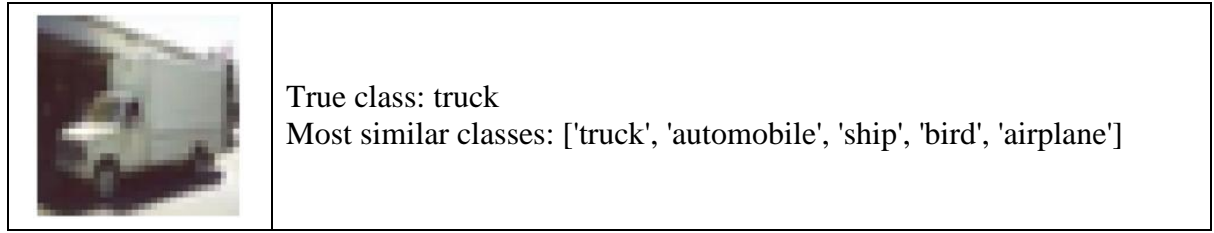
Bu kısımda 30 tane resim için doğru olan sınıflar ve en benzer tahmin edilen 5 sınıf bir tabloda verilmiştir.

	True class: airplane Most similar classes: ['airplane', 'ship', 'bird', 'automobile', 'dog']
	True class: airplane Most similar classes: ['airplane', 'ship', 'automobile', 'bird', 'truck']
	True class: airplane Most similar classes: ['airplane', 'deer', 'ship', 'truck', 'bird']
	True class: automobile Most similar classes: ['automobile', 'truck', 'ship', 'airplane', 'frog']
	True class: automobile Most similar classes: ['automobile', 'truck', 'airplane', 'horse', 'ship']

	<p>True class: automobile</p> <p>Most similar classes: ['automobile', 'truck', 'airplane', 'ship', 'dog']</p>
	<p>True class: bird</p> <p>Most similar classes: ['bird', 'airplane', 'ship', 'frog', 'deer']</p>
	<p>True class: bird</p> <p>Most similar classes: ['bird', 'deer', 'frog', 'horse', 'dog']</p>
	<p>True class: bird</p> <p>Most similar classes: ['bird', 'frog', 'airplane', 'ship', 'deer']</p>
	<p>True class: cat</p> <p>Most similar classes: ['dog', 'bird', 'cat', 'deer', 'horse']</p>
	<p>True class: cat</p> <p>Most similar classes: ['cat', 'dog', 'frog', 'deer', 'horse']</p>
	<p>True class: cat</p> <p>Most similar classes: ['cat', 'dog', 'truck', 'frog', 'automobile']</p>
	<p>True class: deer</p> <p>Most similar classes: ['deer', 'frog', 'bird', 'airplane', 'dog']</p>

	<p>True class: deer</p> <p>Most similar classes: ['deer', 'bird', 'frog', 'cat', 'dog']</p>
	<p>True class: deer</p> <p>Most similar classes: ['cat', 'deer', 'dog', 'bird', 'horse']</p>
	<p>True class: dog</p> <p>Most similar classes: ['dog', 'cat', 'bird', 'horse', 'deer']</p>
	<p>True class: dog</p> <p>Most similar classes: ['dog', 'cat', 'bird', 'frog', 'horse']</p>
	<p>True class: dog</p> <p>Most similar classes: ['dog', 'cat', 'bird', 'deer', 'airplane']</p>
	<p>True class: frog</p> <p>Most similar classes: ['frog', 'bird', 'cat', 'deer', 'horse']</p>
	<p>True class: frog</p> <p>Most similar classes: ['bird', 'frog', 'deer', 'cat', 'dog']</p>
	<p>True class: frog</p> <p>Most similar classes: ['frog', 'cat', 'automobile', 'dog', 'bird']</p>

	<p>True class: horse</p> <p>Most similar classes: ['horse', 'dog', 'deer', 'bird', 'truck']</p>
	<p>True class: horse</p> <p>Most similar classes: ['horse', 'dog', 'deer', 'cat', 'bird']</p>
	<p>True class: horse</p> <p>Most similar classes: ['horse', 'truck', 'airplane', 'automobile', 'bird']</p>
	<p>True class: ship</p> <p>Most similar classes: ['ship', 'airplane', 'cat', 'frog', 'bird']</p>
	<p>True class: ship</p> <p>Most similar classes: ['ship', 'frog', 'airplane', 'bird', 'cat']</p>
	<p>True class: ship</p> <p>Most similar classes: ['ship', 'automobile', 'deer', 'frog', 'horse']</p>
	<p>True class: truck</p> <p>Most similar classes: ['truck', 'airplane', 'automobile', 'ship', 'bird']</p>
	<p>True class: truck</p> <p>Most similar classes: ['truck', 'automobile', 'airplane', 'ship', 'bird']</p>



Tabloda verilen resimler için en başarılı bulduğu sınıflara göre confusion matrix çizdirilmiştir ve Figure 4’te gösterilmektedir.

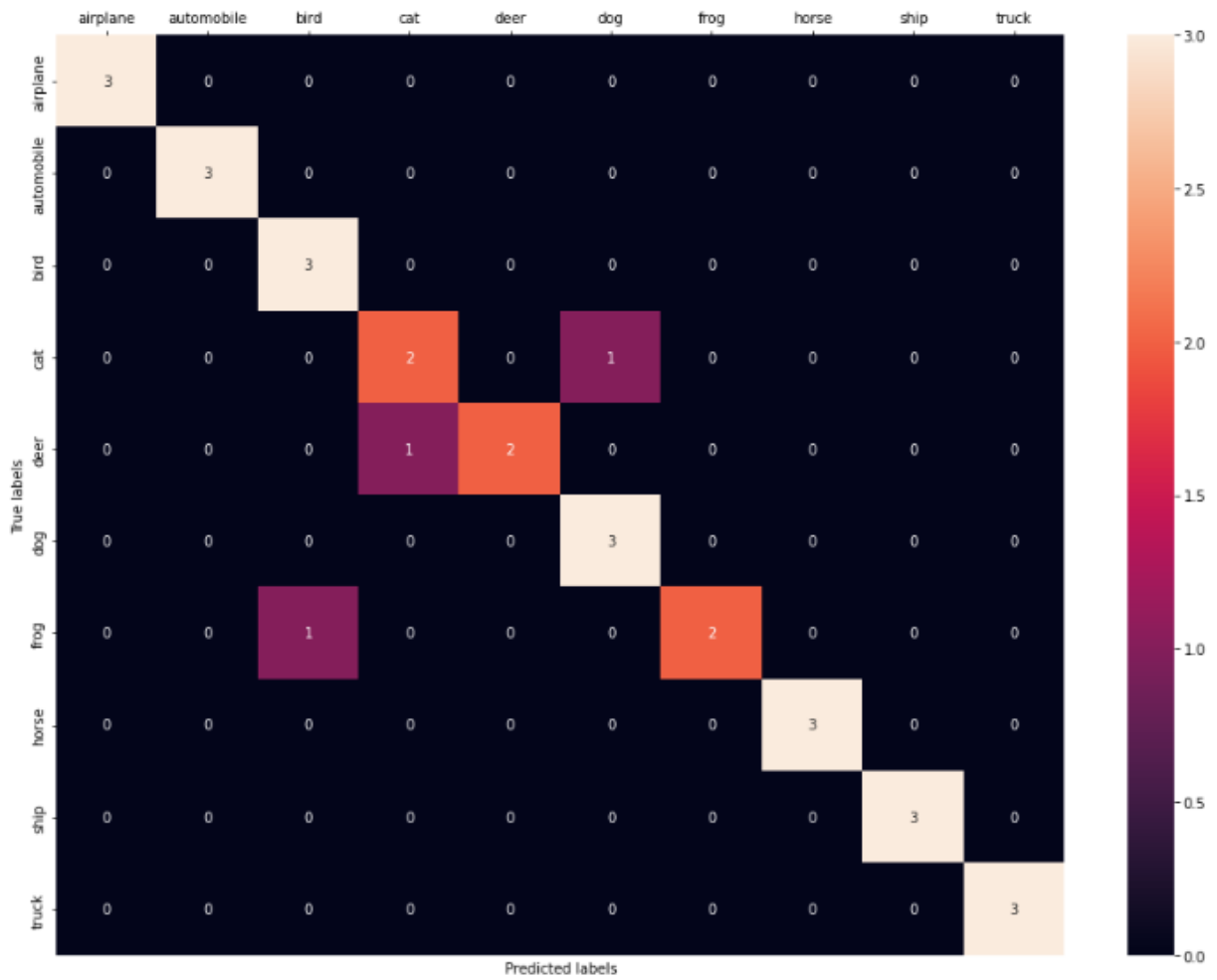


Figure 4: 30 resim için confusion matrix

SONUÇ

Yöntem bölümünde bahsedilen hiperparametrelere göre en başarılı validasyon sonucunu veren konvolüsyonel sinir ağı modeli için başarı 0.8501'dir. Bu aşamadan sonra bu model test seti üzerinde denenmiştir ve Figure 5'teki başarılar elde edilmiştir.

	precision	recall	f1-score	support
0	0.82	0.87	0.84	1000
1	0.89	0.94	0.91	1000
2	0.83	0.72	0.77	1000
3	0.65	0.73	0.69	1000
4	0.82	0.81	0.82	1000
5	0.79	0.72	0.75	1000
6	0.88	0.88	0.88	1000
7	0.91	0.84	0.87	1000
8	0.88	0.93	0.90	1000
9	0.89	0.89	0.89	1000
accuracy			0.83	10000
macro avg	0.84	0.83	0.83	10000
weighted avg	0.84	0.83	0.83	10000

Figure 5: Test seti için doğruluk skorları

Resimlerde CNN kullanılması oldukça etkilidir. Bunun nedenlerinden bir tanesi eskiden özellikler ayrıca çıkartılarak makine öğrenmesi modellerine verilirken CNN sayesinde artık özellikler otomatik olarak çıkartılmaktadır. Ayrıca konvolüsyon işlemleri ile verinin boyutsallığında azaltılma sağlanması ile de öğrenilecek parametre sayısı azalır ve dolayısıyla karmaşıklık da azalır. CNN ile oldukça yüksek başarılar da elde edilebilmektedir.

Konu ile ilgili video linki: <https://youtu.be/kY9mikVhBA4>