Görme Engelliler İçin Para Tanıma Uygulaması (Türkçe)
Money Recognition Application For Visually Impaired (In English)

Alperen TAŞKIN - 152803028 – alperentaskin35@gmail.com
İbrahim YÜLÜCE – 152803025 – ibrahimyuluce@hotmail.com

I. ÖZET

Görme engelli insanlar günlük yaşantılarında birçok problemle karşılaşmaktadırlar. Gelişen görüntü tanıma teknolojileri sayesinde onların sorunlarının bir kısmına çözüm bulunabilir. Bizde görüntü tanıma teknolojisi sayesinde banknot tanıma işlemini yapabileceğimizi düşündük.

Türk Lirası banknotların bulunduğu, hazır bir veri seti bulamadığımız için veri setini sentetik görüntüler üreterek biz oluşturduk. İndirdiğimiz banknot görüntülerini bilgisayar ortamına geçirdik [1]. Bu işlemi yaparken de Open-CV ve Keras-preprocessing kütüphanesinin ön işleme fonksiyonlarını kullandık. [2]

Oluşturduğumuz veri setini kullanarak Evrişimsel sinir ağları ile Keras-CNN üzerinde bir derin öğrenme modeli oluşturduk. Eğittiğimiz model ile 0.9986 başarım oranı elde ettik. Bu modeli oluştururken normalizasyon işleminin, dropout katmanlarının, öğrenme oranı azaltılması işleminin (Learning Rate Reduction) yarattığı değişiklikleri ve farklı optimizasyon, aktivasyon fonksiyonlarının verdiği farklı sonuçları inceledik.

II. GİRİŞ

Teknolojinin ilerlemesi ve GPU'ların gelişmesiyle birlikte derin öğrenme alanında karşılaşılan donanımsal kısıtlamaların da önüne geçilmiştir. Yıllar geçtikçe uzmanlar görüntü tanıma alanında epey yol katetmişlerdir. Görüntü tanıma teknolojisi otomatik olarak yüz, araç, nesne takibi, fotoğraf makinelerinde

ISO değeri ayarlama, görüntü sıkıştırma ve daha birçok alanda kullanılmaktadır. Görme engelli insanlar günlük yaşantılarında birçok problemle karşılaşmaktadırlar. Gelişen görüntü tanıma teknolojileri sayesinde onların sorunlarının bir kısmına çözüm bulunabilir. Biz de görüntü tanıma teknolojisi sayesinde banknot tanıma işlemini yapabileceğimizi düşündük.

Açık kaynaklı olarak paylaşılan veri seti bulamadığımızdan veri setini kendimiz oluşturmak zorunda kaldık. 5, 10, 20, 50, 100, 200 olmak üzere 6 adet Türk Lirası banknotunun önlü arkalı 12 adet resmi modelimizi eğitmek için yeterli olmayacağı için sentetik görüntü üretmemiz gerekti.

Kutuplaşma olmaması için her resimden 199'ar tane sentetik görüntü ürettik. Elimizde orijinalleri de dahil olmak üzere 2400 adet görüntü bulunuyor. Bu 2400 görüntüye karşılık gelecek başlıkları da oluşturduk. (Örneğin: 5 Liranın arka yüzü için 0, ön yüzü için 1, 10 Liranın arka yüzü için 2, ön yüzü için 3 ...)

III. ANAHTAR SÖZCÜKLER

Banknot Tanıma, Görüntü Tanıma, Bilgisayar Görüsü, Evrişimsel Sinir Ağları(CNN), Sentetik Veri Üretme

IV. ÖNCEKİ ÇALIŞMALAR

Türk Telekom'un geliştirdiği Erişilebilir Yaşam isimli mobil uygulama içerisinde para tanıma ve metni seslendirme gibi özellikler içermektedir.[3] Bir de DNC Stüdyo isimli şirketin geliştirdiği ücretli olarak sunulmuş bir Görme Engelli Para Tarayıcı mobil uygulaması bulunmaktadır. [4]

Bu iki uygulamada kullanılan veri setleri açık kaynak olarak paylaşılmadığından onlardan yararlanamadık ve projeleri inceleyemedik. Hindistan Rupisi'ni tanıma işlemi yapan bir proje bulduk fakat bu projede VGG-16 modeli üzerinde transfer öğrenme(Transfer Learning) tekniği kullanılmıştır. [5] Bizim projemizde Transfer Öğrenme tekniği kullanılmamaktadır.

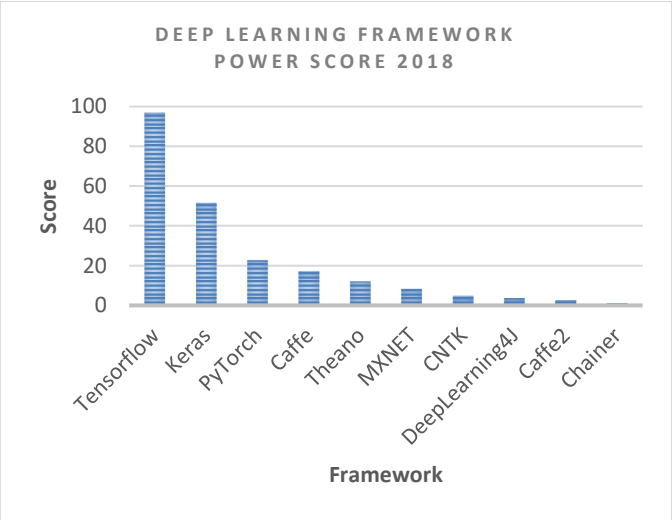
V. YÖNTEM

Oluşturduğumuz veri seti 6 adet Türk Lirası banknotunun önlü arkalı 200'er tane görüntüsünden oluşmaktadır. Görüntülerin boyu 92x200'dür. Çıktı verileri de 200'er tane 0'dan 11'e kadar sıralanmış toplamda 2400 adet etiketten oluşmaktadır.

Derin öğrenme modelimizi oluştururken görüntü tanıma problemlerinde sıkça kullanılan evrişimsel sinir ağlarını (CNN) kullandık.

LeNet-5 [6], LeCun ve ekibi tarafından 1998 yılında yayınlanmış ve ilk başarılı sonucu veren evrişimsel sinir ağı modelidir. 2009 yılında Li ve ekibi, 167 ülkeden yaklaşık 50 bin çevrimiçi çalışanla önişlemleri yapılarak etiketlenmiş 22 bin kategori ve 15 milyon görüntüden oluşan görüntü veri kümesi ImageNet'i [7] oluşturmuştur. Evrişimsel sinir ağları kullanılarak oluşturulan modeller de yıllardır bu veri kümesinde başarılar elde etmektedir.

Modelimizi Python-Keras kütüphanesi yardımıyla oluşturduk. Keras Tensorflow (Google), CNTK (Microsoft), Theano gibi derin öğrenme kütüphanelerinden tercih ettiğiniz bir tanesini arka planda çalıştırmaktadır [8]. Keras, 2018'de yapılan bir araştırmaya göre,derin öğrenme çatıları(framework) arasında üst sıralarda yer almaktadır. (dl_framework) Ayrıca kullanması ve ürün çıkartılması da diğer çatılara göre daha kolaydır. Geliştirmeyi yaparken Anaconda platformu[9] altında bulunan spyder ve jupyter notebook geliştirme ortamlarını kullandık. Yeterli GPU gücümüz olmadığı için modeli eğitirken Google-Collaboratory [10] servisinden yararlandık.



ŞEKİL 1 DEEP LEARNING FRAMEWORK POWER SCORE

VI. DENEYSEL SONUÇLAR

1) En İyi Sonuç Veren Model

Oluşturduğumuz modeller arasında en iyi model 0.9986 başarım oranına sahiptir ve 12 devir sayısında (epoch) elde edilmiştir. Modelin daha iyi sonuç vermesi ve daha performanslı çalışması için normalizasyon işlemi yapılmıştır.

Modelin daha iyi sonuç vermesi ve daha performanslı çalışması için normalizasyon işlemi yapılmıştır. Evrişim katmanlarında Relu (Rectified Linear Units) aktivasyon fonksiyonu kullanılmıştır. Dropout ve öğrenme oranı düşürme yöntemleri uygulanmıştır. Optimizasyon algoritması olarak da RMSprop(Root Mean Square Error Probability) kullanılmıştır.

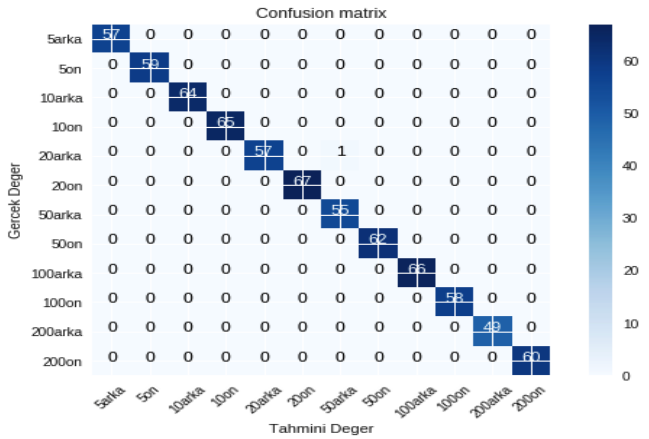
TABLO I. PARAMETRELER KATMANLAR

Layer (type)	Output Shape	Param#
Conv2d_1(Conv2D)	(None,92,200,32)	320
Conv2d_2(Conv2D)	(None,92,200,32)	9248
Max_pooling2d_1	(None,23,50,32)	0
Dropout_1(Dropout)	(None,23,50,32)	0
Conv2d_3(Conv2D)	(None,23,50,64)	51264
Max_pooling2d_2	(None,7,16,64)	0
Dropout_2(Dropout)	(None,7,16,64)	0
Flatten_1(Flatten)	(None,7168)	0
Dense_1(Dense)	(None,256)	1835264
Dropout_3(Dropout)	(None,256)	0
Dense_2(Dense)	(None,64)	16448
Dense_3(Dense)	(None,12)	780
Total params: 1,913,324 Trainable params : 1,913,324 Non-trainable params : 0		

Bu model 3 adet evrişim katmanından(Convolutional Layer) ve 2 adet tam bağlı sinir ağı katmanından (Fully Connected Dense Layer)oluşmaktadır. Model eğitim süresi 12 devir ve devirlerin bir iterasyonunda kullanılan veri boyutu 32 (batch size) olarak ayarlanmıştır.

TABLO II. EPOCH SONUÇLARI

Train on 1680 samples, validate on 720 samples				
	Loss	Acc	Val_loss	Val_acc
Epoch 1/12	2.5393	0.0863	2.4854	0.0806
Epoch 2/12	2.4704	0.0976	2.2521	0.1889
Epoch 3/12	1.9426	0.2589	1.3631	0.4153
Epoch 4/12	1.4058	0.4595	1.2509	0.4319
Epoch 5/12	1.0591	0.5893	0.6572	0.7639
Epoch 6/12	0.7869	0.7006	0.7731	0.7111
Reducing learning rate to 0.0005000000237487257				
Epoch 7/12	0.3764	0.8720	0.2768	0.8944
Epoch 8/12	0.2657	0.9107	0.1090	0.9667
Epoch 9/12	0.1864	0.9423	0.0453	1.0000
Epoch 10/12	0.1497	0.9512	0.0675	0.9847
Reducing learning rate to 0.0002500000118743628				
Epoch 11/12	0.0754	0.9786	0.0123	1.0000
Reducing learning rate to 0.0001250000059371814				
Epoch 12/12	0.0493	0.9851	0.0142	0.9986



ŞEKİL 2 CONFUSION MATRIX

2) NORMALİZASYON İŞLEMİ DENEYİ

Modelin normalizasyon işlemi yapılmadan verdiği başarım oranı çalıştırdığımız 12 devir boyunca 0.0681 çıkmıştır ve hiç ilerleme göstermemiştir.

TABLO III. EPOCH SONUÇLARI

Train on 1680 samples, validate on 720 samples				
	Loss	Acc	Val_loss	Val_acc
Epoch 1/12	14.7204	0.0857	15.0212	0.0681
Epoch 2/12	14.6694	0.0899	15.0212	0.0681
Reducing learning rate to 0.0005000000237487257				
Epoch 3/12	14.6791	0.0893	15.0212	0.0681
Reducing learning rate to 0.0002500000118743628				
Epoch 4/12	14.6598	0.0905	15.0212	0.0681
Reducing learning rate to 0.0001250000059371814				
Epoch 5/12	14.6790	0.0893	15.0212	0.0681
Reducing learning rate to 6.25000029685907e-05				
Epoch 6/12	14.6694	0.0899	15.0212	0.0681
Reducing learning rate to 3.125000148429535e-05				
Epoch 7/12	14.6598	0.0905	15.0212	0.0681
Reducing learning rate to 1.5625000742147677e-05				
Epoch 8/12	14.6694	0.0899	15.0212	0.0681
Reducing learning rate to 1e-05				
Epoch 9/12	14.6694	0.0899	15.0212	0.0681
Epoch 10/12	14.6502	0.0911	15.0212	0.0681
Epoch 11/12	14.6694	0.0899	15.0212	0.0681
Epoch 12/12	14.6625	0.0899	15.0212	0.0681

3) ÖĞRENME ORANI DÜŞÜRME İŞLEMİ DENEYİ

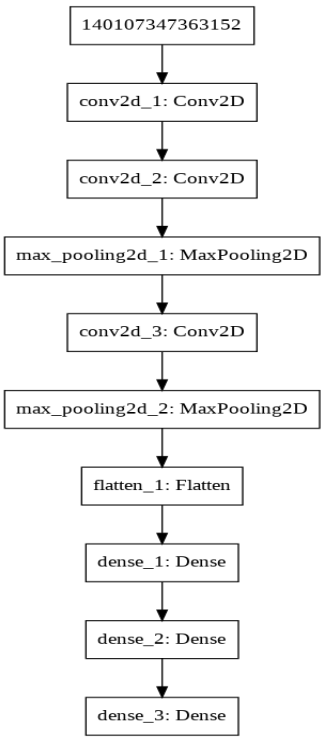
Öğrenme oranı düşürme işlemi uygulamadığımız modelin 12 devir sayısında elde ettiği başarım oranı 0.9931'dir. En yüksek başarım oranı elde ettiğimiz modelde bu başarımı 12 gibi düşük bir devir sayısında elde ettik. 12 devir sayısı az olduğu için öğrenme oranı düşürme işlemi çok fazla gerçekleşmemiştir. Bu yüzden öğrenme oranı düşürme işlemini eklemediğimiz modelin sonuçları ile en başarılı modelimizin sonuçları arasında aşırı derecede bir fark bulunmamaktadır.

TABLO IV. EPOCH SONUÇLARI

Train on 1680 samples, validate on 720 samples				
	Loss	Acc	Val_loss	Val_acc
Epoch 1/12	2.5029	0.1036	2.3365	0.1931
Epoch 2/12	1.9676	0.2625	2.0522	0.2556
Epoch 3/12	1.4512	0.4417	0.9283	0.6389
Epoch 4/12	0.9957	0.6143	0.5447	0.8653
Epoch 5/12	0.6696	0.7583	0.4001	0.8875
Epoch 6/12	0.4364	0.8488	0.1547	0.9681
Epoch 7/12	0.2977	0.9060	0.2612	0.9375
Epoch 8/12	0.2012	0.9381	0.0382	0.9972
Epoch 9/12	0.1438	0.9571	0.0264	0.9944
Epoch 10/12	0.1243	0.9577	0.0148	1.0000
Epoch 11/12	0.0987	0.9661	0.0112	0.9986
Epoch 12/12	0.0745	0.9810	0.0128	0.9931

4) DROPOUT İŞLEMİ DENEYİ

Dropout işlemi uygulamadığımız modelin 12 devir sayısında elde ettiği başarım oranı 0.9958'dir. Dropout işlemi, ağın içerisindeki bağlantıları, istediğiniz yüzdelik oranda rastgele atmanızı sağlar. Bu işlem bazı durumlarda, çıkartılan özellik haritalarının(feature map) iyileştirilmesini sağlamaktadır.



ŞEKİL 3 DROPOUT KULLANILMADAN

TABLO V. EPOCH SONUÇLARI

Train on 1680 samples, validate on 720 samples				
	Loss	Acc	Val_loss	Val_acc
Epoch 1/12	2.4732	0.1393	2.3926	0.1931
Epoch 2/12	1.6676	0.3988	1.0696	0.5653
Epoch 3/12	0.9663	0.6446	0.6641	0.7361
Epoch 4/12	0.6140	0.8190	0.2985	0.8750
Epoch 5/12	0.2597	0.9357	0.0853	0.9778
Epoch 6/12	0.2185	0.9429	0.4970	0.8444
Reducing learning rate to 0.0005000000237487257				
Epoch 7/12	0.0304	0.9958	0.0195	0.9917
Epoch 8/12	0.0263	0.9946	0.0092	0.9972
Epoch 9/12	0.0126	0.9970	0.0567	0.9889
Reducing learning rate to 0.0002500000118743628				
Epoch 10/12	0.0026	1.0000	0.0103	0.9972
Reducing learning rate to 0.0001250000059371814				
Epoch 11/12	2.8629	1.0000	0.0121	0.9958
Reducing learning rate to 6.2500029685907e-05				
Epoch 12/12	5.8040	1.0000	0.0112	0.9958

Dropout işlemi yapılırken atılacak bağlantıların oranını yüksek tuttuğumuz zaman (0.6, 0.6, 0.75) başarım oranı 0.9681 oldu.

TABLO VI. EPOCH SONUÇLARI

Train on 1680 samples, validate on 720 samples				
	Loss	Acc	Val_loss	Val_acc
Epoch 1/12	2.5210	0.0833	2.4846	0.0736
Epoch 2/12	2.4774	0.0976	2.3891	0.1778
Epoch 3/12	2.0843	0.2030	1.7824	0.4667
Epoch 4/12	1.6664	0.3405	1.3554	0.5625
Epoch 5/12	1.3813	0.4512	1.0728	0.5681
Epoch 6/12	1.1615	0.5161	0.8233	0.8139
Epoch 7/12	0.9791	0.6042	0.7183	0.7458
Reducing learning rate to 0.0005000000237487257				
Epoch 8/12	0.7537	0.6935	0.4914	0.7978
Reducing learning rate to 0.0002500000118743628				
Epoch 9/12	0.6734	0.7286	0.3663	0.9361
Epoch 10/12	0.6164	0.7613	0.3251	0.9556
Epoch 11/12	0.5469	0.7810	0.3035	0.9347
Reducing learning rate to 0.0001250000059371814				
Epoch 12/12	0.5016	0.8018	0.2680	0.9681

Buradan çıkarttığımız sonuç, dropout işlemi başarım yüzdesine olumlu yönde etki etmektedir lakin atılacak bağlantıların oranı fazla yüksek tutulmamalıdır.

5) FARKLI OPTİMİZASYON ALGORİTMALARININ DENENMESİ

Optimizasyon algoritması olarak SGD(Stochastic Gradient Descent) kullandığımızda 12 devir sayısında elde ettiği başarım oranı 0.2333'dür. Fakat ilerleme devam etmektedir. Bu yüzden Keras kütüphanesinin geri dönüş fonksiyonlarından (callback functions) erken durdurma (early stopping) fonksiyonunu kullanarak 50 epoch boyunca çalıştırdık. Erken durdurma fonksiyonu, eğitimin oranının yavaşlaması veya durması durumunda eğitimi durdurur. Bu sayede daha kısa süre çalışmasını sağlamış olur. Erken durdurma fonksiyonu sonucunda elde ettiğimiz başarım yüzdesi 0.9819'dir ve 27 devir sayısı sonunda elde edilmiştir. Bu başarımı elde etmek için 14 deneme yapılmıştır. Önceki denemelerin başarım oranları erken durdurma yüzünden düşük çıkmıştır.

TABLO VII. SGD OPTIMIZER EPOCH SONUÇLARI

Train on 1680 samples, validate on 720 samples				
	Loss	Acc	Val_loss	Val_acc
Epoch 1/12	2.4901	0.0720	2.4830	0.0917
Epoch 2/12	2.4844	0.0893	.24817	0.0917
Epoch 3/12	2.4814	0.0988	2.4789	0.1042
Epoch 4/12	2.4786	0.1119	2.4769	0.1625
Epoch 5/12	2.4759	0.1113	2.4728	0.1569
Epoch 6/12	2.4711	0.1155	2.4672	0.1847
Epoch 7/12	2.4644	0.1351	2.4588	0.0958
Epoch 8/12	2.4573	0.1357	2.4449	0.1319
Epoch 9/12	2.4392	0.1554	2.4242	0.1472
Reducing learning rate to 0.004999999888241291				
Epoch 10/12	2.4238	0.1690	2.4080	0.1486
Epoch 11/12	2.4055	0.1738	2.3872	0.2292
Epoch 12/12	2.3818	0.1786	2.3601	0.2333

TABLO VIII. SGD EARLY STOPPING EPOCH SONUÇLARI

Train on 1680 samples, validate on 720 samples				
	Loss	Acc	Val_loss	Val_acc
Reducing learning rate to 0.0006249999860301614				
Epoch 24/50	0.3480	0.9655	0.2590	0.9792
Epoch 25/50	0.3300	0.9726	0.2491	0.9819
Epoch 26/50	0.3278	0.9714	0.2492	0.9806
Epoch 27/50	0.3255	0.9714	0.2546	0.9819

Optimizasyon algoritması olarak Adam [11] kullandığımızda 12 devir sayısında elde ettiği başarım oranı 0.9778'dir.

TABLO IX. ADAM OPTIMIZER EPOCH SONUÇLARI

Train on 1680 samples, validate on 720 samples				
	Loss	Acc	Val_loss	Val_acc
Epoch 1/12	2.4745	0.0982	2.3410	0.1764
Epoch 2/12	1.8101	0.3179	1.2443	0.4847
Epoch 3/12	1.0540	0.5500	0.6752	0.8181
Epoch 4/12	0.6621	0.7250	0.3825	0.8778
Epoch 5/12	0.5962	0.7685	0.4281	0.8153
Reducing learning rate to 0.0005000000237487257				
Epoch 6/12	0.4179	0.8345	0.2219	0.9444
Epoch 7/12	0.3587	0.8595	0.1518	0.9708
Epoch 8/12	0.2804	0.8982	0.1333	0.9667
Reducing learning rate to 0.0002500000118743628				
Epoch 9/12	0.2369	0.9149	0.1224	0.9653
Reducing learning rate to 0.0001250000059371814				
Epoch 10/12	0.1827	0.9363	0.0769	0.9819
Epoch 11/12	0.1710	0.9500	0.1044	0.9722
Reducing learning rate to 6.25000029685907e-05				
Epoch 12/12	0.1695	0.9446	0.0722	0.9778

6) FARKLI OPTİMİZASYON FONKSİYONLARININ DENENMESİ

Modeldeki aktivasyon algoritmalarını Tanh (Hyperbolic Tangent) yaptığımızda, 12 devir sayısında elde ettiği başarım oranı 0.0681'dir.

TABLO X. TANH EPOCH SONUÇLARI

Train on 1680 samples, validate on 720 samples				
	Loss	Acc	Val_loss	Val_acc
Epoch 1/12	2.6850	0.0667	2.4959	0.0764
Epoch 2/12	2.5504	0.0869	2.5062	0.0917
Epoch 3/12	2.5266	0.0714	2.4912	0.0792
Reducing learning rate to 0.0005000000237487257				
Epoch 4/12	2.5133	0.0804	2.4946	0.0792
Reducing learning rate to 0.0002500000118743628				
Epoch 5/12	2.5092	0.0863	2.4916	0.0819
Reducing learning rate to 0.0001250000059371814				
Epoch 6/12	2.4986	0.0899	2.4918	0.0819
Reducing learning rate to 6.25000029685907e-05				
Epoch 7/12	2.5008	0.0762	2.4919	0.0819
Reducing learning rate to 3.125000148429535e-05				
Epoch 8/12	2.4961	0.0863	2.4911	0.0819
Reducing learning rate to 1.5625000742147677e-05				
Epoch 9/12	2.5000	0.0869	2.4909	0.0681
Reducing learning rate to 1e-05				
Epoch 10/12	2.4929	0.0756	2.4907	0.0681
Epoch 11/12	2.4992	0.0750	2.4906	0.0681
Epoch 12/12	2.4960	0.0780	2.4904	0.0681

7) ORTAKLAMA İŞLEMİ (POOLİNG) DENEYİ

Ortaklama işlemi yapıldığında parametre sayısı 1,913,324 iken bu işlem yapılmadan parametre sayısı 301,543,916 olmaktadır. Bu yüzden eğitim süresi de artmaktadır. Banknot tanıma probleminde, banknotlardaki şekillerin yönleri farklı olabilir fakat bulundukları lokasyonlar birbirlerine yakındır. Ortaklama işlemi görüntülerdeki nesnelerin yönlerinden çok lokasyonlarının önemli olduğu durumlarda iyi sonuçlar vermektedir. Ortaklama işlemi kullanmadığımız modelde elde ettiğimiz başarım oranı 0.9806'dır. 12 gibi az sayıda sınıfta bile başarım oranının biraz düşük olduğu için, daha yüksek sınıflı problemlerde başarım oranının daha düşük olacağın çıkarımı yapılabilir.

TABLO XI. POOLING İŞLEMİ EPOCH SONUÇLARI

Train on 1680 samples, validate on 720 samples				
	Loss	Acc	Val_loss	Val_acc
Epoch 1/8	3.6423	0.0804	2.4750	0.2153
Epoch 2/8	2.2470	0.2530	1.5279	0.3847
Epoch 3/8	1.2745	0.5119	0.8601	0.6583
Epoch 4/8	0.8607	0.6589	0.3842	0.8653
Epoch 5/8	0.6405	0.7673	0.5707	0.8069
Reducing learning rate to 0.0005000000237487257				
Epoch 6/8	0.3096	0.8994	0.1051	0.9722
Epoch 7/8	0.2230	0.9268	0.2115	0.9139
Reducing learning rate to 0.0002500000118743628				
Epoch 8/8	0.1304	0.9619	0.0650	0.9806

VII. SONUÇ

Oluşturduğumuz modeller arasında en yüksek başarım oranını elde eden modelde aktivasyon fonksiyonu olarak Relu kullanılmıştır. 2 kere %20 dropout ve çıktı katmanından önce de 1 kere %50 dropout işlemi kullanılmıştır.

Optimizasyon algoritması olarak RMSprop tercih edilmiştir. Normalizasyon ve öğrenme oranı düşürme işlemi uygulanmıştır.

Sonuçlara baktığımızda normalizasyon işleminin sadece performans arttırımı için değil, başarım oranının ciddi şekilde yükselmesi için de kullanılması gerektiğini çıkartabiliriz. Aktivasyon fonksiyonlarından Relu fonksiyonu görüntü tanıma problemlerinde iyi başarım oranı elde etmektedir. Öğrenme oranı başlarda yüksek tutulurken devir sayısı ilerledikçe düşürmek başarım oranına iyi yönde etki etmektedir. Bu yüzden öğrenme oranı düşürme işleminin modellere uygulanması gerekmektedir.

Optimizasyon algoritması olarak Adam ve RMSprop'un denenmesi gerekir çünkü yakın sonuçlar vermektedirler. SGD algoritması daha uzun devir sayılarında Adam ve RMSprop'un elde ettiği başarım değerlerini yakalamaktadır fakat performans bakımından verimsiz olduğu için tercih edilmeyebilir.

VIII. REFERANSLAR

[1]https://www.mypivots.com/dictionary/definition/628/turkish-lira-try
[2] https://opencv.org, https://keras.io/preprocessing/image
[3]https://play.google.com/store/apps/details?id=tr.com.turktelekom.erisilebiliryasam&hl=en_US
[4]https://play.google.com/store/apps/details?id=com.dncsyudyo.gormeengelli
[5]https://software.intel.com/en-us/blogs/2017/11/21/cash-recognition-for-the-visually-impaired-using-deep-learning
[6] http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf
[7] http://www.image-net.org
[8] https://keras.io/why-use-keras
[9] https://www.anaconda.com
[10] https://colab.research.google.com
[11] https://arxiv.org/abs/1412.6980