KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ
ANABİLİM DALI

YÜKSEK LİSANS TEZİ

TRAFİK LEVHALARININ EVRİŞİMSEL SİNİR AĞLARI İLE TANINMASI

BÜŞRA ÖVÜN

KOCAELİ 2022

KOCAELİÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ
ANABİLİM DALI

YÜKSEK LİSANS TEZİ

TRAFİK LEVHALARININ EVRİŞİMSEL SİNİR AĞLARI İLE TANINMASI

BÜŞRA ÖVÜN

Prof.Dr. Yaşar BECERİKLİ .....
Danışman, Kocaeli Üniversitesi

Prof.Dr. Sevinç İLHAN OMURCA .....
Jüri Üyesi, , Kocaeli Üniversitesi

Doç. Dr. Adem TUNCER .....
Jüri Üyesi, , Yalova Üniversitesi

Tezin Savunulduğu Tarih: 22.06.2022

ETİK BEYAN VE ARAŞTIRMA FONU DESTEĞİ

Kocaeli Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırladığım bu tez/proje çalışmasında,

- Bu tezin/projenin bana ait, özgün bir çalışma olduğunu,
- Çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı,
- Bu çalışma kapsamında elde edilen tüm veri ve bilgiler için kaynak gösterdiğimi ve bu kaynaklara kaynakçada yer verdiğimi,
- Bu çalışmanın Kocaeli Üniversitesi'nin abone olduğu intihal yazılım programı kullanılarak Fen Bilimleri Enstitüsü'nün belirlemiş olduğu ölçütlere uygun olduğunu,
- Kullanılan verilerde herhangi bir tahrifat yapmadığımı,
- Tezin/Projenin herhangi bir bölümünü bu üniversite veya başka bir üniversitede başka bir tez/proje çalışması olarak sunmadığımı,

beyan ederim.

☒ Bu tez/proje çalışmasının herhangi bir aşaması hiçbir kurum/kuruluş tarafından maddi/alt yapı desteği ile desteklenmemiştir.

☐ Bu tez/proje çalışması kapsamında üretilen veri ve bilgiler ..... tarafından ..... no'lu proje kapsamında maddi/alt yapı desteği alınarak gerçekleştirilmiştir.

Herhangi bir zamanda, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçları kabul ettiğimi bildiririm.

Büşra ÖVÜN

YAYIMLAMA VE FİKRİ MÜLKİYET HAKLARI

Fen Bilimleri Enstitüsü tarafından onaylanan lisansüstü tezimin/projemin tamamını veya herhangi bir kısmını, basılı ve elektronik formatta arşivleme ve aşağıda belirtilen koşullarla kullanıma açma izninin Kocaeli Üniversitesi'ne verdiğimi beyan ederim. Bu izinle Üniversiteye verilen kullanım hakları dışındaki tüm fikri mülkiyet haklarım bende kalacak, tezimin/projemin tamamının ya da bir bölümünün gelecekteki çalışmalarda (makale, kitap, lisans ve patent vb.) kullanımı bana ait olacaktır. Tezin/projenin kendi özgün çalışmam olduğunu, başkalarının haklarını ihlal etmediğimi ve tezimin/projenin tek yetkili sahibi olduğumu beyan ve taahhüt ederim. Tezimde yer alan telif hakkı bulunan ve sahiplerinden yazılı izin alınarak kullanılması zorunlu metinlerin yazılı izin alarak kullandığımı ve istenildiğinde suretlerini Üniversiteye teslim etmeyi taahhüt ederim. Yükseköğretim kurulu tarafından ***yayınlanan "Lisansüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge"*** kapsamında tezim aşağıda belirtilen koşullar haricinde YÖK Ulusal Tez Merkezi/ Kocaeli Üniversitesi Kütüphaneleri Açık Erişim Sisteminde erişime açılır.

☐ Enstitü yönetim kurulu kararı ile tezimin/projemin erişime açılması mezuniyet tarihinden itibaren 2 yıl ertelenmiştir.

☐ Enstitü yönetim kurulu gerekçeli kararı ile tezimin/projemin erişime açılması mezuniyet tarihinden itibaren 6 ay ertelenmiştir.

☒ Tezim/projem ile ilgili gizlilik kararı verilmemiştir.

Büşra ÖVÜN

ÖNSÖZ VE TEŞEKKÜR

Bu çalışmanın gerçekleştirilmesinde, değerli bilgilerini paylaşan ve zamanını ayıran, kıymetli danışman hocam Prof. Dr. Yaşar BECERİKLİ'ye teşekkürü bir borç biliyor ve şükranlarımı sunuyorum. Yine çalışmamda konu, kaynak ve yöntem açısından bana sürekli yardımda bulunarak yol gösteren Dr. Öğr. Üyesi İrfan KÖSESOY ve Öğr. Gör. Cihan ALBAY'a teşekkür ederim.

Tüm hayatım boyunca benim yanımda olan, aldığım kararları her zaman destekleyen, moral veren annem Zübeyde KÖSESOY, babam Halil KÖSESOY ve Kardeşlerim Amine Nil KÖSESOY, Muhammet KÖSESOY, Ravza Nur KÖSESOY ve Sümeyye KÖSESOY ALBAY'a teşekkür ederim.

Tez sürecim boyunca benden desteğini esirgemeyen, süreç boyunca yanımda olan ve cesaretlendiren sevgili eşim Deniz ÖVÜN ve biricik oğlum Adil Erdem ÖVÜN'e teşekkür ederim.

Haziran – 2022 Büşra ÖVÜN

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltmalar

ConNN	: Convolutional Neural Network (Evrişimsel Sinir Ağları)
ESA	: Evrişimsel Sinir Ağları
ReLU	: Rectified Linear Unit (Rektifiye Lineer Birim)
SVM	: Support Vector Machine (Destek Vektör Makinesi)

TRAFİK LEVHALARININ EVRİŞİMSEL SİNİR AĞLARI İLE TANINMASI

ÖZET

Gelişen teknoloji sayesinde sürücü destek sistemleri ve otonom araç sektörü gün geçtikçe büyümektedir. Bununla birlikte trafik işaretlerinin takibinin zorluğu ve bilinirliğinin düşüklüğü yol güvenliğinin sağlanmasında karşılaşılan en büyük problemler arasında yer almaktadır. Bu sebeple trafik işaretlerinin otomatik tanınması sürücü destek sistemleri ve otonom araç teknolojisinin en önemli konuları arasına girmiştir.

Bu çalışmadaki hedef, derin öğrenme ile trafik levhaları veri setindeki görüntüleri tanıyacak bir ağın eğitilmesi sayesinde yol güvenliğini sağlamada önemli bir faktör olan trafik işaretlerini tanıma üzerine çalışma yaparak sürücü destek sistemleri için daha performanslı çözümler önermektir. Veri seti olarak Kaggle'dan edinilen Alman Trafik İşareti Algılama Benchmark veri seti (GTSRB) kullanılmıştır. Evrişimsel sinir ağları modeli ile geliştirilen uygulamalar, kullanılan mimari ve parametrelere göre kıyaslanarak elde edilen sonuçlar farklı metrikler üzerinde sunulmuştur. LeNet, AlexNet, GoogLeNet, VGGNet ve bu mimariler incelenerek bu çalışmaya özgü geliştirilen Evrişimsel sinir ağı mimarisi, GTSRB veri seti üzerinde denenmiştir. Önerilen mimaride başarım %98.83 ile trafik levhası tanıma problemi üzerine çalışmış öteki çalışmalarla kıyaslandığında en başarılı modeldir. Aynı zamanda gerçek zamanlı çalışabilmesi yönünden değerlendirildiğinde resim başı ortalama tanıma süresi 0.01 s olarak tespit edilmiştir. LeNet mimarisi ile geliştirilen modelde başarım %96.1 iken, AlexNet'te %95.7, GoogleNet'te 97.0 ve VGGNet'te %97'dir.

Anahtar Kelimeler: AlexNet, ConNN, Derin öğrenme, LeNet, Trafik Levhası Tanıma.

RECOGNITION OF TRAFFIC SIGNS USING CONVOLUTIONAL NEURAL NETWORKS

ABSTRACT

Thanks to the developing technology, driver assistance systems and autonomous vehicle sector are growing day by day. However, the difficulty of following traffic signs and their low awareness are among the biggest problems encountered in ensuring road safety. For this reason, automatic recognition of traffic signs has become one of the most important issues of driver support systems and autonomous vehicle technology.
The aim of this study is to propose more performance solutions for driver support systems by working on recognizing traffic signs, which is an important factor in ensuring road safety, by training a network that will recognize images in the traffic sign dataset with deep learning. The German Traffic Sign Detection Benchmark dataset (GTSRB) obtained from Kaggle was used as the dataset. The applications developed with the convolutional neural network model are compared according to the architecture and parameters used and the results obtained are presented on different metrics. LeNet, AlexNet, GoogLeNet, VGGNet and these architectures were examined and the Convolutional neural network architecture developed specifically for this study was tested on the GTSRB dataset. It is the most successful model when compared to other studies on traffic sign recognition problem with 98.83% success in the proposed architecture. At the same time, when evaluated in terms of real-time operation, the average recognition time per picture was determined as 0.01 s. While the performance is 96.1% in the model developed with LeNet architecture, it is 95.7% in AlexNet, 97.0 in GoogleNet and 97% in VGGNet.

Keywords: AlexNet, ConNN, Deep learning, LeNet, Traffic Sign Recognition.

1. GİRİŞ

Trafikte sürücüler birçok uyarana maruz kalmaktadır. Bu durum sürücülerin trafik işaretlerini takip etmelerini ve algılamalarını güçleştirmektedir(Çakıcı & Murat Yetiş Şazi, 2017; Zam, 2019). Levhaların benzerliği, sürücünün hızı, mesafe gibi faktörler ile birlikte düşen algı seviyesi trafik güvenliğini tehlikeye atmaktadır. Trafik güvenliği için trafik işaretleri büyük önem arz etmektedir (Aylak ve diğ., 2021). Sürücülerin trafik işaretlerini algılamaları incelendiğinde, trafik işaretlerinin gözden kaçması kadar ne anlama geldiğinin bilinmesinde de eksiklikler olduğunu ortaya koyan çalışmalar ortaya çıkmıştır(Tiryaki, 2019).

2006 yılında yayınlanan bir çalışmada Türkiye'de sürücüler tarafından trafik işaretlerinin bilinirlik düzeylerini ölçmek amacıyla "Trafik İşaretlerinin Algılanabilirliği Anket Formu" üzerinde çalışma yapılmış ve bu çalışmada 210 gözleme yer verilmiştir. Sürücü belge sınıfı, öğrenim düzeyi, cinsiyet gibi farklı özellikleri olan kişilerin yanıtları ve çalışmanın sonuçları aşağıda gösterilmiştir. Düzenlenmiş olan ankette 20 adet çoktan seçmeli soru bulunmaktadır. Her soru 5 puana tekabül edecek şekilde, toplam 100 puan üzerinden değerlendirilmiştir (Yakut, 2006). Anket sonuçları Şekil 1-3'de paylaşılmıştır.

Trafik işaretlerinin sürücüler tarafından bilinirlik oranlarındaki düşüklüğüne dikkat çeken bir diğer anket çalışması 2017 yılında Denizli'de yapılmıştır. Az sayıda tam olarak doğru bilinen levhanın yanında yorumsuz bırakılan, tam tersi anlam çağrıştıran, kısmen doğru bilinen ve belirgin olarak bir ayrım göstermediği için farklı bir işaretle karıştırılan çok sayıda levha örneği de olduğu sonucuna varılmıştır (Çakıcı & Murat Yetiş Şazi, 2017).

Türkiye'de sadece 2019 yılı içerisinde toplamda 1 milyon 168 bin 144 adet trafik kazası meydana gelmiştir. Kazaların 993 bin 248 adedi maddi hasarlı iken 174 bin 896 adedi ölümlü yahut yaralanmalı trafik kazasıdır. Bu kazalarda 5 bin 473 kişi hayatını kaybederken 283 bin 234 kişi yaralanmıştır (Tüik, 2020).

Yine Tüik raporuna göre; 2020 yılı içerisinde Ülkemiz karayolu ağlarında toplamda 983 bin 808 adet trafik kazası meydana gelmiştir. Bunların 833 bin 533'ü maddi hasarlı iken 150 bin 275'i de ölümlü ya da yaralanmalı trafik kazasıdır. Karayolu ağlarımızdaki 2020 yılında meydana gelen trafik kazaları sonucunda ölen kişilerin %49,4'ü sürücü iken %30,7'si yolcu ve %19,9'u yayadır.

2020 yılında gerçekleşmiş 177 bin 867 ölümlü veya yaralanmalı trafik kazasına neden olan kusurlara bakıldığında %88,3'ünün sürücü kaynaklı, %7,0'ının yaya kaynaklı, %2,7'sinin taşıt kaynaklı, %1,4'ünün yolcu kaynaklı ve %0,5'inin yol sebebi ile gerçekleştiği görülmüştür(Tüik, 2021).Tüik raporuna ilişkin sonuçlar Şekil 1-1 ve Şekil 1-2 olarak eklenmiştir.

Şekil 1.1. Trafik Kazasında Ölü, Yaralı Dağılımları

Trafik kaza istatistikleri, 2009-2020

Yıl	Toplam kaza sayısı	Ölümlü yaralanmalı kaza sayısı	Maddi hasarlı kaza sayısı	Ölü sayısı			Yaralı sayısı
				Toplam	Kaza yerinde	Kaza sonrası(1)	
2009	1 053 346	111 121	942 225	4 324	4 324	-	201 380
2010	1 106 201	116 804	989 397	4 045	4 045	-	211 496
2011	1 228 928	131 845	1 097 083	3 835	3 835	-	238 074
2012	1 296 634	153 552	1 143 082	3 750	3 750	-	268 079
2013	1 207 354	161 306	1 046 048	3 685	3 685	-	274 829
2014	1 199 010	168 512	1 030 498	3 524	3 524	-	285 059
2015	1 313 359	183 011	1 130 348	7 530	3 831	3 699	304 421
2016	1 182 491	185 128	997 363	7 300	3 493	3 807	303 812
2017	1 202 716	182 669	1 020 047	7 427	3 534	3 893	300 383
2018	1 229 364	186 532	1 042 832	6 675	3 368	3 307	307 071
2019	1 168 144	174 896	993 248	5 473	2 524	2 949	283 234
2020	983 808	150 275	833 533	4 866	2 197	2 669	226 266

(1) Trafik kazasında yaralanıp sağlık kuruluşuna sevk edilenlerden kazanın sebep ve tesiriyle 30 gün içinde ölenleri kapsamaktadır.
- Bilgi yoktur.

Şekil 1.2. Trafik Kaza İstatikleri

Her yıl binlerce kişi trafik kazalarında hayatını kaybetmektedir. Kazaların önlenmesinde trafik işaretlerine uymak büyük önem arz etmektedir. Yol güvenliğinin sağlanabilmesi için son yıllarda sürücü destek sistemleri ileri seviyelere taşınmıştır. Trafik işareti tanıma sistemlerinde sürücü destek sistemlerinin önemli bir parçasıdır. Trafik İşareti Tanıma

TRAFİK İŞARETLERİ	DOĞRU VERİLEN CEVAPLAR										
	CİNSİYET		ÖĞRENİM DURUMU				SÜRÜCÜ BELGESİ				
	Kadın	Erkek	İlkokul	Ortaokul	Lise	Üniversite / Y.Okul	A Sınıfı	B Sınıfı	C Sınıfı	D Sınıfı	E Sınıfı
	18 GÖZLEM % 78	150 GÖZLEM % 80	9 GÖZLEM % 90	29 GÖZLEM % 76	64 GÖZLEM % 78	66 GÖZLEM % 83	3 GÖZLEM % 100	150 GÖZLEM % 77	2 GÖZLEM % 50	1 GÖZLEM % 50	7 GÖZLEM % 100
	12 GÖZLEM % 52	101 GÖZLEM % 54	6 GÖZLEM % 60	15 GÖZLEM % 40	42 GÖZLEM % 51	60 GÖZLEM % 75	2 GÖZLEM % 67	106 GÖZLEM % 55	1 GÖZLEM % 25	0 GÖZLEM % 0	4 GÖZLEM % 57
	15 GÖZLEM % 65	144 GÖZLEM % 77	8 GÖZLEM % 80	26 GÖZLEM % 68	60 GÖZLEM % 73	65 GÖZLEM % 81	2 GÖZLEM % 67	145 GÖZLEM % 75	4 GÖZLEM % 100	1 GÖZLEM % 50	2 GÖZLEM % 29
	21 GÖZLEM % 91	181 GÖZLEM % 97	9 GÖZLEM % 90	37 GÖZLEM % 97	79 GÖZLEM % 96	79 GÖZLEM % 99	3 GÖZLEM % 100	187 GÖZLEM % 96	4 GÖZLEM % 100	2 GÖZLEM % 100	7 GÖZLEM % 100
	16 GÖZLEM % 70	140 GÖZLEM % 75	5 GÖZLEM % 50	28 GÖZLEM % 74	61 GÖZLEM % 74	62 GÖZLEM % 78	2 GÖZLEM % 67	148 GÖZLEM % 76	3 GÖZLEM % 75	2 GÖZLEM % 100	6 GÖZLEM % 86
	9 GÖZLEM % 39	109 GÖZLEM %58	4 GÖZLEM % 40	21 GÖZLEM % 55	47 GÖZLEM % 57	48 GÖZLEM % 60	1 GÖZLEM % 33	119 GÖZLEM % 61	3 GÖZLEM % 75	1 GÖZLEM % 50	5 GÖZLEM % 71
	22 GÖZLEM % 96	184 GÖZLEM % 98	9 GÖZLEM % 90	37 GÖZLEM % 97	80 GÖZLEM % 98	80 GÖZLEM % 100	3 GÖZLEM % 100	190 GÖZLEM % 98	4 GÖZLEM % 100	2 GÖZLEM % 100	7 GÖZLEM % 100
	12 GÖZLEM % 52	81 GÖZLEM % 43	6 GÖZLEM % 60	14 GÖZLEM %37	38 GÖZLEM % 46	36 GÖZLEM % 45	2 GÖZLEM % 67	85 GÖZLEM % 44	3 GÖZLEM % 75	1 GÖZLEM % 50	4 GÖZLEM % 57
	18 GÖZLEM % 78	173 GÖZLEM % 93	10 GÖZLEM % 100	35 GÖZLEM % 92	73 GÖZLEM % 89	72 GÖZLEM % 90	3 GÖZLEM % 100	175 GÖZLEM % 90	4 GÖZLEM % 100	2 GÖZLEM % 100	7 GÖZLEM % 100
	15 GÖZLEM % 65	149 GÖZLEM % 80	7 GÖZLEM % 70	33 GÖZLEM % 87	59 GÖZLEM % 72	65 GÖZLEM % 81	2 GÖZLEM % 67	151 GÖZLEM % 78	3 GÖZLEM % 75	2 GÖZLEM % 100	6 GÖZLEM % 86
	17 GÖZLEM % 74	171 GÖZLEM % 91	9 GÖZLEM % 90	34 GÖZLEM % 90	74 GÖZLEM % 90	72 GÖZLEM % 90	3 GÖZLEM % 100	174 GÖZLEM % 90	4 GÖZLEM % 100	2 GÖZLEM % 100	7 GÖZLEM % 100
	22 GÖZLEM % 96	182 GÖZLEM % 95	7 GÖZLEM % 70	36 GÖZLEM % 95	79 GÖZLEM % 96	77 GÖZLEM % 96	3 GÖZLEM % 100	185 GÖZLEM % 95	4 GÖZLEM % 100	2 GÖZLEM % 100	6 GÖZLEM % 86
	21 GÖZLEM % 91	181 GÖZLEM % 97	9 GÖZLEM % 90	35 GÖZLEM % 92	78 GÖZLEM % 95	80 GÖZLEM % 100	2 GÖZLEM % 67	189 GÖZLEM % 97	4 GÖZLEM % 100	2 GÖZLEM % 100	6 GÖZLEM % 86
	17 GÖZLEM % 74	170 GÖZLEM % 91	7 GÖZLEM % 70	36 GÖZLEM % 95	72 GÖZLEM % 88	71 GÖZLEM % 89	2 GÖZLEM % 67	176 GÖZLEM % 91	4 GÖZLEM % 100	1 GÖZLEM % 50	4 GÖZLEM % 57
	11 GÖZLEM % 48	135 GÖZLEM % 72	6 GÖZLEM % 60	25 GÖZLEM % 66	60 GÖZLEM % 73	55 GÖZLEM % 69	1 GÖZLEM % 33	139 GÖZLEM % 72	3 GÖZLEM % 75	1 GÖZLEM % 50	4 GÖZLEM % 57
	20 GÖZLEM % 87	177 GÖZLEM % 95	8 GÖZLEM % 80	36 GÖZLEM % 95	77 GÖZLEM % 94	76 GÖZLEM % 95	3 GÖZLEM % 100	183 GÖZLEM % 94	4 GÖZLEM % 100	2 GÖZLEM % 100	7 GÖZLEM % 100
	18 GÖZLEM % 78	176 GÖZLEM % 94	9 GÖZLEM % 90	35 GÖZLEM % 92	76 GÖZLEM % 93	73 GÖZLEM % 91	2 GÖZLEM % 67	178 GÖZLEM % 92	4 GÖZLEM % 100	2 GÖZLEM % 100	7 GÖZLEM % 100
	23 GÖZLEM % 100	182 GÖZLEM % 97	8 GÖZLEM % 80	37 GÖZLEM % 97	81 GÖZLEM % 99	79 GÖZLEM % 99	3 GÖZLEM % 100	189 GÖZLEM % 97	4 GÖZLEM % 100	2 GÖZLEM % 100	7 GÖZLEM % 100
	7 GÖZLEM % 30	113 GÖZLEM % 60	3 GÖZLEM % 30	23 GÖZLEM % 61	41 GÖZLEM % 50	48 GÖZLEM % 60	3 GÖZLEM % 100	102 GÖZLEM % 53	3 GÖZLEM % 75	2 GÖZLEM % 100	5 GÖZLEM % 71
	23 GÖZLEM % 100	168 GÖZLEM % 90	8 GÖZLEM % 80	36 GÖZLEM % 95	74 GÖZLEM % 90	72 GÖZLEM % 90	2 GÖZLEM % 67	176 GÖZLEM % 91	4 GÖZLEM % 100	2 GÖZLEM % 100	6 GÖZLEM % 86
Başarı Gözlem Oranı	% 73,2	% 81,8	% 73,5	% 80	% 80,1	% 83,5	% 78,4	% 81,1	% 86,2	% 80	% 81,4
Başarı Gözlem Oranı	% 81		% 81				% 81				
Toplam Gözlem Sayısı	23	187	10	38	82	80	3	194	4	2	7

Şekil 1.3. Anket Sonuçları

sistemleri, kamera görüş sistemlerini kullanarak güzergâhtaki trafik işaretlerini algılayan, okuyan ve yorumlayan bir araç içi sürücü destek sistemidir. Araç üreticileri, Trafik İşareti Tanıma sistemlerini kullanarak otonom araç deneyimini iyileştirmeyi hedeflemektedir. Bu sistemler yol emniyetini sağlamada da yardımcı olmaktadır (Roper ve diğ., 2018).

Artan araç trafiği ile birlikte trafik işaretlerini otomatik tanıyan sistemlerin önemi artmıştır (Hatzidimos, 2004).Günümüzde otonom araçlar ve sürücü destek sistemleri için geliştirilen sistemlerde yapay zekâ kullanımı ön plana çıkmıştır. Yüzyıllar boyu insanoğlu işleri kolaylaştırmak için makineler icat etmiştir. Doğayı ve canlıları taklit yoluyla çözümler üretilmiştir (Alpaydin, 2020). Bunun son halkası yapay zekâdır. İnsan zekâsının taklit edilmesi ile davranış olarak insana benzeyen, tecrübe kazanan, kazandığı tecrübeler sayesinde gelişen ve mantık yürütebilen donanım ve yazılım sistemleri bütününe yapay zekâ denir (Jiang ve diğ., 2017; Russell & Norvig, 2010). Günümüzde yapay zekâ birçok bilimde karmaşık problemleri çözmede kullanılmaktadır. Yapay zekâ algoritmaları, özellikle derin öğrenme, görüntü tanıma problemlerinde önemli ilerlemeler kaydetmiştir (Hosny ve diğ., 2018; Samuel, 1967). Astronomi alanında, Milyonlarca görüntü verisi içinden galaksileri sınıflandırmak ve teleskop araştırmalarında süpernovaları bulmak için ayıklama yapabilmektedir. Tıp alanında, genetik dizilerden proteinlerin işlevlerini ve yapılarından yola çıkarak tahminlerde bulanabilmektedir. Bununla birlikte birçok hastalık teşhisinde kullanılmaktadır. Geliştirilmiş algoritmalar sayesinde sistemler doğrudan beyinden kortikal aktiviteyi alabilmektedir. Bu sayede felçli bir insanın motor kontrolünü geri yüklemede kullanılabilecek motor korteksinden gelen sinyalleri gerekli kaslara ileterek tekrardan işlevsel hale gelecekleri duruma döndürebilmektedir (Obermeyer & Emanuel, 2016). Kaydedilen bu ilerlemelerin hem gerçek zamanlı olması hem de yüksek çözünürlüklü fizyolojik verilerden meydana gelmesi sebebi ile makina öğrenme teknikleri olmadan bu aşamalara getirilmeleri mümkün değildir (Bouton ve diğ., 2016; Gilbert ve diğ., 2008).

Yapay zekânın tarihi eskilere dayanmaktadır. Bilgisayarın üretilmesi ile birlikte 1941 yılında makinelerin akıllanması konusu ele alınmıştır (Hamet & Tremblay, 2017). Makinelerin de insanlar gibi karar mekanizmalarına sahip olup olamayacağı yani "Makineler düşünebilir mi?" sorusuna 1950'de Alan Turing yanıt aramıştır. Bu çalışma literatüre Turing testi olarak girmiştir (Warwick & Shah, 2016). Alan Turing'in önermiş

olduğu bu test makinelerin zekâya sahip olup olmadığının tespit edilmesi için tasarlanmıştır (Sindhuja R, 2018). Makine öğrenimi ekonomi (Mullainathan & Spiess, 2017), astronomi, tıp, tarım (Aksoy ve diğ., 2020), otomotiv (Luckow ve diğ., 2018), lojistik (Aylak ve diğ., 2021), siber güvenlik (Sinoplu ve diğ., 2021), askeriye (Johnson, 2019), bankacılık (Fethi & Pasiouras, 2010) gibi birçok bilimde kullanılmaktadır.

Yapay zekâ kullanımının başarılı olduğu alanlardan biri de bilgisayarlı görü sistemleridir. Bu sayede otomotiv sektöründe otonom araçlar ve sürücü destek sistemleri gün geçtikçe daha iyi noktalara gelmektedir (Roper ve diğ., 2018). Trafik işareti tanıma sistemleri otomotiv sektörünün temel bileşenlerinden biridir (Malik & Siddiqi, 2014). İvmeli şekilde ilerleyen araştırmalara rağmen, trafik işareti ile arka planı ayırt etmesi zor görüntüler, doğal sebeplerden dolayı(ağaç, kirlenme vb.) görüşü etkilenen levhalar, değişken aydınlatma koşulları ve farklı görüş açıları sebebi gibi problemler ile karşılaşılmaktadır (Çetinkaya & Acarman, 2020). Trafik işaretlerini tespit etme ve tanıma sorunu günümüzde bir miktar çözüme kavuşturulmuş olmakla birlikte hala daha iyi performanslar ile çalışabilecek sistemler üzerinde çalışılmaktadır.

Bu bölümde Trafik İşareti Tanıma Sistemleri için yapılan bazı çalışmalara yer verilmiştir. Trafik işaretlerinin tanınması için çeşitli makine öğrenme yöntemleri kullanılmaktadır. İncelenen çalışmalardan da görüldüğü üzere son yıllarda veri setlerinin artması ve bu veri setlerini kullanabilecek donanımların da buna paralel olarak gelişmesiyle birlikte birçok tanıma probleminde derin öğrenme metotları yaygın olarak kullanılmaya başlanmıştır.

De La Escalera ve arkadaşları 1997 yılında trafik işaretlerinin tanınması ile alakalı bir çalışma yapmışlardır. Bu çalışma iki kısımdan oluşmaktadır. İlk kısmında trafik levhasının tespiti yapılmaktadır. Tespit yapma işlemi için trafik levhalarının tasarlanırken insanlar için kolay algılanmasına yardımcı olan farklı renk ve şekillerde olmaları özelikleri kullanılmıştır. Görüntüyü kısımlara ayırmak için renk eşiği, şekil tespiti için ise şekil analizi yapılmıştır. İşaret tespiti yapılmasının ardından ikinci aşamada sınıflandırma yapılabilmesi için bir ağ yapısı kullanılmıştır. Görüntü bölümlere ayrılarak renk eşikleme ve işareti algılama için şekil analizi yapılmıştır. İkinci kısımda ise sınıflandırma için bir yapay sinir ağı modellenmiştir (De La Escalera ve diğ., 1997).

Paclík ve arkadaşlarının yaptığı çalışmada laplace olasılık yoğunluğunu kullanarak trafik işaretlerini sınıflandırma problemi üzerinde çalışılmıştır. Normalizasyon işlemi, Laplace Kernel parametrelerini çapraz doğrulama yöntemine tabi tutulmasıyla yapılmıştır. Sınıflandırma işlemi için boyutları 15 ile 150 piksellik 1100 görüntülü veri seti kullanılmıştır. 4945 gürültülü görüntü üzerinde çalışılarak sistem test edilmiştir. Bu 4945 görüntü, orijinal 1100 görüntü üzerinde rastgele döndürme işlemi, gauss gürültüsü ekleme işlemleri yapılarak elde edilmiştir. 45 sınıflı görüntü veri seti renk ve şekillerine göre olmak üzere 9 gruba ayrılmıştır. Veri kümesinin %90'ı eğitim verisi, %10 test verisi olarak kullanılmıştır (Paclík ve diğ., 2000).

Hatzidimos 2004 yılında trafik işaretlerinin geometrik özelliklerinin kullanılmasına dayandırılmış bir yöntem önerir. İlk olarak, renk eşikleme metodu kullanılarak trafik işareti çıkarılır. Renk eşikleme ile siyah beyaz olan resim, kenar bulma algoritmaları sayesinde tespit edilmektedir. Hough dönüşümü, üçgen işaretlerini algılamak için her bir tespit edilmiş işarete uygulanır. Çizgiler arasındaki açılar hesaplanır ve bu açılar belirli eşikler arasında ise tespit edilmiş işaret üçgen trafik işareti olarak algılanır. Daire işaretlerini tespit etmek için Randomize Hough dönüşümü kullanılır (Hatzidimos, 2004).

Ulay 2008 yılında yaptığı çalışmada trafik işareti tanıma uygulamaları için renk ve şekil özeliklerin kullanılmasını öneren bir yöntem kullanır. Kenar tespit algoritmaları ile kenar bilgisine ulaşılır. Renk bilgisi için ise HSV ve RGB renk uzayları kullanılır. Trafik işaretinin tespitinde kullanılan bu iki özelliğin birleştirmesi için iki farklı yol izlenmiştir. İlkinde, renk ve şekil algılama sonuçlarının her bir piksel başı değerlerinin toplamı, kenar bulma algoritmalarıyla şekil tespiti algoritmalarıyla tespit edilecek bölgeyi oluşturmak üzere kullanır. Ancak ikinci yöntemde, tespit edilecek alanı bulmak için bu değerlerin piksel başı çarpımını hesaplayarak kullanır. Veri seti olarak videolardan elde edilmiş görüntüler ve trafik işareti içeren resimler kullanılmıştır (Ulay, 2008).

Özkan'ın 2010 yılındaki çalışmasında FPGA donanımı ile gerçek zamanlı trafik işareti tespiti yapılmıştır. Trafik işaretlerinin kendilerine özgü olan renk ve şekil parametrelerinden faydalanılarak tespit işlemi yapılmaya çalışılmıştır. YUV renk uzayındaki görüntüler RGB renk uzayına dönüştürmüştür. Renk eşiklemesine (colour thresholding) tabi tutulduktan sonra siyah beyaz olan resim Sobel algoritması uygulanarak trafik işaretinin kenarları tespit edilmiştir. Ortam koşullarına göre başarım

oranlarında farklılıklar gözlemlenmiştir. Gölgeli ve netlik seviyesi iyi olmayan görüntülerde başarım %70 iken iyi aydınlatılmış görüntüler için başarım oranı %90'dır (Özkan, 2010).

Daire, üçgen ve kare trafik işaretleri üzerinde çalışılmıştır. Veri seti olarak 500 adet görüntü kullanılmıştır. Bu görüntülerin bir kısmı resim iken bir kısmı videodan yakalanmış görüntülerdir. Uygulama MATLAB'da halka-bölmeli (ring-partitioned) metot ile hazırlanmıştır. Kullanılan yöntem histogram tabanlı bir metottur. Gündüz çekilmiş resimler için iyi sonuç vermektedir. Ancak yüksek düzeyde aydınlatılmış görüntüler için belli hatalar verebilmektedir. Veri setlerindeki görüntüler mükemmel şekilde aydınlatılmış verilerdir. Kenar bulma algoritmaları ile tespit edilmek istenen işaret bulunarak sonrasında tanıma işlemine tabi tutulmaktadır. Şekil ve renklere göre başarı oranları farklılık göstermektedir. Tanıma işlemi yanında tespit işlemi yapılan testlerde tanıma oranının düştüğü gözlemlenmiştir. Üçgen işaretlerden mükemmel şekilde tespit edilmiş olanlarda başarım oranı %88, köşe tespiti aracılığı ile tespit edilen görüntülerde ise başarı oranı %72'dir. Kusursuz olarak tespit edilmiş kare işaretler için tanıma oranı %95, renk tabanlı algılama ile tespit edilen dairesel işaretler için %90'dır (Becer, 2011).

Cireşan ve arkadaşları 2012 yılında yaptıkları trafik işaretlerini tanıma üzerine yaptıkları çalışmada veri seti olarak GTSRB veri seti kullanılmıştır. Veri setinde farklı ölçülerde bulunan görüntüler 48x48 piksel olacak şekilde yeniden boyutlandırılmıştır. Kullanılan resimler arasındaki farklılıklar sebebi ile histogram eşitleme işlemi, adaptif histogram eşitleme işlemi ve kontrast normalizasyonu işlemleri gerçekleştirilmiştir. Derin sinir ağları (DNN) kullanılmıştır. Ağdaki çıkış aktivasyonlarının ortalaması bulunarak çok kolonlu derin sinir ağları (Multi-Column DNN / MCDNN) oluşturulmuştur. 25 katmandan meydana gelmiştir. Sinir ağının eğitimi GPU üzerinde yapılmaktadır. Başarımı %99,46'dır (Cireşan ve diğ., 2012).

Özdamar'ın 2014 yılında yaptığı çalışmada daha küçük bir trafik işaretleri kümesi tercih edilmiştir. Trafik işaretleri kırmızı çerçeveli üçgen ve dairesel işaretler olacak şekilde kapsam daraltılarak belirli işaret üzerine çalışılmıştır. Altuzay yöntemi kullanılarak sınıflandırmalar yapılmış. Matlab üzerinde hazırlanmıştır. Veri seti olarak orijinal görüntüler ve bu görüntülerin yapay olarak aracılığı ile çoğaltılması ile elde edilen görüntüler kullanılmıştır. Ayırtedici Ortak Vektör, Temel Bileşen Analizi, Doğrusal

Ayırtaç Analizi yöntemleri kullanılarak öznitelik vektörleri çıkarılmıştır. Tanıma kısmında ise Yerel Gradyan Histogram, Yerel İkili Örüntü, Yerel Faz Kuantalama ve Gabor Görüntü Tanıma kullanılarak tanıma işlemi gerçekleştiriliştir. En iyi başarımları Ayırtedici Ortak Vektör altuzay yöntemi ve Yerel Gradyan Histogram tanımlaycısı vermiştir. Üçgen trafik işaretleri için %98,38 başarım, dairesel trafik işaretleri için %99,25 sınıflandırabilme başarımları sağlanmıştır (Özdamar, 2014).

Zaklouta ve Stanciulescu tarafından 2014 yılında yapılan çalışmada gerçek zamanlı çalışan bir Trafik İşareti Tanıma sistemi sunulmuştur. Çalışma üç aşamalıdır. Bunlar segmentasyon, algılama ve sınıflandırma aşamalarıdır. Görüntüdeki trafik işaretlerinde yoğunluklu bulunan kırmızı bölgeleri ayırt etmek için renk özelliği üzerinde eşikleme yapılarak ilgili kısmın tespiti yapılmaya çalışılmıştır. Algılama işlemi için, Histogram of Oriented Gradients (HOG) özelliklerine sahip Doğrusal Destek Vektör Makinesi (SVM) kullanılmıştır. Sınıflandırma için ise, Karar Ağaç sınıflandırıcıları kullanılmıştır. Hem regresyon hem de sınıflandırma problemlerinde kullanılabilen Rastgele Orman (Random Forest), K-d ağacı algoritmaları ile tespit edilen trafik işaretlerinin hangi sınıfa dahi edileceğinin kararı verilmiştir. K-d ağacı algoritmasının performansını daha iyi bir seviyeye çıkarmak için uzaysal ağırlıklandırma yöntemi önerilmiştir. Fisher Kriteri ve Rastgele Orman, özellik uzayını daraltarak sınıflandırmayı daha kısa sürede yapabilmek için kullanılmıştır. Üçgen ve daire trafik işaretleri üzerinde çalışılmıştır. 12 üçgen trafik levhası sınıfı, 12 daire trafik levhası sınıfı olacak şekilde toplam 24 sınıflı veri seti kullanılmıştır. Test için Alman Trafik İşareti Algılama Benchmark (GTSRB) veri seti kullanılmıştır. Alman Trafik İşareti Algılama Benchmark (GTSRB) veri setinde %97, kendi oluşturduğu veri setinde %81 başarım sağlamıştır (Zaklouta & Stanciulescu, 2014).

Tiryaki'nin 2019 yılında hazırlamış olduğu çalışması geliştirme ortamı olarak MATLAB üzerinde gerçekleştirilmiştir. Veri seti olarak GTSRB kullanılmıştır. Sinar ağı mimarisi oluşturulurken MatConvNet araç kutusu kullanılmıştır. Oluşturulan yapıda ortalama %94,56 başarım sağlanmıştır. Başarımı artırmak üzere Mekânsal Dönüşüm Ağlarından (STN) faydalanılarak başarım ortalama %96,22'ye taşınmıştır (Tiryaki, 2019).

Sichkar ve Kolyubin tarafından 2020 yılında geliştirilen uygulamada veri seti olarak Alman Trafik İşareti Algılama Benchmark (GTSDB) kullanılmıştır. Tanıma işlemi için YOLO (You Only Look Once) algoritması kullanılmıştır. "NumPy" kütüphanesi ile

geliştirilmiştir. Çalışmanın sonucunda %97,22'ye ulaşan başarım elde edilmiştir (Sichkar & Kolyubin, 2020).

Bölüm 2'de derin öğrenme, Evrişimsel Sinir Ağları, standartlaşmış Evrişimsel Sinir Ağı mimarileri ve eğitilmiş bir sinir ağının performans değerlendirme ölçütleri anlatılmıştır.

Bölüm 3'te Veri kümesinin detayları ve önerilen yöntemin sonuçları paylaşılmıştır.

Bölüm 4'te Standartlaşmış mimariler olan LeNet, AlexNet, VGGNet ve GoogleNet trafik levhalarını tanıma problemine uyarlanmış ve sonuçları paylaşılmıştır.

2. EVRİŞİMSEL SİNİR AĞ MODELLERİ

Yapay sinir ağları canlılardaki sinir sisteminden esinlenmiştir. Beyindeki öğrenme sürecini nöronlar arasındaki iletişim oluşturmaktadır. Bu sayede öğrenebilen modeller inşa etmek mümkün olmaktadır. Akson ve dentritlerin birbirine bağlanması, sinyallerin bu networkten akması ile beyne gelerek bir anlam oluşturması işleyişinin benzeri bir mantığa sahiptir. Yapay sinir ağlarının doğuşu, aynı zamanda yapay sinir ağlarının mimarı olarak da görülen 1943 yılında nöroloji doktoru Warren McCulloch ve matematikçi Walter Pitt'in birlikte modellemesi sayesinde olmuştur(McCulloch & Pitts, 1943).

1957'de bu çalışmanın temelleri üzerine inşa edilmiş ve yazı karakterlerini anlamlandırmak üzerine hazırlanmış bir yapay sinir ağı Frank Rosenblatt tarafından geliştirilmiştir. Frank Rosenblatt geliştirdiği nöron 400 photocell ve 512 dirençten meydana gelen iki nöron tabakasından oluşmuş bir sinir ağıdır. Frank Rosenblatt'ın algılayıcı (perceptron)'u keşfi bu alandaki çalışmalara ivme kazandırmıştır. Modellenen algılayıcı bir katmanı olan ve tek çıkıştan meydana gelmiş bir yapay sinir ağıdır(Öztürk, 2020; Rosenblatt, 1961).

1969'da Marvin Minsk tarafından öne sürülen 2 katmanlı ağların her problem için kullanılamayacağı tezi xor problemi ile doğrulandı. Xor problemi doğrusal ayrılabilir bir problem değildir. Mesela or ve and problemlerinin çıktıları düzlemde 2 gruba ayrılabiliyor iken xor'da 2 gruba ayırma durumu gerçekleştirilememektedir (Minsky & Papert, 1969).

1984 yılında Donalt Hopfield'ın bilgisayarlar aracılığı ile çözümü mümkün olmadığına inanılan problemlerin yapay sinir ağları sayesinde çözülebileceğine dair teorisi dikkatleri tekrardan yapay sinir ağlarına çevirmiştir.

1986 yılında David Rumelhart'ın çalışmaları yapay sinir ağlarının çok katmanlı olacak şekilde tasarlanabileceği ve bu sayede Xor ve daha bir çok doğrusal olmayan komplike problemin yapay sinir ağları sayesinde çözülebileceğini ortaya koymuştur(Rumelhart ve diğ., 1986).

Yann Le Cun tarafından 1988 yılında geliştirilmeye başlanan LeNet 1998 yılında evrişim ve havuzlama katmanlarını art arda kullanan bir mimariyle karşımıza çıkmıştır(Yann Le Cun, 1988).

Tek algılayıcı model ile başlayan modeller daha sonrasında çok katmanlı algılayıcı modeller halini alarak günümüzde kullandığımız derin öğrenme yöntemlerini sunmuştur. Tek katmanlı algılayıcıların geliştirildikleri dönemde çok katmalı (Multilayer Perceptron) sinir ağları da geliştirilebilirdi. Fakat buna engel iki sorun ancak yakın tarihlerde çözüme kavuşturulabilmiştir. Bu sorunlar; veri yetersizliği ve bu veriyi işleyebilecek kapasitedeki donanımların olmamasıdır. Yapay zekâ mantık olarak klasik yaklaşımları bir kenara bırakıp veri işleme yani otomasyon mantığından sıyrılmak ve böylece öznitelik çıkarımı yapılarak sonuçlara ulaşılması demektir (Kızrak, n.d.).

2.1. Derin Öğrenme (Deep learning)

Derin öğrenme öncesinde yeterli veri bulunmaması sebebi ile bir proje için araştırmacının problemi analiz ederek bir model çıkarması gerekmekteydi. Yapay zekâ teknolojileriyle ise öz nitelikleri temel alarak ihtimal dâhilindeki verilerin modellemelerinin gerçekleştirmesi mümkün hale gelmiştir. Diş tedavisinden askeriyeye kadar birçok alanda kullanılabilir bir yöntem olan derin öğrenme üzerine özellikle doğal dil işleme, ses işleme ve görüntü işleme alanında ciddi çalışmalar yapılmaktadır.

Günümüzde tanıma projeleri çok yüksek başarım oranlarını derin öğrenme sayesinde gerçekleştirebilmektedir. Derin öğrenmenin gerçekleştirilebilmesi için öncelikle katman, ileri besleme, geri yayılma vb. yapay sinir ağı kavramlarının iyice anlaşılması gerekmektedir. İlerleyen kısımlarda yapay sinir ağları üzerinde matematiksel işlemlerden faydalanılarak bu kavramlar detaylandırılmıştır.

İleri besleme veya ileri yayılma sürecinde giriş katmanından çıkış katmanına doğru veri akışı gerçekleşir. Katmanda bulunan nöronlar arası herhangi bir işlem yapılmaz yani bu nöronlar arasında bağlantı bulunmamaktadır. İşlem sırasında veriler ilk katmana girilir, daha sonra ardışık katmanlar arasında ileri doğru çıktı katmanına ulaşılana kadar iletilir.

İlk olarak derin öğrenmenin yapı taşı olan nöron yapısı incelenmelidir. Buradaki en önemli soru neden böyle bir yapının modellenmesi gerektiğidir. Nöronlar sayesinde insan

veya hayvan gibi bir canlı organizmanın karar yeteneği kopyalanmak istenmektedir (Torres ve diğ., 2019).

Modelin başarımının artması için iterasyon sayısının artırımı bir yol olabilirken her zaman iterasyon sayısı ile öğrenme doğru orantılı değildir. Yapay sinir ağlarında girişlere göre sonuç üreten çıkış katmanı (output layer) bulunur. Sinir ağının eğitiminde bu giriş verileri ve çıkış verileri yüklenerek öğrenmenin gerçekleşmesi beklenir. Çıkış katmanındaki nöron giriş seviyesinden gelen verilerin iletim ağı üzerindeki ağırlıklarından etkilenmektedir. Kendisine bağlı olan komşu nöronlardan gelen sinyalleri sinir ağı üzerinde bulanan ağırlıklar ile çarparak problemin çözümüne en uygun aktivasyon fonksiyonu seçilerek bu fonksiyondan geçirildikten sonra bir veri elde edilir. Bu veri öteki komşu nöronlara iletilerek sonuç elde edilir. Elde edilen bu veride aktivasyon çıktısının sonuç uzayını kaydırmak için bias değeri kullanılmaktadır. Konuya ilişkin formül (2.1)'dir Bias nöron yapısının canlılığını korumak için ilk değer olarak seçilen aynı zamanda yerel optimumların aşılmasını sağlayarak öğrenimi arttırıcı etkisi olan bir değerdir.

$$z = F(w.x + b) \tag{2.1}$$

z: x değerine bağımlıdır, girdiye ait skoru verir

x: bağımsız değişken, yani girdidir

w: ağırlık parametresi (önyargı yahut önem derecesi, filtre)

b: bias değeri

Yapay sinir ağı ya da Derin Öğrenme modelinde yapılan temel işlem; modelin en iyi başarımı sağlayacağı "w" ve "b" parametrelerinin tespit edilmesidir. "w" değerleri ağırlıkları, önem derecesini (filtreleri) ifade etmektedir. Doğrusal regresyonda parametrelerinden ağırlık ve biasın birbirine göre durumu eğim ve kesişim noktasına benzerdir. Doğrusal regresyon, gözetimli öğrenmenin bir türü olan ve Türkçesi bağınımlı olan regresyon, iki ya da daha çok değişken arasındaki doğrusal ilişkinin fonksiyonel şeklidir. Ancak bu değişkenlerin tamamının niceliksel ölçekli olması zorunluluğu vardır. Bu fonksiyon aynı zamanda noktaların tamamına en yakın geçen doğrudur. "z"

hesaplandıktan sonra, bir aktivasyon işlemine tabi tutularak elde edilen çıktı bir sonraki katmana iletilir (Hinton & Osindero, 2006).

Aktivasyon fonksiyonu doğrusal olmak zorunda değildir. Bu, bir sinir ağının aynı anda birden fazla lineer modeli bir araya getirebilmesinin yanında sinir ağının sonuç etiketlerine bağlı olarak doğrusal olmayan problemlerde uygulanabilmesine yardımcı olur. Model eğitilirken çıktı katmanında doğru sonuçlar tahmini sonuçları ile karşılaştırılır, ağırlık ve bias bu karşılaştırma işlemi yapılırken hatayı yani loss denilen kayıp değeri daha uygun bir seviyeye getirebilmek için geri yayılım denen teknik ile tekrar tekrar hesaplanır (LeCun ve diğ., 2015).

2.1.1. Evrişimsel Sinir Ağları

Evrişimsel Sinir Ağı (ConNN*1 - ESA) insanların görme mekanizmasını baz alan bir modeldir. İnsanlarda göze gelen görüntü verisinin yorumlanması sonucu karar mekanizması ile ne olduğu kanısına varmaktadır. Aynı şekilde ConNN modelinde de önce gelen veriler anlamlandırılmakta, daha sonra uygun bir etiket almaktadır. Araştırmalarda resmi bütün olarak algıladığımız anlaşılmıştır. Detay özelliklerin anlaşılabilmesi daha iyi bir incelemeyle mümkündür. Örneğin gece karanlığında yoldan geçen bir hayvanın rengi gibi detay özellikleri ayırt edilememektedir. Ancak gözdeki koni hücreleri sayesinde genel hatlarıyla geçen hayvanın kuş veya köpek değil kedi olduğunun ayırdına varılabilmektedir. Bu durum insan beyninin bütünsel olarak algılamasından kaynaklanmaktadır (Sabesan ve diğ., 2016).

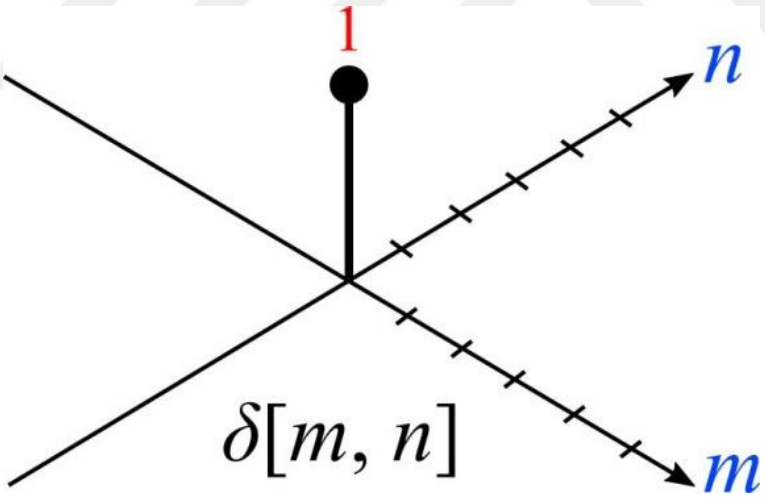
Deneyimsel olarak edinilen kriterlerin (örnek verecek olursak bu fil için uzun hortum yahut geniş kulaklar olabilir, tavşan için uzun kulaklar olabilir.) sağlanmış olduğu hallerde bilinen en yüksek benzerlik sahibi etiket sonuç olarak seçilmektedir. ConNN'de filtre yapısı, insanda ki, fili hortumundan etiketleyebilmemize yardımcı olan tanıma sistematiğidir. ConNN birbiri ile iletişim içinde ve bağlantılı nöronlardan oluşmaktadır (Tiryaki, 2019) ConNN model yapısı katmanlı bir mimariye sahiptir.

[1]∗Evrişimsel Sinir Ağları, CNN veya CoNN değil, ConNN olarak kısaltılmıştır, CNN Hücresel Yapay Sinir Ağları'nın kısaltması olarak kullanılmıştır ve ConNN, literatürde uzun zamandır Kooperatif sinir ağlarının kısaltması olarak kullanılmaktadır.

2.1.2. 2D Evrişim

Evrişim işlemleri sinyal işleme uygulamalarında çokça kullanılmaktadır. Birim impuls (D(m,n), Şekil 2.1) cevabı bilinen bir sisteme giriş olarak uygulanan bir sinyalin, sistemden çıkışının bulunmasında kullanılır. Ne işe yaradığı ne yaptığı bilinmeyen sistemlerin bilinebilmesi için kullanılır. Özetle sistemin işleyişine dair bir fikir edinmemize yardımcı olur. Evrişim, 1D'de ses işleme, 2D'de görüntü işlemede ve 3D'de akıştaki görüntülerin yani videoların işlenmesinde kullanılabilmektedir. Trafik levhalarının tanınması uygulamasında görüntü işlemede kullanılacağından 2D Evrişim tercih edilecektir. 2D Evrişim, **ConNN** temel katmanlarındandır. Özellik çıkarımı (ham pikselleri, daha anlamlı ve işlevsel verilere dönüştürme) için kullanılmaktadır. Bir görüntünün filtrelenmesi Evrişim işlemi ile yapılır.

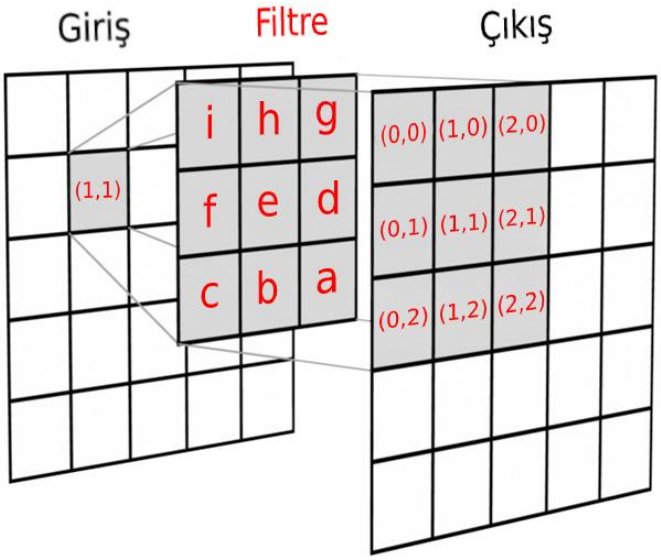
2D Evrişim görsellerle(Şekil 2.3, Şekil 2.4,) izah edilmiştir. Evrişim bir pikselin çıkış değeri kendi ve komşularının çarpılarak ağırlıklı toplam olarak bulunur (Şekil 2.2).



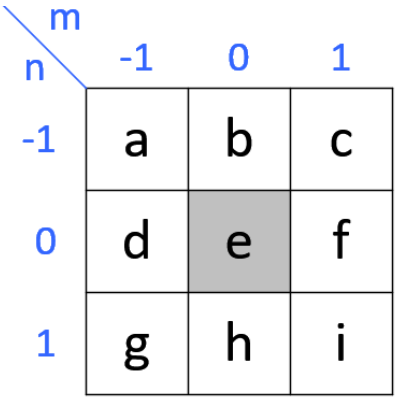
Şekil 2.1. Birim İmpuls

$$\left(\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} * \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}\right)[2,2] = (i\cdot 1)+(h\cdot 2)+(g\cdot 3)+(f\cdot 4)+(e\cdot 5)+(d\cdot 6)+(c\cdot 7)+(b\cdot 8)+(a\cdot 9).$$

Şekil 2.2. Evrişim İşlemi



Şekil 2.3. Filtre Uygulama Örneği



Şekil 2.4. Filtre Örneği

y [m,n] : sonuç sinyali

x [m, n] : giriş sinyali

h [m, n] : dürtü yanıtı(impuls)

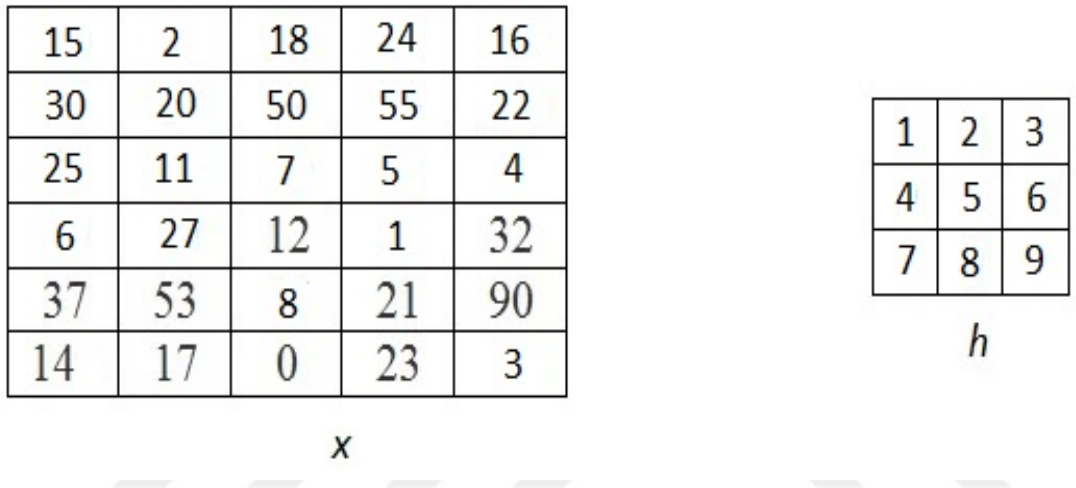
$$x[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot h[m-i, n-j] \quad (2.2)$$

$$y[m,n] = x[m,n] * h[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot h[m-i, n-j] \quad (2.3)$$

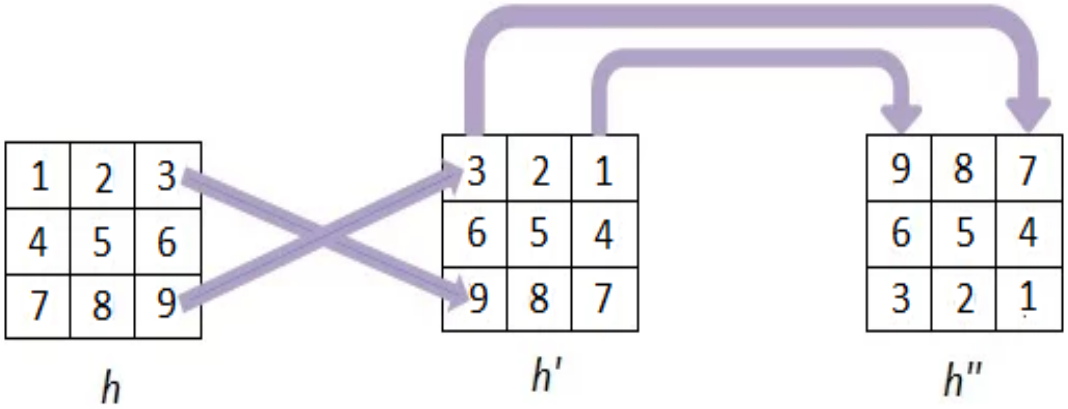
$$y[1,1] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot h[1-i, 1-j] \quad (2.4)$$

$$= x[0,0].h[1,1] + x[1,0].h[0,1] + x[2,0].h[-1,1] + x[0,1].h[1,0] + x[1,1].h[0,0] + x[2,1].h[-1,0] + x[0,2].h[1,-1] + x[1,2].h[0,-1] + x[2,2].h[-1,-1]$$

İki üçe üç matrisimiz varsa, birincisi filtre (çekirdek) ve ikincisi bir görüntü parçası ise, evrişim, çekirdeğin hem satırlarını hem de sütunlarını çevirme ve lokasyon olarak üst üste gelen girişleri çarpma ve toplama işlemidir. Ortaya çıkan görüntünün [2, 2] koordinatlarındaki eleman (yani merkezi eleman), çekirdek tarafından verilen ağırlıklar ile görüntü matrisinin tüm girişlerinin ağırlıklı bir birleşimi olacaktır.



Şekil 2.5. Evrişim İşlemi İçin Örnek Girdi Ve Filtre



Şekil 2.6. Evriştirilmiş Filtre

Çıkışını bulmak istediğimiz piksel (2,3);

2.9+12.8+24.7+20.6+50.5+55.4+11.3+7.2+5.1=924

15	2^{9}	18^{8}	24^{7}	16
30	20^{6}	50^{5}	55^{4}	22
25	11^{3}	7^{2}	5^{1}	4
6	27	12	1	32
37	53	8	21	90
14	17	0	23	3

x

Şekil 2.7. Çıkış matrisinin (2,3) Elamanın Hesaplanması

Tüm çıkış görüntülerini bulmak için bu işlem tekrarlanmalıdır. Filtrenin ağırlıklar üzerinde gezdirilişi örnekler ile (Şekil 2.8., Şekil 2.9., Şekil 2.10., Şekil 2.11., Şekil 2.12., Şekil 2.13., Şekil 2.14., Şekil 2.15.) anlatılmıştır. Önce sağdan sola, sonra aşağıdan yukarıya çekirdek kaydırmak için Her bir nokta için 5x5x3'ten 75 nokta elde edilir.

15^{9}	2^{8}	18^{7}	24	16
30^{6}	20^{5}	50^{4}	55	22
25^{3}	11^{2}	7^{1}	5	4
6	27	12	1	32
37	53	8	21	90
14	17	0	23	3

x

Şekil 2.8. (2,2) Konumlu Pikselin Evrişim Sonucu

Elde edilecek yeni matristeki her elaman için 75+ bias değeri kadar öğrenilmesi gereken parametre bulunmaktadır (Şekil 2.16.). Örnek görselleri paylaşılmıştır.

15	2^{9}	18^{8}	24^{7}	16
30	20^{6}	50^{5}	55^{4}	22
25	11^{3}	7^{2}	5^{1}	4
6	27	12	1	32
37	53	8	21	90
14	17	0	23	3

x

Şekil 2.9. (2,3) Konumlu Pikselin Evrişim Sonucu

15	2	18^{9}	24^{8}	16^{7}
30	20	50^{6}	55^{5}	22^{4}
25	11	7^{3}	5^{2}	4^{1}
6	27	12	1	32
37	53	8	21	90
14	17	0	23	3

x

Şekil 2.10. (2,4) Konumlu Pikselin Evrişim Sonucu

15	2	18	24^{9}	16^{8}	^{7}
30	20	50	55^{6}	22^{5}	^{4}
25	11	7	5^{3}	4^{2}	^{1}
6	27	12	1	32	
37	53	8	21	90	
14	17	0	23	3	

x

Şekil 2.11. (2,5) Konumlu Pikselin Evrişim Sonucu

15	2	18	24	16^{9}	8	7
30	20	50	55	22^{6}	5	4
25	11	7	5	4^{3}	2	1
6	27	12	1	32		
37	53	8	21	90		
14	17	0	23	3		

x

Şekil 2.12. (2,6) Konumlu Pikselin Evrişim Sonucu

15	2	18	24	16
30^{9}	20^{8}	50^{7}	55	22
25^{6}	11^{5}	7^{4}	5	4
6^{3}	27^{2}	12^{1}	1	32
37	53	8	21	90
14	17	0	23	3

x

Şekil 2.13. (3,2) Konumlu Pikselin Evrişim Sonucu

15	2	18	24	16
30	20	50	55	22
25^{9}	11^{8}	7^{7}	5	4
6^{6}	27^{5}	12^{4}	1	32
37^{3}	53^{2}	8^{1}	21	90
14	17	0	23	3

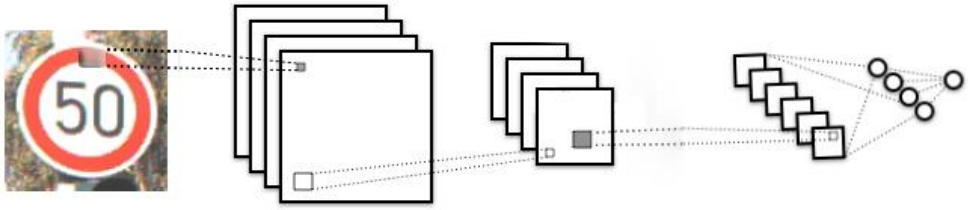
x

Şekil 2.14. (4,2) Konumlu Pikselin Evrişim Sonucu

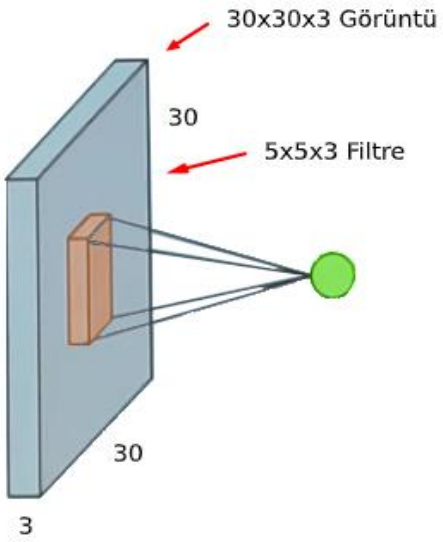
15	2	18	24	16
30	20	50	55	22
25	11	7	5	4
6^{9}	27^{8}	12^{7}	1	32
37^{6}	53^{5}	8^{4}	21	90
14^{3}	17^{2}	0^{1}	23	3

x

Şekil 2.15. (5,2) Konumlu Pikselin Evrişim Sonucu



Şekil 2.16. Görüntüye Filtre Uygulama



Şekil 2.17. Resmin Evrişimsel Sinir Ağında Geçirdiği Aşamalar

Evrişim diğer bir deyişle, bir görüntünün (bilinen boyut ve ölçülerdeki) matris(filtre) ile çarpılan ve elde edilen matrisin öğelerini yeni bir matrise ekleyen işlemdir (Şekil 2.17.). Bu işlem, orijinal görüntünün geri kalanında basamaklar halinde bilinen düzenli adımlarla

tekrarlanır ve yeni bir görüntü elde edilir. Evrişim katmanı her biri farklı bir filtrenin sonucu olan bir görüntü dizini yahut yığını olarak adlandırılabilir. Bu yığındaki her görüntü bir özellik matrisidir (feature map). Filtreler evrişim katmanın ağırlıklarıdır (w). ConNN'nin eğitimi sınıflandırma probleminin çözümünde kullanılacak en uygun özellikleri bulma aşaması demektir. Bu katmanda karşılaşılan problem filtrenin görüntüye tam oturmamasıdır. Bu problem, kenarlarının kesilmesi yahut sıfır eklenerek tamamlanması yöntemi ile çözülebilmektedir. Bu problemin çözümü için kayıplarının oluşmaması adına görüntünün kenarlarına sıfır eklemek ve benzeri dolgulama yöntemleri tercih edilmektedir.

2.1.3. Evrişim Katmanı (Convolutional Layer)

Görüntü tanıma işlemlerinde en önemli özelliklerden biri kenar özelliğidir. Kenar bulmak için geleneksel yöntemler Sobel, Prewitt, Gabor gibi yöntemler mevcuttur. Evrişimsel sinir ağlarında görüntüler özellik çıkarımı için evrişim işlemine tabi tutulur. Elde edilen çıktı işlenen görüntünün kenar bilgilerini de içerir. Uygulanan filtreler sayesinde elde edilen geçişler (aydınlıktan karanlığa yahut karanlıktan aydınlığa) öznitelik olarak değerlendirilmekte ve buna göre gerekli hesaplamalar yapılmaktadır. Bu işlemlerin yapıldığı sırada uygulanan filtre ve görüntü matrisinin boyutuna göre en dışta kalan piksellerde kayıplar oluşabilmektedir. Bu sebeple bazı önemli veriler kaybolabilmektedir. Aynı zamanda görüntü matrisinin boyutu değişmektedir. Önemli verilerin kaybolmasını önlemek ve boyutun korunmasını sağlamak için piksel ekleme (padding) işlemi yapılmaktadır.

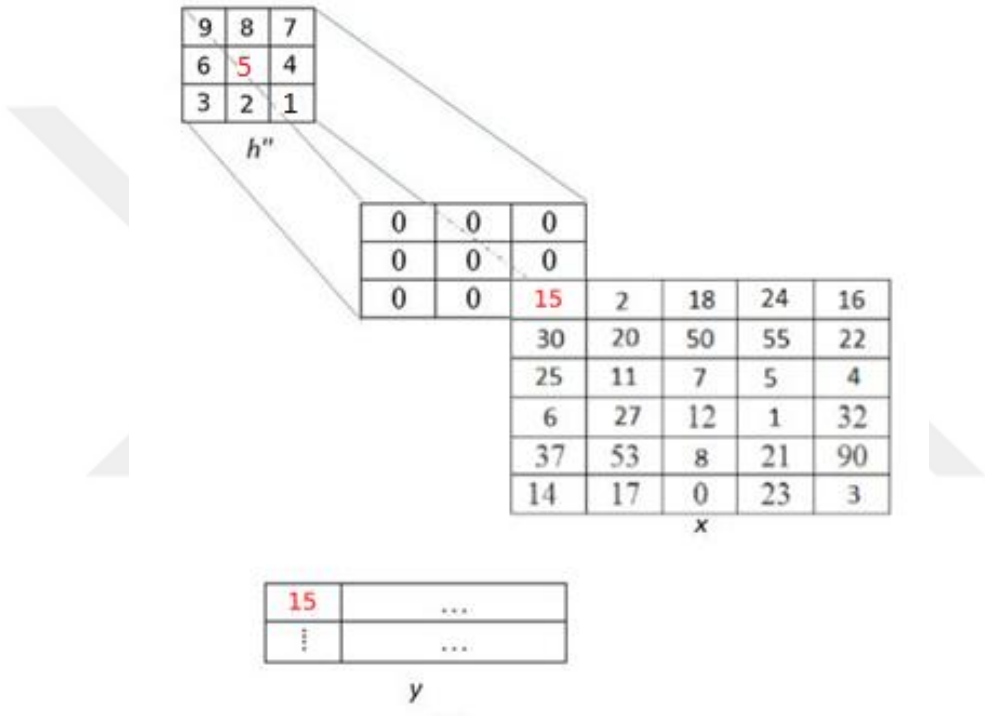
Piksel ekleme, evrişim işlemi sonrasında görüntünün dış sınırlarında kalan bazı piksellerin filtre uygulanması sebebi ile önemsizleşmesini engellemek için kullanılan bir yöntemdir. Üzerinde çalışılan görüntü matrisine en dışardaki pikselleri hesaplamaya dâhil edecek şekilde yeni matris elemanları eklenir. Eklenecek olan piksel sayısı aşağıdaki formül (2.5) yardımı ile hesaplanır. Ekstra sıfır ekleme işlemi, sıfır dolgusu olarak bilinir ve çekirdek pikselleriyle çakışacak görüntü piksellerinin olmadığı her durumda yapılması gerekir. Kenarlar problemleri ile başa çıkmak için burada sıfır dolgulama (zero padding) kullanılmıştır.

p: Piksel

f: Filtre

$$p = \frac{f-1}{2} \tag{2.5}$$

Tensorflowda “conv1_pad = ‘SAME’” metodu ile çözdüğümüz problemin evrişim ardından dolgu uygulanmış hali Şekil 2.18.’deki görseller ile anlatılmıştır. Yapılan işlem filtrenin resim üzerinde kaydırılması ve kaydırılırken üst üste denk gelen hücrelerdeki verilerin çarpılıp toplanarak yeni bir hücreye yazılması işlemidir.



Şekil 2.18. Dolgu Uygulama Örneği

Diğer dolgu teknikleri arasında tekrarlanan dolgu, periyodik uzatma, yansıtma vb. (Gonzalez & Woods, 2008).

Evrişim işlemi sırasında uygulanan filtre görüntü matrisine kaydırılarak uygulanmaktadır. Kaydırma işlemi bir ve daha yüksek adım aralığı olacak şekilde yapılabilmektedir. Bu işlem çıkış matrisi boyutunu doğrudan etkileyen bir işlemdir. Yeni görüntü matrisinin buyutu (2.6)’daki formülde gösterildiği gibi hesaplanır.

n: resim boyutu

p: Piksel Ekleme(Padding)

f: Filtre Boyutu

s: Kaydırma Adım Boyutu

nx n ‘lik bir görüntü için f x f’lik bir filtre s adım boyuyla uygulandığında sonuçta elde edilecek olan görüntünün boyutları aşağıdaki formüldeki gibi hesaplanmaktadır.

$$\left[\frac{n+2p-f}{s}+1\right] \quad x \quad \left[\frac{n+2p-f}{s}+1\right] \tag{2.6}$$

2.1.4. Havuzlama Katmanı (Pooling)

Gerçek dünyada görüntüler birçok özelliğe sahip olmasına karşın her özellik matrisi, tek özellik olarak iş görmektedir. Örneğin, insan yüzlerini hayvan yüzlerinden veya cansız nesnelerin görüntülerinden ayırt etmek istiyorsak, sadece göz, burun ve kulakların özelliklerini tanımlamamız yeterli olmamaktadır. Bu sebepten, görüntülerin karmaşıklığı arttıkça daha fazla filtreye gereksinim duyulmaktadır. Bu durumda karşılaşılan temel sorunlar bilgisayar donanım sorunları yani bellek kısıtlılığıdır. İşlemci hızı problemleri gibi problemler çalışılacak konu görüntü işleme gibi ağır işlem yükü olan konular için büyük bir dezavantajdır. Bu dezavantajları ortadan kaldırmak için işlem yükünü azaltacak çözümler önerilmiştir. Bu yöntem modele havuzlama katmanı eklenmesidir.

Bu yöntemle oluşturulan katmanın temel işlevi alınan veriden sub-samples denilen alt uzaylar (alt örnekler) oluşturmaktır. Bu sayede yoğun işlem yükü hafifletilmektedir. Bir çekirdek ve bir havuzlama işlemi ile tanımlanır. Filtre, evrişim katmanında yapılan işlemlere benzer biçimde burada da ortalama veya maksimum gibi bir havuzlama (pooling) işlemi, uyguladığımız gelen verinin alt örnekleminin boyutunu tanımlamaktadır. Pooling işlemi o bölgedeki piksellerin bir fonksiyon uygulanmış halini elde etmemizi sağlamaktadır. Bir görüntüyü 3-D yani yükseklik, genişlik ve kanal yani renkler olarak ele alınmaktadır.

Görüntüyü filtreye uygulayarak (2.7)’deki formül ile indirgenmiş olur.

$$\text{Görüntünün Yeni Boyutu} = \frac{\text{Yükseklik} * \text{Genişlik}}{\text{Kaydırma Adımı}} \quad (2.7)$$

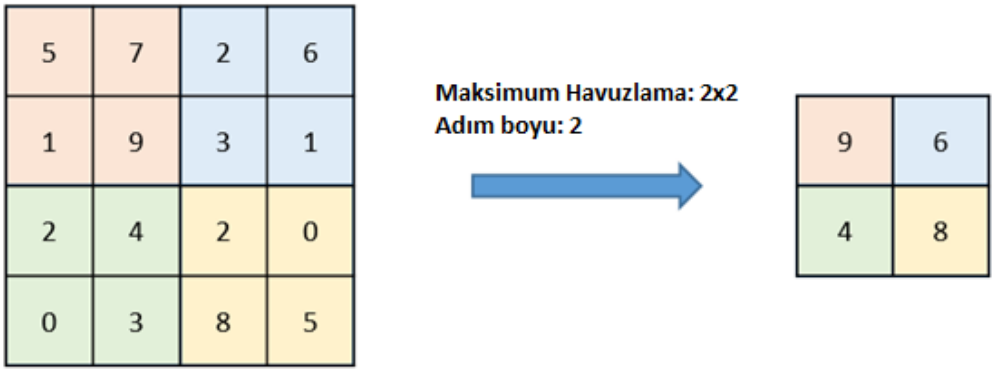
Yani (3,3) boyutlu filtre ve 3 adıma sahip olan pooling katmanın kenarları üçte birine düşerken, asıl resmimiz ile yeni üretilen resmin pikselleri arasında dokuz kat fark oluşmaktadır. Bu da işlem yapma hızımızda ciddi oranda artış sağlamanın yanında işlem yükünü ciddi oranda azaltmaktadır. Havuzlama katmanı ve evrişim katmanı ConNN modelimiz için temel mimari olsa da kullanım sıralamaları farklılık gösterebilmektedir. Havuzlama katmanı öncesinde birkaç evrişim katmanı uygulanıp sonrasında tekrar havuzlama katmanının uygulanması hazırlanan modelde başarımı artırmak için kullanılabilmektedir. ConNN modelleme görüntü işleme gibi zor alanlarda ciddi başarılar göstermektedir, daha fazla özellikli yani daha derin ağların tasarlanama sebebi derin öğrenmenin fazla uyumluluk, hafıza kullanımı ve ağın eğitilmesinde geçen sürenin uzun olması gibi problemlerinin olmasıdır. Her ConNN modelin en yüksek başarımı sağlayacak evrişim ve havuzlama katmanı sıralaması, uygun parametrelerin bulunması gerekliliği sebebi ile modele has tasarlanmaktadır. Evrişim ve havuzlama işlemleri ardından elde edilen özellikleri hafızada tutarak düzleştirme işlemi uygulandıktan sonra görüntünün hangi sınıf ile etiketleneceği belirlenir.

Havuzlama işlemi elde edilen özellik matrislerini küçülterek işlem hızını artırmaktadır. Havuzlama katmanları; özellikleri, görüntüdeki ufak bozulmalardan, görüntü döndürme veya eğme gibi farklılıklardan etkilenmemesini sağlamaktadır.

En çok kullanılan 3 tür pooling;

- Maksimum havuzlama
- Ortalama havuzlama
- Toplama havuzlama

Maksimum havuzlama en yaygın kullanılan pooling metodudur. Maksimum havuzlama, evrişim filtresindeki maksimum değeri alır. (Şekil 2.19)'da maksimum havuzlama örneği gösterilmektedir.



Şekil 2.19. Maksimum Havuzlama

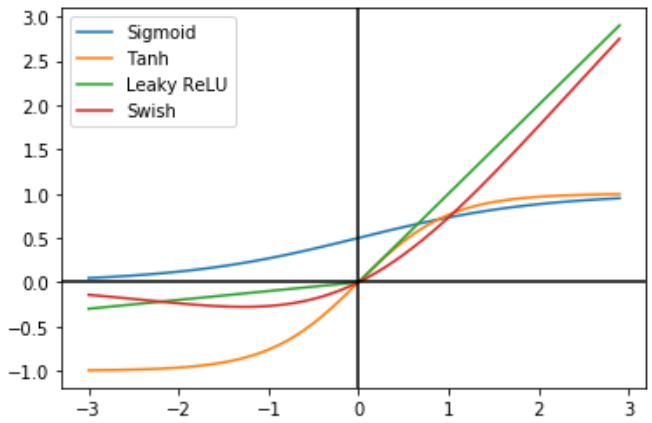
2.1.5. Düzleştirme katmanı (Flattening Layer)

Önceki katmanlardan gelen görüntü mantrisini tek boyutlu bir dizi haline getirilmesi işlemidir.

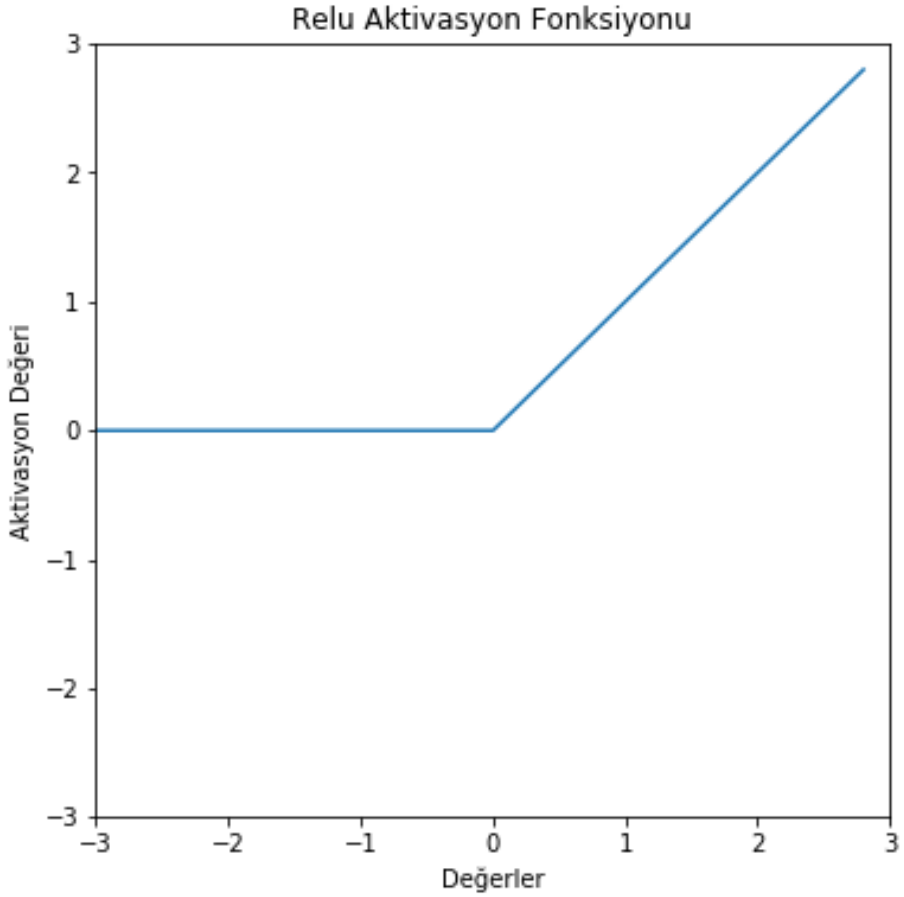
2.1.6. Tam Bağlantılı Katman (Fully Connected Layer)

Evrişimsel ağın bir parçası olarak, evrişim / havuzlama işleminin sonucunu alan ve sınıflandırmanın kararına varan katmana tamamen bağlı katman (fully connected) denir. Önceki iki katmandan geçirilerek elde edilmiş veri, düzleştirme (flatten) işlemine tabi tutulduktan sonra ReLU aktivasyon fonksiyonu (Şekil 2.20., Şekil 2.21.) uygulanıp yüzdelik olasılıklarla hangi sınıfa ait olduğu verisi elde edilir. Örnek olarak %70 durma levhası olarak öngörülen görüntünün sola dönüş yasağını ifade etmesi durumunda bir hata hesaplanmaktadır. Yapay sinir ağları bağlamında, bu hesaplamaya "maliyet fonksiyonu" denmektedir. Ancak evrişimsel sinir ağları çalışılırken, yaygın olarak "kayıp fonksiyonu" olarak adlandırılır. Kayıp fonksiyonları ağın ne oranda doğru olduğunu bildirmesi, başarımının artırılması için gerekli optimizasyonun kullanılmasını sağlar. Ağın optimizasyonu için veriler geriye yayılım ve ileri yayılım işlemleri devam ettirilerek istenilen noktaya gelinene kadar tekrarlanır. Bu işlem sonunda verinin sınıfı belirlenmiş olur.

```
In [6]: runfile('C:/Users/busra.kosesoy.ULASIMPARK/Desk
Desktop/spider')
```



Şekil 2.20. Aktivasyon Fonksyonları



Şekil 2.21. ReLU Aktivasyon Fonksyonu

2.2. Standartlaşmış Ervişimsel Ağı Mimarileri

2.2.1. LeNet

Yann LeCun ve arkadaşları tarafından 1989'da önerilen bir evrişimsel sinir ağları modelidir. Evrişimsel sinir ağları yapay sinir ağlarından ileri beslemeli sinir ağları mantığı ile çalışan sinir ağlarıdır. Görüntü işleme problemlerinde yüksek başarıma sahiptir. Geri yayılım algoritmaları temel alınarak hazırlanmış evrişimsel sinir ağları ilk olarak el yazısı sayıları okumak amacıyla geliştirilmiştir. ABD Posta servisinde el yazısı ile yazılmış posta kodlarını tanımada başarılı şekilde kullanılmıştır (LeCun ve diğ., 1989).

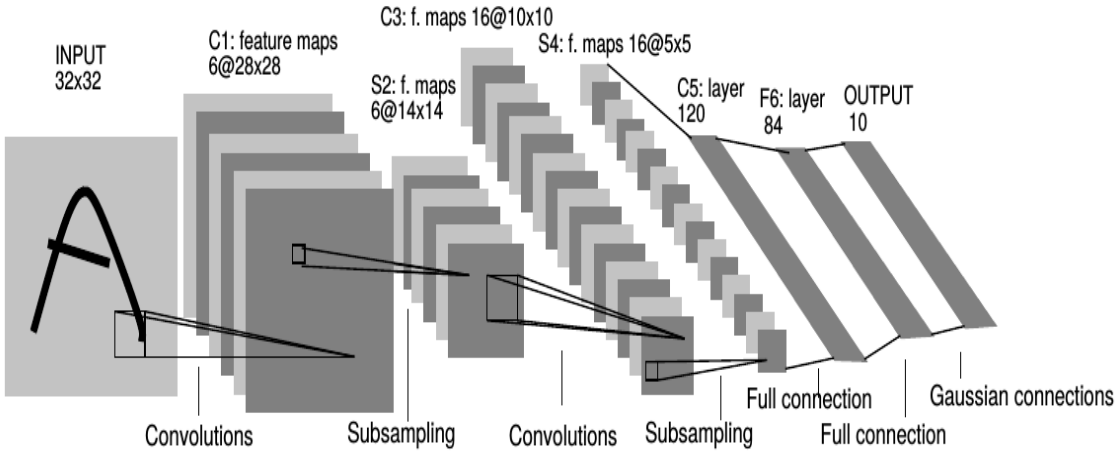
LeCun yine 1989 yılında başka bir makale yayınlayarak lineer olarak ayrılabilen el yazısı ile yazılmış rakamları tek katmanlı sinir ağları ile sınıflandırma kullanarak sonuçlarını paylaşmıştır. Bu çalışmada tek katmanlı sinir ağlarının bu tarz problemlerde yetersiz kaldığı gözlemlenmiştir. Çok katmanlı sinir ağlarında bu problemin iyi sonuçlar verdiği ortaya konmuştur (Lecun ve diğ., 1998). İlk çalışmalar MNIST (Modified National Institute of Standards and Technology) veri seti üzerinde yapılmıştır.

Evrişimsel sinir ağları modellerinin öncüsü olan LeNet Evrişim katmanı, havuzlama katmanı ve tam bağlantı katmanlarından meydana gelmektedir.

Orijinal makale sonradan geliştirilecek olan modellerden iki noktada ayrılmaktadır. İlki boyut azaltma için maksimum hvuzlama yerine ortalama(average) pooling işlemi yapılmasıdır. İkinci farklılık aktivasyon fonksyonu olarak sigmoid ve hiperbolik tanjant kullanılmasıdır. Rakam sınıflandırma veri seti üzerinde çalıştığından 10 sınıflı bir problem olarak tanımlanmış. Orijinal makaledenalınmış görsel (Şekil 2.22.)'de gösterilmiştir.

LeNet mimarisi basitliği ile beraber bellek alanı açısından da kullanıcılara oldukça avantaj sağlamaktadır. Modern ConNN'ler için temel model olarak kabul görmektedir. İlk olarak rakam ve karakter tanıma için kullanılmıştır. Bunlardan en bilineni olan MNIST veri seti üzerinde geniş bir kullanım alanına sahiptir (Şekil 2.23.). Tek kanal sayısına sahip (tek katmanlı) görüntü üzerine 2 evrişim, 2 havuzlama, tam bağlantı ve ardından softmax ile 10 sınıf (rakamlar veri seti için 10 sınıf) çıktısı üretmektedir. 2012

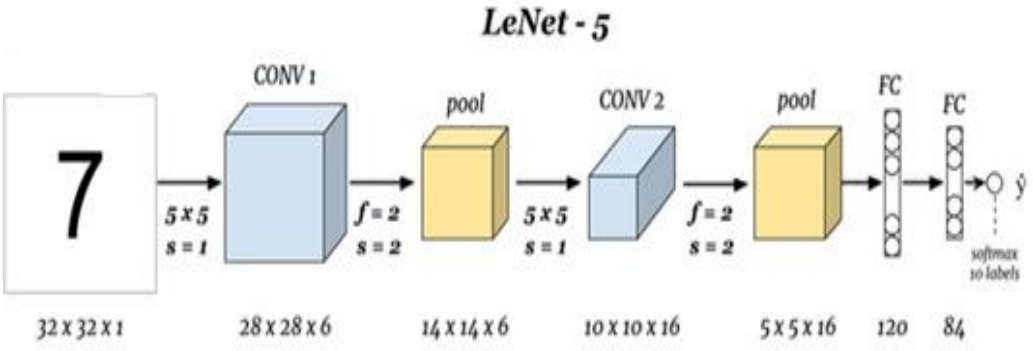
yılına kadarki 14 yılda geliştirme ortamlarında kendine yer bulamamıştır. 2012'deki yarışmadan sonra AlexNet sayesinde adından söz ettirmiştir.



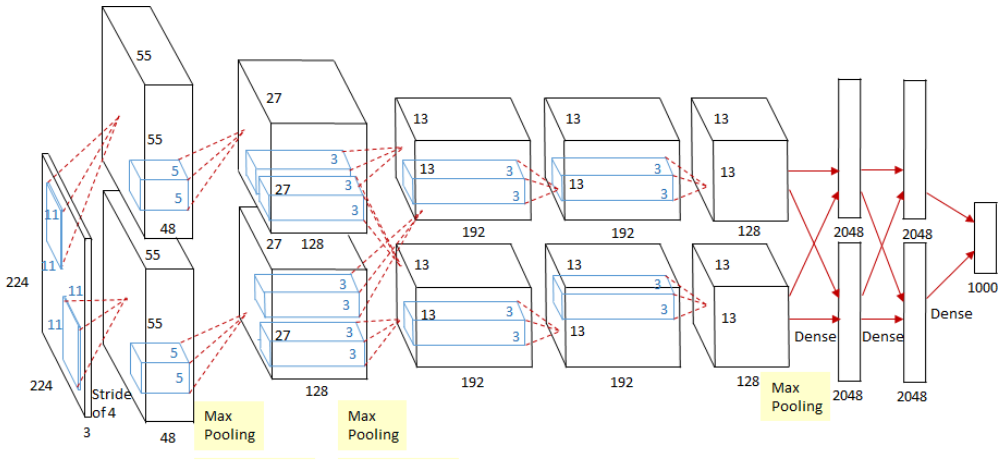
Şekil 2.22. LeNet Mimarisi

2.2.2. AlexNet

30 Eylül 2012 ImageNet'in Large Scale Visual Recognition Challenge (ILSVRC) yarışmasında en yakın rakibine performans açısından ciddi bir fark atarak yarışmada birinci olmuştur. Mimari olarak LeNet'e benzerlikler göstermektedir. LeNet'e göre daha fazla sayıda katmana sahip bir yapısı vardır. 5 evrişim katmanı, 3 tam bağlantı katmanı (Fully Connected) barındırır. Her katmanın ardından ReLU aktivasyonu uygulanmaktadır (Şekil 2.24).



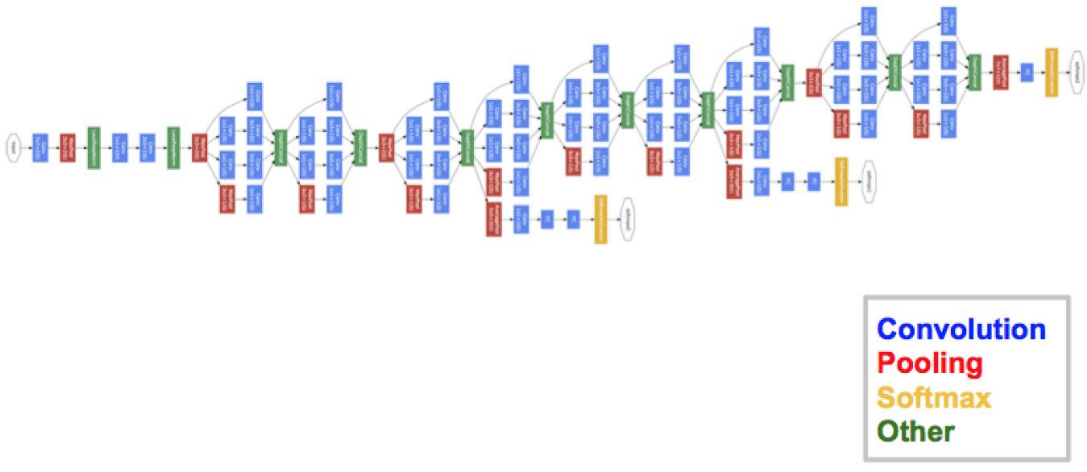
Şekil 2.23. LeNet Model Örneği



Şekil 2.24. İki GPU arasındaki sorumlulukların tanımını açıkça gösteren ConNN mimarisinin bir örneği. Bir GPU, şeklin üstündeki katman parçalarını çalıştırırken, diğeri alttaki katman parçalarını çalıştırır. GPU'lar yalnızca belirli katmanlarda iletişim kurar. Ağın girişi 150.528 boyutludur ve ağın kalan katmanlarındaki nöron sayısı 253.440–186.624–64,896–64,896–43,264–4096–4096–1000 olarak verilmektedir.

2.2.3. GoogleNet (Inception)

ImageNet'in 2014 yarışmasını kazanmıştır. Öteki mimarilerden farkı art arda dizilmiş katmanlar yerine paralel katmanlar şeklinde modellenmiş bir mimaridir. Bu sayede hem aşırı uyum (overfitting) problemi denilen ezberin önüne geçilmiş hem de bellek ve hesaplama maliyetinden kâr edilmiştir. Orijinal makaleden alınmış görsel (Şekil 2.26.)'da eklenmiştir.

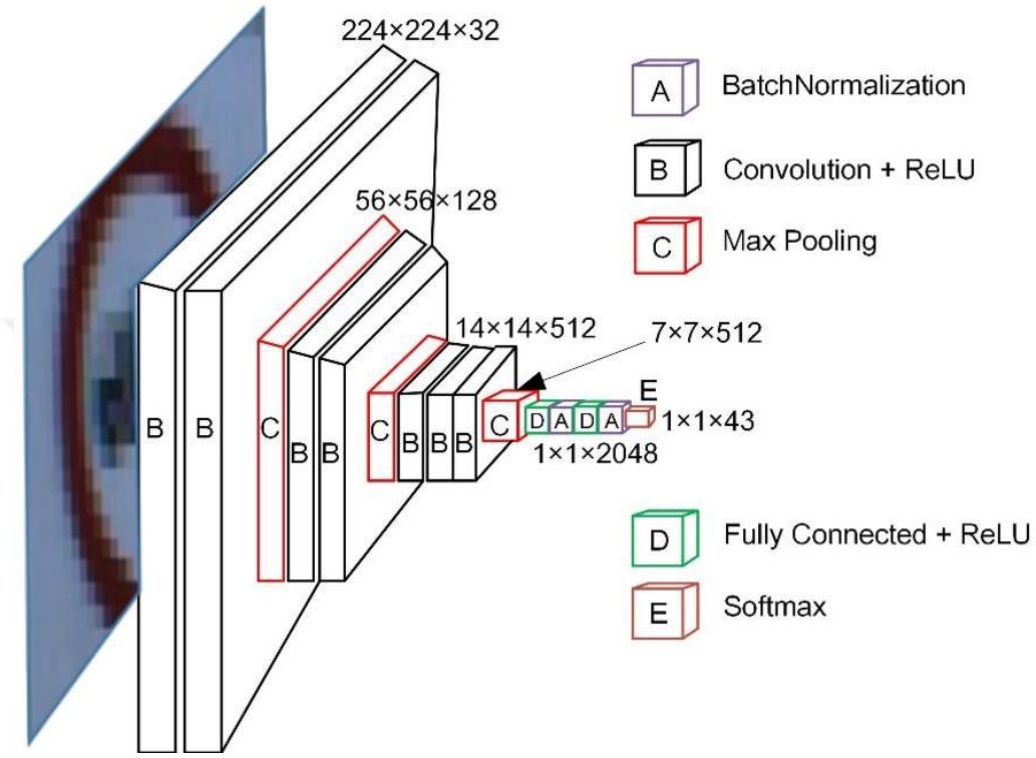


Şekil 2.25. GoogleNet Mimarisi

2.2.4. VGGNet

Visual Geometry Group (VGG) laboratuvarından Karen Simonyan ve Andrew Zisserman tarafından geliştirilmiştir. 3x3'lük filtreli evrişim ve 2x2'lik filtreli havuzlama katmanları

mevcuttur. Her iki katman arasına softmax sınıflandırıcı yerleştirilmiştir. Kullandığı parametre sayısının fazla olması ve geniş bellek ihtiyacı dolayısı ile maliyetli oluşu VGGNet'in dezavantajıdır. Orijinal makaleden alınmış görsel (Şekil 2.26.)'te eklenmiştir.

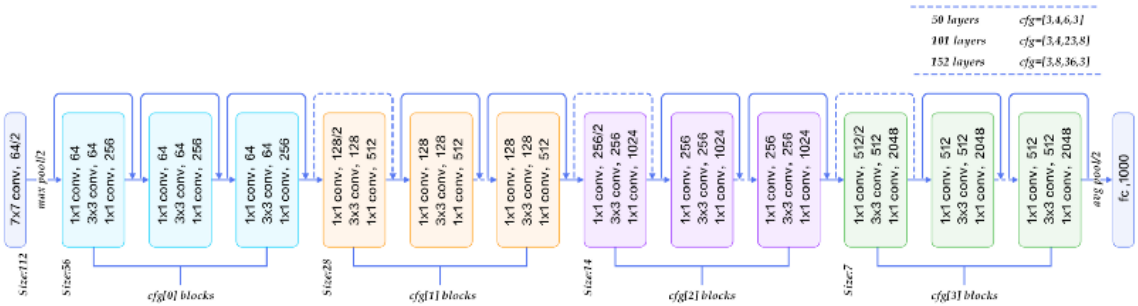


Şekil 2.26. VGGNet Mimarisi

2.2.5. ResNet

2015 yılında geliştirilerek ImageNet yarışmasını kazanmıştır. Hata oranını %3.57'ye düşürmeyi başaran Residual Networks, Microsoft Research Team tarafından geliştirilmiştir. 152 katmandan oluşmaktadır. "Vanishing Gradient" problemine bir çözüm sunmaktadır. Vanishing Gradient problemi ağ üzerinde geri besleme yapılırken bazı nöronların güçlü aktivasyonları olmaması sebebi ile ağın eğitiminden elenmesi olarak açıklanabilir. Ağın sonunda Tam Bağlantı Katmanı (Fully Connected) katmanı bulunmamaktadır. Modelin açıklayıcı olması için (Şekil 2.27.)'de katmanlarının detaylarını, filitre yapılarını ve bu neticede elde edilen çıktıları özetleyen görsel

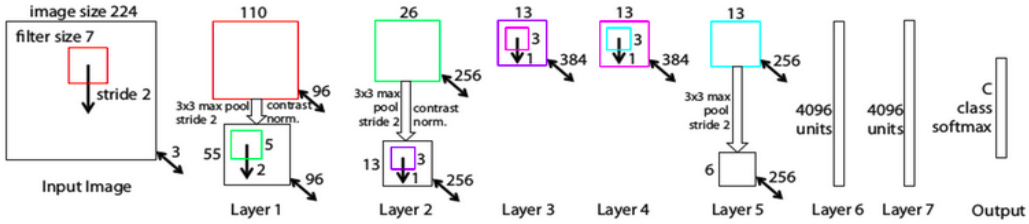
eklenmiştir.



Şekil 2.27. ResNet Mimarisi

2.2.6. ZF Net

AlexNet'in parametreleri üzerinde değişiklikler yapılarak elde edilen bir mimaridir. ZF Net modelinde filtre 7x7, adım sayısı ise 2 olacak şekilde değiştirilmiştir. Evrişim ağına sokulacak resimlerin orijinal halini koruyabilmek için filtre boyutu küçültülmüştür (İnik & Ülker, 2017).Orijinal makaleden alınmış görsel (Şekil 2.28.)'te eklenmiştir.



Şekil 2.28. ZF Net mimarisi

2.3. Performans Değerlendirme Ölçütleri

Hazırlanan modellerin değerlendirilmesi için sıklıkla kullanılan metotlardan bir tanesi karmaşıklık matrisidir (confusion matrix) (Tablo 2.2.). Karmaşıklık matrisi çok sınıflı (ikiden fazla sınıflı) problemlerde performansı ölçmek için kullanılır. Bu matris iki boyuttan oluşmaktadır; Gerçek Sınıf ve Sınıflandırıcının Tahmini. Performans değerlenrime için kullanılan ölçeklerin formülleri (2.8, 2.9, 2.10, 2.11, 2.12, 2.13, 2.14,2.15)'dir.

$$\text{Hassasiyet (Precision)} = \frac{DP}{DP+YP} \tag{2.8}$$

$$\text{Kesinlik (Recall)} = \frac{DP}{DP+YN} \tag{2.9}$$

$$\text{Doğruluk} = \frac{DP+DN}{DP+DN+YP+YN} \tag{2.10}$$

$$\text{F – Skor} = \frac{2DP}{2DP+YP+YN} \tag{2.11}$$

$$\text{Ortalama Doğruluk} = \frac{\sum_{i=1}^{l} \frac{DP_i+DN_i}{DP_i+DN_i+YN_i+YP_i}}{l} \tag{2.12}$$

$$\text{Kesinlik (Precision)} = \frac{\sum_{i=1}^{l} \frac{DP_i}{DP_i+YP_i}}{l} \tag{2.13}$$

$$\text{Ortalama Duyarlılık (Recall)} = \frac{\sum_{i=1}^{l} \frac{DP_i}{DP_i+YN_i}}{l} \tag{2.14}$$

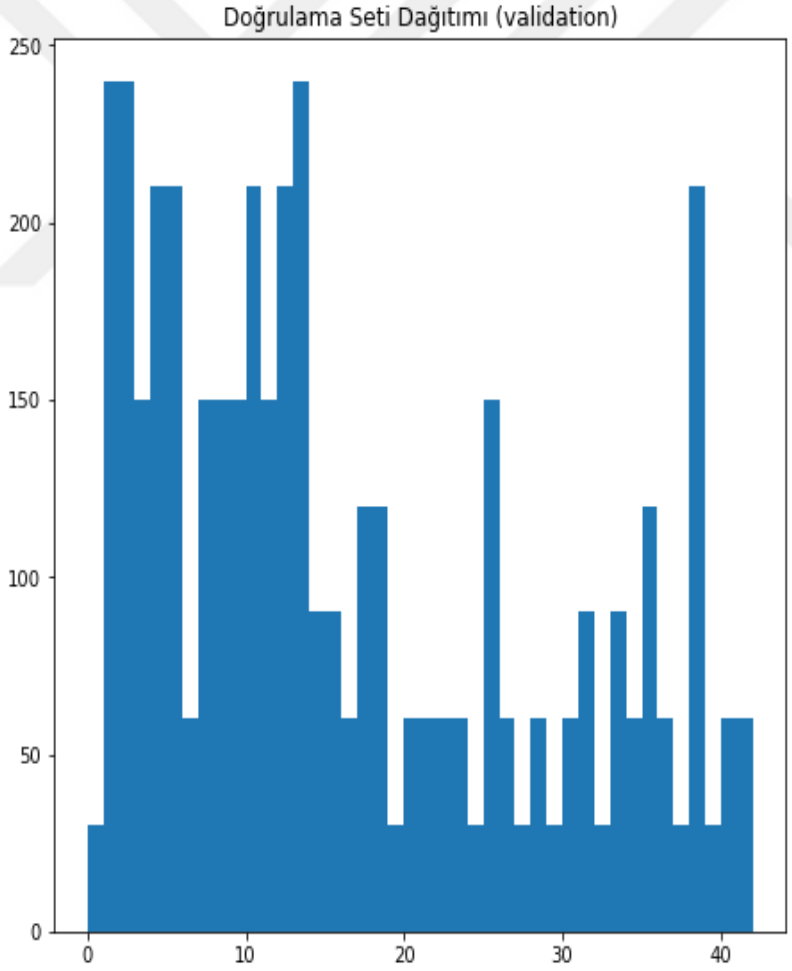
$$\text{Ortalama F-Değer} = \frac{\sum_{i=1}^{l} \frac{2DP_i}{2DP_i+YP_i+YN_i}}{l} \tag{2.15}$$

Tablo 2.1 Karmaşıklık Matris Tablosu

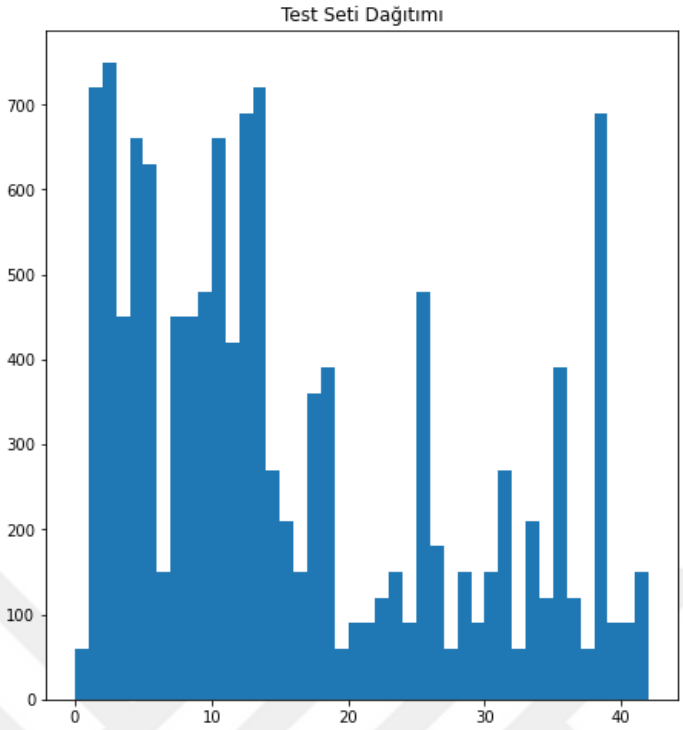
		Tahmin Edilen Sınıf	
		Pozitif	Negatif
Gerçek Sınıf	Pozitif	Doğru Pozitif (DP) (True Positive -TP)	Yanlış Pozitif (YP) (False Postive - FP)
	Negatif	Yanlış Negatif (YN) (False Negative- FN)	Doğru Negatif (DN) (True Negative - TN)

3. VERİ KÜMESİ VE ÖNERİLEN YÖNTEM

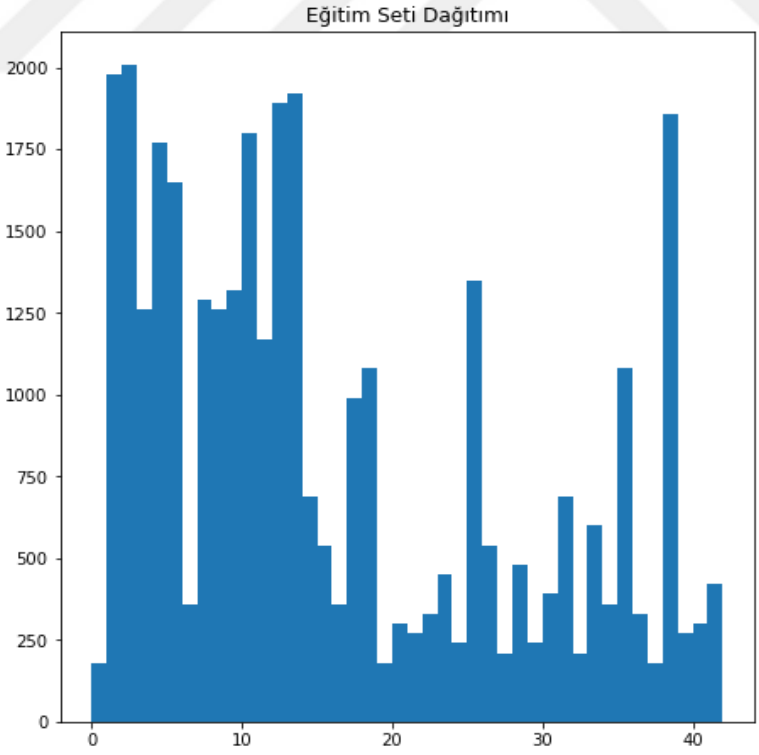
GTSRB veri seti trafik işareti tanıma sistemlerinde sıklıkla kullanılan veri setidir. 2013 yılından günümüze kadar erişilebilir durumdadır. 43 sınıfa ait trafik işareti barındırmaktadır. Toplamda veri setinde 39209 görüntü olmasına karşı bunların 4410 adedi doğrulama testi için ayrılmıştır. Eğitim ve doğrulama verilerinin ayrılması ve görüntülerin eğitimde kullanılacakları hale dönüştürülmeleri için haricen bir uygulama geliştirilmiştir. Şekil 3.1., Şekil 3.2., Şekil 3.3., Şekil 3.4., Şekil 3.5.'te veri setinin gerekli ön işlemlerden sonrasında sayısal dağılımı paylaşılmıştır. Dengesiz bir veri seti kullanıldığı için farklı performans ölçütleri ile değerlendirmeler Şekil 3.7., Şekil 3.8., Şekil 3.9., Şekil 3.10., Tablo 3.1. ve Tablo 3.2.'de paylaşılmıştır.



Şekil 3.1. Doğrulama Veri Seti Dağılımı



Şekil 3.2. Test Veri Seti Dağılımı



Şekil 3.3. Eğitim Veri Seti Dağılımı

Eğitim örneklerinin sayısı = 39209

Doğrulama örneklerinin sayısı (validation) = 4410

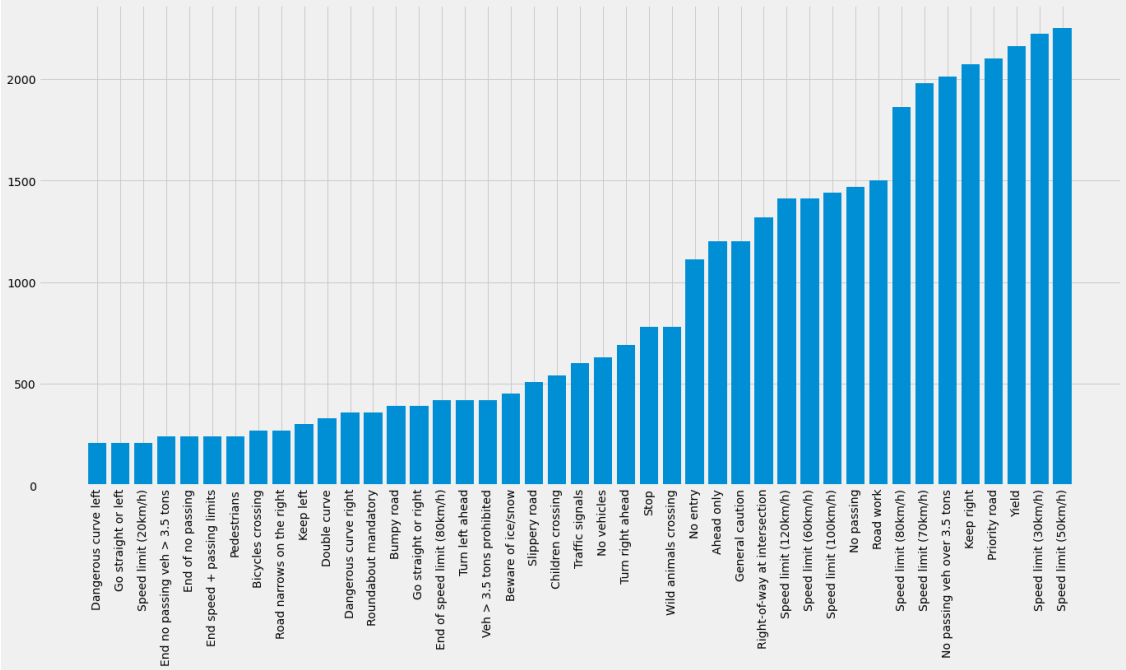
Test örneklerinin sayısı = 12630

Görüntü verilerinin boyutları = (32,32,3)

Makine öğrenmesi algoritmalarının kodlanması için dil olarak python seçilmiştir. Python'un seçilme sebebi sunmuş olduğu geniş kütüphane desteğidir. Geliştirme ortamı olarak ise veri bilimi, yapay zekâ gibi birçok alanda çalışan ve python kullanmak isteyenler için iyi bir tercih olan Anakonda dağıtımı tercih edilmiştir. Anakonda ortam yöneticisi ve açık kaynak paketleri içeren güçlü bir Python/R platformudur. İhtiyaç duyulan sürümlerin rahatça indirilebilmesi, kütüphanelerin kurulumunda sağladığı rahatlık, kod ve çalışma ortamının paylaşımındaki kolaylık Anakonda'nın sağladığı avantajlardan bir kaçıdır. Bununla birlikte ek olarak Microsoft Azure, IBM vb. iPython sürümlerini destekleyen ortamlara aktarımı sorunsuz sağlanabilmektedir.

Linux, Microsoft ve macOS için ayrı dağıtımları mevcuttur. Dikkat edilmesi gereken durumlardan bir tanesi hazırlanacak projeye göre seçilecek python sürümünün farklılık gösterebilecek olmasıdır. Bu çalışmada sürüm olarak (3.X) kullanılmaktadır. Anakonda'nın içerdiği idelerden Jupiter Notebook üzerinde geliştirme yapılmıştır.

Şekil 3.4. Her Sınıfın Etiketli Örnek Görüntüsü



Şekil 3.5. Levha Sınıflarının Sayısal Dağılımı

Jupiter Notebook; çeşitli programlama dilleri için çalışma ortamı sağlamaktadır. Bunların başında R, Ruby, Haskell, Julia gibi diller gelmektedir. Tarayıcı tabanlı çalışmaktadır. İçerisinde kod ile birlikte metin, grafik vs çıktıları barındırabilmektedir. Bu sayede raporlama kod içerisinde yapılabilmekte ve kod anlaşılırlığını artırmaktadır.

Evrişimsel sinir ağlarının sınanmasında ihtiyaç duyulan verinin bir kısmı Kaggele'dan alınmıştır. Kaggele makine öğrenmesi yarışmaları düzenleyen ve bu sayede birçok geliştiriciyi bir araya getiren bir platformdur. Bu platformda veri setleri ve elde edilen sonuçlar paylaşılmaktadır. Bu sayede geliştiriciler hazır veri setleri üzerinde çalışma imkânı bulmaktadır. Kaggele platformu aynı veri seti üzerinde çalışmış binlerce insanın grup veya bireysel olarak bir araya gelerek geliştirmiş oldukları modellerin paylaşılması ile daha gelişkin modelleri geliştirilmesine katkı sağlamaktadır.

Trafik işaretlerini tanımak üzere geliştirilen ConNN modelinin adımları aşağıda açıklanmıştır.

1. Adım: Veriler yüklenmiştir.
2. Adım: Veri kümesinin içerdiği veriler hakkında bilgi edinilmesi ve özetlenmesi yapılmıştır.
3. Adım: Görüntülerin tamamı 32x32x3 boyutlandırılmıştır.

4. Adım: Yüklenen veriler karıştırma işlemine tabi tutulmaktadır. Karıştırma (Shuffling) işlemi modelin daha kararlı olması için eğitim veri setindeki rastgeleliği ve çeşitliliği artırmak için eğitim verileri üzerine uygulanmıştır. Verilerimizi karıştırmak için sklearn kullanılmıştır.

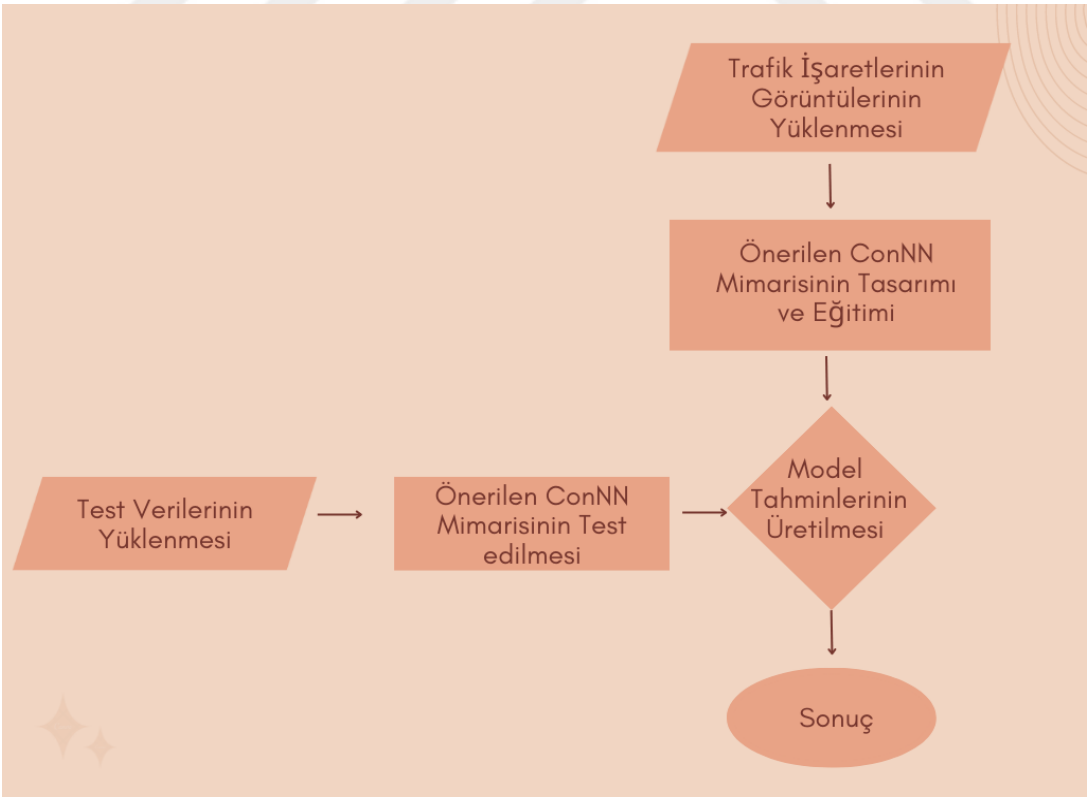
5. Adım: Görüntüler modellenen ağın en başarılı sonucu vermesi çeşitli(kaydırma, döndürme, yaklaştırma, çevirme vb) işlemlerden geçirilir. Bu sayede tanımaya en uygun hali elde edildiğinden model başarısı yükselmiştir.

6. Adım: Tasarlanan mimarinin eğitim katmanları Tablo 3.2. numaralı tabloda ifade edildiği şekliyle gerçekleştirilmiştir.

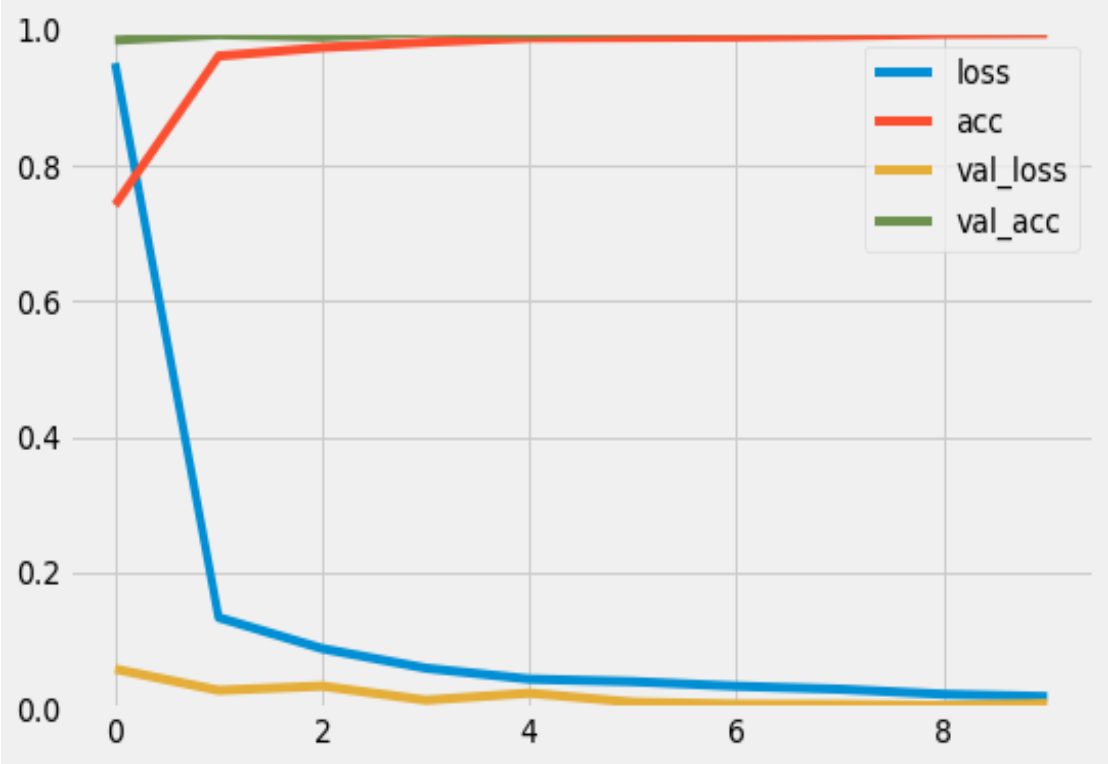
7. Adım: Eğitim tamamlandıktan sonra sonuçlar Şekil 3.8., Şekil 3.9., Şekil 3.10., Şekil 3.11., Tablo 3.1.de gösterilmiştir.

8. Adım: Model zaman performansı ölçülmek üzere kayıt edilmiştir. Mimari görselleştirilerek yapılan işlem adımları Şekil 3.8.'de, akış diyagramı ise Şekil 3.6.'da anlaşılır olması amacı ile paylaşılmıştır.

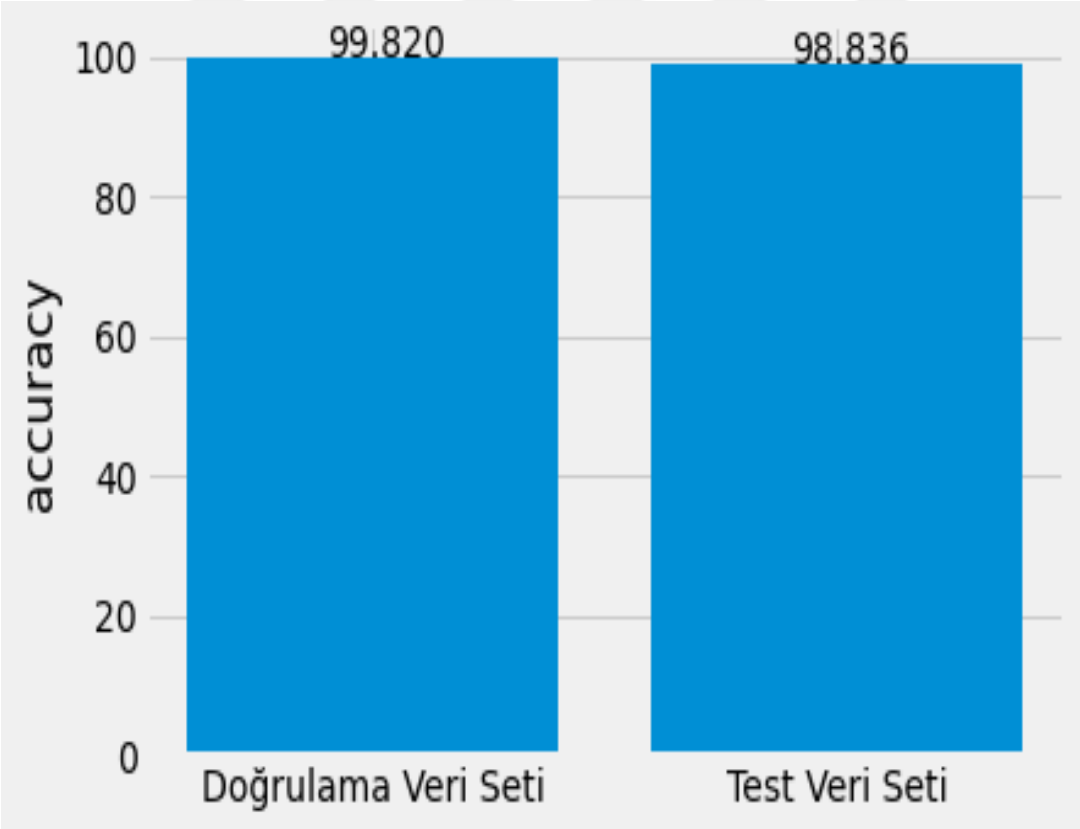
9. Adım: Kayıt edilen model farklı bir ipynb içerisinden çağırılarak zaman performansı ölçülmüş ve paylaşılmıştır.



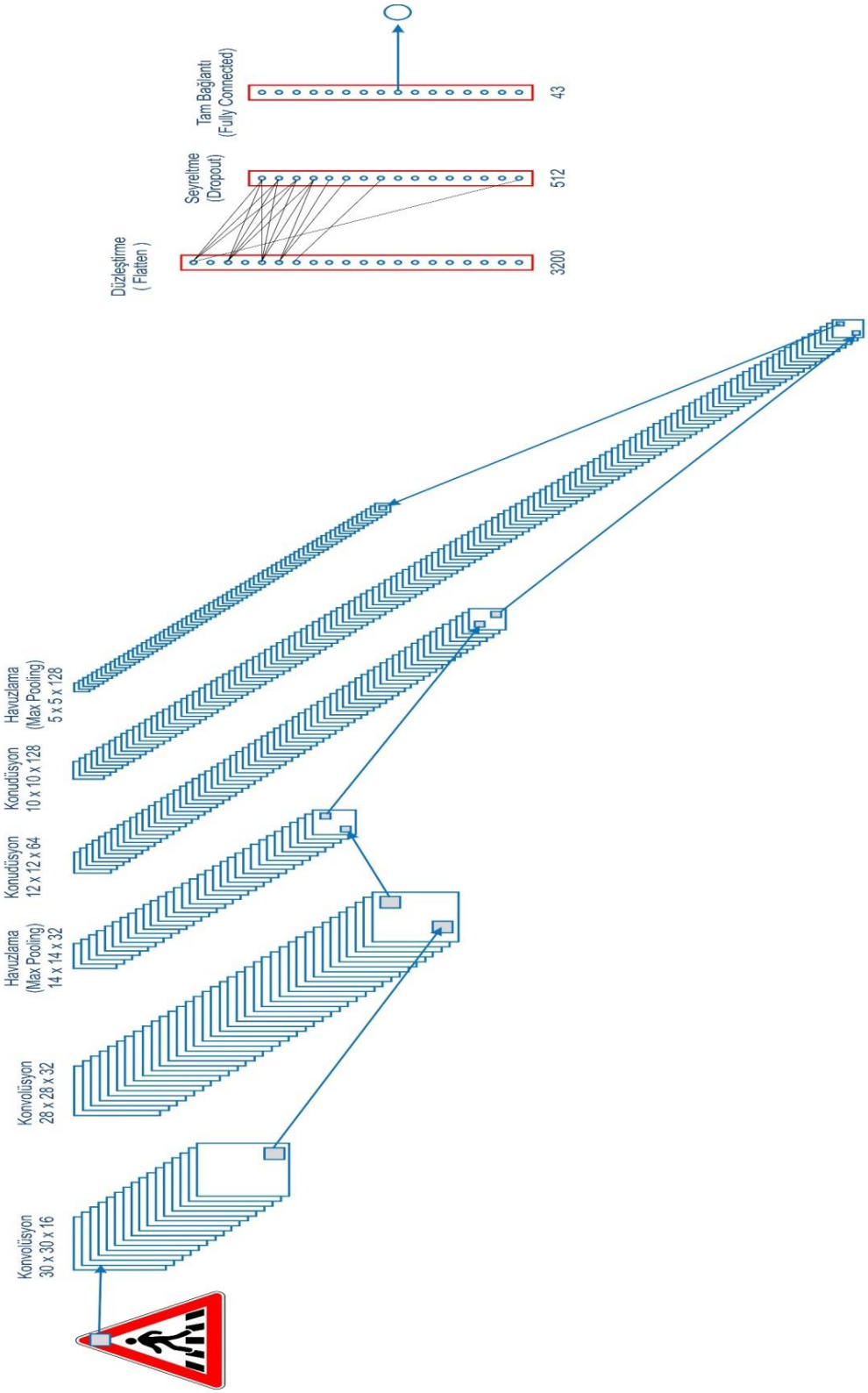
Şekil 3.6. Akış Diyagramı



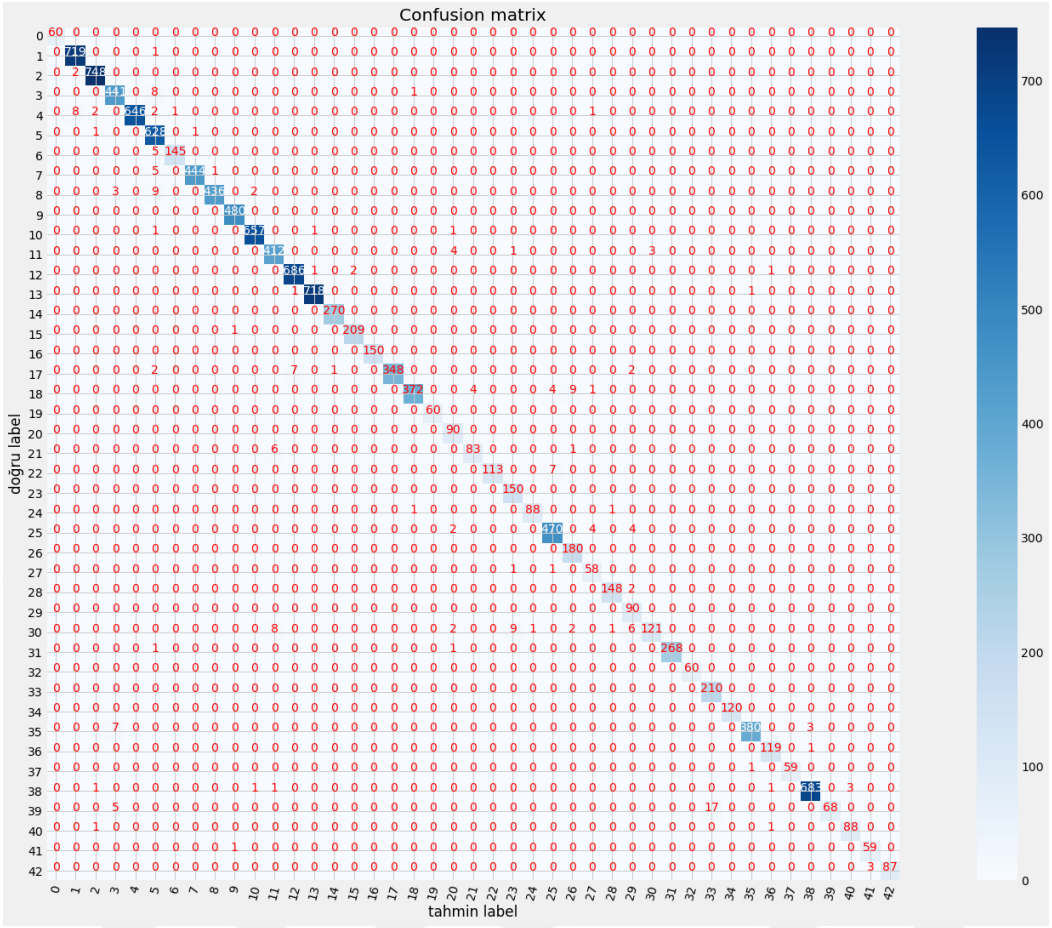
Şekil 3.7. Önerilen Mimarinin Kayıp Değeri ve Başarımı



Şekil 3.8. Önerilen Mimarinin Performans Grafiği



Şekil 3.9. Önerilen Mimarinin Temsili Katmanları



Şekil 3.10. Önerilen Mimarinin Karmaşıklık Matrisi

Şekil 3.11. Önerilen Mimarinin Test Sonuçları

Tablo 3.1. Önerilen Mimarinin Katmanları

Önerilen Modelin Mimarisi			
Katman	**Giriş**	**Açıklama**	**Çıkış**
Evrişim	32x32x3	Filtre: 3x3 Adım: 1x1 Dolgu: Valid	30x30x16
Evrişim	30x30x16	Filtre: 3x3 Adım: 1x1 Dolgu: Valid	28x28x32
Havuzlama	28x28x32	Filtre: 2x2 Adım: 2x2 Dolgu: Valid	14x14x32
Evrişim	14x14x32	Filtre: 3x3 Adım: 1x1 Dolgu: Valid	12x12x64
Evrişim	12x12x64	Filtre: 3x3 Adım: 1x1 Dolgu: Valid	10x10x128
Havuzlama	10x10x128	Filtre: 2x2 Adım: 2x2 Dolgu: Valid	5x5x128
Düzleştirme	5x5x128		3200
Tam Bağlantı	3200	Her nöron mutlak bir katmana bağlıdır.	512
Tam Bağlantı	512	Her etiket için 43 olasılık tespit edilmiştir.	43

Önerilen mimari 2 evrişim ardından 1 maksimum havuzlama tekrar 2 evrişim ve 1 maksimum havuzlama uygulanacak şekilde tasarlanmıştır. Ardından düzleştirme, seyreltme ve tam bağlantı katmanlarından geçirilerek sonuç elde edilir.

Tablo 3.2. Önerilen Yöntemin Performans Değerlendirme Ölçek Sonuçları

Mimari	**Performans ölçeği**	**Precision**	**Recall**	**F1-score**
Önerilen Mimari	**Accuracy**			0.99
	Macro avg	0.98	0.98	0.98
	Weighted avg	0.99	0.99	0.99

4. DENEYSEL SONUÇLAR

4.1. LeNet

LeNet için; 3 aşamalı bir uygulama geliştirilmiştir. İlk aşamada veriler ön işlemlerden (normalizasyon) geçirilerek proje içerisinde kullanılacak hale dönüştürülmüştür. Görüntüler üç kanallı şekilde işlenmiş ve 'picle' formatında kaydedilerek proje içerisine dâhil edilmiştir. İkinci aşamada eğitim gerçekleştirilmiştir. Üçüncü aşamada ise eğitim neticesinde elde edilen model çağırılarak sonuçların görselleştirilmesi ve performans kıyasları sunulmuştur. LeNet mimarisi için iki ayrı optimizasyon algoritması için sonuçlar elde edilmiştir.

1. Adım: Veriler yüklenmiştir.

2. Adım: Veri kümesinin içerdiği veriler hakkında bilgi edinilmesi ve özetlenmesi

Pickle uzantılı edinilen test, eğitim ve doğrulama veri seti bilgilerinin içeriği incelenmiştir. 3. Adımda ifade edilecek ön işlemlerden geçirilerek eğitimde kullanılacağı hali ile kaydedilmiştir.

Veri içeriği;

- "Features", Trafik levhalarının resimlerinin ham piksel verilerini barındıran 4 boyutlu dizisidir (değer, genişlik, yükseklik, kanallar).

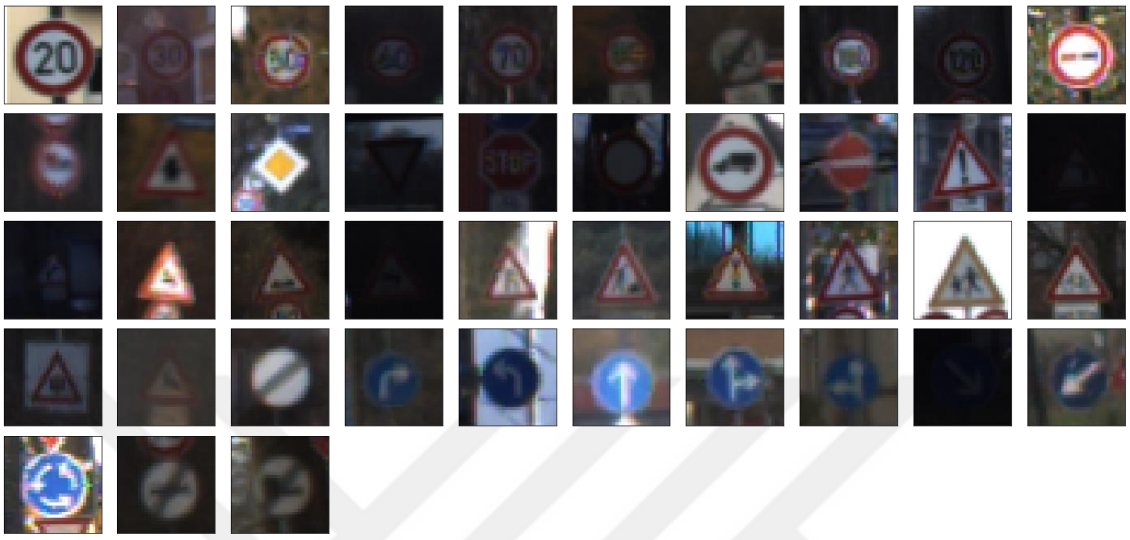
- "Labels", Trafik levhalarının sınıflarını içeren 1 boyutlu bir dizidir.

- Signnames.csv, Csv formatında olan dosya, her bir sınıf için sınıf numarasının sınıf adı ile eşlemelerini içermektedir.

43 sınıftan oluşan veri setinin her sınıfına ait bir görsel (Şekil 4.1)'de örneklendirilmiştir. Görüntülerin tamamı 32x32x3 boyutlarındadır.

3. Adım: Veri kümesinin ön işlemlerden geçirilmesi. Görüntü verilerinin eşit varyansa sahip olması için normalizasyon işlemi gerçekleştirilmiştir. Görüntü verileri için (piksel-128)/128 verileri hızlı bir şekilde normalleştirmenin kolay yollarından biridir. Normalleştirme sayesinde piksel değerleri 0 ile 1 arasında değer almakta ve böylece

kontrastı değişmiştir. Diğer bir ön işlem yöntemi gri tonlamadır. Bu işlem denenmiştir ancak gri tonlama işlemi sonrası başarım düştüğünden uygulanmasından vazgeçilmiştir.



Şekil 4.1. Her Sınıfın Örnek Görüntüsü

4. Adım: Veriler 'Pre-data.pickle' alarak kayıt edilmiş. İçeriği Test, Eğitim ve doğrulama veri seti bilgilerinin tamamını içermektedir. Bu sayede tek dosya ile üç veri setine de erişilebilmektedir.

5. Adım: Yeni bir '.ipynb' dosyası açılarak ön işlemlerden geçirilmiş veriler yüklenmiştir.

6. Adım: Yüklenen veriler karıştırma işlemine tabi tutulmaktadır. Karıştırma (Shuffling) işlemi modelin daha kararlı olması için eğitim veri setindeki rastgeleliği ve çeşitliliği artırmak için eğitim verileri üzerine uygulanmıştır. Verilerimizi karıştırmak için sklearn kullanılmıştır.

7. LeNet mimarisi tasarlanıp katmanlar (Tablo 4-1) ifade edildiği şekliyle gerçekleştir.

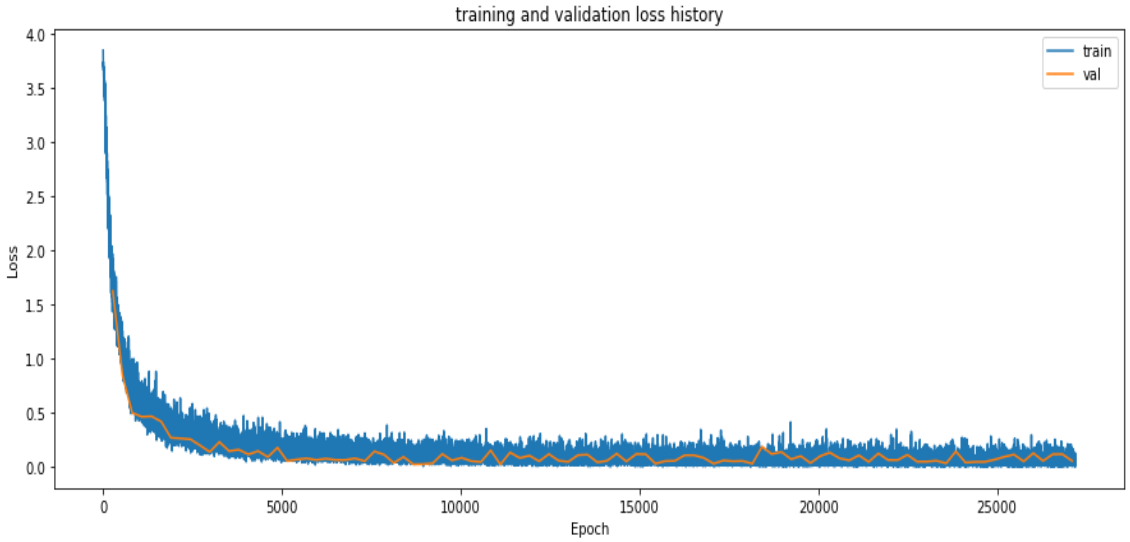
Adam optimizasyon algoritması için en yüksek başarım 0,001 öğrenme katsayısı, 100 iterasyon, 128 küme boyutu seçildiği durumda doğrulama veri setinde %96.9, test veri seti için %96.1 başarım elde edilmiştir (Tablo 4.2., Şekil 4.2., Şekil 4.3., Şekil 4.4..).

Tablo 4.1. LeNet Mimari Katmanları

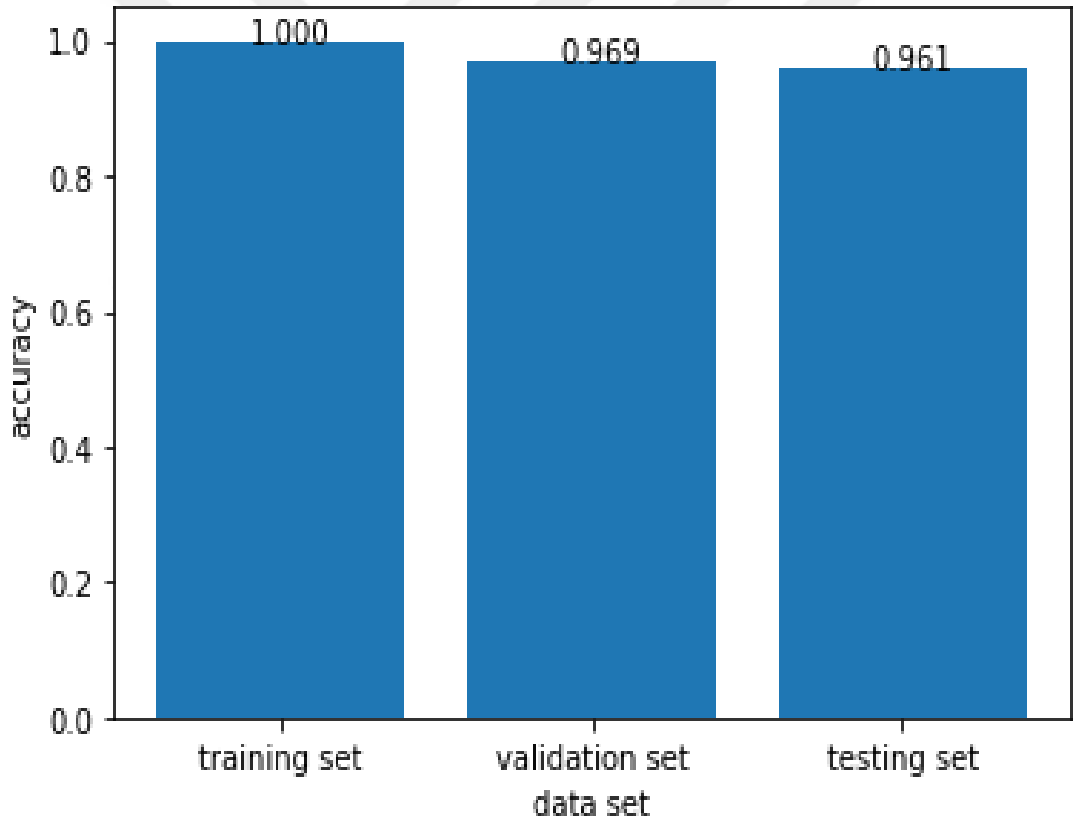
LeNet Mimarsisi			
Katman	**Giriş**	**Açıklama**	**Çıkış**
Evrişim	32x32x3	Filtre: 5x5 Adım: 1x1 Dolgu: Valid	28x28x6
Havuzlama	28x28x6	Filtre: 2x2 Adım: 2x2 Dolgu: Valid	14x14x6
Evrişim	14x14x6	Filtre: 5x5 Adım: 1x1 Dolgu: Valid	10x10x6
Havuzlama	10x10x6	Filtre: 2x2 Adım: 2x2 Dolgu: Valid	5x5x16
Düzleştirme	5x5x16		400
Tam Bağlantı	400	Her nöron mutlak bir katmana bağlıdır.	120
Tam Bağlantı	120	Her nöron mutlak bir katmana bağlıdır.	84
Tam Bağlantı	84	Her etiket için 43 olasılık tespit edilmiştir.	43

Tablo 4.2. LeNet Mimarisinin Adam Optimizasyon Algoritması Uygulanmış Halinin Performans Değerlendirme Sonuçları

Mimari	**Algoritma**	**Performans ölçeği**	**Precision**	**Recall**	**F1-score**
LeNet	**Adam**	**Accuracy**			0.96
		Macro avg	0.94	0.95	0.94
		Weighted avg	0.96	0.96	0.96

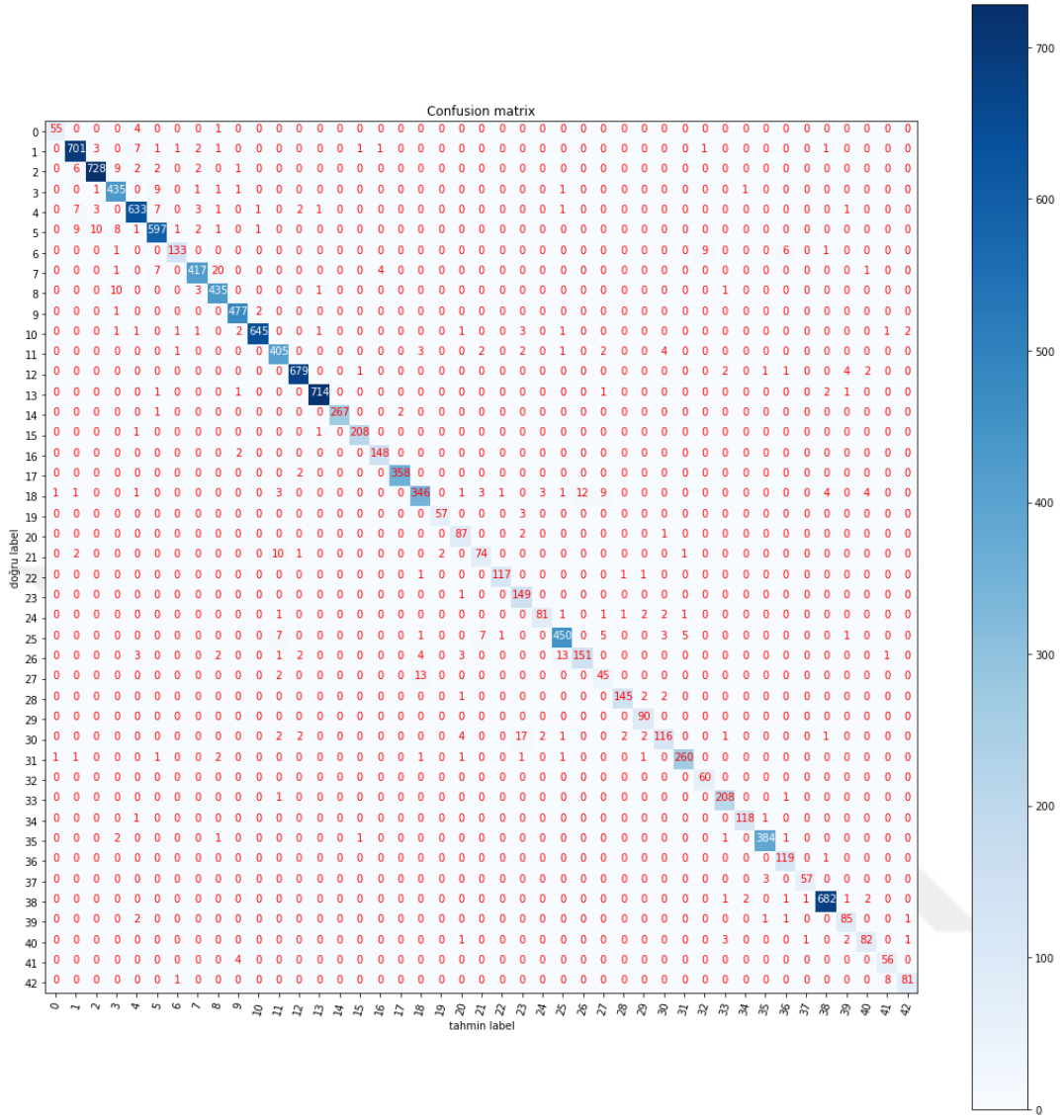


Şekil 4.2. LeNet Adam Optimizasyon Algoritması ile Denenmiş Eğitim ve Doğrulama Kayıp Grafiği



Şekil 4.3. LeNet Performas Grafiği

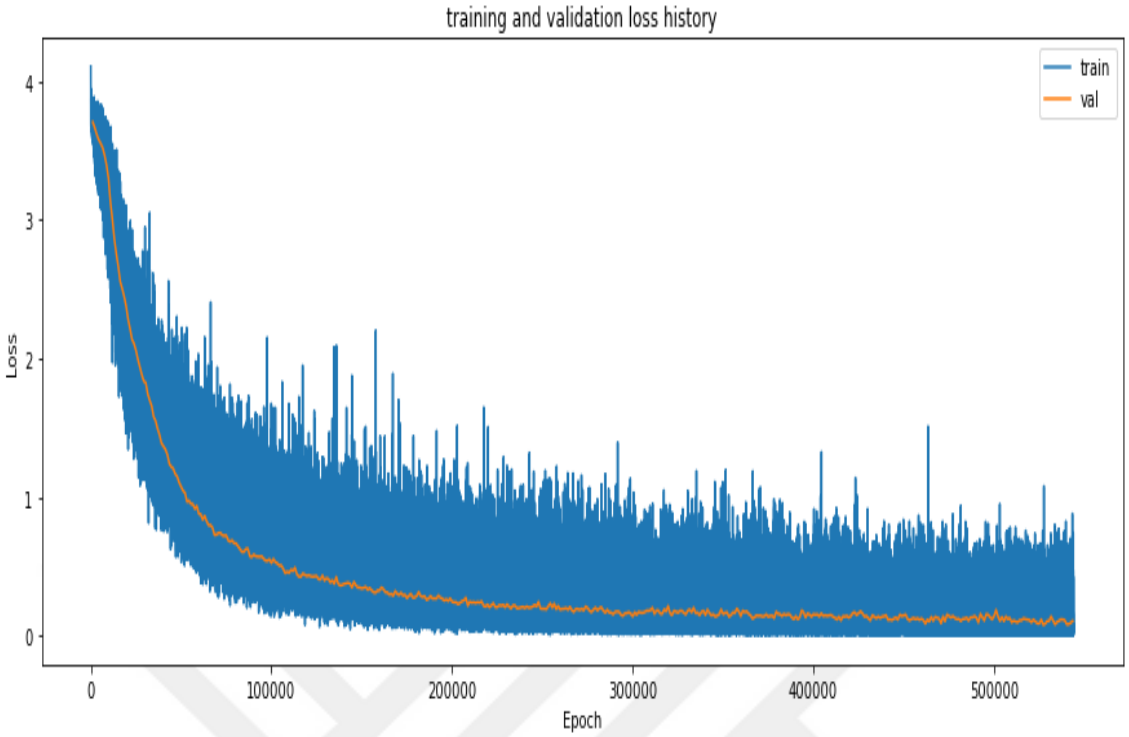
LeNet için; Gradyan iniş algortiması için en yüksek performans 0,001 öğrenme katsayısı, 500 iterasyon, 32 küme boyutu seçildiği durumda doğrulama veri setinde %95.6, test veri seti için %94.4 olarak elde edilmiştir (Tablo 4.3, Şekil 4.5, Şekil 4.6, Şekil 4.7).



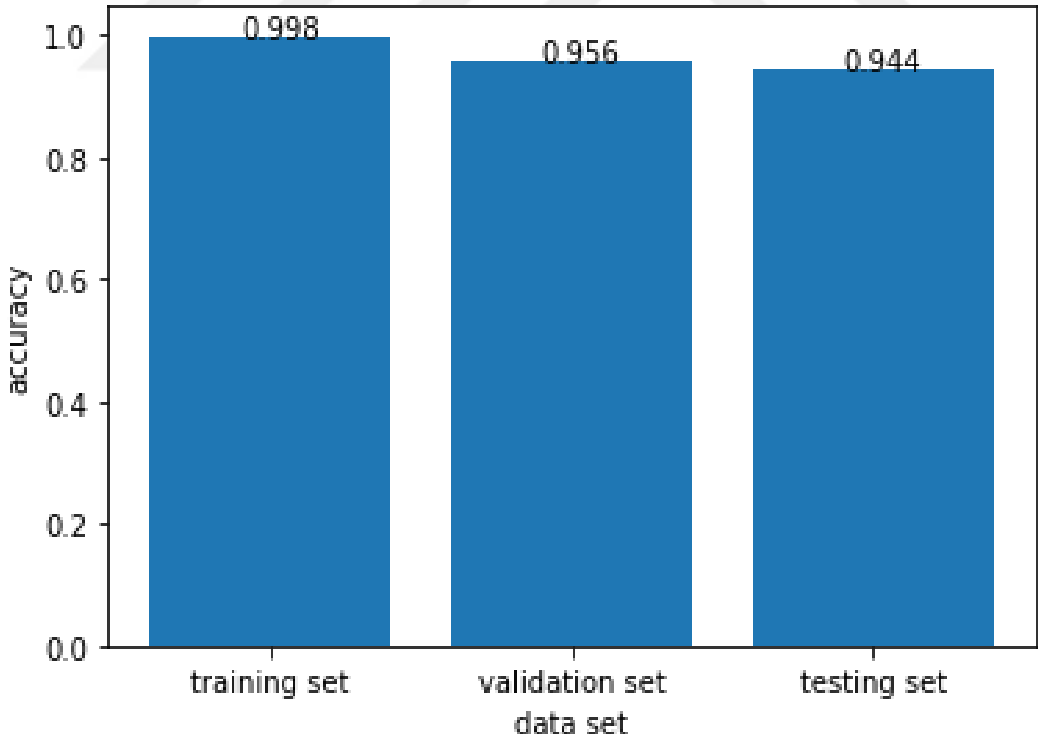
Şekil 4.4. LeNet Karışıklık Matrisi

Tablo 4.3. LeNet Mimarisinin Gradyan İniş Algoritması Uygulanmış Halinin Performans Değerlendirme Sonuçları

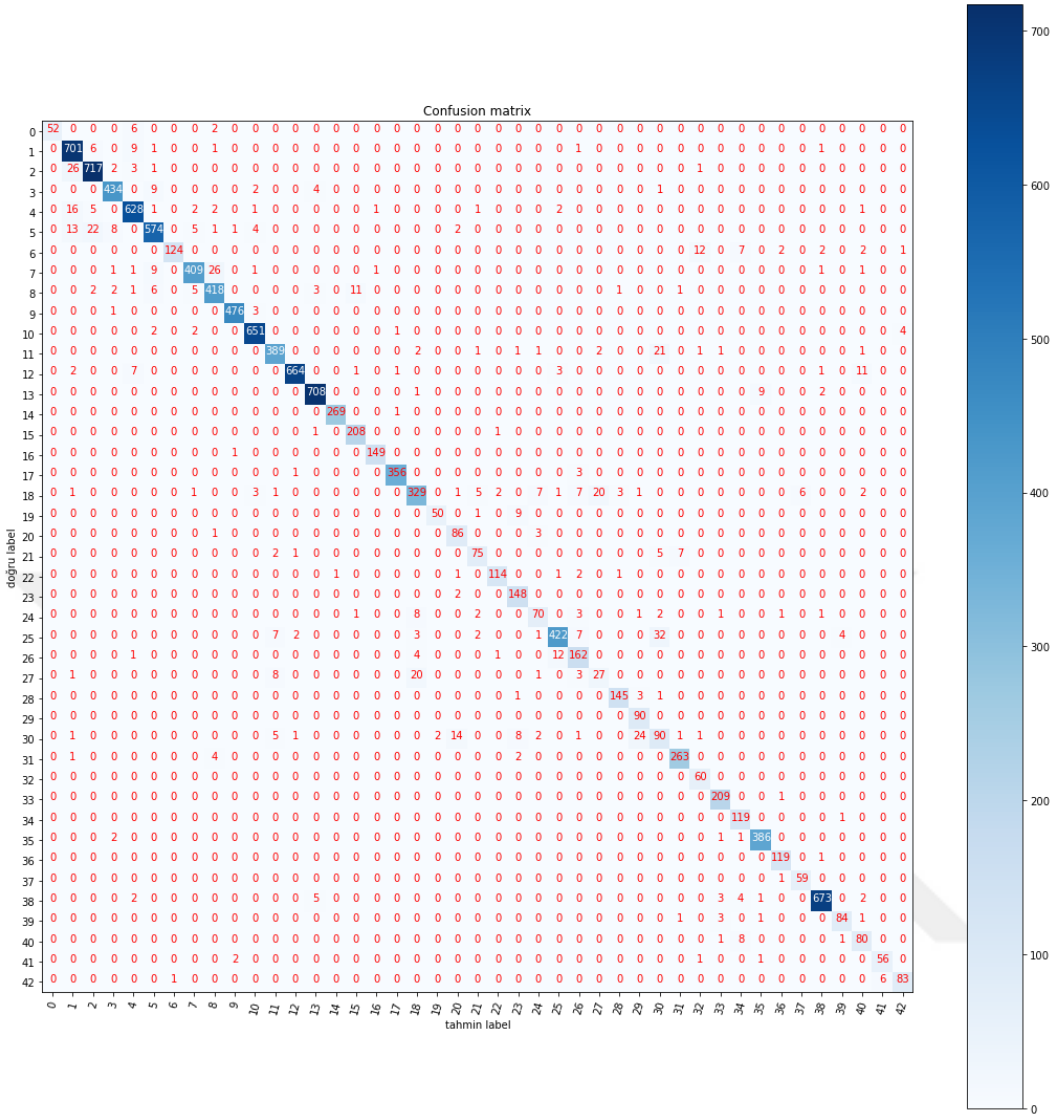
Mimari	**Algoritma**	**Performans ölçeği**	**Precision**	**Recall**	**F1-score**
LeNet	**Gradyan**	**Accuracy**			0.94
		Macro avg	0.91	0.92	0.92
		Weighted avg	0.95	0.94	0.94



Şekil 4.5. LeNet Gradyan İniş Algoritması ile Denenmiş Eğitim ve Doğrulama Kayıp Grafiği



Şekil 4.6. LeNet Performas Grafiği



Şekil 4.7. LeNet Karmaşıklık Matrisi

4.2. Alexnet

AlexNet için 3 aşamalı bir uygulama geliştirilmiştir. İlk aşamada veriler normalize edilerek eğitim sırasında kullanılacak hale dönüştürülmüştür. Görüntüler üç kanallı şekilde işlenmiş ve 'pickle' formatında kaydedilerek proje içerisine dâhil edilmiştir. İkinci aşamada eğitim gerçekleştirilmiştir. Üçüncü aşamada ise eğitim neticesinde elde edilen model çağırılarak sonuçların görselleştirilmesi ve performans kıyasları sunulmuştur (Tablo 4.4, Tablo 4.5, Şekil 4.8, Şekil 4.9, Şekil 4.10).

GTRSB'de trafik işaretleri tanımanın girdi boyutu ve çıktı boyutu AlexNet'in orijinal boyutundan farklı olan 32x32x3 ve 43 olduğundan, gereksinime uyacak bazı değişiklikler yapılmıştır.

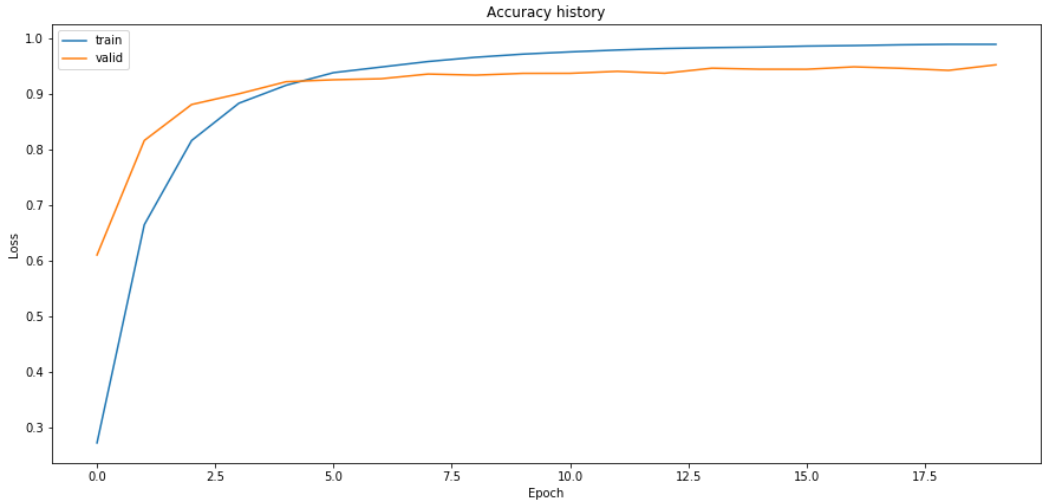
Adam optimizasyon algoritması için en yüksek başarım 5e-4, öğrenme katsayısı 150 iterasyon, 128 küme boyutu seçildiği durumda doğrulama veri setinde %96, test veri seti için %95.7 başarım elde edilmiştir.

Tablo 4.4. AlexNet Mimari Katmanları

AlexNet Mimarisi			
Katman	**Giriş**	**Açıklama**	**Çıktı**
Evrişim	32x32x3	Filtre: 5x5 Adım: 1x1 Dolgu: Valid	28x28x9
Havuzlama	28x28x9	Filtre: 2x2 Adım: 2x2 Dolgu: Valid	14x14x9
Evrişim	14x14x9	Filtre: 3x3 Adım: 1x1 Dolgu: Valid	12x12x32
Havuzlama	12x12x32	Filtre: 2x2 Adım: 2x2 Dolgu: Valid	6x6x32

Tablo 4.4.(Devam) AlexNet Mimari Katmanları

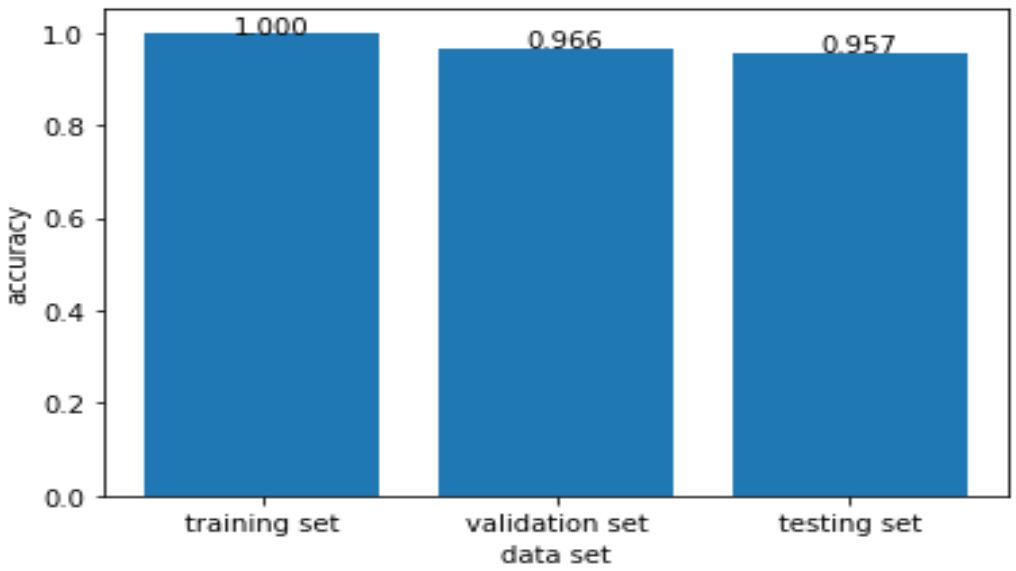
Evrişim	6x6x32	Filtre: 3x3 Adım: 1x1 Dolgu: Same	6x6x48
Evrişim	6x6x48	Filtre: 3x3 Adım: 1x1 Dolgu: Same	6x6x64
Evrişim	6x6x64	Filtre: 3x3 Adım: 1x1 Dolgu: Same	6x6x96
Havuzlama	6x6x96	Filtre: 2x2 Adım: 2x2 Dolgu: Valid	3x3x96
Düzleştirme	3x3x96		864
Tam Bağlantı	864	Her nöron mutlak bir katmana bağlıdır	400
Tam Bağlantı	400	Her nöron mutlak bir katmana bağlıdır	160
Tam Bağlantı	160	Her etiket için 43 olasılık tespit edilmiştir	43



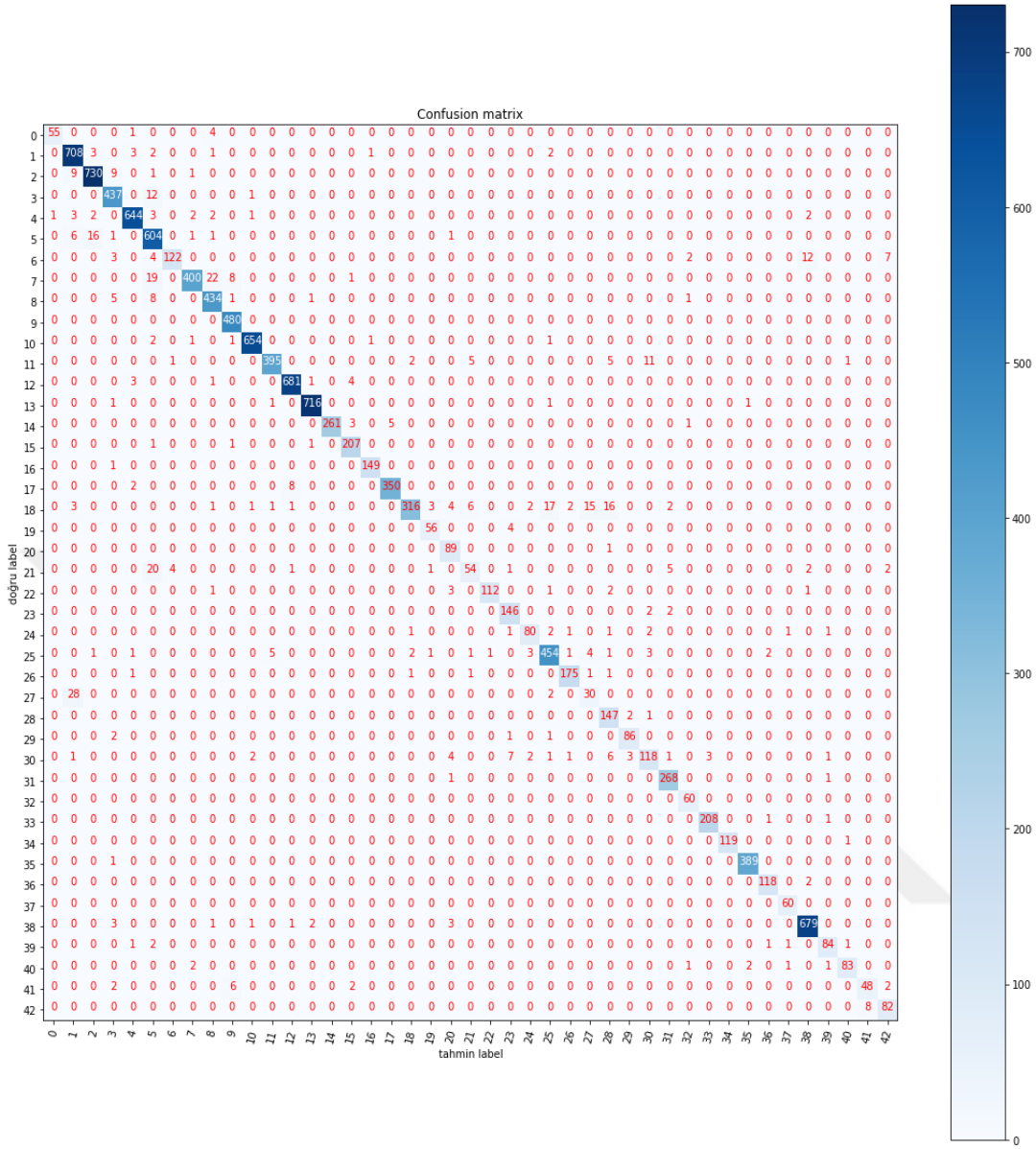
Şekil 4.8. AlexNet Zamana Göre Performans Grafiği

Tablo 4.5. AlexNet Performans Değerlendirme Ölçek Sonuçları

Mimari	**Algoritma**	**Performans ölçeği**	**Precision**	**Recall**	**F1-score**
AlexNet	**Adam**	**Accuracy**			0.96
		Macro avg	0.94	0.93	0.93
		Weighted avg	0.96	0.96	0.96



Şekil 4.9. AlexNet Performas Grafiği



Şekil 4.10. AlexNet Karmaşıklık Matrisi

4.3. VGGNET

VGGNet için 2 aşamalı bir uygulama geliştirilmiştir. İlk olarak 32x32 yeniden boyutlandırılmış resimlerden oluşan üç ".p" dosyamız mevcuttur. Bunlar "Train.p" eğitim seti, "Test.p" test seti, "Valid.p" doğrulama setidir. Bu verilerin içeriği aşağıda açıklanmıştır.

- "Features", trafik işareti görüntülerinin ham piksel verilerini içeren dört boyutlu bir dizidir (örnek numarası, genişlik, yükseklik, kanallar).

- "Labels", trafik işaretinin etiketini/sınıf kimliğini içeren 1 boyutlu bir dizidir.

- "Sizes", görüntünün orijinal genişliğini ve yüksekliğini temsil eden demetler (genişlik, yükseklik) içeren bir listedir.

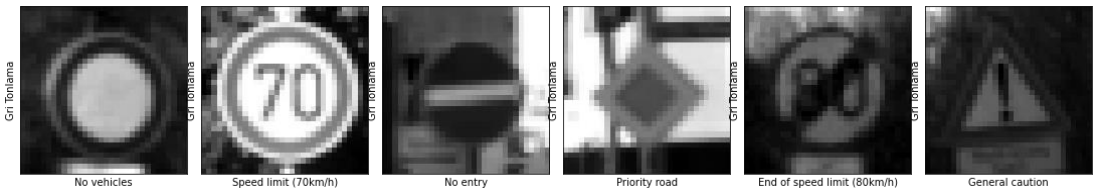
- "Coords", görüntüdeki işaretin etrafındaki sınırlayıcı bir kutunun koordinatlarını temsil eden demetler (x1, y1, x2, y2) içeren bir listedir.

- Signnames.csv, Csv formatında olan dosya, her bir sınıf için sınıf numarasının sınıf adı ile eşlemelerini içermektedir.

Sonrasında veriler yüklenmiş ve en iyi sonucu elde etmek için karıştırma (Shuffling), gri tonlama (Grayscaling), Yerel Histogram eşikleme (Local Histogram Equalization), normalizasyon (Normalization) ön işlemlerinden geçirilmiştir. Bu sayede veri eğitim için hazır hale getirilmiştir.

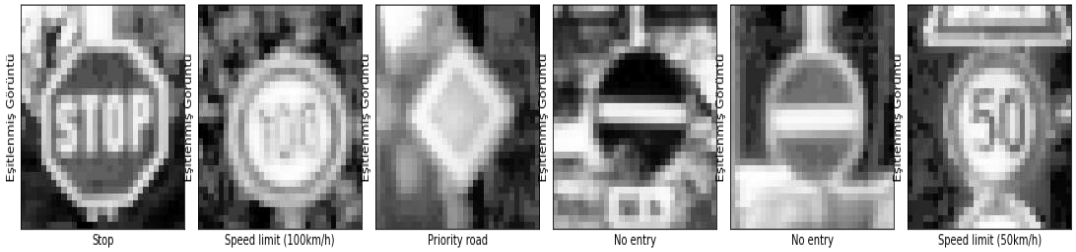
Karıştırma (Shuffling) işlemi modelin daha kararlı olması için eğitim veri setindeki rastgeleliği ve çeşitliliği artırmak için eğitim verileri üzerine uygulanır. Verilerimizi karıştırmak için sklearn kullanılmıştır.

VGGNet gibi yoğun işlem yükü olan algortimalarda 3 kanallı görüntüler yerine gri tonlamalı resimler kullanılması başarımı artırmaktadır. Bu işlem için Opencv Kütüphanesi kullanılmıştır.



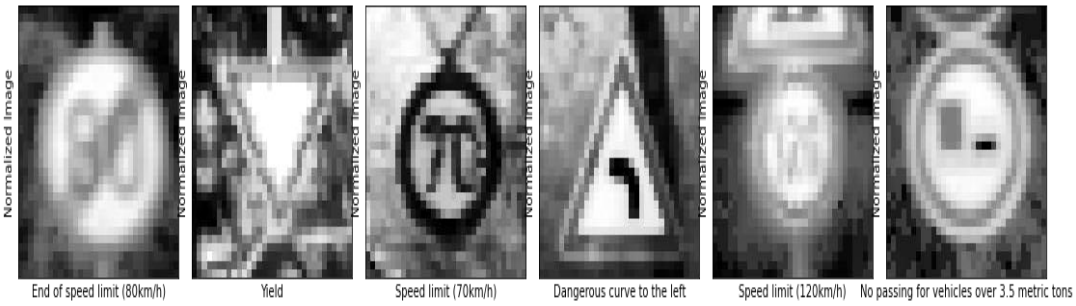
Şekil 4.11. Karıştırma Uygulanmış Veri Setinden Örnekler

Yerel Histogram Eşitleme işlemi görüntünün renkleri histogram haline getirildiği durumda dengeli bir dağılıma sahip olması, piksellerin parlaklık seviyeleri için yaklaşık olarak aynı sayıda pikselle sahip görüntüler elde etmek amaçlanır. Gerçek dünyadan elde edilmiş veri kümesi görüntülerinde düşük kontrast problemi sebebi ile bu işlem başarımın yükseltilmesine katkı sağlamıştır.



Şekil 4.12. Yerel Histogram Eşitleme Uygulanmış Görüntü Örnekleri

Normalleştirme piksel yoğunluk aralığını değiştirerek belli bir aralığa toplayan işlemdir. Kümülatif histogramın değerleri, görüntünün çekilmek istendiği en büyük değere çarpılmaktadır ve piksel sayısı ile bölünmektedir.



Şekil 4.13. Normalleştirme Yapılmış Görüntü Örnekleri

Veriler üzerinde incelemeler yapabilmek için görselleştirmeler hazırlanmıştır. Gerekli ön bilgi edinildikten sonra model eğitilmiştir. İkinci aşamada ise eğitim neticesinde elde edilen model çağırılarak sonuçların görselleştirilmesi ve performans kıyasları sunulmuştur (Tablo 4.7, Şekil 4.14, Şekil 4.15, Şekil 4.16).

VGGNet mimarilerinde katman sayıları farklılık gösterebilmektedir. Orijinal makalede 16-19 katman mevcuttur. Ancak kaynak kullanımından tasarruf edebilmek maksadı ile bu eğitimde 12 katmanlı bir mimari hazırlanmıştır (Tablo 4.6).

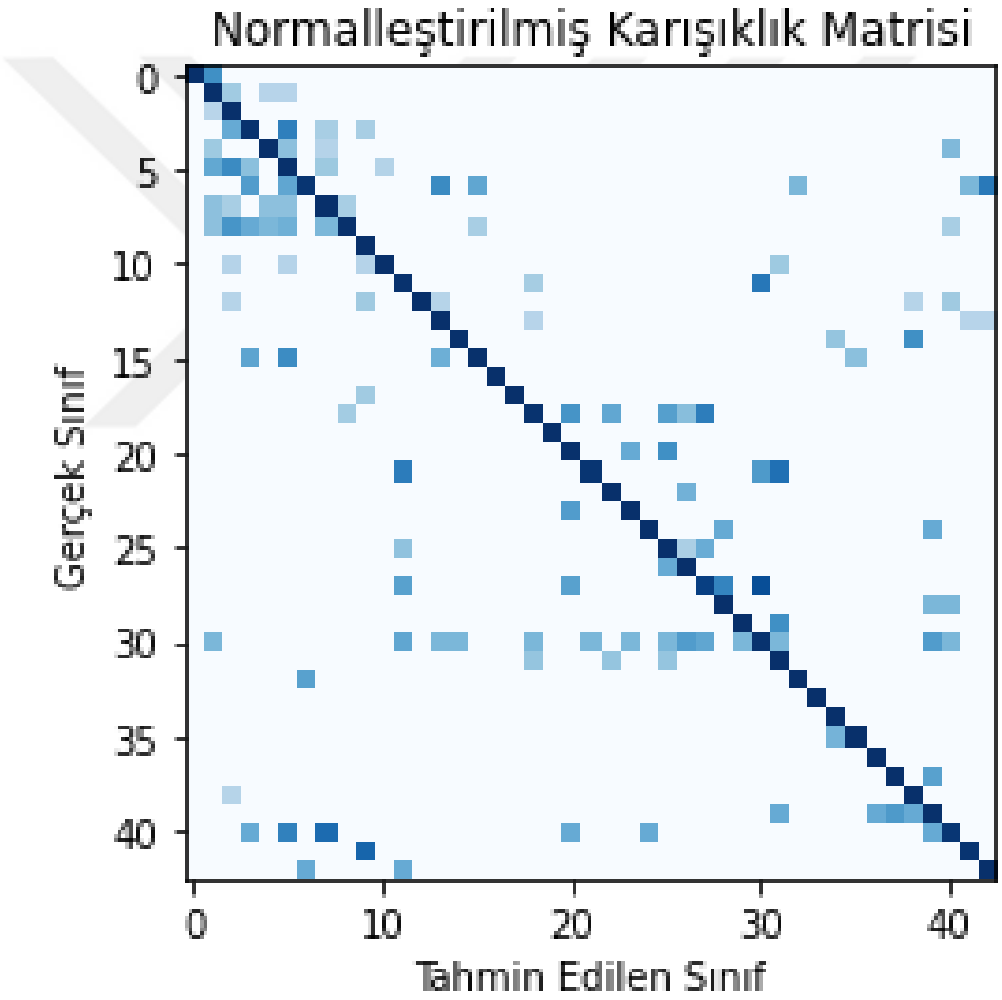
Derin yapay sinir ağları için aşırı uyum probleminin önüne geçmek için özellikle paremetre sayısının çok arttığı bazı katmanlara unutma (Dropout) işlemi gerçekleştirilmiştir. Belli paremetreler kullanılarak eğitim sırasında bazı bağlantılar tamamen hesaplamadan çıkarılmıştır (Srivastava ve diğ., 2014).

Tablo 4.6. VGGNet Mimari Katmanları

VGGNet Mimarsisi			
Katman	**Giriş**	**Açıklama**	**Çıkış**
Evrişim	32x32x1	Filtre: 3x3 Adım: 1x1 Dolgu: Same	32x32x32
Evrişim	32x32x1	Filtre: 3x3 Adım: 1x1 Dolgu: Same	32x32x32
Havuzlama	32x32x32	Filtre: 2x2 Adım: 2x2 Dolgu: Valid	16x16x32
Evrişim	16x16x32	Filtre: 3x3 Adım: 1x1 Dolgu: Same	16x16x64
Evrişim	16x16x64	Filtre: 3x3 Adım: 1x1 Dolgu: Same	16x16x64
Havuzlama	16x16x64	Filtre: 2x2 Adım: 2x2 Dolgu: Valid	8x8x64
Evrişim	8x8x64	Filtre: 3x3 Adım: 1x1 Dolgu: Same	8x8x128
Evrişim	8x8x128	Filtre: 3x3 Adım: 1x1 Dolgu: Same	8x8x128
Havuzlama	8x8x128	Filtre: 2x2 Adım: 2x2 Dolgu: Valid	4x4x128

Tablo 4.6. (Devam) VGGNet Mimari Katmanları

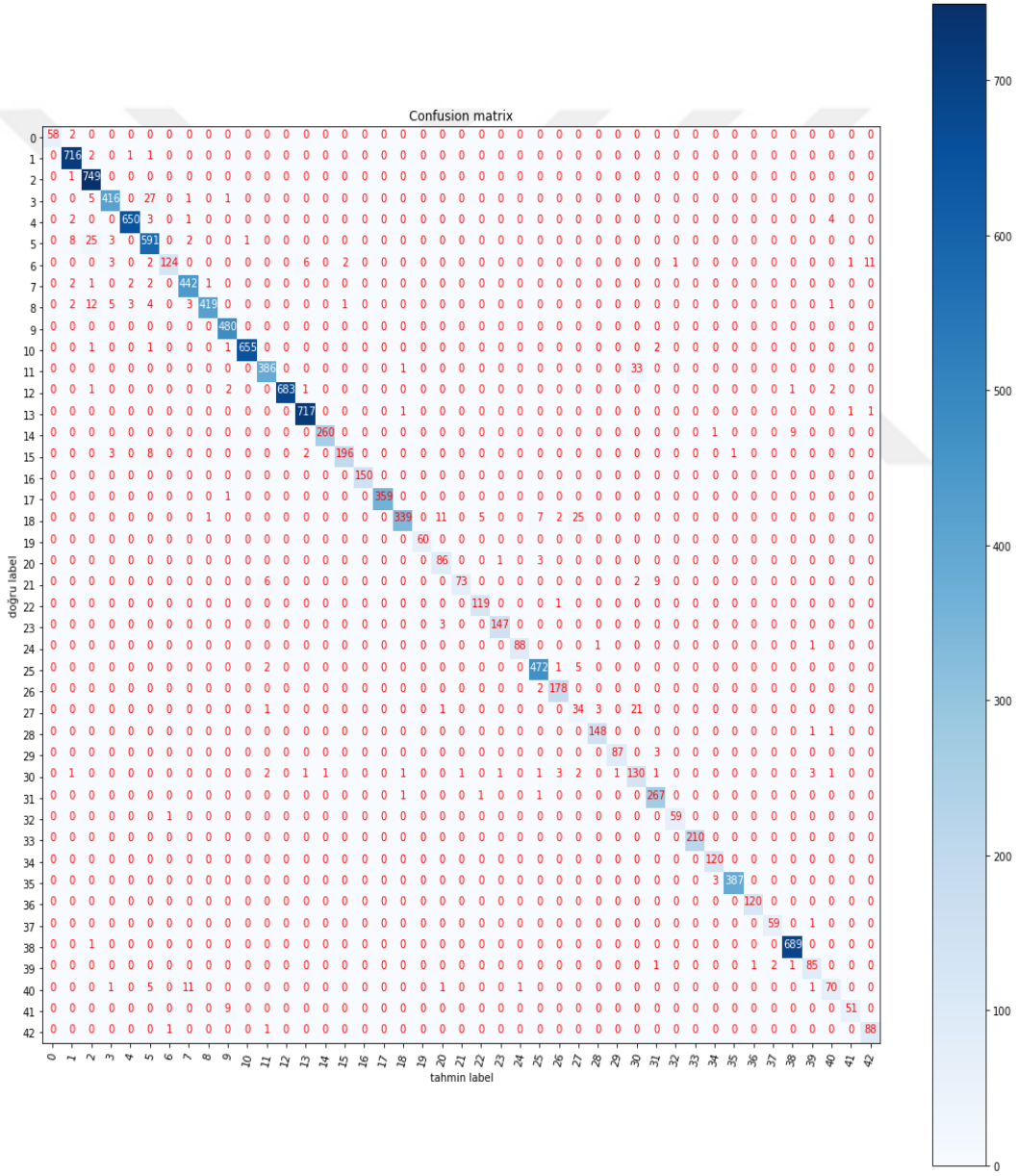
Düzleştirme	4x4x128		2048
Tam Bağlantı	2048	Her nöron mutlak bir katmana bağlıdır.	128
Tam Bağlantı	128	Her nöron mutlak bir katmana bağlıdır.	128
Tam Bağlantı	128	Her etiket için 43 olasılık tespit edilmiştir.	43



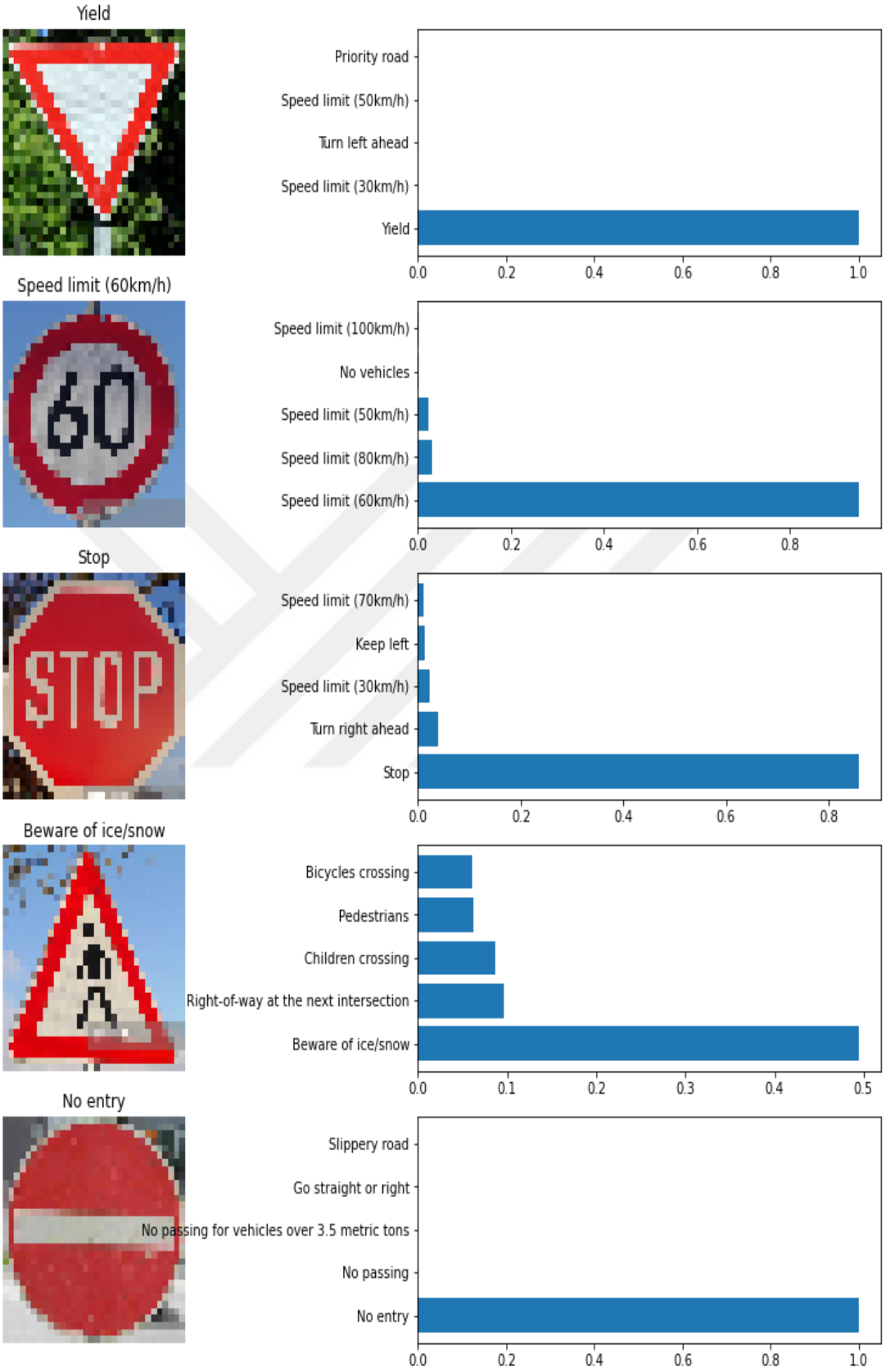
Şekil 4.14. Normalleştirilmiş Karmaşıklık Matrisi

Tablo 4.7. VGGNet Performans Değerlendirme Ölçek Sonuçları

Mimari	**Algoritma**	**Performans ölçeği**	**Precision**	**Recall**	**F1-score**
VGGNet	**Adam**	**Accuracy**			0.97
		Macro avg	0.95	0.95	0.95
		Weighted avg	0.97	0.97	0.97



Şekil 4.15. VGGNet Karmaşıklık Matrisi

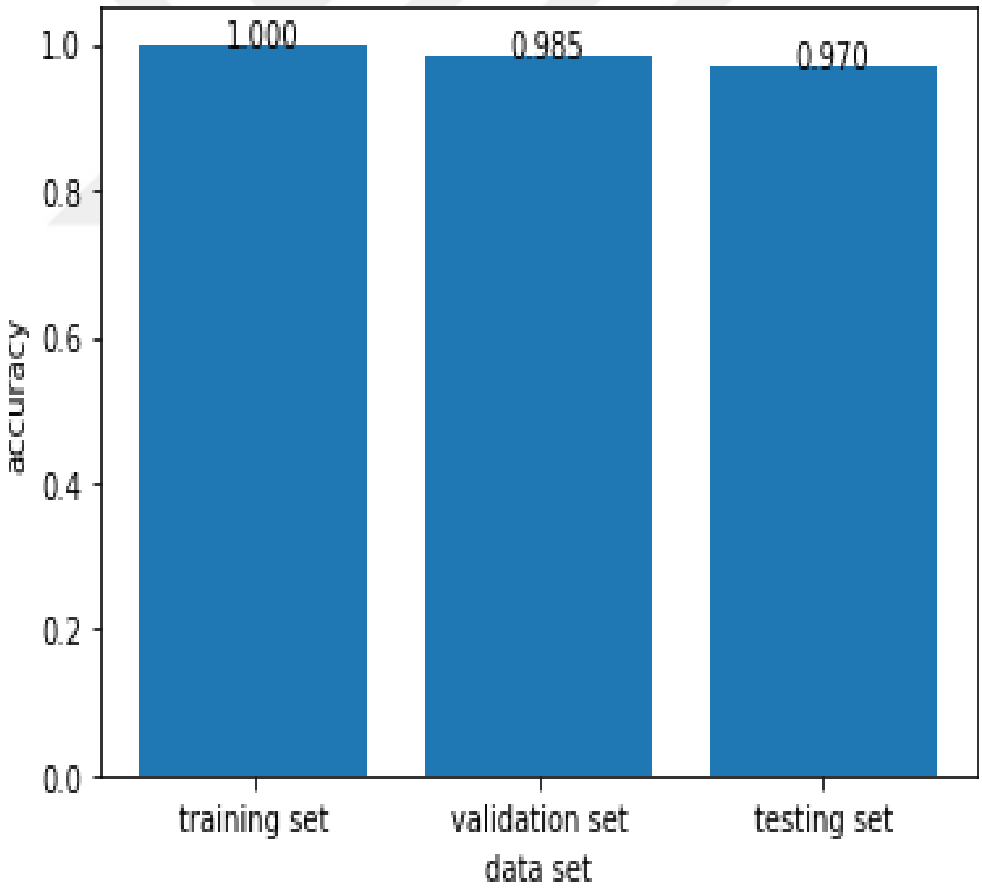


Şekil 4.16. VGGNet Softmax Sonucu

4.4. GoogleNet

GoogleNet için; 3 aşamalı bir uygulama geliştirilmiştir. İlk aşamada veriler ön işlemlerden (normalizasyon) geçirilerek proje içerisinde kullanılacak hale dönüştürülmüştür. Görüntüler üç kanallı şekilde işlenmiş ve 'picle' formatında kaydedilerek proje içerisine dâhil edilmiştir. İkinci aşamada eğitim gerçekleştirilmiştir. Üçüncü aşamada ise eğitim neticesinde elde edilen model çağırılarak sonuçların görselleştirilmesi ve performans kıyasları sunulmuştur(Tablo 4.8., Şekil 4.17., Şekil 4.18.).

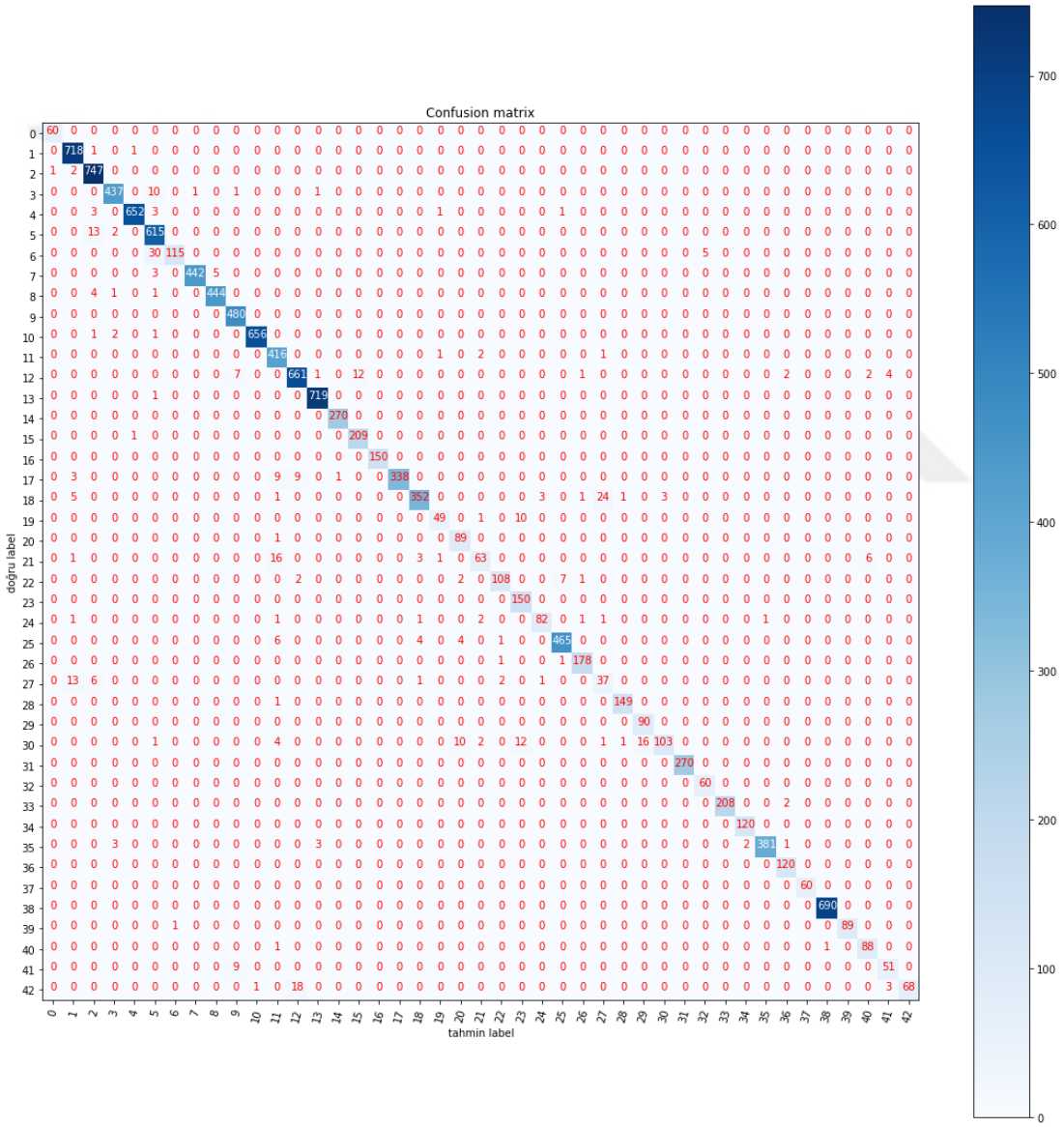
Adam optimizasyon algoritması için en yüksek başarım 4e-4 öğrenme katsayısı, 40 iterasyon, 128 küme boyutu seçildiği durumda doğrulama veri setinde %98.5, test veri seti için %97 başarım elde edilmiştir.



Şekil 4.17. GoogleNet Performans Grafiği

Tablo 4.8. GoogleNet Performans Değerlendirme Ölçek Sonuçları

Mimari	Algoritma	Performans ölçeği	Precision	Recall	F1-score
Googlenet	**Adam**	**Accuracy**			0.97
		Macro avg	0.95	0.94	0.95
		Weighted avg	0.97	0.97	0.97



Şekil 4.18. GoogleNet Karmaşıklık Matrisi

5. TARTIŞMA VE ÖNERİLER

Bu çalışmada derin öğrenme metotlarından biri olan ConNN mimarileri incelenerek yeni bir mimari önerilmiştir. Farklı paremetreler ve bu sonuçların kıyasları Tablo 3.1., Tablo 4.2., Tablo 4.3., Tablo 4.5., Tablo 4.7., Tablo 4.8.'de detaylı biçimde anlatılmıştır. Önerilmiş olan model zaman performansı olarak daha önce sunulmuş modeller ile kıyaslanmış ve buna ait sonuçlar Tablo 5.1. ve Tablo 5.2.'de sunulmuştur. Gerçek zamanlı çalışmaya uygun bir model elde edilmiştir. Yüksek başarımlı olduğu gözlenen bazı çalışmalar incelendiğinde yapılan çalışmanın veri kümesinin şekil yahut renk gibi kısıtlar ile kapsamın daraltıldığı görülmüştür. Önerilen modelde özellik çıkarımı sistem tarafından yapıldığından başarım oranını yükseltmek için bu tarz kısıtlara gerek duyulmamıştır. Bu durum modelin sürücü destek sistemleri için uygun olduğunu göstermektedir. Önerilen metotta eğitim süresi, katman yapısı ve görüntü tanıma işlemleri için kullanılan fonksiyon sayesinde kısaltılmıştır. Önerilen metotta görüntü üzerinde keras kütüphanesinden faydalanılarak görüntü en uygun tanıma konumuna getirilir ve tanıma işlemine tabi tutulur. Bu işlem görüntünün hem daha doğru hem daha kısa sürede tanınmasına olanak sunmuştur.

Tablo 5.1. Modellerin Görüntü Tanıma Zamansal Ve Donanımsal Kıyasları

Kaynak	**Ortalama Görüntü Tanıma Süresi**	**Donanım**
(Yuan ve diğ., 2015)	0.341 s	Intel Core i7 QUAD 3.70 GHz CPU ve GPU
(Xu ve diğ., 2019)	0.930 s	Belirtilmemiş
(Arcos-García ve diğ., 2018)	0.442 s	Intel Core i7 4770 CPU, 16 GB RAM ve NVIDIA Titan Xp 3840 CUDA çekirdeği ve 12 GB RAM içeren GPU
(Torres ve diğ., 2019)	Belirtilmemiş	2 adet intel Xeon E5606 2.13 GHz CPU ve RAM boyutları 24 GB ve 31 GB olan 2 adet NVIDIA TITAN Xp GPU
(Çetinkaya & Acarman, 2020)	0.416 s	Intel i5 8250U 1.60 GHz CPU, 16 GB RAM ve NVIDIA GeForce 940MX 384 CUDA çekirdeği ve 2 GB RAM içeren GPU

Tablo 5.1.(Devam) Modellerin Görüntü Tanıma Zamansal Ve Donanımsal Kıyasları

(Özdamar, 2014)	Belirtilmemiş	Belirtilmemiş
(Sichkar & Kolyubin, 2020)	Belirtilmemiş	GPU Tesla V100 Ve 16 Gb RAM
Önerilen Yöntem	0.01 s	Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz 3.60 GHz , x64 tabanlı işlemci, GPU Intel® HD Grafikleri 630

Tablo 5.2. Önerilen Modelin Diğer Çalışmalarla Başarım Kıyası

Kaynak	**Başarı**
(Yuan ve diğ., 2015)	%88.73
(Xu ve diğ., 2019)	%94.61
(Arcos-García ve diğ., 2018)	%91.59
(Torres ve diğ., 2019)	%93
(Çetinkaya & Acarman, 2020)	%92.74
(Özdamar, 2014)	%98,38
(Sichkar & Kolyubin, 2020)	%97,22
Önerilen Yöntem	%98,83

Yapılan çalışma neticesinde yüksek başarımlı bir sistem geliştirilmiştir. Bu çalışmada trafik işaretlerinin tanınması üzerinde durulmuştur. İleride yapılacak çalışmalarda tanımanın yanında tespit üzerine çalışılarak gerçek zamanlı çalışan sürücü destek sistemleri geliştirilebilir.

KAYNAKLAR

Aksoy, B., Halis, H., Salman, O. (2020). Elma Bitkisindeki Hastalıkların Yapay Zekâ Yöntemleri İle Tespiti Ve Yapay Zekâ Yöntemlerinin Performanslarının Karşılaştırılması. *International Journal Of Engineering And Innovative Research*, *2*(3), 194–210.

Alpaydin, E. (2020). *Introduction To Machine Learning* (Fourth ed.). London :The Mit Press.

Arcos-García, Á., Álvarez-García, J. A., Soria-Morillo, L. M. (2018). Evaluation Of Deep Neural Networks For Traffic Sign Detection Systems. *Neurocomputing*, *316*, 332–344.

Aylak, B. L., Oral, O., Yazici, K. (2021). Yapay Zeka Ve Makine Öğrenmesi Tekniklerinin Lojistik Sektöründe Kullanımı. *El-Cezeri*, *8*(1), 74–93.

Becer, H. C. (2011). *A robust traffic sign recognition system* Yüksek Lisans Tezi, Orta Doğu Teknik Üniveristesi, Fen Bilimleri Enstitüsü, Ankara, 268127.

Bouton, C. E., Shaikhouni, A., Annetta, N. V., Bockbrader, M. A., Friedenberg, D. A., Nielson, D. M., Sharma, G., Sederberg, P. B., Glenn, B. C., Mysiw, W. J., Morgan, A. G., Deogaonkar, M., Rezai, A. R. (2016). Restoring Cortical Control Of Functional Movement In A Human With Quadriplegia. *Nature 2016 533:7602*, 247–250.

Çakıcı, Z., Şazi, M. (2017). Trafik İşaretlerinin Bilinirliği Üzerine Bir Araştırma: Denizli Örneği. *Bitlis Eren Üniversitesi Fen Bilimleri Dergisi*, *6*(1), 21–30.

Çetinkaya, M., Acarman, T. (2020). Trafik Işaret Levhası Tespiti Için Derin Öğrenme Yöntemi. *Akıllı Ulaşım Sistemleri Ve Uygulama Dergisi*, *3*(2), 140–157.

Cireşan, D., Meier, U., Masci, J., Schmidhuber, J. (2012). Multi-Column Deep Neural Network For Traffic Sign Classification. *Neural Networks : The Official Journal Of The International Neural Network Society*, *32*, 333–338.

De La Escalera, A., Moreno, L. E., Salichs, M. A., María, J., Armingol, M. (1997). Road Traffic Sign Detection And Classification. *Ieee Transactions On Industrial Electronics*, *44*(6).

Fethi, M. D., Pasiouras, F. (2010). Assessing Bank Efficiency And Performance With Operational Research And Artificial Intelligence Techniques: A Survey. *European Journal Of Operational Research*, *204*(2), 189–198.

Gilbert, F. J., Astley, S. M., Gillan, M. G. C., Agbaje, O. F., Wallis, M. G., James, J., Boggis, C. R. M., Duffy, S. W. (2008). Single Reading With Computer-Aided Detection For Screening Mammography. *New England Journal Of Medicine*, *359*(16).

Gonzalez, R. C., Woods, R. E. (2008). *Digital Image Processing*. 954.

Hamet, P., Tremblay, J. (2017). Artificial Intelligence In Medicine. *Metabolism*, *69*, (36-40).

Hatzidimos, J. (2004). Automatic traffic sign recognition in digital images. In *Proceedings of the International Conference on Theory and Applications of Mathematics and Informatics-Ictami*, Thessaloniki: Greece, (174-184).

Hinton, G. E., Osindero, S. (2006). *A Fast Learning Algorithm For Deep Belief Nets.* Neural Computation, 18, 1527-1554.

Hosny, A., Parmar, C., Quackenbush, J., Schwartz, L. H., Aerts, H. (2018). Artificial intelligence in radiology. *Nature reviews. Cancer*, 18(8), 500–510. https://doi.org/10.1038/s41568-018-0016-5

İnik, Ö., Ülker, E. (2017). Derin Öğrenme Ve Görüntü Analizinde Kullanılan Derin Öğrenme Modelleri. *Gaziosmanpaşa Bilimsel Araştirma Dergisi (Gbad)*, *6*(3), 85–104.

Jiang, F., Jiang, Y., Zhi, H., Dong, Y., Li, H., Ma, S., Wang, Y., Dong, Q., Shen, H., & Wang, Y. (2017). Artificial Intelligence In Healthcare: Past, Present And Future. *Stroke And Vascular Neurology*, *2*(4), 230–243.

Johnson, J. (2019). Artificial Intelligence & Future Warfare: Implications For International Security. *35*(2), 147–169.

Kızrak, A. (2021). *Yapay Sinir Ağları*. From Https://Ayyucekizrak.Medium.Com, (Ziyaret Tarihi: 19 Aralık, 2021)

Lecun, Y., Bengio, Y., Hinton, G. (2015). Deep Learning. *Nature*, *521*(7553), 436–444.

Lecun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., Jackel, L. D. (1989). Backpropagation Applied To Handwritten Zip Code Recognition. *Neural Computation*, *1*(4), 541–551.

Lecun, Y., Bottou, L., Bengio, Y., Haffner, P. (1998). Gradient-Based Learning Applied To Document Recognition. *Proceedings Of The Ieee*, *86*(11), 2278–2324.

Luckow, A., Kennedy, K., Ziolkowski, M., Djerekarov, E., Cook, M., Duffy, E., Schleiss, M., Vorster, B., Weill, E., Kulshrestha, A., Smith, M. C. (2018). Artificial Intelligence And Deep Learning Applications For Automotive Manufacturing. *2018 Ieee International Conference On Big Data (Big Data)*, 3144–3152.

Malik, Z., Siddiqi, I. (2014). Detection And Recognition Of Traffic Signs From Road Scene Images. *Undefined*, 330–335.

Mcculloch, W. S., Pitts, W. (1943). A Logical Calculus Of The Ideas Immanent In Nervous Activity. *The Bulletin Of Mathematical Biophysics 1943 5:4*, *5*(4), 115–133.

Minsky, M., Papert, S. (1969). *Perceptrons; An Introduction To Computational Geometry*. 258.

Mullainathan, S., Spiess, J. (2017). Machine Learning: An Applied Econometric Approach. *Journal Of Economic Perspectives*, *31*(2), 87–106.

Obermeyer, Z., Emanuel, E. J. (2016). Predicting The Future — Big Data, Machine Learning, And Clinical Medicine. *The New England Journal Of Medicine*, *375*(13), 1216.

Özdamar, M. (2014). *Altuzay yöntemleri ile trafik işareti tanıma*. Yüksek Lisans Tezi, Eskişehir Osmangazi Üniversitesi, Fen Bilimleri Enstitüsü, Eskişehir, 367553.

Özkan, İ. (2010). *Traffic sign detection using fpga*. Yüksek Lisans Tezi, Orta Doğu Teknik Üniveristesi, Fen Bilimleri Enstitüsü, Ankara, 268431.

Öztürk, G. (2020). *Derin evrişimsel sinir ağlarını kullanılarak araç, insan ve trafik işaretlerinin tanınması*. Yüksek Lisans Tezi, Sakarya Uygulamalı Bilimler Üniversitesi, Sakarya, 637493.

Paclík, P., Novovičová, J., Pudil, P., Somol, P. (2000). Road Sign Classification Using Laplace Kernel Classifier. *Pattern Recognition Letters*, *21*(13–14), 1165–1173.

Roper, Y., Rowland, M., Chakich, Z., Mcgill, W., Nanayakkara, V., Young, D., Whale, R. (2018). *Implications Of Traffic Sign Recognition (Tsr) Systems For Road Operators*.

Rosenblatt, F. (1961). *Principles Of Neurodynamics Perceptrons And The Theory Of Brain Mechanisms*. DOI: 10.1007/978-3-642-70911-1_20

Rumelhart, D. E., Hinton, G. E., Williams, R. J. (1986). Learning Representations By Back-Propagating Errors. *Undefined*, *323*(6088), 533–536.

Russell, S. J., Norvig, P. (2010). *Artificial Intelligence A Modern Approach*.

Sabesan, R., Schmidt, B. P., Tuten, W. S., & Roorda, A. (2016). The Elementary Representation Of Spatial And Color Vision In The Human Retina. *Science Advances*, *2*(9).

Samuel, A. L. (1967). Some Studies In Machine Learning Using The Game Of Checkers. Ii-Recent Progress. *Ibm Journal*, 601–617.

Sichkar, V. N., Kolyubin, S. A. (2020). Real Time Detection And Classification Of Traffic Signs Based On Yolo Version 3 Algorithm. *Scientific And Technical Journal Of Information Technologies, Mechanics And Optics*, *20*(3), 418–424.

Sindhuja R. (2018). Artificial Intelligence And Its Applications In Various Fields. *International Journal For Research Trends And Innovation (Www.Ijrti.Org)*, *3*, 137.

Sinoplu, M., Yilmaz, Ö., Gökkaya, G., Durak, H. (2021). Siber Güvenlik Ile İlgili Nesnelerin İnterneti Ve Yapay Zekâ Konularını Temel Alan Tezlerin Yöntemsel Olarak İncelenmesi. *Journal Of Information And Communication Technologies*, *3*(2), 228–242.

Srivastava, N., Hinton, G., Krizhevsky, A., Salakhutdinov, R. (2014). Dropout: A Simple Way To Prevent Neural Networks From Overfitting. *Journal Of Machine Learning Research*, *15*, 1929–1958.

Tiryaki, B. (2019). *Trafik İşaretlerinin Derin Sinir Ağları İle Sınıflandırılması*. Yüksek Lisans Tezi, Atatürk Üniversitesi, Fen Bilimleri Enstitüsü, Erzurum, 584527.

Torres, L. T., Paixão, T. M., Berriel, R. F., De Souza, A. F., Badue, C., Sebe, N., Oliveira-Santos, T. (2019). Effortless Deep Training For Traffic Sign Detection Using Templates And Arbitrary Natural Images. *Proceedings Of The International Joint Conference On Neural Networks*. DOI: 10.1109/IJCNN.2019.8852086

Tüik. (2020). Karayolu Trafik Kaza İstatistikleri, . *Türkiye İstatistik Kurumu Matbaası*.

Tüik. (2021). *Karayolu Trafik Kaza İstatistikleri*. Https://Data.Tuik.Gov.Tr/Bulten/Index?P=Karayolu-Trafik-Kaza-Istatistikleri-2020-37436, (Ziyaret Tarihi:20 Şubat 2022)

Ulay, E. (2008). *Color And Shape Based Traffic Sign Detection*. Orta Doğu Teknik Üniveristesi, Fen Bilimleri Enstitüsü, Ankara, 238597.

Warwick, K., Shah, H. (2016). Can Machines Think? A Report On Turing Test Experiments At The Royal Society. *Journal Of Experimental & Theoretical Artificial Intelligence*, *28*(6), 989–1007.

Xu, X., Jin, J., Zhang, S., Zhang, L., Pu, S., Chen, Z. (2019). Smart Data Driven Traffic Sign Detection Method Based On Adaptive Color Threshold And Shape Symmetry. *Future Generation Computer Systems*, *94*, 381–391.

Yann Le Cun. (1988). *A Theoretical Framework For Back-Propagation*. 21–28.

Yuan, X., Guo, J., Hao, X., Chen, H. (2015). Traffic Sign Detection Via Graph-Based Ranking And Segmentation Algorithms. *Ieee Transactions On Systems, Man, And Cybernetics: Systems*, *45*(12), 1509–1521.

Zaklouta, F., Stanciulescu, B. (2014). Real-Time Traffic Sign Recognition In Three Stages. *Robotics And Autonomous Systems*, *62*(1), 16–24.

Zam, M. (2019). *Trafik İşaretlerini Tanıyan Bir Sürücü Güvenlik Sistemi*. Yüksek Lisans Tezi, Bursa Uludağ Üniversitesi, Fen Bilimleri Enstitüsü, Bursa, 617269.

EKLER

Ek-A Önerilen Model.ipynb

ÖZGEÇMİŞ

İlkokulu ve Ortaokulu Noter Cevdet Altun İlköğretim okulunda tamamladı. Mardin İmkb Anadolu Öğretmen Lisesini tamamladı. Kocaeli Üniversitesi Bilgisayar Mühendisliği bölümünü 2015 yılında tamamladı. 2017 yılında çalışmaya başlamış olduğu Kocaeli Büyük şehir Belediyesi UlaşımPark şirketinde halen Bilgisayar Mühendisi olarak çalışmaktadır.