T.C.
İSTANBUL AYDIN ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

EVRİŞİMLİ SİNİR AĞLARI İLE PLAKA TANIMADA ALGORİTMALARIN KARŞILAŞTIRILMASI

YÜKSEK LİSANS TEZİ

İsa JAVADOV

Bilgisayar Mühendisliği Ana Bilim Dalı
Bilgisayar Mühendisliği Programı

AĞUSTOS, 2020

T.C.
İSTANBUL AYDIN ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

EVRİŞİMLİ SİNİR AĞLARI İLE PLAKA TANIMADA ALGORİTMALARIN KARŞILAŞTIRILMASI

YÜKSEK LİSANS TEZİ

İsa JAVADOV
(Y1713.010061)

Bilgisayar Mühendisliği Ana Bilim Dalı
Bilgisayar Mühendisliği Programı

Tez Danışmanı: Dr. Öğr. Üyesi Ahmet GÜRHANLI

AĞUSTOS, 2020

YEMİN METNİ

Yüksek Lisans tezi olarak sunduğum "Yapay Sinir Ağları ile Plaka Tanımada Algoritmaların Karşılaştırılması" adlı çalışmanın, tezin proje safhasından sonuçlanmasına kadarki bütün süreçlerde bilimsel ahlak ve geleneklere aykırı düşecek bir yardıma başvurulmaksızın yazıldığını ve yararlandığım eserlerin Bibliyografya'da gösterilenlerden oluştuğunu, bunlara atıf yapılarak yararlanılmış olduğunu belirtir ve onurumla beyan ederim. (10/08/2020)

İsa JAVADOV

Canım Annem'e

ÖNSÖZ

Nüfusun artması ve insanların daha kalabalık şehirlerde yaşamaya başlaması ile birlikte araç sayısı artmıştır. Araç sayısının artması ise araçların takip edilmesini zorlaştırmıştır. Bu çalışmada, hareketli veya durağan araçların plaka bölgesi bulunup, plaka tanıma işlemi yapılmaktadır. Plaka Tanıma Sistemi hareketli ya da sabit araçları izlemek ve bu araçların güvenlik kontrollerini sağlamak için kolluk kuvvetleri tarafından yaygın olarak kullanılabilmektedir. Ayrıca araç ücret toplama sistemi gibi yaşamın içindeki birçok yerde de rahatlıkla kullanılabilmektedir.

Bu tez çalışmam esnasında büyük bir sabır ve özveri ile bana destek olan tez danışman hocam Dr. Öğr. Üyesi Ahmet GÜRHANLI'ya teşekkür ederim. Ayrıca bu zorlu süreçte, tezimin her aşamasında bana destek olan, beni yalnız bırakmayan aileme de sonsuz sevgi ve saygılarımı sunarım.

Ağustos 2020 **İsa JAVADOV**

İÇİNDEKİLER

Sayfa

KISALTMALAR

Adadelta	: Adaptive Delta
Adagrad	: Adaptive Gradient Algorithm
Adam	: Adaptive Moment Estimation
Adamax	: Adaptive Maximum
ESA	: Evrişimli Sinir Ağları (Convolutional Neural Networks)
Leaky ReLu	: Leaky Rectified Linear Unit
Nadam	: Nesterov Momentum into Adam
PTS	: Plaka Tanıma Sistemi
ReLu	: Rectified Linear Unit
Rmsprop	: Root Mean Square Propagation
SGD	: Stochastic Gradient Descent
Tanh	: Hyperbolic Tangent
YSA	: Yapay Sinir Ağları

ÇİZELGE LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

EVRİŞİMLİ SİNİR AĞLARI İLE PLAKA TANIMADA ALGORİTMALARIN KARŞILAŞTIRILMASI

ÖZET

Gelişen teknoloji ve akıllı şehirlerin artmasıyla ülkelerdeki araç sayıları da artmıştır. Artan araç sayısı ile birlikte trafik kontrolleri, güvenlik kontrolleri gibi ihtiyaçlar daha da artmış ve herbir aracı takip etmek zorlaşmıştır. Bu nedenle Plaka Tanıma alanında bir geliştirmeye ihtiyaç duyulmuştur. Bu çalışmada plaka tanıma işlemi 3 aşamada yapılır. Bu aşamalar plaka bölgesinin bulunması, karakter ayrıştırma ve karakter tanıma yöntemleridir. Bu aşamalardan önce birtakım işlemler ile görüntünün daha temiz olması sağlanır. Akan trafikte görüntünün yakalanması ve algılanması ışık, hız gibi faktörlerden dolayı oldukça zordur. Bu çalışma plaka tanıma işlemi için önce araçları tespit eder ardından belirsiz görüntülerde karakter tanıma yapmak için Evrişimli Sinir Ağı uygular ve 7 optimizasyon algoritmasını karşılaştırır. Sonuçlar, eski plaka tanıma yöntemlerine kıyasla hem doğruluğun hem de hızın daha üstün olduğunu göstermiştir.

Anahtar Kelimeler : *Karakter Tanıma, Evrişimli Sinir Ağı, Akan Trafik, Plaka Tanıma*

COMPARISION OF ALGORITHMS IN LICENSE PLATAE RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK

ABSTRACT

With the development of technology and increasing smart cities, the number of vehicles in the countries has also incremented. With the increasing number of vehicles, the needs such as traffic controls and security controls have enlarged, and it has been difficult to follow each vehicle. Therefore, there was a need to improve the License Plate Recognition area. In this study, plate recognition has three stages. These stages are finding the plate region, character decomposition and character recognition methods. Before these stages, the image should be made clearer with some operations. Capturing and perceiving the image in flowing traffic is very difficult due to factors such as light and speed. This study first detects tools for License Plate Recognition, then applies a Convolutional Neural Network for character recognition on uncertain images and compares seven optimization algorithms. The results have been shown that accuracy and training time are superior compared to old plate recognition methods.

Keywords: *Character Recognition, Convolutional Neural Network, Flowing Traffic, License Plate Recognition*

1. GİRİŞ

Günümüzde nüfusun artması araç sayısının artmasına neden olmuştur ve araç sayısının artması beraberinde trafikteki sorunların artmasını da getirmiş, araçları takip etmek ve kontrol etmek zorlaşmıştır (Çevik & Çakır,2010). Bu nedenle trafik kontrolüne duyulan ihtiyaç artmış ve Plaka Tanıma Sistemine ihtiyaç duyulmuştur. Plakalar, araçların eşsiz kimlik numaraları gibidir. Tespit edilmek istenen araca ait tüm bilgilere ilgili araçların plakaları bulunduğunda erişilebilir. Bu da Plaka Tanıma Sistemini kullanışlı, pratik hale getirmektedir. Plaka Tanıma Sisteminin başlıca kullanım alanları; giriş-çıkışların kontrol altına alınması gereken yerler, ücret toplama sistemleri ve ücretli otoyollar, trafikte araçların takibi ve belirlenmesi, otomatik geçiş sistemleri gibi uygulamalardır. Bu herbir işlemin insan tarafından yapılması, işlem zamanını artırmaktadır.

Plaka Tanıma Sistemi otomatik geçiş ve park sistemleri, araç takibi ve tespiti, trafik kontrolü gibi birçok alanda kullanılmaktadır. Bu nedenle plaka yerinin bulunması ve plaka tanıma üzerine literatürde birçok çalışma bulunmaktadır. Yaygın olarak ise plaka bölgesi tespit edilerek plaka tanıma yapılmaktadır. Eski yöntemlerde, plaka tanıma yapabilmek için görüntü işleme sıkça kullanılmaktaydı. Fakat görüntünün kirli olması, düşük kontrastlı olması gibi özellikler bu yöntemin kullanılmasını oldukça zorlaştırmuştır ve bu yüzden çok fazla ön işleme gerekmektedir (Çelik, 2003). Bu çalışmada plaka tanıma yapılırken Evrişimli Sinir Ağı (ESA) kullanılmaktadır. ESA hesaplama, görüntü işleme gibi algoritmaları bünyesinde barındırır. Bu yüzden ESA önceden işlenmiş giriş görüntülerini kabul eder ve bu ön işlenmiş görüntüleri denetimli öğrenme ile eğitir.

Bir Evrişimli Sinir Ağı, bir girdi görüntüsünde yer alan, görüntüdeki çeşitli yönlere/nesnelere öğrenilebilir ağırlıklar ve sapmalar atayan ve çeşitli görüntülerin birbirinden ayırt edilebilmesini sağlayan bir Derin Öğrenme algoritmasıdır (Doğan G. , 2010). ESA'da gereken ön işleme, diğer sınıflandırma algoritmalarına kıyasla çok daha düşüktür. Bu nedenle daha az işlem ile çalışma tamamlanmış ve doğru sonuçlara ulaşılmıştır.

Bu tez çalışmasında, araçların plaka bölgesini tespit edip plaka tanıma işlemi yapan bir sitem geliştirilmiştir. Literatürde bu tanıma işleminin hazır modeller ile veya çok fazla ön işlemeye ihtiyaç duyularak, performansı çok yüksek olmayan algoritmalar ile yapıldığı görülmüştür. Bu çalışmanın ana katkısı, hız ve doğruluk açısından yedi algoritmanın sonuçlarına göre en iyi modeli belirlemek için çok fazla önişlemeye ihtiyaç duyulmamasıdır. Bu algoritmalar; Stochastic Gradient Descent (SGD), Root Mean Square Propagation (RMSProp), Adagrad, AdaDelta, Adam, Adamax ve Nadam optimizasyon algoritmalarıdır. Bu çalışma 29260 örnek üzerinde eğitilmiş ve 7316 örnek üzerinde doğrulanmıştır. Bu çalışma Python programlama dili ve Keras kütüphanesi kullanılarak tamamlanmıştır.

Çalışmada kameradan alınan anlık araç görüntülerinde öncelikle plaka bölgesi bulunmaktadır. Elde edilen görüntüdeki plaka karakterleri görüntüden ayrıştırılır. Son olarak, plaka bölgesinden çıkarılan karakterler tanıma işlemi için belirlenen optimizasyon algoritmalarına verilir ve plaka tanıma işlemi tamamlanır. Son aşamada ayıklanan karakterleri tanımak için Evrişimli Sinir Ağı kullanıldı ve yedi optimizasyon algoritması kullanılarak plaka tanıma işlemi tamamlanmıştır.

Bu tez çalışması giriş, kaynak araştırması, materyal ve metot, teorik esaslar, plaka tanıma sistemi ve sonuçlar ve öneriler olmak üzere altı bölüm ve kaynaklardan oluşmaktadır.

Birinci bölüm, giriş bölümü olup konunun genel tanımı yapılmış, çalışmanın amacı ve önemi üzerinde durulmuştur.

İkinci bölümde bu alanda yapılmış önceki çalışmalar hakkında literatür bilgisi verilmiş, bu çalışmaların özellikleri belirtilmiştir.

Üçüncü bölümde tez çalışmasında kullanılan materyal ve metotlar verilmiştir.

Dördüncü bölümde yapay sinir ağları ve evrişimli sinir ağlarının ile ilgili teorik esaslardan bahsedilmiş, bu ağların özellikleri, yapısı ve kullanılan optimizasyon algoritmaları anlatılmıştır.

Beşinci bölümde uygulaması yapılan araç plaka tanıma sistemi aşama aşama anlatılmıştır. Uygulama ile ilgili resimler, şekiller ve tablolar bu bölümde sunulmuştur.

Altıncı bölümde tez çalışmasının sonuçları üzerine genel bir değerlendirme yapılmıştır. Ayrıca çalışma ile ilgili öneriler de bu bölümde yer almaktadır.

Tez çalışmasının sonunda yararlanılan kaynaklar verilmiştir.

1.1 Tezin Amacı

Nüfusları ile orantısız büyüme gösteren ülkeler araçları takip etmekte zorlanmış ve bu zorluk en çok da kolluk kuvvetlerini etkilemektedir. Her geçen gün araç sayısı artmakta ve paralelinde bu ihtiyaç da artmaktadır. Geleneksel yöntemler ile yapılan Plaka tanıma sistemlerinde, plaka bölgesinin bulunabilmesi için çok fazla görüntü işleme tekniği ile ön işlemeler yapılması beklenmektedir. Bu bitirme tezinde ise, çok fazla ön işlemeye tabi tutulmayan girdi görüntüsü ile rahatlıkla plaka tanıma işlemi yapılabilmektedir. Tez çalışmasının akış şeması Şekil A.1'de gösterilmiştir.

2. KAYNAK ARAŞTIRMASI

Plaka tanıma alanında yapılan ilk çalışma 1960'larda ABD'de optik tarayıcı sistemler ile gerçekleştirilmiştir (Hauslen 1977). Araç tanımanın yol ücretlendirme için ilk olarak kullanılması Hong Kong'da 1983 yılında uygulanmıştır. 1988 yılında araçları takip etmek ve güvenliği arttırmak için Japonya'da altı ülkeyi kapsayan bir proje hayata geçirilmiş ve başarılı olunmuştur. Aynı yıllarda Hollanda ve Norveç'te de otomatik ücret toplamak amacı ile plaka tanıma sistemleri geliştirilmiştir (Stoelhurst ve Zandbergen 1990).

Araç plaka tanıma sistemi, "Elysdel" adlı bir şirket tarafından ilk kez gişelerde kullanılmıştır. Akan trafikte kamera ile plakasının resmi yakalanan araç Optik Sensörlerle tespit edilmiştir. Sistem Fransa'da 1980'li yılların sonunda denenmiştir (Setchell 1997).

1989'da ''Computer Recognition Systems'' adlı şirket ''Syntax Forcing'' algoritması kullandığı bir araç plaka tanıma sistemi geliştirmiş ve % 93 başarısının olduğu açıklanmıştır (Williams ve ark. 1989).

1990'larda plakanın yerini tespit edip karakter ayrıştırma ile plaka tanıma yapan sistemde ilk kez yapay sinir ağı kullanılmış ve %90 başarının olduğu açıklanmıştır. Bristol Üniversitesi bünyesinde bulunan araştırma merkezi, görüntüden yatay çizgiler alarak bu çizgiler üzerindeki histogramdan yararlanmıştır. Histogramdaki değişimlerden karakter olabilecek yerler saptanmış, karakterler aynı yöntemle ayrıştırılmış ve karakter tanımayı yapay sinir ağlarıyla gerçekleştirmiştir. Aynı yıllarda Avustralya'da yapılan çalışmada plaka yeri tespitinde kenar belirleme algoritmaları kullanılmış ve yine tanıma işlemi yapay sinir ağları ile yapılmıştır (Fahmy 1993).

Öğrenme tabanlı plaka tanıma sistemleri 2000'lí yıllarda kullanılmaya başlanmış ve Yapay Sinir Ağlarının da gelişmesi ile yöntemler değişerek günümüze kadar gelmiştir.

Öğrenme tabanlı, Yapay Sinir Ağları ile yapılan ilk çalışma Çin'de Changsha Üniversitesi'nde plaka görüntüleri ile yapılmıştır. Daha sonra ise Tayvan'da bir üniversitede, plaka tanıma için plaka yeri tespitinde karakter renklerinden yararlanılmıştır. Karakter tanımada ise birebir karşılaştırma yönteminden yararlanılarak tanıma işlemi tamamlanmıştır (Wei ve ark. 2001).

2004 yılında yine Çin'de Bulanık Mantık kuralları ile plaka tanıma işlemi yapılmış ve %95 başarı elde edilmiştir (Chang ve ark. 2004).

Suudi Arabistan plakaları için yapılan bir çalışmada plaka bölgesinin bulunması için kenar-köşe belirleme algoritmalarından yararlanılmış ve plaka karakterlerini tanımak için de Bulanık Mantık kuralları kullanılmıştır. Kenar-köşe belirleme algoritmaları, plaka bölgesinin tespitinde en etkili yöntemlerden biridir (Sayed ve Sarfraz 2005). 2005-2007 yılları arasında plaka bölgesinin bulunmasında bu kenar-köşe belirleme yöntemi sıkça kullanılmıştır.

2007 yılında Shandong Üniversitesinde plaka yeri tespiti için Hough algoritması kullanmıştır. Plaka karakter tanıma için ise Şablon Eşleme yöntemi kullanmış, saniyeler içinde veritabanındaki örnek şablonlar ile plakaları tespit edebilmiştir (Yang ve ark. 2007).

Plaka tanıma sistemi alanında ülkemizde yapılan çalışmalar ise şunlardır;2003 yılında Mustafa Kemal Üniversitesi'nde, YSA kullanılarak plaka yeri tespiti ve karakter tanıma yapılmıştır (Çelik, 2003).2006 yılında tamamlanan bir çalışmada, plaka bölgesi bulunurken kenar belirleme algoritması kullanılmıştır. Karakter ayrıştırmada ise bazı morfolojik işlemler uygulanmış ve karakter tanıma için Şablon Eşleme yöntemi kullanılmıştır. Sistem %91 başarı ile çalışmaktadır (Özbay 2006).

Hacettepe Üniversitesi'nde yapılan bir çalışmada, gömülü sistemler için çalışan plaka tanıma sistemi uygulanmıştır. Plaka bölgesi bulunması için kenar-köşe belirleme algoritmaları kullanılmış, karakter tanıma işlemi için ise model eğitimli YSA kullanılmıştır. Sistemin başarısı %87 olarak kaydedilmiştir (Caner 2006).

Ankara Üniversitesi'nde yapılan bir uygulamada, plaka tanıma işlemi yapılırken plaka rengi ve plaka karakter renkleri kullanılmıştır. Karakter ayrıştırma

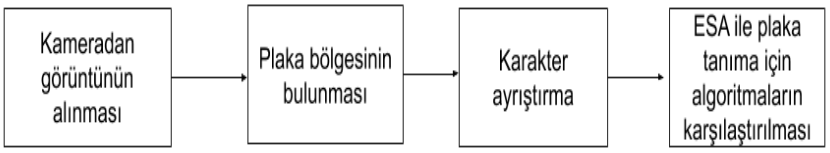
yöntemi olarak Sütun Toplam Vektörü kullanılmıştır. Çıkarılan karakterler YSA kullanılarak plaka tanıma yapılmıştır (Çamaşırcıoğlu 2007).

Bu şekilde geçmişten günümüze kadar plaka tanımada birçok yöntem kullanılmıştır. Çalışmalarda görüldüğü gibi kullanılan yöntemler arasında yapay zeka uygulamalarından yaygın olan; Yapay Sinir Ağları, Bulanık Mantık, Görüntü İşleme, Kenar-köşe algoritmaları, Şablon Eşleme gibi yöntemler kullanılmıştır. Bu tez için görüntü işleme, yapay sinir ağları ve evrişimli sinir ağları alanında yapılan plaka tanıma çalışmaları incelenmiş ve uygun olan yöntemler bu tezde kullanılmıştır. Ayrıca ülkemizdeki ve yabancı ülkelerdeki plaka örnekleri de incelenmiş ve uygun görülen bilgiler bu tezde kullanılmıştır.

3. MATERYAL ve YÖNTEM

Plaka tanıma için önerilen ve kullanılan birçok yöntem vardır. Bu çalışma başlatılırken birçok çalışmanın avantaj ve dezavantajı göz önünde bulundurularak, en iyi yöntem uygulanmaya çalışılmıştır. Bu tez çalışmasında, kameradan anlık yakalanan araç görüntüleri üzerinden plaka bölgesi bulunup, belirlenen bir veri seti ile algoritmalar eğitilerek plaka tanıma işlemi yapılmaktadır. Algoritmaların eğitildiği bilgisayar Intel® Core™ i7, 4.5 GHz'e kadar Turbo Boost ve 12 MB paylaşımlı L3 önbelleğe sahiptir ve 4 GB GDDR6 belleğe sahip AMD Radeon Pro 5300M ve otomatik grafik değiştirme özelliğine sahiptir. Ayrıca 16 GB saklama alanına sahiptir (URL-1).

Gerçekleştirilen plaka tanıma sistemi üç aşamadan oluşmaktadır. İlk aşamada araç plaka görüntüsü alınır. Alınan görüntüde, bir dizi işlem yapılarak plaka bölgesi bulunmaktadır. Bulunan plaka bölgesindeki karakterler ayrıştırılır. Son olarak ise ayrıştırılan karakterler plaka tanıma işlemi için belirlenen Evrişimli Sinir Ağı optimizasyon algoritmalarına verilir ve plakaların tanıma işlemi tamamlanır (Şekil 3.1).



Şekil 3.1: Plaka Tanıma Sistemi Blok Şeması

Plaka bölgesinin bulunması için Sobel kenar algılama algoritması kullanılmış ve plaka olması muhtemel bölgeler tespit edilmiştir. Plaka bölgesi çıkarılan görüntülerde karakter ayrıştırma yapılmıştır (Bakkaloğlu, 2011).

Plaka bölgesi üzerindeki karakterleri ayrıştırmak için Connected Component Analysis yöntemi kullanılmıştır. Böylece her karakter, karakter tanıma için bir veri seti haline gelmektedir (Bakkaloğlu, 2011).

Tanıma aşamasında ise Evrişimli Sinir Ağları kullanılmış ve yedi Gradient Descent optimizasyon algoritması, veri seti üzerindeki eğitim süresi ve doğruluk yönünden karşılaştırılmaktadır.

4. TEORİK ESASLAR

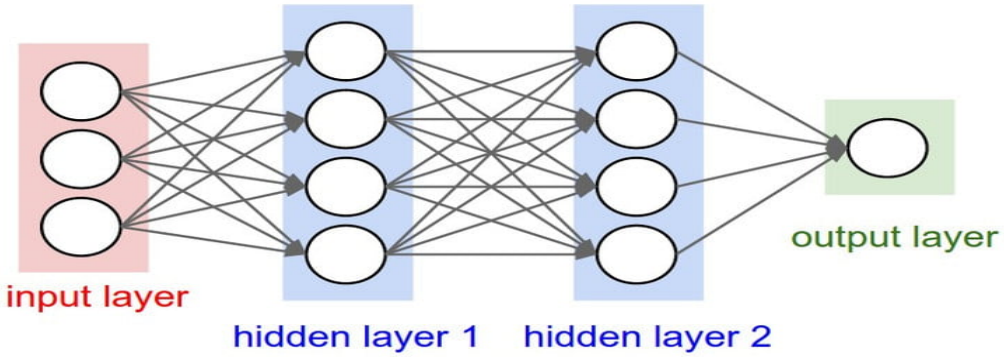
Bu bölümde plaka tanıma için kullanılan Yapay Sinir Ağları ve Evrişimli Sinir Ağları ile ilgili teorik esaslar, özellikler ve karşılşatırılan optimizasyon algoritmaları anlatılmaktadır.

4.1 Yapay Sinir Ağları

Teknolojinin gelişmesi ve yapay zeka alanında yapılan çalışmaların artması ile birlikte yapay sinir ağları geliştirilmiş ve tarihte ilk yapay sinir ağı modeli 1940'larda geliştirilmiştir. Yapay Sinir Ağları, isimlerindeki sinir kısmından da anlaşılacağı gibi; insanların öğrenme şekillerini tekrarlamayı amaçlayan, beyinden ilham alan sistemlerdir (Elmas 2003, Öztemel 2003, Sağıroğlu ve ark. 2003, Allahverdi 2002). Sinir ağları girdi ve çıktı katmanlarının yanı sıra (çoğu durumda) girdiyi çıktı katmanının kullanabileceği bir şeye dönüştüren birimlerden oluşan gizli bir katmandan oluşur. Bir insan programcının makineyi tanımasını ve tanımasını öğretmesi için çok karmaşık veya çok sayıda desen bulmak için mükemmel araçlardır. YSA'lar çok katmanlı bir ağda, farklı katmanların aradıkları özellikleri bulana, tanıyana kadar farklı özellikler çıkarır. Çok katmanlı sinir ağlarının katmanları aşağıdaki tanımlardaki gibidir.

- Giriş Katmanı: İşlenecek verinin, veri setinin ağa verildiği katmandır.
- Ara Katmanlar: Giriş katmanından alınan bilgilerin işlendiği bölüm olup tanıma, özellik çıkarma gibi yapılmak istenen ana işlemleri yapan katmandır. Gizli katman olarak da adlandırılır ve birden fazla olabilmektedir.
- Çıkış Katmanı: Ara katmandaki bilgiyi çıkış olarak ileten katmandır.

Şekil 4.1'de iki katmandan oluşan sinir ağı modeli gösterilmektedir.



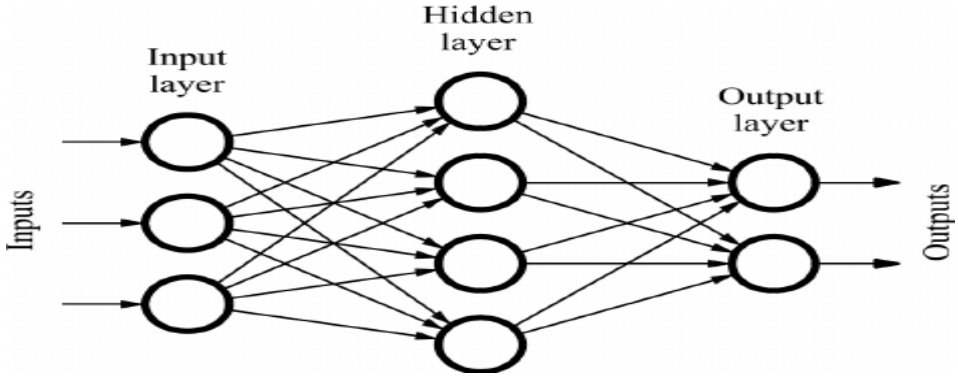
Şekil 4.1: Sinir Ağı Modeli (URL-2)

Şekil 4.1'de gösterilen sinir ağında 3 girdi bulunmaktadır. Örneğin; giriş katmanındaki nöronlara bir evin oda sayısı, metrekare genişliği ve bina yaşı değerlerini verirsek çıkış katmanında evin fiyatını tahmin edecektir. Ayrıca sinir ağında gizli katmandaki nöronlar, tüm 3 girdinin özelliklerini alır. İyi bir sinir ağındaki her girdi, gizli katmanların herbiri ile bağlantılı olmalıdır.

YSA'lar nöronların birbirine bağlanış şekline göre ileri beslemeli ve geri beslemeli olarak ikiye ayrılır.

4.1.1 İleri beslemeli Yapay Sinir Ağları

İleri beslemeli sinir ağı, verilerin giriş katmanından alınıp, varsa ara katmanda işlendiği ve son olarak çıktı katmanında çıktının görüldüğü bir ağ yapısındadır (Özveren, 2006). Ters yöne doğru ilerlemez ve döngü oluşturmamaktadır. Şekil 4.2'de ileri beslemeli nöron ağlarının yapısı gösterilmiştir.

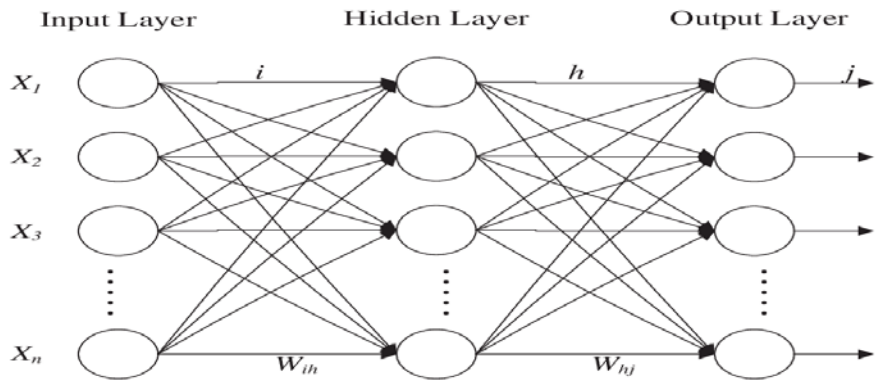


Şekil 4.2: İleri Beslemeli Sinir Ağı Yapısı (Yavuz & Deveci, 2012)

4.1.2 Geri beslemeli Yapay Sinir Ağları

Geri beslemeli sinir ağ yapılarında ise veri ters topolojik sırada döngü yapabilir. Yani çıktı katmanındaki bir veri kendinden önceki katmana ya da kendi

katmanındaki bir nörona girdi olarak verilebilir. Geri beslemeli sinir ağın yapısı Şekil 4.3'de gösterilmiştir.



Şekil 4.3: Geri Beslemeli Sinir Ağı Yapısı (Yavuz & Deveci, 2012)

4.1.3 YSA'nın genel özellikleri

YSA'nın genel özellikleri uygulanan ağ modeline göre değişebilmektedir. YSA genel özellikleri şunlarıdır (Öztemel 2003, Sağıroğlu ve ark. 2003, Allahverdi 2002):

- YSA kendi kendine öğrenme gerçekleştirebilmektedir. Durumları YSA kendi kendine öğrenerek birbirine benzer durumlara aynı kararları vermeye çalışmaktadır.
- YSA kendisine öğretilen örnekleri genelleyerek, kendi kendine sonuç üretebilmektedir.
- Kendisine gösterilen örnekler ile sürekli öğrenebilirler.
- Sadece nümerik bilgiler ile çalışırlar.
- YSA'ya verilen bilgiler eksik olsa bile eksik bilgilerle çalışabilirler.
- Görüntü tanıma yapabilmektedirler. Eğitim için ağa verilen görüntüleri işleyebilir, aynı zamanda sınıflandırabilirler.

4.1.4 YSA dezavantajları

YSA birçok konuda kolaylık sağlarken aynı zamanda dezavantajları da vardır. YSA dezavantajları şunlardır (Sağıroğlu ve ark. 2003, Haykin 1999):

- YSA'lar birçok hücre ve katmandan oluşmaktadır. Bu nedenle doğru ağ yapısının bulunması her zaman kolay olmamaktadır. Bu ağ yapısı her probleme göre farklılık gösterir ve denenerek bulunabilmektedir.
- Problem her ne ise, YSA'ya doğru şekilde tanıtmak gerekir. Bu problem ne olursa olsun nümerik değerlere çevrilerek ağa tanıtmak gerekir.
- YSA için model eğitiminin belirli bi süresi yoktur. Bu nedenle kayıp grafiğinin belirli bir değerin altına inmesi gözlemlenir ve bu şekilde model eğitimi tamamlanmaktadır.

4.1.5 YSA yapısı

YSA nöronlardan oluşmaktadır. Bu nöronlar katmanlarda, belirli bir kurala göre dizilmiştir. YSA'daki probleme göre gizli katman sayısı ve nöron sayısı belirlenmelidir. Doğru belirlenen nöron ve katman sayısı sistemin performansını arttırmaktadır. Genelde, iki veya üç katmanlı ağlar problemleri çözebilmektedir. Katmanlar, hücrelerin birbirine paralel olarak dizilmesiyle oluşmaktadır. Ayrıca girdi ve çıktı katmanlarının sayısı da probleme göre farklılık göstermektedir.

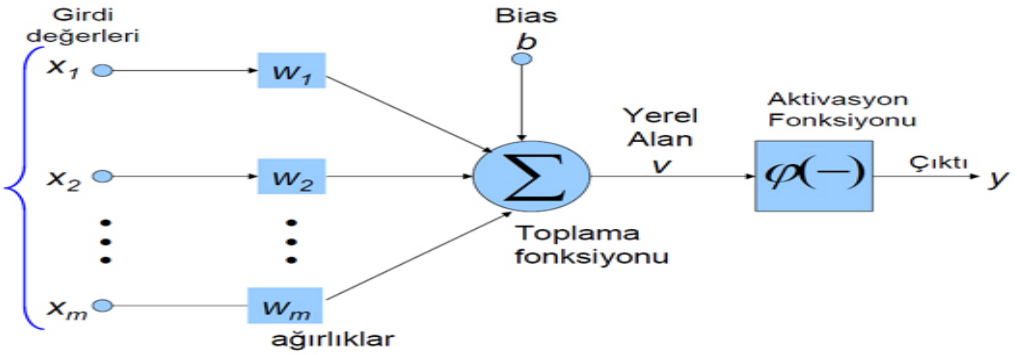
Bu farklılığın belirli bir çözümü olmamakla birlikte probleme göre değişir (Haykin 1999). Deneme-yanılma yolu ile gizli katman sayısı ve katmanlardaki nöron sayısı belirlenmektedir.

YSA modelinin eğitim süresi katmanlardaki nöron sayısı ve katman sayısına bağlı olarak değişmektedir. Ara katmanlardaki nöron sayısı ne kadar az olursa eğitim süresi de o kadar az olmaktadır. Nöron sayısı az olursa, YSA modeli daha performanslı çalışmaktadır. Nöron sayısının gereğinden fazla olması ise YSA'nın verileri öğrenmesini zorlaştırmaktadır. Aynı şekilde nöronların gereğinden az olması YSA'nın verileri öğrenememesine neden olmaktadır.

Son güncel çalışmalar ile YSA'nın ara katman sayısı ve nöron sayısının belirlenmesinde Genetik Algortimalarının oldukça iyi performans gösterdiği kanırlanmıştır. Özellikle optimizasyon uygulamalarında genetik algoritmaları son derece yararlı olmaktadır (Schwarz 1978).

4.1.6 YSA nöronları

Bir YSA modelinde, giriş, ağırlık (w), toplama fonksiyonu (∑), aktivasyon fonksiyonu ve çıkış bulunmaktadır. Giriş verisi, nörona ağırlıklar ile bağlanır. Toplama fonksiyonu ise giriş ve ağırlıkların çarpımını içerir. Aktivasyon fonksiyonu net çıkış değerini hesaplar ve çıkış fonksiyonuna iletir. Aktivasyon fonksiyonları çoğunlukla lineer olmayan fonksiyonlardır (Haykin 1999). YSA hücresi şekil 4.4'te verilmiştir.



Şekil 4.4: YSA Hücresi

Bias değeri bir sabit olup, b ile ifade edilir.

Nöronun çıkış değeri şöyle hesaplanmaktadır.

Çıkış değeri;

$$y = f(w.x + b) \qquad (\mathbf{4.1})$$

şeklinde hesaplanmaktadır. Burada w ağırlık matrisi, x ise giriş matrisidir. N giriş sayısı ise;

$$w = w_1, w_2, w_3 \ldots . w_n$$
$$x = x_1, x_2, x_3 \ldots x_n \qquad (\mathbf{4.2})$$

olarak ifade edilebilir.

Ve;

$$y = f\left(\sum_{i=1}^{n} w_i\, x_i + b\right) \qquad (\mathbf{4.3})$$

olarak da yazılabilmektedir.

Burada f aktivasyon fonksiyonudur. Genelde doğrusal olmayan aktivasyon fonksiyonlarının birçok çeşiti vardır. Eşik aktivasyon fonksiyonlarında f

değerinin parantez içi sıfırdan küçükse sıfıra, büyükse +1 değerine eşittir. Doğrusal aktivasyon fonksiyonlarının çıkışı, giriş değerine eşittir (Bishop 1995). Devamlı çıkış gerektiren ağlarda aktivasyon fonksiyonu olarak doğrusal aktivasyon fonksiyonu kullanılır.

$$f(x) = x \qquad (\mathbf{4.4})$$

olarak ifade edilmektedir. Yapay Sinir Ağlarında hangi aktivasyon fonksiyonunun kullanılacağı probleme göre çeşitlilik göstermektedir.

4.1.7 Aktivasyon fonksiyonları

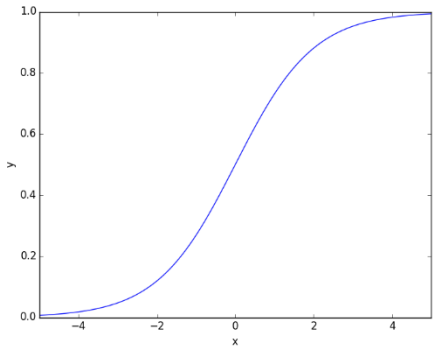
YSA'da, yapacağımız seçimlerden biri gizli katmanlarda hangi aktivasyon fonksiyonunun kullanılacağını belirlemektir. Aktivasyon fonksiyonu, sinir apının çıkış değerini de belirlemektedir.

4.1.7.1 Sigmoid fonksiyonu

Sigmoid fonksiyonu, YSA nöronuna gelen girdiyi alıp, çıkış değerini belirler. Çok katmanlı sinir ağlarında sıkça kullanılan bir aktivasyon fonksiyonudur (Öztemel, 2006). Doğrusal olmayan bir fonksiyon olan Sigmoid, 0 ve 1 arasında çıkış değerleri üretebilmektedir. İkili sınıflandırmada çıktıların 0 yada 1 olması istenir, bu yüzden ikili sınıflandırma kullanıyorsanız sigmoid fonksiyonunu çıktı katmanı için kullanabilirsiniz. Sigmoid fonksiyonunun dezavantajlarından biri eğer giriş değeri çok büyük veya çok küçükse fonksiyonun eğimi çok küçük olur ve bu da ağı yavaşlatmaktadır. Sigmoid fonksiyonunun denklemi, Denklem (4.5)'de gösterilmektedir.

$$F(NET) = \frac{1}{1 + e^{-NET}} \qquad \textbf{(4.5)}$$

Sigmoid, türevi alınabilir bir fonksiyon olup gösterimi ise Şekil 4.5'te gösterilmektedir (Öztemel, 2006).

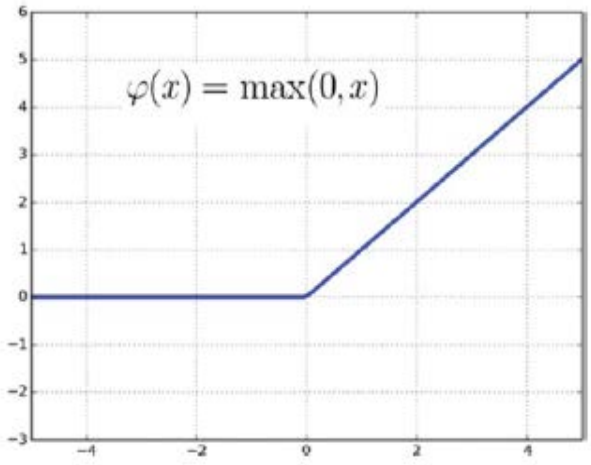


Şekil 4.5: Sigmoid Fonksiyonu Gösterimi

4.1.7.2 ReLu fonksiyonu

ReLu, özellikle Evrişimli Sinir Ağlarında ve Derin Öğrenmede sıkça kullanılan bir aktivasyon fonksiyonudur. Bu fonksiyonun türevi giriş değerinin pozitif olduğu yerde 1, negatif olduğu yerde sıfır çıkışını verir. ReLu'nun dezavantajı ise girişin negatif olması durumunda 0 değerini vermesidir. Fonksiyonun denklemi, Denklem (4.6)'da, gösterimi ise Şekil 4.6'da gösterilmektedir (Kim, 2017).

$$\varphi'(x) = \begin{cases} 1, x > 0 \\ 0, x \leq 0 \end{cases} \quad \textbf{(4.6)}$$

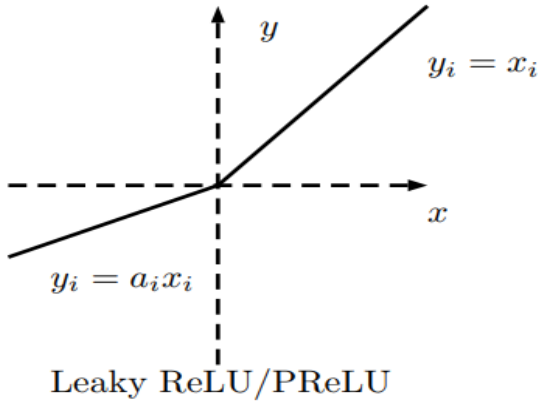


Şekil 4.6: ReLu Fonksiyonu Gösterimi

4.1.7.3 Leaky ReLu fonskiyonu

ReLu'daki negatif değerlerin yok edilmesi için geliştirilmiştir. Leaky ReLu sürümü, giriş değeri negatif olduğunda 0 yerine 0,01gibi hafif bir eğim alır. Leaky ReLu fonksiyonunun denklemi, Denklem (4.7)'de, gösterimi ise Şekil 4.7'de verilmiştir (Kuş, 2019).

$$y_i = \begin{cases} x_i, IF\ x_i > 0 \\ a_i x_i, IF\ x_i \leq 0 \end{cases} \quad (4.7)$$

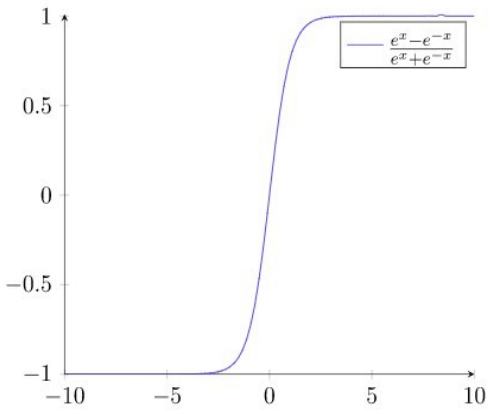


Şekil 4.7: Leaky ReLu Fonksiyonu Gösterimi

4.1.7.4 Tanh fonksiyonu

Tanh fonksiyonu, sigmoid fonksiyonuna çok benzer. Doğrusal olmayan tanh fonksiyonu, -1 ve +1 arasında çıkış değeri üretebilmektedir. Tanh fonksiyonu neredeyse her zaman Sigmoid fonksiyonundan daha iyi çalışır çünkü +1 ve -1 arasındaki değerler, aktivasyon ortalaması ve 0 ortalamasına daha yakındır. Bu, bir sonraki katman için öğrenmeyi biraz daha kolaylaştırmaktadır. Sigmoid fonksiyonunda olduğu gibi; dezavantajlarından biri eğer giriş değeri çok büyük veya çok küçükse fonksiyonun eğimi çok küçük olur ve bu da ağı yavaşlatmaktadır. Fonksiyonun denklemi, Denklem (4.8)'de, gösterimi ise Şekil 4.8'de gösterilmektedir. (Talu, 2017).

$$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (\mathbf{4.8})$$



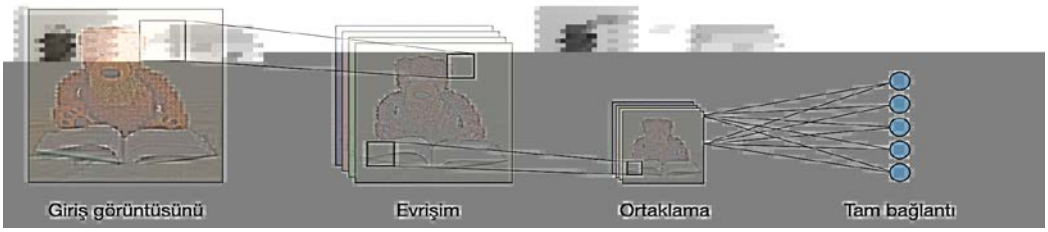
Şekil 4.8: Tanh Fonksiyonu Gösterimi

4.2 Evrişimli Sinir Ağları

Evrişimli sinir ağları (ESA), çok katmanlı sinir ağının bir çeşitidir ve öğrenebilir ağırlıklara ve sezgilere sahip nöronlardan oluşur. Evrişimli sinir ağları, esas olarak görüntüleri sınıflandırmak, görüntüleri benzerliğe göre kümelemek ve nesne tanıma gerçekleştirmek için kullanılan sinir ağlarıdır. Örneğin, evrişimli sinir ağları yüzler, bireyler, sokak işaretleri, tümörler ve görsel verilerin diğer birçok yönünü tanımlamak için kullanılır (LeCun, Bengio, & Hinton, 2015). ESA'ların önemli katkılarından birisi, aynı sayıda gizli birimle bağlı ağlardan daha az sayıda eğitime ve daha az sayıda parametreye sahip olmalarıdır

Evrişimli ağlarda görüntü tanımanın geliştirilmesi ile birlikte derin öğrenme de gelişmiş ve ESA'lar, kendi kendini süren arabalar, robotlar, dronlar, güvenlik, tıbbi teşhisler ve görme engelliler için tedaviler için bariz uygulamalara sahip olan bilgisayar görüsü (Computer Vision) konusunda büyük ilerlemeler sağlamaktadır.

Bununla birlikte, ESA'lar görüntü tanıma ile sınırlı değildir. Bu ağlar, doğrudan metin analizlerinde de kullaılabilmektedir. İleri beslemeli yapay sinir ağı olan ESA, Evrişim Katmanı, Ortaklama katmanı ve Tam Bağlantı Katmanından oluşmaktadır. Bu katmanların mimarisi Şekil 4.9'da gösterilmektedir.



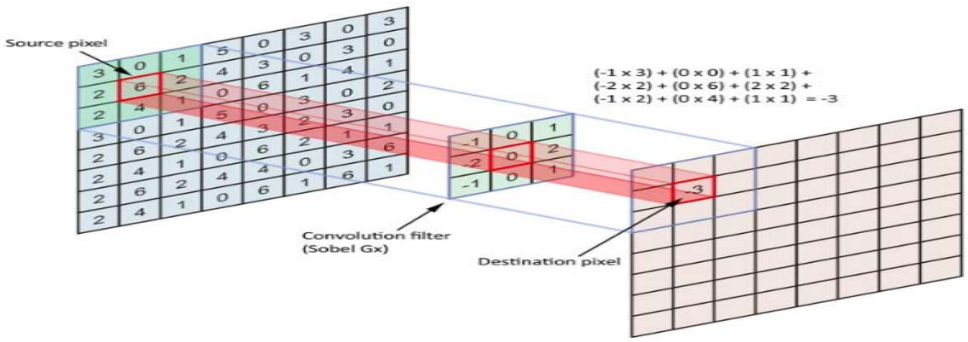
Şekil 4.9: Evrişimli Sinir Ağı Mimarisi (Url-3)

Bu katmanların kısa tanımı aşağıdaki gibidir;

4.2.1 Evrişim katmanı

Evrişim katmanı, konvolüsyonel işlemleri uygulayan filtreleri giriş görüntüsü boyutlarına göre tararken kullanılmaktadır. Yani belirli bir filtre, girdi görüntüsü üzerinde dolaşarak görüntüdeki nitelikleri belirler. Bu sayede girdi görüntüsü boyutunda yeni bir veri elde edilmiş olur (Liu, Shen, & Hengel,

2015). Aynı zamanda bu katmandaki filtreler kendi kendine öğrenebilir yapıdadır. Evrişim katmanının mimarisi, Şekil 4.10'da gösterilmiştir.



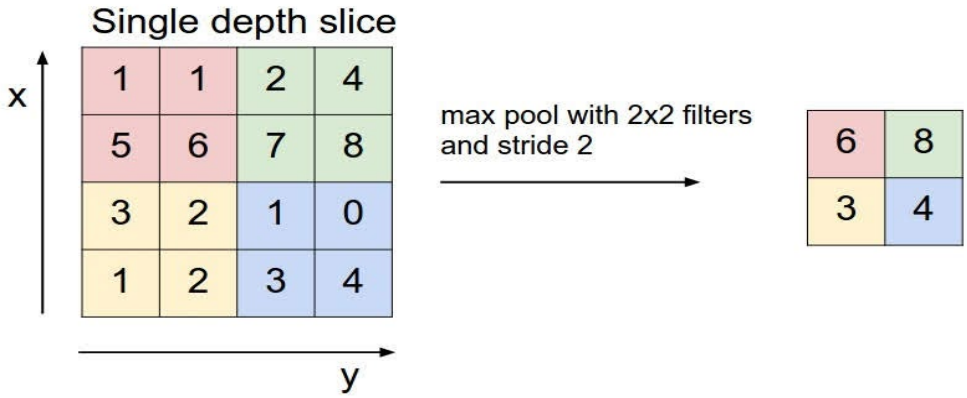
Şekil 4.10: Evrişim Katmanı (Url-4)

Evrişim işleminde kullanılan formül Denklem (4.9)'da gösterilmiştir. Formülde *Ç* işlem sonucu elde edilen çıktıyı, *F* filtre matrisinin boyutunu, *B* filtre matrisini ve *G* girdi matrisini göstermektedir (Doğan M. , 2019)

$$Ç_{i,j} = \sum_{k=1}^{F}\sum_{l=1}^{F} B_{k,l} G_{i+k-1,j+l-1} \quad \textbf{(4.9)}$$

4.2.2 Ortaklama katmanı

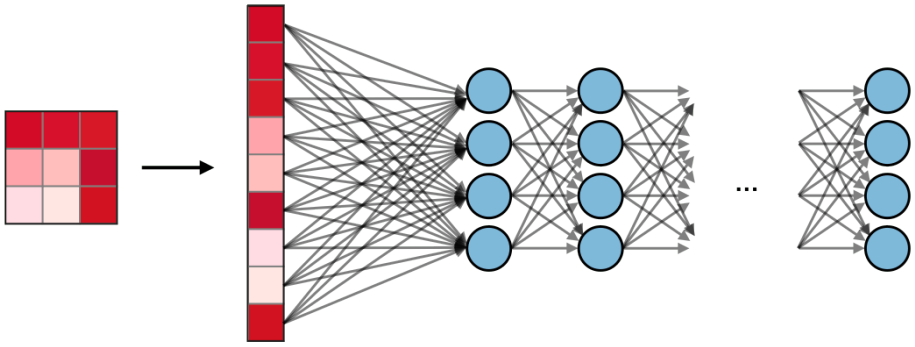
Ortaklama katmanında evrişim katmanında olduğu gibibir takım filtreler uygulanarak filtreler görüntü üzerinde gezdirilmektedir. Her zaman aynı olmamak şartı ile genelde görüntüdeki piksellerin maksimum değerleri alınmaktadır (Hijazi, Kumar, & Chris, 2015). Ortaklama katmanı adımları Şekil 4.11'de gösterilmektedir.



Şekil 4.11: Ortaklama Katmanı (Url-5)

4.2.3 Tam bağlantı katmanı

Tam bağlantı katmanı, sinir ağındaki her girişindeki tüm nöronların önceki katmanlara tam olarak bağlı olduğu katmandır. Çok katmanlı sinir ağına karşılık gelmektedir (Şeker, 2017). Tam bağlantı katmanı Şekil 4.8'de gösterilmiştir.



Şekil 4.12: Tam Bağlantı Katmanı (Url-6)

4.3 Optimizasyon Algoritmaları

Bu tez çalışmasında plaka tanıma işleminin yapıldığı ana bölüm modelin optimizasyon algoritmalarına verildiği bölümdür. Sinir ağı çalışmalarında optimizasyon, uygulama sonucunda hata oranının mutlak minimum değerine olabildiğince yaklaştırmaktır. Optimizasyonda amaç; eğitim sürecinde ağdaki bias (b) ve ağırlık (w) değerlerinin güncellenerek optimum çözümü bulmaktır (Yazan & Talu, 2017). Bu çalışmada optimizasyon için kullanılan yöntem Gradient Descent, kullanılan algoritmalar da Gradient Descent Optimizasyon Algoritmalarıdır. Gradient Descent yönteminin ana amacı; sinir ağı parametrelerine bağlı cost fonksiyonunu minimuma indirmektir. Bu bölümde hız ve doğruluk açısından 7 Gradient Descent optimizasyon algoritması karşılaştırılmış, Python dili ve Keras kütüphanesi kullanılmıştır. Bu algoritmalar Stokastik Gradient Descent (SGD), Root Mean Square Propagation (RMSProp), Adagrad, Adadelta, Adam, Adamax ve Nadam optimizasyon algoritmalarıdır (Kurt, 2018).

4.3.1 RMSprop optimizasyon algoritması

Bu algoritma öğrenme oranını (learning rate) üstel olarak azalan eğimin karesinin ortalamasına bölerek çalışmaktadır. Bu optimize edicinin parametrelerinin varsayılan değerlerinde bırakılması önerilmektedir (serbestçe

ayarlanabilen öğrenme oranı hariç). Algoritmayı güncelleme yöntemi Denklem (4.6)'da gösterilmektedir (Ruder, 2016).

$$E[g^2]_t = 0.9E[g^2]_{t-1} + 0.1g_t^2$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_{t+\epsilon}}} g_t \qquad \textbf{(4.6)}$$

$$g_t = \nabla_{\theta_t} J(\theta_t)$$

4.3.2 SGD optimizasyon algoritması

Bu algoritma her eğitim örneği için parametre güncellemesi gerçekleştirir. Bu optimize edici; momentum, öğrenme hızı azalması ve Nesterov momentumu desteği içerir. SGD yüksek bir varyans ile sık güncellemeler gerçekleştirir. Bu nedenle oldukça hızlıdır. Algoritmanın güncelleme yöntemi Denklem (4.10)'da gösterilmektedir (Yazan & Talu, 2017).

$$\theta = \theta - \eta . \nabla_\theta J\left(\theta; x^{(i)}; y^{(i)}\right) \#(\mathbf{4.10})$$

4.3.3 Adagrad optimizasyon algoritması

Adagrad, bir parametrenin eğitim sırasında ne sıklıkta güncelleneceğine göre uyarlanan, parametreye özgü öğrenme oranlarına sahip bir optimize edicidir. Bir parametre ne kadar çok güncelleme alırsa öğrenme oranı o kadar düşük olur.Bu optimize edicinin parametrelerinin varsayılan değerlerinde bırakılması önerilir. Adagrad optimize edicinin öğrenme oranı güncelleme yöntemi Denklem (4.11)'de gösterilmektedir (Ruder, 2016).

$$g_{t,i} = \nabla_{\theta_t} J\left(\theta_{t,i}\right)$$

$$\theta_{t+1,i} = \theta_{t,i} - \frac{n}{\sqrt{G_{t,ii} + \epsilon}} g_{t,i} \qquad \textbf{(4.11)}$$

4.3.4 Adadelta optimizasyon algoritması

Adadelta, öğrenme oranlarını, geçmiş tüm indirgemeleri biriktirmek yerine, degrade güncellemelerinin hareketli bir penceresine göre uyarlayan Adagrad'ın daha sağlam bir uzantısıdır. Bu şekilde Adadelta, birçok güncelleme yapıldığında bile öğrenmeye devam eder. Adagrad ile karşılaştırıldığında, Adadelta'nın orijinal versiyonunda bir başlangıç öğrenme oranı belirlemenize gerek yoktur. Bu versiyonda, diğer Keras optimize edicilerin çoğunda olduğu gibi başlangıç öğrenme oranı ve bozulma faktörü ayarlanabilir. Bu optimize edicinin parametrelerinin varsayılan değerlerinde bırakılması önerilir. Adadelta algoritmasının güncelleme formülü Denklem (4.12)'de gösterilmiştir (Yazan & Talu, 2017).

$$\Delta\theta_t = -\eta . g_t, i$$
$$\theta_{t+1} = \theta_t + \Delta\theta_t \#(\mathbf{4.12})$$

4.3.5 Adam optimizasyon algoritması

Adam, her parametre için uyarlanabilir öğrenme oranlarını hesaplayan bir başka yöntemdir. Adadelta ve Rmsprop'taki gibi geçmiş eğimlerin karelerinin üssel olarak ağırlıklandırılmış ortalamalarının (v_t) bulunmasının yanında, momentumdaki geçmiş değişikliklerini de (m_t) önbellekte tutar. Adam algoritmasının güncelleme yöntemi Denklem (4.13)'te gösterilmiştir.

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_{t^2}$$

$$m'_t = \frac{m_t}{1-\beta_1^t}, v'_t = \frac{v_t}{1-\beta_2^t} \quad \textbf{(4.13)}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v'_t + \epsilon}} m'_t$$

Bu eşitliğin varsayılan değerleri β_1 için 0.9, β_2 için 0.999, ϵ için 10^{-8} olarak belirtilmiştir (Kingma & Ba, 2017).

4.3.6 Adamax optimizasyon algoritması

Sonsuzluk normuna dayanan bir Adam çeşididir. Adam'da olduğu gibi (v_t) ve (m_t) değerlerini önbellekte tutar. Bu optimize edicinin parametrelerinin varsayılan değerlerinde bırakılması önerilir. Adamax algoritmasının güncelleme formülü Denklem (4.14)'te gösterilmektedir (Yazan & Talu, 2017).

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$$

$$v_t = \beta_2^{\infty} v_{t-1} + (1+\beta_2^{\infty}) |g_t|^{\infty}$$
$$= \max(\beta_2 . v_{t-1}, |g_t|$$
$$m'_t = \frac{m_t}{1-\beta_1^t}, v'_t = \frac{v_t}{1-\beta_2^t}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{v_t} m'_t \#(\mathbf{4.14})$$

4.3.7 Nadam optimizasyon algoritması

Nadam, Nesterov accelerated gradient (NAG) ve Adam kombinasyonundan oluşmuştur. Adam algoritmasını ile NAG algoritması kombine edebilmek için momentum ifadesi üzerinde değişiklik yapılması ile birlikte Nadam algoritması oluşturulur. NAG ile birlikte eğim değeri hesaplamadan önce parametreler momentum ifadesiyle güncellenerek eğim yönünde daha tutarlı bir sonuç alınır. Nadam algoritmasının yöntemi Denklem (4.15)'te gösterilmiştir (Ruder, 2016).

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v'_t + \epsilon}} \left(\beta_1 m'_t + \frac{(1-\beta_1) g_t}{1-\beta_1^t} \right) \quad \textbf{(4.15)}$$

5. PLAKA TANIMA SİSTEMİ

5.1 Veri Setinin Oluşturulması

Veri setinin çok yönlü ve daha yararlı olabilmesi için birçok ülkeden alınan hazır plaka örnekleri, farklı açı ve ışık yoğunluğunda, kontrast değerleri çeşitliliği ile Evrişimli Sinir Ağı modeli eğitilmiştir. Bu tez çalışması 29260 örnek üzerinde eğitilmiş ve 7316 örnek üzerinde doğrulanmıştır. Test edilen plaka veri seti ile ilgili örnek Şekil 5.1'de verilmiştir.

Şekil 5.1: Test Veri Seti Örneği

Görüldüğü gibi farklı ülkelerden de olsa plaka karakterleri 0-9 arası rakamlarıve A-Z arasın harfleri içermektedir. Bu nedenle veri seti olarak hazırlanan farklı ülkelerden bulunan plaka fotoğraflarından ayrıştırılan karakterler ile Evrişimli Sinir Ağı eğitilmiş, yedi optimizasyon algoritması doğruluk ve eğitim süresi

yöninden karşılaştırılmıştır ve 7316 örnek üzerinde doğrulanmıştır. Veri setinin çeşitliliği, farklı açılardan çekilmiş olması, büyüklüğü ve eğitim için verilen görüntülerin temiz olması doğruluğu arttırmaktadır. Görüntülerin %80'i eğitim için ayrılmış, %20'si ise veri setinin testi için ayrılmıştır. Veri seti büyüdükçe modelin eğitim süresi de artmaktadır. Bu da veri seti büyüklülük ve çeşitliliğinin tek dezavantajıdır.

5.2 Plaka Bölgesinin Bulunması

Geleneksel yöntemler incelendiğinde görülmüştür ki, plaka tanıma için önerilen ve kullanılan birçok yöntem vardır. Bu çalışma başlatılırken birçok çalışmanın avantaj ve dezavantajı göz önünde bulundurularak, en iyi yöntem uygulanmaya çalışıldı. Her şeyden önce, yakalanan araba görüntülerinde plaka bölgesi bulunmalıdır.

ESA ile plaka bölgesinin bulunması için bir dizi işlem yapılmalıdır. Plaka tanıma sisteminde anlık alınan görüntüler RGB formatındadır. Bu formattaki görüntüleri bilgisayar hesaplayamadığı için gri formata dönüştürülür. Bundan sonra gri formata çevirilen görüntü Sobel kenar algılama algoritması ile işlenir. Kenar algılama işleminin histogramı üretilir. Eşik değeri, plaka olması muhtemel olmayan bölgeleri belirlemek için belirlenir. Histogram grafiğinin yatay ve dikey sütunları incelenir ve ortalama kenar algılama yöntemi ile plaka bölgesi bulunur. Yakalanan görüntülerdeki plaka bölgesinin bulunması Şekil 5.2'de gösterilmiştir.

Şekil 5.2: Plaka Bölgesinin Bulunması

5.3 Karakter Ayrıştırma

Plaka bölgeinin bulunması adımı ile birlikte görüntüde yalnızca plaka bölgesi kalır. Bu adımdan sonra, plaka bölgesi üzerindeki karakterleri ayrıştırmak için Connected Component Analysis yöntemi kullanılır. Böylece her karakter, karakter tanıma için ayrı bir görüntü olarak çıkar. Ayrıştırılan karakter örnekleri Şekil 5.3'te gösterilmektedir.

Şekil 5.3: Ayrıştırılan Plaka Karakter Örnekleri

5.4 Karakter Tanıma

Kaynak görüntüden çıkarılan ve daha önce plaka karakterlerinin tanıtıldığı karakterler şablon olarak aranır. Bu yönteme Şablon Eşleme yöntemi denir. Plaka karakterlerinin bulunduğu görseller ile Evrişimli Sinir ağının eğitim seti oluşturulmuştur. Ayrıştırılan karakterler Evrişimli Sinir Ağına test verisi olarak verilmiş ve daha önceden hazırlanan örnek şablon karakterleri ile ayrıştırlan karakterler karşılaştırılmıştır. Bu şekilde karakter tanıma yapılmaktadır. Plaka tanıma işlemi Şekil 5.4'te gösterilmektedir.

En son aşamadan sonra, görüntü test edilecek optimizasyon algoritmaları için bir girdidir. Yukarıdaki işlemlerin uygulandığı görüntüler 7 Gradient Descent optimizasyon algoritmasının hepsinde denenir ve sonuçlar hız ve doğruluk açısından karşılaştırılır.

Şekil 5.4: Plaka Tanıma İşlemi

5.5 Plaka Tanıma için Algoritmaların Karşılaştırılması

Bu bölüm, plaka tanıma işleminin yapıldığı ana bölümdür. Bu bölümde hız ve doğruluk açısından 7 optimizasyon algoritması karşılaştırılmış, Python dili ve Keras kütüphanesi kullanılmıştır. Bu algoritmalar Stokastik Degrade İniş (SGD), RMSProp, Adagrad, Adadelta, Adam, Adamax ve Nadam optimizasyon algoritmalarıdır. Algoritmalar ile ilgili detaylı bilgi bölüm 4.3'te verilmiştir.

Belirlenen yedi optimizasyon algoritmasının, farklı ülkelerdeki plaka kodları ile toplanmış veri seti ile ESA üzerindeki doğruluk ve ortalama eğitim süresi karşılaştırılmıştır. Belirlenen ve karşılaştırması yapılan optimize ediciler Çizelge 5.1'de verilmiştir.

Çizelge 5.1: Optimizasyon Algoritmaları Formülleri

Optimize Edici	Formülü
SGD	$\theta = \theta - \eta \cdot \nabla_\theta J(\theta; x^{(i)}; y^{(i)})$
RMSprop	$E[g^2]_t = 0.9E[g^2]_{t-1} + 0.1g_t^2$ $\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_{t+\epsilon}}} g_t$ $g_t = \nabla_{\theta_t} J(\theta_t)$
Adam	$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$ $v_t = \beta_2 v_{t-1} + (1-\beta_2) g_{t^2}$ $m'_t = \frac{m_t}{1-\beta_1^t}, v'_t = \frac{v_t}{1-\beta_2^t}$ $\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v'_t + \epsilon}} m'_t$

Çizelge 5.1: (devam) Optimizasyon Algoritmaları Formülleri

Optimize Edici	Formülü
Adamax	$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$ $v_t = \beta_2{}^{\infty} v_{t-1} + (1+\beta_2{}^{\infty}) \lvert g_t \rvert^{\infty}$ $= \max(\beta_2 . v_{t-1}, \lvert g_t \rvert$ $m'_t = \frac{m_t}{1-\beta_1^t}, v'_t = \frac{v_t}{1-\beta_2^t}$ $\theta_{t+1} = \theta_t - \frac{\eta}{v_t} m'_t$
AdaGrad	$g_{t,i} = \nabla_{\theta_t} J(\theta_{t,i})$ $\theta_{t+1,i} = \theta_{t,i} - \frac{n}{\sqrt{G_{t,ii} + \epsilon}} g_{t,i}$
AdaDelta	$\Delta\theta_t = -\eta . g_t, i$ $\theta_{t+1} = \theta_t + \Delta\theta_t$
Nadam	$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v'_t + \epsilon}}$ $(\beta_1 m'_t + \frac{(1-\beta_1) g_t}{1-\beta_1^t})$

6. SONUÇLAR VE ÖNERİLER

6.1 Sonuçlar

Rastgele çekilen plaka görüntülerinin eğitildiği bu Plaka Tanıma Sisteminde ön işleme yapılmadan Yapay Sinir Ağları ile 7 Optimizayon algoritması karşılşatırılarak sonuçlar ölçülmüştür. Uygulanan yöntemler ile hızlı bir şekilde tespit edilen plaka bölgesinden çıkarılan karakterler ile tanıma işlemi yapılmştır. Yedi algoritmada test edilen ve 7316 farklı örnek üzerinde doğrulanan bu çalışma, Adam algoritmasının doğruluğunun en yüksek olduğunu göstermiştir. En hızlı algoritmanın ise Adagrad algoritması olduğu ispatlanmıştır. Algoritmaların karşılaştırılması Çizelge 6.1'de verilmiştir.

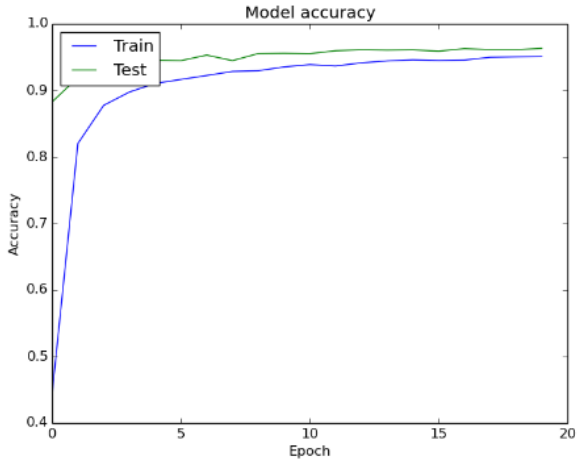
(a)

Çizelge 6.1: Optimizasyon Algoritmaları Performans Karşılaştırması

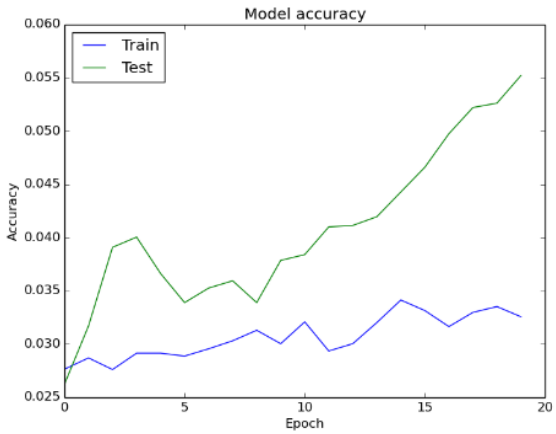
Optimizasyon algoritması	**Ortalama eğitim süresi (sn)**	**1. iterasyondaki doğruluk değeri (%)**	**20. iterasyondaki doğruluk değeri (%)**
SGD	46.40 sn	%6.68	%92.39
RMSProp	45.75 sn	%52.61	%93.51
AdaGrad	44.75 sn	%5.21	%73.95
AdaDelta	46.05 sn	%23.76	%94.32
Adam	45.85 sn	%44.40	%95.09
	45.65 sn	%12.53	%94.15
Adamax	46.40 sn	%47.87	%95.06
Nadam			

Karşılaştırılan bu algoritmaların 20 iterasyon sonucu çıkan doğruluk grafikleri Şekil 6.1'de gösterilmektedir.

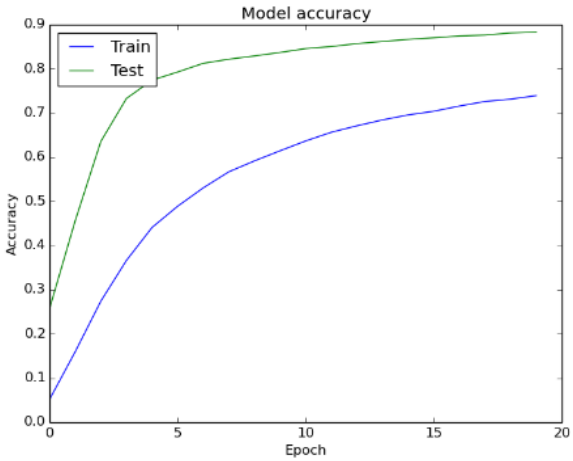
Adam



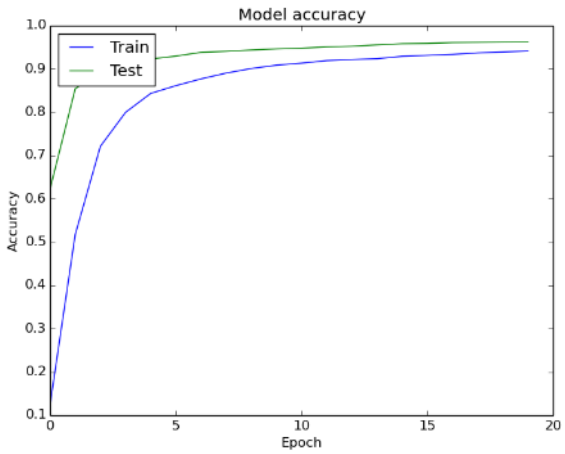
Adadelta



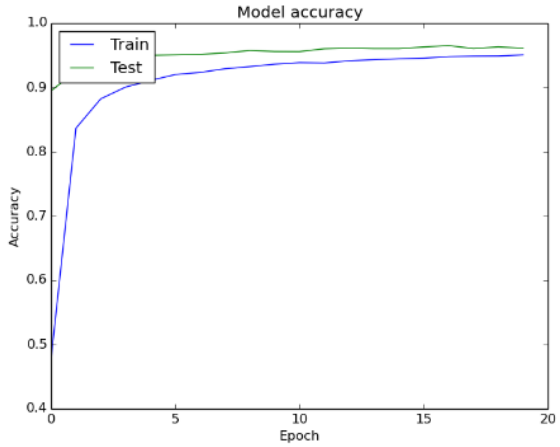
AdaGrad



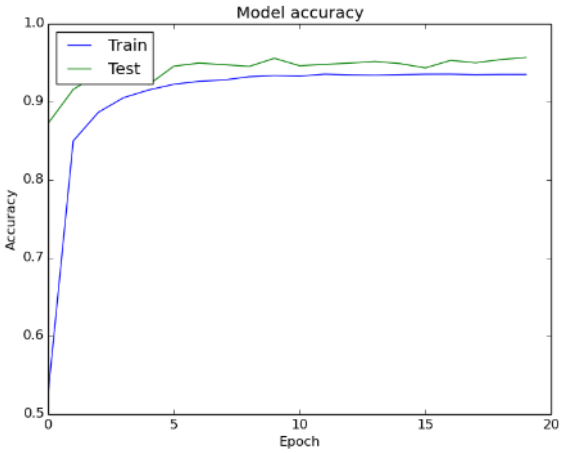
Adamax



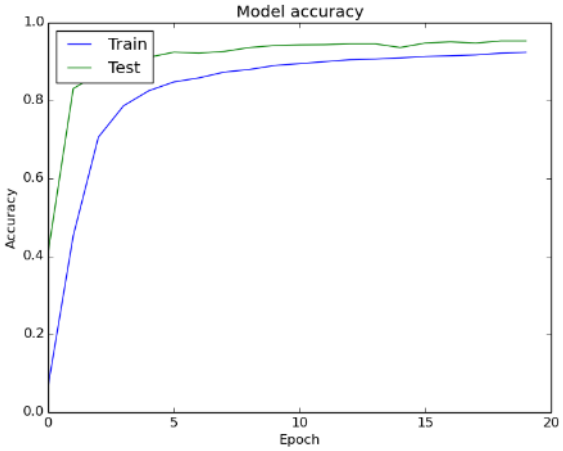
Nadam



Rmsprop



SGD

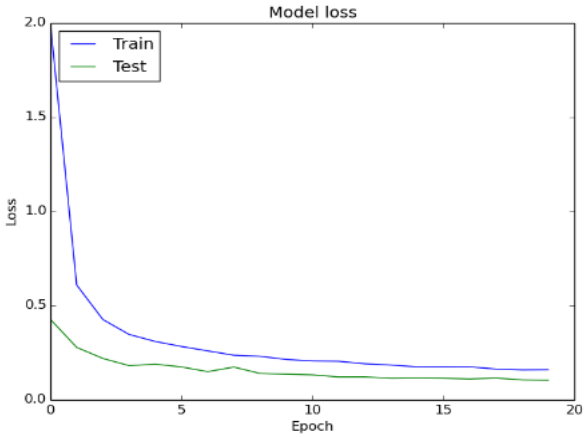


Şekil 6.1: Optimizasyon Algoritmaları Doğruluk Grafikleri

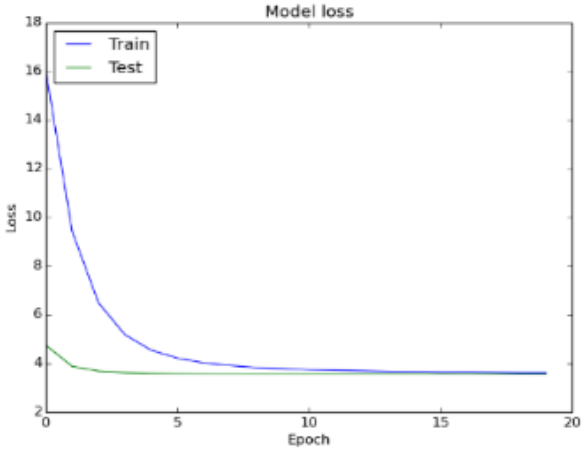
Şekil 6.1’de de görüldüğü gibi karşılaştırılan yedi optimizasyon algoritmasından, Adam algoritmasının test veri seti üzerinde doğruluğu en yüksek algoritma olduğu ispatlanmıştır.

Karşılaştırması yapılan algoritmaların kayıp grafikleri ise Şekil 6.2’de görülmektedir.

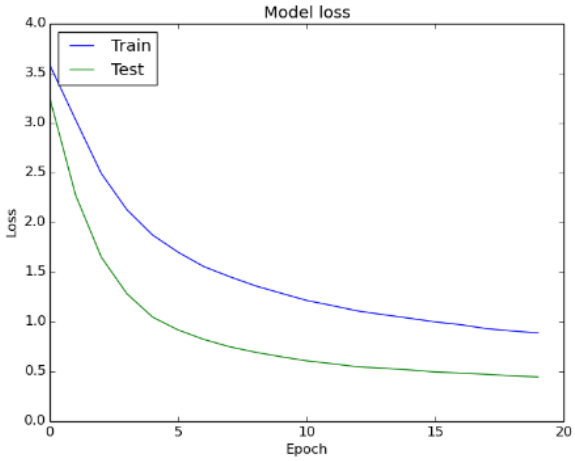
Adam



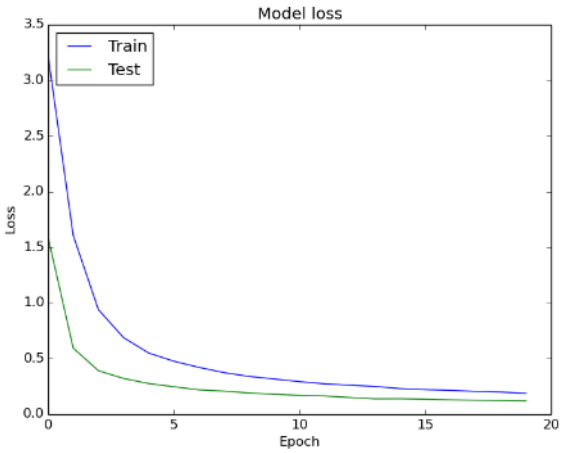
Adadelta



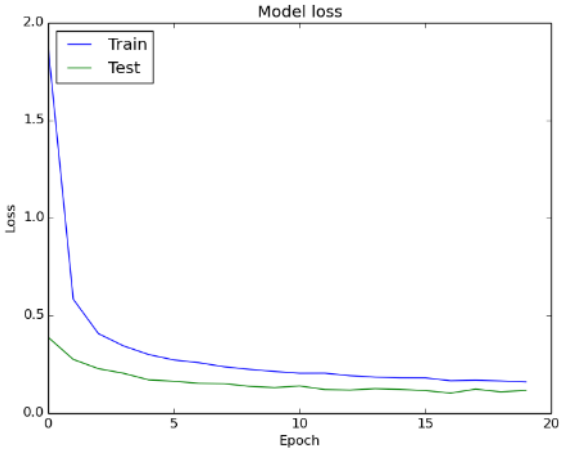
Adagrad



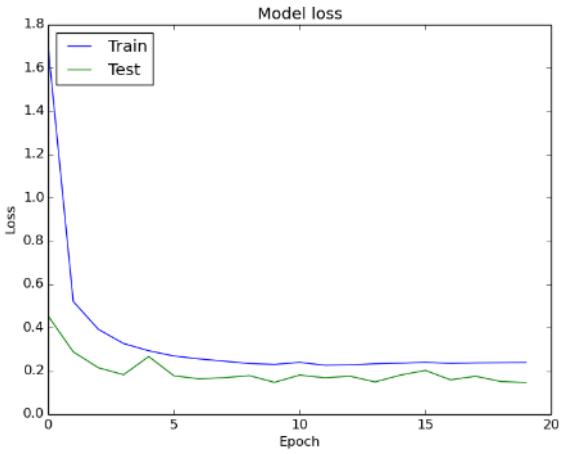
Adamax



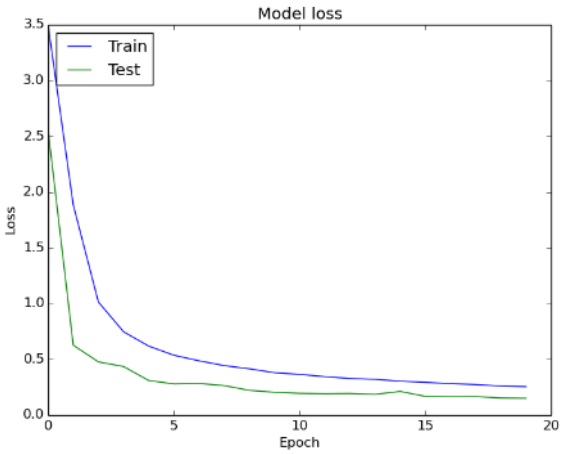
Nadam



Rmsprop



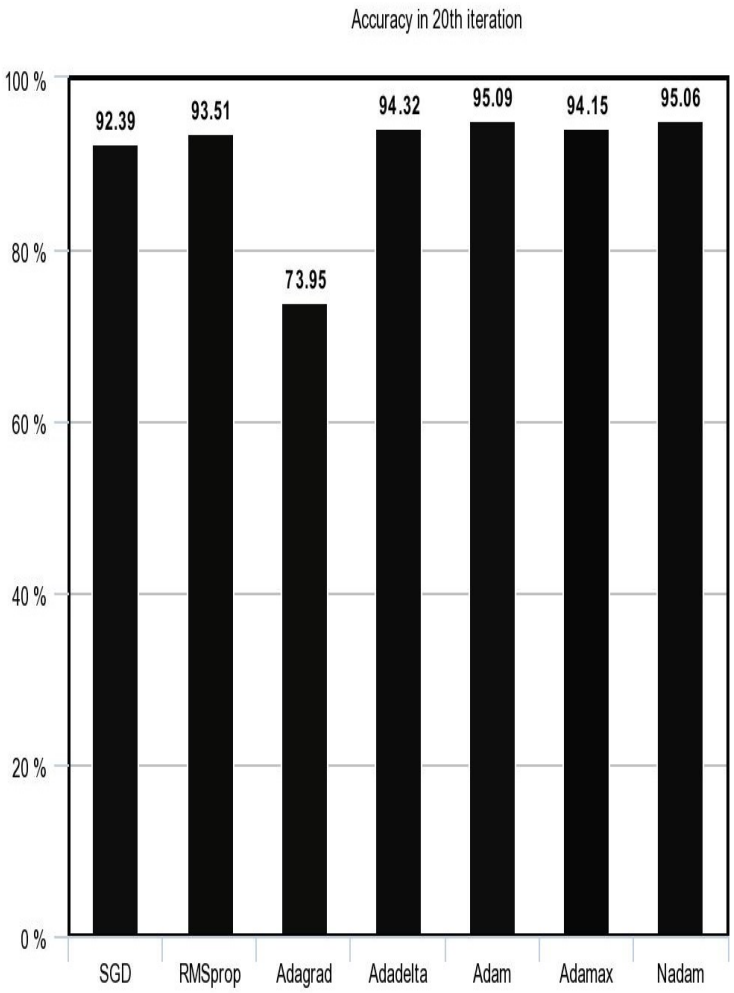
SGD



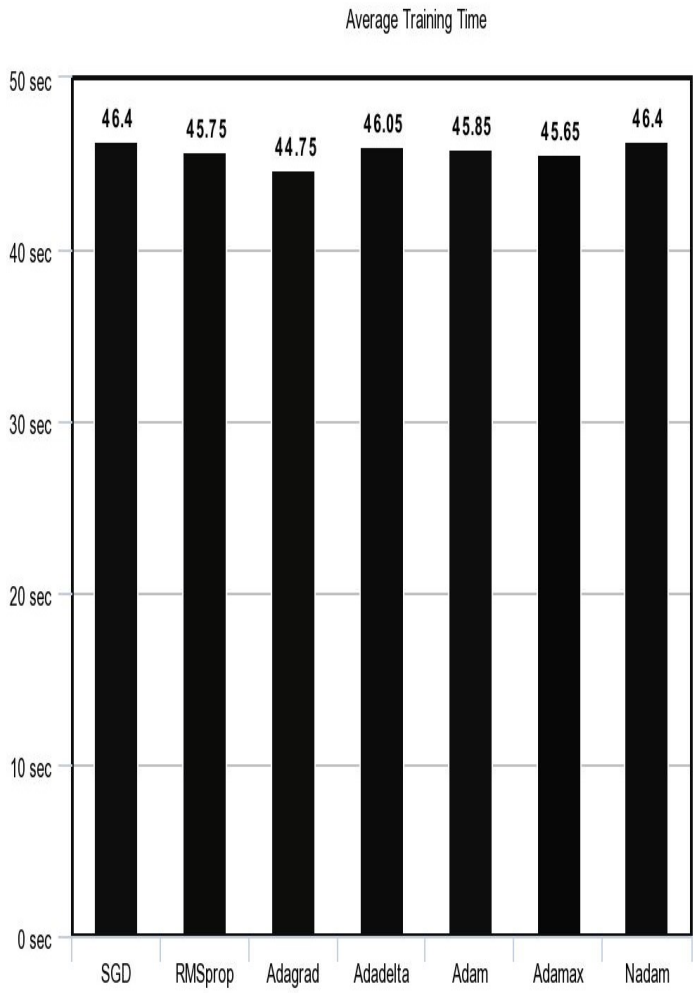
Şekil 6.2: Optimizasyon Algoritmaları Kayıp Grafikleri

Görüldüğü gibi kayıp grafikleri ve doğruluk grafikleri birbirinin zıttı gibidir. Çünkü kayıp fonksiyonları, modelin hata oranını ve beraberinde doğruluk değerini ölçmektedir. Kayıp fonksiyonunun amacı modelin gerçek değerden ne kadar uzak olduğunu belirlemektir. İyi bir modelde kayıp değeri olabildiğince düşük hatta sıfıra yakındır. Kayıp fonksiyonlarından, kayıp grafikleri oluşmaktadır.

Karşılaştırılan algoritmaların 20 iterasyon sonucu, doğruluk değerlerine ait çizgi grafiği Şekil 6.3'te ve eğitim süresine ait çizgi grafiği Şekil 6.4'teki gibidir.



Şekil 6.3: Algoritmaların Doğruluk Değerlerine Ait Çizgi Grafiği



Şekil 6.4: Algoritmaların Eğitim Süresine Ait Çizgi Grafiği

Sonuç olarak; rastgele çekilen plaka görüntülerinin eğitildiği bu Plaka Tanıma Sisteminde ön işleme yapılmadan Yapay Sinir Ağları ile 7 Optimizayon algoritması karşılşatırılarak sonuçlar ölçülmüştür. Uygulanan yöntemler ile hızlı bir şekilde tespit edilen plaka bölgesinden çıkarılan karakterler ile tanıma işlemi yapılmştır. Yedi algoritmada test edilen ve 7316 farklı örnek üzerinde doğrulanan bu çalışma, Adam algoritmasının doğruluğunun en yüksek olduğunu göstermiştir. En hızlı algoritma ise Adagrad algoritması olduğu ispatlanmıştır.

6.2 Öneriler

Geleneksel yöntemler ile yapılan plaka tanıma işlemlerinde çok fazla ön ileme yapılmakta idi. Bu nedenle bu yöntemler ile tamamlanan çalışmaların

performansı düşük olup işlem zamanları da artabilmekte idi. Evrişimli Sinir Ağları ile tamamlanan bu çalışmada, karşılaştırılan optimizasyon algoritmalarının eğitim süresi ve doğruluk oranı oldukça yüksektir. Modelin daha çok ve daha temiz görüntülerle eğitilmesi ile doğruluk oranı daha da artabilecek ve performans da aynı oranda artacaktır.

Sistemin daha çok veri setiyle beslenmesi ve girdi olarak verilen görüntülerin daha temiz ayarlanması ile belirtilen hız ve doğruluk değerleri daha da artırılabilir. Belirlenen 7316 farklı örnek üzerinde doğrulanan bu çalışma, Adam algoritmasının doğruluğunun en yüksek olduğunu göstermiştir. En hızlı algoritmanın ise Adagrad algoritması olduğu ispatlanmıştır.

KAYNAKLAR

Acharya, T., & Ray, A. K. (2005). *Image processing Principles and Applications.* New Jersey: John Wiley & Sons.

Allahverdi, N. 2002. Uzman Sistemler Bir Yapay Zeka Uygulaması, Atlas Yayın Dağıtım, İstanbul.

Alp, E. C. (2018). Makine Öğrenmesi Yöntemleriyle Akan Görüntülerden Otomatik Aktivite Sınıflandırma. Ankara Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, Yüksek Lisans Tezi.

Bakkaloğlu, "*Araç Plaka Tanıma Sistemi*", Yüksek Lisans Tezi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü, Konya, 2011.Çayıroğlu, İ. (2018). Görüntü İşleme. Karabük Üniversitesi, Mühendislik Fakültesi .

Bengio, Y., Boulanger-Lewandowski, N., & Pascanu, R. (2012). Advances in Optimizing Recurrent Networks. arXiv:1212.0901.

C.-T. Hsieh, Y.-S. Juan, and K.-M. Hung, "*Multiple license plate detection for complex background*" in Proc. Int Conf. AINA, 2005, *vol. 2, pp. 389–392.*Kim, P. (2017). *MATLAB Deep Learning: With Machine Learning, Neural Networks and Artificial Intelligence.* Apress.

Caner, H., 2006, FPGA Donanımı Üzerinde Araç Plaka Tanıma Sistemi, Yüksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 11-38.

Chang S.L., Chen L.S., Chung Y.C. and Chen S.W., 2004, Intelligent Transportation Systems, IEEE Transactions, Vol. 5, Issue 1, 42-53.

Çamaşırcıoğlu, E., 2007, Araç Plakası Algılama ve Tanıma, Yüksek Lisans Tezi, Ankara Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 42-67.

Çayıroğlu, İ. (2018). Görüntü İşleme. Karabük Üniversitesi, Mühendislik Fakültesi.

Çelik, U., 2003, Motorlu Araçlar İçin Plaka Tanıma Sistemi, Yüksek Lisans Tezi, Mustafa Kemal Üniversitesi Fen Bilimleri Enstitüsü, Hatay, 40-60.

D. Ballard and C. Brown *Computer Vision*, Prentice-Hall, 1982, Chap. 2.

Doğan, G. (2010). Yapay Sinir Ağları Kullanılarak Türkiye'deki Özel Bir Sigorta Şirketinde Portföy Değerlendirmesi. Hacettepe University, Ankara.

Fahmy, M., 1993, Computer Vision Application to Automatic Number-Plate Recognition, In Proceedings of 26th. International Symposium on Automotive Technology and Automation, Aachen-Germany, 625-633.

H. Chidiac, D. Ziou, "*Classification of Image Edges*",*Vision* Interface'99, Troise-Rivieres, Canada, *pp. 17-24*, 1999.

Hasan, M. u., Ullah, S., Khan, M. J., & Khurshid, K. (2019). Comparative Analysis of SVM, ANN and CNN for Classifying Vegetation Species Using Hyperspectral Thermal Infrared Data. *ISPRS*, 1861-1868.

Hauslen, R. A., 1977, The Promise Of Automatic Vehicle identification. IEEE Transactions on Vehicular Technology, VT-Vol. 26, 30-38.

Herbert Robbins and Sutton Monro *A Stochastic Approximation Method The Annals of Mathematical Statistics*, Vol. 22, No. 3. (Sep. 1951), pp. 400-407.

J. Kiefer and J. Wolfowitz *Stochastic Estimation of the Maximum of a Regression Function* Ann. Math. Statist. Volume 23, Number 3 (1952), 462-466.

J. Redmon. Darknet: Open source neural networks in c. http://pjreddie.com/darknet/, 2013-2016.

Kavuncu, S. K. (2018). Makine Öğrenmesi ve Derin Öğrenme: Nesne Tanıma Uygulaması. Kırıkkale Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, Yüksek Lisans Tezi.

Kim, P. (2017). *MATLAB Deep Learning: With Machine Learning, Neural Networks and Artificial Intelligence.* Apress.

Kingma, D. P., & Ba, J. L. (2017). Adam: A Method for Stochastic Optimization. *ArXiv e-prints*.

Köklü, M. (2014). Sınıflandırma Problemlerinde Kural Çıkarımı için Yeni Bir Yöntem Geliştirilmesi ve Uygulamaları. Selçuk Üniversitesi, Fen Bilimleri Enstitüsü, Doktora Tezi.

Kurt, F. (2018). Evrişimli Sinir Ağlarında Hiper Parametrelerin Etkisinin İncelenmesi. Hacettepe Üniversitesi, Fen Bilimleri Enstitüsü.

Kuş, Z. (2019). Mikrokanonikal Optimizasyon Algoritması ile Konvolüsyonel Sinir Ağlarında Hiper Parametrelerin Optimize Edilmesi. Fatih Sultan Mehmet Vakıf Üniversitesi, Yüksek Lisans Tezi.

L. Dlagnekov "*License Plate Detection Using AdaBoost*". La Jolla: Comput. Sci. Eng. Dept., Univ. California San Diego, Mar. 2004.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep Learning. *Nature*, 436-444.

Leon Bottou and Frank E. Curtis and Jorge Nocedal *Optimization Methods for Large-Scale Machine Learning*, Technical Report, arXiv:1606.04838.

Liu, L., Shen, C., & Hengel, A. v. (2015). The Treasure beneath Convolutional Layers: Cross-convolutional-layer Pooling for Image Classification. *IEEE Journals*, 4749-4757.

Muhammad Sarfraz, Mohammed Jameel Ahmed, Syed A. Ghazi, "*Saudi Arabian License Plate Recognition System*," gmag, *pp.36*, 2003 International Conference on Geometric Modeling and Graphics *(GMAG'03)*, 2003.

Özbay, S., 2006, Automatic Vehicle Identification by Plate Recognition, M.Sc. Thesis in Electrical& Electronics Engineering, Gaziantep University Graduate School of Natural & Applied Sciences, Gaziantep, 45-62.

Öztemel, E. (2006). *Yapay Sinir Ağları.* Papatya Yayıncılık.

Özveren, U. (2006). Pem Yakıt Hücrelerinin Yapay Sinir Ağları İle Modellenmesi. Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü.

Rahmon, G. (2018). Evaluation of Procedurally Generated Terrains via Artificial andConvolutional Neural Networks. İzmir Ekonomi Üniversitesi.

Ruder, S. (2016). An overview of gradient descent optimization. arXiv:1609.04747.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. 1986. Learning representations by back-propagating errors. Nature, 323:533—536.

S. Du, M. Ibrahim, M. Shehata, and W. Badawy. "*Automatic license plate recognition (ALPR): A state-of-the-art review*". Circuits and Systems for Video Technology, IEEE Trans. on, *23(2):311-325*, 2013. Köklü, M. (2014). Sınıflandırma Problemlerinde Kural Çıkarımı için Yeni Bir Yöntem Geliştirilmesi ve Uygulamaları. Selçuk Üniversitesi, Fen Bilimleri Enstitüsü, Doktora Tezi.

Sağıroğlu, Ş., Beşdok, E. ve Erler, M. 2003. Mühendislikte Yapay Zeka Uygulamaları-1 Yapay Sinir Ağları, Ufuk Yayıncılık, Kayseri.

Setchell, J., 1997, Application of Computer Vision to Road-Traffic Monitoring, PhD Thesis, University of Bristol.

Stoelhurst, H. J. , and Zandbergen, A. J. , 1990, The Development Of A Road Pricing System In The Netherlands, Traffic Engineering And Control,Vol. 31, 66-71.

Syed, Y.A. and Sarfraz, M., 2005, Information Visualisation, 2005. Proceedings. Ninth International Conference, Issue Date : 6-8 July 2005, 227-232.

Şeker, A. (2017). Derin Öğrenme Yöntemleri ve Uygulamaları Hakkında Bir İnceleme. Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Doktora Semineri, Bilgisayar Mühendisliği Anabilim Dalı.

Şeker, Diri, Balık / Gazi Mühendislik Bilimleri Dergisi 3 (3). (2017) 47-64.

T. Avery and G. Berlin *Fundamentals of Remote Sensing and Airphoto Interpretation*, Maxwell Macmillan International, 1985, Chap. 15.Erdil, F, & Elbaş, N. Ö. (2012). *Cerrahi Hastalıkları Hemşireliği.* Ankara: Aydoğdu Ofset.

Tekeli, K., & Aşlıyan, R. (2016). Çok Katmanlı Algılayıcı, K-NN ve C4.5 Metotlarıyla İstenmeyen E-postaların Tespiti. Adnan Menderes Üniversitesi.

Wei S. and Yanping B., 2009, Image and Signal Processing, 2009. CISP '09. 2nd International Congress, Tianjin, 1-4.

Wei W., Li Y., Wang M. and Huang Z., 2001, Neural Networks for Signal Processing XI, 2001. Proceedings of the 2001 IEEE Signal Processing Society Workshop, North Falmouth, 529 - 538. Shapiro, Vladimir, Gluhchev, Georgi, Dimov, Dimo, *"Towards a Multinational Car License Plate Recognition System" MVA (17), No. 3, pp. 173-183*, 2006.

Xu, B. W., Chen, T., & Li, M. (2015). Empirical Evaluation of Rectified Activations in Convolution Network. arXiv:1505.00853v2.

Yang H., Xu L., and Shi L., 2007, Information Technologies and Applications in Education, 2007. ISITAE '07. First IEEE International Symposium, Kunming, 602-605.

Yavuz, S., & Deveci, M. (2012). İstatiksel Normalizasyon Tekniklerinin Yapay Sinir Ağın Performansına Etkisi. *Erciyes Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 167-187.

Yazan, E., & Talu, F. M. (2017). *Stokastik Dereceli Alçalma Yöntemi Temelli Optimizasyon Tekniklerinin Karşılaştırılması.*

Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., & Lipson, H. (2015). Understanding Neural Networks Through Deep Visualization. *Deep Learning Workshop, 31 st International.* Lille.

İnternet Kaynakları:

Keras Keras Optimizer https://keras.io/api/optimizers/ Erişim tarihi: 02.09.2019

URL-1 Macbook Pro Özellikler https://www.apple.com/tr/macbook-pro-16/specs/ adresinden alındı. Erişim Tarihi: 18.06.2020

URL-2 *Sinir Ağları* https://medium.com/@ayyucekizrak/%C5%9Fu-kara-kutuyu-a%C3%A7alim-yapay-sinir-a%C4%9Flar%C4%B1-7b65c6a5264a adresinden alındı. Erişim Tarihi: 18.06.2020

URL-3 Evrişimli *Sinir Ağı Mimarisi* https://stanford.edu/~shervine/l/tr/teaching/cs-230/cheatsheet-convolutional-neural-networks adresinden alındı. Erişim Tarihi: 22.06.2020

URL-4 *Exploring Convolutional Neural Networks (CNNs) from an iOS Developer's Perspective* https://heartbeat.fritz.ai/exploring-convolutional-neural-networks-cnns-from-an-ios-developers-perspective-162664130d5b adresinden alındı. Erişim Tarihi: 22.06.2020

URL-5 *Pooling Layer* https://harangdev.github.io/deep-learning/convolutional-neural-networks/24/ adresinden alındı. Erişim Tarihi: 26.06.2020

URL-6 Tam Bağlantı Katmanı https://stanford.edu/~shervine/l/tr/teaching/cs-230/cheatsheet-convolutional-neural-networks adresinden alındı. Erişim Tarihi: 24.06.2020

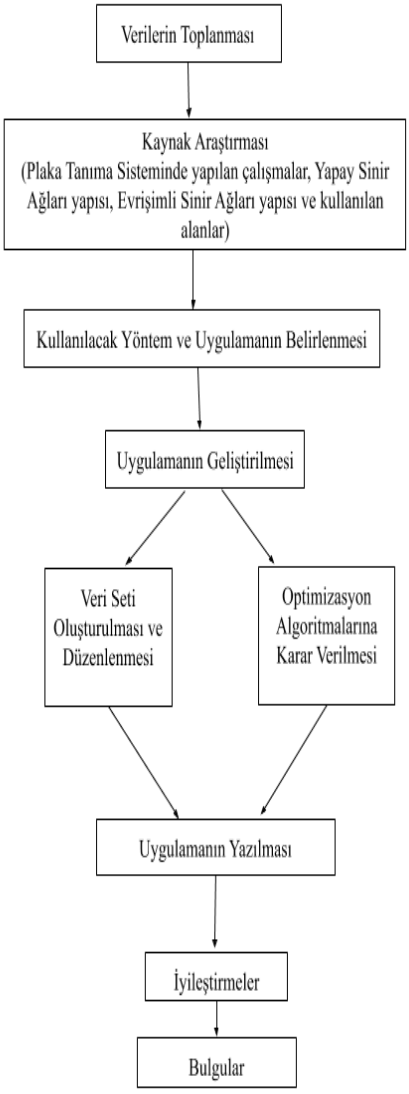
URL-2 *Sinir Ağları* https://medium.com/@ayyucekizrak/%C5%9Fu-kara-kutuyu-a%C3%A7alim-yapay-sinir-a%C4%9Flar%C4%B1-7b65c6a5264a adresinden alındı. Erişim Tarihi: 18.06.2020

Hijazi, S., Kumar, R., & Chris, R. (2015). *Using Convolutional Neural Networks for Image Recognition.* https://ip.cadence.com/uploads/901/cnn_wp-pdf adresinden alındı. Erişim Tarihi: 03.09.2019

EKLER

EK A : Şekiller
EK B : Kodlar

EK A: Şekiller



Şekil A.1 : Tez Çalışması Akış Diyagramı

EK B: Kodlar

Optimizasyon Karşılaştırılması için Kullanılan Python Dilinde Yazılmış Kod

```
# import libraries

import pandas as pd
import numpy as np
import cv2
import os
import pickle
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, MaxPooling2D, Dropout,
Flatten
from matplotlib import pyplot as plt

"""from sklearn.preprocessing import OneHotEncoder
from keras.models import Sequential
from keras.layers import Dense, Conv2D, MaxPooling2D, Dropout, Flatten
from sklearn.model_selection import train_test_split
from keras.utils.vis_utils import plot_model
import matplotlib.pyplot as plt"""

# Load dataset

# Create dictionary for alphabets and related numbers
alphabets_dic = {0: 'A', 1: 'B', 2: 'C', 3: 'D', 4: 'E', 5: 'F', 6: 'G', 7: 'H', 8: 'I', 9: 'J',
          10: 'K', 11: 'L', 12: 'M', 13: 'N', 14: 'O', 15: 'P', 16: 'Q', 17: 'R', 18: 'S', 19: 'T',
          20: 'U', 21: 'V', 22: 'W', 23: 'X', 24: 'Y', 25: 'Z', 26: '0', 27: '1', 28: '2', 29:'3',
          30: '4', 31: '5', 32: '6', 33: '7', 34: '8', 35: '9'}

alphabets =
['0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R',
'S','T','U','V','W','X','Y','Z']
dataset_classes = []

for cls in alphabets:
   dataset_classes.append([cls])

# Load old dataset

d = open("data.pickle","rb")
l = open("labels.pickle","rb")
```

```
data = pickle.load(d)
labels = pickle.load(l)

label_list = []
for l in labels:
    label_list.append([l])

# One hot encoding format for output

ohe = OneHotEncoder(handle_unknown='ignore', categorical_features=None)
ohe.fit(dataset_classes)
labels_ohe = ohe.transform(label_list).toarray()

data = np.array(data)
labels = np.array(labels)

# Split the data
X_train, X_test, y_train, y_test = train_test_split(data, labels_ohe, test_size=0.20,
random_state=42)

X_train = X_train.reshape(29260,28,28,1)
X_test = X_test.reshape(7316,28,28,1)

print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)

# CNN model

model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same', activation='relu',
input_shape=(28,28,1)))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(64, (3, 3), padding='same', activation='relu'))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(64, (3, 3), padding='same', activation='relu'))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
```

```
model.add(Dense(36, activation='softmax'))

print(model.summary())

model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])
#model.compile(loss='categorical_crossentropy', optimizer='sgd',
metrics=['accuracy'])
#model.compile(loss='categorical_crossentropy', optimizer='rmsprop',
metrics=['accuracy'])
#model.compile(loss='categorical_crossentropy', optimizer='adagrad',
metrics=['accuracy'])
#model.compile(loss='categorical_crossentropy', optimizer='adadelta',
metrics=['accuracy'])
#model.compile(loss='categorical_crossentropy', optimizer='adamax',
metrics=['accuracy'])
#model.compile(loss='categorical_crossentropy', optimizer='nadam',
metrics=['accuracy'])
history = model.fit(X_train, y_train,validation_data=(X_test, y_test), epochs=20,
batch_size=64)

model.save('cnn_classifier.h5')
# Visualization
plt.figure(figsize=[8, 6])
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.figure(figsize=[8, 6])
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')

plt.show()
```

ÖZGEÇMİŞ

Ad-Soyad **:** Isa JAVADOV
Doğum Tarihi **:** 17.12.1995
Doğum Yeri **:** Rusya,Moskova
E-posta **:** isa.cavadov@gmail.com

Öğrenim Durumu

- **Lisans** **:** Bakü Devlet Üniversitesi – Bilgisayar Mühensiliği

- **Yüksek Lisans** **:** İstanbul Aydın Üniversitesi – Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği

Yabancı Diller

- İngilizce
- Rusça