EVRİŞİMLİ SİNİR AĞLARI İÇİN REGRESYON TEMELLİ HİPER PARAMETRE OPTİMİZASYONU

YÜKSEK LİSANS TEZİ

Gözde AKBULUT ÖZ

ESKİŞEHİR 2023

EVRİŞİMLİ SİNİR AĞLARI İÇİN REGRESYON TEMELLİ HİPER PARAMETRE OPTİMİZASYONU

Gözde AKBULUT ÖZ

Yüksek Lisans Tezi

Endüstri Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Emre ÇİMEN

Eskişehir

Eskişehir Teknik Üniversitesi

Lisansüstü Eğitim Enstitüsü

Ağustos 2023

JÜRİ VE ENSTİTÜ ONAYI

Gözde AKBULUT ÖZ'ün EVRİŞİMLİ SİNİR AĞLARI İÇİN REGRESYON TEMELLİ HİPER PARAMETRE OPTİMİZASYONU başlıklı çalışması 17/08/2023 tarihinde aşağıdaki jüri tarafından değerlendirilerek "Eskişehir Teknik Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliği"nin ilgili maddeleri uyarınca, Endüstri Mühendisliği Anabilim dalında Yüksek Lisans Tezi olarak kabul edilmiştir.

Jüri Üyeleri	**Unvan Adı Soyadı**	**İmza**
Üye	: Doç. Dr. Emre ÇİMEN	
Üye	: Dr. Öğr. Üyesi Zeliha ERGÜL AYDIN	
Üye	: Dr. Öğr. Üyesi Erdener ÖZÇETİN	

Prof. Dr. Semra KURAMA

Lisansüstü Eğitim Enstitüsü Müdürü

17/08/2023

DANIŞMAN ONAYI

Danışmanlığını yürüttüğüm Yüksek Lisans öğrencisi Gözde AKBULUT ÖZ, EVRİŞİMLİ SİNİR AĞLARI İÇİN REGRESYON TEMELLİ HİPER PARAMETRE OPTİMİZASYONU başlıklı tez çalışmasını tamamlamıştır. Hazırlamış olduğu tez tarafımca incelenmiş ve öğrencinin tez savunma sınavına alınması bilimsel ve etik açıdan uygun görülmüştür.

Tez Danışmanı

Doç. Dr. Emre ÇİMEN

ÖZET

EVRİŞİMLİ SİNİR AĞLARI İÇİN REGRESYON TEMELLİ HİPER PARAMETRE OPTİMİZASYONU

Gözde AKBULUT ÖZ

Endüstri Mühendisliği Anabilim Dalı

Eskişehir Teknik Üniversitesi, Lisansüstü Eğitim Enstitüsü, Ağustos 2023

Danışman: Doç. Dr. Emre ÇİMEN

Makine Öğrenmesi problemlerinde modelin performansını etkileyen, ayarlanabilir parametreler olan hiper parametrelerin kullanıcı tarafından tayin edilmesi beklenir. Fakat modelin performansını, eğitim ve test başarısını etkileyen bu hiper parametrelerin seçimini kullanıcıya bırakmak geniş seçim uzayının olduğu ortamlarda büyük bir risktir. Bu duruma karşı hiper parametre optimizasyonu için geliştirilen pek çok yöntem vardır. Kimisi zaman, kimisi de doğruluk ölçütlerinde avantajlı durumda iken her veri kümesine ya da probleme göre farklı yöntemler kullanmak gerekebilir. Ayrıca iyi sonuçlar verdiği bilinen yöntemler ücretli çözümler sunması sebebiyle de tercih edilemeyebilir.

Bu çalışmada zaman ve doğruluk açısından ön planda olabilecek, farklı veri kümelerinde de kabul görebilecek sonuçlar veren bir hiper parametre optimizasyon yöntemi geliştirilmiştir. Yalnızca makine öğrenmesi problemleri ile çalışan araştırmacılar için değil optimize edilmesi gereken parametrelere sahip tüm problem çözücüler için kullanılabilir bir yöntem olması amaçlanmıştır. Geliştirilen bu yöntem, kabul görmüş rastgele arama ve Bayesian optimizasyon adlı iki farklı yöntem ile karşılaştırılmış olup ayrıca CIFAR-10, FASHIONMNIST ve MNIST adlı 3 farklı veri kümesi kullanılarak test edilmiştir. Rastgele arama ve Bayesian optimizasyonu karşısında avantajlı durumda olduğu sonuçlar elde edilmiştir.

Anahtar Sözcükler: Makine Öğrenmesi, Hiper parametre, Rastgele Arama, Bayesian Optimizasyon

ABSTRACT

REGRESSION BASED HYPER PARAMETER OPTIMIZATION FOR CONVERSIONAL NEURAL NETWORKS

Gözde AKBULUT ÖZ

Department of Industrial Engineering

Eskişehir Technical University, Institute of Graduate Programs, August 2023

Supervisor: Assoc. Prof. Dr. Emre ÇİMEN

In Machine Learning problems, hyperparameters are adjustable parameters that affect the performance of the model. It is expected that users determine these hyperparameters. However, leaving the selection of these hyperparameters, which impact the performance of the model in terms of training and testing success, to the user poses a significant risk, especially in environments with a wide choice space. To address this issue, numerous methods have been developed for hyperparameter optimization. Some of these methods excel in terms of time, while others excel in accuracy criteria. Different methods may need to be employed for each dataset or problem.

Additionally, well-performing methods might not be preferred due to their paid solutions. In this study, a hyperparameter optimization method has been developed that can yield results emphasizing both time and accuracy and is applicable to different datasets. The aim is to provide a method not only for researchers working solely on machine learning problems but also for all problem solvers with parameters that need optimization. This developed method has been compared with two established methods: random search and Bayesian optimization. Furthermore, it has been tested on three different datasets CIFAR-10, FASHIONMNIST, and MNIST. The results indicate its superiority over random search and Bayesian optimization.

Keywords: Machine Learning, Hyper Parameter, Random Search, Bayesian Optimization .

TEŞEKKÜR

Çalışmalarımda bana yol gösteren, desteğini, emeğini, bilgi ve tecrübesini hiçbir zaman esirgemeyen, öğrencisi olmaktan her zaman gurur duyduğum ve duyacağım tez danışmanım sayın Doç. Dr. Emre ÇİMEN'e;

Akademik çalışmalarım sırasında beni yüreklendiren ve destek olan Eczacıbaşı Yapı Gereçleri Bilgi Teknolojileri Müdürü sayın Yüksek Endüstri Mühendisi Mehmet Tahir ÇİFTÇİ'ye;

Hayatım boyunca beni her alanda destekleyen, bu günlere gelmem için emek veren ve fedakarlıklar gösteren annem Ayşen AKBULUT ve babam Süleyman AKBULUT'a, mesleki ve akademik eğitimim için bana cesaret veren eşim Ersin ÖZ'e;

Teşekkür eder, saygı ve sevgilerimi sunarım.

Gözde AKBULUT ÖZ

17/08/2023

ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ

Bu tezin bana ait, özgün bir çalışma olduğunu; çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarında bilimsel etik ve kurallara uygun davrandığımı; bu çalışma kapsamında elde edilen tüm veri ve bilgiler için kaynak gösterdiğimi ve bu kaynaklara kaynakçada yer verdiğimi; bu çalışmanın Eskişehir Teknik Üniversitesi tarafından kullanılan "bilimsel intihal tespit programı"yla tarandığını ve hiçbir şekilde "intihal içermediğini" beyan ederim. Herhangi bir zamanda, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçları kabul ettiğimi bildiririm.

Gözde AKBULUT ÖZ

İÇİNDEKİLER

Sayfa

TABLOLAR DİZİNİ

Sayfa

ŞEKİLLER DİZİNİ

Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

ML	:	Makine Öğrenmesi (Machine Learning)
CNN	:	Evrişimli Sinir Ağı (Convolutional Neural Network)
'n_filter	:	Filtre sayısı
filterSize	:	Filtre boyutu
poolSize	:	Havuzlama boyutu
DenseSize	:	Yoğun katmanlardaki nöron sayısını ifade eder
Beta1 ve Beta2	:	Adam algoritmasında öğrenme oranı için kullanılır.
LearningRate	:	Öğrenme oranı
BatchSize	:	Eğitim döngüsündeki veri örnekleri sayısı
Epochs	:	Eğitim döngüsündeki adım sayısı
nofconv	:	CNN’de evrişimli katmanların sayısı
numberofHiderlayer	:	Gizli katmanların sayısı

1. GİRİŞ

Günümüzde, veri bilimi ve yapay zeka alanlarında hızla ilerleyen teknolojik gelişmeler, makine öğrenmesi yöntemlerine olan talebi artırmış ve bu alanda yapılan araştırmaların önemini vurgulamıştır. Makine öğrenmesi, algoritma ve istatistiksel modelleme tekniklerinin kullanımıyla bilgisayar sistemlerinin verilerden öğrenmesine ve deneyim kazanmasına olanak tanıyan önemli bir yapay zeka alt alanıdır. Bu alandaki başarılar, birçok uygulama alanında kullanılan akıllı sistemlerin geliştirilmesine ve insan hayatını kolaylaştıracak çözümlerin elde edilmesine olanak sağlamıştır.

Makine öğrenmesi algoritmaları;

Gözetimli Öğrenme (Supervised Learning): Eğitim verilerinde giriş ve çıkışlarla etiketlenmiş örnekler kullanarak çalışır. Algoritma, eğitim verilerine dayalı olarak bir girişe uygun çıkışları tahmin etmeyi öğrenir. Örnek olarak, ev fiyatlarını tahmin etmek veya e-postaları spam ve spam olmayan olarak sınıflandırmak gibi görevler gözetimli öğrenme ile çözülebilir.

Gözetimsiz Öğrenme (Unsupervised Learning): Eğitim verilerinde etiketler kullanılmadan sadece giriş verileri kullanılarak çalışır. Algoritma, veri kümesindeki yapıları, desenleri veya benzerlikleri tespit etmeyi amaçlar. Kümeleme (clustering) ve boyut indirgeme (dimensionality reduction) gibi görevler için gözetimsiz öğrenme kullanılabilir.

Pekiştirmeli Öğrenme (Reinforcement Learning): Bir ajanın, çevresiyle etkileşime geçerek deneyim yoluyla öğrendiği bir öğrenme türüdür. Ajan, çevreyle etkileşim sonucunda aldığı ödüller ve cezalar üzerinden belirli bir hedefi başarmayı öğrenir. Oyun oynamak, robot kontrolü ve otomatik arabalar gibi alanlarda pekiştirmeli öğrenme kullanılabilir.

Yarı Gözetimli Öğrenme (Semi-Supervised Learning): Gözetimli ve gözetimsiz öğrenme yöntemlerini birleştiren bir yaklaşımdır. Sınıflandırma veya regresyon gibi gözetimli görevlerde, az sayıda etiketli veri ile birlikte çok miktarda etiketsiz veriyi kullanarak modelin performansını artırmayı amaçlar.

Derin Öğrenme (Deep Learning): Derin sinir ağları aracılığıyla temsil edilen karmaşık ve hiyerarşik modeller kullanarak öğrenme sürecini gerçekleştiren bir alt kümedir. Genellikle büyük veri kümeleri ve yüksek hesaplama gücü gerektiren görevlerde etkili olabilir. Ses ve görüntü tanıma, doğal dil işleme ve oyun stratejileri gibi alanlarda güçlü öğrenme sıkça kullanılır.

Makine öğrenmesi, günümüzde birçok farklı alanda uygulamaları olan önemli bir yapay zeka disiplinidir ve sürekli olarak gelişmeye ve ilerlemeye devam etmektedir.

Makine öğrenmesi genellikle şu adımları izler:

Veri toplama: Öncelikle, ilgili ve temsilci bir veri seti toplanır. Veri, öğrenme modelini eğitmek ve değerlendirmek için kullanılır.

Veri ön işleme: Veri, düzenlenir, eksik değerler doldurulur, aykırı değerler düzeltilir ve veri boyutu azaltılır (gerekiyorsa).

Model seçimi: Belirli bir probleme en uygun algoritma veya model seçilir. Örneğin, sınıflandırma problemleri için destek vektör makineleri, karar ağaçları veya yapay sinir ağları gibi farklı modeller kullanılabilir.

Eğitim: Model, veri setini kullanarak öğrenmeye başlar. Bu süreçte, model, girdi ve çıktı arasındaki ilişkiyi anlamaya çalışır ve içindeki parametreleri ayarlar.

Değerlendirme: Eğitilen modelin performansı, ayrı bir değerlendirme veri seti kullanılarak test edilir. Bu, modelin ne kadar iyi öğrendiğini ve genelleştirme yeteneğini değerlendirmek için yapılır.

Tahmin: Model, yeni verilerle kullanılmak üzere hazır hale gelir ve tahminler yapabilir.

Makine öğrenmesi modelleri, birçok farklı hiper parametre (hiper parametreler, modelin yapılandırılmasını belirleyen ve modelin performansını etkileyen ayarlanabilir parametrelerdir) ile tanımlanır. Bu hiper parametreler, modelin doğruluğu, hızı ve diğer performans metrikleri üzerinde doğrudan etkiye sahiptir. Bu nedenle, uygun hiper parametre değerlerinin seçilmesi, makine öğrenmesi modelinin başarısını ve performansını büyük ölçüde etkileyebilir.

Hiper parametre optimizasyonu, makine öğrenmesi modelinin hiper parametrelerinin en uygun değerlerini bulma sürecidir. Bu optimizasyon süreci, deneme-yanılma yöntemleri ile gerçekleştirildiğinde zaman alıcı ve verimsiz olabilir. Bu nedenle, daha akıllı ve otomatik hiper parametre optimizasyon teknikleri geliştirilmiştir. Bu teknikler, genellikle mevcut kaynakları en iyi şekilde kullanarak, model performansını iyileştirmek için hiper parametre arama uzayında etkili bir şekilde gezinmeyi amaçlar.

Bu tez, makine öğrenmesi ve hiper parametre optimizasyonu alanlarında mevcut literatürü inceleyerek, çeşitli hiper parametre optimizasyon tekniklerinin ve algoritmalarının karşılaştırmalı analizini sunmayı amaçlamaktadır. Aynı zamanda, bu alanda yapılan araştırmalara katkı sağlayarak, makine öğrenmesi modellerinin

performansını artırmak için daha etkili ve verimli hiper parametre optimizasyon yöntemleri sunmayı hedeflemektedir. Eğitim sürelerinin uzunluğu, arama uzaylarının genişliği sebebiyle oluşan maliyetleri azaltma amacı bu çalışmanın motivasyonunu oluşturmaktadır.

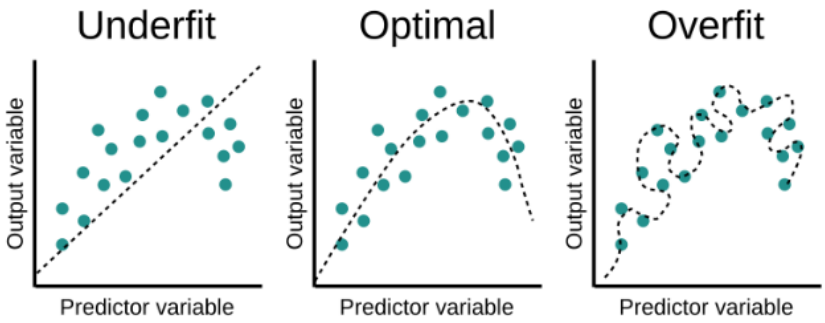
Bu tezin organizasyonunda öncelikle literatür araştırmaları yer almaktadır. Ağırlıklı olarak makine öğrenmesi algoritmaları, hiper parametre optimizasyonu yöntemleri gibi konuları ele alan çalışmalar araştırılmıştır. Sonraki adımda çalışmanın amacı açıklanacaktır. Terimler ve yöntemler tanıtıldıktan sonra geliştirilen yöntem adım adım anlatılacaktır. Sayısal ve sözel sonuçların ardından tartışma bölümüne yer verilecektir.

2. LİTERATÜR ARAŞTIRMASI

Makine öğrenmesi ve hiper parametre optimizasyonu, son yıllarda bilgisayar bilimleri ve yapay zeka alanlarında büyük ilgi çeken önemli konulardır. Makine öğrenmesi, bilgisayar sistemlerinin deneyim ve veri yoluyla otomatik olarak gelişebilme yeteneğini inceleyen ve bu yeteneği sağlayan alandır. Bu alan, giderek artan veri miktarı ve hesaplama gücü ile birlikte büyük ilerlemeler kaydetmiş ve pek çok uygulama alanında çığır açmıştır. Makine öğrenimi modelleri, veriye dayalı deneyimler yoluyla öğrenerek, örüntüleri yakalayabilir, tahminlerde bulunabilir ve kararlar alabilir hale gelmiştir. Ancak, bu modellerin en iyi performansı göstermesi, belirli hiper parametrelerin dikkatli bir şekilde ayarlanmasını gerektirir. Hiper parametreler, modelin yapısı, eğitim süreci gibi alanları etkileyen ayarlanabilir parametrelerdir. Doğru hiper parametre değerlerinin bulunması, bir modelin istenilen performansı elde etmesi için kritik öneme sahiptir. Bu nedenle, hiper parametre optimizasyonu, makine öğrenimi alanının temel bir yönü haline gelmiştir. Bu bölüm, makine öğrenimi modellerinin etkili bir şekilde geliştirilmesi için kullanılan hiper parametre optimizasyon yöntemlerini, bu yöntemlerin avantajlarını ve zorluklarını inceleyerek, mevcut literatürdeki önemli bulguları sunmayı amaçlamaktadır.

Makine öğrenmesi, algoritma ve istatistiksel modelleme tekniklerini kullanarak bilgisayar sistemlerinin verilerden öğrenmesini sağlayan bir yapay zeka alt alanıdır. Arthur Samuel tarafından 1959'da “bilgisayarlar düşünemez, ancak öğrenebilirler” fikriyle ortaya atılan makine öğrenmesi, günümüzde çeşitli alanlarda uygulanarak büyük başarılar elde etmiştir. Özellikle derin öğrenme ve destek vektör makineleri gibi güçlü algoritmalar, görüntü işlemeden doğal dil işlemeye kadar birçok uygulamada olağanüstü sonuçlar vermiştir.

Ancak, makine öğrenmesi modellerinin performansını etkileyen kritik bir faktör, hiper parametrelerin seçimidir. Hiper parametreler, modelin başarı ve genelleme yeteneği üzerinde doğrudan etkisi olan ayarlanabilir parametrelerdir. Literatürde yapılan araştırmalar, hiper parametrelerin yanlış seçilmesinin Şekil 2.1’deki gibi modelin aşırı uymasına (overfitting) veya yetersiz uymasına (underfitting) neden olabileceğini göstermektedir. Bu nedenle, hiper parametre optimizasyonu, makine öğrenmesi modelinin performansını artırmak için kritik bir adımdır.



a) b) c)

Şekil 2.1: *Yetersiz uyum, optimal uyum ve aşırı uyum grafikleri*

Şekil 2.1 a)'da Underfitting (Alt Uyum) grafiği görülmektedir. Bu durum, makine öğrenme modelinin verilere yetersiz şekilde uyum sağladığı bir durumu ifade eder. Model, veri setindeki desenleri ve ilişkileri yeterince yakalayamaz. Sonuç olarak, eğitim verilerine iyi uymaz ve hem eğitim hem de test verilerinde düşük performans gösterebilir. Genelde modelin basit olduğu veya eğitim süresinin yetersiz olduğu durumlar alt uyum örnekleridir.

Şekil 2.1 b)'de Optimal fitting (Uygun Uyum) grafiği görülmektedir. Bu durum, makine öğrenme modelinin veri setindeki desenleri doğru şekilde yakaladığı ve iyi bir denge kurduğu durumu ifade eder. Model, hem eğitim hem de test verilerinde iyi performans gösterir. Model, veri setindeki gerçek ilişkileri genel olarak doğru bir şekilde anlar ve bu nedenle yeni verilere de iyi bir şekilde genelleme yapabilir.

Şekil 2.1 c)'de Overfitting (Aşırı Uyum) grafiği görülmektedir. Bu durumda ise, makine öğrenme modeli, eğitim verilerine aşırı derecede uyum sağlar ve bu da modeli veri setindeki rastgele gürültüye bile tepki verir hale getirir. Sonuç olarak, model eğitim verilerinde yüksek performans gösterebilir ancak yeni ve görmediği verilere uygulandığında kötü performans sergiler. Overfitting genellikle modelin çok karmaşık olduğu veya eğitim süresinin fazla olduğu durumlarla ilişkilidir.

Özetle, makine öğrenmesinde alt uyum, uygun uyum ve aşırı uyum kavramları, bir modelin veriye ne kadar iyi uyum sağladığını ve bu uyumun ne kadar iyi bir şekilde genelleme yapabildiğini açıklar. Optimal uyum, dengeyi bulduğumuz ve modelin veri setindeki gerçek ilişkileri en iyi şekilde anladığı durumdur.

Hiper parametre optimizasyonu, literatürde çeşitli yaklaşımlarla ele alınmıştır. ızgara arama (grid search), rastgele arama (random search), ve Bayesian optimizasyon gibi temel yöntemlerden, genetik algoritmalar, parçacık sürü optimizasyonu tekniklerine kadar çeşitli optimizasyon teknikleri literatürde yer almaktadır. Bu çalışmaların birçoğu, farklı makine öğrenmesi modelleri ve veri setleri üzerinde hiper parametre optimizasyonunun etkinliğini değerlendirmiştir.

Ayrıca, literatürde makine öğrenmesi ve hiper parametre optimizasyonu konusunda yapılan bazı çalışmalar, derin öğrenme modellerinin ve büyük ölçekli veri setlerinin hiper parametre optimizasyon süreçlerini karmaşıklaştırdığını vurgulamaktadır. Bu nedenle, bazı araştırmalar, bu tür modeller ve veri setleri için daha özel ve etkili optimizasyon yöntemleri geliştirmeye odaklanmıştır.

Bengio ve ekibi, "Random Search for Hyper-Parameter Optimization" adlı makalede, hiper parametre optimizasyonu için rastgele arama (random search) yöntemini önermişlerdir [1]. Bu çalışmada, ızgara arama (grid search) yönteminin dezavantajlarından kaçınmak için rastgele arama yönteminin kullanılması önerilmiştir. Yapılan deneylerde, rastgele arama yönteminin ızgara arama yöntemine kıyasla daha verimli ve etkili olduğu gösterilmiştir.

Bir diğer önemli yöntem olan Bayesian Optimization, makine öğrenimi modellerinin hiper parametre optimizasyonunda kullanılan yaygın bir yöntemdir. Snoek ve ekibi, "Practical Bayesian Optimization of Machine Learning Algorithms" makalesinde, Bayesian Optimization'ın makine öğrenimi algoritmalarının hiper parametrelerini optimize etmek için etkili bir şekilde kullanılabileceğini göstermişlerdir [2]. Bu çalışmada, Gaussian Process modelleri kullanılarak hiper parametre alanındaki olasılık dağılımı tahmin edilmiş ve en iyi hiper parametre değerleri elde edilmiştir.

Evrişimli Sinir Ağı (CNN), derin öğrenme alanındaki en önemli ağlardan biridir. CNN, bilgisayarlı görme ve doğal dil işleme gibi birçok alanda etkileyici başarılar elde ettiği için, son birkaç yılda endüstri ve akademiden büyük ilgi görmüştür. [3]

Genetik Algoritmalar, evrimsel stratejilere dayalı bir optimizasyon yöntemi olarak hiper parametre optimizasyonunda yaygın olarak kullanılan başka bir yaklaşımdır. Pelikan ve Goldberg, "BOA: The Bayesian Optimization Algorithm" adlı makalede, Bayesian Optimization algoritmasını kullanarak hiper parametreleri optimize etmek için

bir Genetik Algoritma geliştirmişlerdir [4]. Bu çalışmada, BOA algoritması ile modelin hiper parametrelerinin daha verimli bir şekilde arandığı ve etkili sonuçlar elde edildiği belirtilmiştir.

Mashold ve Graville'in "A Genetic Algorithm for Function Optimization: A Matlab Implementation" adlı çalışmalarında, genetik algoritmaların işlev optimizasyonunda kullanılması üzerine bir Matlab uygulaması sunulmuştur [5].

Katharina Eggensperger ve ekibi "SMAC3: A Versatile Bayesian Optimization Package For Hyperparameter Optimization" adlı çalışmalarında çeşitli hiper parametre optimizasyon yöntemlerini gözden geçirip performanslarını karşılaştırmıştır [6].

Son yıllarda, Parçacık Sürü Optimizasyonu (Particle Swarm Optimization - PSO) da hiper parametre optimizasyonu için ilgi çekici bir yöntem olarak öne çıkmaktadır. Eberhart ve Kennedy, "A New Optimizer Using Particle Swarm Theory" adlı makalede, PSO algoritmasını tanıtmışlardır [7]. Bu çalışmada, parçacıkların hiper parametre alanında gezinerek optimum noktaları keşfettiği ve bu yöntemin diğer optimizasyon algoritmalarına kıyasla daha hızlı sonuçlar elde ettiği gösterilmiştir.

"Hyperband: A novel bandit-based approach to hyperparameter optimization. Journal of Machine Learning Research" adlı çalışma "Hyperband" adlı bir algoritma sunar, bu algoritma hızlı ve verimli bir şekilde hiper parametre uzayını araştırmak için bandit algoritmalarını kullanır [8].

Bergestra ve ekibi "Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures." adlı çalışmalarında hiper parametre optimizasyonunu görüntü işleme mimarileri için yüzlerce boyutta nasıl gerçekleştirdiğine dair bir çalışma yürütmüştür [9].

"Metaheuristics in combinatorial optimization: Overview and conceptual comparison." Adlı çalışmada C. Blum ve ekibi metaheuristik optimizasyon yöntemlerini (örneğin, genetik algoritmalar, simüle edilen tavlama, parçacık sürü optimizasyonu, tabu arama vb.) ayrıntılı bir şekilde tanıtmıştır [10].

PSO, doğal dünya örneklerinden esinlenen bir optimizasyon tekniğidir. Bu algoritma, bir grup "partikül" adı verilen bireylerin belirli bir problemde gezinirken en

iyi çözüme yaklaşmaya çalıştığı bir popülasyon tabanlı yaklaşımdır. Partiküller, geçmiş deneyimlerine dayanarak ve en iyi buldukları çözümlere yönelerek hareket ederler.

"Eğitimli Sürü Optimizasyonu" olarak da adlandırılan PSO, genellikle nümerik bir problemin çözümüne odaklanır. Ancak, "A modified particle swarm optimizer" adlı çalışma, PSO algoritmasının geleneksel versiyonundan farklı olarak nasıl modifiye edildiğini, hangi değişikliklerin yapıldığını ve bu değişikliklerin optimizasyon performansına nasıl etki ettiğini muhtemelen ele almaktadır [11].

"Genetic Programming: On the Programming of Computers by Means of Natural Selection" adlı makalede, genetik algoritmaların programlama dünyasına uyarlanmasının avantajları ve yöntemin nasıl işlediği anlatılmaktadır. Ayrıca, genetik programlamanın potansiyel uygulama alanları ve sınırlamaları da tartışılır. Genetik programlama, bir tür evrimsel algoritma olan genetik algoritmaların bir türevidir. Bu yaklaşım, bilgisayar programlarının genetik operatörler (çaprazlama, mutasyon vb.) kullanılarak evrimleştirilmesini amaçlar. Başlangıçta rastgele yaratılan program popülasyonu zamanla daha iyi sonuçlar üretebilen programlara evrimleşebilir [12].

Optimization by Simulated Annealing adlı çalışma, Tavlama Benzetimi (Simulated Annealing) yönteminin temel ilkesini açıklar. Bu ilke, ısıl işlem sırasında bir maddenin sıcaklıkla nasıl düzenlendiğini taklit eder. Başlangıçta yüksek enerjili durumları kabul ederken zamanla daha düşük enerjili durumlara doğru ilerler [13].

"Stochastic Gradient Descent (SGD) Tricks" adlı bir çalışma, genel olarak stokastik gradyan iniş algoritmasının daha etkili kullanımını ve hızlandırılmasını amaçlayan yöntemler, ipuçları ve önerileri içerir. SGD, makine öğrenimi ve derin öğrenme gibi alanlarda çok yaygın olarak kullanılan bir optimizasyon algoritmasıdır. Ancak SGD bazen yavaş veya istikrarsız olabilir ve bu nedenle early stopping (erken durdurma), öğrenme oranının ayarlanması gibi çeşitli yöntemler kullanılarak daha iyi sonuçlar elde edilir [14].

Tüm bu çalışmalar, hiper parametre optimizasyonunun önemini vurgulamakta ve makine öğrenimi modellerinin başarısını artırmak için çeşitli etkili yöntemler sunmaktadır. Araştırma alanındaki bu gelişmeler, makine öğrenimi uygulamalarında daha güçlü, verimli ve güvenilir modellerin oluşturulmasını sağlamaktadır. Ancak, hiper parametre optimizasyonu alanında daha fazla çalışmaya ihtiyaç duyulmaktadır ve

gelecekte bu yönde yapılan araştırmaların alanın gelişimine katkı sağlaması beklenmektedir.

Sonuç olarak, literatürde yapılan araştırmalar, makine öğrenmesi ve hiper parametre optimizasyonu alanında hala birçok açık soru olduğunu ve daha etkili optimizasyon yöntemleri geliştirmek için sürekli bir çalışma alanı olduğunu göstermektedir. Bu çalışma, mevcut literatürdeki bu açıkları ele alarak, hiper parametre optimizasyonu alanında yeni bilgiler ve kavramlar sunarak, makine öğrenmesi modellerinin performansını artırmak için katkı sağlamayı hedeflemektedir.

3. AMAÇ

Bu çalışmanın temel amacı, makine öğrenmesi modellerinin performansını artırmak için etkili hiper parametre optimizasyon yöntemlerini araştırmak, analiz etmek, karşılaştırmak ve sonuç olarak yeni bir hiper parametre optimizasyon yöntemi geliştirmektir. Makine öğrenmesi, yapay zekanın önemli bir alt alanıdır ve çeşitli uygulama alanlarında büyük başarılar elde etmiştir. Ancak, makine öğrenmesi modellerinin performansı, hiper parametrelerin doğru bir şekilde ayarlanması ile doğrudan ilişkilidir. Bu nedenle, hiper parametre optimizasyonu, makine öğrenmesi modelinin başarısını ve genelleme yeteneğini önemli ölçüde etkileyebilen kritik bir adımdır.

Literatürde, farklı hiper parametre optimizasyon yöntemleri ve algoritmaları hakkında birçok çalışma mevcuttur. Izgara arama (grid search), rastgele arama (random search), Bayesian optimizasyon, genetik algoritmalar, parçacık sürü optimizasyonu gibi temel yaklaşımlardan, son zamanlarda popüler hale gelen metaheuristik yöntemlere kadar birçok farklı yaklaşım incelenmiştir. Bu çalışmada, literatürdeki mevcut yöntemler kapsamlı bir şekilde araştırılarak, bu yöntemlerin farklı makine öğrenmesi modelleri ve veri setleri üzerindeki etkinliği karşılaştırılacaktır.

Örnek olarak Google Vizier [15], Google tarafından geliştirilen bir hiper parametre ayarlama ve otomatik optimizasyon hizmetidir. Makine öğrenimi ve derin öğrenme gibi karmaşık model tabanlı problemleri çözerken, algoritmaların ve modellerin hiper parametrelerini optimize etmek zor olabilir. Google Vizier, bu süreci otomatikleştirerek daha iyi sonuçlar elde etmeyi amaçlar. Ancak otomatik optimizasyon, modele aşırı uyum riskini artırabilir. Bu nedenle, hiper parametre kombinasyonlarının doğru seçimi büyük önem taşımaktadır. Ayrıca Google'ın ürün ve hizmet politikaları, fiyatlandırma ve ücret bilgileri zaman içinde değişebilir. Ücretsiz sunulan yöntemlerin geliştirilmesi araştırmacılar için önem arz etmektedir.

Bununla birlikte, bazı makine öğrenmesi modelleri ve büyük veri setleri için geleneksel hiper parametre optimizasyon tekniklerinin manuel ve deneme yanılma yaklaşımı, hiper parametre uzayının boyutu, hesaplama gücü ve zaman kısıtlamaları gibi sebeplerden dolayı yetersiz kaldığı görülmüştür. Bu bağlamda, çalışmada özellikle derin öğrenme modellerine odaklanarak, bu karmaşık yapıları ve büyük veri setlerini ele alacak daha özel ve etkili hiper parametre optimizasyon yöntemleri araştırılacaktır.

Ana amacımız, derin öğrenme (deep learning) alanında kullanılan ve özellikle görüntü işleme, görüntü sınıflandırma, nesne algılama ve dil işleme gibi alanlarda büyük başarılar elde eden bir yapay sinir ağı türü olan Evrişimli Sinir Ağı (Convolutional Neural Network - CNN)'de mevcut literatürdeki hiper parametre optimizasyonu yöntemlerinin eksikliklerini belirleyerek, bu eksiklikleri gidermek için yeni bir hiper paremetre optimizasyonunu sağlayacak yöntem geliştirmektir.

Bu yeni yöntem, farklı makine öğrenmesi modelleri üzerinde uygulanacak ve performansını mevcut en iyi yöntemlerle karşılaştırılacaktır. Elde edilecek sonuçlar, makine öğrenmesi modellerinin performansını artırmak ve hiper parametre optimizasyon sürecini daha verimli ve etkili hale getirmek için yeni bir yaklaşım sunarak, akademik ve endüstriyel alanda yapay zeka sistemlerinin geliştirilmesine önemli bir katkı sağlamayı hedeflemektedir.

3.1. Hiper Parametre

Hiper parametreler, makine öğrenimi algoritmalarında modelin yapılandırılmasını ve eğitimini kontrol eden, verilerden öğrenilemeyen, kullanıcı tarafından ayarlanması gereken ve bir makine öğrenimi modelinin performansını artırmak için deneme yanılma yoluyla önceden belirlenen parametrelerdir. Bu parametrelerin doğru ayarlanması modelin performansını önemli derecede etkiler. Bu parametreler, modelin nasıl eğitileceği ne kadar hızlı öğreneceği ne kadar genelleştirme yapacağı vb. gibi faktörleri belirler.

Hiper parametrelerin önemi aşağıdaki maddeler yardımıyla açıklanabilir:

Model Performansı: Hiper parametreler, modelin eğitim ve değerlendirme performansını doğrudan etkiler. Doğru hiper parametre ayarları, modelin daha yüksek doğruluk ve daha iyi genelleme yeteneği elde etmesine yardımcı olabilir.

Aşırı Öğrenmeyi Kontrol Etmek: Aşırı öğrenme, modelin eğitim verilerine çok fazla uyması ve yeni verilere kötü bir şekilde genelleme yapması durumudur. Hiper parametreler, aşırı öğrenmeyi kontrol etmeye yardımcı olabilir. Örneğin, dropout oranı gibi hiper parametreler, aşırı öğrenmeyi azaltmak için kullanılabilir.

Hız ve Verimlilik: Bazı hiper parametreler, modelin eğitim hızını ve verimliliğini etkiler. Örneğin, öğrenme hızı (learning rate) gibi hiper parametreler, ne kadar hızlı bir şekilde modelin konverjans (bir algoritmanın veya sürecin belirli bir hedefe veya istenen duruma yaklaşarak durma veya istikrarlı bir şekilde ilerleme yeteneğini ifade eder) gösterdiğini belirler.

Dengeli Model Ayarı: Hiper parametreler, modelin farklı bileşenlerinin arasındaki dengeyi ayarlamak için kullanılabilir. Örneğin, bir sinir ağı modelinde katman sayısı ve her katmandaki birim sayısı hiper parametre olarak ayarlanabilir.

Uyumlu Model Seçimi: Farklı hiper parametre ayarları farklı modellerin oluşturulmasına neden olabilir. Farklı modellerin test edilmesi ve en iyi sonuçları veren hiper parametre ayarlarının seçilmesi önemlidir.

Hiper parametre ayarı için, farklı hiper parametre değerleri deneyerek modelin performansını izleyebilir ve en iyi sonuçları elde etmek için optimal ayarları bulabilirsiniz. Ancak bu süreç zaman alıcı olabilir.

3.2. Yaygın Makine Öğrenimi Algoritmaları ve Örnek Hiper Parametreleri

Makine öğrenimi alanında popüler ve etkili birçok algoritma bulunmaktadır. Bunlardan en sık kullanılanları olan SVM, Random Forest, K-NN ve CNN algoritmaları ve hiper parametleri kısaca tanıtılacaktır.

3.2.1. Destek Vektör Makinesi (Support Vector Machine-SVM)

Özellikle sınıflandırma ve regresyon problemlerinde kullanılan bir makine öğrenimi yöntemidir. Temel amacı, veri noktalarını farklı sınıflara veya değer aralıklarına ayıran bir hiper düzlem bulmaktır. SVM, veri noktalarını bu hiper düzleme olan uzaklıklarına göre sınıflandırır ve böylece sınıflar arasında en geniş marjı (uzaklığı) sağlayan hiper düzlemi bulmaya çalışır.

SVM'in Hiper Parametreleri;

C (Cost Parameter): C, bir tür hata toleransını kontrol eder. Daha yüksek C değerleri, eğitim verilerindeki hataları en aza indirmeye çalışırken, düşük C değerleri daha fazla marj alanına odaklanır. Bu, modelin aşırı uyum (overfitting) ve düşük uyum (underfitting) arasındaki dengeyi kontrol etmede önemlidir.

Kernel Fonksiyonu: SVM, doğrusal olarak ayrılabilir olmayan verileri doğrusal hiper düzlemlerle ayıramaz. Bu durumda kernel fonksiyonları kullanılır. Kernel fonksiyonları, verileri daha yüksek boyutlu uzaylara dönüştürerek doğrusal olarak ayrılabilir hale getirir. Örneğin, RBF (Radyal Temelli Fonksiyon) bir çeşit kernel fonksiyonudur.

Gamma (RBF Kernel için): Gamma, RBF kernelindeki veri noktalarının hiper uzaya ne kadar yayılacağını kontrol eder. Daha yüksek gamma değerleri, veri noktalarını daha sıkı bir şekilde düzenler, düşük gamma değerleri ise daha geniş bir dağılıma neden olur.

3.2.2. Rastgele Orman (Random Forest)

Makine öğrenimi alanında yaygın olarak kullanılan bir öğrenme algoritmasıdır. Temel amacı, birçok karar ağacını (decision tree) bir araya getirerek daha güçlü ve istikrarlı bir tahmin modeli oluşturmaktır. Random Forest, tek bir karar ağacının sahip olduğu eğilimleri (bias) ve varyansı dengelemek amacıyla birden fazla karar ağacını bir araya getirerek daha iyi sonuçlar elde etmeyi hedefler.

Random Forest'ın Hiper Parametreleri;

Ağaç Sayısı (n_estimators): Oluşturulan ağaçların sayısını belirler. Daha fazla ağaç, daha iyi bir genelleme sağlayabilir ancak işlem süresini artırabilir.

Özellik Sayısı (max_features): Her bir ağaç düğümünde kullanılacak maksimum özellik sayısını belirler. Düşük değerler aşırı öğrenmeyi azaltabilir.

Dallanma Kriteri (criterion): Ağaç düğümlerinin bölünmesi için kullanılacak ölçütü belirler. "gini" veya "entropy" gibi seçenekler bulunur.

Derinlik (max_depth): Bir ağacın maksimum derinliğini sınırlar, aşırı öğrenmeyi önlemek için kullanışlıdır.

Minimum Örnekleme Sayısı (min_samples_split): Bir düğümün bölünmesi için gereken minimum örnekleme sayısını belirler.

Minimum Yaprak Sayısı (min_samples_leaf): Bir yaprağın minimum örnekleme sayısını belirler. Daha küçük değerler modeli daha karmaşık hale getirebilir.

Rastgele Örnekleme (bootstrap): Eğitim verilerini rastgele örnekleme ile kullanıp kullanmamayı belirler.

3.2.3. K-En Yakın Komşu (K-Nearest Neighbors veya KNN)

Makine öğrenimi ve veri madenciliği alanında sınıflandırma ve regresyon problemleri için kullanılan bir algoritmadır. Temel fikri, yeni bir veri noktasını sınıflandırmak veya tahmin yapmak için ona en yakın olan eğitim veri noktalarının etrafındaki etiketleri veya değerleri kullanmaktır.

KNN'in Hiper Parametreleri;

Uzaklık Metriği: Veri noktaları arasındaki uzaklığı hesaplarken kullanılacak olan metrik. Öklidyen, Manhattan, Pearson Korelasyonu gibi farklı metrikler tercih edilebilir.

K Değeri: Komşu sayısını belirten parametre. Modelin performansına ve veri dağılımına göre seçilir.

3.3. Bu Çalışmada Kullanılan CNN (Evrişimli Sinir Ağı) ve Hiper Parametreleri

n_filter: Bu terim, evrişimli sinir ağlarında (Convolutional Neural Networks - CNN) filtrelerin veya çekirdeklerin sayısını belirtir. Her filtre, görüntülerde veya veri girişlerinde belirli özellikleri yakalamak için kullanılır. Daha fazla filtre, daha karmaşık özelliklerin algılanmasına ve modelin daha iyi performans göstermesine yardımcı olabilir.

filterSize: Filtrelerin boyutunu belirtir. Örneğin, 3x3, 5x5 gibi sık kullanılan filtre boyutları mevcuttur. Filtre boyutu, giriş verilerinin belirli bir alanını ele alarak özelliklerin tespitini sağlar.

poolSize: Max pooling veya average pooling gibi pooling katmanlarında kullanılan havuzlama boyutunu belirtir. Havuzlama, boyutunu belirtilen bir bölge içindeki verilerden tek bir değeri alarak boyut azaltma ve özellikleri vurgulama işlevi görür.

DenseSize: Yoğun (dense) katmanlardaki nöron sayısını ifade eder. Yoğun katmanlar, özelliklerin öğrenilmesi ve sınıflandırma gibi görevlerin gerçekleştirildiği katmanlardır.

Beta1 ve Beta2: Bu terimler, Adam optimizasyon algoritmasında kullanılan öğrenme oranının düzeltilmesi için kullanılır. Adam, gradyan tabanlı bir optimizasyon algoritmasıdır ve beta1 ve beta2 değerleri gradyanların hareketli ortalamalarını hesaplamak için kullanılır.

LearningRate: Öğrenme oranını ifade eder. Bu, bir modelin eğitim sırasında parametrelerini güncelleme hızını belirleyen bir hiper parametredir. Doğru öğrenme oranı, modelin hızlı bir şekilde öğrenmesine ve optimum sonuçlara ulaşmasına yardımcı olur.

BatchSize: Bir eğitim döngüsünde kullanılan veri örneklerinin sayısını ifade eder. Büyük bir veri kümesi varsa, bellek verimliliği ve paralel işlem avantajları için eğitim verileri küçük toplu (batch) halinde işlenir.

Epochs: Bir eğitim döngüsündeki adımların sayısını belirtir. Her epoch, modelin tüm veri kümesini bir kez tamamlaması anlamına gelir. Daha fazla epoch, modelin daha fazla eğitim almasını sağlayabilir, ancak aşırı uyuma (overfitting) riskini de artırabilir.

nofconv: CNN'lerdeki evrişimli katmanların sayısını ifade eder.

numberofHiderlayer: Modeldeki gizli katmanların (hidden layer) sayısını belirtir. Gizli katmanlar, giriş ve çıkış katmanları arasında bulunur ve verilerin içindeki karmaşık örüntüleri öğrenmeye yardımcı olurlar.

Her algoritmanın kendine özgü hiper parametreleri vardır ve bu parametrelerin ayarlanması, modelin performansını etkileyebilir. İyi bir model performansı için, bu hiper parametrelerin dikkatli bir şekilde seçilmesi ve ayarlanması önemlidir. Hiper parametre ayarlaması, genellikle deneme yanılma yoluyla gerçekleştirilir.

Örneğin, bir sinir ağı modelindeki hiper parametreler; katman sayısı, öğrenme hızı ve aktivasyon fonksiyonları gibi özellikleri içerebilir. Hiper parametrelerin doğru ayarlanması, bir modelin doğruluğunu, hassasiyetini veya diğer performans ölçümlerini artırabilir.

Birçok makine öğrenimi kütüphanesi hiper parametreleri ayarlamak için araçlar sağlar.

Hiper parametreleri otomatik ayarlamada kullanılan rastgele arama (random search), ızgara arama (grid search), Bayesian gibi birçok kabul görmüş yöntem

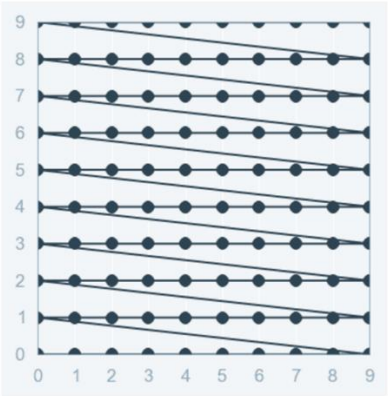
bulunmaktadır. Bu yöntemler ile doğru hiper parametre ayarlamaları, modelin performansını artırabilir ve daha iyi sonuçlar elde etmenizi sağlayabilir. Her yöntemin avantaj ve dezavantajları bulunmaktadır. Bu sebeple eldeki problem için kullanılması gereken yöntem de değişkenlik gösterebilir.

4. HİPER PARAMETRE OPTİMİZASYONUNDA KULLANILAN MEVCUT YÖNTEMLER

Eldeki makine öğrenme modelinin performansını artırmak için hiper parametre optimizasyonu kullanılır. Hiper parametre optimizasyonu, modelin hiper parametrelerini (modelin yapısını belirleyen parametreler) en uygun değerlere ayarlamak için bir dizi teknik kullanır.

4.1. Izgara Arama (Grid Search)

Bu yöntem, hiper parametrelerin Şekil 4.1.1'deki gibi belirli bir grid üzerinde deneme yanılma yöntemiyle test edildiği bir yöntemdir. Tüm hiper parametre kombinasyonları denenir ve en iyi performansa sahip kombinasyon seçilir. Grid Search yöntemiyle tüm kombinasyonlar denenmesi, hesaplama süresi açısından maliyetli olabilir.



Şekil 4.1.1: *Grid Search yönteminin veri tarama tekniği*

Grid search, hiper parametre değerleri için bir ızgara oluşturarak tüm kombinasyonları deneyerek en iyi hiper parametre setini bulmaya çalışan basit bir optimizasyon yöntemidir. Grid search, bir arama uzayında belirli adımlarla ilerler ve tüm olası kombinasyonları değerlendirir. Örneğin, bir sınıflandırma modeli için "learning rate" ve "n_estimators" adında iki hiper parametreyi ele alalım. Grid search, belirli adımlarla her iki hiper parametre için tüm kombinasyonları deneyerek aşağıdaki gibi bir arama yapar:

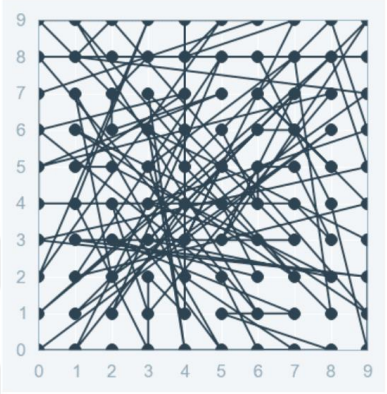
learning_rate = [0.1, 0.01, 0.001]

n_estimators = [50, 100, 150]

Bu durumda, grid search, 3 x 3 = 9 farklı hiper parametre kombinasyonunu değerlendirir ve en iyi sonuç veren hiper parametreleri seçer.

4.2. Rastgele Arama (Random Search)

Random Search yöntemi, hiper parametrelerin belirli bir aralık içinde Şekil 4.2.1'deki gibi rastgele seçildiği bir yöntemdir. Belirli bir sayıda iterasyon boyunca rastgele hiper parametre değerleri seçilir ve en iyi performansa sahip kombinasyon seçilir. Grid Search yöntemine göre daha verimli olabilir, çünkü rastgele seçimler sayesinde daha fazla hiper parametre alanını keşfedebilir.



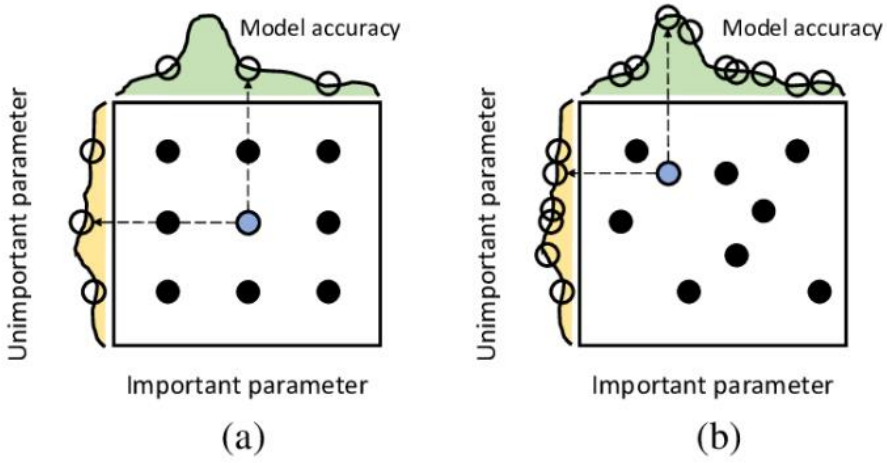
Şekil 4.2.1: *Random Search yönteminin veri tarama tekniği*

Örneğin, random search aşağıdaki gibi rastgele seçilmiş hiper parametre kombinasyonlarını deneyebilir:

learning_rate = [0.05, 0.002, 0.1]

n_estimators = [25, 200, 75]

Random search, belirli bir bütçeye sahipken ve daha büyük arama uzaylarında etkili sonuçlar elde etmek istenirken tercih edilebilir.



Şekil 4.2.2: *Grid Search ve Random Search yöntemlerinin karşılaştırılması*

Şekil 4.2.2, bir fonksiyonu optimize etmek için grid (ızgara) ve random (rastgele) arama yöntemlerini karşılaştıran bir görseldir. Bu fonksiyon, $f(x, y) = a(x) + b(y)$ şeklinde ifade edilir ve $a(x)$ ile $b(y)$ arasında düşük etkin boyutlu bir ilişki olduğu varsayılır $a(x)$ fonksiyonu yeşil renkle, $b(y)$ fonksiyonu ise sarı renkle gösterilmiştir.

Şekil 4.2.2 (a) Grid search, ızgara şeklinde belirli hiper parametre değerlerini denemek için kullanılan bir yöntemdir. Bu görselde, grid search ile yapılan dokuz denemede, $a(x)$ fonksiyonu yalnızca üç farklı noktada test edilmiştir. Bu nedenle, fonksiyon $a(x)$ yalnızca belirli parametre değerlerinde incelenirken, $b(y)$ farklı değerlerde test edilmemiştir. Grid search, düşük etkin boyutlu fonksiyonlarda belirli değerlerin keşfinde etkili olabilir, ancak yüksek boyutlu hiper-parametre optimizasyonunda başarısız olma eğilimindedir.

Şekil 4.2.2 (b) Random search, hiper parametre değerlerini rastgele seçerek deneme yapar. Görselde, random search ile yapılan dokuz denemede, $a(x)$ fonksiyonu farklı noktalarda test edilmiştir. Bu sayede, fonksiyon $a(x)$, çeşitli değerlerde keşfedilmiş ve daha geniş bir alanı kapsamıştır. Random search, yüksek boyutlu hiper parametre optimizasyonunda daha etkili olma eğilimindedir çünkü daha geniş bir parametre uzayında arama yapar ve keşfedilmemiş bölgeleri de dahil edebilir.

Sonuç olarak, bu görseldeki açıklama, düşük etkin boyutlu bir fonksiyonu optimize etmek için grid search'in sınırlı bir alanı ve belirli değerleri denemesi ve random search'in daha geniş bir alanda rastgele değerler keşfetmesi nedeniyle grid search'in sıkça başarısız olabileceğini göstermektedir. Yüksek boyutlu hiper-parametre optimizasyonunda, random search genellikle daha iyi sonuçlar elde etmeye yönelik bir seçenek olarak kabul edilir.

4.3. Bayesian Optimizasyon (Bayesian Optimization)

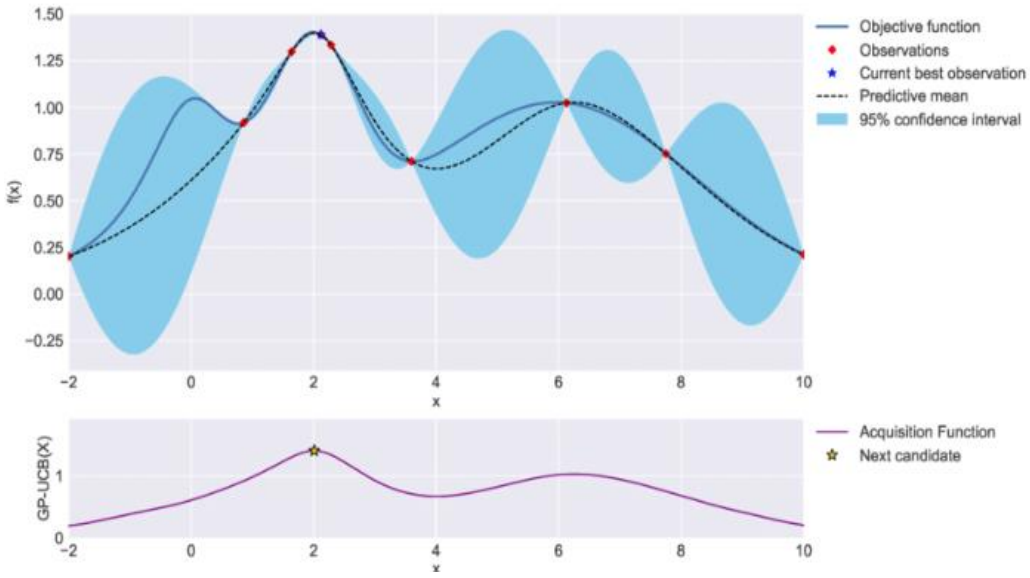
Bayesian Optimizasyon, karmaşık ve yüksek boyutlu uzaylarda verimli bir şekilde en iyi sonucu bulmayı amaçlayan bir optimizasyon yöntemidir. Bu yöntem, genellikle pahalı veya zaman alıcı gerçek deneyler veya simülasyonlar gerektiren problemlerde kullanılır. Özellikle hiper parametre ayarlama, makine öğrenimi modeli düzeni, endüstriyel tasarım optimizasyonu gibi alanlarda yaygın olarak kullanılır.

Temel fikir, daha önceki gözlemlere dayalı olarak en iyi sonraki noktayı tahmin etmek ve bu tahmini optimize etmek için kullanmaktır. Bu, modelin mevcut bilgi ile

gerçek farkındalık arasındaki dengeyi sağlayarak optimize edilen noktayı seçmesini sağlar.

Bayesian Optimizasyon yöntemi, daha az deneme yaparak hiper parametre alanını daha etkin bir şekilde keşfetmek için bir olasılık modeli kullanır. Bu yöntemde, hiper parametre alanını modelleyen bir olasılık dağılımı oluşturulur ve bu dağılım kullanılarak bir sonraki hiper parametre kombinasyonu tahmin edilir. Bu tahminler, modelin performansını en üst düzeye çıkarmak için seçilen bir dizi kombinasyonu değerlendirmek için kullanılır.

Örneğin, Bayesian optimizasyon, Gaussian Process Regression (GPR) veya Tree-structured Parzen Estimators (TPE) gibi olasılık modelleri kullanarak bir akıllı model oluşturabilir.



Şekil 4.3.1: *Bayesian Optimizasyonu yönteminin veri tarama fonksiyonu*

Şekil 4.3.1'de de olduğu gibi Bayesian optimizasyon, optimize etmek istenen fonksiyonu en iyi şekilde açıklayan bir sonlu Gaussian Process ile bir arka dağılım oluşturarak çalışır. Gözlem sayısı arttıkça, arka dağılım iyileşir ve algoritma, hangi parametre uzayı bölgelerinin keşfedilmeye değer olduğunu ve hangilerinin olmadığı daha da kesinleşir [16]. Gaussian Process, mevcut gözlem verilerine dayanarak bir tahmin dağılımı oluşturur. Bu dağılım, belirli bir noktadaki fonksiyon değerinin olasılık dağılımını ifade eder. Bayesian Optimizasyon sürecinde, bu tahmin dağılımı kullanılarak

en iyi sonraki gözlem noktasını seçmek ve optimize etmek için çeşitli stratejiler geliştirilir.

Tekrar tekrar yineledikçe, algoritma keşfetme ihtiyaçlarını dengeleyerek hedef işlev hakkındaki bilgileri dikkate alır. Gaussian Process, Bayesian Optimizasyon'un temelini oluşturan bir bileşendir. GP, hedef fonksiyonunun tahmini dağılımını ve belirsizlikleri modellemek için kullanılır. Bu, veri verimliliğini artırır, akıllı nokta seçimini iyileştirir ve belirsizlik yönetimini sağlar. Bu sayede Bayesian Optimizasyon, gerçek dünyadaki pahalı ve karmaşık optimizasyon problemlerini etkili bir şekilde çözmek için kullanılabilir hale gelir.

4.4. Genetik Algoritması (Genetic Algorithms)

Genetic Algorithms yöntemi, genetik işlemleri taklit eden bir optimizasyon tekniğidir. Rastgele bir popülasyon oluşturulur ve daha iyi performansa sahip olan kombinasyonlar seçilir. Seçilen kombinasyonlar, mutasyon ve çaprazlama operasyonlarıyla yeni nesiller oluşturulur. Bu süreç, en iyi performansa sahip kombinasyonu bulmak için tekrarlanır.

Genetik algoritma genellikle karmaşık ve çok boyutlu optimizasyon problemlerinde kullanılır. Özellikle matematiksel fonksiyonların en iyi veya en kötü değerini bulma, mühendislik tasarım problemleri, programlama, yapay sinir ağı hiper parametre ayarlaması gibi problemlerde etkili olabilirler.

Genetik algoritmanın temel bileşenleri şunlardır:

Popülasyon: Çözüm adaylarının bir koleksiyonudur. Her bir çözüm adayı genetik olarak bir bireyi temsil eder.

Birey (Bireysel): Bir çözüm adayını temsil eden genetik bilgiler kümesidir. Genellikle bu bilgiler bir dizi parametre veya gen olabilir.

Seçilim: Popülasyon içinden daha yüksek fitness değerine sahip bireylerin seçilimini içerir. Bu seçilim, daha iyi çözümleri gelecek nesillere taşımak amacıyla yapılır.

Çaprazlama (Crossover): İki farklı bireyin genetik bilgilerinin karıştırılarak yeni bireylerin üretildiği bir işlemdir. Bu, popülasyonun genetik çeşitliliğini artırır.

Mutasyon: Bireylerin genetik bilgilerinde rastgele değişiklikler yaparak çeşitliliği artıran bir işlemdir. Mutasyon, popülasyonun genetik keşif yapabilmesini sağlar.

Örneğin, genetik algoritmalar, aşağıdaki gibi bir popülasyonu değerlendirebilir:

Populasyon = [H1, H2, H3, H4, H5]

Performans = [0.85, 0.92, 0.87, 0.89, 0.90]

Genetik algoritmalar, en iyi performans gösteren bireyleri çaprazlama ve mutasyon operatörleri ile birleştirerek yeni nesiller oluşturarak en iyi hiper parametre kombinasyonlarını bulmaya çalışır. Bu algoritma, paralel işlem yetenekleri sayesinde birden çok aday çözümü aynı anda değerlendirebilme ve keşfedilmesi zor arama uzaylarında etkili sonuçlar elde etme yeteneği nedeniyle tercih edilir.

4.5. Parçacık Sürü Optimizasyonu (Particle Swarm Optimization)

Particle Swarm Optimization yöntemi, bir sürücü partikülün hareketini taklit eden bir optimizasyon yöntemidir. Her bir partikül, bir hiper parametre kombinasyonunu temsil eder ve en iyi performansa sahip olanları seçer. Parçacık Sürü Optimizasyonu, kendi geçmiş performansları ve sürüdeki en iyi performansları temel alarak hareket eder. Bu yöntem, global optimuma ulaşmak için partiküllerin birlikte çalışmasını sağlar.

PSO'nun çalışma prensibi genel olarak şu şekildedir:

Parçacıklar rastgele başlangıç konumlarına yerleştirilir.

Her parçacık, kendi şu anki konumundan en iyi sonucu veren konumu (bireysel en iyi) ve sürü içindeki tüm parçacıkların en iyi sonucu veren konumunu (sürü en iyi) takip eder.

Parçacıklar, hızlarını ve yönlerini bu iki en iyi konum arasında güncellerken, belirli ağırlık faktörleri ve rastgele faktörler kullanılır.

Hareket ettikten sonra her parçacık, yeni konumunda ne kadar iyi bir sonuç elde ettiğini değerlendirir. Eğer daha iyi bir sonuç elde edildiyse, bireysel en iyi konumunu günceller.

Sürü içindeki en iyi sonuç güncellendikçe, diğer parçacıklar da yeni en iyi sonuca doğru hareket etmeye devam eder.

4.6. Tavlama Benzetimi (Simulated Annealing)

Tavlama Benzetimi (Simulated Annealing), optimize edilmesi zor olan fonksiyonların veya problemlerin çözümünde kullanılan bir optimizasyon algoritmasıdır. İsmi, fiziksel bir süreç olan malzemenin yavaşça soğutulması (annealing) işlemine benzer bir şekilde, problemin optimize edilmesi için yavaşça daha iyi çözümlere yaklaşma

fikrinden gelir. Bu algoritma, genellikle kararlı olmayan ve karmaşık uzaylarda işlem yapan optimizasyon problemleri için kullanılır.

Simulated Annealing nasıl çalışır:

Başlangıç çözümü seçilir.

Yeni bir çözüm önerilir. Bu önerilen çözüm, mevcut çözümden rastgele değişikliklerle elde edilir.

Yeni çözümün mevcut çözümden daha iyi veya daha kötü olup olmadığı değerlendirilir.

Eğer yeni çözüm mevcut çözümden daha iyiyse, bu çözüm kabul edilir.

Eğer yeni çözüm mevcut çözümden daha kötüyse, bir olasılıkla yeni çözüm kabul edilir. Bu olasılık, kabul edilme olasılığı azaldıkça azalır.

Belirli bir sıcaklık (temperature) değeri kullanılarak olasılık hesaplanır. Bu sıcaklık başlangıçta yüksektir ve zamanla düşürülür. Sıcaklık düştükçe, kötü çözümlerin kabul edilme olasılığı azalır.

Belirli bir sayıda adım boyunca bu işlem tekrarlanır veya belirli bir durma kriterine ulaşılıncaya kadar devam eder.

Hiper parametre optimizasyonunda Simulated Annealing kullanımı:

Hiper parametre optimizasyonunda, genellikle bir makine öğrenimi modelinin performansını artırmak için kullanılan hiper parametrelerin en iyi değerlerini bulmaya çalışırız. Simulated Annealing, bu süreçte de kullanılabilir.

Hiper parametre Uzayının Tanımlanması: İlk adım, hiper parametrelerin değer aralıklarını ve adım büyüklüklerini belirlemektir. Bu, algoritmanın hangi değerleri deneyeceğini ve hangi yönde adım atacağını belirler.

Başlangıç Noktasının Seçilmesi: Bir başlangıç noktası seçilir. Bu, hiper parametrelerin başlangıç değerlerini ifade eder.

Yeni Çözüm Üretimi: Mevcut hiper parametre değerlerinden rastgele bir şekilde yeni hiper parametre değerleri üretilir.

Performans Değerlendirmesi: Yeni hiper parametre değerleri kullanılarak model eğitilir ve performans metrikleri değerlendirilir.

Kabul veya Red: Eğer yeni hiper parametre değerleri mevcut değerlere göre daha iyi sonuç veriyorsa, yeni değerler kabul edilir. Eğer daha kötü sonuç veriyorsa, Simulated Annealing mantığına göre belirli bir olasılıkla yeni değerler kabul edilebilir.

Sıcaklık Değerinin Azaltılması: Algoritma boyunca kullanılan sıcaklık değeri zamanla azaltılır. Bu, kötü çözümlerin kabul edilme olasılığının düşmesine ve algoritmanın daha istikrarlı bir şekilde en iyi çözüme yaklaşmasına yardımcı olur.

Adım Sayısı ve Durdurma Kriterleri: Algoritmanın kaç adım boyunca çalışacağı ve hangi durumda duracağı gibi kriterler belirlenmelidir.

Simulated Annealing yöntemi, bir olasılık dağılımı kullanarak olası hiper parametre kombinasyonlarını keşfetmek için bir arama stratejisi kullanır. Başlangıç kombinasyonu rastgele seçilir ve ardışık kombinasyonlar rastgele seçilir veya belirli bir kurala göre seçilir. Kötü performansa sahip kombinasyonlar bile belirli bir olasılıkla kabul edilebilir.

5. HİPER PARAMETRE OPTİMİZASYONU İÇİN GELİŞTİRİLEN YÖNTEM VE ADIMLARI

Hiper parametre optimizasyonu, makine öğrenimi algoritmalarını eğitirken kullanılan hiper parametrelerin en iyi değerlerini bulmayı amaçlar. Doğru hiper parametre değerleri seçilmediğinde, modelin performansı düşebilir veya eğitim süreci istenmeyen sonuçlar verebilir. Hiper parametre optimizasyonu için çeşitli yöntemler ve araçlar mevcuttur (örneğin, Grid Search, Random Search, Bayesian Optimization, vs.). Ancak yeni bir yönteme ihtiyaç duyulmasının verimlilik, hız ve doğruluk gibi nedenleri vardır. Geleneksel yöntemler bazen çok sayıda hiper parametre kombinasyonunu değerlendirmek için çok fazla hesaplama gerektirebilir. Yeni yöntemler, bu süreci daha hızlı ve daha verimli bir şekilde yapabilir, böylece model geliştirme sürecini hızlandırabilir. Ya da hiper parametreleri belirleme süreci uzun olsa da yapılan seçim, doğruluğu sebebiyle modelin performansını iyileştirebilir.

Geliştirilen yönteme ML olarak adlandırılmıştır.

ADIM 1: Kullanılacak olan kütüphaneler ve modüllerin import edilir. Bu kütüphaneler, model oluşturmak, eğitmek, verileri işlemek ve hiper parametre optimizasyonunu gerçekleştirmek için kullanılır.

ADIM 2: Parametre olarak aldığı bir sözlükteki değerlerin tüm kombinasyonlarını döndüren bir fonksiyon (tum_kombinasyonlar fonksiyonu) yazılır.

ADIM 3: Parametre olarak aldığı listedeki en büyük değerlerin indekslerini döndüren bir fonksiyon (en_buyuk_indeksler fonksiyonu) yazılır.

ADIM 4: En iyi performansı veren hiper parametreleri bulan bir fonksiyon (MLOptimizer fonksiyonu) yazılır.

ADIM 5: Verilen hiper parametreleri kullanarak bir model oluşturup test veri kümesindeki performansını hesaplayan bir fonksiyon (objective_func2 fonksiyonu:) yazılır.

ADIM 6: Verilen hiper parametre uzayından rastgele hiper parametreler seçerek model oluşturup performansını hesaplayan bir fonksiyon (objective_func fonksiyonu) yazılır.

ADIM 7: Verilen hiper parametrelerle bir Convolutional Neural Network (CNN) modeli oluşturan bir fonksiyon (createModel fonksiyonu) yazılır.

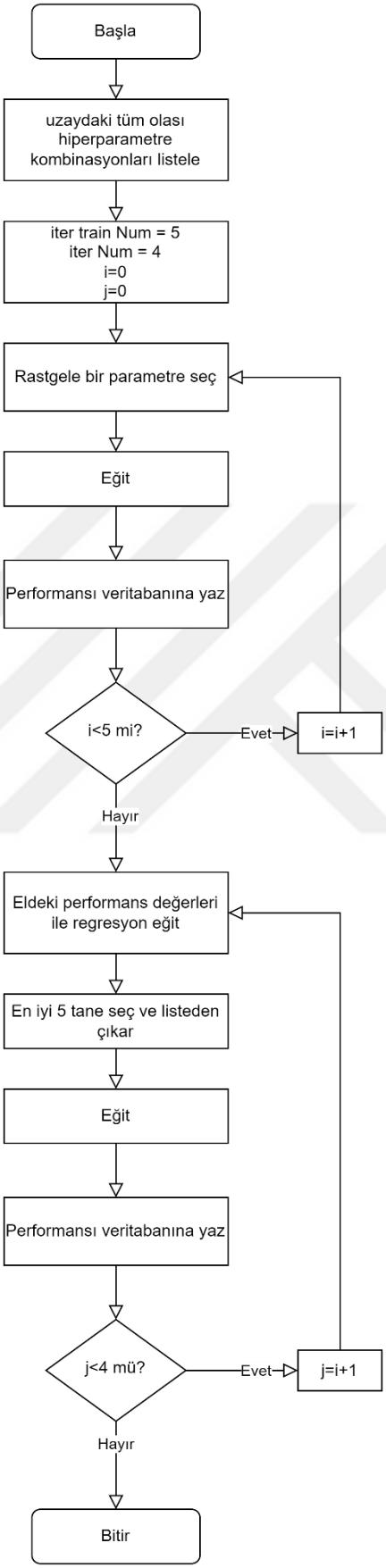
ADIM 8: CIFAR-10 veri kümesi Keras kütüphanesi aracılığıyla yüklenir.

Eğitim, test ve doğrulama veri kümesi olarak ayrıştırılır. Etiketler kategorik hale getirilir ve görüntüler normalize edilir.

ADIM 9: Hiper parametre uzayını tanımlayan bir sözlük (myspace) tanımlanır.

Hiper parametrelerin belirlenmesi sağlanır.

ADIM 10: Belirlenen hiper parametreler ile bir model oluşturulur ve test veri kümesindeki performansı hesaplanır.



Şekil 5.1: *Geliştirilen Algoritmaya ait Akış Diyagramı*

Bu kodda verilen hiper parametre uzayındaki kombinasyonları deneyerek en iyi performansa sahip olan hiper parametreleri bulmak hedeflenmektedir. Hiper parametrelerin farklı değerlerini deneyerek en iyi performansı elde etmeye çalışmaktayız.

“acc” kelimesi, “accuracy” kelimesinin kısaltmasıdır. Accuracy, bir sınıflandırma modelinin doğruluk oranını ifade eder. Sınıflandırma problemlerinde doğruluk, doğru olarak sınıflandırılan örneklerin toplam örnek sayısına oranıdır. Doğru sınıflandırılan örneklerin yüzdesi olarak da düşünülebilir.

Örneğin, bir modelin test veri kümesinde %85 doğruluk elde ettiği söylenirse, bu modelin sınıflandırma işleminde doğru olarak sınıflandırdığı örneklerin oranının %85 olduğu anlaşılır. Yani, 100 örneğin 85'i doğru şekilde sınıflandırılmıştır.

model.evaluate() fonksiyonu kullanılarak test veri kümesindeki model doğruluğu (accuracy) hesaplanır ve acc değişkenine atanır. Bu değer, modelin test veri kümesindeki performansını yansıtır.

Verilen hiper parametre değerleri ile bir CNN modeli oluşturarak ve bu modeli verilen doğrulama veri kümesi (valid_norm ve y_valid) üzerinde değerlendirerek doğruluk (accuracy) değerini döndüren bir işlevdir.

Fonksiyonun çalışma mantığı şu şekildedir:

Fonksiyona parameters adında bir liste verilir. Bu liste, modelin hiper parametrelerini içerir ve sırasıyla şu şekildedir;

n_filter: Conv2D katmanlarının filtre sayısı

filterSize: Conv2D katmanlarının filtre boyutu (kernel boyutu)

poolSize: MaxPooling2D katmanlarının boyutu

DenseSize: Dense (tam bağlantılı) katmanlarının boyutu (nöron sayısı)

Beta1: Adam optimizer'in beta1 parametresi

Beta2: Adam optimizer'in beta2 parametresi

LearningRate: Adam optimizer'in öğrenme hızı

BatchSize: Eğitimde kullanılacak mini-batch boyutu

Epochs: Eğitimdeki toplam epoch sayısı (iterasyon sayısı)

nofconv: Toplam Conv2D katmanı sayısı

numberofHiderlayer: Toplam gizli (hidden) katman sayısı

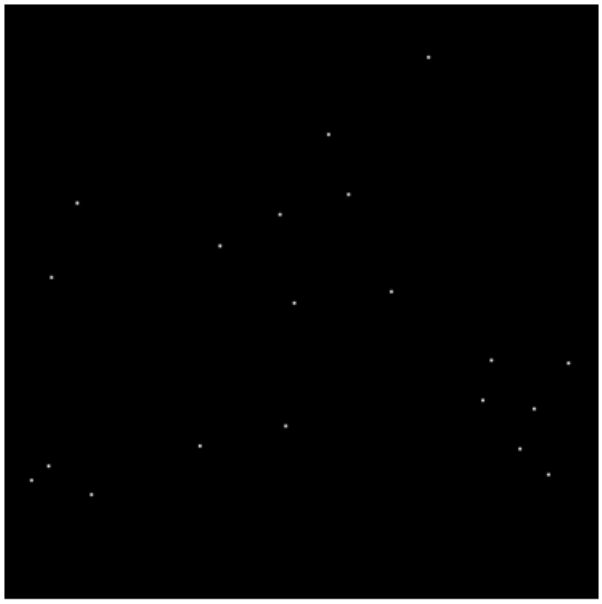
parameters listesinden hiper parametre değerleri çıkarılarak, n_filter, filterSize, poolSize, DenseSize, Beta1, Beta2, LearningRate, BatchSize, Epochs, nofconv ve numberofHiderlayer adlı değişkenlere atanır.

Bu hiper parametrelerle createModel fonksiyonu çağrılarak bir CNN modeli oluşturulur. Modelin derleme işlemi model.compile ve eğitim işlemi model.fit ile gerçekleştirilir.

Model, verilen doğrulama veri kümesi (valid_norm ve y_valid) üzerinde model.evaluate ile değerlendirilir ve doğruluk (accuracy) değeri olan acc elde edilir.

Fonksiyon, elde edilen doğruluk değerini (acc) döndürür.

Bu fonksiyon, hiper parametre uzayındaki farklı hiper parametre kombinasyonları için model performansını değerlendirmek için kullanılır. Hiper parametre optimizasyonu yaparken, en iyi performansa sahip olan hiper parametre değerlerini bulmak için bu fonksiyonu kullanarak modellerin performansını değerlendirebiliriz.



Şekil 5.2: *Veri uzayında bulunan 43200 noktadan seçilen 20 seçimin örnek şekli*

Şekil 5.2’de 10 sınıf içerisindeki 43200 adet hiper parametre kombinasyonlarından yalnızca 20 tanesinin seçimini gösteren durum verilmiştir.

Tablo 5.1: *Veri örneklerinin oluşturduğu uzay*

myspace					
n_filter	4	8	16		
filterSize	1	3			
poolSize	1	3			
DenseSize	8	16	32		
Beta1	0,9	0,93	0,95	0,97	0,99
Beta2	0,9	0,93	0,95	0,97	0,99
LearningRate	0,001	0,01			
BatchSize	8	16	32		
Epochs	20	30			
nofconv	1	2			
numberofHiderlayer	1	2			

Tablo 5.1’de, modelde yer alan hiper parametrelerin oluşturduğu uzay verilmiştir.

6. BU ÇALIŞMADA KULLANILAN VERİ SETLERİ

Çalışmada farklı sınıflarda görseller içeren CIFAR-10, MNIST ve FASHIONMNIST olmak üzere 3 farklı veri seti kullanılmıştır.

6.1. CIFAR-10

CIFAR-10 (Canadian Institute for Advanced Research - 10) veri kümesi, makine öğrenimi ve derin öğrenme alanlarında yaygın olarak kullanılan bir başka popüler veri kümesidir. CIFAR-10, 10 farklı nesne sınıfını içeren renkli (RGB) görüntülerden oluşur.

CIFAR-10 veri kümesinde toplamda 10 sınıf bulunur ve her bir sınıf farklı nesne kategorilerini temsil eder.

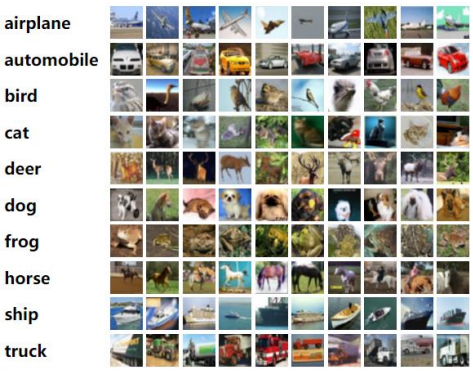
Sınıflar: Uçak, Otomobil, Kuş, Kedi, Geyik, Köpek, Kurbağa, At, Gemi, Kamyon

Veri boyutu: Her görüntü 32x32 piksel boyutlarında ve renklidir (RGB formatında). Yani her piksel, üç renk kanalı (kırmızı, yeşil, mavi) ile temsil edilir ve her kanal 0 ile 255 arasında bir değer alır.

Veri miktarı: CIFAR-10 veri kümesi toplamda 60,000 renkli görüntüden oluşur. Bunların 50,000'i eğitim için, geri kalan 10,000'i ise test için kullanılır.

CIFAR-10 derin öğrenme algoritmalarını ve yapay sinir ağlarını görüntü sınıflandırma, nesne tanıma ve örüntü algılama gibi görsel görevlerde test etmek için sıkça kullanılır. Veri kümesi, MNIST gibi basit ve küçük boyutlu olmayıp, daha gerçek dünya verilerini temsil eden renkli görüntüler içerdiği için daha zorlu bir problem sunar.

Bu nedenle, CIFAR-10 veri kümesi, bilgisayar görüşü alanında çalışan araştırmacılar ve öğrenciler için önemli bir öğrenme ve test kaynağıdır. Ayrıca, derin öğrenme modellerini daha karmaşık veri kümelerinde nasıl uygulayacaklarını öğrenmek isteyenler için de mükemmel bir seçenektir.



Şekil 6.1.1: *CIFAR-10 veri kümesi sınıfları ve örnek görseller*

Şekil 6.1.1'de CIFAR-10 veri kümesinin sınıflarına ait örnek görseller verilmiştir.

6.2. MNIST

MNIST (Modified National Institute of Standards and Technology), makine öğrenimi ve derin öğrenme alanında oldukça popüler olan bir veri kümesidir. Bu veri kümesi, el yazısı rakamlarını içerir ve sınıflandırma problemlerinde sıkça kullanılır.

MNIST veri kümesinin temel özellikleri şunlardır:

Sınıflar: Toplamda 10 sınıf bulunur ve her bir sınıf, 0 ile 9 arasındaki rakamları temsil eder.

Veri boyutu: Her görüntü 28x28 piksel boyutlarında ve siyah-beyaz (grayscale) olarak temsil edilir. Bu nedenle her piksel, 0 (siyah) ile 255 (beyaz) arasında bir değer alır.

Veri miktarı: Toplamda 70,000 görüntüden oluşur. Bunların 60,000'i eğitim için, geri kalan 10,000'i ise test için kullanılır.

MNIST veri kümesi, derin öğrenme algoritmalarını anlamak ve test etmek için temel bir veri kümesi olarak hizmet eder. Özellikle görüntü sınıflandırma algoritmalarını geliştirmek, örüntü tanıma ve optik karakter tanıma (OCR) üzerine çalışmalar yürütmek için kullanılan yaygın bir veri kümesidir.

MNIST veri kümesi, öğrenme algoritmalarının ve yapay sinir ağlarının temellerini anlamak için ideal bir seçimdir. Aynı zamanda, yeni başlayanlar için de çok faydalı bir

kaynaktır, çünkü veri kümesi nispeten küçük ve basittir, bu da hızlı deneyler yapma ve algoritma performansını değerlendirme fırsatı sunar.

***Şekil 6.2.1**: MNIST veri kümesi sınıfları ve örnek görseller*

Şekil 6.2.1'de MNIST veri kümesinin sınıflarına ait örnek görseller verilmiştir.

6.3. FASHIONMNIST

FASHIONMNIST, giyim ve moda ürünlerini içeren popüler bir veri kümesidir. MNIST (Modified National Institute of Standards and Technology) veri kümesinin bir türevidir. Geleneksel MNIST veri kümesi, 28x28 piksel boyutlarındaki siyah-beyaz el yazısı rakamlarını içerirken, FASHIONMNIST, aynı boyutlarda 10 farklı moda kategorisinden resimler içerir.

***Şekil 6.3.1:** FASHIONMNIST veri kümesi sınıfları ve örnek görseller*

Şekil 6.3.1'de FASHIONMNIST veri kümesinin sınıflarına ait örnek görseller verilmiştir.

Toplamda 10 sınıf bulunur ve her bir sınıf farklı bir giyim veya moda ürününü temsil eder.

Sınıflar: T-shirt/Top (Tişört/Üst), Trouser (Pantolon), Pullover (Hırka), Dress (Elbise), Coat (Kaban), Sandal (Sandalet), Shirt (Gömlek), Sneaker (Spor ayakkabı), Bag (Çanta), Ankle boot (Ayak bileğinde biten bot)

Veri boyutu: Her görüntü 28x28 piksel boyutlarında ve siyah-beyaz (grayscale) olarak temsil edilir.

Veri miktarı: Toplamda 70,000 görüntüden oluşur. Bunların 60,000'i eğitim için, geri kalan 10,000'i ise test için kullanılır.

FASHIONMNIST, özellikle makine öğrenimi ve derin öğrenme algoritmalarını anlamak, geliştirmek ve test etmek amacıyla MNIST veri kümesine alternatif olarak oluşturulmuştur. MNIST veri kümesi, bilgisayar görüşü alanında temel bir veri kümesi olmuştu, ancak çok basit olduğu ve günümüzün daha karmaşık görevleri için yeterince zorlayıcı olmadığı düşünülerek FASHIONMNIST oluşturulmuştur.

FASHIONMNIST veri kümesi, derin öğrenme modeli ve algoritmalarını resim sınıflandırma, örüntü tanıma ve moda ürünleri gibi birçok görsel görevde test etmek için yaygın olarak kullanılmaktadır.

6.4. HESAPSAL SONUÇLAR

Hiper parametre optimizasyonu için CIFAR-10 veri kümesi ve Random Search yöntemi kullanıldığında bu yöntemin hiper parametreleri yirmi çevrimde tayin etmesi yaklaşık 5630 saniye sürmektedir. Sınıflandırma modelinin doğruluk oranı ise tablo6.4.2’de de görüldüğü gibi %58,23 çıkmaktadır.

Hiper parametre optimizasyonu için CIFAR-10 veri kümesi ve Bayesian yöntemi kullanıldığında bu yöntemin hiper parametreleri yirmi çevrimde tayin etmesi yaklaşık 6076 saniye sürmektedir. Sınıflandırma modelinin doğruluk oranı ise tablo6.4.2’de de görüldüğü gibi %60,57 çıkmaktadır.

Hiper parametre optimizasyonu için CIFAR-10 veri kümesi ve geliştirilen yöntem kullanıldığında bu yöntemin hiper parametreleri yirmi çevrimde tayin etmesi yaklaşık 10151 saniye sürmektedir. Regresyonda geçen süre ise 13,01 saniyedir. Sınıflandırma modelinin doğruluk oranı ise tablo 6.4.2'de de görüldüğü gibi %63,73 çıkmaktadır.

Tablo 6.4.1: *CIFAR-10 veri kümesinde Geliştirilen Yöntem, Bayesian Yöntemi, Random Search Yöntemi kullanıldığında tayin edilen hiper parametreler*

	n_filter	filterSize	poolSize	DenseSize	Beta1	Beta2	LearningRate	BatchSize	Epochs	nofconv	numberofHiderlayer
CIFAR10-ML	16	3	3	32	0,9	0,97	0,001	32	30	2	2
CIFAR10-Bayes	16	3	1	32	0,9	0,93	0,001	32	20	2	1
CIFAR10-Random	8	3	3	32	0,93	0,97	0,001	32	20	1	1

Tablo 6.4.2: *Tayin edilen hiper parametreler ile alınan sonuçlar*

	Toplam geçen süre	test_acc	train_acc	val_acc
CIFAR10-ML	10151,288	0,6373	0,7085	0,6426
CIFAR10-Bayes	6075,9459	0,6057	0,723	0,6128
CIFAR10-Random	5629,9395	0,5823	0,6236	0,5808

Tablolardaki sonuçlardan da anlaşılacağı gibi geliştirilen yöntem ile sınıflandırma modelinin doğruluk oranı en yüksek geliştirilen yöntem ile sağlanmıştır.

Random Search en kısa sürede sonucu veriyor olsa da doğruluk oranı %58'de kalmaktadır.

Bayesian Hyper Opt. Yöntemi ise bu üç yöntemden hem süre hem de doğruluk bakımından ikinci sırayı almaktadır.

Hiper parametre optimizasyonu için MNIST veri kümesi ve Random Search yöntemi kullanıldığında bu yöntemin hiper parametreleri tayin etmesi yaklaşık 5853 saniye sürmektedir. Sınıflandırma modelinin doğruluk oranı ise tablo 6.4.4'te de görüldüğü gibi %98,1 çıkmaktadır.

Hiper parametre optimizasyonu için MNIST veri kümesi ve Bayesian yöntemi kullanıldığında bu yöntemin hiper parametreleri tayin etmesi yaklaşık 5851 saniye sürmektedir. Sınıflandırma modelinin doğruluk oranı ise tablo 6.4.4'te de görüldüğü gibi %98,38 çıkmaktadır.

Hiper parametre optimizasyonu için MNIST veri kümesi ve geliştirilen yöntem kullanıldığında bu yöntemin hiper parametreleri tayin etmesi yaklaşık 8785 saniye

sürmektedir. Regresyonda geçen süre ise 13,44 saniyedir. Sınıflandırma modelinin doğruluk oranı ise tablo 6.4.4'te de görüldüğü gibi %98,18 çıkmaktadır.

Tablo 6.4.3: *MNIST veri kümesinde Geliştirilen Yöntem, Bayesian Yöntemi, Random Search Yöntemi kullanıldığında tayin edilen hiper parametreler*

	n_filter	filterSize	poolSize	DenseSize	Beta1	Beta2	LearningRate	BatchSize	Epochs	nofconv	numberofHiderlayer
MNIST-ML	16	3	3	32	0,99	0,99	0,001	16	20	2	1
MNIST-Bayes	8	3	3	32	0,97	0,95	0,001	8	20	2	1
MNIST-Random	8	3	1	16	0,9	0,93	0,001	8	30	2	1

Tablo 6.4.4: *Tayin edilen hiper parametreler ile alınan sonuçlar*

	Toplam geçen süre	test_acc	train_acc	val_acc
MNIST-ML	8784,82	0,9818	0,9949	0,9823
MNIST-Bayes	5850,772	0,9838	0,9913	0,9821
MNIST-Random	5853,0889	0,981	0,9912	0,9794

Tablolardaki sonuçlardan da anlaşılacağı gibi geliştirilen yöntem ile sınıflandırma modelinin doğruluk oranları diğer yöntemler ile birbirine çok yakın olsa da Random Search yöntemine göre daha yüksek doğruluk oranı elde edilmiştir.

Hiper parametre optimizasyonu için FASHIONMNIST veri kümesi ve Random Search yöntemi kullanıldığında bu yöntemin hiper parametreleri tayin etmesi yaklaşık 5336 saniye sürmektedir. Sınıflandırma modelinin doğruluk oranı ise tablo6.4.6'da da görüldüğü gibi %89,77 çıkmaktadır.

Hiper parametre optimizasyonu için FASHIONMNIST veri kümesi ve Bayesian yöntemi kullanıldığında bu yöntemin hiper parametreleri tayin etmesi yaklaşık 6243 saniye sürmektedir. Sınıflandırma modelinin doğruluk oranı ise tablo6.4.6'da da görüldüğü gibi %90,48 çıkmaktadır.

Hiper parametre optimizasyonu için FASHIONMNIST veri kümesi ve geliştirilen yöntem kullanıldığında bu yöntemin hiper parametreleri tayin etmesi yaklaşık 8294 saniye sürmektedir. Regresyonda geçen süre ise 13,82 saniyedir. Sınıflandırma modelinin doğruluk oranı ise tablo 6.4.6'da da görüldüğü gibi %91,1 çıkmaktadır.

Tablo 6.4.5: *FASHIONMNIST veri kümesinde Geliştirilen Yöntem, Bayesian Yöntemi, Random Search Yöntemi kullanıldığında tayin edilen hiper parametreler*

	n_filter	filterSize	poolSize	DenseSize	Beta1	Beta2	LearningRate	BatchSize	Epochs	nofconv	numberofHiderlayer
FACION-ML	16	3	3	32	0,95	0,99	0,001	32	30	2	1
FACION-Bayes	16	3	3	16	0,93	0,97	0,001	32	30	2	2
FACION-Random	16	3	1	16	0,97	0,99	0,001	16	30	2	2

Tablo 6.4.6: *Tayin edilen hiper parametreler ile alınan sonuçlar*

	Toplam geçen Süre	**test_acc**	**train_acc**	**val_acc**
FASHION-ML	8294,6785	0,9110	0,9443	0,9149
FASHION-Bayes	6243,45	0,9048	0,9361	0,9079
FASHION-Random	5336,18	0,8977	0,9471	0,8968

Tablolardaki sonuçlardan da anlaşılacağı gibi geliştirilen yöntem ile sınıflandırma modelinin doğruluk oranı diğer yöntemlere göre daha yüksek oranı elde edilmiştir.

7. SONUÇ, YORUM VE TARTIŞMA

Bu çalışmada, farklı veri kümeleri ve hiper parametre optimizasyon yöntemleri kullanılarak sınıflandırma modellerinin performansı değerlendirilmiştir. Elde edilen sonuçlar, modelin hiper parametrelerini tayin etmek için kullanılan yöntemlerin süreç ve başarı açısından önemli farklılıklar gösterdiğini ortaya koymaktadır.

İlk olarak, FASHIONMNIST veri kümesi için Random Search yöntemi kullanıldığında elde edilen doğruluk oranının %89 olduğu görülmüştür. Random Search yöntemi, hiper parametre optimizasyonunu gerçekleştirmek için 5336 saniye süre almıştır. Daha sonra, aynı veri kümesi için Bayesian yöntemi kullanıldığında doğruluk oranının %90 olduğu ve yöntemin 6243 saniye sürdüğü tespit edilmiştir. Geliştirilen yöntem ile elde edilen sonuçlar ise, %91 doğruluk oranı ve 8294 saniyelik süre ile diğer yöntemlere göre daha başarılı bir optimizasyonu işaret etmektedir.

MNIST veri kümesinde de benzer sonuçlar elde edilmiştir. Random Search ve Bayesian yöntemleri, %98 doğruluk oranına ulaşmış ve sırasıyla 5853 saniye ve 5851 saniye süre almıştır. Geliştirilen yöntem, %98 doğruluk oranı ile en yüksek performansa ulaşmış, ancak 8785 saniyelik süresi ile diğer yöntemlere göre daha uzun bir optimizasyon sürecine ihtiyaç duymuştur.

Son olarak, CIFAR-10 veri kümesi için yapılan optimizasyonlarda, Random Search ve Bayesian yöntemleri düşük doğruluk oranları (%58 ve %60.5) ve orta düzeyde sürelerle (5630 saniye ve 6076 saniye) sonuçlanmıştır. Geliştirilen yöntem ise %64 doğruluk oranı ile diğerlerine göre daha iyi bir performans sergilemiştir, ancak 10151 saniyelik süresi ile en yavaş optimizasyon sürecini temsil etmektedir.

Sonuçlar, hiper parametre optimizasyonunun model performansı üzerinde önemli bir etkisi olduğunu göstermektedir. Yapılan analizler, geliştirilen yöntemin, bazı veri kümelerinde daha iyi sonuçlar verdiğini ortaya koymaktadır. Ancak, geliştirilen yöntemin süresi diğer yöntemlere göre daha uzun olabilmektedir.

Bu durum, hiper parametre optimizasyonunun etkili bir şekilde yönetilmesi ve hesaplama kaynaklarından optimum şekilde faydalanılması gerektiğini vurgulamaktadır.

Özetle, bu çalışma, farklı veri kümeleri ve hiper parametre optimizasyon yöntemlerinin sınıflandırma modellerinin performansı üzerinde farklı etkilere sahip olduğunu göstermektedir. Geliştirilen yöntemin belirli veri kümeleri için Random Search ve Bayesian Optimizasyon'a göre daha iyi sonuçlar verdiği tespit edilmiştir. Bu çalışma, gelecekte hiper parametre optimizasyonu konusunda daha kapsamlı araştırmaların yapılmasına ve veri kümelerine uygun en uygun yöntemlerin seçilmesine katkı sağlayabilir. Tez çalışmasının ileriki adımlarında yapılacak olan iyileştirmelerde eğitim ve test sürelerinin kısaltılması amaçlanmaktadır, regresyon süreleri ise benzer değerlerdedir.

KAYNAKÇA

[1] Bengio, Y., et al. (2012). "Random Search for Hyper-Parameter Optimization." Journal of Machine Learning Research.

[2] Snoek, J., et al. (2012). "Practical Bayesian Optimization of Machine Learning Algorithms." Advances in Neural Information Processing Systems.

[3] Z. Li, F. Liu, W. Yang, S. Peng and J. Zhou, (2021) "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects," in IEEE Transactions on Neural Networks and Learning Systems, vol. 33, no. 12, pp. 6999-7019.

[4] Pelikan, M., Goldberg, D.E. (1999). "BOA: The Bayesian Optimization Algorithm." Genetic Programming and Evolvable Machines.

[5] Houck, C., Joines, J. and Kay, M. (1995). "A Genetic Algorithm for Function Optimization: A Matlab Implementation."

[6] Eggensperger, K., et al. (2022). "SMAC3: A Versatile Bayesian Optimization Package For Hyperparameter Optimization."

[7] Eberhart, R.C., Kennedy, J. (1995). "A New Optimizer Using Particle Swarm Theory." Proceedings of the Sixth International Symposium on Micro Machine and Human Science.

[8] Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., & Talwalkar, A. (2017). "Hyperband: A novel bandit-based approach to hyperparameter optimization." Journal of Machine Learning Research.

[9] Bergstra, J., et al. (2013). "Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures." Proceedings of the 30th International Conference on Machine Learning.

[10] Blum, C., Roli, A. (2003). "Metaheuristics in combinatorial optimization: Overview and conceptual comparison." ACM Computing Surveys.

[11] Shi, Y., Eberhart, R.C. (1998). "A modified particle swarm optimizer." Proceedings of the IEEE International Conference on Evolutionary Computation.

[12] Koza, J.R. (1992). "Genetic Programming: On the Programming of Computers by Means of Natural Selection." MIT Press.

[13] Kirkpatrick, S., Gelatt Jr, C.D., Vecchi, M.P. (1983). "Optimization by Simulated Annealing." Science.

[14] Bottou, L. (2012). "Stochastic Gradient Descent Tricks." Neural Networks: Tricks of the Trade.

[15] Golovin, D., Solnik, B., Moitra, S., Kochanski, G., Karro, J., & Sculley, D. (2017, August). Google vizier: A service for black-box optimization. In Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 1487-1495).

[16] Joy, T. T. (2018). "A flexible transfer learning framework for Bayesian optimization with convergence guarantee."

http-1 https://www.researchgate.net/figure/Comparison-between-a-grid-search-and-b-random-search-for-hyper-parameter-tuning-The_fig2_341691661 (Erişim tarihi: 07.01.2023)

http-2 https://thecorrelation.in/overfitting-and-underfitting/ (Erişim tarihi: 11.02.2023)

http-3 https://www.cs.toronto.edu/~kriz/cifar.html (Erişim tarihi: 18.03.2023)

http-4 https://en.wikipedia.org/wiki/MNIST_database (Erişim tarihi: 22.04.2023)

http-5 https://en.wikipedia.org/wiki/Fashion_MNIST (Erişim tarihi: 22.04.2023)

ÖZGEÇMİŞ

ORCID NO: 0009 0004 6140 5869

Ad Soyad : Gözde AKBULUT ÖZ

Yabancı Dil : İngilizce

Eğitim ve Mesleki Geçmiş:

- 2022-Devam Ediyor, BT Kıdemli Uzmanı, Eczacıbaşı Yapı Gereçleri, Bilgi Teknolojileri
- 2021-2022, BT Kıdemli Uzmanı, GÜROK HOLDİNG, Bilgi Teknolojileri
- 2020-2021, Sakarya Üniversitesi, Fen Bilimleri Enstitüsü, Bilişim Teknolojileri Ana Bilim Dalı YL (Tezsiz)
- 2020-2023, Eskişehir Teknik Üniversitesi, Lisansüstü Eğitim Enstitüsü, Endüstri Mühendisliği YL (Tezli)
- 2019-2021, BT Uygulamaları Uzmanı, ZORLU ENERJİ, Bilgi Teknolojileri
- 2017-2019, ERP Mühendisi, ISIDEM INSULATION, Bilgi Teknolojileri
- 2013-2017, Eskişehir Osmangazi Üniversitesi, Endüstri Mühendisliği

Yayınlar ve/veya Bilimsel/Sanatsal Faaliyetler:

- 2023, 9. ULUSLARARASI MÜHENDİSLİK VE TEKNOLOJİ YÖNETİMİ KONGRESİ
  ÇİMEN, E. ve AKBULUT ÖZ, G. (2023). Makine Öğrenmesi Problemlerinde Hiper Parametre Optimizasyonu. 9. Uluslararası Mühendislik ve Teknoloji Yönetimi Kongresi, syf. 880.

Ödüller:

- 2017, İkincilik, TMMOB, Endüstri Mühendisi Adayları Yarışıyor, Eskişehir

Mesleki Birlik/Dernek/Kuruluş Üyelikleri:

- 2017 – Devam Ediyor, TMMOB Makina Mühendisleri Odası, Eskişehir Şubesi