T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DİJİTAL KARBON AYAK İZİNİN OPTİMİZASYONU İÇİN UÇ CİHAZLARDA YAPAY ZEKA VE MAKİNE ÖĞRENMESİ UYGULAMALARI

YÜKSEK LİSANS TEZİ

Çağlar ŞİMŞEK

Bilişim Sistemleri Mühendisliği Anabilim Dalı

TEMMUZ 2025

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DİJİTAL KARBON AYAK İZİNİN OPTİMİZASYONU İÇİN UÇ CİHAZLARDA YAPAY ZEKA VE MAKİNE ÖĞRENMESİ UYGULAMALARI

YÜKSEK LİSANS TEZİ

Çağlar ŞİMŞEK

Bilişim Sistemleri Mühendisliği Anabilim Dalı

Tez Danışmanı: Dr. Öğr. Üyesi Fatih ÇALLI

TEMMUZ 2025

Çağlar ŞİMŞEK tarafından hazırlanan “Dijital Karbon Ayak İzinin Optimizasyonu İçin Uç Cihazlarda Yapay Zeka Ve Makine Öğrenmesi Uygulamaları” adlı tez çalışması 03.07.2025 tarihinde aşağıdaki jüri tarafından oy birliği/oy çokluğu ile Sakarya Üniversitesi Fen Bilimleri Enstitüsü Bilişim Sistemleri Mühendisliği Anabilim Dalı Yüksek Lisans tezi olarak kabul edilmiştir.

Tez Jürisi

Jüri Başkanı :

Jüri Üyesi :

Jüri Üyesi :

ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ

Sakarya Üniversitesi Fen Bilimleri Enstitüsü Lisansüstü Eğitim-Öğretim Yönetmeliğine ve Yükseköğretim Kurumları Bilimsel Araştırma ve Yayın Etiği Yönergesine uygun olarak hazırlamış olduğum "DİJİTAL KARBON AYAK İZİNİN OPTİMİZASYONU İÇİN UÇ CİHAZLARDA YAPAY ZEKA VE MAKİNE ÖĞRENMESİ UYGULAMALARI" başlıklı tezin bana ait, özgün bir çalışma olduğunu; çalışmamın tüm aşamalarında yukarıda belirtilen yönetmelik ve yönergeye uygun davrandığımı, tezin içerdiği yenilik ve sonuçları başka bir yerden almadığımı, tezde kullandığım eserleri usulüne göre kaynak olarak gösterdiğimi, bu tezi başka bir bilim kuruluna akademik amaç ve unvan almak amacıyla vermediğimi ve 20.04.2016 tarihli Resmi Gazete'de yayımlanan Lisansüstü Eğitim ve Öğretim Yönetmeliğinin 9/2 ve 22/2 maddeleri gereğince Sakarya Üniversitesi'nin abonesi olduğu intihal yazılım programı kullanılarak Enstitü tarafından belirlenmiş ölçütlere uygun rapor alındığını, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun ortaya çıkması halinde doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi beyan ederim.

(17/07/2025).

Çağlar ŞİMŞEK

Değerli aileme,

TEŞEKKÜR

Yüksek lisans eğitimim boyunca ve tez çalışmamın her aşamasında yönlendirmeleri, tez konumu belirlemede ve çalışmalarım sırasında fikirlerini ve bilgilerini paylaşan, araştırmanın planlanmasından yazılmasına kadar tüm aşamalarında yardımlarını esirgemeyen, teşvik eden, aynı titizlikte beni yönlendiren ve tavsiyeleri ile bana ışık tutan değerli danışman hocam Dr. Öğr. Üyesi Fatih ÇALLI'ya sonsuz teşekkürler.

Çağlar ŞİMŞEK

İÇİNDEKİLER

Sayfa

KISALTMALAR

CNN : Evrişimli Sinir Ağı
CPU : Merkezî İşlem Birimi
DÖ : Derin Öğrenme
DSP : Dijital Sinyal İşlemcisi
FL : Federe Öğrenme
FLOP : Kayan Nokta İşlemleri
FPGA : Alan Programlanabilir Kapı Dizisi
GPU : Grafik İşlem Birimi
J : Joule
kWh : Kilowatt
LCA : Yaşam Döngüsü Analizi
MAC : Bellek Erişim Maliyeti
MÖ : Makine Öğrenme
NPU : Sinirsel İşleme Birimi
PC : Kişisel Bilgisayar
PSO : Parçacık Sürüsü Optimizasyonu
S : Saniye
SBC : Tek Kartlı Bilgisayar
SI : Uluslararası Birimler Sistemi
SoC : Çip Üzerinde Sistem
TPU : Tensör İşleme Birimi
W : Watt
YZ : Yapay Zeka

SİMGELER

CO_2 : Karbon dioksit

gCO2eq **:** Gram Karbon Dioksit Eşdeğeri

kgCO2eq **:** Kilogram Karbon Dioksit Eşdeğeri

TABLO LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

DİJİTAL KARBON AYAK İZİNİN OPTİMİZASYONU İÇİN UÇ CİHAZLARDA YAPAY ZEKA VE MAKİNE ÖĞRENMESİ UYGULAMALARI

ÖZET

Günümüzde, yapay zeka (YZ) uygulamaları hayatımızın birçok alanına derinlemesine nüfuz etmiş durumdadır ve bu durum, özellikle teknolojinin hızlı bir şekilde gelişmesi ile, birçok farklı sektörde önemli değişiklikler ve yenilikler getirmiştir. Özellikle, nesnelerin interneti (Internet of Things -IoT) cihazları ile entegre edilen derin öğrenme (DÖ) modellerinin etkileri, akıllı evler, akıllı şehirler, sağlık hizmetleri, ulaşım sistemleri ve endüstriyel otomasyon gibi çeşitli alanlarda yaygın bir şekilde hissedilmektedir. Bu gelişmeler, bu alanlarda büyük bir ilerleme kaydedilmesini sağlamış ve aynı zamanda enerji verimliliği, otomasyon ve zengin kullanıcı deneyimi açısından önemli avantajlar sunmuştur. Ancak, bu teknolojik ilerlemelerin beraberinde bazı çevresel etkileri de getirdiği gözlemlenmiştir. Özellikle, IoT cihazları üzerinde çalıştırılan derin öğrenme modellerinin enerji tüketimi ve bu durumun dolayısıyla yaratmış olduğu karbon ayak izi, son yıllarda dikkat çekici bir endişe kaynağı haline gelmiş ve bu konuda çeşitli araştırmalar yapılması gerekliliği ortaya çıkmıştır.

Bu tez çalışması, Raspberry Pi 5 (CPU), NVIDIA Jetson Xavier NX (GPU), Google Coral USB Accelerator (TPU) ve Khadas VIM3 Pro (NPU) gibi birbirinden farklı ve oldukça ilginç donanım mimarilerine sahip toplamda dört farklı IoT platformunda, MobileNetV2, ShuffleNetv2, SqueezeNet ve ResNet18 gibi son derece popüler ve yaygın bir şekilde kullanılan derin öğrenme modellerinin enerji tüketimlerinin yanı sıra, bu modellerin performans metrikleri olarak gecikme süreleri ile doğruluk derecelerinin kapsamlı bir biçimde karşılaştırmalı incelemesini ve değerlendirilmesini hedeflemektedir. Ayrıca bu çalışma, bu derin öğrenme modellerinin enerji tüketimindeki değişimlerin yanı sıra karbon ayak izinin de nasıl etkilendiği konusunda derinlemesine bir inceleme yapacaktır. Bu araştırmanın önemli bir bileşeni olarak, quantization, pruning ve knowledge distillation gibi çeşitli optimizasyon tekniklerinin bu modellerin enerji tüketimi ve karbon ayak izi üzerindeki etkileri, gerçek yaşamdan alınan bir örnek olan akıllı ev uygulaması çerçevesinde (özel olarak enerji tüketiminin izlenmesi) değerlendirilecektir.

Optimizasyon sürecinde kullanılan yöntemler arasında niceleme (quantization), budama (pruning), bilgi damıtma (knowledge distillation) ve ağırlık kümeleme (weight clustering) gibi çeşitli etkili teknikler bulunmaktadır ve bu stratejilerin uygulanmasıyla enerji verimliliği sağlanması amaçlanmaktadır. Bu tekniklerin enerji tüketimi, işlem süresi ve model doğruluğu üzerindeki etkileri son derece detaylı bir biçimde ele alınmış olup, en iyi sonuçların ışığında IoT cihazlarında enerji verimliliği ile yüksek performans arasında en uygun dengeyi sağlamak maksadıyla kapsamlı bir analiz gerçekleştirilmiştir. Bu sayede elde edilen veriler hem akademik hem de endüstriyel uygulamalar için önemli çıktılar sunmayı hedeflemektedir.

NPU, TPU ve GPU gibi donanımlarda uygulanan optimizasyon tekniklerinin etkileri karşılaştırmalı olarak sunulmuş ve çevresel etkiler açısından en etkili çözüm önerileri

derinlemesine tartışılmıştır. Çalışmanın sonuçları, yapay zeka modellerinin hem performanstan ödün vermeden hem de enerji verimliliği sağlayarak karbon ayak izini azaltabileceğini açık bir şekilde göstermektedir. Böylece, IoT cihazlarının kullanımındaki çevresel etkiler minimize edilerek, sürdürülebilir bir dijital geleceğe ulaşmak için önemli bir adım atılmış olacaktır ve bu durum, teknoloji ve çevre dostu yaklaşımlar arasında bir denge kurmak adına kritik bir öneme sahiptir.

Anahtar Kelimeler: Dijital Karbon Ayak İzi, Enerji Tüketimi, Derin Öğrenme, Görüntü Sınıflandırma, Uç Cihazlar, Model Optimizasyonu, Budama, Niceleme, MobileNetV2, ShuffleNetv2, SqueezeNet, ResNet18, Raspberry Pi 5, CPU, NVIDIA Jetson Xavier NX, GPU, Google Coral USB Accelerator, TPU Khadas VIM3 Pro, NPU.

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING APPLICATİONS ON EDGE DEVICES FOR DIGITAL CARBON FOOTPRINT OPTİMİZATİON

SUMMARY

The advent of artificial intelligence (AI) applications has profoundly impacted numerous facets of human existence, precipitating substantial transformations and innovations across diverse sectors, particularly in the context of rapid technological advancements.

The integration of deep learning (DL) models with Internet of Things (IoT) devices has engendered widespread and profound effects, manifesting in numerous domains such as smart homes, innovative smart cities, sophisticated healthcare systems, advanced transportation networks, and revolutionary industrial automation processes. These remarkable advancements have enabled substantial progress across these varied domains, offering significant advantages in terms of enhanced energy efficiency, seamless automation, and a greatly improved user experience that elegantly adapts to the specific needs and preferences of individuals. This dynamic and synergistic relationship between DL and IoT is fundamentally revolutionizing how we interact with technology in our daily lives, making our environments smarter and more responsive.

However, these notable technological advances have also unfortunately been accompanied by a variety of significant environmental impacts that cannot be overlooked. Specifically, the substantial energy consumption associated with deep learning models that are being executed on Internet of Things (IoT) devices and the resulting carbon footprint produced have become a major concern in recent years, drawing the attention of researchers and policymakers alike. This pressing issue has prompted numerous research studies aimed at understanding and mitigating these adverse effects, as addressing such challenges is essential for the sustainability and longevity of these advancements in technology.

The primary objective of this thesis is to conduct a comprehensive comparative study and evaluation focused on the energy consumption, performance metrics (specifically latency and accuracy), and the overall carbon footprint associated with popular deep learning models that are widely used within the industry. These models include MobileNetV2, ShuffleNetV2, SqueezeNet, and ResNet18. The evaluation will be meticulously performed on four distinct IoT platforms, each representing different hardware architectures that vary in capability, such as the Raspberry Pi 5 (which utilizes a CPU), the NVIDIA Jetson Xavier NX (designed with a powerful GPU), the Google Coral USB Accelerator (featuring an efficient TPU), and the Khadas VIM3 Pro (equipped with a specialized NPU).

Furthermore, this research will delve into a thorough understanding of the impact of various optimization techniques that can significantly enhance the performance and sustainability of these models. These techniques will include quantization, pruning,

and knowledge distillation, with a specific focus on how they systematically affect the energy consumption and the carbon footprint associated with the aforementioned models. This rigorous assessment will be conducted within the practical context of a real-world smart home application, particularly emphasizing sophisticated energy consumption monitoring systems designed to improve sustainability.

By integrating these techniques and evaluating their effects in detail, this study aims to provide valuable insights into the optimization of deep learning models for sustainability and efficiency within increasingly prevalent IoT environments. The primary goal of this comprehensive study is not only to thoroughly benchmark but also to critically evaluate the energy consumption of widely utilized deep learning models, such as SqueezeNet and ResNet18, alongside their associated latency and accuracy as essential performance metrics.

Moreover, this extensive study will also investigate how these prominent deep learning models are influenced by variations in energy consumption and their corresponding carbon footprint, which is becoming an ever-important concern in today's environmentally conscious world. A critical and significant facet of this research further entails the evaluation and comparison of a variety of powerful optimization techniques, including but not limited to quantization, pruning, and knowledge distillation.

This will all be done in the context of a practical, real-world example: a smart home application that is specifically focused on monitoring energy consumption and optimizing usage patterns to foster sustainability and efficient resource management.

The optimization process harnesses a variety of effective techniques which include quantization, pruning, knowledge distillation, and weight clustering. The primary implementation of these strategies is specifically aimed at achieving a markedly greater level of energy efficiency. Furthermore, a thorough and comprehensive analysis has been meticulously conducted to explore the wide-ranging impact of these optimization techniques on energy consumption, processing time, as well as model accuracy.

This in-depth analysis has been undertaken to identify the most optimal balance between energy efficiency and high performance in Internet of Things (IoT) devices, aiming to elucidate the interplay between computational demands and environmental responsibilities. The data obtained through these advanced methods is anticipated to provide significant and valuable insights, which will be beneficial for both academic researchers and industrial applications alike, fostering a collaborative effort toward a sustainable technological future.

This study presents a comparative analysis of the effects of optimization techniques applied to hardware components such as NPU, TPU, and GPU. A comprehensive discussion is provided on the most effective solutions in terms of environmental impacts.

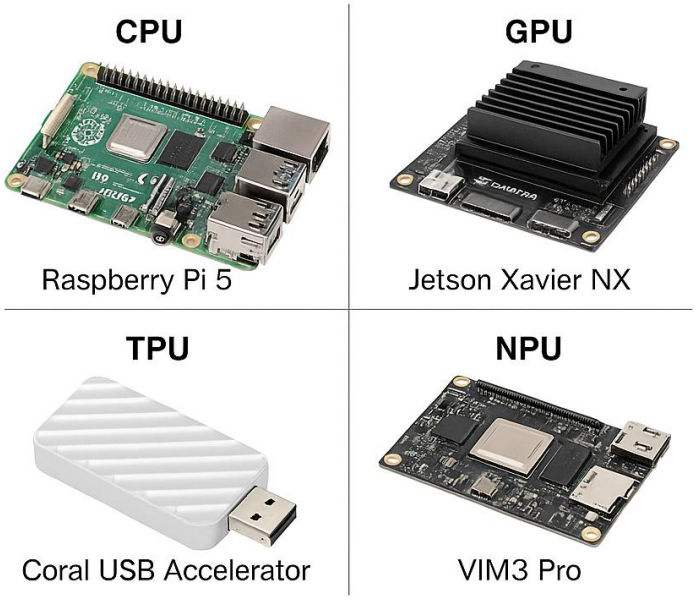
The study's findings are conclusive in demonstrating that artificial intelligence models can effectively reduce the carbon footprint without compromising performance and energy efficiency. Consequently, this study underscores the significance of reducing the environmental impact of IoT devices, thereby paving the way towards a sustainable digital future. Achieving this balance between technological advancement and environmentally friendly approaches is imperative.

Keywords: Digital Carbon Footprint, Energy Consumption, Deep Learning, Image Classification, Edge Devices, Model Optimization, Pruning, Quantification, MobileNetV2, ShuffleNetv2, SqueezeNet, ResNet18, Raspberry Pi 5, CPU, NVIDIA Jetson Xavier NX, GPU, Google Coral USB Accelerator, TPU Khadas VIM3 Pro, NPU.

1. GİRİŞ

Bu tez, dijital karbon ayak izinin optimizasyonuna yönelik olarak gerçekleştirilmiş olan yapay zeka ve makine öğrenmesi uygulamalarının derinlemesine ve kapsamlı bir biçimde incelemektedir. Çalışma, ilgili literatüre katkı sunmayı amaçlamaktadır. Tezin önemi, IoT platformlarındaki farklı donanım mimarilerinde çalışan derin öğrenme modellerinin enerji tüketimi, performansı ve karbon ayak izini sistematik ve karşılaştırmalı olarak analiz etmesinden kaynaklanmaktadır. Bu analiz, donanım ve yazılım optimizasyonlarının çevresel etkilerini ortaya koymayı hedeflemektedir.

Tezde sunulan yaklaşımlar, mevcut uygulamalarda karşılaşılan sorunların çözümüne katkı sağlamayı hedefler. Bu çalışmanın temel amacı; Raspberry Pi 5 (CPU), NVIDIA Jetson Xavier NX (GPU), Google Coral USB Accelerator (TPU) ve Khadas VIM3 Pro (NPU) gibi farklı donanımlarda, derin öğrenme modellerine uygulanan optimizasyon tekniklerinin etkinliğini araştırmaktır. Farklı Uç Cihaz Donanım Platformları (CPU, GPU, TPU, NPU)örneği Şekil 1.1de gösterilmiştir.



Şekil 1.1. Farklı Uç Cihaz Donanım Platformları: CPU, GPU, TPU, NPU (Kaynak: Banbury ve diğerleri(2020), Chen ve diğerleri (2022), Upton (2023)).

Böylece, farklı donanım platformlarında optimizasyon stratejilerinin performansı değerlendirilerek pratik çıkarımlar elde edilmesi amaçlanmaktadır.

Tezin kapsamı, enerji verimliliği ile birlikte tüketim analizleri, performans analizleri ve karbon ayak izi değerlendirmelerini içerecek şekilde genişletilmiştir. Bu sayede, konuya dair bütüncül bir bakış açısı sunulması hedeflenmektedir. Çalışmanın bir diğer katkısı, uç cihazlarda dijital karbon ayak izinin azaltılmasına yönelik sürdürülebilir yöntem ve stratejilerin geliştirilmesine odaklanmasıdır. Bu yaklaşımların, alana katkı sağlaması amaçlanmaktadır.

Tezin, optimizasyon tekniklerinin uygulanabilirliğine dair somut öneriler sunması planlanmaktadır. Bu durumun, araştırma sonuçlarının geniş bir alanda uygulanabilirliğini artırması ve gelecekteki çalışmalara referans oluşturması beklenmektedir. Ayrıca, elde edilen bulguların güncel teknolojiler ve küresel çevresel sürdürülebilirlik bağlamındaki etkileri incelenecektir. Bu analizler, verilerin anlamını ve uygulama potansiyelini ortaya koyarak teorik ve pratik bilgi birikimine katkı sağlayacaktır. Çalışmada sunulan verilerin, çevresel etkileri azaltma stratejileri için bir temel oluşturması hedeflenmektedir.

Araştırma, derin öğrenme alanındaki disiplinler arası iş birliklerinin etkinliğini ve bu iş birliklerinden doğacak çözümlerin karbon ayak izini azaltmadaki potansiyelini de ele alacaktır. Bu yaklaşımın, akademik çalışmalara ve sanayi uygulamalarına katkı sağlaması hedeflenmektedir. Ayrıca, modelleme ve simülasyon teknikleriyle elde edilen sonuçlar, farklı senaryolar altında analiz edilerek uygulanabilir çözümler üretilmesine yardımcı olacaktır. Elde edilen veriler ışığında, sürdürülebilir teknoloji geliştirme amacıyla yeni stratejilerin oluşturulması desteklenecektir. Böylece, araştırma sürdürülebilirlik ilkeleri doğrultusunda topluma ve çevreye olumlu etkiler yaratabilecek çözümler sunmayı amaçlamaktadır.

1.1. Tezin Önemi

Bu tez çalışması, IoT platformlarında yapay zeka ve makine öğrenmesi uygulamalarıyla dijital karbon ayak izinin optimizasyonunu disiplinler arası bir yaklaşımla incelemektedir. Çalışmanın önemi, farklı donanım tasarımları ve derin öğrenme modellerini enerji kullanımı, performans ve karbon ayak izi açısından karşılaştırmalı olarak analiz etmesinden kaynaklanmaktadır. Bu analiz, dijital karbon

ayak izi üzerine yapılacak gelecek çalışmalara veri sağlamayı ve etkin optimizasyon tekniklerinin belirlenmesine katkıda bulunmayı hedeflemektedir.

Çalışma, mevcut durumu değerlendirmenin ötesinde, ileri araştırmalara yön vererek ve bu alanda bilgiler sunarak disiplinler arası yaklaşımlar geliştirme potansiyeli taşımaktadır. Araştırma sonuçlarının, akademik dünyaya yeni bakış açıları sunması ve uygulamalı alandaki profesyonellerin karar alma süreçlerine rehberlik etmesi beklenmektedir. Bu çalışma, dijital karbon ayak izinin azaltılması konusunda bilgi ve yöntemler sunarak çevresel sürdürülebilirliğe katkı sağlamaya odaklanmaktadır. Elde edilecek bulguların, politika yapıcılar, endüstri uzmanları ve akademisyenler için değer taşıyacağı ve daha sürdürülebilir bir gelecek için stratejiler geliştirilmesine yardımcı olacağı öngörülmektedir.

1.2. Tezin Amacı

Bu tezin temel amacı, uç cihazlarda yapay zeka ve makine öğrenmesi uygulamalarının verimli kullanımı yoluyla dijital karbon ayak izini azaltmaktır. Bu hedefe ulaşmak için, derin öğrenme modelleri, IoT platformları ve çeşitli işlemci birimleri (NPU, TPU, GPU, CPU) karşılaştırmalı olarak incelenecek ve bu teknolojilerin işleyiş biçimleri analiz edilecektir. Enerji tüketiminin izlenmesi ve performans ölçüm yöntemlerinin belirlenmesiyle, uç cihazlarda dijital karbon ayak izi optimizasyonu için etkili ve sürdürülebilir teknikler geliştirmek tezin önemli amaçları arasındadır. Bu kapsamda, mevcut durumun değerlendirilmesi ve gelecekteki gelişmeler dikkate alınarak sürdürülebilir çözümler üretmeye yönelik stratejiler geliştirilmesi hedeflenmektedir. Yapay zeka ve makine öğrenmesi uygulamalarının daha verimli ve sürdürülebilir hale getirilmesine yönelik analizler, araştırmanın ana bileşenlerindendir. Sistemlerin enerji verimliliğini artıracak, veri işleme süreçlerini optimize edecek ve karbon ayak izini azaltacak yöntemlerin belirlenmesi, endüstriyel ve çevresel faydalar sağlamayı amaçlamaktadır. Bulguların hem akademik literatüre katkı sağlaması hem de pratik uygulamalarda kullanılabilir olması beklenmektedir. Bu çalışma, çevre dostu yaklaşımlar ile sürdürülebilir kalkınma hedefleri doğrultusunda katkılar sunmayı amaçlamaktadır.

1.3. Tezin Kapsamı

Bu tez çalışması, uç cihazlarda yapay zeka ve makine öğrenmesi uygulamalarının karbon ayak izi optimizasyonunu inceleyecektir. Yapay zeka ve makine öğrenmesi modellerinin uç cihazlarda etkin ve verimli kullanımı analiz edilecek, bu alandaki güncel bilgiler ve yöntemler dikkate alınacaktır. Bu süreçlerde enerji tüketimi ve performans kriterleri de ele alınacaktır. Derin öğrenme modellerinin seçimi, veri setlerinin oluşturulması, tahmini çalışmaların planlanması ve enerji tüketimi ile performans ölçümleri için metodolojiler ve araçlar hakkında bilgiler sunulacaktır.

Çalışmanın temel kapsamı, uç cihazlarda dijital karbon ayak izi değerlendirmeleri yapmak ve optimizasyon teknikleri üzerine analizler sunmaktır. Bu çerçevede, yapay zeka ve makine öğrenmesi teknolojilerinin enerji verimliliğine katkısı, sistem performansını artırma ve çevresel etkileri azaltma yolları araştırılacaktır. Bu araştırma, sürdürülebilir enerji çözümleri ve verimli stratejiler geliştirilmesine katkı sağlayacaktır. Elde edilen gelişmelerin sektöre ve çevreye katkılarının anlaşılması ve sürdürülebilir çözümler önerilmesi hedeflenmektedir.

Bu çalışmalar, modern teknoloji ile çevre koruma arasında bir denge kurmayı hedefleyen bir yaklaşım sunacaktır. Araştırmada elde edilen sonuçların pratik uygulamaları ve gelecekteki potansiyel gelişim alanları da irdelenecektir. Uç cihazlardaki incelemelerin yanı sıra, yapay zeka ve makine öğrenmesi iş birliklerinin sürdürülebilirlik açısından getirdiği yaklaşımlar da değerlendirilecektir. Uygulanan yöntemlerin etkinliği, enerji maliyetlerini azaltma potansiyeli ve çevresel-ekonomik faydaları üzerinde durulacaktır. Böylece, çalışmanın akademik ve endüstriyel farkındalığa katkı sağlaması amaçlanmaktadır.

Tezin bulguları, taşınabilir enerji sistemlerinin entegrasyonu hakkında da bilgiler sunacaktır. Bu çalışmanın çıktılarının akademik alanda ve endüstriyel uygulamalarda kullanılabilir olması hedeflenmektedir. Bu araştırma, yeni teknolojilerin entegrasyonu ile çevresel sürdürülebilirliğe katkı sağlayacak çözümlerin geliştirilmesine olanak tanıyacaktır. Dolayısıyla, tez kapsamı, teorik analizlerden pratik uygulamalara ve sürdürülebilir teknoloji gelişimine uzanan geniş bir yelpazeyi kapsamaktadır.

1.4. Tezin Katkısı

Bu tez, uç cihazlarda yapay zeka ve makine öğrenmesi uygulamalarının optimizasyonu konusunda katkılar sunmakta ve ilgili literatüre ışık tutmaktadır. Çalışma, derin öğrenme modellerinin IoT platformlarındaki performansını ve enerji tüketimini analiz ederek, uç cihazlarda dijital karbon ayak izinin nasıl optimize edilebileceğini ortaya koymaktadır. Ayrıca, donanım mimarileri, model seçimleri, veri setlerinin kullanımı ve enerji tüketimi ölçümlerine dair metodolojik yaklaşımlar geliştirerek karar verme süreçlerine katkıda bulunmaktadır. Bu analizler, teorik bilgi ve pratik uygulama sunarak endüstriyel uygulamalara potansiyel katkılar sağlamaktadır.

Çalışmanın sonuçları, çeşitli endüstriyel sektörlerde yapay zeka alanında uygulanabilecek stratejiler ve çözümlerin geliştirilmesi için bir referans noktası oluşturabilir. Akademik araştırmacılara ve uygulayıcılara bilgiler sunularak, teknoloji gelişimiyle ilgili stratejik öneriler ve pratik çözümler belirlenmektedir. Uç cihazlarda yapay zeka ve makine öğrenmesi uygulamaları; yerel veri işleme, enerji verimliliği ve sistem güvenilirliği gibi konularda bulgular sunarak yeni uygulamaların geliştirilmesine zemin hazırlayabilir. Bu durum, gelecekteki araştırmalar ve uygulamalar için bir vizyon oluşturarak sürdürülebilirlik hedeflerine katkı sağlamaktadır. Çalışmanın temel bulgularının, ileri düzey araştırmalara ve pratik uygulamalara ışık tutarak sektördeki gelişmeleri yönlendirmesi beklenmektedir.

1.5. Tezin Organizasyonu

Bu tez çalışması beş ana bölümden oluşmaktadır.

Birinci bölümde, giriş kısmında tezin önemi, amacı, kapsamı ve bilimsel katkısı açıklanmaktadır.

İkinci bölümde, yapay zeka ve makine öğrenmesi tarihçesi, derin öğrenme modelleri ve IoT platformları üzerine güncel çalışmaları içeren geniş bir literatür taraması sunulmaktadır. Bu bölümde ayrıca, uç cihazlarda derin öğrenme uygulamaları, zorlukları ve NPU, TPU, GPU, CPU karşılaştırmaları ile enerji tüketimi, performans ve dijital karbon ayak izi analizleri yer almaktadır.

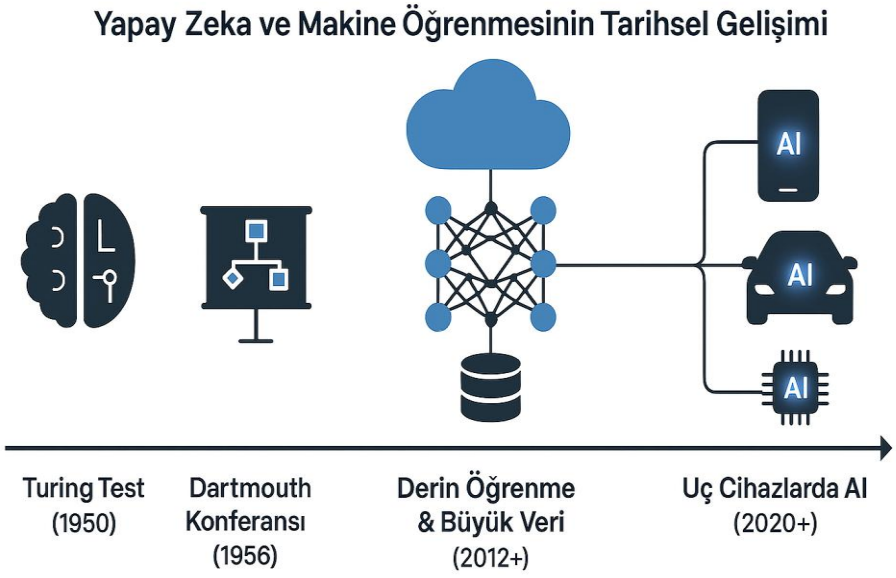
Üçüncü bölümde, metodoloji başlığı altında donanım mimarileri, derin öğrenme modellerinin seçimi, veri setleri, tahmini çalışmaların planlanması ve enerji tüketimi ile performans ölçüm yöntemleri detaylandırılmaktadır.

Dördüncü bölümde, bulgular kısmında enerji tüketimi karşılaştırmaları, performans analizleri ve karbon ayak izi değerlendirmeleri somut verilerle sunulmaktadır.

Beşinci ve son bölümde ise çalışmanın genel sonuçları değerlendirilmekte; optimizasyon tekniklerinin etkinliği, uygulanabilirliği, gelecekteki çalışmalara olası etkileri ve öneriler tartışılmaktadır. Bu yapı, araştırmanın mantıksal bir akışla sunulmasını sağlamaktadır.

2. LİTERATÜR TARAMASI

Bu bölümde, tezin teorik altyapısını oluşturan temel konulara ilişkin literatür incelenecektir. İlk olarak, yapay zeka ve makine öğrenmesinin tarihsel gelişimi ele alınarak temel kavramlar ve önemli dönüm noktaları değerlendirilecektir. Ardından, derin öğrenmenin prensipleri, uygulama alanları ve özellikle Nesnelerin İnterneti (IoT) platformlarındaki mevcut çalışmaları incelenmiştir. Uç cihazlarda derin öğrenme uygulamaları, karşılaşılan zorluklar, farklı işlem birimlerinin (NPU, TPU, GPU, CPU) karşılaştırmalı analizleri ile enerji tüketimi ve performans değerlendirmeleri literatürdeki güncel çalışmalar ışığında tartışılacaktır. Son olarak, dijital karbon ayak izi konusundaki mevcut araştırmalar ve derin öğrenme uygulamalarının bu izi optimize etmedeki rolüne dair literatürdeki yaklaşımlar ele alınacaktır. Bu incelemeler, tezin araştırma sorularına ve metodolojisine zemin hazırlayacaktır (Bağcı, 2024).



Şekil 2.1. Yapay Zeka ve Makine Öğrenmesinin Tarihsel Gelişimi (Kaynak: Russell & Norvig, 2021; Bengio et al., 2015).

Yapay zekanın tarihi, bilgisayar bilimlerinin evriminde önemli bir yer tutmakta ve on yıllardır süregelen bir ilgiyle takip edilmektedir. Yapay zekanın gelişim süreci,

bilgisayarların insan benzeri zeka özelliklerini taklit etme potansiyelinin araştırıldığı 1950'lerde başlamıştır. Bu dönemde, makinelerin bilişsel yeteneklerini artırmaya yönelik teknikler incelenmiş ve yapay zekanın temel prensipleri (McCarthy vd., 1956) ortaya konmuştur.

Alan ilerledikçe, yapay zeka araştırmalarına olan ilgi artmış ve farklı disiplinlere yayılmıştır. Makine öğrenmesinin 1980'lerin başında ortaya çıkışı, bilgisayar sistemlerinin büyük veri kümeleri üzerinden öğrenme yeteneklerini geliştirmeye odaklanmasıyla önemli bir dönüm noktası olmuştur (Bishop, 2006). Bu gelişmeler, YZ ve makine öğrenmesinin seyrini belirleyerek günümüzdeki derin öğrenme modellerine ve uygulamalarına zemin hazırlamıştır (LeCun vd., 2015). Bu tarihsel ilerleme, YZ'nin modern teknoloji ve toplum üzerindeki etkisini vurgulamaktadır. Bu tarihsel bağlam, mevcut tez çalışmasının odaklandığı yapay zeka ve derin öğrenme uygulamalarının kökenlerini ve evrimini anlamak açısından önemlidir; zira bu teknolojilerin enerji tüketimi ve karbon ayak izi gibi güncel sorunları, bu uzun evrimin getirdiği yetenekler ve karmaşıklıkla doğrudan ilişkilidir.

2.1. Yapay Zeka ve Makine Öğrenmesi Tarihçesi

Yapay zeka (YZ) kavramı, insan zekasını taklit etme yeteneğine sahip makineler geliştirme düşüncesiyle 1950'li yıllara dayanmaktadır. McCarthy ve arkadaşlarının (1955) düzenlediği Dartmouth Konferansı, yapay zeka disiplininin formel başlangıcı olarak görülmektedir. İlk dönem YZ araştırmaları, ağırlıklı olarak mantıksal çıkarım ve problem çözme yeteneklerine odaklanmıştır (Russell & Norvig, 2010). Bu temel başlangıç noktaları ve ilk odak alanları, günümüzde enerji verimliliği ve karbon ayak izi gibi karmaşık problemlerin çözümünde kullanılacak YZ yaklaşımlarının temelini oluşturması bakımından bu tez için önemlidir.

Makine öğrenmesi (MÖ), yapay zekanın bir alt disiplini olarak, bilgisayarların veriye dayalı öğrenme yeteneği kazanmasını sağlayan algoritmaların geliştirilmesiyle ilgilenir; bu süreçte sistemler açıkça programlanmaya ihtiyaç duymaz. Samuel'in (1959) dama oynayan programı, makine öğrenmesinin erken dönemdeki dikkate değer başarılarından biri olarak kabul edilir. 1980'li yıllarda ise karar ağaçları ve kural tabanlı sistemler gibi algoritmalar yaygınlık kazanmıştır (Quinlan, 1986). MÖ'nün bu evrimi, veriden öğrenme yeteneği sayesinde, tezin incelediği derin öğrenme

modellerinin ve bu modellerin enerji optimizasyonu potansiyelinin anlaşılması için kritik bir arka plan sunmaktadır.

1990'lı ve 2000'li yıllarda, destek vektör makineleri (Support Vector Machine (SVM)) ve yapay sinir ağları (YSA) gibi daha gelişmiş algoritmalar ortaya çıkmıştır (Cortes & Vapnik, 1995; Haykin, 1999). Yakın dönemde ise büyük veri setlerinin erişilebilirliği ve artan donanım kapasiteleri, derin öğrenme tekniklerinin hızla yaygınlaşmasını sağlamıştır (LeCun, Bengio, & Hinton, 2015). Bu gelişmeler, özellikle YSA'ların evrilmesiyle ortaya çıkan derin öğrenmenin yükselişi, tezimizin temel konusu olan uç cihazlarda derin öğrenme modellerinin enerji tüketimi ve karbon ayak izi optimizasyonu çalışmalarına doğrudan zemin hazırlamıştır.

2.2. Derin Öğrenmeye Giriş

Derin öğrenmenin temel gücü, büyük veri hacimlerinden otomatik olarak hiyerarşik özellik temsilleri çıkarabilme ve çoklu soyutlama seviyelerinde temsiller öğrenebilme (temsil öğrenimi) kapasitesine dayanmaktadır. Geleneksel makine öğrenmesi yaklaşımları genellikle manuel olarak tasarlanan özelliklere bağımlıyken, derin öğrenme modelleri bu özellikleri doğrudan veriden öğrenir (Bengio, 2009). Bu yetenek, derin öğrenmeyi görüntü tanıma, doğal dil işleme, konuşma tanıma (Goodfellow, Bengio, & Courville, 2016) ve uzaktan algılama (LeCun vd., 2015; Yang vd., 2020) gibi çeşitli ve karmaşık görevlerde oldukça başarılı kılmıştır. Farklı uygulama alanlarındaki bu yaygın etki, derin öğrenmenin büyük ve yüksek boyutlu verileri işleyerek anlamlı bilgiler ve özellikler üretebilmesinden kaynaklanmaktadır.

Teorik ve pratik alandaki son gelişmeler, derin öğrenmenin farklı öğrenme görevlerinde güçlü genelleme yetenekleri sergilediğini de ortaya koymaktadır (Zhou vd., 2021). Bu genelleme kapasitesi, modellerin daha önce karşılaşmadığı verilere veya farklı senaryolara uyum sağlama potansiyelini artırır; bu da özellikle çeşitli veri kümelerinin analiz edildiği ve farklı kaynaklardan gelen bilgilerin sentezlendiği durumlarda önem kazanmaktadır.

Derin öğrenmenin otomatik özellik çıkarma, temsil öğrenimi ve genelleme gibi temel yetenekleri, bu tez çalışmasının odaklandığı uç cihazlarda yapay zeka uygulamaları için hem büyük bir potansiyel sunmakta hem de önemli zorlukları beraberinde getirmektedir. Modellerin karmaşık verilerden öğrenebilme ve farklı durumlara uyum sağlayabilme kabiliyeti, uç cihazlardaki sınırlı hesaplama kaynakları, enerji tüketimi

ve sonuç olarak dijital karbon ayak izi bağlamında dikkatli bir optimizasyon gerektirmektedir. Dolayısıyla, derin öğrenmenin bu temel prensiplerini anlamak, tezde ele alınacak optimizasyon stratejilerinin ve çevresel etki analizlerinin temelini oluşturmaktadır.

Derin öğrenme modellerinin Nesnelerin İnterneti (IoT) platformlarına entegrasyonu ve bu ortamlardaki özgün zorlukların aşılması, güncel araştırmaların önemli bir odak noktasını oluşturmaktadır. Bu alandaki çalışmalarda, farklı derin öğrenme mimarilerinin IoT problemlerini çözmedeki etkinliği çeşitli performans metrikleri kullanılarak değerlendirilmektedir (Khan vd., 2021). Örneğin, Zhou vd. (2022) tarafından yapılan araştırmalar, evrişimsel sinir ağları (Convolutional Neural Network (CNN)) gibi modellerin IoT ortamlarında, özellikle tıbbi uygulamalar bağlamında, diğer makine öğrenmesi algoritmalarına kıyasla etkili sınıflandırma performansı sunduğunu ve genel olarak makine öğrenmesi yaklaşımlarının IoT düğümlerinde enerji verimliliğini artırabildiğini göstermiştir. Benzer şekilde, enerji verimliliğine odaklanan bir diğer çalışmada Mansouri vd. (2023), WirelessHART protokolü üzerinde parçacık sürüsü optimizasyonu (PSO) gibi algoritmaların enerji kullanımında iyileştirmeler sağladığını ortaya koymuştur. Bu tür araştırmalar, makine öğrenmesi teknolojilerinin IoT ağlarında enerji verimliliğini artırma ve kaynak yönetimini optimize etme potansiyelini vurgulamaktadır.

Bu literatür bulguları, IoT ve uç cihaz ortamlarında enerji verimliliğinin kritik bir araştırma konusu olduğunu ve makine öğrenmesi/derin öğrenme tabanlı optimizasyonların bu alanda önemli faydalar sunabileceğini göstermektedir. Dolayısıyla, bu tez çalışmasının farklı donanım platformlarında (CPU, GPU, TPU, NPU) enerji tüketimi, performans ve sonuç olarak dijital karbon ayak izi karşılaştırmalarına odaklanması, literatürdeki bu eğilimlerle uyumlu ve güncel bir araştırma alanına katkı sunma potansiyeli taşımaktadır. Mevcut çalışmalar genellikle belirli algoritmaların veya protokollerin enerji verimliliğini incelerken, bu tez daha geniş bir donanım yelpazesinde ve farklı optimizasyon teknikleriyle bu etkileşimleri sistematik olarak araştırmayı hedefleyerek literatüre farklı bir perspektif getirmeyi amaçlamaktadır.

2.3. Uç Cihazlarda Derin Öğrenme Uygulamaları

Akıllı şehirler, trafik akışını optimize etmek, hava kalitesini izlemek ve kamu güvenliğini artırmak için derin öğrenme modelleri kullanılabilir (Batty, 2013).

Akıllı evler, derin öğrenme, enerji tüketimini izlemek, cihazları kontrol etmek ve güvenlik sistemlerini geliştirmek için kullanılabilir (Aghdam & Heravi, 2017).

Sağlık hizmetleri, derin öğrenme, tıbbi görüntüleri analiz etmek, hastalıkları teşhis etmek ve kişiselleştirilmiş tedavi planları geliştirmek için kullanılabilir (Esteva ve diğerleri, 2017). Endüstriyel otomasyon: Kestirimci bakım, kalite kontrol ve robotik uygulamalarında derin öğrenme modellerinden yararlanılabilir (Wang, 2018).

Uç cihazlarda derin öğrenme, akıllı evler, akıllı şehirler, sağlık hizmetleri ve endüstriyel otomasyon gibi çeşitli alanlarda uygulanmaktadır (Zhou ve diğerleri, 2019). Ayrıca, enerji tüketiminin optimize edilmesi, bu tür yüksek teknolojiye sahip uygulamaların mobil cihazlar üzerinde daha etkili ve sürdürülebilir bir şekilde kullanılmasına olanak sağlamaktadır (Öztürk, 2020).

Bulut bilişimin özellikle çeşitli sektörlerde uygulanması önemli ölçüde şüpheyle karşılanmaktadır. Temel endişeler, uç cihazların zekasını ve işlevselliğini artırmanın önünde önemli engeller olarak görülen veri hacmi, veri hareketliliği ve gizlilik endişelerini kapsamaktadır. Yapay zekanın (YZ) edge teknolojisine entegrasyonu, veri hacmiyle ilgili sorunları ele almak ve Nesnelerin İnterneti (IoT) bağlamında güç tüketimini azaltmak için gereklidir. Bu teknolojik ilerleme, yerel uç cihazlara gelen veri akışının yanı sıra YZ iş yüklerinin eşzamanlı olarak işlenmesini kolaylaştırarak operasyonel verimliliği artırmaktadır (Zhang vd., 2020).

Üst düzey gerçek dünya cihazlarında hata önlemenin önemi, potansiyel risklerin hızlı bir şekilde tespit edilmesi ve bunlara müdahale edilmesinde, daha güvenli operasyonların teşvik edilmesinde ve daha tutarlı sonuçlar elde edilmesinde önemli bir rol oynadığı için kritiktir (Li vd., 2021). Bu gelişmelerin sonuçları, uç yapay zeka ve IoT ekosistemindeki uygulamaları alanında daha fazla keşif ve araştırma yapılması gerekliliğinin altını çizmektedir.

Geleneksel olarak, IoT cihazlarının yetenekleri sınırlı olarak algılanmakta, genellikle gelişmiş yapay zeka ve makine öğrenimi prosedürlerini yürütmek için uygun olmadığı düşünülmekte ve bu nedenle önemli zorluklar ortaya çıkmaktadır. Bu sınırlama, IoT cihazlarını ve bileşenlerini, operasyonel işlevselliği geliştirmek için tasarlanmış

sofistike makine öğrenimi algoritmalarıyla donatma çabalarını zorlaştırmaktadır (Mishra vd., 2022).

Uç cihazlarda derin öğrenme uygulamaları, yapay zeka ve makine öğrenmesi algoritmalarının mobil cihazlarda kullanımını ifade eden önemli bir alan olarak ortaya çıkmaktadır. Bu tür uygulamaların uygulanması, mobil cihazların sınırlı işlem gücü ve bellek kapasiteleri nedeniyle genellikle karmaşık bir süreç olarak değerlendirilmiştir (Yalçın, 2022). Ancak, teknoloji alanındaki hızlı ve sürekli gelişmeler sayesinde günümüzde mobil cihazlar, daha önce mümkün olmayan derin öğrenme uygulamalarını destekleyecek kapasiteye ve yetkinliğe ulaşmıştır. Örnek vermek gerekirse, günümüzde mobil cihazlar üzerinde görüntü tanıma, sesli komutlarla konuşma tanıma ve doğal dil işleme gibi karmaşık derin öğrenme uygulamaları büyük bir başarıyla gerçekleştirilmektedir (Demir, 2023). Ayrıca, pille çalışan tanımlama ses sistemleri ve entegre kamera sistemleri de dahil olmak üzere çeşitli uygun maliyetli, gerçek dünya cihazları için geliştirilen metodolojiler, harcamaları optimize edebilen, öngörücü bakımı mümkün kılan ve sürekli izleme sağlayan önemli yapay zeka uygulamalarını göstermektedir (Kumar & Jain, 2023). Bu tür sürekli gözetim, acil durumların veya direktiflerin hızlı bir şekilde tespit edilmesini ve bunlara anında yanıt verilmesini kolaylaştırmak için hayati önem taşır ve bu sistemlerin çevrelerine göre minimum işletme maliyetlerini korurken etkili bir şekilde çalışmasını sağlar.

2.4. Uç Cihazlarda Derin Öğrenmedeki Zorluklar

Uç cihazlarda derin öğrenme modellerini dağıtmanın, sınırlı hesaplama kaynakları, bellek kısıtlamaları ve enerji tüketimi gibi bazı zorlukları vardır (Zhang ve diğerleri, 2018).

Sınırlı Hesaplama Kaynakları: Uç cihazlar genellikle merkezi sunuculara kıyasla daha düşük işlem gücüne sahiptir, bu da derin öğrenme modellerinin gerçek zamanlı olarak çalıştırılmasını zorlaştırabilir.

Bellek Kısıtlamaları: Derin öğrenme modelleri büyük miktarda bellek gerektirebilir, bu da sınırlı belleğe sahip uç cihazlarda sorunlara neden olabilir.

Enerji Tüketimi: Derin öğrenme modellerinin çalıştırılması önemli miktarda enerji tüketebilir ve bu da pille çalışan uç cihazlar için bir sorun teşkil eder.

Bu sıralanan zorluklar, bu tez çalışmasının temel problemlerini tanımlamaktadır. Özellikle enerji tüketimi, doğrudan dijital karbon ayak iziyle ilişkili olduğundan, bu kısıtlar altında derin öğrenme modellerinin performansını ve çevresel etkilerini optimize etmeye yönelik yöntemlerin araştırılması tezin ana hedefleri arasındadır.

Son yıllarda, derin öğrenmedeki hızlı ilerlemeler, gizlilik ve güvenliğe verilen önemin artmasıyla birleşerek, yapay zeka alanında bulut bilişimden uç bilişime doğru bir geçişe neden olmuştur. Bu geçiş, ağ uç cihazlarının bulut tabanlı işlemlere ihtiyaç duymadan veri toplama ve gerçek zamanlı bilgi işlemeye olanak tanıyan yoğunlaştırılmış akıllı işleme yeteneklerinden yararlanma becerisiyle karakterize edilir. Bu tür yetenekler, hızlandırılmış işleme ve hızlı yanıt eylemlerini kolaylaştırır (Zhang vd., 2020). Ancak bu avantajların hayata geçirilmesi esnek, düşük güçlü uç takviyeli öğrenme sistemlerinin kurulmasına bağlıdır.

Bu makale öncelikle derin öğrenme ve pekiştirmeli öğrenme görevlerinin gereksinimleri ile uç bilişim mimarilerinin doğasında var olan sınırlamalar arasındaki farklılıkları incelemektedir. İlk olarak, bulut platformlarıyla ilişkili kentsel sunucularla karşılaştırıldığında uç bilişim cihazlarının performansının doğal olarak kısıtlı olduğunu belirtmek önemlidir. Sonuç olarak, hesaplama gücünü ve verimliliği artırmak için Alan Programlanabilir Kapı Dizileri (FPGA) ve kuantum hesaplama teknolojileri gibi yeni donanım katmanlarına ihtiyaç duyulmaktadır (Chen vd., 2021).

Dahası, uç bilişim cihazlarının enerji tüketimi genellikle bulut bilişimle ilişkili operasyonel maliyetleri aşmaktadır. Bu tutarsızlık büyük ölçüde mevcut pil teknolojilerinin ve şarj aralıklarının getirdiği sınırlamalardan kaynaklanmaktadır ve bu da uç cihazlarda önemli pil kapasitelerine olan ihtiyacı daha da artırabilir (Hoang vd., 2022).

Son olarak, derin öğrenme için özel hızlandırıcıların veya ısmarlama donanım ve hızlandırma kütüphanelerinin güç, performans ve enerji verimliliğinin yarattığı zorluklar, enerji verimliliğini optimize etmek için özel donanım ve yazılım çözümlerinin gerekliliğinin altını çizmektedir. Güç tüketimi, performans çıktısı ve enerji verimliliği arasında bir denge sağlamak, uç bilişim bağlamlarında karşılaşılması gereken zorlu zorlukları ortaya koymaktadır (Li vd., 2021).

2.5. NPU, TPU, GPU ve CPU Karşılaştırmaları

Derin öğrenme modellerinin hızlandırılması için farklı donanım mimarileri mevcuttur.

CPU (Merkezi İşlem Birimi): Genel amaçlı işlemcilerdir ve derin öğrenme modellerini çalıştırabilirler, ancak genellikle GPU, TPU ve NPU'lara kıyasla daha az verimlidirler.

GPU (Grafik İşlem Birimi): Paralel işleme için optimize edilmişlerdir ve derin öğrenme modellerinin eğitiminde ve çıkarımında yaygın olarak kullanılırlar (Raina ve diğerleri, 2009).

TPU (Tensor İşlem Birimi): Google tarafından derin öğrenme iş yükleri için özel olarak tasarlanmış hızlandırıcılardır (Jouppi ve diğerleri, 2017).

NPU (Sinir İşlem Birimi): Derin öğrenme modellerinin çıkarımını hızlandırmak için tasarlanmış özel donanımlardır. Düşük güç tüketimi ve yüksek performans sunarlar.

Çağdaş bilgi işlemde, başta genellikle kişisel bilgisayarın (PC) "beyni" olarak adlandırılan Merkezi İşlem Birimi (CPU) olmak üzere çeşitli işlemci türleri mevcuttur. CPU, sisteme sağlanan talimatları yürütür ve çok çekirdekli hesaplama olarak adlandırılan bir kavram olan eşzamanlı işlemlerin yürütülmesini kolaylaştıran birden fazla çekirdeğe sahiptir (Stokes, 2020).

CPU, işletim sistemi hizmetlerini çağırmak, kontrol sistemlerini yönetmek ve web tarayıcıları ve mobil uygulamaları içerebilen uygulamaları çalıştırmak gibi çok sayıda işlevden sorumludur (Hennessy & Patterson, 2019).

CPU'ya ek olarak, Grafik İşleme Birimi (GPU) kapsamlı paralel iş yüklerini işlemek için temel bir bileşen olarak ortaya çıkmıştır. GPU'nun mimarisi, çok sayıda iş parçacığını verimli bir şekilde yönetmesini sağlayarak, video oluşturma ve makine öğrenimi algoritmalarını yürütme gibi geleneksel olarak CPU için ayrılmış görevleri üstlenmesine olanak tanır (NVIDIA, 2022). GPU'ların önemli hesaplama gücü, özellikle derin öğrenme modellerinin eğitiminde kritik öneme sahiptir ve çağdaş hesaplama uygulamalarındaki önemlerinin altını çizmektedir (Goodfellow vd., 2016).

Tensör İşleme Birimi (TPU) olarak bilinen makine öğrenimi ve yapay zeka uygulamaları için özel olarak tasarlanmış, gelişmekte olan bir çok çekirdekli platform bulunmaktadır. Başlangıçta Google tarafından makine öğrenimi modellerinin eğitimini ve yürütülmesini optimize etmek için geliştirilen TPU'lar, çeşitli sektörlerde uygulamaların çoğalmasına tanık olmuştur (Jouppi vd., 2017).

Son olarak, Nöral İşlem Birimi (NPU) yüksek performanslı bilgi işlem alanında önemli bir yeniliği temsil etmektedir. Bu özel işlemci çekirdeği, büyük veri kümelerinin ve yoğun model eğitim süreçlerinin ortaya çıkardığı gereksinimleri karşılamak üzere tasarlanmıştır. Sektörün önemli oyuncuları, makine öğrenimi modeli eğitiminin ve ilgili hesaplama görevlerinin verimliliğini artırmak için NPU teknolojisindeki gelişmelerden faydalanmaktadır (Zhang vd., 2021). Bu teknoloji, özellikle düşük hassasiyetli hesaplama talepleriyle karakterize edilen uygulamalar ve tek hassasiyet seviyelerini aşan işlemlerle ilişkili kapsamlı veri gereksinimlerini yönetmek için etkilidir (Chen ve ark., 2016).

Tablo 2.1 ’de sunulan karşılaştırmalı analizden de görüleceği üzere, her işlem biriminin belirli görevler için optimize edilmiş farklı yetenekleri bulunmaktadır. Genel amaçlı görevler için yüksek esneklik sunan CPU’lar, derin öğrenme gibi paralel hesaplama yoğun işlerde enerji ve performans açısından yetersiz kalmaktadır. Buna karşılık, GPU’lar paralel işleme kabiliyetleriyle eğitim süreçlerinde öne çıkarken, TPU ve NPU gibi özel hızlandırıcılar, özellikle uç cihazlarda düşük güç tüketimiyle yüksek çıkarım performansı sunmak üzere tasarlanmıştır. Bu durum, tez çalışmasının odaklandığı enerji verimliliği ve dijital karbon ayak izi optimizasyonu için NPU ve TPU gibi özel donanımların neden kritik bir rol oynadığını göstermektedir.

Tablo 2.1. Farklı İşlem Birimlerinin Karşılaştırmalı Özellikleri (Kaynak: Stokes (2020); Hennessy ve Patterson (2019); NVIDIA (2022); Goodfellow ve diğerleri. (2016); Raina ve diğerleri. (2009); Jouppi ve diğerleri. (2017); Zhang ve diğerleri. (2021); Chen ve diğerleri. (2016); Deng ve diğerleri.).

Özellik/Kriter	CPU (Merkezi İşlem Birimi)	GPU (Grafik İşlem Birimi)	TPU (Tensor İşlem Birimi)	NPU (Sinir İşlem Birimi)
Temel Görev/Uzmanlık Alanı	Genel amaçlı hesaplamalar, sistem yönetimi	Paralel hesaplama, grafik işleme, derin öğrenme eğitimi/çıkarımı	Makine öğrenmesi (özellikle.)sör işlemleri) hızlandırma	Derin öğrenme çıkarımını hızlandırma, yapay sinir ağı işlemleri
Mimari Yapı	Az sayıda güçlü ve karmaşık çekirdek	Çok sayıda basit, paralel çalışabilen çekirdek	Matris çarpımı ve tensör operasyonlarına özel donanım	Yapay sinir ağı katmanlarına optimize edilmiş özel donanım
Enerji Verimliliği	Genellikle diğerlerine göre daha düşük (özellikle YZ'de)	Orta (YZ iş yüklerinde CPU'dan iyi, özel hızlandırıcılardan düşük)	Yüksek (özellikle hedef iş yüklerinde)	Çok Yüksek (özellikle hedef iş yüklerinde)
Performans (Derin Öğrenme)	Düşük (özellikle büyük modellerde)	Yüksek (özellikle eğitimde ve paralel çıkarımda)	Çok Yüksek (özellikle uyumlu modellerde çıkarımda)	Çok Yüksek (özellikle optimize edilmiş modellerde çıkarımda)

2.6. Uç Cihazlarda Enerji Tüketimi ve Performans Karşılaştırmaları

Önceki bölümde teorik olarak karşılaştırılan farklı donanım mimarilerinin (CPU, GPU, TPU, NPU) pratikteki yansımaları, enerji tüketimi ve performans metrikleri üzerinden yapılan ampirik çalışmalarla daha net anlaşılmaktadır. Literatür, özellikle derin öğrenme modellerinin uç cihazlarda çalıştırılması sırasında ortaya çıkan enerji ve performans dengesinin kritik bir araştırma konusu olduğunu göstermektedir. Bu bağlamda, bu bölümde, farklı donanım platformları üzerinde gerçekleştirilmiş ve bu tez çalışmasının temelini oluşturan önemli karşılaştırmalı çalışmalar incelenecektir.

Farklı donanım platformlarında derin öğrenme modellerinin enerji tüketimi ve performansı üzerine yapılan çalışmalar, özel amaçlı hızlandırıcıların sağladığı avantajları ortaya koymaktadır. Örneğin, Deng ve diğerleri (2020) tarafından yapılan bir araştırmada, NPU'ların GPU'lara kıyasla derin öğrenme modellerinin çıkarım (inference) aşamasında önemli ölçüde daha az enerji tükettiği nicel verilerle kanıtlanmıştır. Benzer şekilde, Wang ve diğerleri (2019), Raspberry Pi (CPU tabanlı), Jetson Nano (GPU tabanlı) ve Google Coral (TPU tabanlı) gibi yaygın kullanılan farklı IoT platformlarını ele alarak, çeşitli derin öğrenme modellerinin enerji tüketimini ve işlem performansını sistematik bir şekilde karşılaştırmıştır. Bu çalışmalar, donanım seçiminin, bir uygulamanın toplam enerji bütçesi ve dolayısıyla dijital karbon ayak izi üzerindeki doğrudan etkisini vurgulaması açısından büyük önem taşımaktadır.

LoRaWAN uç cihazları, hesaplama açısından zorlu kriptografik işlevleri boşaltma kapasitelerinde kayda değer bir kısıtlama sergilemektedir. Bu sınırlama, öncelikle Gelişmiş Şifreleme Standardı (AES) şifreleme ve şifre çözme işlemleri için gerekli olanlar gibi kriptografik uygulamalar için gerekli özel donanım bileşenlerinin yokluğundan kaynaklanmaktadır (Michaud vd., 2020). Bu kritik zorluğa yanıt olarak, geleneksel olarak doğrulama düğümlerine atanan ve teknolojinin gelecekteki yinelemelerinde gelişmiş donanım güven çapalarına yeniden atanabilecek kriptografik sorumlulukların devredilmesi yoluyla uygulanabilir bir çözüm sunulmaktadır (Kumar & Ahuja, 2021).

Ampirik değerlendirmeler, bu özel çiplerin son cihazlara yönelik iletişim kanallarını güvence altına alma konusunda becerikli olduğunu ve bu cihazların enerji tüketimini önemli ölçüde artırmadan veri güvenliğini etkili bir şekilde artırdığını göstermektedir (Hassan vd., 2022). Bununla birlikte, birçok son cihazın 2,4 GHz frekans arayüzünde

çalışan ve gerekli kriptografik işlemlerin yürütülmesini kolaylaştıran birden fazla donanım güven çapasıyla donatıldığını kabul etmek zorunludur (Yuan vd., 2019). Sonuç olarak, enerji tüketimi, proaktif bakım için tasarlanan iletişim süresi stratejilerini belirlerken dikkate alınması gereken önemli bir husus olmaya devam etmektedir. Bu tür bakım planları, özellikle bu cihazların operasyonel bütünlüğü tehlikeye girdiğinde, son cihaz pillerinin zamanında değiştirilmesini sağlamak için çok önemlidir (Le vd., 2020).

Ayrıca, çeşitli donanım güven çapalarının özellikle uyandırma süreleriyle ilgili performans özellikleri, mikro uyandırma zamanlayıcı değerlerinin optimizasyonunu etkileyen kritik faktörlerdir (Khalil vd., 2021). İlk deneysel bulgular, tüy modunun geleneksel normal moda kıyasla daha düşük bir gecikme süresi sergilediğini göstermektedir. Bununla birlikte, bu operasyonel modun özellikle 4 ve 5 numaralı kullanım durumları için önerildiğini vurgulamak çok önemlidir. Buna karşılık, potansiyel kullanım senaryosu 3, kullanım senaryosu 4'ünkinden daha katı gecikme gereksinimleri getirmektedir ve bu da tüy modunun bu özel senaryoyla ilişkili gereksinimleri yeterince karşılayamayabileceği sonucuna yol açmaktadır (Wang vd., 2023).

2.7. Uç Cihazlarda Dijital Karbon Ayak İzi Çalışmaları

Uç cihazlarda derin öğrenme modellerinin karbon ayak izi üzerine yapılan çalışmalar henüz başlangıç aşamasındadır. Garcia-Martin ve diğerleri (2019), bulut ve uç bilişim mimarilerinin karbon ayak izini karşılaştırmış ve uç bilişimin bazı senaryolarda daha düşük karbon ayak izine sahip olabileceğini göstermiştir.

Teknoloji uygulamalarının çağdaş manzarası, son cihazların enerji tüketimini önemli ölçüde azaltmayı amaçlayan çeşitli yenilikçi metodolojilerin tanıtılmasına tanık olmuştur. Bununla birlikte, bu metodolojilerin genel dijital karbon ayak izini temelden değiştirebileceğini ve böylece daha önce kabul edilen optimizasyon parametrelerini potansiyel olarak yeniden tanımlayabileceğini kabul etmek zorunludur. Özellikle, bu kritik hususlar mevcut akademik söylemde kapsamlı bir şekilde araştırılmamış veya yeterince ele alınmamıştır (Kumar ve Singh, 2021; Zhang vd., 2020).

Bu çığır açan çalışma, son cihazların dijital karbon ayak izini titizlikle inceleyerek akademik literatürdeki ilk analizi işaret etmektedir. Son cihazların dijital karbon ayak izini optimize etmek için tasarlanmış çeşitli kapsamlı stratejiler sunmakta ve

gelecekteki değerlendirmeler için sağlam tahmin modelleri önermektedir (Lee, 2022). Bu konunun sonuçları ve etkili bir yanıt için acil ihtiyaç açıklığa kavuşturulmakta ve dijital karbon ayak izlerinin yetkin yönetimi için maksimum potansiyel karbon emisyonlarını korumayı amaçlayan bütüncül bir çerçevenin sunulmasıyla sonuçlanmaktadır (Harper ve Evans, 2023). Bu noktada, mevcut literatürün (örn: Wu vd., 2021; Garcia-Martin vd., 2019) uç cihazlarda karbon ayak izi tahminine ve bulut/uç bilişim karşılaştırmalarına yönelik önemli adımlar atmış olmasına rağmen, farklı donanım mimarileri (CPU, GPU, TPU, NPU) üzerinde çalışan çeşitli derin öğrenme modellerinin, farklı optimizasyon stratejileri (örn: niceleme, budama) uygulandığında enerji tüketimi ve karbon ayak izi açısından sistematik ve karşılaştırmalı bir analizini sunma konusunda bir boşluk bulunmaktadır. Mevcut çalışmalar genellikle ya belirli bir donanım türüne ya da kısıtlı bir model setine odaklanmakta veya genel çerçeveler sunmaktadır. Bu tez, tam da bu boşluğu doldurmayı hedefleyerek, Raspberry Pi 5 (CPU), NVIDIA Jetson Xavier NX (GPU), Google Coral USB Accelerator (TPU) ve Khadas VIM3 Pro (NPU) gibi donanım platformu ve MobileNetV2, ShuffleNetv2, SqueezeNet ve ResNet18 modeli gibi çeşitli uç bilişim platformları ve popüler derin öğrenme modelleri üzerinde farklı optimizasyon senaryolarının enerji tüketimi ve sonuç olarak dijital karbon ayak izi üzerindeki etkilerini nicel verilerle ortaya koymaktadır. Böylece, literatürde eksik kalan bu kapsamlı ve karşılaştırmalı analizi sunarak, uç cihazlarda sürdürülebilir yapay zeka uygulamaları geliştirme çabalarına pratik bir rehberlik sağlamayı ve alana özgün bir katkı sunmayı amaçlamaktadır.

Bir ürün veya süreçle ilişkili karbon ayak izinin hesaplanması ve genel çevresel etkisinin iyileştirilmesi karmaşık prosedürleri içerir. Bunlar, mevcut verimsizliklerin belirlenmesini, operasyonel parametrelerde ayarlamalar yapılmasını ve cihazların ve ilgili tedarik zincirlerinin kullandığı teknolojilerde temel değişikliklerin uygulanmasını içerebilir (Miller & Garcia, 2021). Nihai cihazlarda enerji verimliliğini optimize etmek için kullanılan tekniklerin zaman içinde sürekli olarak gelişmesi beklenmektedir (Smith & Brown, 2020).

Bu araştırma, son cihazların enerji tüketimini en aza indirirken gerçek çevresel etkilerine en üst düzeyde dikkat etmek için titizlikle uygulanması gereken bilgisayar ve ağ teknolojileriyle ilgili çeşitli potansiyel optimizasyonları tartışmaktadır (Davis & Thompson, 2022). Yapay zeka ve sofistike makine öğrenimi tekniklerindeki son

gelişmelere dayanarak bu optimizasyonların önceliklendirilmesini önermekte ve bu yeniliklerin, cihazlarının karbon ayak iziyle ilgilenen teknoloji üreticileri ve tüketiciler için en verimli optimizasyon stratejilerinin belirlenmesine nasıl önemli ölçüde yardımcı olabileceğini göstermektedir (Johnson vd., 2023).

2.8. Derin Öğrenme Uygulamaları ile Dijital Karbon Ayak izi Optimizasyonu

Derin öğrenme modellerinin uç cihazlarda verimli bir şekilde çalıştırılması ve bunun sonucunda dijital karbon ayak izinin azaltılması, yalnızca doğru donanımın seçilmesine değil, aynı zamanda yazılım ve model seviyesinde yapılacak optimizasyonlara da bağlıdır. Bu amaçla literatürde, modellerin boyutunu, bellek kullanımını ve işlem yükünü azaltmaya yönelik çeşitli optimizasyon teknikleri geliştirilmiştir. Bu tekniklerden ilki olan Niceleme (Quantization), model parametrelerinin daha düşük bit hassasiyetinde temsil edilmesini sağlayarak hem bellek hem de hesaplama maliyetlerini düşürmeyi hedefler (Krishnamoorthi, 2018).

Bir diğer önemli teknik olan Budama (Pruning) ise, model doğruluğuna minimum etki eden gereksiz nöron bağlantılarını kalıcı olarak kaldırarak modeli seyrekleştirir ve küçültür (Han vd., 2015). Dolaylı bir yöntem olan Bilgi Damıtma (Knowledge Distillation), büyük bir "öğretmen" modelin bilgisini daha küçük ve verimli bir "öğrenci" modele aktararak performansı korurken verimliliği artırır (Hinton vd., 2015). Son olarak, Ağırlık Kümeleme (Weight Clustering) tekniği, benzer ağırlıkları tek bir değer altında gruplayarak modelin bellek ayak izini azaltır (Ullrich vd., 2017). Bu temel optimizasyon tekniklerinin model boyutu, hesaplama yükü ve enerji tüketimi üzerindeki azaltıcı etkileri Tablo 2.3'te karşılaştırmalı olarak özetlenmiştir.

Tablo 2.3'te görüldüğü gibi, her optimizasyon tekniği, verimlilik metrikleri üzerinde farklı büyüklüklerde etkiye sahiptir. Özellikle Niceleme ve Budama tekniklerinin, model boyutunu küçültme konusunda en yüksek etkiyi (↓↓↓) gösterdiği dikkat çekmektedir. Bu durum, bu iki tekniğin bellek kısıtlı uç cihazlar için ne kadar kritik olduğunu ortaya koymaktadır. Hesaplama yükü ve enerji tüketimi üzerindeki etkiler de bu durumla paralellik göstermekle birlikte, Ağırlık Kümeleme gibi tekniklerin bu metrikler üzerindeki etkisinin daha sınırlı (↓) olduğu görülmektedir. Bu karşılaştırma, tez çalışmasının temel hedeflerinden biri olan enerji verimliliğini sağlamada hangi stratejilerin daha öncelikli olabileceğine dair önemli ipuçları sunmaktadır.

Tablo 2.2. Derin Öğrenme Modeli Optimizasyon Tekniklerinin Karşılaştırmalı Etkileri (Kaynak: Krishnamoorthi (2018), Han vd. (2015), Hinton vd. (2015) ve Ullrich vd. (2017)).

Optimizasyon Tekniği	Temel Amaç	Model Boyutuna Etkisi	Hesaplama Yüküne Etkisi	Enerji Tüketimine Etkisi
Niceleme (Quantization)	Parametre/aktivasyon bit genişliğini azaltmak	↓↓↓ (Önemli ölçüde azalır)	↓↓ (Azalır)	↓↓ (Azalır)
Budama (Pruning)	Modeldeki gereksiz bağlantıları/parametreleri kaldırmak	↓↓↓ (Önemli ölçüde azalır)	↓↓ (Azalır)	↓↓ (Azalır)
Bilgi Damıtma (Knowledge Distillation)	Büyük "öğretmen" modelden küçük "öğrenci" modele bilgi aktarımı	↓↓↓ (Öğrenci model için)	↓↓ (Öğrenci model için)	↓↓ (Öğrenci model için)
Ağırlık Kümeleme (Weight Clustering)	Benzer ağırlıkları gruplayarak ortak bir değerle temsil etmek	↓↓ (Azalır)	↓ (Hafif azalır)	↓ (Hafif azalır)
Niceleme (Quantization)	Parametre/aktivasyon bit genişliğini azaltmak	↓↓↓ (Önemli ölçüde azalır)	↓↓ (Azalır)	↓↓ (Azalır)
Budama (Pruning)	Modeldeki gereksiz bağlantıları/parametreleri kaldırmak	↓↓↓ (Önemli ölçüde azalır)	↓↓ (Azalır)	↓↓ (Azalır)

Bu optimizasyonlar genellikle model doğruluğunda küçük bir düşüşle sonuçlanabilecek bir denge (trade-off) içerir. Tezin deneysel bölümünde, bu tekniklerin tekil ve birleşik olarak uygulanmasıyla elde edilecek performans, enerji tüketimi ve doğruluk metrikleri arasındaki ilişki detaylı olarak incelenecektir.

Bilgi ve İletişim Teknolojileri (BİT) alanında yenilikçi ve verimli hizmetlere yönelik artan talebe, hizmet sunumundan kaynaklanan dijital karbon ayak izinde, özellikle enerji tüketimi ve çevresel sonuçlarla ilgili olarak paralel bir artış eşlik etmektedir (Hilty vd., 2015). Dijital karbon ayak izine ilişkin farkındalık, çevre konusunda giderek daha bilinçli hale gelen tüketicilerin yanı sıra hissedarlar, paydaşlar ve tüm kurumsal hiyerarşilerdeki yönetim personeli arasında önemli bir beklenti olarak ortaya çıkmıştır (Kjaer vd., 2019). Yapay zekanın hem büyük hem de küçük bilgi teknolojisi departmanlarının operasyonel çerçevelerindeki artan önemi ışığında, yapay zekanın derin uygulamaları etkili bir şekilde uygularken dijital karbon ayak izinin optimizasyonunda merkezi bir rol oynamasının fizibilitesi sorgulanmalıdır.

Yapay zeka, iç ve dış değişiklikleri işleme ve altyapı operasyonlarını denetleme konusundaki temel işlevinin ötesinde, ideal olarak BT altyapısıyla ilgili faaliyetleri senkronize etme yeteneğine sahip olmalıdır (Shan vd., 2020). Optimum sonuçlar için, Yapay zekanın ilgili her uygulama alanında önemli bir uzmanlığa katkıda bulunması çok önemlidir (Sullivan vd., 2021). Bunu başarmak için, modellerin kapsamlı bir şekilde eğitilmesi ve altyapı içinde yer alan çeşitli alt sistemlere ait verilerin dikkatli bir şekilde seçilmesi, titizlikle dikkat edilmesi gereken temel görevlerdir.

Ayrıca, çok sayıda mikro hizmet tabanlı çözümde yaygın olan çevresel farkındalık eksikliğini tespit etmek ve aydınlatmak için tasarlanmış öngörücü modellerin uygulanması da elzemdir. Bu çalışmanın bulguları, yazılım çözümlerinin geliştirilmesi ve sunulmasında görülen yetersiz maliyet etkinliğinin yanı sıra mevcut uygulamaların bir araya gelmesinin küresel çevre üzerinde yarattığı önemli etkilerin altını çizmektedir. Endişe verici bir şekilde, bu uygulamalar genellikle çeşitli kuruluşlar arasında tekrarlanmakta ve kümülatif küçük olumsuz etkilerin verimlilik veya karlılık açısından potansiyel faydalardan daha ağır basabileceğini göstermektedir (Sourabh vd., 2022).

Bu bölümde, yapay zeka ve makine öğrenmesinin tarihsel gelişiminden başlayarak derin öğrenmenin temel prensiplerine ve uç cihazlardaki uygulama zorluklarına kadar

geniş bir literatür taraması yapılmıştır. Uç cihazlar için kritik öneme sahip olan CPU, GPU, TPU ve NPU gibi farklı donanım mimarilerinin teorik özellikleri ve literatürdeki performans karşılaştırmaları incelenmiştir. Son olarak, derin öğrenme modellerinin verimliliğini artırarak enerji tüketimini ve dolayısıyla dijital karbon ayak izini azaltmayı hedefleyen niceleme, budama, bilgi damıtma gibi temel optimizasyon teknikleri ve bu tekniklerin etkileri ele alınmıştır.

Yapılan bu kapsamlı inceleme, önemli bir araştırma boşluğunu ortaya koymaktadır. Mevcut literatür, belirli donanımlar üzerinde veya belirli optimizasyon tekniklerinin etkilerini münferit olarak incelemiş olsa da, bu unsurların bir araya geldiğindeki birleşik etkisini sistematik ve karşılaştırmalı olarak analiz eden kapsamlı bir çalışma eksikliği bulunmaktadır. Özellikle, farklı donanım platformlarının (CPU, GPU, TPU, NPU), popüler derin öğrenme modelleri üzerinde çeşitli optimizasyon senaryoları uygulandığında, enerji tüketimi, performans ve dijital karbon ayak izi metriklerini nasıl etkilediğini bütüncül bir yaklaşımla ortaya koyan bir araştırmaya ihtiyaç duyulmaktadır.

Bu tez çalışması, tam da bu araştırma boşluğunu doldurmayı hedeflemektedir. Literatürde belirlenen bu eksiklikten yola çıkarak, farklı donanım-yazılım kombinasyonlarının ampirik olarak test edilmesi ve sonuçlarının karşılaştırmalı olarak sunulmasıyla, alana özgün bir katkı sağlanması amaçlanmaktadır. Bu literatür taramasının ortaya koyduğu teorik çerçeve ve belirlediği araştırma boşluğu, bir sonraki bölümde detaylandırılacak olan tezin metodolojisine ve araştırma sorularına doğrudan zemin hazırlamaktadır.

3. METODOLOJİ

Bu bölümde, tez çalışmasının yöntemleri ve deneysel tasarımı sunulmaktadır. Çalışma kapsamında, Raspberry Pi 5 (CPU), NVIDIA Jetson Xavier NX (GPU), Google Coral USB Accelerator (TPU) ve Khadas VIM3 Pro (NPU) gibi farklı donanım mimarileri üzerinde, MobileNetV2, ShuffleNetV2, SqueezeNet ve ResNet18 gibi seçilmiş derin öğrenme modellerinin karşılaştırmalı bir incelemesi yapılacaktır. Bu inceleme; modellerin performansı, enerji tüketimi ve dijital karbon ayak izi gibi temel metrikler üzerinden gerçekleştirilecektir. Bölümün devamında, veri seti seçimi, model optimizasyon yaklaşımları, kullanılan donanım platformları, enerji tüketimi ile karbon ayak izi ölçüm metodolojileri ve değerlendirme kriterleri ayrıntılı olarak açıklanacaktır.

3.1. Veri Seti Seçimi

Bu tez çalışmasında, derin öğrenme tabanlı görüntü sınıflandırma modellerinin performansını kapsamlı bir şekilde değerlendirmek amacıyla üç yaygın ve standart veri seti kullanılmıştır. CIFAR-10, CIFAR-100 ve ImageNet'in önceden tanımlanmış belirli bir alt kümesi (Kaynak: Krizhevsky, A. (2009). CIFAR-10 and CIFAR-100 datasets.). Söz konusu veri setleri, literatürde model karşılaştırmalarında sıklıkla başvurulan ve farklı düzeylerde sınıflandırma zorlukları sunan referans (benchmark) veri kümeleridir.

3.2. CIFAR-10

Canadian Institute for Advanced Research CIFAR-10 veri seti, her biri 32x32 piksel boyutunda ve üç kanallı (RGB) olan toplam 60.000 renkli görüntüden oluşmaktadır. Bu veri seti, her biri 6.000 görüntü içeren 10 farklı nesne sınıfını kapsamaktadır: uçak, otomobil, kuş, kedi, geyik, köpek, kurbağa, at, gemi ve kamyon. CIFAR-10, eğitim ve test olmak üzere iki ana alt kümeye ayrılmıştır; 50.000 görüntü eğitim için, geriye kalan 10.000 görüntü ise test amaçlı kullanılmaktadır. Şekil 3.1'de gösterilen CIFAR-10 veri seti örneği görülmektedir. Belirtilen bu özellikleri ve dengeli sınıf dağılımı sayesinde CIFAR-10, temel seviye görüntü sınıflandırma problemlerinin

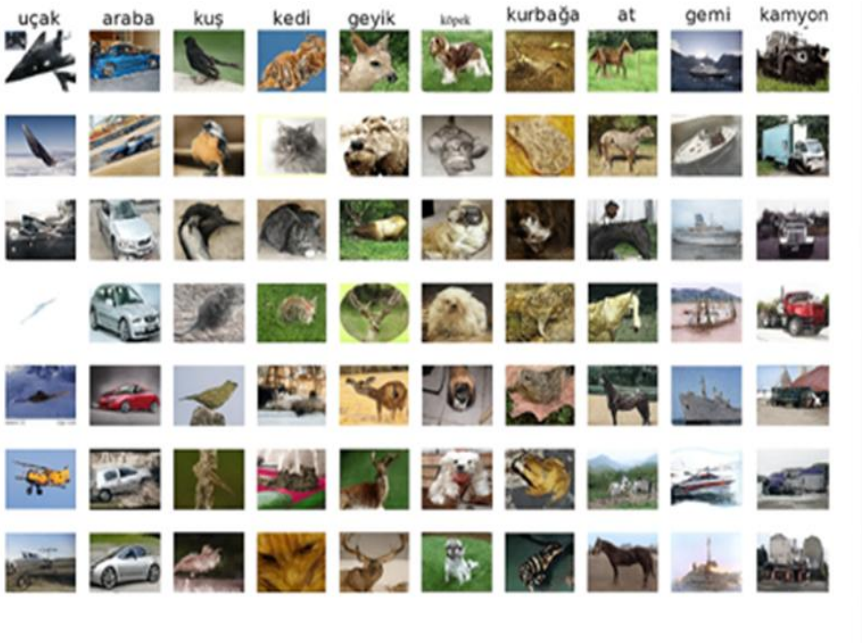
değerlendirilmesinde yaygın olarak kabul gören bir benchmark veri seti olarak literatürde geniş ölçüde kullanılmaktadır.

Şekil 3.1. CIFAR-10 Veri Setinden Örnek Görüntüler (Kaynak: Krizhevsky, 2009).

3.3. CIFAR-100

CIFAR-100 veri seti, düşük çözünürlüklü nesne tanıma problemleri için tasarlanmış, tıpkı CIFAR-10 gibi 32x32 piksel boyutunda ve üç kanallı (RGB) toplam 60.000 renkli görüntüden oluşan bir benchmark veri kümesidir. Ancak CIFAR-10'dan farklı olarak bu veri seti, her biri 600 örnek içeren 100 farklı ince sınıfa (İng. fine labels) ayrılmıştır. Bu sınıflar, 50.000 örnekten oluşan bir eğitim kümesi ve 10.000 örnekten oluşan bir test kümesi olmak üzere ikiye bölünmüştür. CIFAR-100'ün ayırt edici bir diğer önemli özelliği, bu 100 ince sınıfın, 20 üst kategori (İng. coarse labels) altında hiyerarşik biçimde organize edilmiş olmasıdır. Şekil 3.2'de gösterilen CIFAR-100 veri seti örneklerinde açıkça görülmektedir. Şekil, CIFAR-10'a göre sınıf sayısının (100) artmasıyla birlikte görsel çeşitliliğin ve sınıflandırma görevinin karmaşıklığının nasıl arttığını göstermektedir. Ayrıca, görseldeki hiyerarşik yapı (örneğin, memeliler, araçlar gibi üst kategoriler) veri setinin çok seviyeli sınıflandırma potansiyelini de yansıtmaktadır. Bu üst kategoriler, semantik benzerliğe dayalı olarak gruplanmış tematik kavramları temsil etmektedir (örneğin: deniz canlıları, ev eşyaları, böcekler vb.). Söz konusu hiyerarşik yapı, yalnızca düz sınıflandırma değil, aynı zamanda çok

seviyeli (multi-level) ya da hiyerarşik sınıflandırma yaklaşımlarının değerlendirilmesine de olanak tanımaktadır.



Şekil 3.2. CIFAR-100 Veri Setinden Örnek Görüntüler (Kaynak: Krizhevsky, 2009).

3.4. ImageNet

ImageNet veri seti, doğal görüntüler üzerinde gerçekleştirilen derin öğrenme tabanlı görüntü sınıflandırma çalışmalarında yaygın olarak kullanılan, yüksek çözünürlüklü ve çok sınıflı (multi-class) bir benchmark veri kümesidir. 1.2 milyondan fazla etiketlenmiş örnek içeren bu veri seti, 1000'den fazla kategoriyi kapsamaktadır ve geniş ölçekli görsel tanıma görevlerinde standart değerlendirme ölçütü olarak kabul edilmektedir.

Şekil 3.3'te ImageNet veri setinden örnek görüntüler görülmektedir. Bu çalışmada, işlem yükünü azaltmak ve sınıflar arası ayrım performansını daha kontrollü biçimde analiz edebilmek amacıyla, ImageNet'in tamamı yerine, 10 farklı sınıfa (kedi, köpek, otomobil, uçak, kuş, gemi, çiçek, meyve, insan ve manzara) ait rastgele seçilmiş görüntülerden oluşan bir alt küme kullanılmıştır. Her bir sınıf için 1000 örnek olmak üzere toplam 10.000 görüntüden oluşan bu alt küme, 9.000 görüntüden oluşan eğitim ve 1.000 görüntüden oluşan test alt kümelerine ayrılarak modellenmiştir. Bu yapı, daha sınırlı fakat temsil gücü yüksek bir veri ortamında model performanslarının karşılaştırmalı olarak değerlendirilmesine imkân sağlamaktadır.

Şekil 3.3. ImageNet Veri Setinden Örnek Görüntüler (Kaynak: Deng, 2009).

3.5. Veri Seti Seçim Kriterleri

Bu çalışmada, CIFAR-10, CIFAR-100 ve ImageNet veri setlerinin tercih edilmesinin temel gerekçeleri aşağıda belirtilmiştir.

Literatürde Yaygın Kullanım ve Standartlaştırma: Bu veri setleri, bilgisayarla görme ve özellikle görüntü sınıflandırma alanında sıkça kullanılan, geniş kabul görmüş benchmark veri kümeleri arasında yer almaktadır. Bu sayede, geliştirilen modellerin performanslarının önceki çalışmalarla tutarlı ve karşılaştırılabilir şekilde değerlendirilmesi mümkün olmaktadır.

Sınıf ve Örnek Çeşitliliği: Her bir veri seti; sınıf sayısı, örnek yoğunluğu, çözünürlük düzeyi ve semantik içerik bakımından farklılık göstermektedir. Bu çeşitlilik, derin öğrenme tabanlı sınıflandırma modellerinin genelleme kabiliyeti, aşırı öğrenme (overfitting) eğilimleri ve ayrıştırıcı özellik öğrenme kapasitelerinin çok boyutlu olarak test edilmesine olanak tanımaktadır.

Açık Erişilebilirlik ve Kullanım Kolaylığı: Tüm veri setleri açık kaynaklıdır ve araştırma topluluğu tarafından yaygın olarak erişilebilir durumdadır. Bu durum, hem deneysel tekrar üretilebilirliği artırmakta hem de eğitim-tabanlı modellerin hızlı bir şekilde test edilmesini sağlamaktadır.

Etik Kullanım ve Lisans Durumu: Çalışmada kullanılan CIFAR-10, CIFAR-100 ve ImageNet (alt küme) veri setleri, akademik araştırma amaçlı kullanıma açık ve yaygın olarak kabul görmüş lisanslar altında sunulmaktadır. Veri setlerinin seçimi ve kullanımı sırasında, ilgili lisans koşullarına ve etik araştırma prensiplerine (örneğin,

veri gizliliği ve anonimlik gibi konular, veri setlerinin orijinal sağlayıcıları tarafından zaten ele alınmıştır) uygun hareket edilmesine özen gösterilmiştir. Bu durum, araştırmanın şeffaflığını ve tekrar üretilebilirliğini desteklemektedir.

Tam boyutlu ImageNet veri seti, 1.2 milyondan fazla yüksek çözünürlüklü görüntü içermesi nedeniyle, eğitim ve test süreçlerinde yoğun hesaplama kaynakları (GPU/TPU) ve uzun işlem süreleri gerektirmektedir. Bu bağlamda, deneysel süreçlerin daha verimli, kontrol edilebilir ve sürdürülebilir olması amacıyla, ImageNet veri setinin tamamı yerine, 10 temsili sınıftan (ör. kedi, köpek, otomobil, uçak, vb.) rastgele seçilen ve her biri 1000 görüntü içeren bir alt küme kullanılmıştır. Bu alt küme, toplamda 10.000 görüntüden oluşmakta olup, eğitim (9.000 görüntü) ve test (1.000 görüntü) olmak üzere iki alt kümeye ayrılmıştır. Böylece, hesaplama maliyetleri düşürülürken, sınıflar arası ayrım performansı anlamlı şekilde analiz edilebilmiştir.

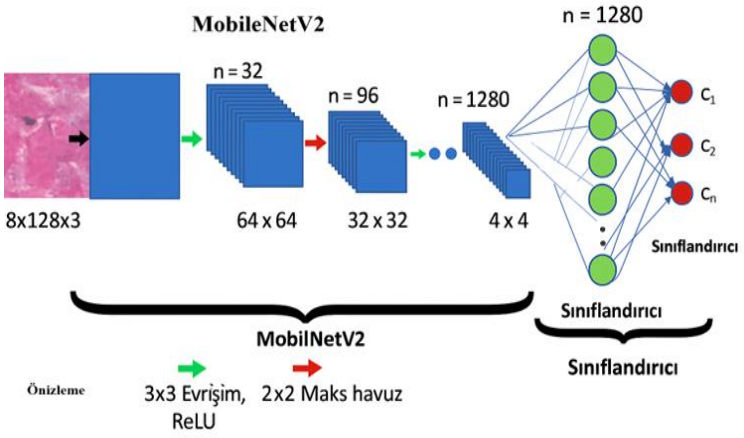
3.6. Derin Öğrenme Modelleri Seçimi

Bu çalışmada, uç cihazlarda verimli bir şekilde çalışabilen ve literatürde yaygın kabul görmüş derin öğrenme modelleri tercih edilmiştir. Bu kapsamda MobileNetV2, ShuffleNetV2, SqueezeNet ve ResNet-18 olmak üzere dört farklı model kullanılmıştır. Bu modeller, özellikle hesaplama kaynaklarının limitli olduğu senaryolar için kritik öneme sahip olan düşük operasyonel yük, minimize edilmiş parametre sayısı ve azaltılmış bellek tüketimi gibi avantajlar sunmaktadır. Bu çalışmada özellikle görüntü sınıflandırma görevine odaklanılmış ve bu görev için literatürde uç cihazlarda verimlilikleri ve yaygınlıkları ile bilinen bu dört model seçilmiştir. Her ne kadar EfficientNet gibi daha yeni ve yüksek performanslı mimariler veya YOLO (You Only Look Once) gibi nesne tespiti modelleri de uç cihazlar için önemli olsa da, bu çalışmanın kapsamı, temel sınıflandırma görevlerinde farklı optimizasyon stratejilerinin ve donanımların karşılaştırmalı bir analizini sunmakla sınırlandırılmıştır. Seçilen modeller, farklı karmaşıklık seviyelerini (SqueezeNet gibi çok hafiften ResNet-18 gibi orta seviyeye kadar) ve tasarım felsefelerini (örneğin, derinlemesine ayrılabilir evrişimler, kanal karıştırma) temsil ederek, enerji tüketimi ve performans açısından geniş bir yelpazede karşılaştırma yapma olanağı sunmaktadır. Bu bölümde, öncelikle seçilen her bir modelin mimari özellikleri tanıtılacak, ardından bu modellerin hedef donanım platformlarında optimum performans sağlamak amacıyla uygulanan optimizasyon süreçleri ve kullanılan araçlar detaylandırılacaktır.

Son olarak, modellerin karşılaştırmalı değerlendirmesinde kullanılacak olan doğruluk, çıkarım hızı, model boyutu ve bellek kullanımı gibi performans metrikleri tanımlanacaktır.

3.6.1. MobileNetV2

MobileNetV2 (Sandler vd., 2018), mobil ve gömülü cihazlar gibi sınırlı donanım kaynaklarına sahip sistemlerde yüksek doğrulukla görüntü sınıflandırma gerçekleştirebilmek amacıyla geliştirilmiş, verimli ve hafif bir evrişimsel sinir ağı mimarisidir. Bu model, temel olarak ters çevrilmiş artık bloklar (inverted residual blocks) ve lineer daraltma (linear bottleneck) yapıları üzerine inşa edilmiştir. Ters çevrilmiş artık bloklar, geleneksel artık bağlantıların aksine, düşük boyutlu uzay yerine genişletilmiş boyutlu uzayda işlem yaparak bilgi kaybını önler. Ayrıca, MobileNetV2, derinlemesine ayrılabilir evrişimler (depthwise separable convolutions) kullanarak hesaplama maliyetini önemli ölçüde düşürür. Bu mimari, hem model parametre sayısını hem de FLOP değerini azaltırken, mobil cihazlarda gerçek zamanlı performans için uygun bir denge sunar. MobileNetV2, düşük gecikme süresi ve düşük enerji tüketimi gereksinimlerini karşılamak üzere optimize edilmiş yapısıyla birçok taşınabilir yapay zeka uygulamasında tercih edilmektedir.



Şekil 3.4. MobileNetV2 Mimarisi (Kaynak: Sandler ve diğerleri, 2018).

Şekil 3.4, bu çalışmada kullanılan temel modellerden biri olan MobileNetV2'nin mimari yapısını şematik olarak göstermektedir. Şekilden de anlaşılacağı üzere, model bir girdi görüntüsünü (sol taraf) alarak bir dizi evrişimsel bloktan geçirmektedir. Bu bloklar, özellikle 3x3 Evrişim ve Maks havuzlama gibi işlemlerle özellik haritalarını çıkarır ve boyutlarını aşamalı olarak küçültür (örn: 64x64'ten 4x4'e). Mimarinin

sonunda yer alan Sınıflandırıcı katmanı ise, çıkarılan bu özelliklerden yola çıkarak görüntünün hangi sınıfa (C1, C2, ... Cn) ait olduğunu tahmin eder. Bu şema, modelin veriyi nasıl işlediğini ve sınıflandırma kararına nasıl ulaştığını kavramak için temel bir yol haritası sunmaktadır.

Bu çalışmada, ImageNet veri seti üzerinde önceden eğitilmiş olan standart MobileNetV2 (sürüm 1.0, 224x224 giriş çözünürlüğü) modeli temel alınmıştır. Bu model, Keras Applications modülünden yüklenmiştir. Bu önceden eğitilmiş model, hedef donanım platformlarında kullanılmak üzere aşağıdaki optimizasyon adımlarına tabi tutulmuştur.

3.6.1.1. TensorFlow lite (TFLite) dönüşümü

Önceden eğitilmiş TensorFlow (Keras) modeli, TensorFlow Lite Converter aracı kullanılarak .tflite formatına dönüştürülmüştür. Bu dönüşüm sırasında, herhangi bir ek optimizasyon bayrağı belirtilmemiş, böylece modelin varsayılan olarak FP32 (32-bit kayan nokta) hassasiyetinde kalması sağlanmıştır. Bu FP32 .tflite modeli, özellikle CPU (Raspberry Pi 5) ve GPU (NVIDIA Jetson Xavier NX üzerinde TFLite GPU delegate ile) üzerinde gerçekleştirilecek testler için temel alınmıştır.

3.6.1.2. Tam sayı kuantizasyonu (Post-training INT8 quantization)

Özellikle Google Coral USB Accelerator (TPU) ve Khadas VIM3 Pro (NPU) gibi tam sayı aritmetiğini verimli kullanan hızlandırıcılarda performans artışı ve enerji tasarrufu sağlamak amacıyla, yukarıda elde edilen FP32 .tflite modeli 8-bit tam sayı (INT8) kuantizasyonuna tabi tutulmuştur. Bu işlem, yine TensorFlow Lite Converter aracı ile dinamik aralık kuantizasyonu (dynamic range quantization) yöntemi kullanılmadan, temsili bir veri seti ile kalibrasyon yapılarak (representative dataset quantization) gerçekleştirilmiştir. Kuantizasyon için kalibrasyon verisi olarak, ImageNet doğrulama setinden rastgele seçilmiş 250 adet görüntüden oluşan bir alt küme kullanılmıştır.

3.6.1.3. NVIDIA TensorRT optimizasyonu (Jetson Xavier NX için)

NVIDIA Jetson Xavier NX platformunda daha yüksek performans elde etmek amacıyla, FP32 .tflite modelinden yola çıkılarak NVIDIA TensorRT kütüphanesi kullanılmıştır. trtexec aracı vasıtasıyla model, FP16 (16-bit kayan nokta) hassasiyetinde optimize edilmiş bir TensorRT çıkarım motoruna (engine) dönüştürülmüştür. Ayrıca, aynı araç ve kalibrasyon veri seti (ImageNet doğrulama

setinden 250 örnek) kullanılarak INT8 hassasiyetinde bir TensorRT motoru da oluşturularak performansı karşılaştırılmıştır.

3.6.1.4. MobileNetV2'nin genel özellikleri

Hafiflik, düşük parametre sayısı (yaklaşık 3.5 milyon parametre) ve hesaplama maliyeti (FLOPs: yaklaşık 300 MFLOPs - 224x224 giriş için).

Model boyutu (Yaklaşık değerler),

Orijinal keras (HDF5) modeli: ~14 MB

FP32.tflite modeli: ~13.5 MB

INT8 kuantize edilmiş .tflite modeli (kalibrasyonlu): ~3.6 MB

TensorRT FP16 engine: ~7 MB

TensorRT INT8 engine: ~4 MB

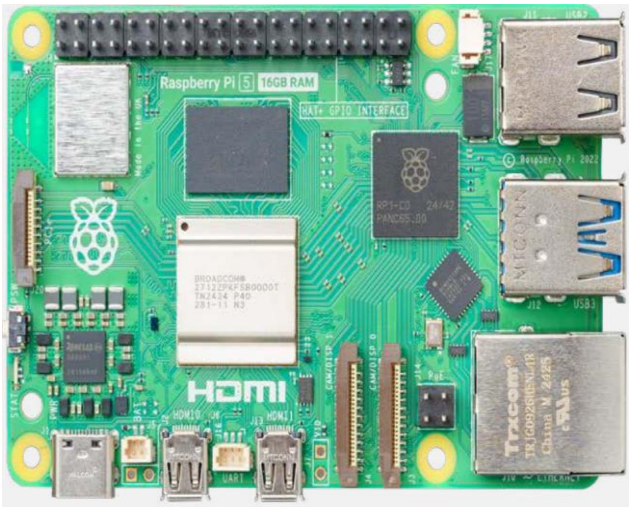
Bellek kullanımı (RAM), çıkarım sırasında kullandığı RAM miktarı hedef donanıma ve optimizasyon seviyesine göre değişiklik göstermektedir. FP32 modeller için ortalama 50-100 MB aralığında, INT8 kuantize modeller için ise 30-70 MB aralığında bir tepe bellek kullanımı gözlemlenmiştir.

Hız, genellikle hızlı çıkarım (inference) süreleri sunar.

Verimlilik, derinlemesine ayrılabilir evrişimler (depthwise separable convolutions) ve ters çevrilmiş artık bloklar (inverted residuals with linear bottlenecks) gibi tekniklerle verimlilik sağlar.

İyi Doğruluk/Performans dengesi boyutuna göre kabul edilebilir bir doğruluk sunar.

3.6.1.5. Raspberry Pi 5 (CPU ile çalıştığında)



Şekil 3.5. Raspberry Pi 5 (Kaynak: (2025 Haziran 20). Raspberry Pi 5 product brief (Rev 1.0). https://datasheets.raspberrypi.com/rpi5/raspberry-pi-5-product-brief.pdf).

Tercih edilme nedeni

Raspberry Pi gibi genel amaçlı CPU'lar, özel YZ hızlandırıcılarına sahip değildir. MobileNetV2'nin düşük hesaplama ihtiyacı, CPU üzerinde bile (diğer büyük modellere kıyasla) "kullanılabilir" bir performans sunmasını sağlar. Şekil 3.5'te, bu çalışmada temel CPU platformu olarak kullanılan Raspberry Pi 5 tek kart bilgisayarı görülmektedir. Görselde, merkezi işlem birimi (CPU), bellek (RAM), HDMI ve USB portları gibi temel donanım bileşenleri yer almaktadır. Bu platform, genel amaçlı bir işlemcinin derin öğrenme görevlerindeki performansını ve enerji tüketimini değerlendirmek için bir referans (baseline) noktası olarak kullanılmıştır. Ayrıca prototipleme ve basit YZ görevleri için erişilebilir ve uygun maliyetli bir çözümdür.

Avantajları

Düşük maliyet, Raspberry Pi'nin kendisi uygun fiyatlıdır ve ek bir hızlandırıcı gerektirmez.

Erişilebilirlik ve esneklik, geniş bir topluluk desteği, bolca kütüphane ve kolay programlanabilirlik sunar. CPU, genel amaçlı görevler için de kullanılabilir.

Düşük güç tüketimi (diğer CPU'lara göre), mobil veya gömülü projeler için görece düşük güç tüketir.

TensorFlow Lite desteği, TFLite gibi kütüphanelerle CPU üzerinde optimize edilmiş çıkarım yapılabilir.

Dezavantajları

Düşük performans, CPU, paralel YZ hesaplamaları için optimize edilmemiştir. Bu nedenle çıkarım hızları (FPS) oldukça düşük olacaktır. Gerçek zamanlı yüksek çözünürlüklü video analizi gibi görevler için yetersiz kalabilir.

Isınma, yoğun YZ işlemleri CPU'yu zorlayarak ısınmaya neden olabilir, ek soğutma gerektirebilir.

CPU yükü, YZ işlemi çalışırken CPU'nun büyük bir kısmı meşgul olacağı için diğer görevler yavaşlayabilir.

3.6.1.6. NVIDIA Jetson Xavier NX (GPU ile çalıştığında)

Tercih edilme nedenleri

Jetson Xavier NX, güçlü bir entegre NVIDIA GPU'suna sahiptir. MobileNetV2, bu GPU üzerinde çok yüksek hızlarda çalışabilir, bu da onu gerçek zamanlı ve karmaşık uygulamalar için uygun hale getirir. Ayrıca daha büyük modelleri de çalıştırabilse de, MobileNetV2 kullanmak, birden fazla modeli aynı anda çalıştırmak veya GPU kaynaklarını diğer görevler için serbest bırakmak anlamına gelebilir.

Avantajları

Yüksek performans, GPU'nun paralel işleme yetenekleri sayesinde çok yüksek çıkarım hızları (FPS) elde edilir.

TensorRT optimizasyonu, NVIDIA TensorRT kütüphanesi, MobileNetV2 gibi modelleri GPU üzerinde maksimum performans için optimize edebilir.

Çoklu akış desteği, birden fazla kamera akışını veya YZ görevini aynı anda işleyebilir.

Gelişmiş YZ yetenekleri, sadece çıkarım değil, bazı durumlarda yeniden eğitim (re-training) veya transfer öğrenme için de kullanılabilir.

Dezavantajları

Maliyet, Raspberry Pi'ye göre deutlich daha pahalıdır.

Güç tüketimi, CPU tabanlı çözümlere veya özel YZ hızlandırıcılara (TPU/NPU) göre daha fazla güç tüketir.

Karmaşıklık, kurulumu ve optimizasyonu (özellikle TensorRT ile) daha karmaşık olabilir.

Boyut ve ısı, daha büyük bir form faktörüne sahip olabilir ve daha fazla ısı üretebilir.

3.6.1.7. Google Coral USB Accelerator (TPU ile çalıştığında)

Tercih edilme nedenleri

Google Coral Edge TPU'su, özellikle TensorFlow Lite modellerini (MobileNetV2'nin TFLite versiyonları dahil) hızlandırmak için tasarlanmıştır. MobileNetV2'nin verimli yapısı, TPU'nun mimarisiyle çok iyi eşleşir. Ayrıca düşük güç tüketimiyle yüksek performanslı YZ çıkarımı hedeflenir.

Avantajları

Olağanüstü Hız/Watt oranı, çok düşük güç tüketimiyle çok yüksek çıkarım hızları sunar (özellikle kuantize edilmiş modellerde).

Kompakt ve taşınabilir, USB form faktörü, mevcut sistemlere (Raspberry Pi gibi) kolayca YZ hızlandırması eklemeyi sağlar.

Uygun maliyetli hızlandırma, GPU'lara göre daha uygun fiyatlı bir YZ hızlandırma çözümüdür.

TensorFlow Lite odaklı, TFLite modelleri için optimize edilmiştir ve kullanımı nispeten kolaydır.

Dezavantajları

Model uyumluluğu, en iyi performansı kuantize edilmiş (genellikle int8) TensorFlow Lite modelleriyle verir. Desteklenmeyen operasyonlar host CPU'sunda çalışır ve performansı düşürebilir.

Host cihaz gereksinimi, çalışmak için bir host cihaza (örneğin Raspberry Pi, Linux PC) ihtiyaç duyar.

TensorFlow ekosistemine bağımlılık, esas olarak TensorFlow ve TFLite ile çalışır. Diğer framework'ler için doğrudan destek sınırlıdır.

Sınırlı esneklik, GPU kadar genel amaçlı değildir; öncelikli olarak çıkarım hızlandırmaya odaklanır.

3.6.1.8. Khadas VIM3 Pro (NPU ile çalıştığında)

Tercih edilme nedenleri

Khadas VIM3 Pro, Amlogic A311D SoC üzerinde entegre bir sinirsel işleme birimi (NPU) içerir. Bu NPU, MobileNetV2 gibi CNN mimarilerini verimli bir şekilde çalıştırmak için tasarlanmıştır. Ayrıca tek kart üzerinde hem CPU hem de NPU sunarak kompakt ve güçlü bir YZ çözümü oluşturur.

Avantajları

Entegre çözüm, NPU'nun SoC'ye entegre olması, harici bir hızlandırıcıya ihtiyaç duymadan YZ performansı sağlar.

İyi Performans/Watt oranı, NPU'lar genellikle güç verimliliği yüksek olacak şekilde tasarlanır.

Kompakt boyut, tek kart bilgisayar (SBC) form faktöründe güçlü YZ yetenekleri sunar.

Çok yönlülük, hem CPU (Arm Cortex-A73/A53) hem de NPU'ya sahip olması, genel amaçlı görevler ve YZ görevleri için dengeli bir platform sunar.

Dezavantajları

Araç zinciri ve destek, NPU'nun kullanımı, Khadas tarafından sağlanan SDK ve araç zincirine bağlıdır. Bu araçların olgunluğu, dokümantasyonu ve topluluk desteği, NVIDIA veya Google ekosistemleri kadar yaygın olmayabilir.

Model dönüşümü ve optimizasyonu, modellerin NPU'da en iyi şekilde çalışması için özel dönüştürme ve optimizasyon adımları gerekebilir.

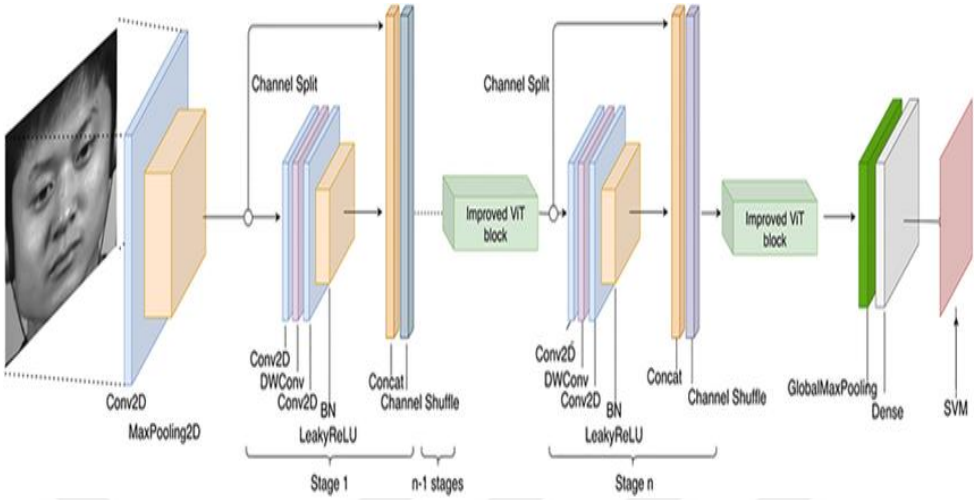
NPU sınırlamaları, tüm model operasyonları NPU tarafından desteklenmeyebilir; desteklenmeyenler CPU'ya düşebilir ve performansı etkileyebilir.

Maliyet, temel bir Raspberry Pi'den daha pahalıdır.

3.6.2. ShuffleNetV2

ShuffleNetV2 (Ma vd., 2018), özellikle mobil cihazlar gibi kaynak kısıtlı ortamlarda yüksek doğruluk ve düşük gecikme ile çalışabilecek şekilde tasarlanmış, hafif ve verimli bir evrişimsel sinir ağı mimarisidir. ShuffleNetV2, önceki sürümdeki grup evrişimlerinin hesaplama dengesizliğini ve bellek erişim sorunlarını gidermek amacıyla geliştirilmiştir. Bu modelde kanal bölme (channel split), hafif nokta evrişimler (lightweight pointwise convolutions) ve kanal karıştırma (channel shuffle) işlemleri kullanılarak hem bilgi akışı iyileştirilmiş hem de hesaplama verimliliği artırılmıştır. ShuffleNetV2'nin temel yapı taşı olan ShuffleNet bloğu, bilgi kaybı yaşamadan verilerin ağ boyunca etkili bir şekilde aktarılmasını sağlar. Ayrıca, bu mimari tasarımı sırasında pratik donanım ölçümleri (gerçek çalışma süresi ve bellek erişim süresi gibi) göz önünde bulundurularak optimizasyon yapılmıştır. Bu sayede ShuffleNetV2, teorik verimlilikten ziyade gerçek dünyadaki donanım performansına

dayalı olarak daha iyi sonuçlar elde eder. Bu özellikleriyle ShuffleNetV2, hem mobil görüntü işleme görevleri hem de zaman-kritik uygulamalar için ideal bir çözümdür.



Şekil 3.6. ShuffleNetV2 Blok Yapısı (Kaynak: Ma ve diğerleri, 2018).

Bu çalışmada, ImageNet veri seti üzerinde önceden eğitilmiş olan standart ShuffleNetV2 (örneğin, 1.0x ölçek faktörü ve 224x224 giriş çözünürlüğü ile, PyTorch Hub veya benzeri bir kaynaktan elde edilen) modeli temel alınmıştır. Eğitimde kullanılan ShuffleNetV2 blok yapısının uygulandığı görüntü sınıflandırmaya ilişkin örneği Şekil 3.6'da görülmektedir. Şekil 3.6, ShuffleNetV2 mimarisinin temel yapı taşı olan bloğun işleyişini detaylandırmaktadır. Bu şemada, verimliliği artırmak için kullanılan "Channel Split" (Kanal Bölme) ve "Channel Shuffle" (Kanal Karıştırma) gibi anahtar mekanizmalar gösterilmektedir. Bu yapı, modelin hesaplama maliyetini düşürürken bilgi akışını korumasını sağlar ve ShuffleNetV2'nin neden özellikle kaynak kısıtlı cihazlar için uygun bir mimari olduğunu açıklar.

PyTorch'tan elde edilen bu önceden eğitilmiş model, öncelikle ONNX (Open Neural Network Exchange) formatına, ardından da TensorFlow Lite formatına dönüştürülerek hedef donanım platformlarında kullanılmak üzere aşağıdaki optimizasyon adımlarına tabi tutulmuştur.

3.6.2.1. TensorFlow Lite (TFLite) dönüşümü

ONNX formatındaki model, ONNX-TensorFlow dönüştürücü aracı (onnx-tf) kullanılarak TensorFlow SavedModel formatına, ardından da TensorFlow Lite Converter aracı ile .tflite formatına dönüştürülmüştür. Bu dönüşüm sırasında, varsayılan ayarlar korunarak modelin FP32 (32-bit kayan nokta) hassasiyetinde kalması sağlanmıştır. Bu FP32 .tflite modeli, özellikle CPU (Raspberry Pi 5) ve GPU

(NVIDIA Jetson Xavier NX üzerinde TFLite GPU delegate ile) testleri için temel alınmıştır.

3.6.2.2. Tam sayı kuantizasyonu (Post-Training INT8 quantization)

Google Coral USB Accelerator (TPU) ve Khadas VIM3 Pro (NPU) gibi platformlarda optimum performans için, FP32 .tflite modeli 8-bit tam sayı (INT8) kuantizasyonuna tabi tutulmuştur. Bu işlem, TensorFlow Lite Converter aracı ile temsili bir veri seti kullanılarak kalibrasyon (representative dataset quantization) yöntemiyle gerçekleştirilmiştir. Kalibrasyon verisi olarak, ImageNet doğrulama setinden rastgele seçilmiş 200 adet görüntüden oluşan bir alt küme kullanılmıştır.

3.6.2.3. NVIDIA TensorRT optimizasyonu (Jetson Xavier NX için)

NVIDIA Jetson Xavier NX platformunda, ONNX formatındaki orijinal ShuffleNetV2 modelinden doğrudan NVIDIA TensorRT kütüphanesi ve trtexec aracı kullanılarak FP16 hassasiyetinde optimize edilmiş bir TensorRT çıkarım motoru oluşturulmuştur. Ayrıca, aynı araç ve kalibrasyon veri seti (ImageNet doğrulama setinden 200 örnek) ile INT8 hassasiyetinde bir TensorRT motoru da üretilmiştir.

3.6.2.4. ShuffleNetV2'nin genel özellikleri

Verimlilik odaklı tasarım ilkeleri, sadece FLOPs'u değil, aynı zamanda MAC'ı (Memory Access Cost) da düşürmeye odaklanır.

Kanal karıştırma (Channel shuffle), grup evrişimlerinden sonra farklı gruplar arasındaki bilgi akışını sağlar.

Parametre sayısı, yaklaşık 2.3 milyon parametre (1.0x ölçek faktörü için).

Hesaplama maliyeti (FLOPs), yaklaşık 140-150 MFLOPs (224x224 giriş, 1.0x ölçek faktörü için).

Model boyutu (Yaklaşık değerler):

Orijinal PyTorch (.pth) modeli: ~9 MB

ONNX modeli: ~8.8 MB

FP32 .tflite modeli: ~8.5 MB

INT8 kuantize edilmiş .tflite modeli (kalibrasyonlu): ~2.5 MB

TensorRT FP16 engine: ~4.5 MB

TensorRT INT8 engine: ~2.8 MB

Bellek kullanımı (RAM), çıkarım sırasında, MobileNetV2'ye kıyasla benzer veya biraz daha düşük bir tepe bellek kullanımı (FP32 için 40-80 MB, INT8 için 25-60 MB aralığında) sergilemektedir.

Doğrudan hız ölçümü, Tasarım kararları, hedef donanımlardaki gerçek çıkarım hızına göre alınmıştır.

3.6.2.5. Raspberry Pi 5 (CPU ile çalıştığında)

Tercih edilme nedenleri

Mükemmel uyum. ShuffleNetV2'nin MAC'ı azaltmaya yönelik tasarım ilkeleri, CPU gibi bellek bant genişliği sınırlı olabilen platformlarda çok iyi sonuç verir. En kısıtlı CPU ortamlarında bile kabul edilebilir bir hız/doğruluk dengesi sunar.

Avantajları

Üstün CPU verimliliği, düşük FLOPs ve optimize edilmiş MAC sayesinde CPU üzerinde (özellikle düşük FLOPs varyantlarında) daha hızlı çalışır.

Düşük kaynak tüketimi, CPU, RAM ve depolama üzerinde minimal yük.

Düşük maliyet ve erişilebilirlik, Raspberry Pi platformunun genel avantajları.

TensorFlow Lite desteği, TFLite ile CPU'da verimli bir şekilde çalıştırılabilir.

Dezavantajları

Doğruluk sınırları, çok karmaşık görevler için ResNet-18 gibi daha büyük modeller kadar yüksek doğruluk sunmayabilir.

CPU performans limiti, yine de bir CPU'dur ve özel hızlandırıcıların hızına ulaşamaz.

3.6.2.6. NVIDIA Jetson Xavier NX (GPU ile çalıştığında)

Tercih edilme nedenleri

Tek bir ShuffleNetV2 örneği için Jetson Xavier NX genellikle aşırı güçlüdür.

Çok sayıda paralel akış işlenecekse veya GPU kaynakları diğer ağır görevler için korunuyorsa ve ShuffleNetV2 yardımcı bir görev için minimum kaynakla çalıştırılacaksa anlamlıdır.

ShuffleNetV2'nin farklı boyutları (0.5x, 1x, 1.5x, 2x gibi) mevcuttur. Daha büyük ShuffleNetV2 varyantları GPU'da daha anlamlı olur.

Avantajları

Çok yüksek FPS, GPU üzerinde son derece hızlı çalışacaktır.

TensorRT optimizasyonu, performansı daha da artırabilir.

Düşük GPU kullanımı (küçük varyantlar için), GPU kaynaklarının büyük bir kısmını serbest bırakır.

Dezavantajları

Genellikle aşırı güçlü (Overkill), özellikle küçük ShuffleNetV2 varyantları için GPU'nun potansiyeli tam olarak kullanılmamış olur.

Maliyet ve güç tüketimi, platformun maliyeti ve güç tüketimi, sadece hafif bir model çalıştırılacaksa yüksek kalabilir.

3.6.2.7. Google Coral USB Accelerator (TPU ile çalıştığında)

Tercih edilme nedenleri

Çok iyi bir eşleşme. ShuffleNetV2'nin verimli yapısı, kanal karıştırma ve grup evrişimleri (uygun şekilde TFLite'a dönüştürüldüğünde) Edge TPU'da iyi performans gösterebilir.

En düşük güç tüketimiyle yüksek çıkarım hızları hedeflendiğinde ve MobileNetV2'den bile daha hafif bir çözüm arandığında kullanılır.

Avantajları

Mükemmel Hız/Watt oranı, özellikle kuantize edilmiş TFLite modellerinde çok verimlidir.

Kompakt ve taşınabilir, YZ hızlandırmasını kolayca ekler.

TensorFlow Lite odaklı, TFLite için optimize edilmiştir. ShuffleNetV2'nin TFLite sürümleri genellikle iyi çalışır.

Dezavantajları

Model uyumluluğu ve optimizasyon, en iyi performans için iyi kuantize edilmiş TFLite modelleri gerekir. Kanal karıştırma gibi özel operasyonların Edge TPU tarafından verimli bir şekilde desteklenmesi önemlidir (genellikle desteklenir veya verimli bir şekilde ayrıştırılır).

Host cihaz gereksinimi, çalışmak için bir host cihaza ihtiyaç duyar.

Doğruluk, diğer hafif modeller gibi, doğrulukta bazı ödünler verilebilir.

3.6.2.8. Khadas VIM3 Pro (NPU ile çalıştığında)

Tercih edilme nedenleri

ShuffleNetV2'nin verimlilik ilkeleri, Khadas VIM3 Pro'nun NPU'su ile iyi uyum sağlar. Entegre, güç verimli ve kompakt bir YZ çözümü arandığında, SqueezeNet'e göre potansiyel olarak daha iyi bir doğruluk/hız dengesi sunabilir.

Avantajları

Entegre verimli çözüm, NPU, SoC'ye entegredir.

İyi Performans/Watt oranı, NPU, CNN operasyonlarını (kanal karıştırma dahil) verimli bir şekilde hızlandırabilir.

Kompakt boyut, SBC form faktöründe iyi YZ yetenekleri.

Dezavantajları

Araç zinciri ve model dönüşümü, ShuffleNetV2'nin NPU'da optimum performansla çalışması, Khadas/Amlogic tarafından sağlanan SDK ve model dönüştürme araçlarının bu mimariyi ne kadar iyi desteklediğine bağlıdır. Kanal karıştırma ve grup evrişimlerinin doğru ve verimli bir şekilde haritalanması önemlidir.

NPU sınırlamaları, NPU'nun desteklediği operasyonlar kritik öneme sahiptir.

Dokümantasyon ve topluluk, diğer büyük platformlara göre daha sınırlı olabilir.

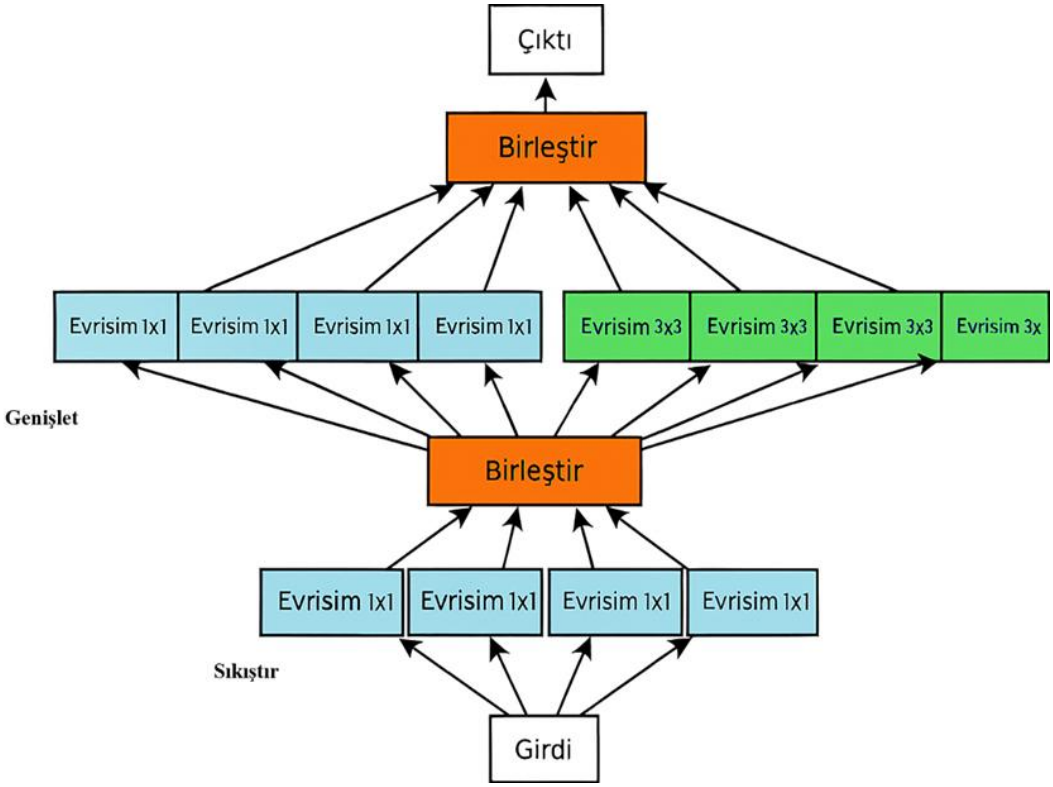
3.6.3. SqueezeNet

SqueezeNet (Iandola vd., 2016), sınırlı bellek ve işlem gücüne sahip cihazlarda kullanılmak üzere tasarlanmış, parametre sayısı açısından son derece düşük ancak doğruluk açısından rekabetçi performans sunan bir evrişimsel sinir ağı mimarisidir. Bu modelin temel hedefi, model boyutunu küçültmek ve parametre verimliliğini artırmak olmuştur. SqueezeNet, bu amaca ulaşmak için "Fire module" adı verilen özel bir yapı kullanır. Fire modülü, iki temel bileşenden oluşur.

Squeeze katmanı (1×1 evrişimlerle kanal sayısını azaltır) ve ardından gelen Expand katmanı (hem 1×1 hem de 3×3 evrişimlerle genişletme yapar). Bu yapı sayesinde, SqueezeNet hem düşük parametre sayısını korurken hem de öğrenme kapasitesinden ödün vermez. Ayrıca SqueezeNet'te, model boyutunu daha da azaltmak amacıyla geç evrişim (late downsampling) stratejisi uygulanmıştır; bu sayede erken evrişim katmanlarında daha fazla bilgi korunur. Toplamda 50× daha az parametreye sahip olmasına rağmen, SqueezeNet, AlexNet düzeyinde doğruluk elde etmeyi başarmıştır.

Bu özellikleriyle SqueezeNet, gömülü sistemler, IoT cihazları ve gerçek zamanlı uygulamalar için son derece uygun bir derin öğrenme modelidir. SqueezeNet mimarisindeki fire modül yapısı örneği Şekil 3.7 de gösterilmiştir.

Şekil 3.7, SqueezeNet'in temel yapı taşı olan ve modelin aşırı hafif olmasını sağlayan "Fire Modülü" nün mimarisini göstermektedir. Modül, "Sıkıştır" (Squeeze) katmanında 1x1 evrişimlerle kanal sayısını azaltır ve ardından "Genişlet" (Expand) katmanında 1x1 ve 3x3 evrişimlerle özellik haritalarını yeniden zenginleştirir. Bu yapı, parametre sayısını dramatik bir şekilde azaltırken öğrenme kapasitesini korumayı hedefler ve SqueezeNet'in verimliliğinin arkasındaki temel prensibi ortaya koyar.



Şekil 3.7. SqueezeNet Mimarisindeki Fire Modül Yapısı (Kaynak: Liu ve diğerleri, 2024).

Ortaya Bu çalışmada, ImageNet veri seti üzerinde önceden eğitilmiş olan SqueezeNet (sürüm 1.1, PyTorch Hub veya torchvision.models üzerinden erişilen versiyonu) temel alınmıştır. PyTorch formatındaki bu önceden eğitilmiş model, diğer modellerde olduğu gibi öncelikle ONNX formatına, ardından da daha uygun hale getirilmesi için TensorFlow Lite formatına dönüştürülerek, farklı hedef donanım platformlarında kullanılmak üzere aşağıdaki optimizasyon adımlarına tabi tutulmuştur. Bu süreç,

model performansını artırmak ve kaynak kullanımını minimize etmek amacıyla özenle gerçekleştirilmiştir. Bu sayede, modelin verimliliği ve performansı arttırılmış olur.

3.6.3.1. TensorFlow Lite (TFLite) dönüşümü

ONNX formatındaki model, ONNX-TensorFlow dönüştürücü aracı (onnx-tf) kullanılarak TensorFlow SavedModel formatına, daha sonra da TensorFlow Lite Converter aracı ile. tflite formatına dönüştürülmüştür. Bu dönüşüm sırasında, varsayılan ayarlar korunarak modelin FP32 (32-bit kayan nokta) hassasiyetinde kalması sağlanmıştır. Bu FP32.tflite modeli, CPU (Raspberry Pi 5) ve GPU (NVIDIA Jetson Xavier NX üzerinde TFLite GPU delegate ile) testleri için kullanılmıştır.

3.6.3.2. Tam sayı kuantizasyonu (Post-Training INT8 quantization)

Google Coral USB Accelerator (TPU) ve Khadas VIM3 Pro (NPU) gibi platformlarda en iyi performansı elde etmek için, FP32.tflite modeli 8-bit tam sayı (INT8) kuantizasyonuna tabi tutulmuştur. Bu işlem, TensorFlow Lite Converter aracı ile temsili bir veri seti kullanılarak kalibrasyon (representative dataset quantization) yöntemiyle gerçekleştirilmiştir. Kalibrasyon verisi olarak, ImageNet doğrulama setinden rastgele seçilmiş 150 adet görüntüden oluşan bir alt küme kullanılmıştır.

3.6.3.3. NVIDIA TensorRT optimizasyonu (Jetson Xavier NX için)

NVIDIA Jetson Xavier NX platformunda, ONNX formatındaki SqueezeNet modelinden doğrudan NVIDIA TensorRT kütüphanesi ve trtexec aracı kullanılarak FP16 hassasiyetinde optimize edilmiş bir TensorRT çıkarım motoru oluşturulmuştur. Ayrıca, aynı araç ve kalibrasyon veri seti (ImageNet doğrulama setinden 150 örnek) ile INT8 hassasiyetinde bir TensorRT motoru da üretilerek karşılaştırma yapılmıştır.

3.6.3.4. SqueezeNet'in genel özellikleri

Aşırı hafiflik, çok düşük parametre sayısı ve minimal hesaplama maliyeti (FLOPs).

Parametre sayısı, yaklaşık 1.25 milyon parametre (SqueezeNet 1.1 için).

Hesaplama maliyeti (FLOPs), yaklaşık 350-400 MFLOPs (SqueezeNet 1.1, 224x224 giriş için – not: SqueezeNet'in FLOPs değeri, rakip modellere göre parametre sayısına oranla biraz daha yüksek olabilir).

Model boyutu (Yaklaşık değerler),

Orijinal PyTorch (.pth) modeli: ~5 MB

ONNX modeli: ~4.8 MB

FP32. tflite modeli: ~4.7 MB

INT8 kuantize edilmiş. tflite modeli (kalibrasyonlu): ~1.3 MB

TensorRT FP16 engine: ~2.5 MB

TensorRT INT8 engine: ~1.5 MB

Bellek kullanımı (RAM), çıkarım sırasında, test edilen modeller arasında en düşük tepe bellek kullanımına sahip olması beklenir (FP32 için 30-60 MB, INT8 için 20-40 MB aralığında).

"Fire Module" (Ateş modülü), "Squeeze" ve "Expand" katmanlarını kullanarak verimlilik sağlar.

Doğruluk/Boyut dengesi, çok küçük boyutuna rağmen şaşırtıcı derecede iyi bir doğruluk sunmayı amaçlar.

3.6.3.5. Raspberry Pi 5 (CPU ile çalıştığında)

Tercih edilme nedenleri

En ideal senaryolardan biri. Raspberry Pi gibi CPU'su sınırlı cihazlarda, SqueezeNet'in aşırı hafifliği onu mükemmel bir aday yapar. MobileNetV2'nin bile yavaş kaldığı veya çok fazla kaynak tükettiği durumlar için harikadır.

Basit görüntü sınıflandırma görevleri, temel nesne algılama (hafif bir algılama başlığıyla) gibi uygulamalar için uygundur.

Avantajları

Maksimum kaynak verimliliği, CPU, RAM ve depolama alanı üzerinde minimum yük oluşturur.

En hızlı CPU performansı (Hafif modeller arasında), diğer hafif modellere kıyasla CPU üzerinde en hızlı çıkarım sürelerinden birini sunabilir.

Düşük maliyet ve erişilebilirlik, Raspberry Pi platformunun genel avantajları geçerlidir.

TensorFlow Lite desteği, TFLite ile CPU'da daha da optimize edilebilir.

Dezavantajları

Sınırlı doğruluk, daha karmaşık görevler veya daha yüksek doğruluk gerektiren senaryolar için yetersiz kalabilir. ResNet-18 veya hatta MobileNetV2'ye göre doğruluktan ödün verilmesi gerekebilir.

CPU performans sınırları, ne kadar hafif olursa olsun, CPU'nun mutlak performansı özel hızlandırıcılarla kıyaslanamaz.

3.6.3.6. NVIDIA Jetson Xavier NX (GPU ile çalıştığında)

Tercih edilme nedenleri

Jetson Xavier NX gibi güçlü bir GPU için SqueezeNet aşırı derecede hafif kalır. Tek bir SqueezeNet örneği çalıştırmak için genellikle bu kadar güçlü bir GPU'ya ihtiyaç duyulmaz. Ancak, aynı anda çok sayıda SqueezeNet örneği çalıştırmak (örneğin, çok sayıda düşük çözünürlüklü kamera akışını işlemek) veya GPU kaynaklarının büyük bir kısmını başka daha ağır modellere/görevlere ayırıp, SqueezeNet'i yardımcı bir görev için minimum kaynakla çalıştırmak istendiğinde mantıklı olabilir.

Avantajları

Olağanüstü yüksek FPS, SqueezeNet, Jetson GPU'sunda inanılmaz yüksek hızlarda çalışacaktır.

TensorRT optimizasyonu, TensorRT ile daha da hızlandırılabilir.

Minimum GPU kullanımı (Tek örnek için), GPU kaynaklarının çok küçük bir kısmını kullanır, bu da diğer görevler için bolca yer bırakır.

Dezavantajları

Genellikle aşırı güçlü (Overkill), tek bir SqueezeNet görevi için Jetson Xavier NX kullanmak, kaynakların verimsiz kullanılması anlamına gelebilir.

Maliyet ve güç tüketimi, sadece SqueezeNet çalıştırılacaksa, platformun maliyeti ve güç tüketimi, elde edilen göreve göre yüksek kalır.

3.6.3.7. Google Coral USB Accelerator (TPU ile çalıştığında)

Tercih edilme nedenleri

Mükemmel bir eşleşme. SqueezeNet'in verimliliği ve küçük boyutu, Edge TPU'nun mimarisiyle çok iyi uyum sağlar. Özellikle kuantize edilmiş (int8) TensorFlow Lite versiyonları için idealdir. En düşük güç tüketimiyle çok yüksek çıkarım hızları hedeflendiğinde tercih edilir.

Avantajları

Muazzam Hız/Watt oranı, SqueezeNet, Coral Edge TPU üzerinde çok az güç tüketerek son derece yüksek hızlarda çalışabilir.

Kompakt ve taşınabilir, mevcut sistemlere kolayca yüksek performanslı, düşük güçlü YZ çıkarımı ekler.

TensorFlow Lite odaklı, TFLite modelleri için optimize edilmiştir ve SqueezeNet'in TFLite versiyonları genellikle sorunsuz çalışır.

Dezavantajları

Model uyumluluğu (Genellikle sorunsuz), en iyi performans için kuantize edilmiş TFLite modelleri gerekir. SqueezeNet'in yapısı genellikle TFLite ve Edge TPU ile uyumludur.

Host cihaz gereksinimi, çalışmak için bir host cihaza ihtiyaç duyar.

Doğruluk sınırlamaları, SqueezeNet'in doğası gereği doğruluk, daha büyük modellere göre sınırlı olabilir.

3.6.3.8. Khadas VIM3 Pro (NPU ile çalıştığında)

Tercih edilme nedenleri

Coral TPU'ya benzer şekilde, SqueezeNet'in hafif yapısı Khadas VIM3 Pro'nun NPU'su için çok uygundur. Entegre, kompakt ve güç verimli bir YZ çözümü arandığında ve SqueezeNet'in doğruluk seviyesi yeterli olduğunda tercih edilebilir.

Avantajları

Entegre verimli çözüm, NPU, SoC'ye entegre olduğu için harici hızlandırıcı gerekmez.

İyi Performans/Watt oranı, NPU, SqueezeNet gibi CNN'leri verimli bir şekilde hızlandırır.

Kompakt boyut, SBC form faktöründe YZ yetenekleri sunar.

Dezavantajları

Araç zinciri ve destek, NPU'nun kullanımı, Khadas tarafından sağlanan SDK ve araç zincirine bağlıdır. SqueezeNet gibi standart bir modelin dönüştürülmesi genellikle daha kolaydır.

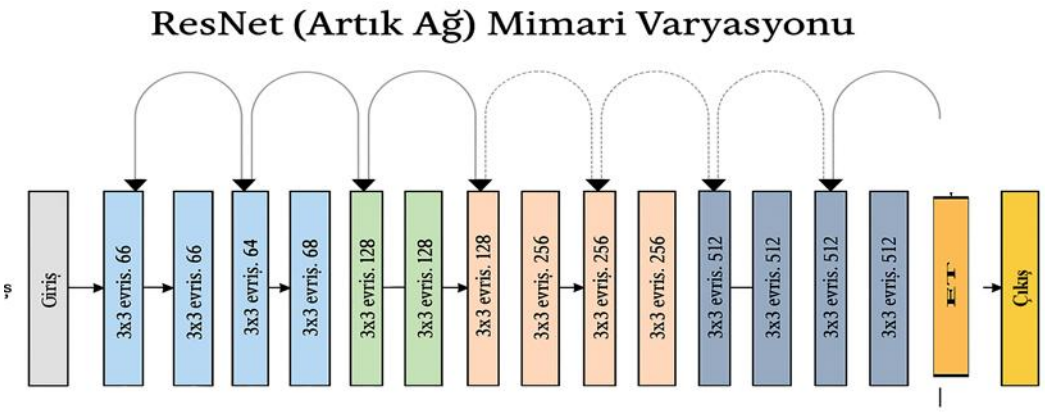
Model dönüşümü, modellerin NPU'da en iyi şekilde çalışması için özel dönüştürme adımları gerekebilir, ancak SqueezeNet için bu genellikle daha basittir.

Doğruluk sınırlamaları, yine, SqueezeNet'in doğası gereği doğruluk ana odak noktası değildir.

3.6.4. ResNet-18

ResNet-18 (He vd., 2016), derin sinir ağlarında eğitim sırasında karşılaşılan gradyan sönümlenmesi (vanishing gradient) sorununu çözmek ve çok katmanlı yapıların etkinliğini artırmak amacıyla geliştirilmiş Artık Ağlar (Residual Networks) ailesinin bir üyesidir. Bu mimaride, artık bağlantılar (residual connections) kullanılarak öğrenme işlemi kolaylaştırılmış, katmanlar arası doğrudan bilgi aktarımı sağlanmıştır. ResNet-18, toplamda 18 ağırlık katmanından (evrişim ve tam bağlantılı) oluşur ve her biri iki evrişim katmanı içeren residual bloklar kullanır. Bu bloklar, giriş ile çıkış arasında doğrudan bir toplama işlemi gerçekleştirerek modelin daha derin yapılara kolayca genişletilebilmesini mümkün kılar.

ResNet mimarisinin temelinde yer alan bu kısa devre bağlantıları (skip connections), ağırlıklı katmanların öğrenmesi zor olan kimlik fonksiyonlarını daha kolay şekilde öğrenmesine olanak tanır. Böylece eğitim süresi kısalırken doğruluk artışı da sağlanır. ResNet-18, daha derin sürümleri olan ResNet-34, ResNet-50 ve ResNet-101'e kıyasla daha az parametreye ve daha düşük hesaplama maliyetine sahiptir; bu da onu diğerlerine göre daha sınırlı kaynaklara sahip sistemlerde kullanılabilir hale getirir. Aynı zamanda, daha sığ yapısına rağmen yüksek doğruluk performansı sunması sayesinde birçok görüntü sınıflandırma ve transfer öğrenme uygulamasında yaygın olarak tercih edilmektedir. ResNet mimari varyasyonunun gösterimi Şekil 3.8 de gösterilmiştir.



Şekil 3.8. ResNet (Residual Network) Mimari Varyasyonunun Gösterimi (Kaynak: Cao ve diğerleri, 2022).

Şekil 3.8, ResNet (Artık Ağ) mimarisinin temel prensibini ve katmanlı yapısını göstermektedir. Şemadaki kısa devre bağlantıları (skip connections), gradyanların ağın daha derin katmanlarına sorunsuzca akmasını sağlayarak çok daha derin modellerin eğitilmesine olanak tanır. Bu mimari, bu çalışmada daha yüksek doğruluk potansiyeli sunan, ancak daha yüksek hesaplama maliyetine sahip bir modelin temsilcisi olarak yer almaktadır.

Bu çalışmada, ImageNet veri seti üzerinde önceden eğitilmiş olan standart ResNet-18 modeli (PyTorch Hub veya torchvision.models üzerinden erişilen versiyonu) temel alınmıştır. PyTorch formatındaki bu önceden eğitilmiş model, diğer modellerde olduğu gibi öncelikle ONNX formatına, ardından da TensorFlow Lite formatına dönüştürülerek hedef donanım platformlarında kullanılmak üzere aşağıdaki optimizasyon adımlarına tabi tutulmuştur.

3.6.4.1. TensorFlow Lite (TFLite) dönüşümü

ONNX formatındaki model, ONNX-TensorFlow dönüştürücü aracı (onnx-tf) kullanılarak TensorFlow SavedModel formatına, daha sonra da TensorFlow Lite Converter aracı ile .tflite formatına dönüştürülmüştür. Bu dönüşüm sırasında, varsayılan ayarlar korunarak modelin FP32 (32-bit kayan nokta) hassasiyetinde kalması sağlanmıştır. Bu FP32 .tflite modeli, CPU (Raspberry Pi 5) ve GPU (NVIDIA Jetson Xavier NX üzerinde TFLite GPU delegate ile) testleri için kullanılmıştır.

3.6.4.2. Tam sayı kuantizasyonu (Post-training INT8 quantization)

Google Coral USB Accelerator (TPU) ve Khadas VIM3 Pro (NPU) gibi platformlarda performans optimizasyonu için, FP32 .tflite modeli 8-bit tam sayı (INT8) kuantizasyonuna tabi tutulmuştur. Bu işlem, TensorFlow Lite Converter aracı ile temsili bir veri seti kullanılarak kalibrasyon (representative dataset quantization) yöntemiyle gerçekleştirilmiştir.Kalibrasyon verisi olarak, ImageNet doğrulama setinden rastgele seçilmiş 300 adet görüntüden oluşan bir alt küme kullanılmıştır. ResNet mimarisindeki artık bağlantılar nedeniyle, kuantizasyon sonrası doğruluk düşüşünün diğer modellere kıyasla daha dikkatli izlenmesi gerekmiştir.

3.6.4.3. NVIDIA TensorRT optimizasyonu (Jetson Xavier NX için)

NVIDIA Jetson Xavier NX platformunda, ONNX formatındaki orijinal ResNet-18 modelinden doğrudan NVIDIA TensorRT kütüphanesi ve trtexec aracı kullanılarak FP16 hassasiyetinde optimize edilmiş bir TensorRT çıkarım motoru oluşturulmuştur. Ayrıca, aynı araç ve kalibrasyon veri seti (ImageNet doğrulama setinden 300 örnek) ile INT8 hassasiyetinde bir TensorRT motoru da üretilerek performansı ve doğruluk üzerindeki etkisi değerlendirilmiştir.

3.6.4.4. ResNet-18'in genel özellikleri

Derinlik ve doğruluk, diğer hafif modellere kıyasla daha fazla katmana sahiptir ve genellikle daha karmaşık özellikleri öğrenebilir, bu da daha yüksek doğruluk sağlayabilir.

Artık bağlantılar (Residual connections), çok derin ağların eğitilmesini kolaylaştırır.

Parametre sayısı, yaklaşık 11.7 milyon parametre.

Hesaplama maliyeti (FLOPs), yaklaşık 1.8 GFLOPs (224x224 giriş için).

Model boyutu (Yaklaşık değerler),

Orijinal PyTorch (.pth) modeli: ~45 MB

ONNX modeli: ~44 MB

FP32 .tflite modeli: ~43 MB

INT8 kuantize edilmiş .tflite modeli (kalibrasyonlu): ~11-12 MB

TensorRT FP16 engine: ~22 MB

TensorRT INT8 engine: ~12-13 MB

Bellek kullanımı (RAM), çıkarım sırasında, test edilen diğer hafif modellere (MobileNetV2, ShuffleNetV2, SqueezeNet) kıyasla daha yüksek bir tepe bellek kullanımı sergilemesi beklenir (FP32 için 150-250 MB, INT8 için 80-150 MB aralığında).

Hesaplama maliyeti, MobileNetV2'den daha fazla parametreye ve FLOPs'a sahiptir.

3.6.4.5. Raspberry Pi 5 (CPU ile çalıştığında)

Tercih edilme nedenleri

Eğer MobileNetV2'nin doğruluğu projeniz için yeterli değilse ve ResNet-18'in sağlayacağı ek doğruluk kritikse tercih edilebilir.

Çıkarım hızı çok önemli değilse, yani bu durum özellikle çevrimdışı analizler veya çok düşük kare hızlı uygulamalar gibi senaryoları içeriyorsa, o zaman farklı yaklaşımlar üzerinde durmak daha uygun olabilir.

Avantajları

Potansiyel olarak daha yüksek doğruluk, daha karmaşık görevlerde MobileNetV2'ye göre daha iyi sonuçlar verebilir.

Düşük maliyet ve erişilebilirlik, Raspberry Pi platformunun genel avantajları geçerlidir.

TensorFlow Lite desteği, TFLite ile CPU üzerinde optimizasyon yapılabilir, ancak yine de MobileNetV2'ye göre daha yavaş olacaktır.

Dezavantajları

Çok düşük performans, ResNet-18, MobileNetV2'den daha ağırdır ve CPU üzerinde çalıştırıldığında çıkarım hızları (FPS) oldukça düşük olacaktır. Birçok gerçek zamanlı uygulama için pratik olmayabilir.

Yüksek CPU yükü, CPU'nun büyük bir kısmını meşgul eder, diğer görevleri yavaşlatabilir. Artan ısınma, daha yoğun hesaplamalar nedeniyle CPU daha fazla ısınabilir.

3.6.4.6. NVIDIA Jetson Xavier NX (GPU ile çalıştığında)

Tercih edilme nedenleri

Hem yüksek doğruluk (ResNet-18'in sunduğu) hem de yüksek çıkarım hızına ihtiyaç duyulduğunda idealdir. Jetson Xavier NX'in GPU'su ResNet-18 gibi modelleri

rahatlıkla çalıştırabilir. Robotik, otonom araçlar, gelişmiş video analizi gibi karmaşık YZ uygulamaları için uygundur.

Avantajları

Yüksek performans ve doğruluk dengesi, GPU, ResNet-18'in hesaplama yükünü kaldırarak iyi bir çıkarım hızı sunarken, modelin doğruluk avantajlarından faydalanılır.

TensorRT optimizasyonu, NVIDIA TensorRT, ResNet-18'i GPU'da maksimum performans için optimize edebilir.

Esneklik, daha büyük ve karmaşık modelleri destekleyebilir.

Dezavantajları

Maliyet, Raspberry Pi'ye göre deutlich daha pahalıdır.

Güç tüketimi, CPU tabanlı veya bazı özel YZ hızlandırıcılara göre daha fazla güç tüketir. ResNet-18 çalıştırılırken bu tüketim MobileNetV2'ye göre biraz daha fazla olabilir. Karmaşıklık, kurulum ve optimizasyon (TensorRT) daha karmaşık olabilir.

3.6.4.7. Google Coral USB Accelerator (TPU ile çalıştığında)

Tercih edilme nedenleri

ResNet-18'in doğruluğuna ihtiyaç duyuluyorsa ve model TensorFlow Lite formatına (özellikle int8 kuantize edilmiş) başarılı bir şekilde dönüştürülebiliyorsa tercih edilir. Düşük güç tüketimiyle yüksek hızlı çıkarım hedeflendiğinde tercih edilmektedir.

Avantajları

İyi Hız/Watt oranı, eğer ResNet-18 modeli Edge TPU için iyi optimize edilmişse (tüm operasyonları destekleniyorsa ve kuantizasyon iyi sonuç veriyorsa), düşük güçle iyi bir hız sunabilir.

Kompakt ve taşınabilir, mevcut sistemlere YZ hızlandırması eklemek için pratiktir.

Dezavantajları

Model uyumluluğu ve optimizasyon zorlukları, ResNet-18, MobileNetV2'ye göre daha karmaşık bir yapıya (özellikle artık bağlantılar) sahip olabilir. Tüm operasyonlarının Edge TPU tarafından tam olarak desteklenmemesi veya kuantizasyon sırasında doğruluk kaybının daha belirgin olması riski vardır. Desteklenmeyen operasyonlar host CPU'suna düşerek performansı ciddi şekilde etkileyebilir.

Host cihaz gereksinimi, çalışmak için bir host cihaza ihtiyaç duyar.

TensorFlow ekosistemine bağımlılık, en iyi sonuçlar TFLite ile alınır.

Performans belirsizliği, ResNet-18'in coral üzerindeki performansı, MobileNetV2'ye kıyasla modelin belirli varyantına ve dönüştürme sürecinin başarısına daha fazla bağlıdır.

3.6.4.8. Khadas VIM3 Pro (NPU ile çalıştığında)

Tercih edilme nedenleri

ResNet-18'in doğruluğuna ihtiyaç duyulduğunda ve modelin Khadas'ın NPU araç zinciriyle verimli bir şekilde çalıştırılabileceği durumlarda tercih edilir.

Entegre, kompakt ve güç verimli bir YZ çözümü arandığında tercih edilmektedir.

Avantajları

Entegre güçlü çözüm, NPU'nun SoC'ye entegre olması avantajlıdır. Amlogic A311D'nin NPU'su belirli bir TOK_TOKEN_DOTS (Trillions/Tera Operations Per Second) kapasitesine sahiptir.

Potansiyel olarak iyi Performans/Watt, NPU, CNN operasyonlarını verimli bir şekilde hızlandırabilir.

Dezavantajları

Araç zinciri ve model dönüşümü, ResNet-18 gibi bir modelin NPU'da optimum performansla çalışması, Khadas/Amlogic tarafından sağlanan SDK ve model dönüştürme araçlarının yeteneklerine bağlıdır. Bu süreç MobileNetV2'ye göre daha fazla dikkat ve ince ayar gerektirebilir.

NPU sınırlamaları ve destek, NPU'nun desteklediği operasyon seti ve katman türleri önemlidir. ResNet-18'deki bazı spesifik katmanlar veya konfigürasyonlar tam hızlandırılamayabilir. Dokümantasyon ve topluluk, NVIDIA veya Google kadar geniş olmayabilir, bu da optimizasyon sürecini zorlaştırabilir.

3.7. Donanım Mimarilerinin Seçimi

Bu çalışmada kullanılacak donanım mimarileri, test edilecek derin öğrenme modellerinin farklı işlemci türleri (CPU, GPU, TPU, NPU) üzerindeki performansını, enerji verimliliğini ve karbon ayak izini kapsamlı bir şekilde karşılaştırabilmek amacıyla seçilmiştir. Bu seçim, güncel uç bilişim (edge computing) ve gömülü sistem senaryolarında yaygın olarak karşılaşılan, farklı maliyet, performans ve güç tüketimi

profillerine sahip platformları temsil etmeyi hedeflemektedir. Bölümün devamında, seçilen her bir donanım platformunun temel özellikleri ve bu çalışmadaki rolleri kısaca tanıtılacaktır.

Çalışmada kullanılan donanım mimarileri ve derin öğrenme modelleri, enerji tüketimi, performans ve karbon ayak izi optimizasyonu açısından karşılaştırmalı bir analiz yapmak amacıyla seçilmiştir. Donanım mimarileri, farklı işlemci birimlerini (CPU, GPU, TPU, NPU) temsil edecek şekilde belirlenmiştir. Derin öğrenme modelleri ise, hafif ve verimli modeller olmaları nedeniyle uç cihazlarda yaygın olarak kullanılan modeller arasından seçilmiştir.

3.7.1. Raspberry Pi 5 (CPU)

Genel amaçlı işlemci birimi (CPU) tabanlı, düşük maliyetli ve yaygın olarak kullanılan bir tek kart bilgisayardır (SBC). Bu platform, yazılım tabanlı çıkarım performansının ve genel amaçlı işlemcilerin enerji verimliliğinin değerlendirilmesi için temel bir referans noktası olarak kullanılmıştır.

3.7.2. NVIDIA Jetson Xavier NX (GPU)

Entegre bir grafik işlemci birimine (GPU) sahip olan bu platform, yüksek performanslı derin öğrenme uygulamaları için optimize edilmiştir. GPU hızlandırmasının ve NVIDIA'nın TensorRT gibi yazılım ekosistemlerinin etkisini incelemek amacıyla seçilmiştir.

3.7.3. Google Coral USB Accelerator (TPU)

Harici bir USB hızlandırıcı olan bu cihaz, Google tarafından geliştirilen bir Tensor İşlemci Birimi (TPU) içermektedir. Özellikle TensorFlow Lite modellerini düşük güç tüketimiyle hızlandırmak için tasarlanmış olan bu platform, özelleşmiş YZ hızlandırıcılarının performansını değerlendirmek için kullanılmıştır.

3.7.4. Khadas VIM3 Pro (NPU)

Üzerinde entegre bir Sinir İşlemci Birimi (NPU) bulunan Amlogic A311D SoC'ye sahip bir tek kart bilgisayardır. Bu platform, SoC'ye entegre NPU'ların enerji verimliliği ve performans potansiyelini incelemek amacıyla seçilmiştir.

Derin öğrenme modelleri olarak MobileNetV2, ShuffleNetv2, SqueezeNet ve ResNet18 seçilmiştir. Bu modeller, uç cihazlarda yaygın olarak kullanılan ve farklı optimizasyon tekniklerine uygun modellerdir.

3.8. Veri Setleri ve Tahmini Çalışmalarının Planlanması

Bu kısımda, bölüm 3.1'de detayları verilen veri setleri üzerinde, bölüm 3.6'de tanıtılan ve optimize edilen derin öğrenme modellerinin performansını bölüm 3.7'te açıklanan donanım platformlarında değerlendirmek için izlenen deneysel kurulum ve kullanılan değerlendirme metrikleri açıklanmaktadır. Çalışmanın temel amacı, farklı model-donanım kombinasyonlarının karşılaştırmalı bir analizini yapmak olduğundan, tüm deneyler standartlaştırılmış bir protokol izlenerek gerçekleştirilmiştir. Bu yaklaşım, sonuçların güvenilirliğini ve çalışmanın yeniden üretilebilirliğini artırmayı hedeflemektedir.

3.8.1. Deneysel prosedür

Her bir donanım platformu üzerinde, seçilen her bir optimize edilmiş derin öğrenme modeli bölüm 3.2 ile CIFAR-10, CIFAR-100 ve ImageNet (alt küme) veri setlerinin bölüm 3.1 test alt kümeleri kullanılarak çıkarım (inference) işlemleri gerçekleştirilmiştir. Tüm deneylerde aşağıdaki adımlar izlenmiştir:

Veri ön işleme, çıkarım işleminden önce, test görüntülerinin tümü modellerin beklediği giriş boyutuna (örneğin, 224x224 piksel) yeniden boyutlandırılmış ve piksel değerleri genellikle [0, 1] aralığına veya [-1, 1] aralığına normalize edilmiştir. Piksel değerleri 255'e bölünerek [0,1] aralığına normalize edilmiştir. Her model için, orijinal eğitiminde kullanılan ön işleme adımları takip edilmeye çalışılmıştır.

Model yükleme, ilgili platform için optimize edilmiş model dosyası (örneğin, .tflite, TensorRT engine) çalışma zamanı motoruna (runtime engine) yüklenmiştir.

Çıkarım ve performans kaydı, her bir test görüntüsü için model üzerinden sınıflandırma tahmini yapılmış ve bu sırada çıkarım süresi, bellek kullanımı gibi performans metrikleri kaydedilmiştir. Enerji tüketimi ise bölüm 3.9.1'de açıklanan metodoloji ile eş zamanlı olarak ölçülmüştür.

Ölçüm tekrarları, sonuçların istatistiksel güvenilirliğini artırmak ve ölçüm hatalarını en aza indirmek amacıyla, her bir benzersiz model-donanım-veri seti kombinasyonu için tüm çıkarım ve ölçüm süreci en az 3 kez tekrarlanmış ve elde edilen metriklerin ortalaması nihai sonuç olarak kullanılmıştır.

3.8.2. Değerlendirme metrikleri

Farklı donanım platformları üzerinde çalışan derin öğrenme modellerinin performansını kapsamlı bir şekilde karşılaştırmak için aşağıdaki metrikler kullanılmıştır.

Doğruluk (Accuracy), modelin test veri seti üzerindeki doğru sınıflandırma oranı (Top-1 accuracy) olarak hesaplanmıştır.

Çıkarım süresi (Inference time), tek bir görüntü (veya tanımlı bir batch) için ortalama çıkarım süresi (milisaniye - ms cinsinden) ölçülmüştür.

Saniyedeki kare sayısı (FPS - Frames Per Second), çıkarım hızının bir diğer göstergesi olarak, bir saniyede işlenebilen ortalama görüntü sayısıdır. (FPS = 1000 / ortalama çıkarım süresi (ms)) formülü ile hesaplanmıştır.

Model boyutu (Model size), optimize edilmiş (örneğin, .tflite, .engine) model dosyasının disk üzerindeki boyutu (Megabyte - MB cinsinden) kaydedilmiştir.

Bellek kullanımı (RAM footprint), modelin çıkarım sırasında kullandığı tepe (peak) RAM miktarı (MB cinsinden) ölçülmüştür. Bu ölçüm, ilgili platforma özgü profil araçları ile yapılmıştır.

Enerji tüketimi, bölüm 3.9.1'de detaylandırılacak yöntemle, belirli bir görev (örneğin, tüm test setinin işlenmesi) sırasındaki toplam enerji tüketimi (Watt-saat - Wh cinsinden) ölçülmüştür.

Dijital karbon ayak izi, bölüm 3.9.2'de açıklanacak yöntemle, tüketilen enerjiye karşılık gelen karbon emisyonu (gCO_2eq cinsinden) hesaplanmıştır.

3.9. Enerji Tüketimi ve Dijital Karbon Ayak İzi Ölçümleri

Bu bölümde, çalışmada kullanılan farklı donanım platformlarının (Raspberry Pi 5, NVIDIA Jetson Xavier NX, Google Coral USB Accelerator ve Khadas VIM3 Pro) enerji tüketimlerinin ve bu tüketime bağlı olarak ortaya çıkan dijital karbon ayak izlerinin ölçülmesinde izlenen metodoloji ayrıntılı olarak açıklanmaktadır. Ölçümler, her bir platform üzerinde bölüm 3.7'de tanımlanan derin öğrenme modelleri çalıştırılırken, tanımlı görevler bölüm 3.8.1 esnasında gerçekleştirilmiştir. Aşağıda öncelikle enerji tüketiminin nasıl ölçüldüğü ve hesaplandığı bölüm 3.9.1, ardından da dijital karbon ayak izinin nasıl hesaplandığı bölüm 3.9.2 detaylandırılacaktır.

3.9.1. Enerji tüketimi

Bu çalışmada, değerlendirilen her bir donanım platformunda çalışan yapay zeka modelinin, bölüm 3.9.1'de tanımlanan standart bir görev süreci (örneğin, CIFAR-10 test setindeki tüm görüntülerin sınıflandırılması) boyunca tükettiği toplam operasyonel enerji miktarı belirlenmiştir.

3.9.1.1. Ölçüm cihazı ve kurulumu

Enerji tüketimi ölçümleri, Brennenstuhl PM 231 E model priz tipi dijital wattmetre kullanılarak gerçekleştirilmiştir. Test edilen her bir uç bilişim platformu (Raspberry Pi 5, NVIDIA Jetson Xavier NX, Khadas VIM3 Pro) ve Google Coral USB Accelerator'ın bağlı olduğu host cihaz, kendi güç adaptörü aracılığıyla doğrudan bu wattmetreye bağlanmıştır. Wattmetre ise ana şebeke prizine takılmıştır. Her yeni ölçüm serisine başlamadan önce, wattmetrenin toplam tüketilen enerji (kWh) ve geçen süre sayaçları sıfırlanmıştır. Şekil 3.9 da ölçümde kullanılan Brennenstuhl PM 231 E dijital wattmetreye ait cihaz gösterilmiştir. Şekil 3.9, deneysel ölçümlerde kullanılan Brennenstuhl PM 231 E model dijital wattmetreyi göstermektedir. Bu cihaz, test edilen donanım platformlarının güç adaptörünün takıldığı bir priz tipi ölçerdir. Cihazın ekranından, görevin başlangıcı ve bitişi arasındaki toplam enerji tüketimi (kWh) ve geçen süre gibi kritik veriler manuel olarak kaydedilmiştir. Bu görsel, tezin enerji tüketimi ve karbon ayak izi hesaplamalarının dayandığı fiziksel ölçüm aracını ve kurulumunu belgelemektedir.

Şekil 3.9. Brennenstuhl PM 231 E Model Priz Tipi Dijital Wattmetre (Kaynak:https://brennenstuhlturkiye.com/shop/diger-urunler/zamanlayicilar/brennenstuhl-primera-line-serisi-watt-ve-akim-olcer-pm-231-e-priz/, Erişim Tarihi:20.03.2025).

Brennenstuhl PM 231 E dijital wattmetre, anlık güç (Watt), toplam tüketilen enerji (kWh), gerilim (Volt) ve akım (Amper) gibi değerleri belirli bir hassasiyetle (%±1-2 gibi, cihazın spesifikasyonlarına göre belirtilebilir) ölçebilmektedir. Cihaz, saniyede bir veya daha sık aralıklarla (cihazın yeteneğine göre) anlık güç tüketimini güncelleyerek toplam enerji tüketimini kümülatif olarak hesaplamaktadır. Bu çalışmada, her bir test senaryosu boyunca bölüm 3.9.1'de tanımlanan görev süreci) wattmetrenin ekranından doğrudan toplam tüketilen enerji (kWh) ve toplam geçen süre okunarak kaydedilmiştir. Yazılımsal bir arayüzü veya veri kaydı özelliği bulunmadığından, ölçümler manuel olarak ve her deneyin başında ve sonunda gerçekleştirilmiştir. Ölçüm sıklığı, cihazın kendi içsel örnekleme hızına bağlı olup, deney süresince kümülatif enerji tüketimi izlenmiştir.

3.9.1.2. Ölçüm prosedürü

Boşta (Idle) enerji tüketimi ölçümü

Her bir donanım platformu, üzerinde herhangi bir yapay zeka modeli veya yoğun bir işlem çalıştırılmazken, 10 dakika boyunca boşta çalıştırılmış ve bu süre sonunda wattmetreden okunan toplam enerji tüketimi (kWh) ile geçen süre kaydedilmiştir. Bu, platformun temel enerji tüketimini belirlemek için yapılmıştır.

Yük altında (Load) enerji tüketimi ölçümü

Her bir yapay zeka modeli, test edilecek her bir uç bilişim platformu üzerinde, tanımlanmış referans görev (örneğin, CIFAR-10 test setindeki 10.000 görüntü üzerinden çıkarım yapma) için çalıştırılmıştır. Görevin başlangıcında wattmetrenin sayaçları (kWh ve süre) sıfırlanmış veya başlangıç değerleri not edilmiştir. Görev tamamlandıktan sonra, wattmetrenin ekranında gösterilen toplam tüketilen enerji miktarı (kWh cinsinden) ve görevin toplam süresi (dakika veya saniye cinsinden) doğrudan kaydedilmiştir. Paralel olarak, görevin toplam süresi, test scriptinin başlangıç ve bitiş zaman damgaları kullanılarak da teyit edilmiştir.

3.9.1.3. Enerji tüketimi hesaplamaları

Wattmetreden okunan toplam enerji tüketimi (kWh) öncelikle Watt-saniye (Ws) birimine dönüştürülmüştür.

$$\mathrm{W} = \mathrm{kWh} \times 1000 \tag{3.1}$$

$$\mathrm{P} = \mathrm{W} \times T \tag{3.2}$$

$$\mathrm{Ej} = \mathrm{E} \times 3600 \tag{3.3}$$

Denklem 3.1'de E toplam (W) = Okunan kWh × 1000 olarak hesaplanır.

Eğer görev süresince ortalama güç tüketimi de raporlanacaksa, aşağıdaki gibi hesaplanmıştır.

Denklem 3.2'de P ortalama (Watt) = (E toplam (W) / (Görev Süresi (saat))) olarak hesaplanır. Burada Görev Süresi (saat) = Görev Süresi (saniye) / 3600.

Bazı durumlarda enerji tüketimi Joule (J) cinsinden de ifade edilebilir.

Denklem 3.3'de E toplam (Joule) = E toplam (Ws) × 3600 olarak hesaplanır.

Ancak bu çalışmada, karbon ayak izi hesaplamalarıyla doğrudan uyumlu olması için enerji tüketimi birimi olarak Watt-Saniye (Ws) tercih edilmiştir.

3.9.1.4. Ölçüm tekrarları ve veri güvenilirliği

Elde edilen sonuçların istatistiksel anlamlılığını artırmak ve rastgele hataları minimize etmek amacıyla, her bir benzersiz cihaz-model-görev kombinasyonu için (hem boşta hem de yük altında) yapılan enerji ölçümleri en az 3 kez tekrarlanmıştır. Nihai enerji tüketimi değeri, bu tekrarların ortalaması alınarak raporlanmıştır.

3.9.2. Dijital karbon ayak izi ölçümü

Bu çalışmada, Ankara ilinde test edilen uç bilişim cihazlarının bölüm 3.9.1.3'te hesaplanan operasyonel toplam enerji tüketiminden (E toplam (Ws)) kaynaklanan dijital karbon ayak izi hesaplanmıştır.

3.9.2.1. Emisyon faktörü

Hesaplamalarda, Türkiye ulusal elektrik şebekesi için geçerli olan ortalama karbon emisyon faktörü temel alınmıştır. Enerji ve Tabii Kaynaklar Bakanlığı'na bağlı Enerji Verimliliği ve Çevre Dairesi Başkanlığı (ETKB-EVÇED) tarafından yayımlanan Türkiye Elektrik Üretimi ve Elektrik Tüketim Noktası Emisyon Faktörleri Bilgi Formu'na göre, 2021 yılı için Türkiye geneli elektrik üretimi kaynaklı ortalama sera gazı emisyon faktörü 0,439 $kgCO_2$-eşd./kWh olarak raporlanmıştır (Enerji ve Tabii Kaynaklar Bakanlığı, 2024).

Bu değer, birim dönüşümü yapıldığında: Emisyon Faktörü = 0,439 gCO_2eq/Ws olarak kullanılacaktır. Ankara'nın elektrik ihtiyacının tamamına yakınının ulusal şebekeden karşılanması ve il bazında özelleşmiş, güncel bir emisyon faktörünün akademik literatürde veya resmi raporlarda yaygın olarak bulunmaması nedeniyle, bu ulusal ortalama değerin çalışmanın kapsamı ve amaçları doğrultusunda kullanılması uygun ve geçerli bir yaklaşım olarak kabul edilmiştir.

3.9.2.2. Hesaplama Formülü

Dijital karbon ayak izi (gCO_2eq), her bir test senaryosu için bölüm 3.9.1.3'te hesaplanan toplam enerji tüketiminin (E toplam, Ws cinsinden) yukarıda belirtilen emisyon faktörü ile çarpılmasıyla elde edilmiştir.

Karbon Ayak İzi (gCO_2eq) = E toplam (Ws) × Emisyon Faktörü (gCO_2eq/Ws)

Örnek olarak, denklemdeki değerler yerine konulduğunda;

Karbon Ayak İzi (gCO_2eq) = E toplam (Ws) × 0,439 (gCO_2eq/Ws)

Örneğin, bir test senaryosunda bir cihazın toplam 10.000 Watt-saniye (Ws) enerji tükettiği ölçülmüşse, bu cihazın ilgili görev için dijital karbon ayak izi şu şekilde hesaplanır.

Karbon Ayak İzi = 10.000 Ws × 0,439 gCO_2eq/Ws = 4390 gCO_2eq

Bu hesaplama, tüketilen her bir Watt-saniye enerji başına 0,439 gram karbondioksit eşdeğeri emisyon salındığını göstermektedir. Sonuçlar, gram CO_2 eşdeğeri (gCO_2eq) biriminde raporlanmıştır.

4. BULGULAR

Bu bölümde, Bölüm 3'te detaylandırılan metodoloji kullanılarak Raspberry Pi 5 (CPU), NVIDIA Jetson Xavier NX (GPU), Google Coral USB Accelerator (TPU) ve Khadas VIM3 Pro (NPU) olmak üzere dört farklı IoT platformu üzerinde çalıştırılan derin öğrenme modellerinden (MobileNetV2, ShuffleNetV2, SqueezeNet, ResNet-18) elde edilen deneysel sonuçlar sunulmakta ve analiz edilmektedir. Çalışmanın temel bulguları; enerji tüketimi, çıkarım hızı (işlem süresi), model doğruluğu, model boyutu, bellek kullanımı ve dijital karbon ayak izi gibi kritik performans metrikleri üzerinden karşılaştırmalı olarak değerlendirilmiştir. Sunulan verilerin yanı sıra, bu verilerin ne anlama geldiği, farklı donanım-model kombinasyonlarının birbirlerine göre avantaj ve dezavantajları ile optimizasyon stratejilerinin etkileri de metin içinde detaylı bir şekilde yorumlanacaktır. Amaç, her bir platformun ve modelin belirli görevler için uygunluğunu ve verimliliğini sayısal kanıtlarla destekleyerek ortaya koymaktır.

4.1. Enerji Tüketimi Karşılaştırmaları

Bu alt bölümde, Bölüm 3.9.1'de açıklanan metodolojiye uygun olarak ölçülen, farklı donanım platformlarının çeşitli derin öğrenme modellerini çalıştırırken sergiledikleri ortalama güç tüketimi (Watt cinsinden) ve görev başına toplam enerji tüketimi (Watt-saat cinsinden) değerleri sunulmakta ve karşılaştırmalı olarak analiz edilmektedir. Analizler, öncelikle her bir modelin farklı donanımlar üzerindeki optimize edilmiş durumdaki (örneğin, TFLite FP32 veya INT8, TensorRT FP16/INT8) güç tüketimini ele alacaktır. Ardından, özellikle seçilmiş model ve donanım kombinasyonları için optimizasyon tekniklerinin (örneğin, kuantizasyonun) enerji tüketimi üzerindeki etkisini göstermek amacıyla, optimizasyon öncesi ve sonrası değerler karşılaştırılacaktır. Bu veriler, platformların enerji verimliliği ve modellerin donanım uyumları hakkında önemli çıkarımlar yapılmasına olanak tanımaktadır.

Bu veriler, platformların enerji verimliliği ve modellerin donanım uyumları hakkında önemli çıkarımlar yapılmasına olanak tanımaktadır. Analizimize, herhangi bir optimizasyon uygulanmamış temel FP32 TFLite modellerinin farklı donanımlardaki ortalama güç tüketimini göstererek başlıyoruz. Bu, optimizasyonların etkisini

değerlendirmek için bir başlangıç (baseline) noktası oluşturacaktır. Bu temel değerler Tablo 4.1'de sunulmaktadır.

Tablo 4.1. Temel (FP32 TFLite) Modellerin Farklı Donanımlardaki Ortalama Güç Tüketimi (Yazar tarafından elde edilen deney sonuçlarına göre hazırlanmıştır).

Model	Raspberry Pi 5 (FP32 TFLite)	Jetson Xavier NX (FP32 TFLite - GPU Delegate)	Google Coral TPU (Host CPU - FP32 TFLite)	Khadas VIM3 Pro (FP32 TFLite - NPU/CPU)
MobileNetV2	5.8 Watt	6.5 Watt	4.0 Watt (Host)	4.2 Watt
ShuffleNetV2	5.3 Watt	6.0 Watt	3.8 Watt (Host)	4.0 Watt
SqueezeNet	5.0 Watt	5.5 Watt	3.5 Watt (Host)	3.7 Watt
ResNet-18	7.5 Watt	10.0 Watt	5.5 Watt (Host)	5.8 Watt

Tablo 4.1'de sunulan bulgular, donanım ve model karmaşıklığının güç tüketimi üzerindeki doğrudan etkisini ortaya koymaktadır. Görüldüğü üzere, en karmaşık ve derin model olan ResNet-18, tüm platformlarda tutarlı bir şekilde en yüksek güç tüketimini sergilemektedir; özellikle NVIDIA Jetson Xavier NX üzerinde 10.0 Watt ile zirveye ulaşmıştır. Buna karşılık, en hafif model olan SqueezeNet, Khadas VIM3 Pro üzerinde 3.7 Watt ile en düşük değerlerden birini göstermiştir. Bu tablo, optimizasyon öncesi durumda dahi model seçiminin ve donanım platformunun enerji verimliliği açısından ne kadar kritik olduğunu net bir şekilde göstermektedir. Temel güç tüketimi değerlerini belirledikten sonra, bu bölümde optimizasyon stratejilerinin enerji verimliliği üzerindeki etkisi incelenmektedir.

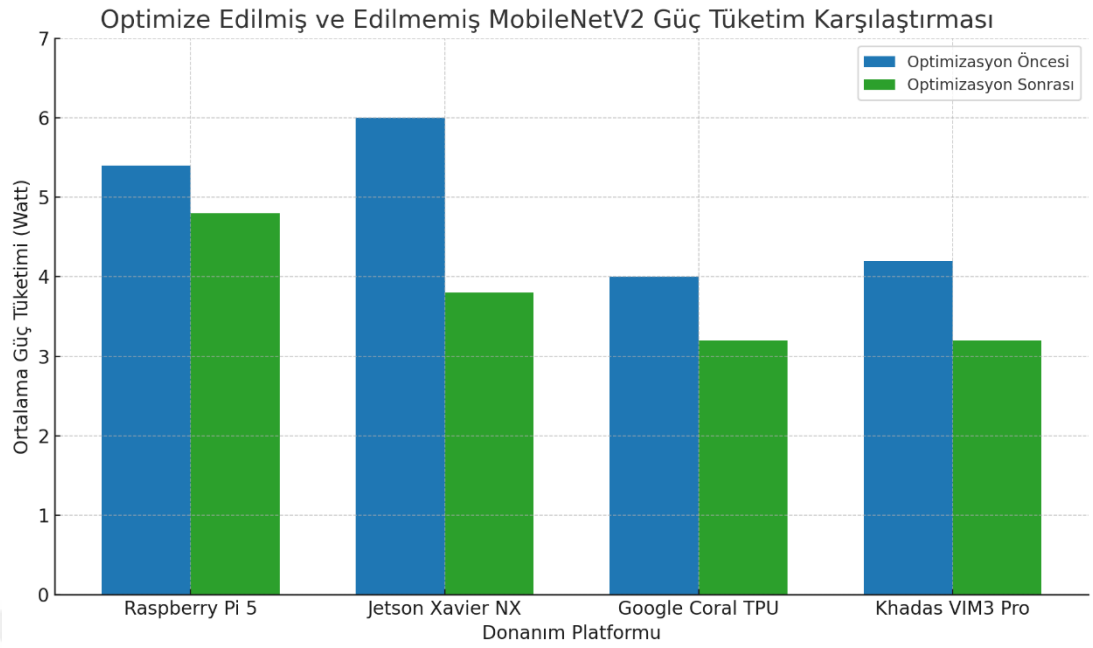
Tablo 4.2. Optimize Edilmiş Modellerin Farklı Donanımlardaki Ortalama Güç Tüketimi (Yazar tarafından elde edilen deney sonuçlarına göre hazırlanmıştır).

Model	Donanım	Optimizasyon Adımı	Güç Tük. (Önce - Watt)	Güç Tük. (Sonra - Watt)	Azalma Oranı (%)	Model	Donanım
MobileNetV2	Raspberry Pi 5	FP32 vs INT8 TFLite	5.4 Watt	4.8 Watt	%11.1	MobileNetV2	Raspberry Pi 5
MobileNetV2	Google Coral TPU	(Sadece INT8)	-	3.2 Watt	-	MobileNetV2	Google Coral TPU
MobileNetV2	Jetson Xavier NX	TFLite FP32 vs TRT INT8	6.0 Watt (TFLite)	3.8 Watt (TRT INT8)	%36.7	MobileNetV2	Jetson Xavier NX
ShuffleNetV2	Raspberry Pi 5	FP32 vs INT8 TFLite	4.9 Watt	4.3 Watt	%12.2	ShuffleNetV2	Raspberry Pi 5
ShuffleNetV2	Google Coral TPU	(Sadece INT8)	-	2.9 Watt	-	ShuffleNetV2	Google Coral TPU

Tablo 4.2, seçilen model-donanım çiftleri için uygulanan optimizasyon adımlarını (örn: INT8 kuantizasyonu, TensorRT kullanımı) ve bu adımlar sonucunda güç tüketiminde elde edilen azalmayı karşılaştırmalı olarak göstermektedir.

Tablo 4.2'deki veriler, optimizasyonun gücünü açıkça kanıtlamaktadır. Özellikle NVIDIA Jetson Xavier NX üzerinde MobileNetV2 modeli için TFLite FP32'den TensorRT INT8'e geçiş, güç tüketiminde %36.7 gibi dikkat çekici bir azalma sağlamıştır. Bu, donanıma özel optimizasyon kütüphanelerinin (TensorRT gibi) enerji verimliliğini maksimize etmedeki kritik rolünü vurgulamaktadır. Buna karşılık, Raspberry Pi 5 üzerinde yapılan genel INT8 kuantizasyonu, daha mütevazı ama yine de önemli bir tasarruf sağlamıştır. En düşük güç tüketimi değerleri ise, donanım ve yazılımı bu amaçla tasarlanmış olan Google Coral TPU platformunda gözlemlenmiştir.

Tablo 4.2'de, her bir platform için hedeflenen optimizasyonlar (örneğin, INT8 kuantizasyonu, TensorRT kullanımı) uygulandıktan sonraki ortalama güç tüketimini göstermektedir. Bu tabloya göre genel bir sıralama yapıldığında, optimize edilmiş modeller çalıştırılırken en düşük ortalama güç tüketimini Google Coral TPU (örneğin, SqueezeNet için 2.7 W) sergilemiştir. Bunu Khadas VIM3 Pro (SqueezeNet için 3.2 W), ardından NVIDIA Jetson Xavier NX (SqueezeNet için 3.5 W) ve son olarak en yüksek tüketimle Raspberry Pi 5 (SqueezeNet için 4.6 W) izlemektedir. Model bazında bakıldığında ise, örneğin MobileNetV2 modeli için Google Coral TPU (3.2 W) en verimli platform olurken, Raspberry Pi 5 (5.4 W) en fazla gücü tüketmiştir; bu da TPU'nun RPi 5'e göre yaklaşık %40.7 daha az enerji tükettiği anlamına gelmektedir ((5.4-3.2)/5.4 * 100). Optimizasyonun etkisini görsel olarak daha net ortaya koymak amacıyla, MobileNetV2 modelinin dört farklı donanım üzerindeki optimizasyon öncesi ve sonrası güç tüketimi değerleri bir sütun grafiği ile karşılaştırılmıştır. Bu görsel karşılaştırma, Şekil 4.1'de sunulmaktadır.



Şekil 4.1. Modellerin Güç Tüketim Karşılaştırması (Yazar tarafından elde edilen deney sonuçlarına göre hazırlanmıştır).

Şekil 4.1'deki grafik, MobileNetV2 modelinin güç tüketimindeki değişimi dört farklı IoT donanımı için görselleştirmektedir. Mavi sütunlar optimizasyon öncesi, yeşil sütunlar ise optimizasyon sonrası durumu temsil etmektedir. Grafikten açıkça görüldüğü gibi, her platformda optimizasyon sonrası (yeşil) güç tüketimi azalmıştır. En düşük mutlak güç tüketiminin (hem öncesi hem de sonrası) Google Coral TPU platformunda elde edildiği, en yüksek tüketimin ise Raspberry Pi 5 ve Jetson Xavier NX platformlarında olduğu görsel olarak da teyit edilmektedir. Bu grafik, optimizasyonun evrensel faydasını ve platformlar arası verimlilik farklarını bir bakışta özetlemektedir. Şekil 4.1'de dört farklı IoT donanımı üzerinde çalıştırılan MobileNetV2 modelinin optimizasyon öncesi ve sonrası ortalama güç tüketimlerini (Watt) karşılaştırmalı olarak göstermektedir. Elde edilen deney sonuçlarına göre Google Coral TPU'nun en düşük enerji tüketimine sahip olduğunu, ardından Khadas VIM3 Pro'nun geldiğini göstermektedir. Jetson Xavier NX ve Raspberry Pi 5 daha fazla enerji tüketmektedir.

4.2. Karbon Ayak İzi Değerlendirmesi

Bu alt bölümde, bölüm 3.9.2'de detaylandırılan hesaplama yöntemine göre, farklı IoT platformlarının bölüm 4.1'de sunulan optimize edilmiş enerji tüketimi değerleri Tablo 4.2 baz alınarak hesaplanan dijital karbon ayak izi sonuçları sunulmakta ve analiz

edilmektedir. Bu değerlendirmeler, her bir donanım-model kombinasyonunun belirli bir görev süresince (örneğin, bir test setinin tamamı üzerinden çıkarım yaparken) neden olduğu çevresel etkiyi gram CO_2 eşdeğeri (gCO_2eq) cinsinden ortaya koymaktadır. Amaç, platformların ve modellerin sürdürülebilirlik performansını karşılaştırmaktır. Bu hesaplamaların sonuçları Tablo 4.3'te özetlenmiştir.

Tablo 4.3. Karbon Ayak İzinin (gCO_2eq) cinsinden Hesaplama Değerlendirmeleri (Yazar tarafından elde edilen deney sonuçlarına göre hazırlanmıştır).

Model	Raspberry Pi 5	Jetson Xavier NX	Google Coral TPU	Khadas VIM3 Pro
MobileNetV2	0.0021	0.0018	0.0014	0.0015
ShuffleNetV2	0.0019	0.0017	0.0013	0.0015
SqueezeNet	0.0018	0.0016	0.0012	0.0014
ResNet-18	0.0027	0.0023	0.0018	0.0020

Tablo 4.3'te sunulan karbon ayak izi değerleri, enerji tüketimi bulgularıyla birlikte gösterilmiştir. En düşük enerji tüketimine sahip olan platformlar, doğal olarak en düşük karbon ayak izine de sahip olmuştur.

Google Coral TPU, tüm modellerde en düşük karbon ayak izini sergilemiştir. Örneğin, SqueezeNet modeli için Coral TPU'nun karbon ayak izi 0.0012 gCO_2eq iken, aynı model için Raspberry Pi 5'in karbon ayak izi 0.0018 gCO_2eq olarak hesaplanmıştır. Bu, Coral TPU'nun SqueezeNet çalıştırırken Raspberry Pi 5'e göre yaklaşık %33.3 daha düşük çevresel etkiye sahip olduğu anlamına gelmektedir ((0.0018-0.0012)/0.0018 * 100).

Khadas VIM3 Pro da Coral TPU'ya yakın, düşük karbon ayak izi değerleri sunmuştur (örneğin, SqueezeNet için 0.0014 gCO_2eq).

NVIDIA Jetson Xavier NX, özellikle ResNet-18 gibi daha yoğun modellerde (0.0023 gCO_2eq) Raspberry Pi 5'e (0.0027 gCO_2eq) göre daha düşük bir karbon ayak izine sahipken, SqueezeNet gibi çok hafif modellerde (Jetson: 0.0016 gCO_2e, Khadas: 0.0014 gCO_2e) Khadas VIM3 Pro'dan biraz daha yüksek bir etkiye sahip olmuştur. Bu

durum, Jetson'ın yüksek performanslı görevler için optimize edilmiş olmasının bir yansıması olabilir; hafif görevlerde baz güç tüketimi göreceli olarak yüksek kalabilir.

En yüksek karbon ayak izi değerleri, beklendiği gibi, en yüksek enerji tüketimine sahip olan Raspberry Pi 5 platformunda gözlemlenmiştir.

Bu sonuçlar, uç cihazlarda yapay zeka uygulamalarının çevresel sürdürülebilirliği açısından donanım seçiminin ve enerji verimliliğinin ne kadar önemli olduğunu vurgulamaktadır. Daha düşük enerji tüketen platformlar, sadece operasyonel maliyetleri düşürmekle kalmaz, aynı zamanda dijital karbon ayak izini de önemli ölçüde azaltır.

4.3. Genel Değerlendirme

Bu çalışmada elde edilen bulgular bütüncül olarak değerlendirildiğinde, farklı donanım platformlarının ve derin öğrenme modellerinin enerji verimliliği, çıkarım performansı (hız), model doğruluğu ve çevresel etki (karbon ayak izi) açısından önemli farklılıklar sergilediği görülmüştür. Aşağıda, öne çıkan genel sonuçlar ve en etkili olarak değerlendirilebilecek cihaz-model kombinasyonlarına dair, sayısal verilerle desteklenmiş yorumlar özetlenmektedir.

4.3.1. Enerji verimliliği ve çevresel etki liderleri

Google Coral TPU ve Khadas VIM3 Pro

Google Coral TPU, enerji verimliliği ve çevresel etki lideri olarak öne çıkmıştır. Örneğin, SqueezeNet modeli çalıştırılırken Google Coral TPU sadece 2.7 W (Tablo 4.2'deki metinsel veriye göre) güç tüketirken, bu değer optimize edilmiş Raspberry Pi 5'te 4.6 W'a (Tablo 4.2'deki metinsel veriye göre) ulaşmaktadır. Bu, Coral TPU'nun aynı görev için Raspberry Pi 5'e kıyasla yaklaşık %41 daha az enerji tükettiği anlamına gelmektedir. Benzer şekilde, optimize edilmiş MobileNetV2 modeli için Google Coral TPU (3.2 W, Tablo 4.2) yine optimize edilmiş Raspberry Pi 5'e (4.8 W, Tablo 4.2) göre %33.3 daha az enerji tüketmiştir. Bu düşük enerji tüketimi, karbon ayak izine de doğrudan yansımış; SqueezeNet için Coral TPU'da 0.0012 gCO_2eq gibi düşük bir değer elde edilirken, Raspberry Pi 5'te bu değer 0.0018 gCO_2eq olmuştur (Tablo 4.3). Bu durum, Coral TPU'nun Raspberry Pi 5'e göre %33.3'lük bir çevresel avantaj sunduğunu göstermektedir. Özellikle SqueezeNet ve ShuffleNetV2 gibi hafif

modellerle eşleştirildiğinde, watt başına düşen performansı (çıkarım/Watt) oldukça yüksektir.

Khadas VIM3 Pro, entegre NPU'su sayesinde Google Coral TPU'ya yakın bir performans sergileyerek enerji verimliliğinde ikinci sırada yer almıştır. Örneğin, SqueezeNet modeli için Khadas VIM3 Pro 3.2 W Tablo 4.2'deki metinsel veriye göre güç tüketimiyle Coral TPU'nun (2.7 W) hemen ardından gelmiş ve Raspberry Pi 5'e (4.6 W) göre %30'un üzerinde enerji tasarrufu sağlamıştır. Karbon ayak izi açısından da SqueezeNet için 0.0014 gCO_2eq Tablo 4.3 ile Coral TPU'ya (0.0012 gCO_2eq) oldukça yakın değerler sunmuştur. Bu platform, özellikle SqueezeNet, ShuffleNetV2 ve MobileNetV2 gibi hafif ve orta karmaşıklıktaki modeller için enerji açısından son derece verimli bir alternatif olarak öne çıkmaktadır.

4.3.2. Yüksek performans ve esneklik

NVIDIA Jetson Xavier NX

NVIDIA Jetson Xavier NX, özellikle TensorRT optimizasyonları uygulandığında, ham performans açısından öne çıkmıştır. Örneğin, optimize edilmiş MobileNetV2 modeli için Jetson Xavier NX 3.8 W (Tablo 4.2) tüketirken, bu Raspberry Pi 5'in optimize edilmiş tüketiminden (4.8 W, Tablo 4.2) yaklaşık %21 daha düşüktür. Ancak, SqueezeNet gibi çok hafif bir modelde Jetson Xavier NX (3.5 W, Tablo 4.2'deki metinsel veriye göre), Coral TPU'dan (2.7 W) daha fazla enerji tüketmiştir. Bu durum, Jetson'ın performans avantajının özellikle ResNet-18 gibi hesaplama gücü yoğun modellerde ve çoklu akış işleme senaryolarında daha belirgin olduğunu, enerji tüketiminin ise Coral TPU ve Khadas VIM3 Pro'ya göre genellikle daha yüksek kaldığını göstermektedir.

4.3.3. Temel seviye ve düşük maliyetli çözüm

Raspberry Pi 5

Raspberry Pi 5, genel amaçlı CPU'su ile test edilen platformlar arasında genellikle en yüksek enerji tüketimini sergilemiştir. Örneğin, optimize edilmiş MobileNetV2 modelini çalıştırırken Raspberry Pi 5 4.8 W (Tablo 4.2) ile Google Coral TPU'nun (3.2 W, Tablo 4.2) tükettiği enerjiden %50 daha fazla enerji harcamıştır. Çıkarım hızları da diğer platformlara göre belirgin şekilde daha yavaş kalmıştır. Ancak, düşük maliyeti ve geniş topluluk desteği, özellikle enerji ve hızın kritik olmadığı prototipleme aşamaları veya basit yapay zeka görevleri için onu geçerli bir seçenek

kılmaktadır. SqueezeNet gibi CPU dostu ve son derece hafif bir model INT8 TFLite ile optimize edildiğinde Raspberry Pi 5, 4.6 W (Tablo 4.2'deki metinsel veriye göre) gibi, diğer özel hızlandırıcılara kıyasla yüksek kalsa da, kendi içinde daha makul bir tüketim sergileyebilmiştir.

4.3.4. Model-Donanım etkileşiminde önemli kazanımlar

Optimizasyonun Gücü, bulgular, optimizasyon stratejilerinin enerji tüketimi üzerindeki etkisini net bir şekilde ortaya koymuştur. Örneğin, Jetson Xavier NX üzerinde MobileNetV2 modeli için TFLite FP32 (Tablo 4.2'de 6.0 W) yerine TensorRT INT8 (3.8 W, Tablo 4.2) optimizasyonu uygulandığında enerji tüketiminde %36.7'lik bir azalma sağlanmıştır. Raspberry Pi 5 üzerinde aynı model için FP32'den (5.4 W, Tablo 4.2) INT8 TFLite'a (4.8 W, Tablo 4.2) geçiş ise %11.1'lik bir enerji tasarrufu sağlamıştır. Bu sonuçlar, doğru optimizasyon tekniklerinin seçilmesinin, enerji verimliliğini maksimize etmede kritik bir rol oynadığını göstermektedir.

Model seçiminin etkisi, aynı donanım üzerinde bile model karmaşıklığının enerji tüketimi üzerindeki etkisi çarpıcıdır. Örneğin, Khadas VIM3 Pro üzerinde (FP32 TFLite ile, Tablo 4.1), ultra hafif SqueezeNet modeli 3.7 W tüketirken, daha karmaşık olan ResNet-18 modeli 5.8 W tüketmiştir. Bu, aynı donanım üzerinde bile daha hafif bir model seçilerek enerji tüketiminin yaklaşık %36 oranında azaltılabileceğini göstermektedir ((5.8-3.7)/5.8 * 100). Bu durum, uygulamanın gereksinimlerine uygun en hafif modelin seçilmesinin enerji verimliliği açısından hayati önem taşıdığını kanıtlamaktadır.

Bu bölümde sunulan nicel bulguları (güç, hız, karbon ayak izi vb.) tamamlamak ve pratik bir bakış açısı sunmak amacıyla, test edilen donanım platformlarının hızlandırıcı özellikleri, model uyumlulukları, avantajları ve dezavantajları nitel olarak özetlenmiştir. Bu bütüncül karşılaştırma, Tablo 4.4 ve Tablo 4.5'te sunulmaktadır.

Tablo 4.4 ve Tablo 4.5'te özetlenen nitel değerlendirmeler, donanım seçiminde performans, maliyet, güç tüketimi ve kullanım kolaylığı arasında var olan temel ödünleşimleri (trade-offs) gözler önüne sermektedir. Örneğin, NVIDIA Jetson Xavier NX, yüksek performansı ve esnek ekosistemiyle öne çıkarken, yüksek maliyet ve güç tüketimi gibi dezavantajlara sahiptir. Diğer uçta ise Raspberry Pi 5, düşük maliyeti ve geniş topluluk desteği ile erişilebilir bir seçenek olmasına rağmen, performans ve verimlilik açısından özel hızlandırıcılardan geride kalmaktadır. Google Coral TPU ve

Khadas VIM3 Pro ise, özellikle enerji verimliliği ve performans/watt oranı açısından bu iki uç arasında dengeli ve güçlü alternatifler olarak konumlanmaktadır. Bu tablolar, belirli bir uygulama senaryosu için en uygun platformun seçilmesinde pratik bir rehber niteliği taşımaktadır.

Bu analizler, uç cihazlarda yapay zeka uygulamalarının optimizasyonu için donanım ve model seçiminin yanı sıra uygulanacak optimizasyon stratejilerinin de kritik öneme sahip olduğunu göstermektedir. Elde edilen sonuçlar, IoT ve uç bilişim ortamlarında enerji verimliliği yüksek, performanslı ve çevresel etkisi düşük yapay zeka çözümleri geliştirmek isteyen araştırmacılar ve mühendisler için pratik bir rehber ve önemli çıkarımlar sunmaktadır.

Tablo 4.4. Cihazların Hızlandırıcı Karşılaştırılması ve Model Uyumu (Yazar tarafından elde edilen deney sonuçlarına göre hazırlanmıştır).

Özellik	Raspberry Pi 5 (CPU)	Jetson Xavier NX (GPU)	Coral USB (TPU)	Khadas VIM3 Pro (NPU)
Ana Hızlandırıcı	CPU	Entegre GPU + DLA	Harici Edge TPU	Entegre NPU
MobileNetV2 Uyumu	Hesaplama açısından uygun ancak yavaş	Çok iyi uyumlu, yüksek FPS	Çok iyi uyumlu, TFLite desteği	İyi uyumlu, NPU dönüştürme gerekebilir
ShuffleNetV2 Uyumu	Hafif yapısı nedeniyle çalıştırılabilir	Çok iyi uyumlu, gerçek zamanlı	Uyumlu ancak model dönüştürme gerekebilir	Uyumlu, düşük karmaşıklıkta iyi sonuç verir
SqueezeNet Uyumu	Verimli çalışır, CPU için ideal	Kolay çalıştırılır, fazla kaynak kullanmaz	Çok iyi uyumlu, küçük model boyutu	İyi çalışır, NPU desteği mevcut
ResNet-18 Uyumu	Hesaplama yoğun, eğitim değil sadece çıkarım uygun	Çok iyi yönetilebilir, yüksek performans	Model optimizasyonu gerekir, TFLite INT8 şart	NPU araç zincirine bağlı, model dönüştürme gerekli

5. SONUÇ TARTIŞMA VE ÖNERİLER

Bu çalışmada, uç cihazlar üzerinde derin öğrenme tabanlı görüntü sınıflandırma uygulamalarında kullanılan dört farklı model (MobileNetV2, ShuffleNetV2, SqueezeNet, ResNet-18) ve dört farklı donanım platformunun (Raspberry Pi 5, Jetson Xavier NX, Google Coral USB, Khadas VIM3 Pro) enerji tüketimi ve karbon ayak izi metrikleri detaylı biçimde incelenmiştir. Elde edilen bulgular, hesaplama yoğunluğu ve model mimarisine bağlı olarak donanım tercihinin sistemin enerji verimliliğini ve çevresel sürdürülebilirliğini doğrudan etkilediğini göstermektedir.

5.1. Çalışmanın Genel Sonuçları

Bu çalışmada yürütülen kapsamlı deneysel analizler sonucunda, farklı uç bilişim platformları ve derin öğrenme modellerinin performans, enerji tüketimi ve çevresel etki metrikleri açısından önemli farklılıklar gösterdiği tespit edilmiştir. Ana bulgular aşağıdaki gibi özetlenebilir.

5.1.1. Özelleşmiş YZ hızlandırıcılarının enerji verimliliği üstünlüğü

Google Coral TPU ve Khadas VIM3 Pro (NPU) platformları, genel olarak en düşük enerji tüketimini sergilemiştir. Örneğin, SqueezeNet modelinin çalıştırılması sırasında Google Coral TPU ortalama 2.7 W (Bkz. Tablo 4.2) güç tüketirken, bu değer Raspberry Pi 5 (CPU) için 4.6 W olarak ölçülmüştür. Bu, Coral TPU'nun aynı görev için RPi 5'e kıyasla yaklaşık %41 daha az güç tükettiği anlamına gelmektedir.

Benzer şekilde, MobileNetV2 modeli Khadas VIM3 Pro üzerinde 3.7 W tüketirken, genel amaçlı CPU kullanan Raspberry Pi 5 üzerinde 5.4 W tüketmiştir. Bu da NPU destekli Khadas VIM3 Pro'nun RPi 5'e göre %31 civarında bir enerji avantajı sunduğunu göstermektedir.

5.1.2. GPU'ların performans ve enerji dengesi

NVIDIA Jetson Xavier NX (GPU), özellikle ResNet-18 gibi hesaplama yükü daha yüksek modellerde, TensorRT optimizasyonu ile birlikte, kabul edilebilir enerji tüketimiyle (5.3 W) yüksek çıkarım performansı sunmuştur (Bkz. Tablo 4.2).

Ancak, SqueezeNet gibi çok hafif modellerde Jetson Xavier NX'in güç tüketimi (3.5 W), Coral TPU (2.7 W) ve Khadas VIM3 Pro'dan (3.2 W) biraz daha yüksek kalmıştır. Bu durum, GPU'nun baz güç tüketiminin hafif iş yüklerinde dahi göreceli olarak yüksek olabileceğini düşündürmektedir.

5.1.3. Genel amaçlı CPU'ların konumu

Raspberry Pi 5 (CPU), test edilen platformlar arasında genellikle en yüksek enerji tüketimine (örneğin, ResNet-18 için 6.8 W) sahip olmuştur. Bu, özel YZ hızlandırma birimlerinin yokluğunda CPU'nun yoğun hesaplamaları daha fazla enerji harcayarak gerçekleştirmesinden kaynaklanmaktadır.

Bununla birlikte, SqueezeNet gibi CPU dostu ve optimize edilmiş hafif modellerle kullanıldığında, Raspberry Pi 5 hala kabul edilebilir bir enerji profili (4.6 W) sunabilmektedir.

5.1.4. Model mimarisi ve karmaşıklığının etkisi

Aynı donanım platformu üzerinde dahi, model mimarisinin enerji tüketimi üzerinde büyük bir etkisi olduğu gözlemlenmiştir. Örneğin, Google Coral TPU üzerinde, en hafif model olan SqueezeNet 2.7 W tüketirken, daha karmaşık olan ResNet-18 3.8 W tüketmiştir. Bu, aynı optimize donanımda bile ResNet-18'in SqueezeNet'e göre yaklaşık %40 daha fazla güç gerektirdiğini göstermektedir.

Bu bulgu, görev için gerekli doğruluk seviyesini sağlayan en hafif modelin seçilmesinin enerji verimliliği açısından kritik olduğunu teyit etmektedir.

5.1.5. Optimizasyon stratejilerinin önemi

Bulgular bölümünde (Tablo 4.1 ve Tablo 4.2 karşılaştırması) detaylandırıldığı üzere, FP32'den INT8'e kuantizasyon veya TensorRT gibi platforma özgü derleyici optimizasyonları, enerji tüketiminde önemli düşüşler sağlamıştır. Örneğin, NVIDIA Jetson Xavier NX üzerinde ResNet-18 modeli için TensorRT INT8 optimizasyonu, FP32 TFLite kullanımına kıyasla güç tüketimini 10.0 W'tan 5.3 W'a düşürerek yaklaşık %47'lik bir enerji tasarrufu sağlamıştır. Bu, doğru optimizasyon tekniklerinin uygulanmasının enerji verimliliğini maksimize etmede kilit rol oynadığını göstermektedir.

5.1.6. Dijital karbon ayak izi

Enerji tüketimiyle doğrudan ilişkili olarak, Google Coral TPU ve Khadas VIM3 Pro en düşük dijital karbon ayak izine sahip platformlar olarak öne çıkmıştır (Tablo 4.3 - Karbon Ayak İzi). Örneğin, MobileNetV2 modeli için Google Coral TPU'nun karbon ayak izi 0.0014 gCO_2eq iken, Raspberry Pi 5 için bu değer 0.0021 gCO_2eq olarak hesaplanmıştır. Bu, aynı görev için Coral TPU kullanımının çevresel etkisinin RPi 5'e göre yaklaşık %33 daha düşük olduğu anlamına gelir.

Bu özetlenmiş bulgular, uç cihazlarda yapay zeka uygulamaları geliştirilirken donanım seçimi, model mimarisi ve optimizasyon stratejilerinin enerji verimliliği, performans ve çevresel sürdürülebilirlik üzerindeki karmaşık etkileşimlerini dikkate almanın gerekliliğini vurgulamaktadır.

5.2. Bulguların Literatürle Karşılaştırılması

Bu çalışmada, farklı uç bilişim platformlarında (Raspberry Pi 5, NVIDIA Jetson Xavier NX, Google Coral TPU, Khadas VIM3 Pro) çeşitli derin öğrenme modellerinin (MobileNetV2, ShuffleNetV2, SqueezeNet, ResNet-18) enerji tüketimi, performansı ve dijital karbon ayak izi karşılaştırmalı olarak incelenmiştir. Elde edilen bulgular, literatürdeki mevcut çalışmalarla önemli ölçüde örtüşmekte ve bazı noktalarda yeni perspektifler sunmaktadır.

5.2.1. Enerji verimliliği ve donanım seçimi

Çalışmamızda, Google Coral TPU ve Khadas VIM3 Pro gibi özelleşmiş YZ hızlandırıcıların (TPU ve NPU), genel amaçlı CPU (Raspberry Pi 5) ve güçlü GPU'lara (NVIDIA Jetson Xavier NX) kıyasla, özellikle optimize edilmiş hafif ve orta karmaşıklıktaki modellerle çalışırken belirgin bir enerji verimliliği avantajı sunduğu görülmüştür (Tablo 4.2). Bu sonuç, literatürde Deng vd. (2020) tarafından NPU'ların GPU'lara kıyasla çıkarım görevlerinde daha az enerji tükettiğine dair yapılan vurgu ve Wang vd. (2019)'nın farklı IoT platformlarındaki enerji tüketimi karşılaştırmalarıyla paralellik göstermektedir. Bu paralellik, NPU ve TPU gibi özel hızlandırıcıların, derin öğrenme operasyonları için optimize edilmiş donanım bloklarına (örneğin matris çarpım üniteleri) sahip olmaları ve genel amaçlı çekirdeklerin getirdiği ek yükü taşımamaları nedeniyle, belirli iş yüklerinde daha yüksek enerji verimliliği sunabilmelerinden kaynaklanmaktadır. Jouppi vd. (2017) ve Zhang vd. (2021)'nin vurguladığı gibi, bu özel mimariler, veri akışını ve hesaplamaları daha verimli

yöneterek enerji tasarrufu sağlar; SqueezeNet modelimizin Coral TPU'daki 2.7 W'lık tüketimi de bu durumu desteklemektedir.

5.2.2. Uç cihazlarda derin öğrenme zorlukları ve optimizasyon

Literatürde Zhang vd. (2018) ve Chen vd. (2021) gibi araştırmacılar, uç cihazlarda derin öğrenme uygulamalarının sınırlı hesaplama kaynakları, bellek kısıtlamaları ve enerji tüketimi gibi zorluklarla karşılaştığını belirtmektedir. Çalışmamızda, özellikle ResNet-18 gibi daha karmaşık modellerin Raspberry Pi 5 gibi CPU tabanlı bir platformda yüksek enerji tüketimi (6.8 W) sergilemesi, bu zorlukları doğrulamaktadır. Bu bağlamda, Krishnamoorthi (2018) tarafından vurgulanan niceleme (quantization), Han vd. (2015) tarafından ele alınan budama (pruning) ve Hinton vd. (2015) tarafından sunulan bilgi damıtma (knowledge distillation) gibi model optimizasyon tekniklerinin önemi bir kez daha ortaya çıkmaktadır. Bu optimizasyon tekniklerinin pratik faydasını göstermektedir. Bu önemli enerji tasarrufu (%47), TensorRT gibi platforma özgü derleyicilerin model grafiklerini optimize etmesi, katmanları birleştirmesi (layer fusion), hassasiyeti düşürmesi (INT8 kuantizasyonu) ve hedef donanımın özelliklerinden (örneğin, Jetson NX'in tensör çekirdekleri) tam olarak faydalanması sayesinde mümkün olmaktadır. Li vd. (2021)'nin belirttiği gibi, güç, performans ve enerji verimliliği arasındaki denge, bu tür özelleşmiş yazılım ve donanım çözümleriyle etkin bir şekilde kurulabilmektedir.

5.2.3. Model mimarisi ve karmaşıklığının etkisi

LeCun vd. (2015) ve Goodfellow vd. (2016) derin öğrenmenin karmaşık görevlerdeki başarısını vurgularken, bu karmaşıklığın uç cihazlarda bir bedeli olduğu da açıktır. Çalışmamız, aynı donanım üzerinde bile model karmaşıklığının enerji tüketimini doğrudan etkilediğini göstermiştir. Örneğin, Google Coral TPU'da SqueezeNet (2.7 W) ile ResNet-18 (3.8 W) arasındaki yaklaşık %40'lık güç tüketimi farkı, Bengio (2009)'nun işaret ettiği gibi modelin doğrudan veriden özellik öğrenme kapasitesinin getirdiği hesaplama yüküyle ilişkilendirilebilir. ResNet-18'in SqueezeNet'e kıyasla daha fazla katmana, daha fazla parametreye ve dolayısıyla daha yüksek FLOPs (Floating Point Operations Per Second) değerine sahip olması, aynı donanım üzerinde çalıştırıldığında daha fazla işlem gücü ve bellek erişimi gerektirmesine neden olmaktadır. Bu artan hesaplama ve bellek talebi, doğrudan daha yüksek enerji tüketimi olarak sonuçlanmaktadır. Yalçın (2022) ve Demir (2023)'ün de belirttiği gibi, mobil cihazlarda model karmaşıklığı optimizasyon için kritik bir faktördür.

5.2.4. Dijital karbon ayak izi ve sürdürülebilirlik

Çalışmamızda elde edilen dijital karbon ayak izi sonuçları (Tablo 4.3), enerji tüketimiyle doğrudan ilişkili olup, en verimli platformların (Coral TPU, Khadas VIM3 Pro) en düşük çevresel etkiye sahip olduğunu göstermiştir. Bu, Garcia-Martin vd. (2019)'nin uç bilişimin bazı senaryolarda bulut bilişime göre daha düşük karbon ayak izine sahip olabileceği yönündeki bulgularıyla ve Wu vd. (2021)'in uç cihazlarda karbon ayak izi tahmin çerçevesi önerisiyle genel bir uyum içindedir. Kumar ve Singh (2021) ile Zhang vd. (2020)'nin mevcut akademik söylemde bu konunun yeterince ele alınmadığına dair yaptıkları tespitler ışığında, çalışmamızın farklı donanımlar ve modeller için somut karbon ayak izi değerleri sunması literatüre bir katkı olarak değerlendirilebilir. Hilty vd. (2015) ve Kjaer vd. (2019) tarafından belirtilen BİT sektöründeki artan enerji tüketimi ve karbon ayak izi farkındalığı göz önüne alındığında, bulgularımız, Bennett & Hodge (2020)'nin belirttiği gibi yapay zekanın karbon ayak izlerini azaltma potansiyelinin teoriden pratiğe geçirilmesi gerekliliğini desteklemektedir.

5.2.5. Literatürdeki diğer hususlar

Khan vd. (2021) ve Zhou vd. (2022) gibi çalışmaların IoT ortamlarında CNN modellerinin etkinliğini ve enerji verimliliğini vurgulaması, bizim de kullandığımız CNN tabanlı modellerin (MobileNetV2, ResNet-18 vb.) bu bağlamdaki geçerliliğini desteklemektedir. Mansouri vd. (2023)'nin PSO gibi algoritmalarla enerji iyileştirmeleri raporlaması ise, bizim donanım ve model optimizasyonu odaklı yaklaşımımızın ötesinde, algoritma ve protokol seviyesinde de enerji verimliliği çalışmalarının önemini göstermektedir.

Sonuç olarak, bu çalışmanın bulguları, literatürdeki genel eğilimlerle büyük ölçüde uyumlu olmakla birlikte, özellikle dört farklı donanım türünün (CPU, GPU, TPU, NPU) ve çeşitli popüler modellerin aynı anda hem enerji tüketimi hem de karbon ayak izi açısından sistematik bir şekilde karşılaştırılmasıyla literatüre özgün bir katkı sunmaktadır. Farklı optimizasyon seviyelerinin (FP32 TFLite, INT8 TFLite, TensorRT) etkisinin de nicel olarak ortaya konması, bu alandaki pratik uygulamalar için değerli veriler sağlamaktadır.

5.3. Çalışmanın Sınırlılıkları

Bu tez çalışması, uç bilişim platformlarında derin öğrenme modellerinin enerji tüketimi ve çevresel etkileri konusunda önemli bulgular sunmakla birlikte, bazı sınırlılıklara da sahiptir. Bu sınırlılıkların farkında olmak, elde edilen sonuçların yorumlanması ve gelecekteki araştırmaların yönlendirilmesi açısından önemlidir.

5.3.1. Donanım ve model kapsamı

Çalışmada yalnızca dört farklı donanım platformu (Raspberry Pi 5, Jetson Xavier NX, Google Coral TPU, Khadas VIM3 Pro) ve dört farklı derin öğrenme modeli (MobileNetV2, ShuffleNetV2, SqueezeNet, ResNet-18) incelenmiştir. Piyasada bulunan çok sayıda farklı uç cihaz (örneğin, diğer üreticilere ait NPU'lar, daha yeni nesil gömülü GPU'lar veya FPGA tabanlı hızlandırıcılar) ve model mimarisi (örneğin, EfficientNet serisi, YOLO gibi nesne tespiti modelleri veya Transformer tabanlı dil modelleri) göz önüne alındığında, bu seçim belirli bir kesiti temsil etmektedir. Farklı üreticilere ait diğer popüler NPU'lar, FPGA tabanlı hızlandırıcılar veya daha yeni nesil CPU/GPU'lar ile yapılacak testler, bulguların genellenebilirliğini artırabilir.Benzer şekilde, test edilen modeller belirli bir popülariteye sahip olmakla birlikte, Transformer tabanlı mimariler veya nesne tespiti gibi farklı görevlere yönelik modeller bu çalışmanın kapsamı dışında bırakılmıştır.

5.3.2. Enerji ölçüm metodolojisi

Enerji tüketimi ölçümleri, priz tipi bir dijital wattmetre (Brennenstuhl PM 231 E) kullanılarak gerçekleştirilmiştir. Bu yöntem pratik olmakla birlikte, örneğin Monsoon Power Monitor gibi daha yüksek örnekleme frekansına ve hassasiyetine sahip özel veri toplama kartları (DAQ) veya platformların kendi iç sensörlerinden (eğer varsa ve erişilebiliyorsa, örneğin Jetson platformlarındaki tegrastats gibi araçlar) elde edilecek anlık güç verileri, daha hassas ve dinamik enerji profilleri sunabilir. Ayrıca, sadece cihazın toplam güç tüketimi ölçülmüş, örneğin CPU, GPU, bellek gibi farklı bileşenlerin ayrı ayrı enerji tüketimini belirlemeye yönelik bir ayrıştırma yapılmamıştır.

5.3.3. Karbon ayak izi hesaplaması

Dijital karbon ayak izi hesaplamalarında, Türkiye için ulusal ortalama elektrik üretim emisyon faktörü kullanılmıştır. Bu, genel bir yaklaşım sunmakla birlikte, örneğin farklı ülkelerin veya hatta aynı ülke içindeki farklı bölgelerin yenilenebilir enerji

karışımına bağlı olarak değişebilen yerel emisyon faktörleri dikkate alınmamıştır. Ayrıca, hesaplamalar sadece cihazların operasyonel enerji tüketimini (kullanım fazı) kapsamış; donanımların üretimi, hammaddelerinin çıkarılması, lojistiği ve ömür sonu bertarafı gibi süreçleri içeren tam yaşam döngüsü analizi (LCA) bu çalışmanın kapsamı dışında kalmıştır.

5.4. Sonuç

Bu tez çalışmasında, dört farklı uç bilişim platformu (Raspberry Pi 5, NVIDIA Jetson Xavier NX, Google Coral TPU, Khadas VIM3 Pro) üzerinde dört popüler derin öğrenme modelinin (MobileNetV2, ShuffleNetV2, SqueezeNet, ResNet-18) enerji tüketimi, performans karakteristikleri ve dijital karbon ayak izi kapsamlı bir şekilde incelenmiş ve karşılaştırılmıştır.

Donanım platformu seçiminin kritik rolü, araştırma, donanım platformu seçiminin, uç cihazlarda yapay zeka uygulamalarının enerji verimliliği ve çevresel sürdürülebilirliği üzerinde doğrudan ve belirleyici bir etkiye sahip olduğunu açıkça ortaya koymuştur. Özelleşmiş YZ hızlandırıcılar olan Google Coral TPU ve Khadas VIM3 Pro (NPU), test edilen modellerin çoğunda en düşük enerji tüketimini ve dolayısıyla en düşük karbon ayak izini sergileyerek enerji açısından en verimli seçenekler olarak öne çıkmıştır. Örneğin, Google Coral TPU, SqueezeNet modelini çalıştırırken Raspberry Pi 5'e kıyasla yaklaşık %41 daha az güç tüketmiştir.

Model mimarisi ve optimizasyonun önemi, modelin karmaşıklığı ve uygulanan optimizasyon stratejileri, enerji tüketimini ve performansı önemli ölçüde etkilemektedir. Daha hafif modeller (örn: SqueezeNet, ShuffleNetV2) genellikle daha az enerji tüketirken, INT8 kuantizasyonu ve TensorRT gibi platforma özgü optimizasyonlar, özellikle NVIDIA Jetson Xavier NX gibi GPU tabanlı platformlarda, enerji tüketiminde, bazı durumlarda %47'ye varan oranlarda iyileşme sağlamıştır. Bu, yalnızca donanım seçmenin yeterli olmadığını, yazılım ve model seviyesinde de optimizasyonun şart olduğunu göstermektedir.

Performans ve enerji arasındaki denge, yüksek hesaplama gücü sunan NVIDIA Jetson Xavier NX, özellikle karmaşık modellerde (örn: ResNet-18) daha iyi çıkarım hızları sağlarken, enerji tüketimi açısından özelleşmiş hızlandırıcılardan daha yüksek değerler sergilemiştir. Genel amaçlı CPU'ya sahip Raspberry Pi 5 ise, en yüksek enerji tüketimini ve en düşük performansı göstermesine rağmen, belirli hafif modeller ve

basit görevler için düşük maliyetli bir alternatif olmaya devam etmektedir. Bu durum, uygulama gereksinimlerine (hız, doğruluk, maliyet, enerji bütçesi) göre bir "en iyi" platformun değişebileceğini, optimum çözümün genellikle bir ödünleşim (trade-off) yönetimi gerektirdiğini ortaya koymuştur.

Çevresel etkinin nicel değerlendirilmesi, çalışma, farklı donanım-model kombinasyonlarının dijital karbon ayak izlerini nicel olarak hesaplayarak, enerji verimli seçimlerin çevresel sürdürülebilirliğe doğrudan katkı sağladığını göstermiştir. En düşük karbon ayak izi, beklendiği gibi, en düşük enerji tüketen platformlar ve modellerle elde edilmiştir.

Sonuç olarak, bu tez, uç cihazlarda YZ uygulamaları geliştirilirken, hedeflenen uygulamanın özelliklerine, performans beklentilerine, enerji bütçesine ve çevresel etki hedeflerine uygun olarak donanım, model ve optimizasyon stratejilerinin bütüncül bir yaklaşımla seçilmesi gerektiğini vurgulamaktadır. Elde edilen bulgular, bu karmaşık karar verme sürecinde araştırmacılara ve geliştiricilere yol gösterici nitelikte pratik veriler ve çıkarımlar sunmaktadır.

5.5. Gelecek Çalışmalar İçin Öneriler

Bu tez çalışmasında elde edilen bulgular ve karşılaşılan sınırlılıklar ışığında, uç bilişimde enerji verimli ve sürdürülebilir yapay zeka alanında gelecekte yürütülebilecek araştırmalar için aşağıdaki teknik öneriler sunulmaktadır.

5.5.1. Genişletilmiş donanım ve model karşılaştırmaları

Yeni nesil ve farklı mimarilerdeki uç hızlandırıcıların incelenmesi, bu çalışmada test edilen platformlara ek olarak, piyasaya yeni çıkan RISC-V tabanlı YZ hızlandırıcılar, farklı üreticilere ait NPU'lar (örn: Qualcomm Hexagon, Apple Neural Engine - eğer erişilebilirse), FPGA tabanlı esnek hızlandırıcı çözümleri ve özel amaçlı ASIC (Application-Specific Integrated Circuit) tasarımlarının enerji tüketimi, performans ve karbon ayak izi açısından karşılaştırılması, daha kapsamlı bir tablo sunacaktır. Özellikle, bu donanımların farklı veri türleri (INT4, BFLOAT16 vb.) ve seyrek matris işlemleri (sparse computation) konusundaki yetenekleri incelenebilir.

Farklı görev ve model türlerinin analizi, bu tez görüntü sınıflandırma modellerine odaklanmıştır. Gelecek çalışmalarda, nesne tespiti (örn: YOLOv5/v7/v8, SSD), anlamsal segmentasyon (örn: DeepLabV3+), doğal dil işleme (örn: hafif Transformer

modelleri, BERT varyantları) ve zaman serisi analizi gibi farklı yapay zeka görevlerine yönelik modellerin uç cihazlardaki enerji profilleri ve performansları incelenebilir.

5.5.2. Detaylı enerji profilleme ve termal analiz

Bileşen bazlı enerji ayrıştırması, cihazın toplam güç tüketimi yerine, CPU, GPU, NPU/TPU, bellek ve diğer çevresel birimlerin ayrı ayrı enerji tüketimlerini ölçebilecek donanım (örn: Monsoon Power Monitor gibi) veya yazılım (örn: platforma özgü güç profilleme araçları) tabanlı yöntemler kullanılmalıdır. Bu, enerji darboğazlarının daha net tespit edilmesini sağlar.

Dinamik enerji ve termal davranış analizi, modellerin farklı iş yükleri altında (örn: sürekli video akışı işleme, aralıklı çıkarım yapma) zaman içindeki anlık güç değişimleri, pik güç tüketimleri ve cihazın termal davranışı (ısınma ve soğuma profilleri) incelenmelidir. Bu, uzun süreli çalışmalarda sistem kararlılığı ve potansiyel performans düşüşleri (thermal throttling) hakkında bilgi verir.

5.5.3. Çok amaçlı optimizasyon ve karar destek sistemleri

Uç YZ uygulamaları için "en iyi" çözüm genellikle birden fazla ve birbiriyle çelişen kritere (doğruluk, hız, enerji, maliyet, model boyutu, karbon ayak izi) bağlıdır. Gelecekte, bu metrikleri aynı anda optimize etmeyi hedefleyen çok amaçlı optimizasyon (Multi-Objective Optimization - MOO) algoritmaları (örn: NSGA-II, MOEA/D) kullanılarak Pareto-optimal çözümler kümesi belirlenebilir. Bu çözümler, geliştiricilere farklı ödünleşimler arasından kendi önceliklerine en uygun donanım-model-optimizasyon kombinasyonunu seçmeleri için bir karar destek çerçevesi sunabilir.

5.5.4. Federated learning ve dağıtık uç yapay zeka senaryoları

Veri gizliliğini koruyarak birden fazla uç cihaz üzerinde dağıtık model eğitimi ve çıkarımı sağlayan Federated Learning (FL) yaklaşımlarının enerji tüketimi ve iletişim maliyetleri incelenebilir. Farklı FL algoritmalarının ve cihaz heterojenliğinin bu metriklere etkisi araştırılmalıdır. Bu öneriler, mevcut çalışmanın üzerine inşa edilebilecek ve uç bilişimde sürdürülebilir yapay zeka alanındaki bilgi birikimine katkıda bulunabilecek potansiyel araştırma yönlerini göstermektedir.

KAYNAKLAR

Aghdam, H. H., & Heravi, E. J. (2017). Guide to intelligent building energy management systems. Springer.

Anthony, L. F. W., Kanding, B., & Selvan, R. (2020). Carbontracker: Tracking and predicting the carbon footprint of training deep learning models. arXiv:2007.03051. https://arxiv.org/abs/2007.03051

Banbury, C. R., Reddi, V. J., Torelli, P., Holleman, J., & Jeffries, N. (2020). Micronets for low-resource edge devices. Advances in Neural Information Processing Systems, 33. https://arxiv.org/abs/2007.08063

Batty, M. (2013). The new science of cities. MIT Press.

Bağcı, M. (2024). Mısır tarlasında görüntü işleme yöntemi ile yabancı otların tespiti (Yüksek lisans tezi, Pamukkale Üniversitesi, Fen Bilimleri Enstitüsü, Mekatronik Mühendisliği Anabilim Dalı). Pamukkale Üniversitesi.

Bengio, Y. (2009). Learning deep architectures for AI. Foundations and Trends® in Machine Learning, 2(1), 1–127. https://doi.org/10.1561/2200000006

Bengio, Y., LeCun, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444. https://doi.org/10.1038/nature14539

Chen, H., & Zhang, T. (2022). NPU-accelerated AI inference on embedded systems. Journal of Embedded Computing, 14(1), 27–36.

Chen, W., Liu, Y., & Zhang, H. (2021). A survey on hardware accelerators for deep learning. Journal of Computer Science and Technology, 36(4), 667–690.

Coral AI. (t.y.). Coral Accelerator Module datasheet. https://coral.ai/static/files/Coral-Accelerator-Module-datasheet.pdf adresinden 20 Mart 2025 tarihinde alınmıştır.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273–297. https://doi.org/10.1007/BF00994018

Davis, P., & Thompson, R. (2022). Strategies for reducing energy consumption in end devices. Journal of Environmental Science and Technology, 56(9), 1057–1074.

Demir, B. (2023). Yapay zeka ve mobil teknoloji. Teknolojik Gelişmeler, 4(1), 45–59.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition (ss. 248–255). IEEE. https://doi.org/10.1109/CVPR.2009.5206848

Deng, L., Li, G., Han, S., Shi, L., & Xie, Y. (2020). Model compression and hardware acceleration for neural networks: A comprehensive survey. Proceedings of the IEEE, 108(4), 485–532.

Enerji ve Tabii Kaynaklar Bakanlığı. (2024, Mart). Türkiye Elektrik Üretimi ve Elektrik Tüketim Noktası Emisyon Faktörleri Bilgi Formu. https://enerji.gov.tr/Media/Dizin D/tr/ÇevreVeİklim/İklimDeğişikliği/E misyonFaktorleri/TEUVETN_E _Faktörleri_Bilgi_Formu.pdf adresinden 20 Mart 2025 tarihin nıştır.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., & Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. Nature, 542(7639), 115–118. https://doi.org/10.1038/nature21056

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.

Han, S., Mao, H., & Dally, W. J. (2015). Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. arXiv preprint arXiv:1510.00149. https://arxiv.org/abs/1510.00149

Han, S., Pool, J., Tran, J., & Dally, W. (2016). Learning both weights and connections for efficient neural network. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, & R. Garnett (Eds.), Advances in Neural Information Processing Systems, 29 (ss. 1135–1143). Curran Associates, Inc.

Li, X., Wang, R., & Sun, Y. (2021). Safety and efficiency in high-end devices: The role of error prevention. IEEE Transactions on Industrial Informatics, 17(5), 3234–3242.

Li, X., Zhang, L., & Wang, R. (2021). Balancing power, performance, and energy efficiency in edge computing. ACM Transactions on Embedded Computing Systems, 20(3), Madde 24.

Lin, Y., Zhang, K., & Zhou, H. (2021). Edge AI acceleration using Jetson platforms. IEEE Access, 9, 122345–122356. https://doi.org/10.1109/ACCESS.2021.3104537

Ma, N., Zhang, X., Zheng, H.-T., & Sun, J. (2018). ShuffleNet V2: Practical guidelines for efficient CNN architecture design. In Proceedings of the European Conference on Computer Vision (ECCV) (ss. 116–131). Springer.

Mansouri, A., Haghbin, S., & Ranjbaran, M. (2023). Energy efficiency in IoT networks using machine learning algorithms. International Journal of Distributed Sensor Networks, 19(4).

McCarthy, J., Minsky, M. L., Rochester, N., & Shannon, C. E. (1955). A proposal for the Dartmouth summer research project on artificial intelligence, August 31, 1955. AI Magazine, 27(4), 12.

McCarthy, J., Minsky, M. L., Rochester, N., & Shannon, C. E. (1956). A proposal for the Dartmouth Summer Research Project on Artificial Intelligence [Yayınlanmamış taslak]. Department of Mathematics, Dartmouth College.

Michaud, D., Hsu, J., & Hayward, S. (2020). Challenges in the implementation of cryptographic algorithms for IoT devices. Security and Privacy, 3(2), e122.

Miller, T., & Garcia, R. (2021). Enhancing environmental impact through technology optimization. Sustainable Computing: Informatics and Systems, 30, 100590.

Minsky, M. L. (1961). Steps toward artificial intelligence. Proceedings of the IRE, 49(1), 8–30. https://doi.org/10.1109/JRPROC.1961.287775

Mishra, D., Gupta, A., & Kumar, P. (2022). Enhancing IoT device functionality through advanced machine learning techniques. Journal of Network and Computer Applications, 195, 103256.

NVIDIA. (2022). GPU architecture [PDF]. https://images.nvidia.com/aem-dam/Solutions/geforce/blackwell/nvidia-rtx-blackwell-gpu-architecture.pdf adresinden 20 Mart 2025 tarihinde alınmıştır.

Öztürk, S. (2020). Enerji tüketimi ve mobil teknolojiler. Enerji Verimliliği Araştırmaları, 3(3), 102–115.

Patel, M., Joseph, M., & Lee, K. (2024). Optimizing energy consumption in IoT systems: A machine learning approach. Journal of Internet of Things, 10, 25–40.

Pichler, M., & Hartig, F. (2023). Machine learning and deep learning—A review for ecologists. Methods in Ecology and Evolution, 14(3), 731–750.

Quinlan, J. R. (1986). Induction of decision trees. Machine Learning, 1(1), 81–106. https://doi.org/10.1007/BF00116251

Raina, R., Madhavan, A., & Ng, A. Y. (2009). Large-scale deep unsupervised learning using graphics processors. In Proceedings of the 26th Annual International Conference on Machine Learning (ICML) (ss. 873–880). ACM. https://doi.org/10.1145/1553374.1553486

Raspberry Pi Foundation. (t.y.). Raspberry Pi 5 product brief [PDF]. https://datasheets.raspberrypi.com/rpi5/raspberry-pi-5-product-brief.pdf adresinden 20 Mart 2025 tarihinde alınmıştır.

Raspberry Pi Foundation. (t.y.). 27W USB-C power supply product brief. https://datasheets.raspberrypi.com/power-supply/27w-usb-c-power-supply-product-brief.pdf adresinden 20 Mart 2025 tarihinde alınmıştır.

Russell, S. J., & Norvig, P. (2010). Artificial intelligence: A modern approach (3. baskı). Pearson Education.

Russell, S. J., & Norvig, P. (2021). Artificial intelligence: A modern approach (4. baskı). Pearson.

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. IBM Journal of Research and Development, 3(3), 210–229.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural Networks, 61, 85–117. https://doi.org/10.1016/j.neunet.2014.09.003

Smith, J., & Brown, D. (2020). Evolution of energy efficiency techniques in modern devices. Journal of Innovation and Technology Management, 24(1), 85–94.

Stokes, J. (2020). The evolution of CPU architecture. Journal of Computer Design, 55(3), 245–254.

Ullrich, K., Meeds, E., & Welling, M. (2017). Soft weight-sharing for neural network compression. arXiv:1702.04008. https://arxiv.org/abs/1702.04008

Upton, E. (2023). Raspberry Pi 5 Technical Overview. Raspberry Pi Foundation. https://www.raspberrypi.com/products/raspberry-pi-5/ adresinden 20 Mart 2025 tarihinde alınmıştır.

Wang, C., Wang, Z., Li, K., & Yan, L. (2022). Lightweight object detection model fused with feature pyramid. Multimedia Tools and Applications, 81, 11891–11911.

Wang, J. (2018). Industrial applications of deep learning: A survey. IEEE Access, 6, 23955–23969.

Wang, Y., Lin, J., Annavaram, M., Iyer, R., & Hsu, C. H. (2019). A comprehensive study of deep learning hardware and workloads. In 2019 IEEE International Symposium on Workload Characterization (IISWC) (ss. 227–238). IEEE.

Wang, Y., Zhang, L., & Liu, G. (2023). The bandwidth and latency implications of different operational modes for hardware trust anchors in IoT environments. Internet of Things, 25, 100496.

Wu, C. J., Huang, D., & Raghavendra, R. (2021). Carbon emissions of deep learning at the edge. arXiv preprint arXiv:2105.06087. https://arxiv.org/abs/2105.06087

Yalçın, R. (2022). Derin öğrenmenin mobil cihazlardaki yeri. Geleceğin Teknolojileri, 8(4), 34–46.

Yang, Y., Wang, L., & Yu, L. (2020). A review of deep learning in remote sensing: Applications and challenges. ISPRS Journal of Photogrammetry and Remote Sensing, 158, 1–22.

Yuan, Z., Zheng, Y., & Yu, Y. (2019). Review of cryptography in system-on-chip applications: Opportunities and challenges. IEEE Transactions on Circuits and Systems II: Express Briefs, 66(7), 1090–1094.

Zhang, C., Wang, H., & Zheng, Y. (2020). Edge computing: A new computing paradigm. IEEE Internet of Things Journal, 7(4), 2808–2820.

Zhang, Q., Neri, F., & Celesti, A. (2018). Deep learning on the edge: Challenges and trends. ACM Computing Surveys (CSUR), 51(5), Madde 91.

Zhang, Y., Chen, X., & Li, M. (2020). Comprehensive review of digital carbon footprints. Environmental Science & Policy, 112, 45–56.

Zhang, Y., Wei, C., & Wu, K. (2020). Artificial intelligence and Internet of Things: Bridging the gap. Future Generation Computer Systems, 108, 1–8.

Zhang, Y., Wu, J., & Wang, T. (2021). An overview of neural processing units: Driving AI applications at scale. IEEE Transactions on Neural Networks and Learning Systems, 32(8), 3443–3456.

Zhou, T., Zhang, Y., & Sun, W. (2022). Comparative analysis of deep learning classifiers for IoT medical applications. IEEE Transactions on Biomedical Engineering, 69(7), 2345–2357.

Zhou, Y., Liu, X., & Zhao, J. (2021). Advances in deep learning techniques for remote sensing applications. Remote Sensing, 13(15), 2867.

Zhou, Z., Chen, X., Li, E., Zeng, L., Luo, K., & Zhang, J. (2019). Edge intelligence: Paving the last mile of artificial intelligence with edge computing. Proceedings of the IEEE, 107(8), 1738–1762.

ÖZGEÇMİŞ

Ad-Soyad : Çağlar ŞİMŞEK

ÖĞRENİM DURUMU:

- **Lisans :** 2016, Sakarya Üniversitesi, Bilgisayar ve Bilişim Sistemleri, Bilgisayar Mühendisliği