T.C.

BİLECİK ŞEYH EDEBALİ ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

DERİN ÖĞRENME TABANLI BİTKİ BAKIM VE HASTALIK TESPİT SİSTEMİ

Zeynep KAYA

37558061498

DANIŞMAN

Prof. Dr. Cihan KARAKUZU

BM332 Bilgisayar Mühendisliği Tasarım Çalışması II

BİLECİK 2025

BİLDİRİM

Bu çalışmada bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

İmza

ZEYNEP KAYA

Tarih: 15 Haziran 2025

ÖZET

DERİN ÖĞRENME TABANLI BİTKİ BAKIM VE HASTALIK TESPİT SİSTEMİ

Bu çalışma, mısır bitkisine ait yaprak hastalıklarının erken teşhisini sağlamak amacıyla derin öğrenme tabanlı bir görüntü sınıflandırma sistemi geliştirmeyi hedeflemektedir. Tarımsal üretimde verimliliği artırmak ve kimyasal kullanımını azaltmak adına, yaprak görüntüleri analiz edilerek hastalıklı ve sağlıklı yapraklar sınıflandırılmıştır. Kullanılan veri seti, PlantVillage veri tabanından alınmış ve yalnızca mısır bitkisine ait dört sınıf içermektedir. Çalışmada transfer öğrenme yöntemine dayalı olarak VGG16, InceptionV3 ve MobileNetV2 modelleri kullanılmıştır. Modellerin başarımları karşılaştırılmış; MobileNetV2 modeli hem eğitim süresi hem de doğruluk açısından en başarılı sonuçları vermiştir. Bu model üzerine ince ayar (fine-tuning), sezgisel hiperparametre optimizasyonu ve gelişmiş veri artırma yöntemleri uygulanarak doğruluk oranı %97 seviyelerine çıkarılmıştır. Sınıflandırma sonuçları, doğruluk, hassasiyet, duyarlılık ve F1-score gibi metriklerle değerlendirilmiş; elde edilen sonuçlar modelin genelleme gücünün yüksek olduğunu ortaya koymuştur. Sonuç olarak geliştirilen sistem, tarım sektöründe yapay zekâ destekli hastalık teşhisi konusunda etkili bir çözüm sunmakta ve sürdürülebilir tarım uygulamalarına katkı sağlamaktadır.

Anahtar Kelimeler: Derin Öğrenme, Görüntü Sınıflandırma, Bitki Hastalıkları, Transfer Öğrenme, MobileNetV2

ABSTRACT

DEEP LEARNING-BASED PLANT CARE AND DISEASE DETECTION SYSTEM

This study aims to develop a deep learning-based image classification system for the early detection of leaf diseases in maize plants. In order to increase agricultural productivity and reduce chemical usage, leaf images are analyzed to classify diseased and healthy leaves. The dataset used was obtained from the PlantVillage database and contains only four classes related to maize leaves. In the study, transfer learning-based architectures such as VGG16, InceptionV3, and MobileNetV2 were employed. The performance of these models was compared; among them, MobileNetV2 provided the most successful results in terms of both training duration and classification accuracy. Fine-tuning, heuristic hyperparameter optimization, and advanced data augmentation techniques were applied to further improve the model's accuracy, which reached up to 97%. Classification results were evaluated using accuracy, precision, recall, and F1-score metrics; the findings indicated that the model demonstrated strong generalization capability. As a result, the developed system offers an effective artificial intelligence–supported solution for disease diagnosis in agriculture and contributes to sustainable farming practices.

Keywords: Deep Learning, Image Classification, Plant Diseases, Transfer Learning, MobileNetV2

ÖNSÖZ

Tasarım II çalışmamın başından sonuna kadar emeği geçen ve beni bu konuya yönlendiren saygı değer hocam ve danışmanım Sayın Prof. Dr. Cihan KARAKUZU'ya, ayrıca tüm katkılarından ve hiç eksiltmediği desteğinden dolayı Dr. Öğr. Üyesi Burakhan ÇUBUKÇU'ya teşekkür ederim.

İmza

ZEYNEP KAYA

Tarih: 15 Haziran 2025

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

1 GİRİŞ

Tarım, insanlık tarihinin en eski uğraş alanlarından biri olmasının yanı sıra, günümüzde de toplumların sürdürülebilirliği, ekonomik refahı ve gıda güvenliği açısından temel bir sektördür. Ancak artan dünya nüfusu, iklim değişiklikleri, toprak verimliliğindeki azalma ve tarımsal hastalıklar gibi etkenler, bu sektörün verimli işlemesini olumsuz yönde etkilemektedir. [1] [2] [3] [4] [5] [6]. Bitki hastalıkları bu tehditler arasında en kritik olanlardan biridir. Birleşmiş Milletler Gıda ve Tarım Örgütü'ne (FAO) göre, her yıl küresel gıda üretiminin yaklaşık %20'si hastalıklar ve zararlılar nedeniyle kaybedilmektedir [7]. Bu durum yalnızca üretim kaybına yol açmamakta, aynı zamanda küçük ölçekli çiftçilerin geçim kaynaklarını da tehdit etmektedir.

Yaprak hastalıkları, bitkilerde fotosentezi doğrudan etkileyerek gelişimi yavaşlatmakta, ürün kalitesini düşürmekte ve bazı durumlarda bitkinin tamamen ölmesine neden olabilmektedir [8] [9] [10]. Bu hastalıkların çoğu erken evrede tespit edilemediği için hızla yayılmakta ve kontrol altına alınması zorlaşmaktadır. Özellikle mısır, buğday, domates, pirinç gibi temel tarım ürünleri yaprak hastalıklarına karşı son derece hassastır [11].

Geleneksel teşhis yöntemleri genellikle tarım uzmanlarının görsel değerlendirmesine dayanmakta, bu da hem zaman alıcı hem de insan hatasına açık bir sürece sebep olmaktadır. Bu yöntemler büyük arazilerde ölçeklenebilir değildir ve kırsal bölgelerde uzman erişimi yetersiz olduğunda ciddi gecikmelere neden olmaktadır. Bu bağlamda, bilgisayarla görme (computer vision) ve yapay zekâ tabanlı teşhis sistemleri; hızlı, doğru ve otomatik karar desteği sağlama potansiyeliyle öne çıkmaktadır [12] [13] [14] [15]. Özellikle derin öğrenme (deep learning) mimarileri, son yıllarda tarım teknolojileri alanında yaygın bir şekilde kullanılmaya başlanmıştır. Evrişimli Sinir Ağları (CNN), görüntülerdeki desen, renk ve yapısal özellikleri insan müdahalesi olmadan öğrenebilme yeteneğiyle, bitki hastalıklarının tanı ve sınıflandırılmasında önemli başarılar elde etmektedir [16]. Mohanty vd. (2016), PlantVillage veri seti ile AlexNet ve GoogLeNet kullanarak %99'un üzerinde doğruluk elde etmiştir [17]. Sladojevic vd. (2016), 13 farklı bitki hastalığını sınıflandırmak için eğittiği bir CNN modeli ile %96.3 doğruluk elde etmiştir [18].

Brahimi vd. (2017), sınırlı veri ortamında transfer öğrenme kullanarak başarıyı artırmış-

tır [19]. Bu yaklaşımla daha küçük veri setlerinde bile etkili modeller oluşturmak mümkün hale gelmiştir. Too vd. (2019), VGG16, InceptionV3 ve MobileNetV2 modellerini karşılaştırmış; VGG16'nın yüksek doğruluk sunduğunu ancak hesaplama maliyetinin yüksek olduğunu vurgulamıştır [20]. MobileNetV2 ise diğer modellere kıyasla özellikle düşük kaynaklı ortamlarda ve mobil cihazlarda kullanılabilirliği ile dikkat çekmiştir [21].

Liu vd. (2018) yaygın elma yaprağı hastalıklarının yüksek doğrulukla tanımlanabilmesi için AlexNet tabanlı derin evrişimli sinir ağı (CNN) mimarisi kullanmışlardır. Görüntü üretiminde yön bozulması, ışık bozulması ve PCA jittering gibi teknikler kullanılarak doğal veri yetersizliği aşılmış ve modelin genelleme yeteneği artırılmıştır. Elde edilen sonuçlar, önerilen modelin %97,62 doğruluk oranıyla çalıştığını, standart AlexNet modeline göre 51 milyondan fazla parametre azaltılarak hem daha verimli hem de daha hızlı yakınsayan bir yapı elde edildiğini göstermiştir [22]. Lu vd. (2023) MobileNetV2 mimarisini temel alarak, RepMLP (Re-parameterized Multi-Layer Perceptron), Efficient Channel Attention (ECA) ve Hardswish aktivasyon fonksiyonu gibi bileşenleri entegre ettikleri yeni bir hafif model geliştirmiştir. Geliştirilen model, PlantVillage veri seti üzerinde %99,53 gibi yüksek bir doğruluk oranı elde etmiştir. Bu oran, orijinal MobileNetV2'ye kıyasla %0,3 daha yüksektir. Modelin parametre sayısı %59 azaltılmış ve bu sayede modelin hesaplama yükü önemli ölçüde hafifletilmiştir. RepMLP modülü ile uzun mesafeli özellik ilişkileri yakalanmış, böylece modelin küresel algılama yeteneği artırılmıştır. Ancak, mevcut veri setinin kontrollü koşullarda toplanmış olması, gerçek dünya senaryolarında modelin performansını sınırlayabilmektedir. [23].

Bitki hastalıkları tespitinde derin öğrenme modellerinde önemli ilerlemeler kaydedilmiş olsa da literatürde dikkat çeken önemli boşluklar vardır. Örnek vermek gerekirse modellerin çoğu, laboratuvar ortamında, homojen arka plana sahip yüksek kaliteli görüntülerle eğitilmiştir [24]. Gerçek dünyada ise yaprak üzerinde toz, su damlası, yırtık gibi faktörler bulunmakta; bu da modellerin genelleme kabiliyetini sınırlandırmaktadır [25] [26]. Düşük çözünürlüklü görüntülerle InceptionV3 %87 ile yüksek oranda doğruluk sağlasa da sınıflar arası ayırt ediciliği zayıf kalmıştır [27] [28]. Bir diğer eksiklik, sınıf dengesizliği problemidir. Buda vd. (2018), CNN'lerin nadir sınıflara karşı duyarsız olduğunu ve bu durumun genel doğruluk metriğini yanıltıcı hale getirdiğini belirtmiştir. Buda vd. (2018), azınlık sınıflarına ait görsellerin çoğaltılmadan modele verilmesinin modelin karar sınırlarını daralttığını göstermiştir [29].

Mevcut çalışmalar genellikle sadece doğruluk (accuracy) oranına odaklanmakta, ancak hassasiyet (precision), duyarlılık (recall), F1-score gibi metriklere yeterince yer verilmemektedir. Oysa bu metrikler, özellikle sınıf dengesizliği içeren durumlarda daha güvenilir performans ölçütleri sunmaktadır.

Model başarımını artırmak için son yıllarda çeşitli stratejiler benimsenmiştir. Bunlar arasında veri artırma (data augmentation), öğrenme oranı planlaması (learning rate scheduling), ince ayar (fine-tuning) ve hiperparametre optimizasyonu yer almaktadır [30] [31] [32] [33]. Wang (2017), özellikle görüntü döndürme, parlaklık ayarı ve çevirme gibi yöntemlerin CNN modellerinin ezberleme (overfitting) eğilimini azalttığını belirtmiştir [34]. Bitki hastalığı şiddetinin otomatik ve hassas bir şekilde sınıflandırılması için derin öğrenme temelli bir yöntem önermiştir. Az sayıda eğitim verisiyle çalışılan bu çalışmada, sıfırdan eğitilen küçük CNN modelleri ile birlikte dört farklı derin öğrenme mimarisi (VGG16, VGG19, Inception-v3, ResNet50) transfer öğrenme yöntemiyle karşılaştırılmıştır. En iyi performans, transfer öğrenme ile eğitilen VGG16 modeli ile elde edilmiştir ve bu model %90,4 doğruluk oranına ulaşmıştır. Bu sonuç, önceden eğitilmiş derin modellerin az veriyle bile yüksek doğruluk sağlayabildiğini ve derin öğrenmenin tam otomatik bitki hastalığı şiddeti sınıflandırmasında umut vadeden bir teknoloji olduğunu göstermektedir [35].

Ayrıca klasik CNN mimarilerine ek olarak BiLSTM, GRU ve Transformer tabanlı karma (hibrit) modellerin de son yıllarda kullanılmaya başlandığı görülmektedir [36] [37] [38]. Ancak bu karma (hibrit) yaklaşımlar çoğu zaman daha fazla veri ve hesaplama gücü gerektirdiğinden mobil uygulamalarda sınırlı kullanıma sahiptir.

Bu çalışmada, yukarıda belirtilen eksiklikleri dikkate alarak mısır bitkisine ait dört yaprak sınıfı (Cercospora Leaf Spot, Common Rust, Northern Leaf Blight, Healthy) üzerinde çalışan bir sınıflandırma sistemi geliştirilmiştir. VGG16, InceptionV3 ve MobileNetV2 mimarileri karşılaştırılmış; en iyi sonucu veren MobileNetV2 modeli ince ayar (fine-tuning), gelişmiş veri artırma ve sezgisel hiperparametre optimizasyonları ile iyileştirilmiştir. Böylece, gerçek tarla koşullarında uygulanabilirliği yüksek bir karar destek sistemi önerilmiştir.

2 KULLANILAN YAZILIM ARAÇLARI VE YÖNTEMLER

Bu çalışmada, mısır bitkisine ait yaprak görüntülerinden hastalıkların tespit edilmesi amacıyla derin öğrenme tabanlı görüntü sınıflandırma yaklaşımları uygulanmıştır. Kullanılan veri seti, PlantVillage veri tabanından temin edilmiş olup dört sınıfa ayrılmıştır: Cercospora Leaf Spot, Common Rust, Northern Leaf Blight ve Healthy (sağlıklı). Görüntüler, her sınıf için ayrı klasörlerde tutulmakta ve eğitim, doğrulama ve test veri kümeleri oluşturulurken bu sınıflandırma yapısına sadık kalınmıştır.

Veri ön işleme adımlarında, tüm görüntüler MobileNetV2 gibi transfer öğrenme modelleriyle uyumlu hale getirilmek üzere 224x224 piksel boyutuna yeniden ölçeklendirilmiş ve RGB kanal yapısında normalize edilerek 0 ile 1 arasında değer aralığına dönüştürülmüştür. Veri artırma (augmentation) işlemleri yalnızca eğitim veri kümesine uygulanmış; bu kapsamda döndürme (rotation), yatay-dikey kaydırma (shift), yakınlaştırma (zoom), kesme (shear), yatay çevirme (horizontal flip) ve parlaklık değişimi (brightness range) gibi teknikler kullanılmıştır. Doğrulama ve test verilerine yalnızca yeniden boyutlandırma ve normalizasyon işlemleri uygulanmıştır.

Model mimarileri olarak üç farklı önceden eğitilmiş (pre-trained) derin öğrenme ağı tercih edilmiştir: VGG16, InceptionV3 ve MobileNetV2. Bu modellerin ortak noktası, temel katmanlarının dondurularak yalnızca sınıflandırma katmanlarının yeniden yapılandırılmasıdır. Ancak ince ayar (fine-tuning) işlemi sadece MobileNetV2 modeline uygulanmış ve bu modelin bazı son katmanları yeniden eğitilmiştir. Tüm modeller için eğitim şu sabit hiperparametrelerle yürütülmüştür: yığın büyüklüğü (batch size) 32, tur (epoch) sayısı 10 (MobileNetV2 modelinde veri oaranları düzeltildikten sonra epoch sayısı 15 olarak çalışmalar yapılmıştır), terk (dropout) oranı 0.4, eğitim algoritması olarak Adam, öğrenme oranı 0.00005 ve kayıp fonksiyonu olarak kategorik çarpraz entropi (categorical crossentropy) seçilmiştir.

Eğitim sırasında, EarlyStopping fonksiyonu ile aşırı öğrenme (overfitting) riski azaltılmış ve en iyi sonuçlar veren ağırlıklar korunmuştur. Ayrıca, modelin başarımı eğitim ve doğrulama süreci boyunca grafiksel olarak izlenmiş ve eğitim sonuçları CSV formatında kaydedilmiştir.

Elde edilen modellerin başarımı sadece doğruluk (accuracy) değeriyle değil; aynı zamanda

karmaşıklık matrisi (confusion matrix), sınıflandırma raporu (classification report), precision, recall ve F1-score gibi metriklerle ayrıntılı olarak değerlendirilmiştir. Bu çok yönlü başarım değerlendirmesi, özellikle sınıf dengesizliği gibi problemlere karşı daha güvenilir analiz sağlamıştır.

Ayrıca, yalnızca MobileNetV2 mimarisi üzerinde sezgisel hiperparametre optimizasyonları (örnek vermek gerekirse dropout oranları ve dense katman boyutları) uygulanmış; bu sayede modelin genelleme yeteneği iyileştirilmiştir. Farklı mimarilerin sonuçları karşılaştırmalı olarak analiz edilmiştir. Tüm bu süreçlerin sonucunda, tarla koşullarında uygulanabilirliği yüksek, hafif ve etkili bir sınıflandırma modeli önerilmiştir.

2.1 Veri Seti

Bu projede kullanılan veri seti, Kaggle platformunda yayımlanan açık erişimli PlantVillage veri setidir. Bu veri seti, dokuz farklı bitki türüne ait hem sağlıklı hem de çeşitli hastalıklarla ilgili görseller içeren 70.000'den fazla yüksek kaliteli yaprak görüntüsünü içermektedir [39]. Görseller, farklı büyüme aşamalarını, değişken ışık koşullarını ve hastalık çeşitliliğini yansıtmaktadır. Veri seti; eğitim (train), doğrulama (validation) ve test (test) kümelerine ayrılmıştır.

Bu çalışmada yalnızca mısır (Zea mays) bitkisine ait yaprak görüntüleri kullanılmış ve bu görüntüler dört sınıfa ayrılmıştır: Cercospora Leaf Spot (Yaprak Lekesi), Common Rust (Pas Hastalığı), Northern Leaf Blight (Kuzey Yaprak Yanıklığı) ve Healthy (Sağlıklı). Başlangıçta veri setinde test verilerinin sınıflar arasında dengesiz ve yetersiz olduğu görülmüş, bu nedenle veri seti elle yeniden yapılandırılmıştır. Her sınıftan dengeli sayıda örnek içerecek şekilde eğitim, doğrulama ve test alt kümeleri oluşturulmuştur.

Yapılan bu düzenleme sonucunda veri seti yaklaşık olarak %70 eğitim, %15 doğrulama ve %15 test oranlarında olacak şekilde dağıtılmıştır. Görsel sayıları bu oranlara karşılık olarak sırasıyla 6176 (eğitim), 1645 (doğrulama) ve 1384 (test) şeklindedir.

Tüm görüntüler RGB formatında olup, derin öğrenme modelleri ile uyumlu hâle getirilmek amacıyla 224x224 piksel boyutuna yeniden ölçeklendirilmiş ve 0 ile 1 aralığına normalize edilmiştir. Bu işlem, modelin görselleri etkin biçimde işlemesini ve öğrenmesini sağlamaktadır.

Görsellere ait bazı örnekler Şekil 2.1'de sunulmuştur.

Cercospora Leaf Spot

Common Rust

Northern Leaf Blight

Healthy

Şekil 2.1: Mısır yapraklarına ait örnek görüntüler

2.2 Veri Ön İşleme ve Veri Artırma

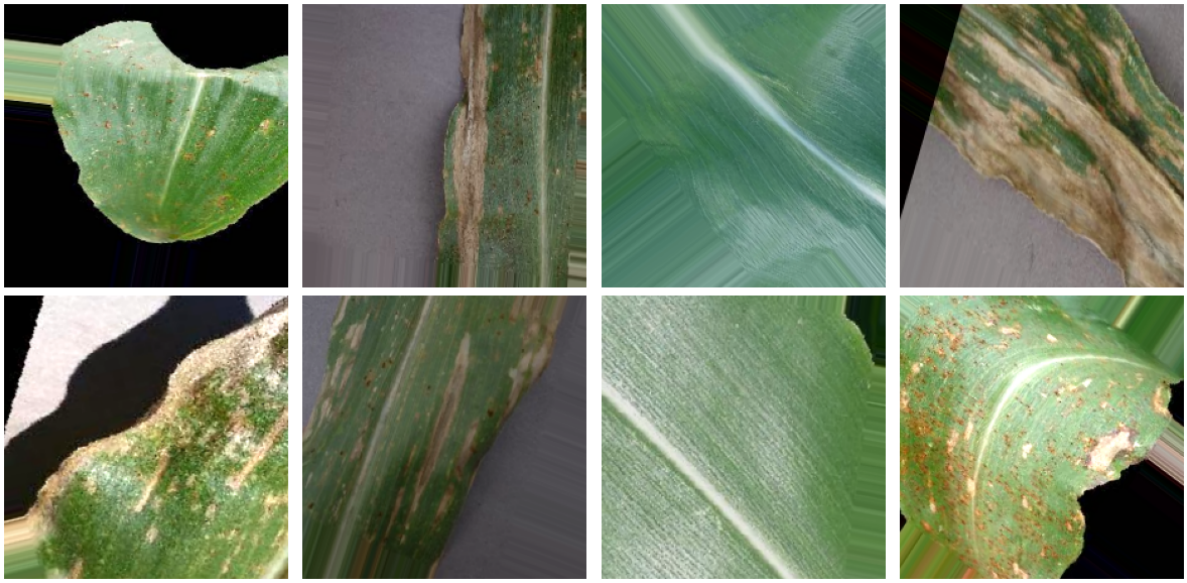
Bu çalışmada kullanılan mısır yaprağı görüntüleri, derin öğrenme modelleriyle uyumlu hale getirilmek amacıyla bir dizi ön işleme adımından geçirilmiştir. Tüm görüntüler, modellerin beklediği giriş boyutları doğrultusunda 224x224 piksel boyutuna yeniden boyutlandırılmış ve piksel değerleri 0 ile 1 aralığına normalize edilmiştir. Bu sayede modelin giriş verilerini daha etkin işlemesi sağlanmış, aynı zamanda eğitim sürecinin daha hızlı ve kararlı bir şekilde gerçekleşmesine katkı sunulmuştur.

Modelin genelleme yeteneğini artırmak ve eğitim verisi üzerindeki aşırı öğrenme (overfitting) riskini azaltmak amacıyla, sadece eğitim verisi üzerinde veri artırma (data augmentation) işlemleri uygulanmıştır. Bu amaçla TensorFlow Keras kütüphanesinde yer alan *ImageDataGe-*

nerator sınıfı kullanılmış ve çeşitli dönüşümler gerçekleştirilmiştir. Görseller, rastgele olarak 30 dereceye kadar döndürülmüş, yatay ve dikey eksende yüzde 20 oranında kaydırılmış, yüzde 20 oranında kesme (shear) işlemi uygulanmış, yüzde 20 oranında yakınlaştırılmış ve yatay olarak çevrilmiştir. Ayrıca, MobileNetV2 mimarisi için geliştirilmiş sürümde parlaklık aralığı (brightness_range[0.7, 1.3]) da veri artırma sürecine dâhil edilmiştir. Bu işlemler, modelin farklı konum, boyut ve ışık koşullarına sahip yaprak görüntülerini tanıyabilmesini sağlamış ve gerçek dünyadaki çeşitliliğe karşı daha dayanıklı hale gelmesine yardımcı olmuştur.

Doğrulama ve test veri kümelerinde ise yalnızca yeniden ölçeklendirme (rescale=1./255) işlemi uygulanmış, veri artırma işlemleri bilinçli olarak devre dışı bırakılmıştır. Bu karar, doğrulama ve test aşamalarında modelin başarımının objektif ve tutarlı bir şekilde değerlendirilmesi amacıyla alınmıştır.

Eğitim, doğrulama ve test veri kümeleri için sırasıyla *train_generator*, *validation_generator* ve *test_generator* adlarıyla üç farklı veri üretici oluşturulmuştur. Bu üreteçler sayesinde görseller diskten doğrudan okunarak, sistem belleğine yüklenmeden eğitim sürecine dâhil edilmiştir. Bu yapı, özellikle büyük veri setleriyle çalışırken bellek kullanımı açısından önemli bir avantaj sağlamaktadır. Görsellere ait bazı örnekler Şekil 2.2'de verilmiştir.



Şekil 2.2: Veri artırma işlemleri sonucunda elde edilen örnek yaprak görselleri

2.3 Kullanılan Yöntemler

2.3.1 Evrişimsel Sinir Ağı (CNN)

Evrişimsel Sinir Ağları (CNN – Convolutional Neural Networks), bilgisayarla görü uygulamalarında yaygın olarak kullanılan ve yüksek doğruluk oranlarıyla dikkat çeken derin öğrenme mimarileridir. Görüntü sınıflandırma, nesne tanıma ve tıbbi görüntü analizi gibi görevlerde özellikle başarılı sonuç vermektedirler [40]. İlk kez LeCun ve arkadaşları tarafından el yazısı rakamların tanınmasında kullanılan LeNet-5 modeli ile önerilen bu yapı, günümüzde çok daha gelişmiş sürümleriyle kullanılmaktadır [41].

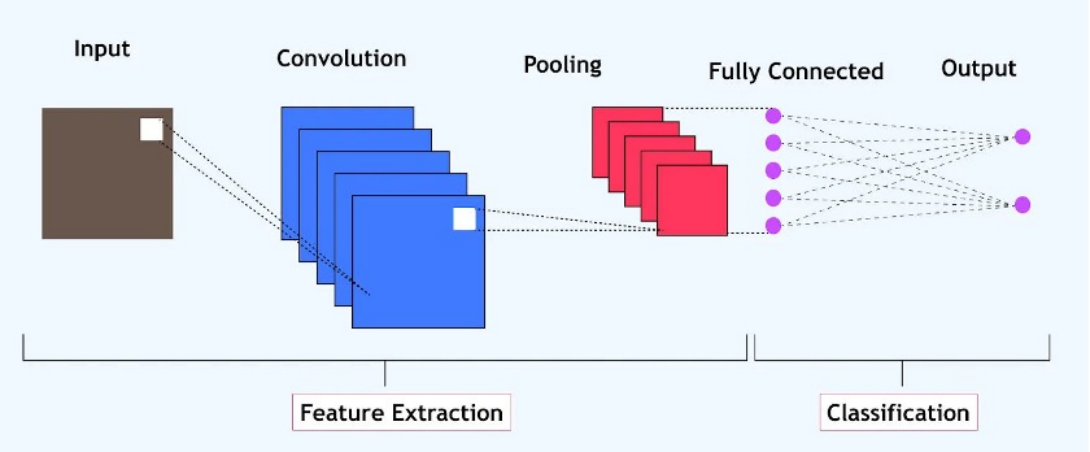
CNN mimarileri genel olarak üç temel yapı biriminden oluşur: evrişim (convolution) katmanları, havuzlama (pooling) katmanları ve tam bağlantılı (fully connected) katmanlar. Evrişim katmanları, görüntüden kenar, köşe ve renk geçişi gibi düşük seviyeli öznitelikleri çıkaran filtreler kullanır. Bu filtreler, görüntü üzerinde kayan pencere (sliding window) yöntemiyle çalışır ve her adımda yeni öznitelik haritaları (feature maps) üretir. Havuzlama katmanları ise bu haritaların boyutlarını küçültür, hesaplama yükünü azaltır ve aşırı öğrenmenin önlenmesine yardımcı olur. En sık tercih edilen havuzlama yöntemi maksimum havuzlamadır (max pooling). Son olarak tam bağlantılı katmanlar, çıkarılan özniteliklerin sınıflandırılmasını gerçekleştirir [42] [43].

CNN'lerin en büyük üstünlüklerinden biri, geleneksel yöntemlerde ihtiyaç duyulan elle yapılan öznitelik çıkarımı işlemlerini otomatikleştirmeleridir. Bu sayede model, öğrenme süreci boyunca veriden kendi anlamlı temsillerini oluşturabilir. Özellikle büyük hacimli ve çeşitli görüntü veri setlerinde bu yapı, yüksek doğruluk oranları elde etmek için vazgeçilmez hâle gelmiştir [44].

Bu çalışmada, transfer öğrenme yöntemiyle önceden eğitilmiş olan MobileNetV2, VGG16 ve InceptionV3 gibi modeller kullanılmıştır. Bu modeller, ImageNet veri kümesi üzerinde önceden eğitilmiş ağırlıklarla yüklenmiş ve mısır bitkisine ait yaprak hastalıklarının sınıflandırılması amacıyla yeniden yapılandırılarak eğitilmiştir.

CNN mimarisinin sunduğu otomatik öznitelik çıkarımı, parametrik verimlilik ve yüksek doğruluk getirileri sayesinde, bitki hastalıklarının erken tespiti ve doğru sınıflandırılması mümkün

hâle gelmiştir [45]. Şekil 2.3'te bir CNN mimarisine ait yapısal bileşenler örnek olarak sunulmuştur.



Şekil 2.3: CNN (Evrişimsel Sinir Ağı) Mimarisi
[46]

2.3.2 VGG16

VGG16, 2014 yılında Visual Geometry Group (Oxford Üniversitesi) tarafından geliştirilen derin bir evrişimli sinir ağı mimarisidir. Bu model, özellikle basit ve düzenli yapısı ile dikkat çeker. VGG16'nın en temel özelliği, küçük boyutlu (3x3) konvolüsyon filtrelerini arka arkaya kullanarak derinliği artırmasıdır. Bu yapı, parametre verimliliği ile birlikte daha karmaşık özelliklerin öğrenilmesini mümkün kılar. Ayrıca, ağın her evresinde MaxPooling katmanları ile boyut küçültülerek öznitelik çıkarımı derinleştirilir. Modelin sonunda tam bağlantılı (fully connected) katmanlar bulunur ve genellikle softmax aktivasyon fonksiyonu ile sınıflandırma yapılır [47].

Bu çalışmada VGG16 modeli, transfer öğrenme yöntemiyle kullanılmış ve modelin ImageNet üzerinde önceden eğitilmiş ağırlıkları ile temel evrişim katmanları sabitlenmiştir (include_top=False). Modelin en üst katmanı çıkartılarak yalnızca özellik çıkarımı amacıyla kullanılmıştır. Sınıflandırma kısmı, veri kümesindeki sınıflara özel olacak şekilde yeniden tasarlanmıştır.

Şekil 2.3.2'deki Python kodu, önceden eğitilmiş modelin yüklenmesini ve temel katmanların sabitlenmesini (feature extraction) göstermektedir:

```
1 # VGG16 modelini önceden eğitilmiş ImageNet ağırlıkları ile yükle
2 # Üst sınıflandırma katmanı çıkarılır (include_top=False)
```

```
# Giriş boyutu 224x224x3 olarak belirlenir
base_model = VGG16(weights="imagenet", include_top=False, input_tensor=
    Input(shape=(224, 224, 3)))
# Tüm katmanlar dondurularak eğitilemez hale getirilir
for layer in base_model.layers:
    layer.trainable = False
```

Şekil 2.3.2: VGG16 modelinin yüklenmesi ve tüm katmanların dondurulması

Modelin üstüne özelleştirilmiş sınıflandırıcı katmanlar eklenmiştir. Bu yapı, önce Flatten katmanı ile temel modelin çıkışının düzleştirilmesiyle başlar. Ardından 512 nöronlu ve ReLU aktivasyon fonksiyonuna sahip birinci tam bağlantılı (Dense) katman eklenmiştir (fc1). Bu katmandan sonra, aşırı öğrenmeyi (overfitting) önlemek amacıyla %40 oranında Dropout uygulanmıştır. İkinci Dense katman (fc2) ise 256 nöronlu olup yine ReLU aktivasyonu kullanmaktadır ve aynı şekilde %40 oranında Dropout ile takip edilmiştir. Son olarak, dört sınıfa ait tahmin yapılmasını sağlayan, softmax aktivasyon fonksiyonuna sahip çıkış katmanı (predictions) eklenmiştir. Bu sınıflandırıcı yapı, modelin genelleme yeteneğini artırmak ve sınıflar arasında doğru ayrım yapabilmesini sağlamak amacıyla tasarlanmıştır. İlgili katman yapısının modele entegrasyonuna ilişkin Python kodu Şekil 2.3.3'te sunulmuştur.

```
# VGG16 modelini önceden eğitilmiş ImageNet ağırlıklarıyla yükle
# Üst sınıflandırma katmanı dahil edilmez (include_top=False)
# Giriş boyutu 224x224x3 olarak tanımlanır
base_model = VGG16(weights="imagenet", include_top=False, input_tensor=
    Input(shape=(224, 224, 3)))
# Önceden eğitilmiş temel katmanların tümü dondurulur (eğitilemez hale
    getirilir)
for layer in base_model.layers:
    layer.trainable = False
# Temel modelin çıkışı düzleştirilerek tam bağlantılı katmanlara aktarılı
    r
x = Flatten()(base_model.output)
# 512 nöronlu, ReLU aktivasyonlu birinci Dense katman
x = Dense(512, activation="relu", name="fc1")(x)
# %40 oranında Dropout uygulanarak aşırı öğrenme riski azaltılır
x = Dropout(dropout_rate)(x)
```

```
# 256 nöronlu, ReLU aktivasyonlu ikinci Dense katman
x = Dense(256, activation="relu", name="fc2")(x)
# Yine %40 Dropout uygulanır
x = Dropout(dropout_rate)(x)
# 4 sınıf için softmax aktivasyonlu çıkış katmanı tanımlanır
output = Dense(fcDense, activation="softmax", name="predictions")(x)
# Giriş temel modelden, çıkış yeni sınıflandırıcı katmanlardan alınarak model oluşturulur
model = Model(inputs=base_model.input, outputs=output)
# Adam optimizasyon algoritması düşük öğrenme oranıyla tanımlanır
opt = Adam(learning_rate=0.00005)
# Model, kategorik sınıflandırma için uygun şekilde derlenir
model.compile(optimizer=opt, loss="categorical_crossentropy", metrics=["accuracy"])
# Modelin özet yapısı konsola yazdırılır
model.summary()
```

Şekil 2.3.3: VGG16 modelinin üzerine Flatten, Dense ve softmax katmanları eklenerek sınıflandırma yapısının oluşturulması.

Model, Adam optimizasyon algoritması ve categorical_crossentropy kayıp fonksiyonu ile eğitilmiştir. Bu eğitim süreci, doğruluk (accuracy) metriği temel alınarak Şekil 2.3.4'deki Python kodu ile gerçekleştirilmiştir.

```
# Adam optimizasyon algoritması düşük öğrenme oranı ile tanımlanır
opt = Adam(learning_rate=0.00005)
# Model derlenir: optimizer olarak Adam, kayıp fonksiyonu categorical_crossentropy
# Başarı metriği olarak doğruluk (accuracy) kullanılır
model.compile(optimizer=opt,
              loss="categorical_crossentropy",
              metrics=["accuracy"])
```

Şekil 2.3.4: VGG16 tabanlı modelin Adam optimizasyonu ile derlenmesi

Eğitim süreci boyunca, model 10 tur (epoch) boyunca eğitilmiş ve erken durdurma (early stopping) tekniği ile aşırı öğrenme (overfitting) riski azaltılmıştır. Eğitim verileri için çeşitli

veri artırma (data augmentation) teknikleri uygulanmıştır. Bu veri artırma işlemlerinin nasıl uygulandığı Şekil 2.3.5'te verilen Python kodunda ayrıntılı biçimde gösterilmiştir.

```
# Eğitim verileri için veri artırma işlemleri tanımlanır
train_datagen = ImageDataGenerator(
    rescale=1./255,              # Piksel değerlerini 0-1 aralığına
        normalize et
    rotation_range=30,           # Görüntüleri 30 dereceye kadar rastgele
        döndür
    width_shift_range=0.2,       # %20 oranında yatay kaydırma
    height_shift_range=0.2,      # %20 oranında dikey kaydırma
    shear_range=0.2,             # Kesme (shear) dönüşümü uygula
    zoom_range=0.2,              # %20 oranında yakınlaştırma/uzaklaştırma
    horizontal_flip=True,        # Rastgele yatay çevirme uygula
    fill_mode='nearest'          # Boşlukları en yakın piksel değeriyle
        doldur)
```

Şekil 2.3.5: VGG16 modeli için Eğitim verilerine uygulanan veri artırma işlemleri

Eğitim tamamlandıktan sonra modelin başarımı doğruluk ve kayıp grafikleri ile analiz edilmiş; ayrıca karmaşıklık matrisi (confusion matrix) ve sınıflandırma raporu (classification report) çıktıları ile detaylı başarım değerlendirmesi yapılmıştır. Model, özellikle "Healthy" ve "Common Rust" sınıflarında yüksek doğruluk ve F1-score değerleri elde etmiştir.

2.3.3 InceptionV3

InceptionV3, Google tarafından geliştirilen ve evrişimli sinir ağı mimarileri arasında derinliği ve hesaplama verimliliğini dengeleyen bir modeldir. Model, farklı boyutlardaki filtrelerin aynı anda uygulandığı Inception blokları sayesinde giriş görüntüsünden çok yönlü öznitelikler çıkarmayı başarır. Bu yapı, klasik konvolüsyon katmanlarına kıyasla daha geniş bağlamda ve çeşitli ölçeklerde bilgi yakalanmasını sağlar [48].

Modelin özgün tasarımında farklı boyutlardaki konvolüsyon filtreleri (örneğin, 1×1, 3×3, 5×5) paralel olarak uygulanır ve ardından bu çoklu çıktılar birleştirilir. Bu çoklu yolların bir araya getirilmesi sayesinde, hem düşük seviye hem de yüksek seviye özelliklerin yakalanması mümkün hâle gelir.

Bu çalışmada, InceptionV3 modeli transfer öğrenme yöntemiyle kullanılmış, yani ImageNet üzerinde önceden eğitilmiş ağırlıklar kullanılarak modelin temel katmanları alınmıştır. Modelin en üstteki sınıflandırma katmanı çıkartılmış ve sınıflandırma kısmı, veri kümesindeki sınıflara özel olacak şekilde yeniden tasarlanmıştır. InceptionV3 modeli Şekil 2.3.6'da gösterildiği gibi ImageNet ağırlıkları ile yüklenmiş ve sınıflandırma katmanı hariç tutulmuştur.

```
# InceptionV3 modelini önceden eğitilmiş ImageNet ağırlıkları ile yükle
# Üst sınıflandırma katmanı dahil edilmez (include_top=False)
# Giriş boyutu 224x224x3 olarak tanımlanır
base_model = InceptionV3(
    weights="imagenet",
    include_top=False,
    input_tensor=Input(shape=(224, 224, 3))
)
```

Şekil 2.3.6: InceptionV3 modelinin yüklenmesi ve giriş yapılandırması

Yukarıdaki kodda, include_top=False parametresi ile modelin orijinal sınıflandırma katmanı çıkarılmış ve giriş görüntüleri 224×224 piksel boyutuna göre ayarlanmıştır.

Modelin temel katmanları dondurularak sadece son katmanlar eğitime açılmıştır. Bu işlem, Şekil 2.3.7'deki kodla gerçekleştirilmiştir:

```
# Tüm katmanları eğitilemez (donmuş) hale getirerek transfer öğrenme yapılacak temeli sabitler
for layer in base_model.layers:
    layer.trainable = False
```

Şekil 2.3.7: Tüm temel model katmanlarının eğitilemez (donmuş) hâle getirilmesi

Son olarak, çıkarım yapacak sınıflandırıcı katmanlar eklenmiştir. Bu yeni katmanlar Şekil 2.3.8'de gösterildiği gibi modelin özgün veri kümesine uyum sağlamasını kolaylaştırmak amacıyla oluşturulmuştur:

```
# Temel modelin çıkışı düzleştirilerek tam bağlantılı katmanlara aktarılır
x = Flatten()(base_model.output)
# 512 nöronlu ReLU aktivasyonlu dense katman
```

```
x = Dense(512, activation="relu")(x)
# %40 oranında dropout uygulanarak aşırı öğrenme riski azaltılır
x = Dropout(0.4)(x)
# 4 sınıf için softmax aktivasyonlu çıkış katmanı
output = Dense(4, activation="softmax")(x)
# Giriş ve çıkışı tanımlayarak yeni model oluşturulur
model = Model(inputs=base_model.input, outputs=output)
```

Şekil 2.3.8: Tam bağlantılı katmanlar ve modelin son yapısının oluşturulması

Model, Adam optimizasyon algoritması ile eğitilmiş ve categorical_crossentropy kayıp fonksiyonu kullanılmıştır. Bu işlem, Şekil 2.3.9'da gösterilen kod ile tamamlanmıştır:

```
# Model derleniyor: optimizer olarak Adam, düşük öğrenme oranı ile
# Kayıp fonksiyonu: categorical_crossentropy (çok sınıflı sınıflandırma i
    çin)
# Performans metriği olarak accuracy (doğruluk) kullanılır
model.compile(optimizer=Adam(learning_rate=0.00005),
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

Şekil 2.3.9: Modelin düşük öğrenme oranı ile derlenmesi

Model, toplam 10 tur (epoch) boyunca eğitilmiş ve önceden eğitilmiş katmanlar dondurularak yalnızca son sınıflandırma katmanları yeniden eğitilmiştir. Eğitim sürecinde erken durdurma (early stopping) yöntemi uygulanmamış, yalnızca doğruluk (accuracy) metriği ile değerlendirme yapılmıştır. Eğitim sonunda model, özellikle Healthy ve Common Rust sınıflarında yüksek doğruluk göstermiştir.

2.3.4 MobileNetV2

MobileNet mimarisi, mobil cihazlar ve gömülü sistemler gibi sınırlı donanımsal kaynaklara sahip ortamlarda derin öğrenme modellerinin verimli şekilde çalışmasını sağlamak amacıyla geliştirilmiş hafif yapılı bir evrişimsel sinir ağı mimarisidir. Bu yapı, geleneksel evrişim işlemleri yerine "derin ayrık evrişim" (depthwise separable convolutions) kullanarak modelin parametre sayısını ve hesaplama maliyetini önemli ölçüde azaltır. MobileNetV2, önceki sürüm olan

MobileNetV1'in üzerine eklenen yapısal iyileştirmelerle geliştirilmiştir. Bu sürümde, "darboğaz (bottleneck)" blokları ve "atlamalı bağlantılar (skip connections)" gibi mekanizmalar yer alır. Bu sayede model, daha az parametreyle daha iyi genelleme başarımı sergileyebilir ve hem doğruluk hem de işlem hızı açısından dengeli bir yapı sunar [49] [50].

Bu çalışmada MobileNetV2 mimarisi, transfer öğrenme yöntemiyle bitki hastalıklarının sınıflandırılmasında kullanılmıştır. Model, ImageNet veri kümesi üzerinde önceden eğitilmiş ağırlıklarla başlatılmış ve include_top=False parametresiyle üst sınıflandırıcı katmanları çıkarılmıştır. Böylece yalnızca temel evrişimsel katmanlar kullanılarak, mısır bitkisine özgü sınıflandırma görevine uygun yeni tam bağlantılı katmanlar eklenmiştir.

Giriş görselleri, MobileNetV2 mimarisinin gerektirdiği şekilde 224×224 piksel boyutuna yeniden ölçeklendirilmiş ve piksel değerleri 0 ile 1 aralığına normalize edilmiştir. Eğitim verisi üzerinde veri artırma (data augmentation) teknikleri uygulanarak modelin genelleme yeteneği artırılmıştır. Bu kapsamda; döndürme (rotation), yakınlaştırma (zoom), yatay ve dikey kaydırma (shift), parlaklık değişimi (brightness) ve yatay çevirme (horizontal flip) gibi işlemler kullanılmıştır.

Modelin çıkış katmanı, dört sınıflı mısır yaprak hastalığı veri setine uygun şekilde yapılandırılmıştır. Çıkış katmanında softmax aktivasyon fonksiyonu kullanılan 4 nöronlu bir Dense katman yer almakta; bu yapı, "Cercospora Leaf Spot", "Common Rust", "Northern Leaf Blight" ve "Healthy" sınıflarını ayırt etmek üzere tasarlanmıştır. Kayıp fonksiyonu olarak çok sınıflı sınıflandırma için uygun olan *categorical_crossentropy* seçilmiştir.

Yeni eklenen sınıflandırıcı katmanlar sırasıyla Flatten, 256 ve 128 nöronlu Dense katmanlar ile her iki katman arasında %40 oranında Dropout katmanlarını içermektedir. Modelin eğitimi düşük öğrenme oranına (0.00005) sahip Adam optimizasyon algoritmasıyla gerçekleştirilmiştir. Ayrıca EarlyStopping tekniği kullanılarak, doğrulama başarımı belli bir süre boyunca artmazsa eğitim süreci erken sonlandırılmış ve overfitting riski azaltılmıştır.

Elde edilen model, 10 tur (epoch) boyunca eğitilmiş ve eğitim sürecinin sonunda doğruluk (accuracy), karmaşıklık matrisi (confusion matrix) ve sınıflandırma raporu (precision, recall, F1-score) gibi performans metrikleri ile değerlendirilmiştir. MobileNetV2'nin sunduğu hafiflik,

hız ve doğruluk dengesi, bu çalışmada hedeflenen sınıflandırma görevine oldukça uygun bir çözüm sunmuştur [51] [52].

MobileNetV2 modeli, düşük parametre sayısı ve yüksek doğruluk oranı ile sunduğu verimli başarım sayesinde bu çalışma kapsamında uygulanan tüm modeller arasında en dengeli ve başarılı sonuçları vermiştir. Bu nedenle, model üzerinde çeşitli geliştirme yöntemleri uygulanarak daha iyi sonuçlar elde edilmesi hedeflenmiştir. Uygulanan geliştirme sürecinde üç temel teknikten yararlanılmıştır: ince ayar (fine-tuning), hiperparametre iyileştirmeleri ve gelişmiş veri artırma (data augmentation). Bu yaklaşımlar sayesinde modelin öğrenme yetisi artırılmış ve genelleme başarısı iyileştirilmiştir.

İnce ayar (fine-tuning), önceden eğitilmiş bir derin öğrenme modelinin belirli katmanlarının yeniden eğitilmesini sağlayarak modele, çalışılan veri kümesine özgü bilgileri daha iyi öğrenme imkânı tanır. Bu çalışmada, MobileNetV2 modelinin son 30 katmanı tekrar eğitilebilir hâle getirilmiş ve model, 0.00001 öğrenme oranı ile eğitilmiştir. Böylece, mısır yaprağı hastalıklarına özgü görsel özniteliklerin daha etkin şekilde öğrenilmesi hedeflenmiştir.

Model, 15 tur (epoch) boyunca eğitilmiş ve early stopping yöntemiyle overfitting riski azaltılmıştır. Eğitim sonunda, doğruluk değerleri aşağıdaki şekilde elde edilmiştir: Eğitim doğruluğu: %95,71 Doğrulama doğruluğu: %96,72 “Common Rust” ve “Healthy” sınıflarında F1-score: %99 Confusion matrix analizine göre, “Northern Leaf Blight” sınıfında diğer sınıflara göre daha fazla yanlış sınıflandırma gözlemlenmiştir. İnce ayar (fine-tuning) için kullanılan Python kodu Şekil 2.3.10’ da verilmiştir.

```python
# MobileNetV2 modelini yükle
base_model = MobileNetV2(weights="imagenet", include_top=False,
    input_tensor=Input(shape=(224, 224, 3)))
# Tüm katmanları dondur
for layer in base_model.layers:
    layer.trainable = False
# Son 30 katmanı yeniden eğitilebilir hale getir
for layer in base_model.layers[-30:]:
    layer.trainable = True
```

Şekil 2.3.10: MobileNetV2 modeline son 30 katmanda ince ayar (fine-tuning) uygulanarak

sınıflandırıcı katmanlar ekleme kod kümesi.

Modelin başarımını artırmak amacıyla bazı hiperparametreler, önceki deneyimlere ve eğitim çıktılarından elde edilen gözlemlere dayanarak sezgisel bir yaklaşımla yeniden ayarlanmıştır. Bu kapsamda tam bağlantı (fully connected) katman boyutları 512'den 256'ya ve 256'dan 128'e düşürülmüş, terk (dropout) oranı %40 olarak belirlenmiştir. Ayrıca, öğrenme oranı 0.00001'den 0.00005'e çıkarılmış ve tur (epoch) sayısı 15 olarak yeniden düzenlenmiştir. Bu iyileştirmelerin sonucunda modelin doğruluk değeri %96'nın üzerine çıkmış, F1-score ortalaması 0.96 olarak elde edilmiştir. Özellikle "Healthy" ve "Common Rust" sınıflarında doğruluk oranları %100'e yaklaşmış, eğitim ve doğrulama eğrilerinde dengeli bir ilerleme kaydedilmiş, aşırı ezber (overfitting) eğilimine rastlanmamıştır. Modelin çıkış katmanları Şekil 2.3.11'de gösterildiği gibi yeniden yapılandırılmıştır.

```
x = Flatten()(base_model.output)
x = Dense(256, activation='relu')(x)
x = Dropout(0.4)(x)
x = Dense(128, activation='relu')(x)
x = Dropout(0.4)(x)
outputs = Dense(4, activation='softmax')(x)
model = Model(inputs=base_model.input, outputs=outputs)
```

Şekil 2.3.11: İnce ayar (fine-tuning) sonrası yeniden yapılandırılan tam bağlantı katmanları ekleme kod kümesi

Modelin genelleme kabiliyetini artırmak ve aşırı öğrenme (overfitting) riskini azaltmak amacıyla, yalnızca eğitim verileri üzerinde çeşitli veri artırma (data augmentation) teknikleri uygulanmıştır. Bu kapsamda, görüntülere 40 dereceye kadar rastgele döndürme, %30 oranında yakınlaştırma ve uzaklaştırma, %20 oranında yatay ve dikey kaydırma, kesme etkisi (shear), parlaklık değerlerini değiştirme ve yatay çevirme gibi dönüşümler uygulanmıştır. Doğrulama ve test veri setlerinde ise yalnızca normalize işlemi (rescale=1./255) gerçekleştirilmiş, herhangi bir artırma tekniği kullanılmamıştır. Bu stratejilerin sonucunda modelin genel doğruluk değeri %97 seviyesine ulaşmış; precision, recall ve F1-score gibi performans metrikleri 0.93 ile 1.00 aralığında değerler göstermiştir. Özellikle dört sınıfta da dengeli ve yüksek başarı elde edilmiştir. Uygulanan veri artırma stratejileri Şekil 2.3.12'de gösterilmektedir.

```
# Eğitim verileri için veri artırma (data augmentation) işlemleri tanımlanıyor
train_datagen = ImageDataGenerator(
    rescale=1./255,                      # Piksel değerlerini 0 1 aralığına ölçekle
    rotation_range=40,                   # 40 dereceye kadar rastgele döndürme
    zoom_range=0.3,                      # %30 oranında yakınlaştırma/uzaklaştırma
    width_shift_range=0.2,               # %20 oranında yatay kaydırma
    height_shift_range=0.2,              # %20 oranında dikey kaydırma
    shear_range=0.2,                     # Kesme (shear) etkisi
    brightness_range=[0.7, 1.3],         # Parlaklık değişimi (0.7 ile 1.3 arası)
    horizontal_flip=True                 # Görüntüleri yatay eksende çevir)
```

Şekil 2.3.12: Eğitim verilerine uygulanan veri artırma işlemleri

3 BULGULAR

Bu bölümde, çalışma kapsamında kullanılan veri seti üzerinde uygulanan derin öğrenme yöntemlerinden elde edilen sonuçlara ayrıntılı olarak yer verilmiştir. Sınıflandırma amacıyla PlantVillage veri seti kullanılmış ve bu veri seti üzerinde MobileNetV2, VGG16 ve InceptionV3 olmak üzere üç farklı önceden eğitilmiş model denenmiştir. Her bir modelin başarımı, sınıflara göre doğruluk (accuracy), kesinlik (precision), duyarlılık (recall) ve F1 skoru (F1-score) gibi temel sınıflandırma metrikleri üzerinden değerlendirilmiştir. Elde edilen bu değerlendirme sonuçları, ilgili metrikler çerçevesinde Tablo 3.1'den Tablo 3.4'e kadar olan tablolarda, VGG16, InceptionV3 ve MobileNetV2 modelleri için ayrı ayrı ve karşılaştırmalı şekilde sunulmaktadır. Bu sayede, her modelin sınıflandırma başarımı daha kapsamlı biçimde analiz edilmiştir.

Tablo 3.1: VGG16 modeli başarım metrikleri çizelgesi

Sınıf	**Precision**	**Recall**	**F1-Score**	**Support**
Cercospora Leaf Spot	0.91	1.00	0.95	41
Common Rust	0.98	1.00	0.99	48
Healthy	0.98	0.98	0.98	47
Northern Leaf Blight	1.00	0.90	0.95	48
Accuracy	—	—	**0.97**	184
Macro avg	0.97	0.97	0.97	184
Weighted avg	0.97	0.97	0.97	184

VGG16 modeli, mısır yaprağı hastalıklarının sınıflandırılmasında genel doğruluk (accuracy) oranı %97 olarak yüksek bir başarı göstermiştir. Model, özellikle Common Rust ve Healthy sınıflarında %98–%99 düzeyinde F1-score ile oldukça etkili sonuçlar üretmiştir. Cercospora Leaf Spot sınıfında %100 duyarlılık (recall) sağlanmış; ancak %91 kesinlik (precision) değeri, bazı örneklerin bu sınıfla karıştırıldığını göstermektedir. Northern Leaf Blight sınıfında ise %100 kesinlik ve %90 duyarlılık ile dengeli bir performans sergilenmiştir. Makro ve ağırlıklı ortalama F1-score değerleri 0.97 olup, modelin tüm sınıflar arasında dengeli ve başarılı bir genelleme yeteneğine sahip olduğunu göstermektedir. Bu sonuçlar, transfer öğrenme yöntemiyle eğitilen VGG16 modelinin mısır yaprağı hastalıklarının doğru şekilde ayrıştırılmasında etkili bir çözüm sunduğunu ortaya koymaktadır.

Tablo 3.2: InceptionV3 modeli başarım metrikleri çizelgesi

Sınıf	Precision	Recall	F1-Score	Support
Cercospora Leaf Spot	0.92	0.80	0.86	41
Common Rust	1.00	0.98	0.99	48
Healthy	0.98	0.98	0.98	47
Northern Leaf Blight	0.83	0.94	0.88	48
Accuracy	—	—	**0.93**	184
Macro avg	0.93	0.93	0.93	184
Weighted avg	0.93	0.93	0.93	184

InceptionV3 modeli, mısır yaprağı hastalıklarının sınıflandırılması görevinde genel doğruluk (accuracy) açısından %93 başarı elde etmiştir. Modelin özellikle Common Rust ve Healthy sınıflarında yüksek bir performans sergilediği görülmektedir. Her iki sınıf için F1-score değerleri sırasıyla 0.99 ve 0.98 olup, bu sınıfların doğru sınıflandırılmasında güçlü bir genel öğrenme sağlandığını göstermektedir.

Cercospora Leaf Spot sınıfında ise precision (%92) yüksek olmasına rağmen recall değeri %80 ile diğer sınıflara kıyasla düşüktür; bu da modelin bazı örnekleri bu sınıfa ait olarak tanımlamakta zorlandığını göstermektedir. Buna karşılık, Northern Leaf Blight sınıfında %94 recall ve %83 precision değeri ile dengeli fakat nispeten düşük bir F1-score (0.88) elde edilmiştir.

Makro ve ağırlıklı ortalama (macro avg, weighted avg) değerleri incelendiğinde tüm metriklerde 0.93 seviyesinde sonuçlar gözlemlenmektedir. Bu da modelin sınıflar arasında genel olarak dengeli bir başarı sağladığını, ancak bazı sınıflarda (özellikle Cercospora Leaf Spot) iyileştirme çalışmaları yapmak gerektiğini göstermektedir.

Tablo 3.3: MobileNetV2 modeli başarım metrikleri çizelgesi

Sınıf	Precision	Recall	F1-Score	Support
Cercospora Leaf Spot	0.95	0.95	0.95	41
Common Rust	0.98	1.00	0.99	48
Healthy	0.98	1.00	0.99	47
Northern Leaf Blight	0.96	0.92	0.94	48
Accuracy	—	—	**0.97**	184
Macro avg	0.97	0.97	0.97	184
Weighted avg	0.97	0.97	0.97	184

MobileNetV2 modeli, mısır yaprağı hastalıklarının sınıflandırılmasında genel doğruluk açısından %97 gibi yüksek bir başarı göstermiştir. Özellikle Common Rust ve Healthy sınıflarında elde edilen %99 F1-score değerleri, modelin bu sınıfları oldukça başarılı şekilde ayırt edebildiğini ortaya koymaktadır. Cercospora Leaf Spot ve Northern Leaf Blight sınıflarında ise sırasıyla %95 ve %94 F1-score değerleri elde edilmiş olup, bu sınıflarda da tatmin edici sonuçlara ulaşılmıştır. Tüm sınıflar için dengeli bir başarım sergileyen model, hem macro hem de weighted ortalamada %97 F1-score ile genelleme yeteneğinin güçlü olduğunu kanıtlamaktadır. Bu sonuçlar, MobileNetV2 modelinin sınıf dengesini koruyarak yüksek doğrulukla tahminler yapabildiğini ve gerçek dünya uygulamaları için uygun bir çözüm sunduğunu göstermektedir.

Tablo 3.4: MobileNetV2 modeli başarım metrikleri çizelgesi

Sınıf	**Precision**	**Recall**	**F1-Score**	**Support**
Cercospora Leaf Spot	0.94	0.92	0.93	341
Common Rust	0.99	1.00	0.99	348
Healthy	1.00	1.00	1.00	347
Northern Leaf Blight	0.93	0.94	0.94	348
Accuracy	—	—	**0.97**	1384
Macro avg	0.97	0.97	0.97	1384
Weighted avg	0.97	0.97	0.97	1384

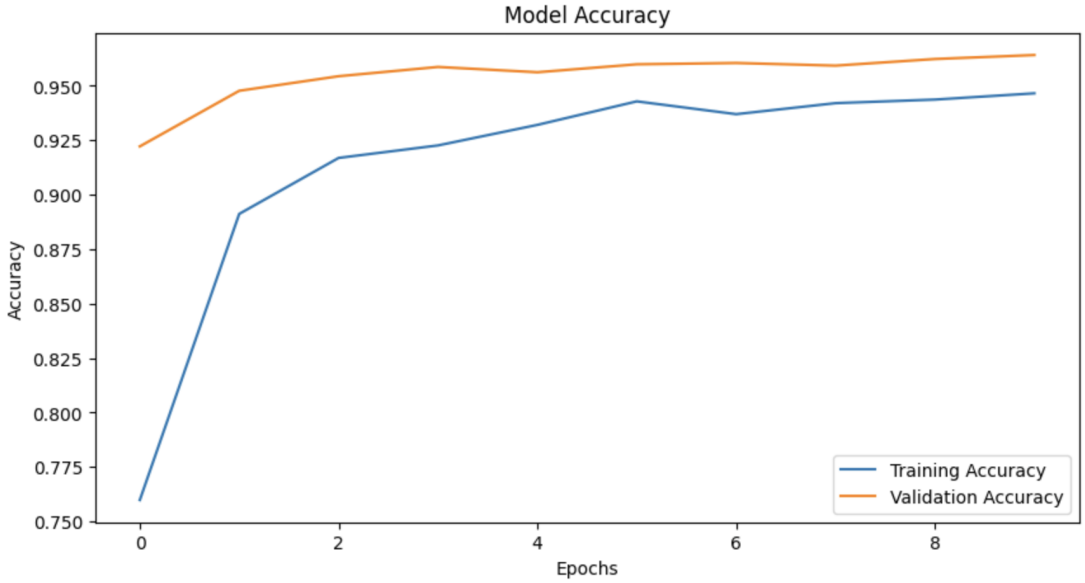
Veri setindeki test verilerinin dengesiz dağılımı, sınıflar arası başarım karşılaştırmalarını olumsuz etkileme potansiyeline sahipti. Bu nedenle test verileri manuel olarak eşit dağıtılacak şekilde yeniden düzenlenmiş ve bu iyileştirme sonrasında modelin başarımı güncellenmiştir. MobileNetV2 modeli, düzenlenmiş veri dağılımıyla birlikte %97 genel doğruluk (accuracy) değeriyle güçlü bir sınıflandırma başarısı sergilemiştir.

Özellikle Healthy sınıfında %100 precision, recall ve F1-score değerlerine ulaşılması, modelin bu sınıfı kusursuz şekilde ayırt edebildiğini göstermektedir. Common Rust sınıfında da %99 F1-score ile oldukça yüksek başarı sağlanmıştır. Northern Leaf Blight ve Cercospora Leaf Spot sınıflarında sırasıyla %94 ve %93 F1-score elde edilmiştir. Bu değerler, modelin sınıflar arasında dengeli ve güvenilir bir başarım gösterdiğini ortaya koymaktadır.

Macro ve weighted ortalama metriklerinin her ikisinin de %97 düzeyinde gerçekleşmiş olması, modelin genelleme yeteneğinin yüksek olduğunu ve tüm sınıflara eşit duyarlılıkla yak-

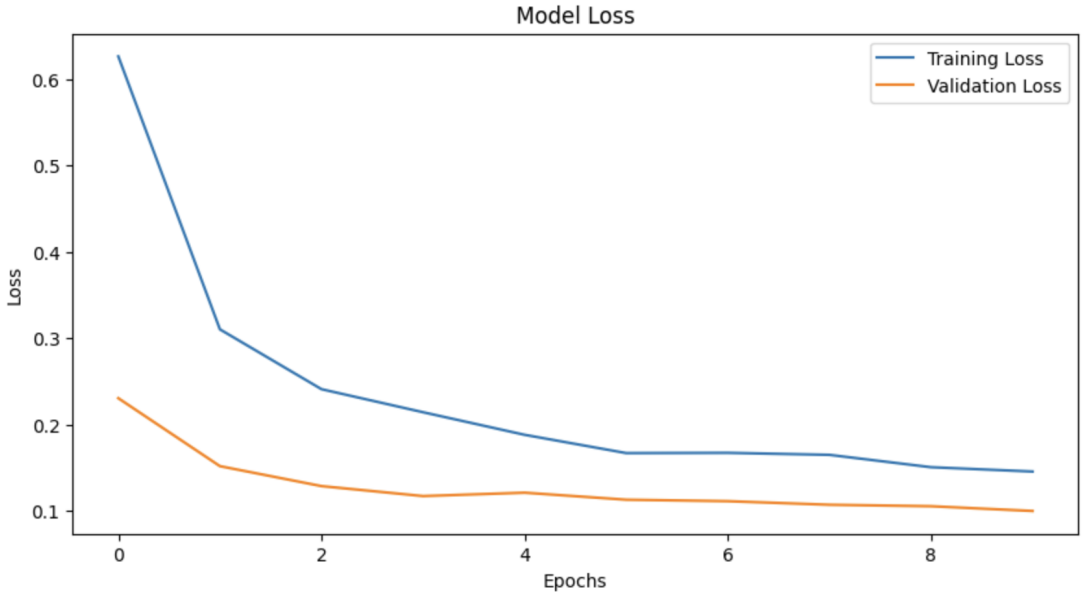
laşabildiğini doğrulamaktadır. Sonuç olarak, dengelenmiş test veri dağılımıyla birlikte MobileNetV2 modeli, mısır yaprağı hastalıklarının sınıflandırılmasında sağlam ve tutarlı bir çözüm sunmaktadır.

Modellerin başarımını daha ayrıntılı değerlendirebilmek adına, eğitim ve doğrulama sürecine ait doğruluk ve kayıp grafiklerine de bu bölümde yer verilmiştir. Bu grafikler, her bir modelin öğrenme sürecinde ne ölçüde başarılı olduğunu görsel olarak analiz etme imkânı sunmaktadır. Doğruluk grafikleri, modelin her tur (epoch) boyunca elde ettiği başarı oranlarını ortaya koyarken; kayıp grafikleri ise modelin öğrenme sırasında yaptığı hataların zamanla nasıl azaldığını göstermektedir. Bu sayede, modellerin eğitim ve doğrulama başarımları karşılaştırmalı olarak değerlendirilebilmekte, öğrenme eğrileri arasındaki farklılıklar net biçimde gözlemlenebilmektedir. Bu grafikler Şekil 3.1'den Şekil 3.8'e kadar sırasıyla VGG16, InceptionV3,MobileNetV2 transfer öğrenim modelleri için sunulmuştur.



Şekil 3.1: VGG16 modelinin eğitim sürecinde doğruluk değişimi

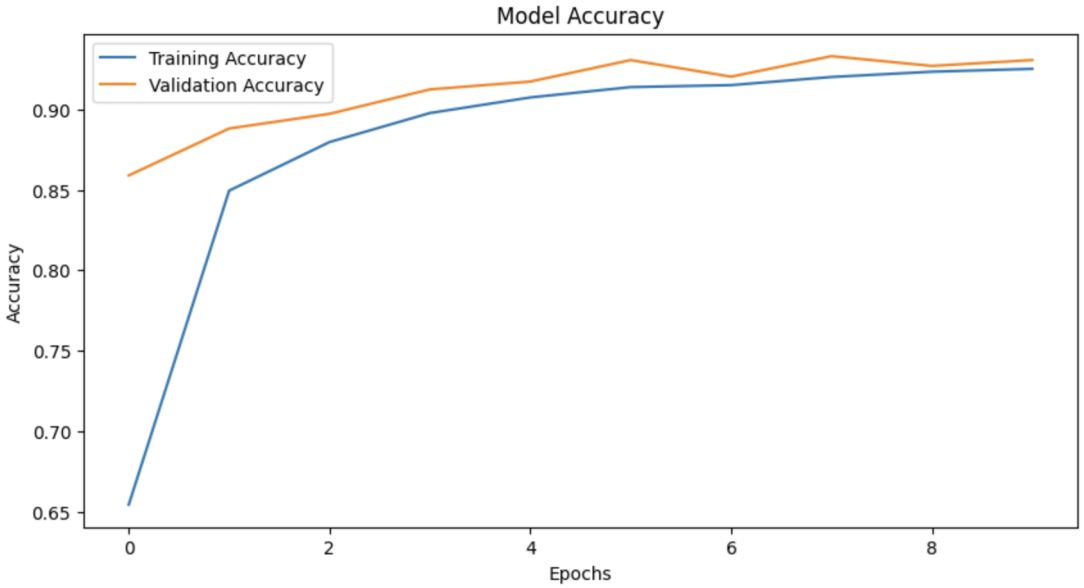
VGG16 doğruluk grafiğine bakıldığında, eğitim doğruluğunun epoch sayısı arttıkça düzenli bir şekilde yükseldiği ve yaklaşık %93 seviyesine ulaştığı görülmektedir. Doğrulama doğruluğu ise benzer bir artış göstermiş, ancak eğitim doğruluğuna kıyasla daha yüksek bir seviyede (%95) sabitlenmiştir. Bu durum, modelin doğrulama verisi üzerindeki performansının oldukça başarılı olduğunu ve aşırı öğrenme (overfitting) riskinin düşük olduğunu göstermektedir. Eğitim ve doğrulama eğrileri arasındaki farkın az olması, modelin genelleme kabiliyetinin yeterli düzeyde olduğunu ortaya koymaktadır.



Şekil 3.2: VGG16 modelinin eğitim sürecinde kayıp değişimi

VGG16 kayıp grafiğine bakıldığında, eğitim kaybının epoch sayısı arttıkça düzenli şekilde azaldığı ve yaklaşık 0.15 seviyesine kadar düştüğü gözlemlenmektedir. Benzer şekilde, doğrulama kaybı da dengeli bir azalma göstermiş ve düşük seviyelerde seyretmiştir.

Eğitim ve doğrulama kayıplarının birbirine yakın ilerlemesi, modelin eğitim süreci boyunca istikrarlı bir öğrenme gerçekleştirdiğini ve doğrulama verisinde ani sapmalar göstermediğini ortaya koymaktadır. Bu da modelin yalnızca eğitim verisine değil, aynı zamanda yeni veriler üzerinde de yeterli başarı sergileyebildiğini göstermektedir.

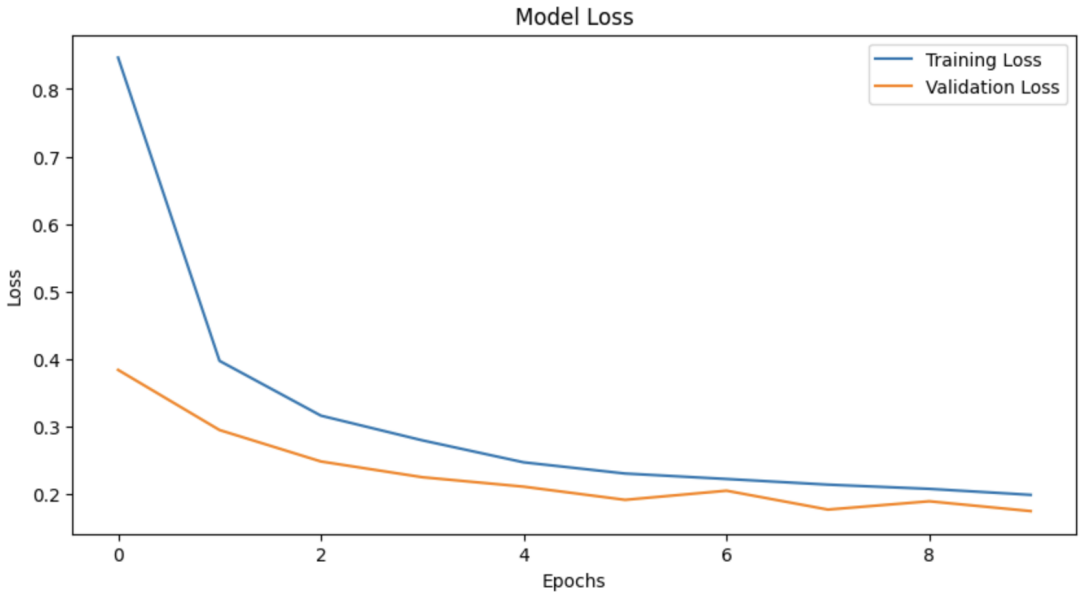


Şekil 3.3: InceptionV3 modelinin eğitim sürecinde doğruluk değişimi

InceptionV3 doğruluk grafiğine bakıldığında, eğitim doğruluğunun epoch sayısı arttıkça düzenli bir şekilde yükseldiği ve yaklaşık %93 seviyesinde sabitlendiği görülmektedir. Doğrulama doğruluğu ise eğitim doğruluğunun üzerinde seyretmiş ve %95 civarında dengeli bir performans sergilemiştir.

Doğruluk eğrilerinin birbirine yakın ilerlemesi, modelin hem eğitim hem de doğrulama verileri üzerinde tutarlı bir başarı yakaladığını göstermektedir. Ayrıca doğrulama doğruluğunda dalgalanma olmaması, modelin öğrenme sürecinin dengeli ilerlediğini ve yeni verilere karşı istikrarlı sonuçlar üretebildiğini göstermektedir.

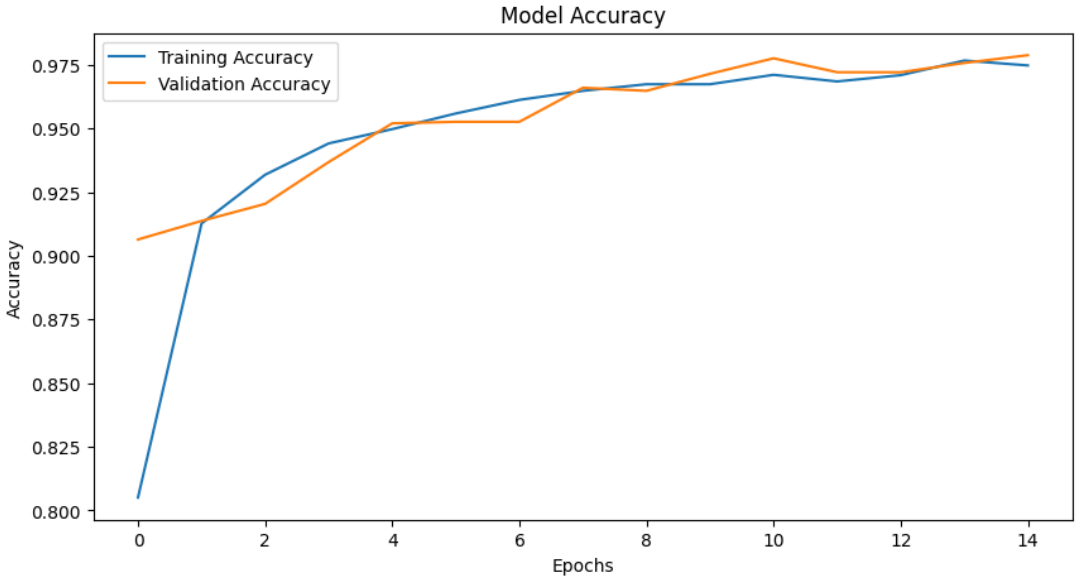
Bu grafik, InceptionV3 modelinin yalnızca eğitim verisini ezberlemeden, genel desenleri başarılı bir şekilde öğrenebildiğine işaret etmektedir.



Şekil 3.4: InceptionV3 modelinin eğitim sürecinde kayıp değişimi

InceptionV3 kayıp grafiğine bakıldığında, eğitim ve doğrulama kayıplarının epoch sayısı arttıkça düzenli bir şekilde azaldığı görülmektedir. Eğitim kaybı yaklaşık 0.9 seviyesinden başlayarak 0.15 düzeyine kadar düşerken, doğrulama kaybı da daha düşük ve dengeli bir seyir izlemiştir.

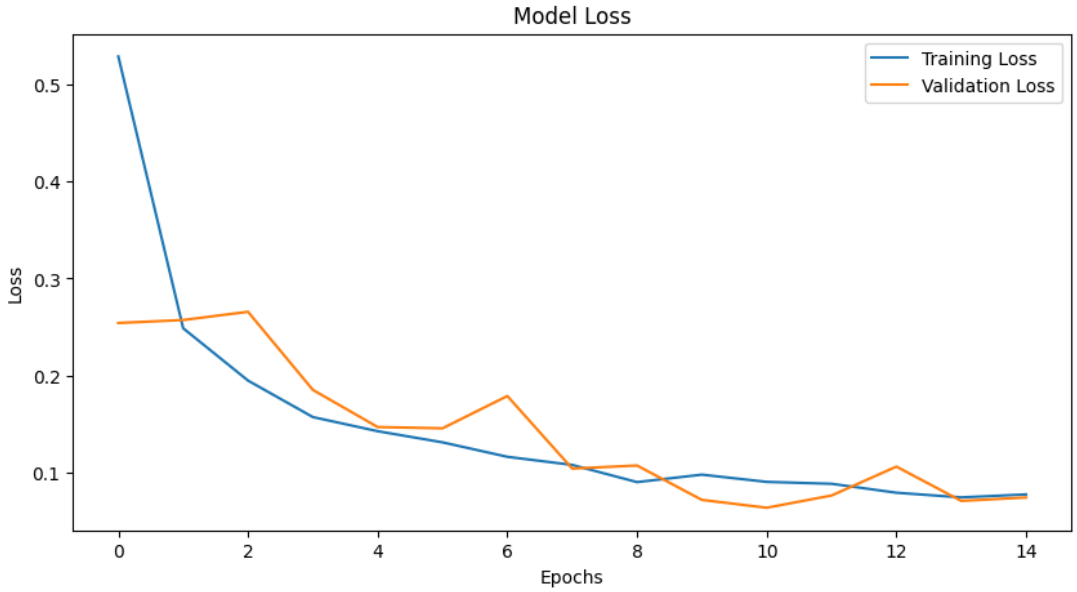
Her iki eğrinin de paralel şekilde ilerlemesi, modelin öğrenme sürecinin dengeli gerçekleştiğini ve eğitim verisine aşırı bağımlı kalmadan genel başarıyı sürdürebildiğini göstermektedir.



Şekil 3.5: MobileNetV2 modelinin eğitim sürecinde doğruluk değişimi

Şekil 3.5'teki grafik incelendiğinde, modelin ilk birkaç epoch'ta doğruluğunu hızla artırdığı ve yaklaşık 4. turdan itibaren hem eğitim hem de doğrulama doğruluklarının %95 seviyesinin üzerine çıktığı gözlemlenmektedir. Eğitim süreci boyunca doğruluk değerleri artmaya devam etmiş ve 13. tur (epoch) itibarıyla eğitim doğruluğu %97'nin üzerine ulaşmıştır.

Grafik genel olarak, modelin öğrenme sürecinde istikrarlı bir başarı yakaladığını ve veri seti üzerindeki sınıflandırma görevini yüksek doğrulukla yerine getirdiğini göstermektedir.



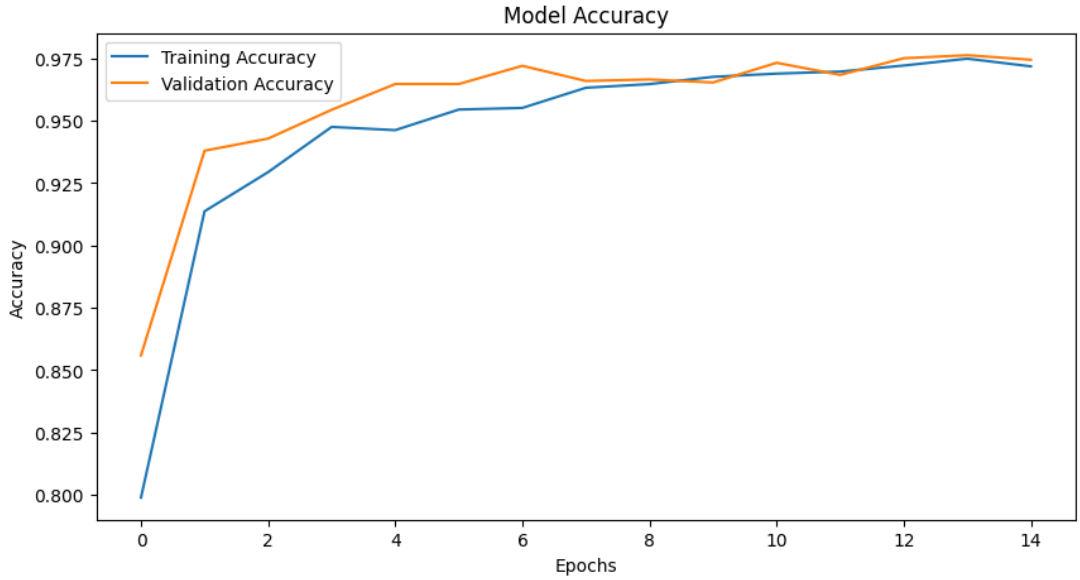
Şekil 3.6: MobileNetV2 modelinin eğitim sürecinde kayıp değişimi

Şekil 3.6'da gösterilen MobileNetV2 kayıp grafiği, modelin eğitim (training) ve doğrulama

(validation) verileri üzerindeki kayıplarının turlara göre değişimini yansıtmaktadır. Grafik incelendiğinde, ilk turlardan itibaren hem eğitim hem de doğrulama kayıplarında belirgin bir azalma olduğu gözlemlenmektedir. Bu durum, modelin her yeni tur (epoch) ile birlikte daha iyi öğrendiğini ve hatalarını azalttığını göstermektedir.

Eğitim kaybı istikrarlı şekilde düşerek yaklaşık 0.05 seviyelerine kadar gerilerken, doğrulama kaybı zaman zaman küçük dalgalanmalar gösterse de genel eğilim olarak düşüş yönlüdür. Son turlarda eğitim ve doğrulama kayıplarının birbirine yaklaşması, modelin aşırı öğrenmeden kaçındığını ve eğitim sürecinin dengeli ilerlediğini ortaya koymaktadır.

Bu grafik, modelin öğrenme sürecinin başarılı geçtiğini ve genelleme kabiliyetinin güçlü olduğunu desteklemektedir.

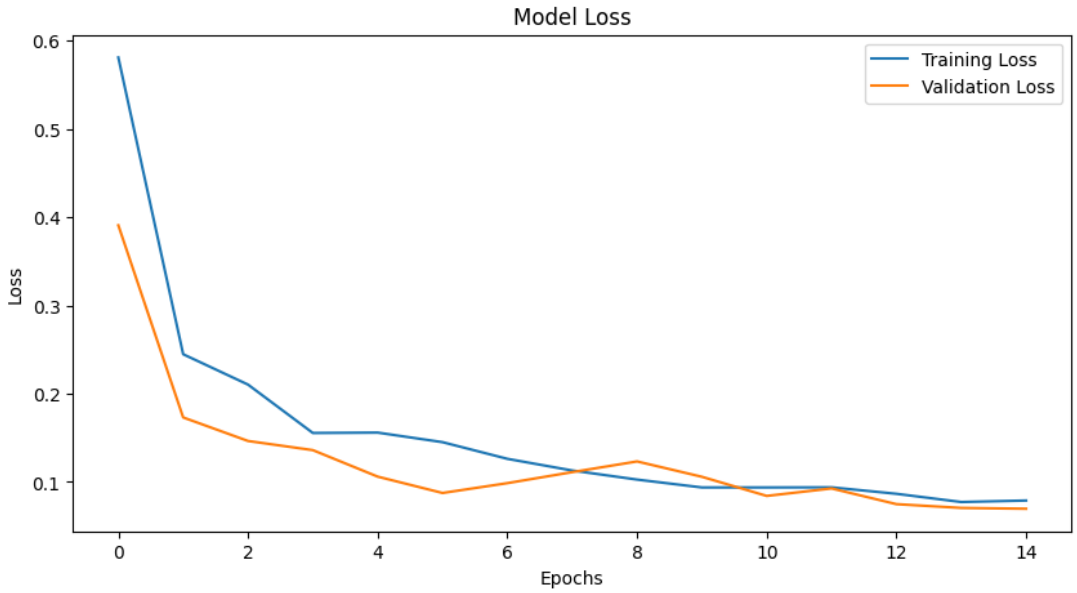


Şekil 3.7: MobileNetV2 veri oranları düzenlendikten sonraki doğruluk grafiği

Şekil 3.7'de sunulan doğruluk grafiği, eğitim ve doğrulama veri oranlarının dengelenmesi sonrasında elde edilen eğitim sürecini yansıtmaktadır. Grafik incelendiğinde, modelin hem eğitim hem de doğrulama verilerinde hızlı ve kararlı bir doğruluk artışı gösterdiği görülmektedir. İlk birkaç turda doğrulama doğruluğu eğitim doğruluğunun üzerinde seyretmiş, bu da modelin genelleme yeteneğinin güçlü olduğunu göstermektedir.

Tur (epoch) sayısı arttıkça her iki doğruluk değeri de %97 seviyelerine yaklaşmış ve grafik dengeli bir şekilde yatayda sabitlenmiştir. Bu durum, modelin hem eğitim verisini başarıyla öğrendiğini hem de aşırı öğrenmeden kaçındığını ortaya koymaktadır. Veri oranlarının dengelen-

mesiyle birlikte modelin performansı daha tutarlı hâle gelmiş ve genelleme kapasitesi artmıştır.



Şekil 3.8: MobileNetV2 veri oranları düzenlendikten sonraki kayıp grafiği

Şekil 3.8'de gösterilen kayıp grafiği, eğitim ve doğrulama süreçlerindeki model kayıplarının tur (epoch) boyunca nasıl değiştiğini göstermektedir. Eğitim sürecinin başlarında her iki kayıpta da hızlı bir azalma gözlemlenmiş, bu da modelin öğrenme sürecine etkili bir şekilde başladığını ortaya koymuştur.

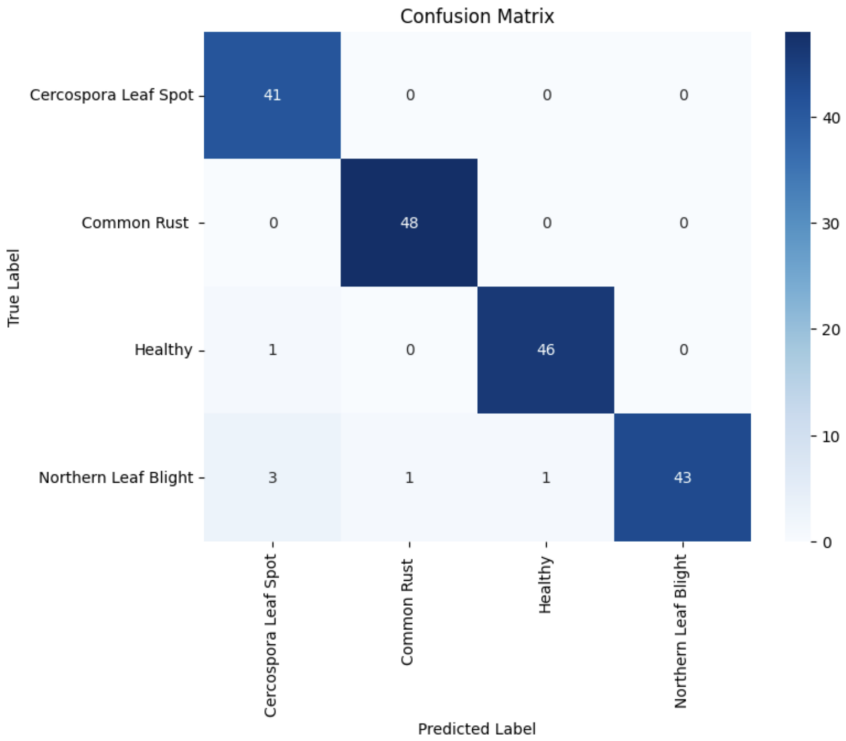
Eğitim ilerledikçe eğitim ve doğrulama kayıpları 0.1 seviyelerine kadar gerilemiş ve birbirine yaklaşarak istikrarlı bir seyir izlemiştir. Bu uyumlu ilerleyiş, modelin aşırı öğrenmeden kaçındığını ve hem eğitim hem de doğrulama verileri üzerinde dengeli bir başarım sergilediğini göstermektedir. Ayrıca, doğrulama kaybındaki dalgalanmaların düşük düzeyde kalması, modelin genelleme yeteneğinin korunduğunu da desteklemektedir.

Sonuç olarak, bu grafik veri oranlarının dengelenmesinin ardından MobileNetV2 modelinin öğrenme sürecinde kararlı, dengeli ve etkili bir şekilde ilerlediğini göstermektedir.

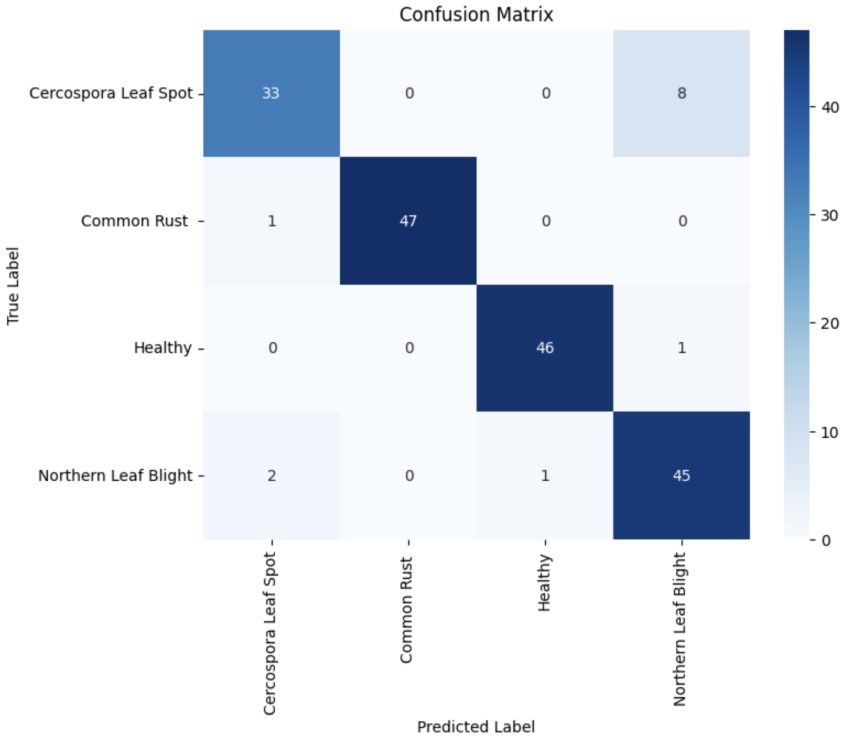
Modellerin sınıflandırma başarımını daha ayrıntılı değerlendirebilmek adına karmaşıklık (confusion) matrisleri analiz edilmiştir. Bu karmaşıklık matrisleri, modellerin test verisi üzerindeki başarımlarına dayanarak oluşturulmuştur. Bu matrisler, her sınıf için doğru tahmin edilen örnekler ile yanlış sınıflandırmaların (yanlış pozitif ve yanlış negatifler) dağılımını açıkça ortaya koymaktadır. Böylece, modellerin hangi sınıflarda daha başarılı olduğu ve hangi sınıflarda hataya daha açık olduğu detaylı biçimde incelenmiştir.

Ayrıca bu matrisler, sadece genel doğruluk metriğiyle gözlemlenemeyen sınıf içi başarım farklılıklarını ortaya koyması bakımından da oldukça değerlidir. Özellikle dengesiz veri dağılımı bulunan sınıflarda, modelin yanılma eğilimleri daha net şekilde görülebilmekte; bu da gelecekte yapılacak iyileştirme çalışmalarına yön vermektedir.

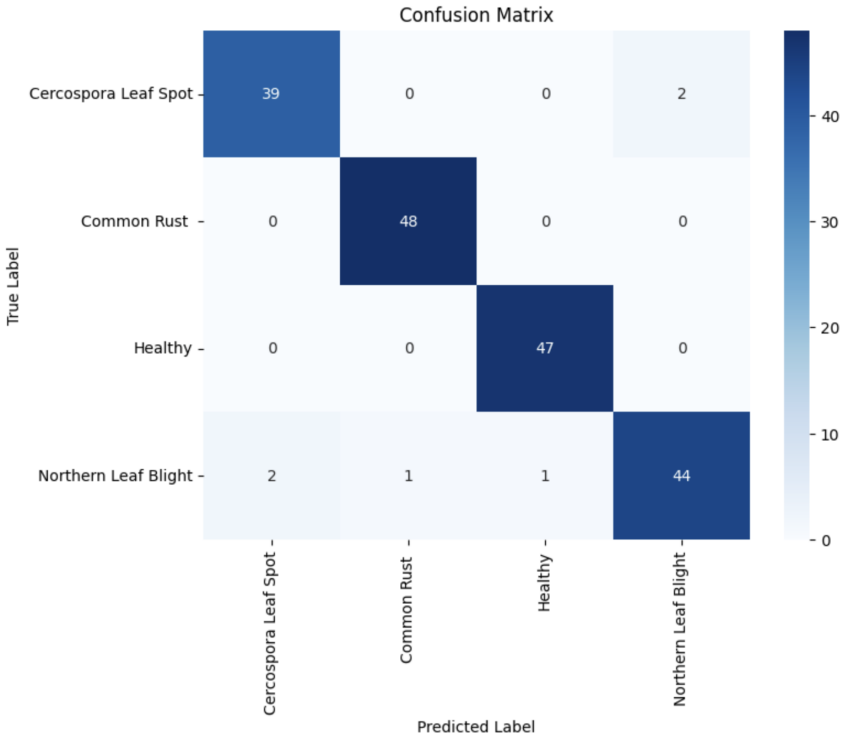
Şekil 3.9–3.12 arasında yer alan görsellerde sırasıyla VGG16, InceptionV3, MobileNetV2 modellerine ait karmaşıklık matrisleri sunulmuştur. Bu görseller, modellerin genel başarımının yanı sıra sınıf bazlı güçlü ve zayıf yönlerinin değerlendirilmesi açısından önemli bir kaynak oluşturmaktadır.



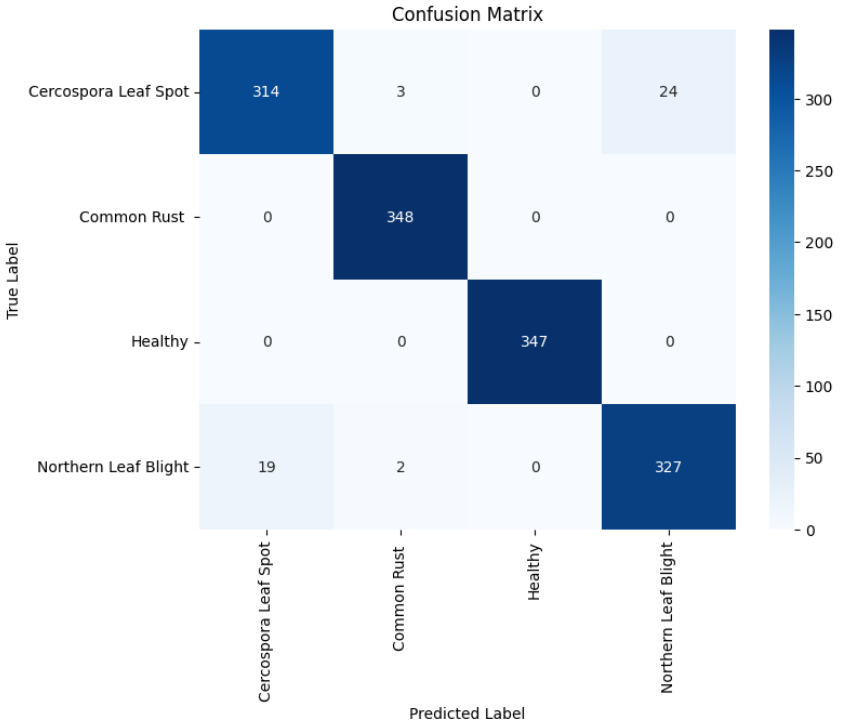
Şekil 3.9: VGG16 karmaşıklık matrisi



Şekil 3.10: InceptionV3 karmaşıklık matrisi



Şekil 3.11: MobileNetV2 karmaşıklık matrisi



Şekil 3.12: MobileNetV2 veri oranları düzenlendikten sonraki karmaşıklık matrisi

Mısır yaprağı hastalıklarını sınıflandırmak üzere kullanılan dört farklı derin öğrenme modeli olan VGG16, InceptionV3, MobileNetV2 ve veri oranları dengelenmiş MobileNetV2 modellerinin başarımları karşılaştırmalı olarak sunulmuştur. Her bir model, PlantVillage veri seti kullanılarak eğitilmiş ve doğruluk, kesinlik (precision), duyarlılık (recall) ve F1-score gibi sınıflandırma performans metrikleri üzerinden değerlendirilmiştir.

Modellerin değerlendirmesi, dört hastalık sınıfına (Cercospora Leaf Spot, Common Rust, Healthy, Northern Leaf Blight) yönelik ayrı ayrı yapılmış ve her modelin genel başarı durumu ayrıca belirtilmiştir. Böylece, transfer öğrenme tabanlı farklı mimarilerin sınıflandırma başarımı üzerindeki etkisi analiz edilmiştir. Buna ek olarak, her bir modelin tahmin başarımı sınıf bazında kıyaslanarak, hangi modelin hangi sınıflarda daha başarılı sonuçlar verdiği gözlemlenmiştir. Bu değerlendirme, modellerin sınıf içi dengesi ve genelleme yetenekleri açısından da önemli ipuçları sağlamaktadır.

Karşılaştırmalı sonuçlar aşağıda Tablo 3.5'te detaylı şekilde sunulmaktadır.

Model	Doğruluk	Sınıf	Kesinlik	Duyarlılık	F1-Score
VGG16	0.97	Cercospora Leaf Spot	0.91	1.00	0.95
		Common Rust	0.98	1.00	0.99
		Healthy	0.98	0.98	0.98
		Northern Leaf Blight	1.00	0.90	0.95
InceptionV3	0.93	Cercospora Leaf Spot	0.92	0.80	0.86
		Common Rust	1.00	0.98	0.99
		Healthy	0.98	0.98	0.98
		Northern Leaf Blight	0.83	0.94	0.88
MobileNetV2	0.97	Cercospora Leaf Spot	0.95	0.95	0.95
		Common Rust	0.98	1.00	0.99
		Healthy	0.98	1.00	0.99
		Northern Leaf Blight	0.96	0.92	0.94
MobileNetV2 (Veri Oranları Düzenlenmiş Hali)	0.97	Cercospora Leaf Spot	0.94	0.92	0.93
		Common Rust	0.99	1.00	0.99
		Healthy	1.00	1.00	1.00
		Northern Leaf Blight	0.93	0.94	0.94

Tablo 3.5: Modellerin genel başarım karşılaştırma tablosu

4 SONUÇLAR

Mısır yaprağı hastalıklarının tespiti amacıyla kullanılan farklı derin öğrenme modellerinin başarım analizleri ve karşılaştırmaları yapılmıştır. Çalışmada; doğruluk, kayıp, kesinlik, duyarlılık ve F1-score gibi temel metrikler üzerinden modellerin başarı düzeyleri değerlendirilmiştir. Elde edilen bulgular, özellikle sınıflandırma başarımı açısından hangi modelin daha etkili olduğunu ortaya koymaktadır. Ayrıca, dört farklı sınıf özelinde yapılan değerlendirmeler ile modellerin güçlü ve zayıf yönleri vurgulanmıştır. Tablo 3.5'te kullanılan tüm modellerin detaylı sonuçları verilmiştir.

VGG16 modeli, genel doğruluk oranı %97 olup, tüm sınıflarda dengeli bir başarı göstermiştir. Özellikle "Common Rust" ve "Healthy" sınıflarında yüksek F1-score değerlerine ulaşılmıştır (%99 ve %98). Ancak "Cercospora Leaf Spot" sınıfında kesinlik değerinin %91 olması, modelin bazı örneklerde hata yaptığını göstermektedir. Kayıp oranının düşük seyretmesi ve doğrulama eğrisinin kararlı olması, modelin istikrarlı bir genelleme gücüne sahip olduğunu ortaya koymaktadır.

InceptionV3 modeli, %93 doğruluk oranıyla diğer modellere kıyasla en düşük başarıyı göstermiştir. Özellikle "Cercospora Leaf Spot" sınıfında duyarlılığın %80 seviyesinde kalması, modelin bu sınıfı ayırt etmede zorluk yaşadığını göstermektedir. 0.07 üzerindeki kayıp değeri ve doğrulama eğrisindeki dalgalanmalar, modelin genelleme açısından daha zayıf kaldığını işaret etmektedir. Ayrıca, F1-score değerlerinin bazı sınıflarda düşüş göstermesi, modelin istikrarı açısından önemli bir dezavantaj oluşturmuştur.

MobileNetV2 modeli, %97 genel doğruluk oranı ile en başarılı modellerden biri olmuştur. Özellikle "Common Rust" ve "Healthy" sınıflarında F1-score %99–%100 seviyesine ulaşmıştır. Kayıp değeri 0.0325 ile düşük bir seviyededir ve modelin tüm sınıflarda dengeli bir başarım sunduğu gözlemlenmiştir. Özellikle modelin son 30 katmanının fine-tuning ile eğitilmesi, hiperparametre ayarlarının sezgisel olarak optimize edilmesi ve gelişmiş veri artırma tekniklerinin uygulanması ile başarı oranı önemli ölçüde artırılmıştır.

MobileNetV2 modeline uygulanan geliştirmeler sonucunda genel doğruluk oranı %97'ye ulaşmış; F1-score ortalaması 0.96–0.99 aralığında gerçekleşmiştir. Özellikle veri oranlarının

dengelenmesiyle birlikte “Healthy” sınıfında %100 doğruluk sağlanmış, diğer sınıflarda ise %93–%100 aralığında F1-score elde edilmiştir. Bu sonuçlar, modelin genelleme yeteneğini artırmak adına yapılan optimizasyonların etkili olduğunu göstermektedir.

Ayrıca elde edilen bulgular, MobileNetV2 modeli üzerinde yapılan iyileştirmelerin, diğer transfer öğrenme tabanlı mimarilere göre daha yüksek doğruluk ve istikrar sağladığını göstermektedir. VGG16 modeli ile %97, InceptionV3 ile %93 doğruluk elde edilirken; MobileNetV2 modeli, hem temel haliyle hem de geliştirilmiş versiyonunda %97 oranında doğruluk sağlayarak en kararlı sonuçları vermiştir.

Bu çalışmada tüm modellerin eğitimi, Google Colab ortamında GPU desteği ile gerçekleştirilmiştir. Eğitim süreci boyunca tur sayısı, işlem süresi göz önünde bulundurularak 10–15 aralığında sınırlandırılmıştır. Eğitim sürecinin erken durdurma (early stopping) ile kontrol altına alınması sayesinde aşırı öğrenme riski minimize edilmiştir.

Sonuç olarak, en başarılı şekilde performans gösteren model olarak geliştirilmiş MobileNetV2 modelinin, tüm metrikler açısından en iyi başarımı sergilediği görülmektedir. Buna karşılık, InceptionV3 modeli gerek kayıp oranı gerek doğruluk düzeyi açısından daha düşük başarım göstermiştir. Bu nedenle, InceptionV3 modeli üzerinde daha ileri düzey optimizasyon çalışmaları yapılması önerilmektedir.

KAYNAKLAR

[1] M. M. Dağ and C. Akbay, "Sürdürülebilir tarımsal uygulamalar ile küresel gıda krizine karşı alternatif çözümler," *Tarım Ekonomisi Araştırmaları Dergisi*, vol. 8, no. 2, pp. 182–194, 2022.

[2] T.C. Tarım ve Orman Bakanlığı, "Tarım ve ormancılık alanında yeni trendler," T.C. Tarım ve Orman Bakanlığı, Rapor, 2022.

[3] T.C. Samsun İl Tarım ve Orman Müdürlüğü, *Organik tarım ve bitki koruma açısından organik tarımda kullanılacak yöntemler*. T.C. Samsun İl Tarım ve Orman Müdürlüğü, 2012.

[4] B. Akbaş, "Bitki sağlığının sürdürülebilir tarımdaki yeri," *Ziraat Mühendisliği Dergisi*, no. 368, pp. 6–13, 2019.

[5] Y. Zhang, Y. Wang, and X. Li, "Plant disease: A growing threat to global food security," *Agronomy*, vol. 14, no. 8, p. 1615, 2024.

[6] S. Jauhari and K. K. Agrawal, "A comprehensive review on various plant diseases and impact on crop yield and quality," *Journal of Information Systems Engineering & Management*, vol. 10, no. 38s, pp. 430–439, 2025.

[7] Food and Agriculture Organization of the United Nations. (n.d.) About fao's work on plant production and protection. Accessed: 2025-06-07. [Online]. Available: https://www.fao.org/plant-production-protection/about/en

[8] Y. Zhou, X. Wang, and Z. Liu, "Changes in photosynthesis could provide important insight into the early detection of plant diseases," *Frontiers in Plant Science*, vol. 12, p. 839628, 2021.

[9] G. N. Agrios, *Plant pathology*, 5th ed. Elsevier Academic Press, 2005.

[10] United States Department of Agriculture Agricultural Research Service, "Monitoring and mitigating the spread of plant disease," United States Department of Agriculture, Tech.

Rep., 2021, accessed: 2025-06-07. [Online]. Available: https://www.ars.usda.gov/research/annual-report-on-science-accomplishments/fy-2021/monitoring-and-mitigating-the-spread-of-plant-disease/

[11] S. Savary *et al.*, "The global burden of pathogens and pests on major food crops," *Nature Ecology & Evolution*, vol. 3, pp. 430–439, 2019.

[12] W. B. Demilie, "Plant disease detection and classification techniques: A comparative study of the performances," *Journal of Big Data*, vol. 11, no. 1, p. Article 5, 2024.

[13] B. Sambana, H. S. Nnadi, M. A. Wajid, N. O. Fidelia, C. Camacho-Zuñiga, H. D. Ajuzie, and E. M. Onyema, "An efficient plant disease detection using transfer learning approach," *Scientific Reports*, vol. 15, p. Article 19082, 2025.

[14] J. Abbas, B. Nabila, R. N. Ali, A. Sadeghi-Niaraki, and D. Jeong, "Revolutionizing agriculture with artificial intelligence: Plant disease detection methods, applications, and their limitations," *Frontiers in Plant Science*, vol. 15, p. Article 1356260, 2024.

[15] A. K. Mahlein, U. Steiner, C. Hillnhütter, H. W. Dehne, and E. C. Oerke, "Recent advances in sensing plant diseases for precision crop protection," *Plant Disease*, vol. 96, no. 11, pp. 164–173, 2012.

[16] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[17] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in Plant Science*, vol. 7, p. 1419, 2016.

[18] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, "Deep neural networks based recognition of plant diseases by leaf image classification," *Computational Intelligence and Neuroscience*, vol. 2016, pp. 1–11, 2016.

[19] M. Brahimi, K. Boukhalfa, and A. Moussaoui, "Deep learning for tomato diseases: Classification and symptoms visualization," *Applied Artificial Intelligence*, vol. 31, no. 4, pp.

299–315, 2017.

[20] E. C. Too, Y. Li, S. Njuki, and Y. Liu, "A comparative study of fine-tuning deep learning models for plant disease identification," *Computers and Electronics in Agriculture*, vol. 161, pp. 272–279, 2019.

[21] A. G. Howard *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint*, vol. arXiv:1704.04861, 2017.

[22] B. Liu, Y. Zhang, D. He, and Y. Li, "Identification of apple leaf diseases based on deep convolutional neural networks," *Symmetry*, vol. 10, no. 1, p. 11, 2018.

[23] J. Lu, X. Liu, X. Ma, J. Tong, and J. Peng, "Improved mobilenetv2 crop disease identification model for intelligent agriculture," *PeerJ Computer Science*, vol. 9, p. e1595, 2023.

[24] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," *Computers and Electronics in Agriculture*, vol. 145, pp. 311–318, 2018.

[25] Y. Lu, S. Yi, N. Zeng, Y. Liu, and Y. Zhang, "Identification of rice diseases using deep convolutional neural networks," *Neurocomputing*, vol. 267, pp. 378–384, 2017.

[26] S. U. Khan, A. Alsuhaibani, A. Alabduljabbar, F. Almarshad, and Y. Altherwy, "A review on automated plant disease detection: Motivation, limitations, challenges, and recent advancements for future research," *Journal of King Saud University - Computer and Information Sciences*, vol. 37, p. 34, 2025.

[27] M. Singh, S. Nagpal, M. Vatsa, and R. Singh, "Enhancing fine-grained classification for low resolution images," arXiv preprint arXiv:2105.00241, 2021, accessed: 2025-06-07. [Online]. Available: https://arxiv.org/abs/2105.00241

[28] D. Cai, K. Chen, Y. Qian, and J.-K. Kämäräinen, "Convolutional low-resolution fine-grained classification," arXiv preprint arXiv:1703.05393, 2017, accessed: 2025-06-07. [Online]. Available: https://arxiv.org/abs/1703.05393

[29] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, pp. 249–259, 2018.

[30] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, 2019.

[31] D. Singh, N. Jain, P. Jain, P. Kayal, S. Kumawat, and N. Batra, "Plantdoc: A dataset for visual plant disease detection," in *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD*, 2020.

[32] J. A. Ilemobayo, O. I. Durodola, O. Alade, O. J. Awotunde, T. O. Adewumi, O. Falana, A. Ogungbire, A. Osinuga, D. Ogunbiyi, A. Ifeanyi, I. E. Odezuligbo, and O. E. Edu, "Hyperparameter tuning in machine learning: A comprehensive review," *Journal of Engineering Research and Reports*, vol. 26, no. 6, pp. 388–395, 2024.

[33] T. Yu and H. Zhu, "Hyper-parameter optimization: A review of algorithms and applications," 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2003.05689

[34] F. Xiao, "Image augmentation improves few-shot classification performance in plant disease recognition," 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2208.12613

[35] G. Wang, Y. Sun, and J. Wang, "Automatic image-based plant disease severity estimation using deep learning," *Computational Intelligence and Neuroscience*, 2017.

[36] S. Mehnaz and M. T. Islam, "Rice leaf disease detection: A comparative study between cnn, transformer and non-neural network architectures," 2025, preprint. [Online]. Available: https://doi.org/10.48550/arXiv.2501.06740

[37] L. Zhang, C. Li, X. Wu, H. Xiang, Y. Jiao, and H. Chai, "Bo-cnn-bilstm deep learning model integrating multisource remote sensing data for improving winter wheat yield estimation," *Frontiers in Plant Science*, vol. 15, p. 1500499, 2024. [Online]. Available: https://doi.org/10.3389/fpls.2024.1500499

[38] S. H. Lee, H. Goëau, P. Bonnet, and A. Joly, "Attention-based recurrent neural network for plant disease classification," *Frontiers in Plant Science*, vol. 11, p. 601250, 2020. [Online]. Available: https://doi.org/10.3389/fpls.2020.601250

[39] T. Sharma, "Plant village dataset (updated)," 2023, accessed: 2025-06-10. [Online]. Available: https://www.kaggle.com/datasets/tushar5harma/plant-village-dataset-updated

[40] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[41] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[42] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural Computation*, vol. 29, no. 9, pp. 2352–2449, 2017.

[43] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," *2017 International Conference on Engineering and Technology (ICET)*, pp. 1–6, 2017.

[44] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in Plant Science*, vol. 7, p. 1419, 2016.

[45] E. C. Too, L. Yujian, S. Njuki, and L. Yingchun, "A comparative study of fine-tuning deep learning models for plant disease identification," *Computers and Electronics in Agriculture*, vol. 161, pp. 272–279, 2019.

[46] U. Blog. (2023) Basic cnn architecture with applications. Accessed: 2025-06-03. [Online]. Available: https://www.upgrad.com/blog/basic-cnn-architecture/

[47] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[48] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke,

and A. Rabinovich, “Going deeper with convolutions,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[49] A. G. Howard *et al.*, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” *arXiv preprint*, vol. arXiv:1704.04861, 2017.

[50] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510–4520.

[51] D. Pineda Medina, I. Miranda Cabrera, R. Alfonso de la Cruz, L. Guerra Arzuaga, S. Cuello Portal, and M. Bianchini, “A mobile app for detecting potato crop diseases,” *Journal of Imaging*, vol. 10, no. 2, p. 47, 2024.

[52] F. Zubair, M. Saleh, Y. Akbari, and S. Al Maadeed, “A robust ensemble model for plant disease detection using deep learning architectures,” *AgriEngineering*, vol. 7, no. 5, p. 159, 2025.