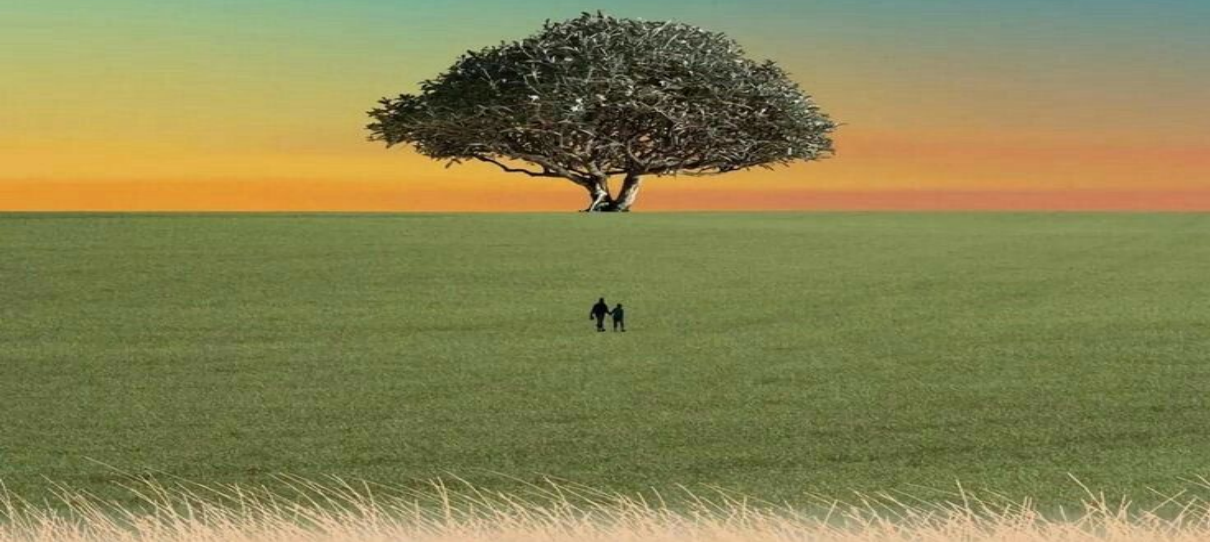
AKADEMİK PERSPEKTİFTEN BİLGİSAYAR BİLİMLERİ VE MÜHENDİSLİĞİ

Editör: Dr.Öğr.Üyesi Mehmet Süleyman YILDIRIM



yaz
yayınları

Akademik Perspektiften Bilgisayar Bilimleri ve Mühendisliği

Editör

Dr.Öğr.Üyesi Mehmet Süleyman YILDIRIM

yaz yayınları

2025

Akademik Perspektiften Bilgisayar Bilimleri ve Mühendisliği

Editör: Dr.Öğr.Üyesi Mehmet Süleyman YILDIRIM

E_ISBN 978-625-5596-67-3

Haziran 2025 – Afyonkarahisar

Dizgi/Mizanpaj: YAZ Yayınları

Kapak Tasarım: YAZ Yayınları

YAZ Yayınları. Yayıncı Sertifika No: 73086

M.İhtisas OSB Mah. 4A Cad. No:3/3
İscehisar/AFYONKARAHİSAR

www.yazyayinlari.com

yazyayinlari@gmail.com

info@yazyayinlari.com

İÇİNDEKİLER

DERİN ÖĞRENME UYGULAMALARINDA OPTİMİZASYON ALGORİTMALARININ TAHMİN PERFORMANSINA ETKİSİ

Abdurrahman Burak GÜHER[1]

Haydar TUNA[2]

1. GİRİŞ

Derin öğrenme, çok katmanlı yapay sinir ağları kullanarak genelleme yapma yeteneğine sahip bir makine öğrenimi alt alanıdır. Genellikle büyük veri setlerinden anlamlı desenler çıkarabilen ve karmaşık problemlerin çözümünde sıklıkla kullanılan bir yapa zekâ türüdür. Çoğunlukla görüntü işleme, doğal dil işleme, otonom sistemler, tıp, enerji gibi alanlarda amaca uygun çözümler geliştirilebilmektedir (LeCun, Bengio, & Hinton, 2015).

Farklı alanlarda kullanılmak üzere birçok derin öğrenme mimarisi geliştirilmiştir. Örneğin, Convolutional Neural Network (CNN) mimarisinin görüntü işleme alanında kullanılması ile bu alanda çalışan uzmanlarla yarışabilir yüksek doğruluk oranlarına ulaşılmıştır. Benzer şekilde, Recurrent Neural Network (RNN) mimarisi ve türevleri ile dil modelleme ve makine çevirisi gibi alanlarda yüksek başarıya ulaşılmış, bu alanlarda büyük

[1] Dr. Öğr. Üyesi, Osmaniye Korkut Ata Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi, Bilgisayar Mühendisliği Bölümü, burakguher@osmaniye.edu.tr, ORCID: 0000-0002-3971-6765.

[2] Dr. Öğr. Üyesi, Osmaniye Korkut Ata Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi, Bilgisayar Mühendisliği Bölümü, haydartuna@osmaniye.edu.tr, ORCID: 0000-0003-2388-653X.

gelişmeler yaşanmıştır (Goodfellow, Bengio, Courville, & Bengio, 2016).

Derin öğrenmenin günümüzde yaygınlaşmasının en önemli sebeplerinden birisi büyük veri setleriyle çalışabilmesi ve geliştirilen uygulamaların daha üstün sonuçlar vermesidir. Veri miktarının göreceli olarak artması geliştirilen modelin karmaşık yapıları daha kolay öğrenmesini sağlamaktadır. Bu durum özellikle içinde bulunduğumuz çağda derin öğrenmeyi diğer yöntemlere kıyasla daha avantajlı bir konuma getirmiştir (Kelleher, 2019).

Geliştirilen uygulamalar birçok alanda başarılı sonuçlar verse de modellerde çeşitli zorluklarla karşılaşılabilmektedir. Genel olarak uygulama geliştirebilmek için yüksek işlem gücüne gereksinim duyulmaktadır. Ayrıca düşük veri setlerinde aşırı uyum (overfitting) gibi sorunlarla karşılaşılabilmektedir. Diğer makine öğrenme yöntemlerine kıyasla açıklana bilirliği düşüktür ve karar verme süreçleri insanlar tarafından kolaylıkla anlaşılamamaktadır (Zhang, Bengio, Hardt, Recht, & Vinyals, 2021). Bununla birlikte uygulama geliştirirken modeli en yüksek performans seviyesine çıkaracak hiperparemetrelerin ne olduğunun önceden bilinmesi mümkün değildir. Eğitim sürecini kontrol eden çeşitli parametrelerin optimal değerlerinin belirlenmesi uzun ve meşakkatli bir süreçtir ve bu işleme hiperparametre optimizasyonu adı verilmektedir. Uygulama geliştirirken kullanılan veri setinin tipi, yapısı, boyutu, dağılımı ve özellikleri ayarlanacak hiperparametrelerin en iyi kombinasyonunun belirlenmesinde etkili olmaktadır (Bengio, 2012).

Derin öğrenme modellerinde hiperparametre optimizasyonu genellikle iki farklı şekilde ele alınmaktadır. İlki, katman sayısı, her katmandaki nöron sayısı, kullanılan aktivasyon fonksiyonları gibi sinir ağı modelinin mimarisine ait

parametrelere uygun değerlerin ayarlanmasıdır. İkincisi ise öğrenme oranı, epoch sayısı, yığın boyutu (batch size), optimizasyon algoritmaları gibi eğitim sürecine ait parametrelerin belirlenmesidir. Yapılan işlemler genellikle zaman alıcıdır ve uzmanlık gerektirmektedir. Yanlış ayarlamalar modelin performansını önemli ölçüde düşürebilmektedir (Yu & Zhu, 2020).

Optimizasyon algoritmaları, derin öğrenme modellerinin performansını ve eğitim sürecini önemli ölçüde etkileyen parametrelerin başında gelmektedir. Öyle ki modelleme çalışmalarında ayarlanan birçok hiperparametre, optimizasyon algoritmaları ile doğrudan ilişkilidir (Altun & Talu, 2021). Bu algoritmaların temel amacı modelin kayıp fonksiyonunu minimize etmektir. Kayıp fonksiyonu ise modelin tahmin ettiği değerler ile gerçek değerler arasındaki farkın ölçümüdür. Aradaki fark ne kadar küçük olursa geliştirilen modelin genelleme yeteneği o oranda artmış olur. Derin öğrenmede, optimizasyon algoritmaları modelin ağırlıklarını güncelleyerek tahmin performansını iyileştirmeyi amaçlar (Yang & Shami, 2020).

Bu çalışmada derin öğrenme tabanlı tahmin modelleri geliştirilmiş ve farklı optimizasyon algoritmalarının modellerin performansı üzerindeki etkileri analiz edilmiştir. Modellerin geliştirilmesinde TensorFlow'un resmi yüksek seviyeli uygulama programlama arayüzü (Application Programming Interface - API) olan Keras'tan faydalanılmıştır. Keras, python programlama dili ile yazılmış, açık kaynak kodlu ve yüksek seviyeli bir derin öğrenme çerçevesidir. Keras, François Chollet tarafından geliştirilmiş ve ilk olarak 2015 yılında yayınlanmıştır. 2017 yılında Google, Keras'ı TensorFlow'un resmi üst düzey API'si olarak tanıttıktan sonra geniş kitlelere ulaşmıştır (Chollet, 2018). Keras'ın kütüphanesinde var olan ve sıklıkla kullanılan on farklı optimizasyon algoritması seçilmiş ve tahmin doğrulukları karşılaştırılmıştır.

2. MATERYAL VE YÖNTEMLER

Bu bölümde, modelleme çalışmasında kullanılan optimizasyon algoritmalarının temel ve yapısal özellikleri, avantajlı ve dezavantajlı yönleri karşılaştırmalı olarak ele alınmaya çalışılmıştır. Osmaniye ilinin rüzgâr potansiyelinin tahmin edilmesinde yararlanılan veri setinin özellikleri ve derin öğrenme tabanlı tahmin modellerinin mimari yapısı anlatılmıştır.

2.1. Optimizasyon Algoritmalarının Yapısal Özellikleri

2.1.1. SGD (Stochastic Gradient Descent) Algoritması

Bu algoritma özellikle büyük veri kümeleri üzerinde derin öğrenme modellerini optimize etmek için kullanılan temel bir optimizasyon algoritmasıdır. SDG algoritması ilk olarak Robbins ve Monro tarafından 1951 yılında tanıtılmıştır (Robbins & Monro, 1951). SGD çalışma mantığı açısından klasik Gradient Descent yönteminin daha kullanışlı bir varyasyonu olarak bilinmektedir. Temel olarak, her adımda tüm veri kümesinin kullanılması yerine, yalnızca tek bir örnek veya küçük alt grupları (mini-batch) kullanarak model parametrelerini güncellemesi ile öne çıkmaktadır. Bu sayede büyük veri kümeleriyle çalışırken hesaplama maliyetini önemli ölçüde azaltılmış olmaktadır. Matematiksel formülü Denklem 1 de gösterilmiştir.

$$\theta_{t+1} = \theta_t - \eta \nabla J(\theta_t; x_i, y_i) \quad (1)$$

Formülde θ_t, t. iterasyondaki ağırlıkları, θ_{t+1}, t+1. iterasyondaki güncellenmiş ağırlıkları, η, öğrenme oranını, $\nabla J(\theta_t; x_i, y_i)$, modelin tahmin değerleri (y_i) ile gerçek değerleri (x_i) arasındaki hatanın kayıp fonksiyonunun gradyanını temsil eder. SGD, her iterasyonda yalnızca bir ve birkaç veri örneği üzerinde kayıp fonksiyonunun gradyanını hesaplar. Hesaplanan gradyan değerinin aldığı değere göre ağırlıkları günceller (Bottou, 2010).

2.1.2. Adam (Adaptive Moment Estimation) Algoritması

Derin öğrenme modellerinin eğitim sürecinde yaygın olarak tercih edilen adaptif öğrenme tabanlı bir algoritmadır. Adam algoritması, Kingma ve Ba tarafından 2014 yılında önerilmiştir (Kingma & Ba, 2014). Bu algoritma, klasik SGD algoritmasının eksikliklerini gidermek amacıyla hem momentum hem de RMSProp tekniklerini birleştirerek daha kararlı ve hızlı bir yakınsama sağlar. Özellikle büyük ve yüksek boyutlu veri kümelerinde öğrenme sürecinin dinamik yapısını daha efektif kontrol ettikleri için birçok öğrenme çerçevesinde varsayılan olarak tercih edilmektedir (Ruder, 2016).

Yapısal özellikleri sayesinde hem gradyanların ani değişimlerinin etkisi azaltılır hem de farklı ölçeklerdeki parametrelerin uyumlu şekilde optimize edilmesi sağlanmış olmaktadır. Adam algoritması, özellikle seyrek veri kümelerinin yer aldığı doğal dil işleme ve öneri sistemleri gibi alanlarda oldukça etkili sonuçlar vermektedir (Bengio, 2012; Ruder, 2016). Matematiksel güncelleme formülü Denklem 7'de verilmiştir.

$$g_t = \nabla J(\theta_t;\, x_i, y_i) \tag{2}$$

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \tag{3}$$

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2 \tag{4}$$

$$\acute{m}_t = \frac{m_t}{1-\beta_1^t} \tag{5}$$

$$\tilde{v}_t = \frac{v_t}{1-\beta_2^t} \tag{6}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\tilde{v}_t}+\varepsilon}\acute{m}_t \tag{7}$$

Yukarıdaki formüllerde daha önce verilen özelliklere ek olarak, g_t, t zamanındaki gradyanı, m_t, gradyanların birinci moment hesabını, v_t, gradyanların ikinci moment hesabını, $\acute{m}_t$ ve $\tilde{v}_t$, birinci ve ikinci momentin bias düzeltmesi sonrası değerlerini, β_1 ve β_2, hareketli ortalama katsayılarını ve son

olarak ε ise sayısal kararlılığı sağlamak adına kullanılan pozitif bir değeri temsil eder. Formüldeki yapısal özellikler sayesinde Adam, birçok optimizasyon algoritmasına kıyasla daha hızlı ve kararlı bir şekilde, yapılan her güncelleme sonucunda kayıp fonksiyonunun değerini azaltarak en düşük değere ulaşma eğilimindedir (Li, Rakhlin, & Jadbabaie, 2023).

2.1.3. RMSprop (Root Mean Square Propagation) Algoritması

RMSprop algoritması, özellikle değişken gradyan büyüklüklerinin öğrenme sürecinde oluşturduğu olumsuzlukların giderilmesi maksadıyla geliştirilmiş adaptif tabanlı bir optimizasyon algoritmasıdır. Bu algoritma, ilk olarak Geoffrey Hinton tarafından 2012 yılında Coursera'daki derin öğrenme derslerinde önerilmiş ve daha sonra geniş kullanım kitlesine sahip olmuştur (Hinton, Srivastava, & Swersky, 2012). RMSprop, SGD algoritmasının zayıf yönlerini hedef alarak özellikle gürültülü ve seyrek veri kümelerinde performansı arttırıcı bir yakınsama süreci sağlamayı amaçlamaktadır.

Bu algoritma, geliştirilen modeldeki ağırlık ve bias değerlerinin geçmişteki gradyan karelerinin hareketli ortalamasını hesaplayarak, öğrenme oranlarını bu bilgiye göre dinamik olarak ayarlamaktadır. Özellikle tekrar eden gradyan patlamaları veya gradyan sönmesi gibi problemlerin çözümünde önemli rol üslenmektedir (Tieleman, 2012). Algoritmanın matematiksel güncelleme formülü Denklem 9'da verilmiştir.

$$E[g^2]_t = \rho \cdot E[g^2]_{t-1} + (1-\rho) \cdot g_t^2 \quad (8)$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t \quad (9)$$

Formülde $E[g^2]_t$, gradyan karesinin t zamanındaki hareketli ortalamasını ve ρ ise sönümleme katsayısını ifade eder. Bu değer genellikle 0.9 civarında alınır. Formüllerde geçen diğer parametreler daha önce verilen formüllerde ayrıntısı ile

açıklanmıştır. RMSprop algoritması öğrenme süreci boyunca ağırlıkları daha dengeli günceller. Algoritmanın eğitim süreci hızlıdır ve kararlı bir şekilde kayıp fonksiyonunun minimal seviyelere ulaşması sağlanır (Goodfellow et al., 2016).

2.1.4. AdamW (Adam with decoupled Weight decay) Algoritması

Derin öğrenme modellerinin optimal ağırlıklarının belirlenmesi sürecinde, ağırlık çürümesini (weight decay) daha etkili şekilde kontrol altına almak maksadıyla geliştirilmiştir. Adam algoritmasının bir varyasyonudur. Ağırlık çürümesi, makine öğrenmesi modellerinin aşırı uyum (overfitting) yapmasını engellemek maksadıyla kullanılan bir düzenleme (L2 regularization) tekniğidir. Bu algoritma, Loshchilov ve Hutter tarafından 2017 yılında önerilmiştir (Loshchilov & Hutter, 2017).

Adam algoritmasında, ağırlık çürümesi doğrudan gradyanlara eklenmektedir. Ancak bu yöntem ağırlık çürümesinin etkisinin zayıflamasına yol açmaktadır. AdamW algoritması ise ağırlık çürümesini gradyan güncellemesinden ayırarak doğrudan parametrelerin kendisine uygular. Böylece ağırlıkların aşırı büyümesini daha doğru biçimde sınırlanmakta ve modelin genelleme performansı artmaktadır (Liu et al., 2019; Ruder, 2016). Bu nedenle PyTorch ve TensorFlow gibi derin öğrenme çerçevelerinde giderek daha yaygın bir şekilde varsayılan optimizasyon algoritması olarak kullanılmaktadır. AdamW optimizasyon algoritmasının matematiksel güncelleme formülü Denklem 10'da verilmiştir.

$$\theta_{t+1} = \theta_t - \eta \left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t + \varepsilon}} + \lambda\theta_t\right) \quad (10)$$

Gradyan moment hesaplamaları, bias düzeltmeleri ve parametre güncellemeleri Adam algoritması gibidir. Ancak Denklem 9 da verilen algoritmanın güncelleme formülünde tek fark, λ ağırlık çürüme katsayısının t. iterasyonundaki θ_t ağırlıkları

ile çarpılıp formülün kendisine eklenmesidir. Formülün bu şekilde uygulanması ağırlıkların aşırı büyümesini doğrudan engelleyerek düzenli bir öğrenme süreci sağlamaktadır (Liu et al., 2019).

2.1.5. Adagrad (Adaptive Gradient Algorithm) Algoritması

Adagrad, John Duchi, Elad Hazan ve Yoram Singer tarafından 2011 yılında önerilen bir optimizasyon algoritmasıdır. SGD tabanlı olan bu algoritma, öğrenme oranını her bir ağırlık için adaptif hale getirerek daha etkili bir optimizasyon süreci sunmayı hedeflemektedir. Diğer algoritmalara kıyasla model ağırlıklarının her biri için güncellemeleri geçmişteki gradyanlara bağlı olarak dinamik bir şekilde güncellemesiyle öne çıkmaktadır (Duchi, Hazan, & Singer, 2011). Bu algoritmanın matematiksel güncelleme formülü Denklem 12 de verilmiştir.

$$G_t = G_{t-1} + g_t^2 \tag{11}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t + \varepsilon}} g_t \tag{12}$$

Formülde geçen g_t, t. iterasyondaki gradyanı (Denklem 2), G_t, gradyanların karelerinin kümülatif toplamını ifade eder. Diğer parametreler daha önce verilen optimizasyonların güncelleme denklemleinde açıklanmıştır. Veri kümelerindeki özelliklerin farklı ölçeklerde olduğu durumlarda daha kararlı bir optimizasyon süreci sağlamaktadır. Ancak, Adagrad'ın öğrenme oranını zamanla sıfıra yaklaştırması, algoritmanın uzun vadede yakınsama problemlerinin ortaya çıkmasına yol açabilmektedir (Zou & Shen, 2018).

2.1.6. Adadelta Algoritması

Adadelta, 2012 yılında Matthew D. Zeiler tarafından önerilen ve SGD tabanlı bir optimizasyon algoritmasıdır. Bu algoritma, Adagrad algoritmasının öğrenme oranını zamanla sıfıra yakınsaması problemlemini çözmek üzere geliştirilmiştir.

Bu sorun uzun vadeli öğrenmeyi engellemiştir. Adadelta ile SGD algoritmasında yaygın olarak karşılaşılan sabit öğrenme oranı seçme problemi çözülmüştür (Zeiler, 2012). Bu sayede büyük veri kümeleri üzerinde çalışan modellerde, öğrenme oranının yanlış seçilmesi durumunda ortaya çıkabilecek yakınsama problemleri önlenmiş olmaktadır. Adadelta algoritmasının matematiksel güncelleme formülü Denklem 15 de gösterilmiştir.

$$\Delta\theta_t = -\frac{\sqrt{E[\Delta\theta^2]_{t-1}+\varepsilon}}{\sqrt{E[g^2]_t+\varepsilon}} g_t \quad (13)$$

$$E[\Delta\theta^2]_t = \rho \cdot E[\Delta\theta^2]_{t-1} + (1-\rho)\cdot \Delta\theta_t^2 \quad (14)$$

$$\theta_{t+1} = \theta_t + \Delta\theta_t \quad (15)$$

Denklem 13'de, ağırlıkların güncellenmesini ve bu formülde geçen $E[g^2]_t$ ise gradyan karelerinin üstel hareketli ortalamasının hesabını ifade eder. Denklem 14, ağırlıkların güncellemelerinin karesinin üstel hareketli ortalamasını ve Denklem 15 ise ağırlıkların nihai güncellemelerinin hesabıdır. Formüllerde geçen diğer parametreler daha önce verilen formüllerde ayrıntısı ile ele alınmıştır. Adadelta, görüntü işleme, doğal dil işleme ve öneri sistemleri gibi birçok derin öğrenme uygulamasında başarıyla kullanılmıştır (Ruder, 2016).

2.1.7. Adamax Algoritması

Adamax, Adam algoritmasının genelleştirilmiş bir versiyonu olarak ortaya çıkmıştır. Adam algoritması temelde birinci moment ve ikinci moment (L2 form) tahminlerini kullanarak adaptif bir öğrenme yöntemi kullanmaktadır. Adamax ise bu algoritmanın ikinci moment tahminini sonsuz norm olarak adlandırdıkları (L∞ norm) bir form ile değiştirerek, özellikle gradyan değerlerinin büyük dalgalanmalar gösterdiği durumlarda daha kararlı bir optimizasyon süreci sunmayı hedeflemektedirler. Bu algoritma, derin öğrenme modellerinin farklı alanlarında başarıyla uygulanmaktadır (Kingma & Ba, 2014). Adamax

algoritmasının matematiksel güncelleme formülü Denklem 16 da gösterilmiştir.

$$u_t = max(\beta_2 u_{t-1}, |g_t|) \quad (15)$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{u_t} * \frac{m_t}{1-\beta_1^t} \quad (16)$$

Güncelleme formülünde geçen, m_t, birinci moment tahminini (Denklem 3), u_t, ikinci moment tahminini ifade etmektedir. Diğer parametreler daha önceki formüllerde detaylı olarak açıklanmıştır. Adamax'ın en önemli özelliği ikinci moment tahminini L∞ norm kullanarak hesaplamasıdır. Önerilen bu mekanizma modern optimizasyon algoritmaları arasında önemli bir yer tutmaktadır.

2.1.8. Adafactor Algoritması

Adafactor, Noam Shazeer ve Mitchell Stern tarafından önerilen bir optimizasyon algoritmasıdır. Bu algoritma büyük ölçekli modellerin eğitimi sırasında bellek kullanımını azaltmak amacıyla geliştirilmiştir. Adafactor, Adam algoritmasının adaptif öğrenme oranı avantajlarını korurken, bellek verimliliğini artırmak için ikinci moment hesabını faktörize ederek saklamaktadır. Özellikle milyarlarca parametreye sahip transformer tabanlı büyük dil modellerinin eğitiminde önemli avantajlar sağlamaktadır (Shazeer & Stern, 2018). İlgili algoritmanın matematiksel güncelleme formülü Denklem 20'de gösterilmiştir.

$$R_t = \beta_2 R_{t-1} + (1 - \beta_2)(g_t g_t^T) \quad (17)$$

$$C_t = \beta_2 C_{t-1} + (1 - \beta_2)(g_t g_t^T) \quad (18)$$

$$\widehat{V}_t(i,j) = \sqrt{\frac{R_t(i)C_t(i)}{sum(R_t).sum(C_t)}.sum(g_t \odot g_t)} \quad (19)$$

$$\theta_{t+1} = \theta_t - \eta * \frac{g_t}{\sqrt{\widehat{V}_t}+\varepsilon} \quad (20)$$

Adafactor'un temel amacı, her bir ağırlık değeri için ayrı bir ikinci moment hesabını tutmak yerine ağırlık matrisinin satır

ve sütun faktörlerini kullanarak bellek kullanımını azaltmaktır. Formülde geçen g_t, t. iterasyondaki gradyanı (Denklem 2), R_t ve C_t, sırasıyla satır ve sütun faktörlerini, $\odot$, hadamard (eleman bazında) çarpımı, $\widehat{V}_t(i,j)$ ise bu algoritmaya özgü ikinci moment hesabını temsil eder. Diğer değişkenler daha önce verilen algoritmaların formüllerinde detaylı olarak açıklanmıştır. Adafactor'un ayrıca Adam gibi popüler optimizasyon algoritmalarına benzer veya daha iyi bir performans sergilediğini gösterilmiştir. Örneğin, Google'ın T5 (Raffel et al., 2020) ve Meta AI'in LLaMA (Touvron et al., 2023) gibi büyük dil modelleri bu algoritma kullanılarak eğitilmiştir.

2.1.9. Nadam (Nesterov-accelerated Adaptive Moment Estimation) Algoritması

Nadam, Timothy Dozat tarafından önerilen bir optimizasyon algoritmasıdır. Bu algoritma, Adam optimizasyon algoritması ile Nesterov Momentum tekniğini birleştirerek, daha hızlı ve daha kararlı bir yakınsamayı hedeflemektedir. Nesterov Momentum, klasik momentum yönteminden farklı olarak, gradyanı hesaplamadan önce momentum adımını devre dışı bırakmaktadır. Böylelikle algoritmanın daha öngörülü olması ve daha hızlı yakınsaması sağlanmış olmaktadır. Nadam, özellikle karmaşık ve yüksek boyutlu optimizasyon problemlerinde etkili sonuçlar vermektedir (Dozat, 2016). Nadam'ın matematiksel güncelleme formülü Denklem 22'de verilmiştir.

$$\acute{m}^{nesterov} = \frac{\beta_1 \acute{m} + (1-\beta_1) g_t}{1-\beta_1^{t+1}} \tag{21}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \varepsilon} \acute{m}^{nesterov} \tag{22}$$

Formülde, $\acute{m}^{nesterov}$, Nesterov momentum ile düzeltilmiş birinci moment hesabını temsil etmektedir. Diğer parametreler daha önceki denklemlerde ayrıntısı ile açıklanmıştır. Klasik Adam algoritmasında güncellenmiş ağırlık değerlerinin hesaplanmasında Denklem 3'te verilen birinci moment hesabı

kullanılırken, Nadam'da $\acute{m}^{nesterov}$ kullanılmaktadır. Diğer tüm hesaplamalar Adam algoritmasında açıklandığı gibidir. Nadam algoritmasının, Adam'a kıyasla bazı problemlerde daha iyi sonuçlar verdiği gösterilmiştir.

2.1.10. Ftrl (Follow The Regularized Leader) Algoritması

Bu algoritma, 2013 yılında H. Brendan McMahan ve arkadaşları tarafından önerilmiştir. Ftrl ile özellikle büyük ölçekli ve seyrek veri kümeleri üzerinde çalışan doğrusal modeller geliştirilmiştir. Reklamcılık, öneri sistemleri gibi yüksek hacimli ve seyrek veri kümelerinin kullanıldığı alanlarda başarıyla uygulanmaktadır. Çalışma mekanizması, önemsiz özelliklerin parametrelerini sıfıra iterek modelin seyrekliğini artırmaktadır. Ayrıca, adaptif öğrenme oranı mekanizması sayesinde her bir parametre için farklı bir öğrenme oranı hesaplanmasını garanti etmektedir (McMahan et al., 2013). FTRL algoritmasının matematiksel güncellemesi Denklem 25 de verilmiştir.

$$\sigma_t = \frac{1}{\propto}(\sqrt{\sum_{s=1}^{t} g_s^2} - \sqrt{\sum_{s=1}^{t-1} g_s^2}) \tag{23}$$

$$z_t = z_{t-1} + g_t - \sigma_t \theta_t \tag{24}$$

$$\theta_{t+1,i} = \begin{cases} 0 & \text{eğer } |z_{t,i}| \leq \lambda_1 \\ -\dfrac{1}{\lambda_2 + \dfrac{\beta + \sqrt{\sum_{s=1}^{t} g_s^2}}{\alpha}} (z_{t,i} - sign(z_{t,i})\lambda_1) & \text{diğer durumlarda} \end{cases} \tag{25}$$

Formülde, λ_1 ve λ_2, sırasıyla L1 ve L2 düzenlileştirme parametrelerini, z_t, kümülatif gradyanları, σ_t, adaptif öğrenme oranını temsil eder. Eğer $|z_{t,i}| \leq \lambda_1$ ise, ilgili parametre sıfıra eşitlenir. Formülde geçen diğer özellikler daha önce detaylıca açıklanmıştır. FTRL'nin en önemli özelliği, L1 düzenlileştirme sayesinde seyrek modeller oluşturabilmesidir. Bu mekanizması sayesinde Google, FTRL algoritmasını çevrimiçi reklamcılık sistemlerinde kullanmaktadır. Aynı zamanda, yüksek hacimli seyrek veri kümeleri üzerinde diğer algoritmalara kıyasla daha iyi

performans gösterdiğini raporlamıştır (Meng, Ge, Tian, An, & Gao, 2023).

2.2. Veri Seti ve Özellikleri

Çalışmada, optimizasyon algoritmaları temelli derin öğrenme modellerinin tahmin doğruluğunun analiz edilmesinde, Osmaniye ilinin 2017 yılı saatlik olarak ölçülen meteorolojik verilerinden faydalanılmıştır. Hedef bölgenin rüzgar gücü potansiyelinin belirlenmesi hedeflenmiştir. Bu amaçla Meteoroloji Genel Müdürlüğünden (MGM) saatlik olarak ölçülen, Bağıl Nem (%), Aktuel Basınç (hPa), Sıcaklık (°C) ve Rüzgar Hızı (m/sn) ve Rüzgar Yönü (Derece) verileri talep edilmiştir. Ayrıca çalışmanın gerçekçi temellere dayanması açısından, Siemens Gamesa SG 4.5-145 model 4.5 MW'lık bir rüzgar türbininin spesifik özelliklerinden faydalanılarak girdi verileri optimize edilmiştir.

İlk olarak bağıl nem, basınç ve sıcaklık verileri kullanılarak Denklem 26'da verilen formülden nemli hava için yoğunluğuk değerleri bulunmuştur. Formülde geçen ρ, havanın yoğunluğunu, p_d, kuru havanın kısmi basıncını, p_v, su buharının kısmi basıncını, R_d, kuru hava gaz sabitini, R_v, su buharı için gaz sabitini, T, kelvin cinsinden sıcaklık değerini temsil eder. İkinci olarak Denklem 27'de verilen formülden faydalanılarak 10 metrede ölçülen rüzgar hızı verisi rüzgar türibinin 110 metre yükseklikte çalışması (hub yüksekliği) varsayıldığından bu yükseklik için normalize edilmiştir. Formülde, v_{110}, 110 metre yükseklikteki hesaplanan rüzgar hızını, v_{10}, 10 metreyükseklikte ölçülen rüzgar hızını, h_{110} ve h_{10}, sırasıyla 110 ve 10 metre yüksek değerlerini, α ise rüzgar profili katsayısını temsil eder. Bu katsayı genellikle Hellmann katsayısı olarak bilinmektedir. Rüzgar hızının yükseklikle nasıl değiştiğini tanımlayan logaritmik rüzgar profili modelinde kullanılmaktadır. Hellmann katsayısı, yüzey pürüzlülüğüne ve arazi tipine bağlı olarak değişmektedir

(Hansen, 2015). Şehir içi alanlar için 0.2 değeri varsayılan olarak kullanıldığından, çalışmada bu katsayı değerinden faydalanılmıştır. Son olarak Denklem 28'deki matematiksel formülden yararlanılarak SG 4.5-145 model rüzgar türibininin üretebileceği güç değerleri hesaplanmıştır. Formülde *P*, rüzgar gücünü, ρ, havanın yoğunluğunu, *A*, türbinin süpürme alanını, *v*, rüzgar hızını temsil eder. Süpürme alanı rüzgar türbininin rotor çapına bağlıdır ve dairesel alan formülüyle (πr^2) bulunmaktadır (Hansen, 2015).

$$\rho = \frac{p_d}{R_d T} + \frac{p_v}{R_v T} \tag{26}$$

$$v_{110} = v_{10}\left(\frac{h_{110}}{h_{10}}\right)^{\alpha} \tag{27}$$

$$P = \frac{1}{2}\rho A v^3 \tag{28}$$

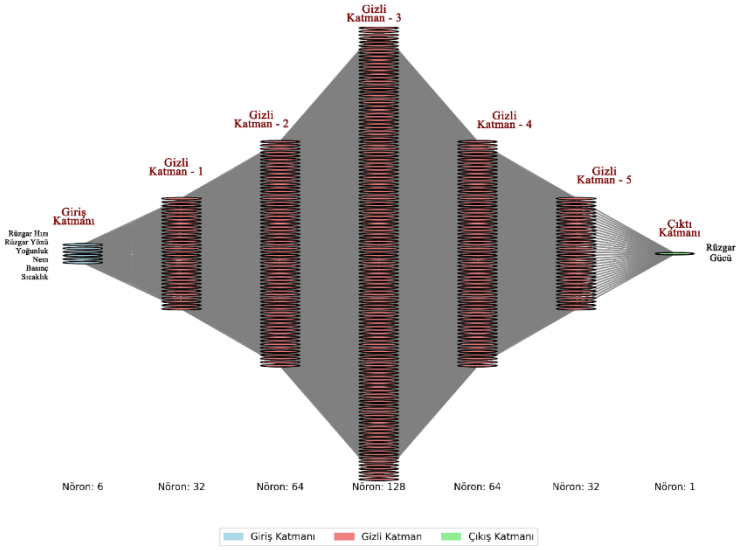
Gerçek dünyada bir rüzgar türbini tüm kinetik enerjiyi elektrik enerjisine dönüştürememektedir. Bu nedenle çalışmada yararlanılan rüzgar türbininin nominal üreteceği güç değerleri baz alınmıştır. Siemens Gamesa'nın resmî sitesinde, SG 4.5-145 model rüzgar türibini 4.2 – 4.8 MW arasında güç üretebileceği belirtilmiştir (Siemens Gamesa Renewable Energy). Çalışmada 4.8 MW üzerinde üretilen güç değerleri, 4.65 MW değerine sabitlenmiştir. Ayrıca ham veriseti üzerinde çeşitli veri ön işleme teknikleri uygulanarak eksik veriler tamamlanmıştır. Eksik verilerin boyutu tüm verilerin % 2'si kadar bulunmuştur. Düzenleme sonucunda modelleme çalışmalarında kullanılan (Ocak ayının 10. Günü saat 10:00 – 20:00 arası düzenlenen) verilerin bir kısmı Şekil 1'de verilmiştir.

Yıl	Ay	Gün	Saat	Nem (%)	Basınç(hPa)	Sıcaklık (°C)	Yoğunluk (kg/m³)	Rüzgar Yönü (Derece)	10 m Rüzgar Hızı (m/s)	110 m Rüzgar Hızı (m/s)	Güç (kW)	Optimize Güç (kW)
2017	1	10	10	66	1009.0	9.4	1.2404	67.5	3.5	5.7	740.39	740.39
2017	1	10	11	45	1008.2	11.8	1.2297	90.0	5.6	9.0	3006.46	3006.46
2017	1	10	12	48	1008.0	12.7	1.2252	67.5	6.2	10.0	4065.13	4065.13
2017	1	10	13	54	1008.0	11.7	1.2294	67.5	6.3	10.2	4279.64	4279.64
2017	1	10	14	51	1007.9	11.9	1.2285	67.5	6.4	10.3	4483.40	4483.40
2017	1	10	15	56	1007.9	10.1	1.2364	45.0	8.5	13.7	10570.81	4650.00
2017	1	10	16	57	1008.5	9.9	1.2380	67.5	6.2	10.0	4107.60	4107.60
2017	1	10	17	54	1008.7	9.8	1.2389	112.5	5.1	8.2	2287.91	2287.91
2017	1	10	18	42	1008.3	10.6	1.2354	135.0	3.7	6.0	871.18	871.18
2017	1	10	19	44	1008.5	11.3	1.2324	135.0	6.0	9.7	3705.94	3705.94
2017	1	10	20	49	1008.7	11.6	1.2310	90.0	2.4	3.9	236.91	236.91

Şekil 1. Osmaniye İline Ait Düzenlenmiş Veri Setinden Bir Kesit

2.3. Geliştirilen Derin Öğrenme Modellerinin Mimari Yapısı

Keras ile optimizasyon algoritmaları temelli derin öğrenme modelleri geliştirebilmek için Sequential (Sıralı) API mimarisinden faydalanılmıştır. Bu yapı derin öğrenme modellerini hızlı ve kolay bir şekilde oluşturmak için kullanılan bir yöntemdir. Sıralı denmesinin sebebi modelin katmanlarının sıralı bir şekilde düzenlenmesinden kaynaklanmaktadır. Yani her katman bir önceki katmanın çıktısını alır ve bir sonraki katmana giriş olarak verir. Regresyon tabanlı uygulamalarda sıklıkla kullanılan bir yöntem olduğu için tercih edilmiştir (Gulli, Kapoor, & Pal, 2019).



Şekil 2. Derin Öğrenme Modellerinin Mimari Yapısı

Modellerde toplamda 6 adet tam bağlantılı yoğun (dense) katman ve her bir yoğun katmandan sonra %20'lik bir Dropout katmanı eklenmiştir. Dropout, aşırı öğrenmeyi önlemek için kullanılan bir düzenleme tekniği olarak bilinmektedir (Metin & Karasulu, 2022). Geliştirilen modellerde 5 tane gizli katman kullanılmıştır. Her gizli katmanda yüksek başarım oranı sebebiyle

GELU (Gaussian Error Linear Unit) aktivasyon fonksiyonundan faydalanılmıştır. GELU doğrusal olmayan bir aktivasyon fonksiyonudur ve özellikle derin öğrenme modellerinde performansı artırmak amacıyla yaygın olarak kullanılmaktadır. Bu yapı yumuşak geçişi ve negatif değerleri tamamen sıfırlamaması nedeniyle daha kararlı bir öğrenme süreci sunmaktadır (Hendrycks & Gimpel, 2016). Çıkış katmanında ise linear aktivasyon fonksiyonu tercih edilmiştir. Tercih edilen aktivasyon fonksiyonu sayesinde modelin sürekli bir değer tahmin etmesini sağlanmış olacaktır. Çalışmada kullanılan derin öğrenme modellerinin mimari yapısı Şekil 2'de gösterilmiştir.

3. DENEYSEL ÇALIŞMALAR

Optimizasyon algoritmalarının derin öğrenme uygulamaları üzerindeki etkisini analiz etmek amacıyla tahmin modelleri geliştirilmiştir. Regresyon tabanlı veri setinin seçilmesinin nedeni ise türevlenebilir olması ve sürekli kayıp fonksiyonları kullanılarak optimize edilebilmesinden kaynaklanmaktadır. Ayrıca varsayılan olarak tüm optimizasyon algoritmaları için uygulanabilirdir.

Keras ile geliştirilen modellerin her çalıştırıldığında, başlangıç ağırlıklarının değişmemesi, eğitim verilerinin karıştırılma sırasının farklılaşmaması ve dropout gibi işlemlerde hep aynı nöronların maskelenmesi amacıyla rastgelelik tohumu (random seed) olarak adlandırılan bir yapı koda eklenmiştir. Özellikle deneysel tekrarların önemli olduğu bilimsel araştırmalarda deneysel sonuçların rasgelelikten ayrıştırılarak doğruluğunun sağlanması temel bir gerekliliktir. Rastgelelik tohumu sayesinde modellerin her çalıştığında farklı sonuçların üretilmesinin önüne geçilmektedir (Colas, Sigaud, & Oudeyer, 2018).

Geliştirilen modellerde her bir optimizasyon algoritmasının tahmin performansına etkisinin değerlendirilmesinde MSE (Mean Squared Error), RMSE (Root Mean Squared Error) ve MAE (Mean Absolute Error) gibi hata metriklerinden yararlanılmıştır. MSE ile tahmin edilen değerler ile gerçek değerler arasındaki farkların karesinin ortalaması alınmaktadır. RMSE, MSE'nin karekökünün alınmasıyla elde edilmektedir. Yorumlamayı kolaylaştırmak açısından sık kullanılan bir metriktir. MAE ise tahmin edilen değerler ile gerçek değerler arasındaki farkın mutlak değerinin ortalaması olarak bilinmektedir. Her veriye özel işlem yaptığından aşırı değerlerden daha az etkilenmesi nedeniyle tercih edilmiştir. Geliştirilen modellerde hata metrikleri 0'a ne kadar yakın olursa tahmin edilen değerlerin gerçek değerlere yakınsadığı söylenebilmektedir. Ayrıca modelin hedef değişkenin toplam varyansının ne kadarı tarafında açıklandığını ortaya koyabilmek için Belirtme Katsayısı (Determination Coefficient-R^2) olarak bilinen performans metriğinden faydalanılmıştır. R^2 değeri yaygın olarak 0 ile 1 arasında olmakla birlikte bazı durumlarda negatif değerler de görülebilmektedir. 1'e yakın değerler modelin kullanılan veri setinin varyansını büyük ölçüde açıklayabildiğini gösterirken, 0'a yakın değerler modelin performansının yetersiz olduğuna işaret etmektedir (Montgomery, Runger, & Hubele, 2009).

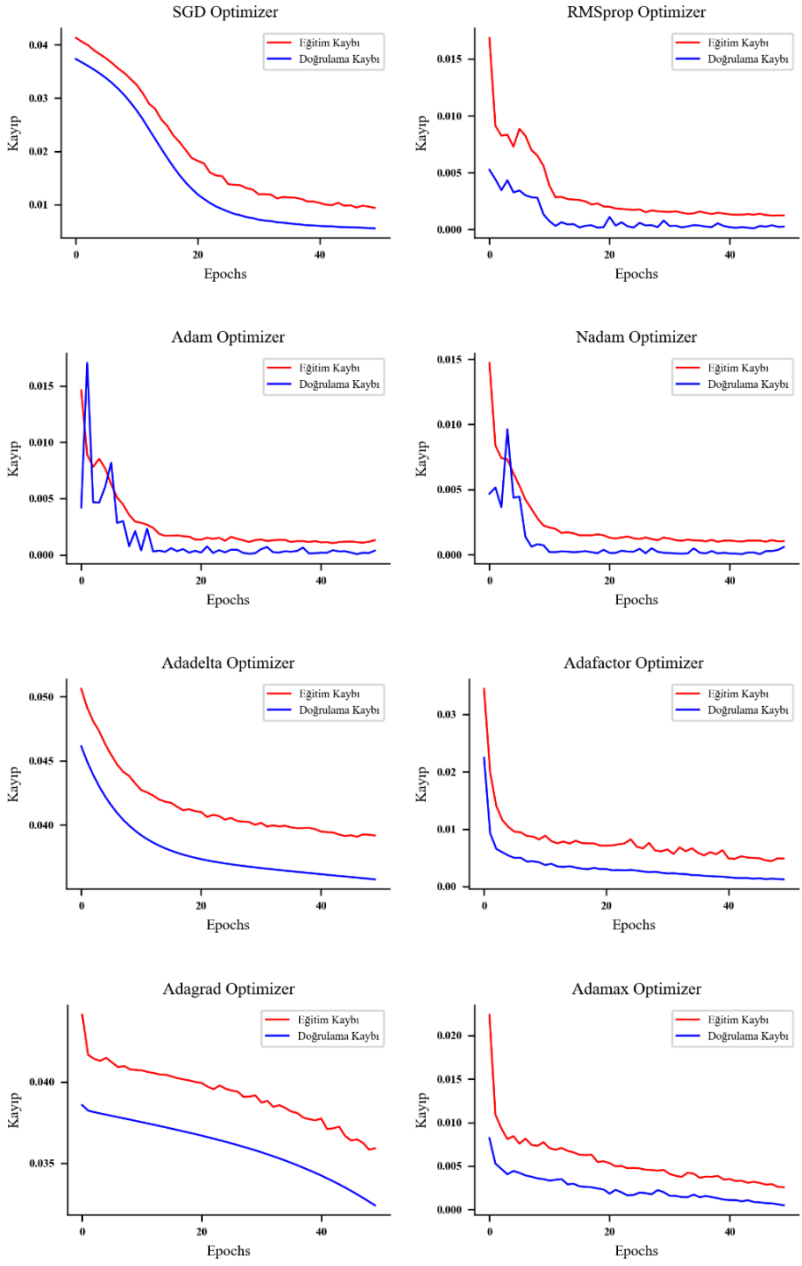
Modelleme çalışmasında kullanılmak üzere veri seti %80'i eğitim ve %20'si test olmak üzere rastgele ayrılmıştır. Eğitim sırasında yanlılığı engellemek maksadı ile Min-Max Scaler ölçeklendirme tekniğinden faydalanılmıştır. Bu sayede veri seti 0 ile 1 arasına ölçeklendirilmiş aynı zamanda verinin orijinal dağılımının şeklinin korunması sağlanmıştır. Modellerin eğitiminde ayırt ediciliği sağlaması açısından her bir optimizasyon algoritmasının eğitim süresi saniye cinsinden kaydedilmiştir. Ayrıca tüm modellerde yığın boyutu 16, epoch

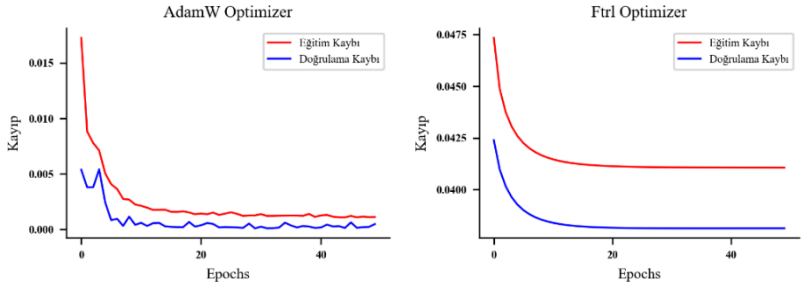
sayısı 50 olarak belirlenmiştir. Eğitim süreci sonunda geliştirilen optimizasyon algoritmaları temelli derin öğrenme modellerinin test verisi kullanılarak hesaplanan performans analizi Tablo 1'de ayrıntılı olarak verilmiştir.

Tablo 1. Optimizasyon Algoritması Temelli Derin Öğrenme Modellerinin Performans Analizi

Optimizasyon Algoritması	MSE	MAE	RMSE	R^2	Eğitim Süresi (sn)
Adam	0.0004	0.0130	0.0196	0.9900	147.8710
SGD	0.0056	0.0529	0.0746	0.8542	112.5484
RMSprop	**0.0002**	**0.0077**	**0.0156**	**0.9936**	**117.1958**
AdamW	0.0005	0.0108	0.0218	0.9875	155.2554
Adadelta	0.0358	0.1200	0.1891	0.0620	173.5880
Adagrad	0.0324	0.1158	0.1800	0.1500	151.5151
Adamax	0.0005	0.0084	0.0219	0.9875	147.7357
Adafactor	0.0012	0.0181	0.0353	0.9673	133.6262
Nadam	0.0006	0.0159	0.0245	0.9843	135.1226
Ftrl	0.0381	0.1278	0.1953	-0.0001	174.3156

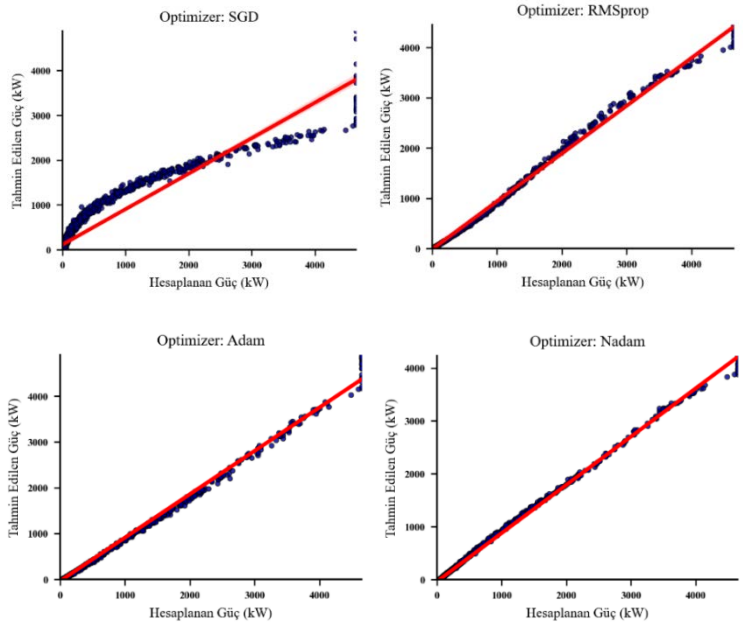
Tablo 1'den görüleceği üzere RMSprop optimizasyon algoritması temelli derin öğrenme modelli diğerlerine kıyasla Osmaniye ilinin rüzgar gücünü başarılı bir şekilde tahmin etmiştir. MSE, RMSE, MAE ve R^2 değerleri sırasıyla **0.0002**, **0.0156**, **0.0077** ve **0.9936** hesaplanmıştır. Hata metriklerinin 0'a, R^2 performans metriğinin de 1'e oldukça yakın çıkması modelin tahmin doğruluğunun gücünü göstermektedir. RMSprop optimizasyon algoritması kullanılarak geliştirilen modelin tahmin performansına etkisinin çok yüksek olduğu sonucuna varılabilir. Bu optimizasyon algoritmasını, Adam algoritması **MSE: 0.0004**, **RMSE: 0.0196**, **MAE: 0.0130** ve **R^2: 0.9900** değerleri ile takip etmektedir. En kötü performans ise **Ftrl**, **Adadelta** ve **Adagrad** optimizasyon algoritmalarını kullanarak geliştirilen modellerde elde edilmiştir. Her bir modelin eğitim ve doğrulama kayıp grafikleri (loss graphs) Şekil 3'te verilmiştir.

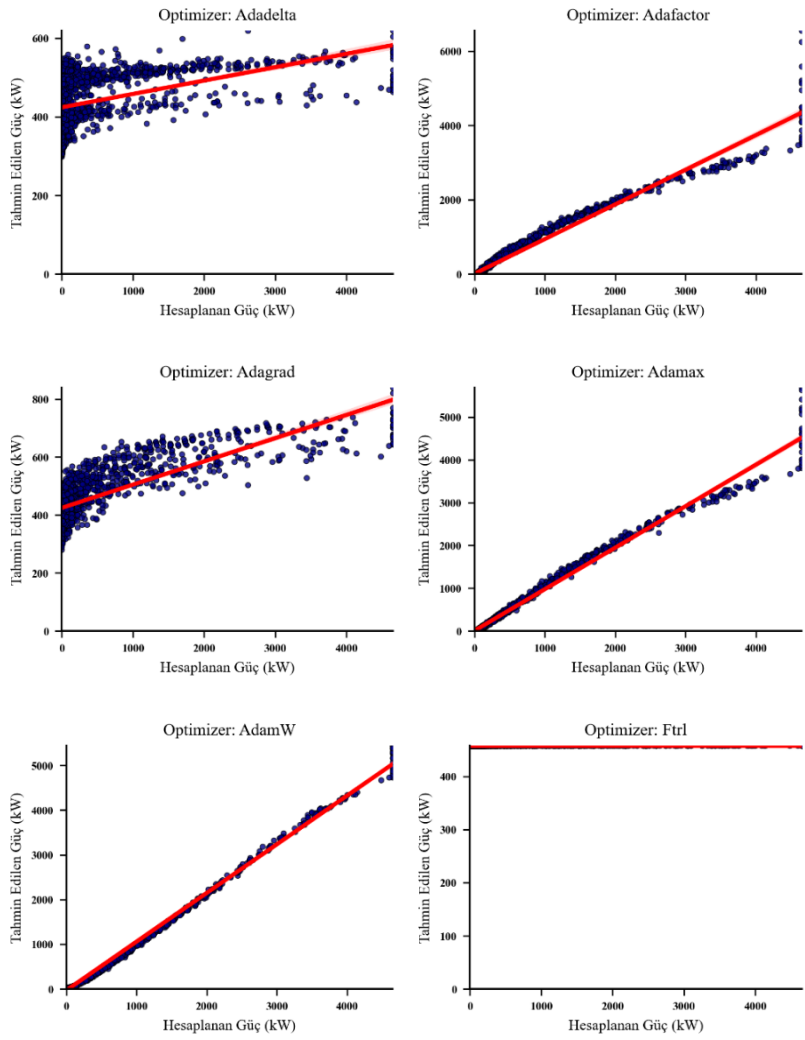

SGD Optimizer
RMSprop Optimizer
Adam Optimizer
Nadam Optimizer
Adadelta Optimizer
Adafactor Optimizer
Adagrad Optimizer
Adamax Optimizer
Eğitim Kaybı
Doğrulama Kaybı
Kayıp
Epochs



Şekil 3. Geliştirilen Modellerin Kayıp Grafikleri

RMSprop ile Adam algoritmasının eğitim sürelerinin kıyaslanmasında, RMSprop algoritması ile geliştirilen model, Adam algoritmasına göre yaklaşık **30.7** saniye gibi daha erken bir sürede öğrenmesini tamamlamıştır. Modellerin eğitim süreleri Tablo 1'den görüldüğü üzere en kısa süreli ikinci eğitim modelinin RMSprop olduğu anlaşılmaktadır. Osmaniye ilinin rüzgâr potansiyelinin belirlenmesinde tahmin edilen değerlerle ve gerçek değerler arasındaki dağılım grafikleri Şekil 4'te verilmiştir.





Şekil 4. Geliştirilen modellerin dağılım grafikleri

Dağılım grafiklerinden RMSprop, Adam, Nadam ve AdamW optimizasyon algoritmaları ile geliştirilen modellerin birbirlerine oldukça benzer olduğu anlaşılmaktadır. Tablo 1'de bu algoritmalarla geliştirilen modellerin R^2 değerleri yaklaşık olarak **0.99** bulunmuştur. Bu sonuç benzer dağılım grafiklerinin ortaya çıkmasında etkili olmuştur.

4. SONUÇ VE TARTIŞMA

Bu kitap bölümünde, son yıllarda oldukça fazla alanda kullanılan derin öğrenme uygulamalarının tahmin kesinliğinin arttırılmasında pek tercih edilmeyen bir yöntem olan optimizasyon algoritmalarının etkisi detaylı bir şekilde analiz edilmiştir. Çoğu uygulamalarda varsayılan olarak kullanılan ve birçok uygulayıcının dikkate dahi almadığı bir hiperparametre olan bu yapının model performansını önemli derecede etkilediği gösterilmiştir.

Optimizason algoritmalarının derin öğrenme modelleri üzerindeki etkisinin analiz edilmesinde regresyon tabanlı Osmaniye ili 2017 MGM verileri kullanılmıştır. Modelleme sürecinde derin öğrenme çalışmalarında yaygın olarak kullanılan on farklı optimizasyon algoritması belirlenmiş ve bu algortimaların özellikleri, avantaj ve dejavantajlı yönleri ayrıntılı bir biçimde teorik yönden ele alınmıştır. Ardından seçilen algoritmalar özelinde derin öğrenme tabanlı tahmin modelleri geliştirilerek deneysel çalışmalar yapılmıştır. Her bir algoritmanın tahmin modelleri üzerindeki etkisi karşılaştırmalı olarak analiz edilmiştir. Süreç sonunda optimizasyon algoritmalarının kullanılan veri seti ile alakalı olarak tahmin performansı üzerinde göz ardı edilemeyecek bir etkisinin olduğu anlaşılmıştır. Yenilenebilir enerji alanında herhangi bir ilin rüzgâr potansiyelinin derin öğrenme tabanlı tahmin edilmesinde RMSprop optimizasyon algoritmasının hem eğitim süresi hem de model performansı açısından başarılı bir şekilde uygulanabileceği ifade edilmiştir.

Derin öğrenme çalışmalarında, en optimal hiperparametrelerin belirlenmesi sürecinde, farklı optimizasyon algoritmalarının kullanılması performansı yüksek daha kararlı modellerin geliştirilmesini sağlayacaktır. Optimizasyon algoritmaları üzerinde yapılan çalışmalar arttıkça derin öğrenme

uygulamalarının performanslarının bir o kadar artması beklenmektedir. Gelecekte daha az işlem gücü ile daha yüksek performans seviyelerine ulaşan modeller geliştirilmesindeki anahtar güç, optimizasyon algoritmalarının iyileştirilmesinden geçmektedir.

KAYNAKÇA

Altun, S., & Talu, M. F. (2021). Derin sinir ağları için hiperparametre metodlarının ve kitlerinin incelenmesi. *Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi, 12*(2), 187-199.

Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade: Second edition* (pp. 437-478): Springer.

Bottou, L. (2010). *Large-scale machine learning with stochastic gradient descent.* Paper presented at the Proceedings of COMPSTAT'2010: 19th International Conference on Computational Statistics Paris France, August 22-27, 2010 Keynote, Invited and Contributed Papers.

Chollet, F. (2018). *Deep learning with Python*. Shelter Island, New York: Manning Publications.

Colas, C., Sigaud, O., & Oudeyer, P.-Y. (2018). How many random seeds? statistical power analysis in deep reinforcement learning experiments. *arXiv preprint arXiv:.08295*.

Dozat, T. (2016). Incorporating nesterov momentum into adam. *ICLR Workshop*.

Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research, 12*(7).

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1): MIT press Cambridge.

Gulli, A., Kapoor, A., & Pal, S. (2019). *Deep learning with TensorFlow 2 and Keras: regression, ConvNets, GANs, RNNs, NLP, and more with TensorFlow 2 and the Keras API*: Packt Publishing Ltd.

Hansen, M. (2015). *Aerodynamics of wind turbines*: Routledge.

Hendrycks, D., & Gimpel, K. (2016). Gaussian error linear units (gelus). *arXiv preprint arXiv:.08415*.

Hinton, G., Srivastava, N., & Swersky, K. (2012). Neural networks for machine learning. *Coursera, video lectures, 264*(1), 2146-2153.

Kelleher, J. D. (2019). *Deep learning*: MIT press.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436-444.

Li, H., Rakhlin, A., & Jadbabaie, A. (2023). Convergence of adam under relaxed assumptions. *Advances in Neural Information Processing Systems, 36*, 52166-52196.

Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., & Han, J. (2019). On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:.03265*.

Loshchilov, I., & Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:.05101*.

McMahan, H. B., Holt, G., Sculley, D., Young, M., Ebner, D., Grady, J., . . . Golovin, D. (2013). *Ad click prediction: a view from the trenches.* Paper presented at the Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining.

Meng, L., Ge, Z., Tian, P., An, B., & Gao, Y. (2023). *Deep FTRL-ORW: An Efficient Deep Reinforcement Learning Algorithm for Solving Imperfect Information Extensive-Form Games.* Paper presented at the 37th AAAI Conference on Artificial Intelligence.

Metin, B., & Karasulu, B. (2022). Derin Öğrenme Modellerini Kullanarak İnsan Retinasının Optik Koherans Tomografi Görüntülerinden Hastalık Tespiti. *Veri Bilimi, 5*(2), 9-19.

Montgomery, D. C., Runger, G. C., & Hubele, N. F. (2009). *Engineering statistics*: John Wiley & Sons.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., . . . Liu, P. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research, 21*(140), 1-67.

Robbins, H., & Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, 400-407.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:.04747*.

Shazeer, N., & Stern, M. (2018). *Adafactor: Adaptive learning rates with sublinear memory cost.* Paper presented at the International Conference on Machine Learning.

Siemens Gamesa Renewable Energy. SG 4.5-145: New SGRE turbine with the best-in-class LCoE >4 MW. Retrieved from https://it.wind-turbine.com/cms/storage/uploads/2021/11/16/siemensgamesaonshorewindturbinesg45145en_uid_6193dae36a732.pdf

Tieleman, T. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning, 4*(2), 26.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., . . . Azhar, F. (2023). Llama: Open and efficient foundation language models. *arXiv preprint arXiv:.13971*.

Yang, L., & Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing, 415*, 295-316. doi:https://doi.org/10.1016/j.neucom.2020.07.061

Yu, T., & Zhu, H. (2020). Hyper-parameter optimization: A review of algorithms and applications. *arXiv preprint arXiv:2003.05689v1*.

Zeiler, M. D. (2012). Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2021). Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM, 64*(3), 107-115.

Zou, F., & Shen, L. (2018). On the convergence of adagrad with momentum for training deep neural networks. arXiv preprint arXiv:.03408, 2(3), 5.

CLASSIFICATION OF FRUIT AND VEGETABLE IMAGES USING DEEP TRANSFER LEARNING MODELS

Mehmet BURUKANLI[1]

Davut ARI[2]

1. INTRODUCTION

The accurate and effective image processing of fruits and vegetables holds critical importance across numerous domains such as agriculture, the food industry, and robotic applications. In post-harvest processes particularly in the assessment of product quality, sorting, and storage the demand for automated and precise classification methods has been steadily increasing. In this context, fruits and vegetables are subjected to various classification criteria based on multiple distinguishing characteristics. From a physiological perspective, fruits are classified into two main categories: climacteric and non-climacteric. This distinction serves as a fundamental basis for monitoring their ripening processes and metabolic activities. On the other hand, one of the major challenges encountered in machine vision and robotic applications is the color similarity between fruits or vegetables and their surrounding background. This issue can significantly compromise the accuracy of color-based image processing algorithms. Furthermore, classification types such as growth patterns influenced by physiological factors

[1] Lecturer Dr., Bitlis Eren University, Rectorate, Department of Common Courses, mburukanli@beu.edu.tr, ORCID: 0000-0003-4459-0455.

[2] Research Assistant Dr., Bitlis Eren University, Faculty of Engineering and Architecture, Computer Engineering, dari@beu.edu.tr, ORCID: 0000-0001-6439-7957.

and commercial grading schemes are designed in accordance with the specific requirements of various applications. The integration of image processing techniques with machine learning and deep learning methods enhances classification performance by leveraging multiple features of fruits and vegetables, such as structural characteristics, color, and texture. Intelligent packaging systems also contribute to classification by utilizing advanced sensor and indicator technologies to monitor the freshness and quality status of products. These multidimensional classification approaches have emerged as essential tools for improving agricultural efficiency and delivering high-quality products to consumers(Lv et al. 2022)(He et al. 2023)(Parlak Sönmez and Kılıç 2024)(Ukwuoma et al. 2022).

Ghazal and others(Ghazal et al. 2021), conducted a comprehensive performance comparison in fruit classification by employing handcrafted visual features based on color, texture, and shape such as Hue, Color-SIFT, Haralick, DWT, and HOG in combination with traditional machine learning algorithms, including SVM, KNN, and BPNN. Following dimensionality reduction using PCA, the best results were achieved with BPNN and SVM. The study demonstrated the effectiveness of interpretable and cost-efficient classical approaches as viable alternatives to deep learning methods. Yuesheng and others(Yuesheng et al. 2021), optimized the GoogLeNet architecture for fruit and vegetable classification, achieving a threefold increase in training speed and improving accuracy to 98.82%. By incorporating the Swish activation function and DropBlock regularization layers, they enhanced the model's performance. The optimized model outperformed classical CNN architectures such as AlexNet, VGGNet, and ResNet18 in both accuracy and speed. Furthermore, real-time testing demonstrated that the model operates with accuracy reaching up to 99%. Hameed and others (Hameed, Chai, and Rassau 2021), proposed

an adaptive angular margin and cluster embedding-based method to overcome challenges such as intra-class variation and inter-class similarity in fruit and vegetable classification. This approach aims to compact intra-class samples while enhancing inter-class separability within the DCNN feature space. Experiments conducted with a ResNet50-based model achieved 99% accuracy across 15 classes along with high Silhouette scores indicating effective clustering performance. Moreover, the method demonstrated stability on imbalanced datasets and was deemed suitable for real-time applications. Hussain and others (Hussain et al. 2022), proposed a simple yet effective DCNN architecture for fruit and vegetable recognition under varying illumination and background conditions. Trained on a custom dataset comprising 10,000 images from the Gilgit-Baltistan region, the model achieved a classification accuracy of 96%. The system was trained using data with backgrounds removed through a split-and-merge algorithm, resulting in a low-complexity, fast-performing solution well-suited for real-world scenarios. Tapia-Mendez and others (Tapia-Mendez et al. 2023) developed a deep learning-based two-stage system aimed at reducing fruit and vegetable waste. Utilizing the MobileNetV2 architecture, the system performs classification of up to 32 fruit and vegetable types and distinguishes between fresh and spoiled produce across six classes. Models trained with transfer learning and data augmentation techniques achieved accuracies of 97.86% and 100%, respectively. Operating on their self-constructed datasets, the system's applicability in supermarkets and agricultural sectors was emphasized, with future goals including real-time processing and multi-object detection support. Varghese and others (Varghese et al. 2021), developed a mobile, real-time grading system that evaluates the quality of fruits and vegetables and predicts their shelf life. Supported by a MobileNet-based CNN, the system employs transfer learning and operates on a custom image dataset. Functioning through an Android application, it

predicts product quality and shelf life with 70% accuracy without requiring network connectivity. While the application's portability and cost-effectiveness are highlighted, its limited accuracy and challenges in distinguishing between similar products are identified as shortcomings to be addressed in future work. Gill and others (Gill et al. 2022) proposed a hybrid system based on image enhancement, segmentation, and deep learning (CNN-RNN-LSTM) to address shortcomings in accuracy and quantitative analysis in fruit classification. The approach involved image enhancement using Type-II fuzzy logic, segmentation via TLBO-MCET, CNN-based feature extraction, followed by labeling with RNN and classification using LSTM. Experiments on 300 real-time images achieved an accuracy of 96.08%. The proposed method outperformed other models such as SVM, FFNN, and ANFIS. The system demonstrated success in both coarse and fine-grained classification, distinguishing itself through multi-object classification capability and robust feature extraction. Latha and others (Latha et al. 2022) proposed a deep learning approach based on YOLOv4-tiny for real-time fruit and vegetable identification in vegetable markets. The system aimed to perform fast and accurate multi-class object detection by collecting video data through IoT devices. Trained on a dataset of 9,800 images spanning 12 classes and annotated with Roboflow, the model achieved 51% mAP, 0.63 precision, 0.41 recall, and a 0.50 F1-score after 15,700 epochs. Higher accuracy was observed in certain classes such as bottle gourd (77.29% AP) and cauliflower (66.66% AP). The model's low inference time of approximately 18 ms emphasized its suitability for embedded devices, while performance degradation on blurry images was identified as a primary limitation. Zhang and others (Zhang et al. 2021) developed a method enabling robots to recognize the firmness levels of fruits and vegetables to grasp them without causing damage. Using an experimental platform equipped with a WSG 50 manipulator and tactile array sensors, the study

collected 400 datasets from four different product classes (apple, kiwi, orange, tomato). The resulting tactile time series were classified using KNN and SVM classifiers after dimensionality reduction with PCA, with the best performance achieved by the PCA–SVM model at 94.27% accuracy. Online recognition experiments also reached 90% accuracy. Uncertainty between classes B and C was identified as the main limitation of the model. This work lays the foundation for precise grasping strategies in nondestructive agricultural robotics. Mimma and others (Mimma et al. 2022) conducted research on fruit classification and detection using deep learning. They applied data augmentation and domain adaptation techniques on both publicly available and proprietary datasets. The study achieved fast and accurate fruit detection with YOLOv7, and high-accuracy classification using ResNet50 and VGG16. Additionally, the system was deployed as both web and mobile applications. This work provides significant contributions toward developing practical and real-time fruit recognition solutions. Min and others (Min et al. 2023) proposed the Multi-Scale Attention Network (MSANet) model, which integrates multi-scale attention mechanisms from different CNN layers to address the challenges of fruit recognition. The study demonstrated the model's superior performance using four distinct fruit datasets, highlighting that the hybrid attention block—simultaneously processing spatial and channel attention—significantly improved recognition accuracy. MSANet outperformed other methods on datasets simulating real-world conditions and enabled practical use through mobile application integration.

In this study, fruit and vegetable classifications were performed using alexnet, densenet121, resnet50, vgg16 and vit_b_16 models. 7000 training, 1500 validation and 1500 testing data were used in the study. Deep transfer learning models were compared with each other in terms of accuracy values on the

testing dataset. When the obtained results were examined in detail, the densenet121 model achieved better results than other models with an accuracy value of 91.13%, but the alexnet model achieved worse results than other models with an accuracy value of 85.13%. These results indicate that deep transfer learning models are quite successful in fruit and vegetable classifications.

2. FRUIT AND VEGETABLE DATASET AND DEEP TRANSFER LEARNING BASED MODELS

In this section, the fruit and vegetable dataset and deep transfer learning based architectures used in this study are explained in detail.

2.1. Fruit and vegetable dataset

The fruit and vegetable dataset consists of 10 classes and a total of 10000 images (Nguyen 2025). These classes are: “Tomato”, “Potato”, “Carrot”, “Banana”, “Broccoli”, “Eggplant”, “Corn”, “Pineapple”, “Orange”, “Asparagus”. In addition, each class consists of 1000 images (750 training, 150 validation, 150 testing). In this study, 7000 images were used for training, 1500 images for validation and 1500 images for testing in the classification phase of deep learning models. The details of this fruit and vegetable datasets are shown in Table 1.

Table 1. The details of this fruit and vegetable datasets (Nguyen 2025)

Class Name	Amount of dataset		
	Training	Validation	Testing
Tomato	700	150	150
Potato	700	150	150
Carrot	700	150	150
Banana	700	150	150
Broccoli	700	150	150
Eggplant	700	150	150
Corn	700	150	150
Pineapple	700	150	150
Orange	700	150	150
Asparagus	700	150	150
Total	7000	1500	1500

Figure 1 shows examples of images for each class (Tomato, Potato, Carrot, Banana, Broccoli, Eggplant, Corn, Pineapple, Orange, Asparagus) in the fruit and vegetable dataset.

Figure 1. image examples for each class in the fruit and vegetable dataset (Nguyen 2025)

2.2. Deep transfer learning models

In this study, the details of alexnet, densenet121, resnet50, vgg16 and vision transformer (vit_b_16) deep transfer learning models used for the classification of fruits and vegetables are explained.

2.2.1. alexnet model

The alexnet model is a CNN-based model trained on the ImageNet dataset consisting of 1000 classes(Krizhevsky, Sutskever, and Hinton 2012). It was quite successful in the competition held in 2012.

2.2.2. densenet121 model

DenseNet121 model is a CNN based model presented for object detection, classification and segmentation. By designing the DenseNet121 model, the vanishing gradient problem has been eliminated (Huang et al. 2017).

2.2.3. resnet50 model

The resnet50 model, trained over the ImageNet dataset consisting of 1000 classes, is based on a CNN model .It was quite successful in the competition held in 2015 (He et al. 2016).

2.2.4. vgg16 model

The vgg16 model, trained on the ImageNet dataset consisting of 1000 classes, consists of 13 convolutional layers, 5 maxpolling and 3 fully-connected layers. The vgg16 model, which uses 3x3 filters in each convolutional layer, is based on a CNN model (Simonyan and Zisserman 2015).

2.2.5. vit_b_16 model

The vit (vit_b_16) model is a transformer-based architecture designed for NLP tasks. The Vit model is widely

used in classification, segmentation and object detection tasks (Dosovitskiy et al. 2021).

3. RESULTS

In this study, each deep transfer learning model (alexnet, densenet121, resnet50, vgg16 and vit_b_16) was selected as epoch number = 60, learning rate = 0.0001, optimizer algorithm = AdamW and batch size = 128 during training for the classification of fruit and vegetable images. True Positive (TP), True Negative (TN), False Negative (FN), False Positive (FP), True Negative (TN) values were used to calculate the performance value of each model. The calculation of the accuracy value is shown in Equation (1)(Burukanli and Ari 2025).

$$Accuracy = \frac{TP+TN}{TP+FN+FP+TN} \quad (1)$$

The accuracy values of deep transfer learning based models on the testing dataset are shown in Table 2.

Table 2. The accuracy values of deep transfer learning based models on the testing dataset

Model	**Accuracy (%)**
alexnet	85.13
densenet121	**91.13**
resnet50	85.80
vgg16	88.20
vit_b_16	86.47

As seen in Table 2, the densenet121 model achieved the best value among deep transfer learning models with an accuracy value of 91.13%. On the other hand, the alexnet model achieved worse results than the other models with an accuracy value of 85.13%. The confusion matrix obtained by the densenet121

model on the testing data set in the classification of fruits and vegetables is shown in Figure 2.

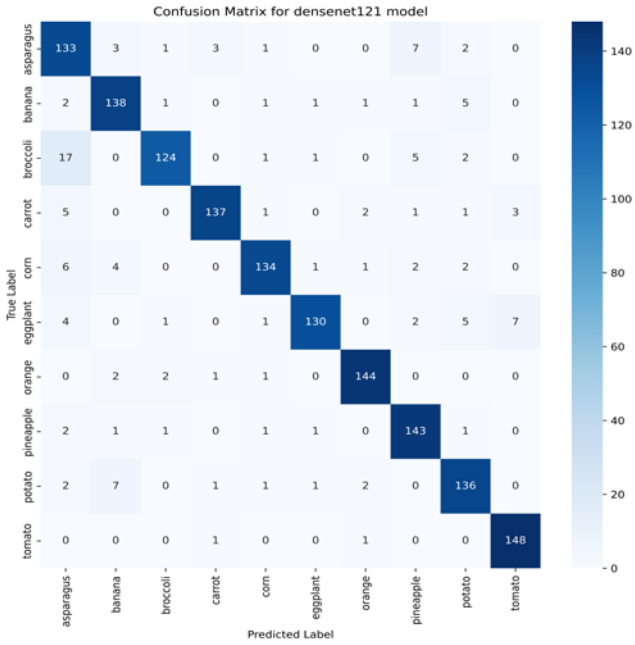


Figure 2. The confusion matrix obtained by the densenet121 model on the testing dataset.

When Figure 2 is examined in detail, we can observe how many test samples of each class are correctly classified by the densenet121 model. For example, we can see that the densenet121 model correctly classified 148 samples out of 150 test samples in the tomato class. This shows how successful the densenet121 model is in classifying fruits and vegetables.

4. CONCLUSION

In this work, fruit and vegetable classifications were performed using alexnet, densenet121, resnet50, vgg16 and vit_b_16 models. The dataset used in the work comprise 7000

training, 1500 validation and 1500 testing data. The performance values obtained by the deep transfer learning architectures on the testing dataset were also compared with each other. As a result of the study, the densenet121 model obtained better results than other models with an accuracy value of 91.13%, while the alexnet model obtained worse results than other models with an accuracy value of 85.13%. In the next study, we plan to increase the performance rates of the deep transfer learning architectures preferred in this study and compare them with the SOTA models.

REFERENCES

Burukanli, Mehmet, and Davut Ari. 2025. “Brain Cancer Prediction Using Deep Transfer Learning Models.” Pp. 184–92 In *Ases Ix. International Scientific Research Congress*. Adıyaman, Turkey.

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. “An Image Is Worth 16X16 Words: Transformers for Image Recognition At Scale.” *ICLR 2021 - 9th International Conference on Learning Representations*.

Ghazal, Sumaira, Waqar S. Qureshi, Umar S. Khan, Javaid Iqbal, Nasir Rashid, and Mohsin I. Tiwana. 2021. “Analysis of Visual Features and Classifiers for Fruit Classification Problem.” *Computers and Electronics in Agriculture* 187(February):106267. doi: 10.1016/j.compag.2021.106267.

Gill, Harmandeep Singh, Ganpathy Murugesan, Baljit Singh Khehra, Guna Sekhar Sajja, Gaurav Gupta, and Abhishek Bhatt. 2022. “Fruit Recognition from Images Using Deep Learning Applications.” *Multimedia Tools and Applications* 81(23):33269–90. doi: 10.1007/s11042-022-12868-2.

Hameed, Khurram, Douglas Chai, and Alexander Rassau. 2021. “Class Distribution-Aware Adaptive Margins and Cluster Embedding for Classification of Fruit and Vegetables at Supermarket Self-Checkouts.” *Neurocomputing* 461:292–309. doi: 10.1016/j.neucom.2021.07.040.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. “Deep Residual Learning for Image Recognition.”

Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2016-Decem:770–78. doi: 10.1109/CVPR.2016.90.

He, Xu, Yijing Pu, Luyao Chen, Haitao Jiang, Yan Xu, Jiankang Cao, and Weibo Jiang. 2023. "A Comprehensive Review of Intelligent Packaging for Fruits and Vegetables: Target Responders, Classification, Applications, and Future Challenges." *Comprehensive Reviews in Food Science and Food Safety* 22(2):842–81. doi: 10.1111/1541-4337.13093.

Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. 2017. "Densely Connected Convolutional Networks." *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017* 2017-Janua:2261–69. doi: 10.1109/CVPR.2017.243.

Hussain, Dostdar, Israr Hussain, Muhammad Ismail, Amerah Alabrah, Syed Sajid Ullah, and Hayat Mansoor Alaghbari. 2022. "A Simple and Efficient Deep Learning-Based Framework for Automatic Fruit Recognition" edited by A. M. Khalil. *Computational Intelligence and Neuroscience* 2022:1–8. doi: 10.1155/2022/6538117.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. 2012. "ImageNet Classification with Deep Convolutional Neural Networks." *Advances in Neural Information Processing Systems 25*.

Latha, R. S., G. R. Sreekanth, R. Rajadevi, S. K. Nivetha, K. Ajith Kumar, V. Akash, S. Bhuvanesh, and Pon Anbarasu. 2022. "Fruits and Vegetables Recognition Using YOLO." *2022 International Conference on Computer Communication and Informatics, ICCCI 2022* 1–6. doi: 10.1109/ICCCI54379.2022.9740820.

Lv, Jidong, Hao Xu, Liming Xu, Ling Zou, Hailong Rong, Biao Yang, Liangliang Niu, and Zhenghua Ma. 2022. "Recognition of Fruits and Vegetables with Similar-color Background in Natural Environment: A Survey." *Journal of Field Robotics* 39(6):888–904. doi: 10.1002/rob.22074.

Mimma, Nur E. Azni., Sumon Ahmed, Tahsin Rahman, and Riasat Khan. 2022. "Fruits Classification and Detection Application Using Deep Learning." *Scientific Programming* 2022. doi: 10.1155/2022/4194874.

Min, Weiqing, Zhiling Wang, Jiahao Yang, Chunlin Liu, and Shuqiang Jiang. 2023. "Vision-Based Fruit Recognition via Multi-Scale Attention CNN." *Computers and Electronics in Agriculture* 210(March):107911. doi: 10.1016/j.compag.2023.107911.

Nguyen, Che Dinh. 2025. "Fruits and Vegetables Image Dataset." Retrieved (https://www.kaggle.com/datasets/chinhde/vegetable-classification).

Parlak Sönmez, Demet, and Şafak Kılıç. 2024. "Deep Learning for Automatic Classification of Fruits and Vegetables: Evaluation from the Perspectives of Efficiency and Accuracy." *Türkiye Teknoloji ve Uygulamalı Bilimler Dergisi* 5(2):151–71. doi: 10.70562/tubid.1520357.

Simonyan, Karen, and Andrew Zisserman. 2015. "Very Deep Convolutional Networks for Large-Scale Image Recognition." *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings* 1–14.

Tapia-Mendez, Enoc, Irving A. Cruz-Albarran, Saul Tovar-Arriaga, and Luis A. Morales-Hernandez. 2023. "Deep Learning-Based Method for Classification and Ripeness

Assessment of Fruits and Vegetables." *Applied Sciences (Switzerland)* 13(22). doi: 10.3390/app132212504.

Ukwuoma, Chiagoziem C., Qin Zhiguang, Md Belal Bin Heyat, Liaqat Ali, Zahra Almaspoor, and Happy N. Monday. 2022. "Recent Advancements in Fruit Detection and Classification Using Deep Learning Techniques" edited by D. M. Khan. *Mathematical Problems in Engineering* 2022:1–29. doi: 10.1155/2022/9210947.

Varghese, Renju Rachel, Pramod Mathew Jacob, S. Sooraj, Daniel Mathew Ranjan, Jino Cherian Varughese, and Hegsymol Raju. 2021. "Detection and Grading of Multiple Fruits and Vegetables Using Machine Vision." *2021 8th International Conference on Smart Computing and Communications: Artificial Intelligence, AI Driven Applications for a Smart World, ICSCC 2021* 85–89. doi: 10.1109/ICSCC51209.2021.9528165.

Yuesheng, Fu, Song Jian, Xie Fuxiang, Bai Yang, Zheng Xiang, Gao Peng, Wang Zhengtao, and Xie Shengqiao. 2021. "Circular Fruit and Vegetable Classification Based on Optimized GoogLeNet." *IEEE Access* 9:113599–611. doi: 10.1109/ACCESS.2021.3105112.

Zhang, Zhen, Jun Zhou, Zhenghong Yan, Kai Wang, Jiamin Mao, and Zizhen Jiang. 2021. "Hardness Recognition of Fruits and Vegetables Based on Tactile Array Information of Manipulator." *Computers and Electronics in Agriculture* 181(July 2020):105959. doi: 10.1016/j.compag.2020.105959.

MOBİL UYGULAMA GELİŞTİRME: ANKET HAZIRLAMA UYGULAMASI

Onur KASAP[1]

Serpil TÜRKYILMAZ[2]

1. GİRİŞ

Mobil uygulamalar, günümüz teknolojisinin ayrılmaz bir parçası haline gelmiştir. Akıllı telefonlar sayesinde fatura ödeme, bankacılık işlemleri, haber takibi, oyun oynama, sosyal medya kullanımı ve hatta araç kontrolü gibi birçok işlem kolayca gerçekleştirilebilmektedir. Bu dönüşüm, sadece bireysel yaşamı değil, aynı zamanda eğitim, sağlık, turizm ve ticaret gibi çeşitli sektörleri de derinden etkilemektedir.

Cep telefonlarının evrimi 1973 yılında Martin Cooper'ın Motorola'da gerçekleştirdiği ilk cep telefonu icadıyla başlamış, ardından 2007 yılında Apple'ın iPhone ile yaptığı devrimsel atılım, mobil teknolojileri yeni bir boyuta taşımıştır. Bu gelişmeleri takip eden Android ve iOS işletim sistemleri, kullanıcı dostu arayüzleri ve geniş uygulama ekosistemleri ile mobil cihazları adeta taşınabilir bilgisayarlara dönüştürmüştür. Özellikle Android işletim sistemi, açık kaynak yapısı, yaygın geliştirici ağı ve geniş pazar payı ile öne çıkmakta; iOS ise güvenlik ve performans avantajlarıyla dikkat çekmektedir (Ensonshaber, 2025).

[1] Bilecik Şeyh Edebali Üniversitesi, Fen Fakültesi, İstatistik ve Bilgisayar Bilimleri, Lisans Öğrencisi, onurkasapdev@gmail.com, ORCID: 0009-0001-8985-5376.

[2] Prof. Dr., Bilecik Şeyh Edebali Üniversitesi, Fen Fakültesi, İstatistik ve Bilgisayar Bilimleri Bölümü, serpil.turkyilmaz@bilecik.edu.tr, ORCID: 0000-0002-7193-4148.

Çeşitli araştırmalar, mobil uygulamaların günlük yaşam üzerindeki etkisini açıkça ortaya koymaktadır. Örneğin, Çınar ve Bilici (2020), kullanıcıların mobil cihazlarda harcadıkları zamanın yaklaşık %80'inin uygulamalara ayrıldığını belirtmiştir. Özdamar Keskin ve Kılınç (2015) ise genç bireylerin mobil cihaz kullanımındaki artışı vurgulayarak, bu teknolojilerin erken yaşta yaygınlaştığını göstermiştir. Büyükgöze (2019) ve Uslu vd. (2020), iş dünyasında mobil uygulamalara yapılan yatırımların rekabet gücü ve gelir artışı sağladığını öne sürmektedir.

Mobil uygulamaların özellikle eğitim ve sağlık gibi alanlarda önemli katkılar sunduğu da farklı çalışmalarla ortaya konmuştur. Arista ve Kuswando (2018), eğitim amaçlı geliştirilen uygulamaların yüksek oranda olumlu geri bildirim aldığını belirtmiştir. Şencan Karakuş (2021) ise psikolojik danışmanlıkta kullanılan mobil uygulamaların, tedavi sürecine katkı sağlayabileceğini ifade etmiştir. Ayrıca Kopmaz ve Arslanoğlu (2018), sağlık uygulamalarının kullanımındaki dramatik artışa da dikkat çekmiştir.

Mobil anket uygulamaları da bu dönüşümün önemli bir parçası haline gelmiştir. Geleneksel anket yöntemlerinin yerini alan dijital anketler, hız, erişilebilirlik ve maliyet açısından önemli avantajlar sunmaktadır. ITU (Uluslar arası Telekominasyon Birliği' nin (2023) verilerine göre, dünya genelinde 5,4 milyar kişi internet erişimine sahiptir. GSMA Intelligence (2024) ise dünya nüfusunun %69,4'ünün mobil cihaz kullandığını belirtmiştir. Bu veriler, dijital anketlerin geniş kitlelere ulaşabilme potansiyelini ve mobil cihazların bu alandaki stratejik rolünü de gözler önüne sermektedir. Regmi vd. (2016) gibi araştırmacılar, web anketlerinin özellikle ulaşılması güç topluluklara erişimde etkili olduğunu vurgulamaktadır.

Bu amaçla çalışmada, mobil cihazlarda en yaygın kullanılan işletim sistemi olan Android üzerinde geliştirilen bir

anket hazırlama uygulaması ile bu alandaki literatüre katkı sunulması amaçlanmaktadır.

2. MOBİL UYGULAMA GELİŞTİRME AŞAMALARI

Mobil uygulama geliştirme; akıllı telefonlar, tabletler, araç içi sistemler (CarPlay, Android Auto), akıllı saatler gibi cihazlara yönelik yazılım üretimini kapsayan bir mühendislik sürecidir (AppMaster, 2023). Geliştirme süreci genellikle bir fikirle başlamaktadır. İzleyen kısımda süreçlerle ilgili özet bilgiler sunulmaktadır.

2.1. Fikir ve Algoritma

Fikir, mevcut bir problemi çözmeye yönelik yaratıcı bir başlangıçtır. Bu aşamada, uygulamanın işleyişini belirleyen bir algoritma oluşturulmaktadır. Algoritma, sürecin yönünü belirleyen temel bir yapıdır. Derleyiciler sayesinde yazılan kod, hedef cihazlara uygun hale getirilmektedir. Derinlemesine sistem erişimi gerektirmeyen projelerde çapraz platform çözümleri tercih edilebilmektedir.

2.2. Geliştirme Platformları

Bu bölümde mobil uygulama geliştirme platformlarından bahsedilmektedir.

2.2.1. Android

Android, Google tarafından geliştirilen açık kaynaklı bir işletim sistemidir (Türk, 2014). 2023 yılının ikinci çeyreği itibarıyla %70,89'luk pazar payına sahiptir (MOBİSAD, 2023). Android Studio ise en yaygın geliştirme ortamıdır. Java ile başlayan geliştirme süreci günümüzde daha çok Kotlin diliyle yürütülmektedir. Jetpack Compose, XML'e alternatif modern bir UI kütüphanesidir.

2.2.2. iOS

Apple tarafından geliştirilen iOS, kapalı kaynaklı ve yüksek güvenlikli bir sistemdir (Çiloğlu vd., 2021). %28,36'lık pazar payına sahiptir. Geliştirme ortamı Xcode; kullanılan programlama dili ise Swift'tir. Swift, Objective-C'ye kıyasla daha sade bir yapıya sahiptir.

2.2.3. Çapraz Platformlar

Farklı platformlar için ayrı kod yazma ihtiyacını azaltan çapraz platform çözümleri, maliyeti düşürmektedir. Google destekli Flutter ve Microsoft'un .NET MAUI araçları, hem mobil hem de masaüstü/web uygulamaları geliştirmeye olanak tanımaktadır. Ancak bu çözümler, performans açısından yerel uygulamalara göre bazı sınırlamalara da sahiptir.

2.3. Yayınlama Süreci

Uygulama geliştirme tamamlandıktan sonra yayın sürecine geçilmektedir.

Google Play Store: Geliştirici hesabı 25$'dır, tek seferliktir. Yayın süreci daha hızlı ve esnektir.

App Store: Yıllık 99$ ücretlidir. Yayın öncesi daha sıkı güvenlik kontrolleri uygulanmaktadır.

Bu farklılıklar, bağımsız geliştiricileri genellikle Android'e yönlendirmektedir.

2.4. Android – iOS Karşılaştırması

Tablo 1' de Android ve iOS özellik karşılaştırmaları verilmiştir.

Tablo 1. Android - iOS Karşılaştırması

Özellik	Android	iOS
Geliştirme Dili & Ortam	Java/Kotlin - Android Studio	Swift - Xcode
Kaynak Kodu Yapısı	Açık kaynak	Kapalı kaynak
Yayın Ücreti	Tek seferlik 25$	Yıllık 99$
Yayın Süreci	Esnek ve hızlı	Sıkı denetim, daha uzun süre
Uygulama Sayısı (2017 yılı için) (Çiloğlu, 2021)	3,6 milyon	2,6 milyon
Güvenlik	Daha esnek	Daha sıkı ve güvenli

3. GELİŞTİRİLEN UYGULAMANIN KURULUMU

Bu çalışmada Anket Hazırlama Uygulaması Android platformu için geliştirilmiştir. Bunun için bu geliştirme ortamının öncelikle yüklenmesi gerekmektedir.

3.1. Android Studio Kurulumu

Android uygulama geliştirilirken teknik bakımdan ilk adım, bilgisayara Android Studio programının kurulmasıdır. Kurulumun adımları ise sırasıyla aşağıda verilmiştir:

- İlk adım https://developer.android.com/studio?hl=tr adresinden Android Studio'yu indirmektir.
- Dosya indirildikten sonra kurulum tamamlanır.
- Kurulum bittikten sonra ekrana gelen diğer kurulumun da başlatılması önemlidir. Bu kurulumda gerekli Android SDK, Emulator gibi önemli bileşenlerin olduğu unutulmayıp eksiksiz bir kurulum sağlanması gerekmektedir.

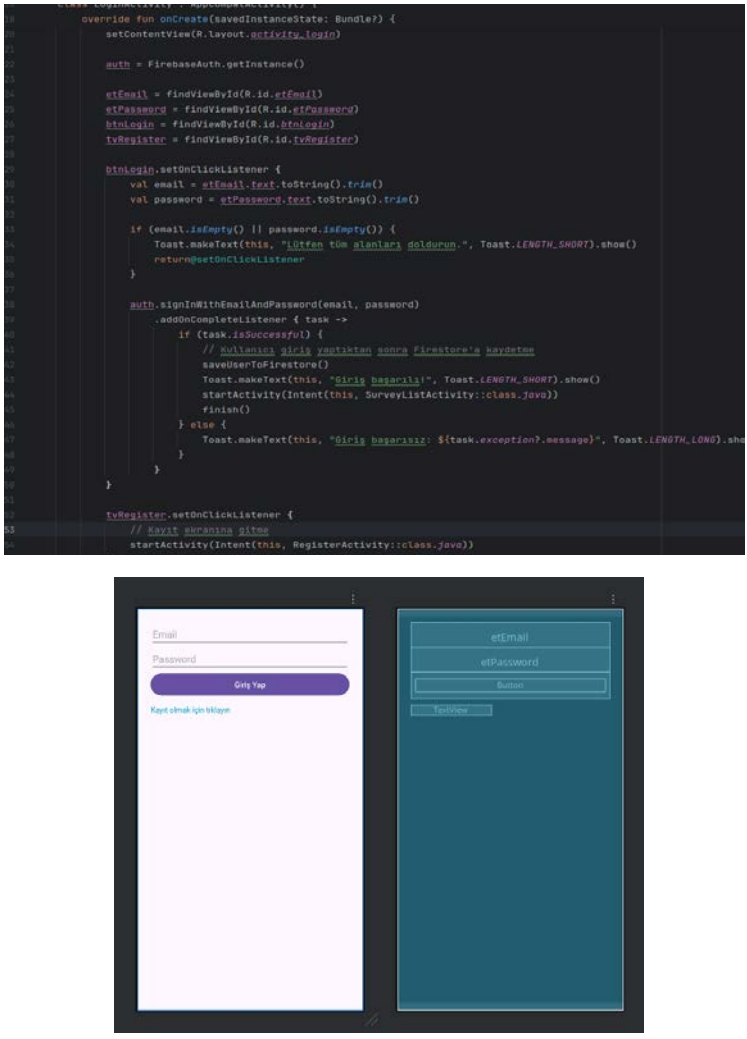
- Bu kurulumların tümü tamamlandığında Android Studio kullanıma hazır olup giriş ekranı ekrana gelecektir. Bu aşamadan sonra geliştirici, fikrine bağlı olarak Android bir uygulama hazırlayabilecektir.

3.2. Anket Hazırlama Uygulaması Geliştirme

Proje, veri tabanı olarak Google'ın geliştirdiği Firebase Studio ile geliştirilmiştir. Bu seçimin sebebi, kolaylık ve güvenliktir.

Çalışmada anket hazırlama uygulaması için ilk olarak giriş-kayıt ekranı, daha sonra sırasıyla ana sayfa ekranı, anket görüntüleme ekranı, anket oluşturma ekranı, anket cevaplama ekranı, anket cevap ekranı oluşturulmuştur. Aşağıda oluşturulan ekranların görüntüleri ve bazı önemli ekranların kodları adım adım paylaşılmıştır.

Uygulama, "Login" ekranı ile başlamaktadır. Kullanıcı uygulamaya kayıtlı ise direkt giriş yapabilecek iken, kayıtlı değil ise "Kayıt olmak için tıklayın" butonu ile yeni kayıt oluşturabilmektedir. Kullanıcı uygulamaya girer girmez veritabanında kullanıcı için ekstra bir kayıt daha olmaktadır. Bunun sebebi ise anket oluştururken uygulamaya en az bir kere girmiş kullanıcıların ankete erişim sağlayabilmeleridir. Şekil 1' de geliştirilen uygulamanın "login ekranı" için komutlar ve ekranın görüntüsü verilmektedir.

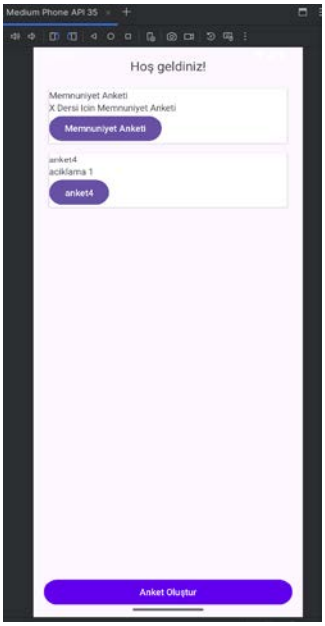


Şekil 1. Anket Hazırlama Uygulaması Login Ekranı

Kullanıcıya daha önceden tanımlı anketler varsa Anasayfa ekranında bu anketler de listelenmektedir. Kullanıcı tanımlı olan anketlerden herhangi birini seçip anketi tamamlayabilmektedir. Aynı zamanda da anket oluşturmak isterse alttaki "Anket Oluştur" butonu ile anket oluşturabilmektedir. Bu durum ileride yapılabilecek güncellemeler ile birlikte sadece belli mailler ile girmiş kişilerin anket oluşturmasını sağlama yönünde de

geliştirilebilmektedir. Şekil 2' de Anket Anasayfa kodları ekran görüntüsü olarak verilmiştir.

Şekil 3.2 Anket Hazırlama Uygulaması Ana Sayfa Kodları

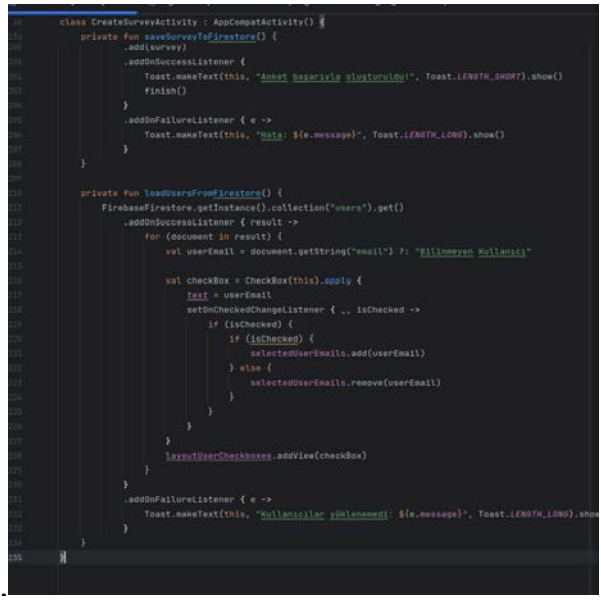


Şekil 3.2 Anket Hazırlama Uygulaması Ana Sayfa Ekranı

Şekil 4-5' de anket hazırlama uygulamasının anket hazırlama ortamı için kodlar ve ekran görüntüsü verilmiştir.

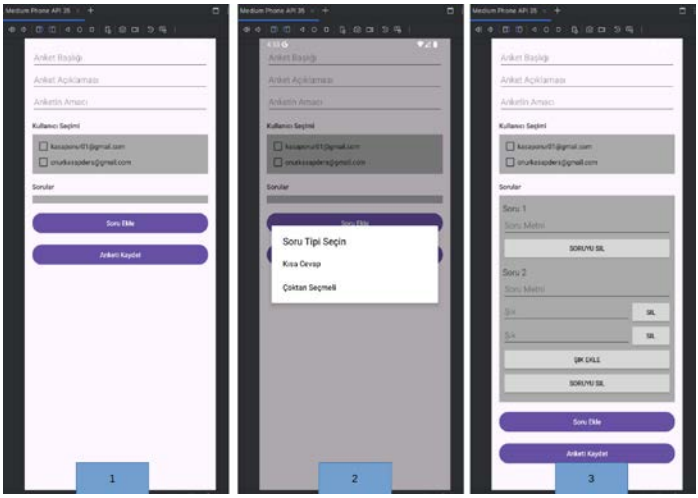
Geliştirilen uygulamada, yeni bir anket oluşturmak için kullanıcı "Anketi Oluştur" butonuna tıkladıktan sonra sırasıyla hazırlayacağı anket başlığını, anketin açıklamasını ve anketin amacını girmek zorundadır. Daha sonra anket katılımcılarının mailleri seçilmelidir. Ardından anket madde/sorularına geçilebilmektedir (Şekil 5(1)).

Geliştirilen uygulamada girilebilecek iki tip soru ortamı söz konusudur. Bunlar cevaplaması metin olacak şekilde sorulan açık uçlu sorular ve çoktan seçmeli sorulardır. Kullanıcı hazırlamak istediği ankete göre her soruda bu iki soru tipini de kullanabilmektedir (Şekil 5(2)).



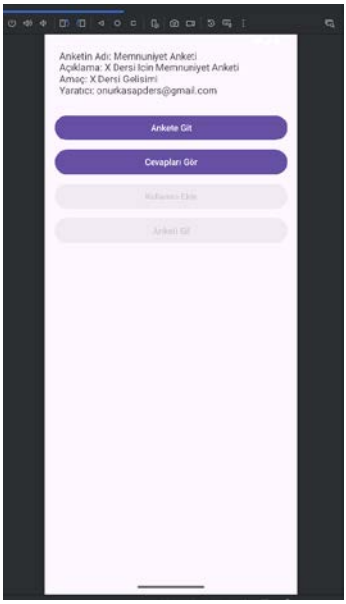
```
class CreateSurveyActivity : AppCompatActivity() {
    private fun saveSurveyToFirestore() {
            .add(survey)
            .addOnSuccessListener {
                Toast.makeText(this, "Anket başarıyla oluşturuldu!", Toast.LENGTH_SHORT).show()
                finish()
            }
            .addOnFailureListener { e ->
                Toast.makeText(this, "Hata: ${e.message}", Toast.LENGTH_LONG).show()
            }
    }

    private fun loadUsersFromFirestore() {
        FirebaseFirestore.getInstance().collection("users").get()
            .addOnSuccessListener { result ->
                for (document in result) {
                    val userEmail = document.getString("email") ?: "Bilinmeyen Kullanıcı"

                    val checkBox = CheckBox(this).apply {
                        text = userEmail
                        setOnCheckedChangeListener { _, isChecked ->
                            if (isChecked) {
                                if (isChecked) {
                                    selectedUserEmails.add(userEmail)
                                } else {
                                    selectedUserEmails.remove(userEmail)
                                }
                            }
                        }
                    }
                    layoutUserCheckboxes.addView(checkBox)
                }
            }
            .addOnFailureListener { e ->
                Toast.makeText(this, "Kullanıcılar yüklenemedi: ${e.message}", Toast.LENGTH_LONG).show
            }
    }
}
```

Şekil 4. Geliştirilen Uygulamanın Anket Hazırlama Ortamı Kodları



Şekil 5. Geliştirilen Uygulamanın Anket Hazırlama Ortamı Ekran Görüntüsü

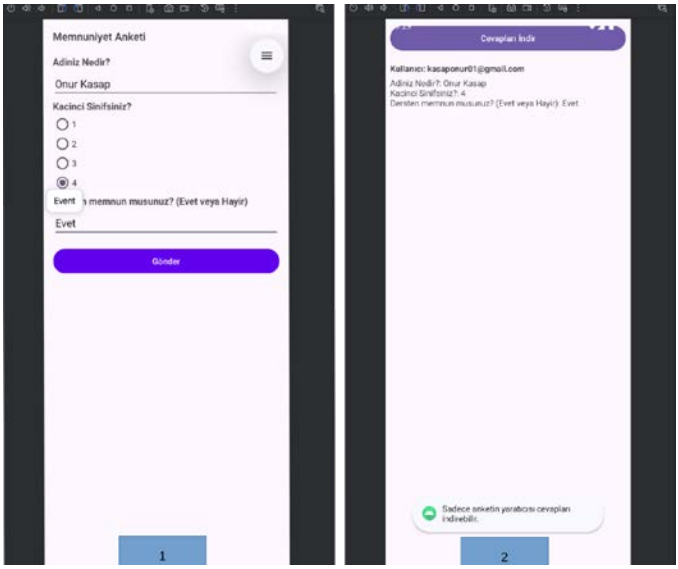
Anket oluşturma işlemi tamamlandıktan sonra, kullanıcı "Anketi Kaydet" butonuna basmalıdır (Şekil 5(3)). Bu adımın ardından, sistemde bir anket oluşturulmaktadır ve kullanıcı tarafından tanımlanan e-posta adreslerine bağlı kullanıcılar, giriş yaptıktan sonra ilgili anketi ana sayfalarında görüntüleyebilmektedir. Kullanıcıya tanımlanmış olan anket uygulaması, Şekil 3'de verilmiş olan ekranda yer almakta olup, kullanıcı bu ekrandaki ilgili buton aracılığıyla anket formunu görüntüleyebilmekte ve anketi doldurmaya başlayabilmektedir.



Şekil 6. Geliştirilen Anket Hazırlama Uygulaması Anket Ekranı-1

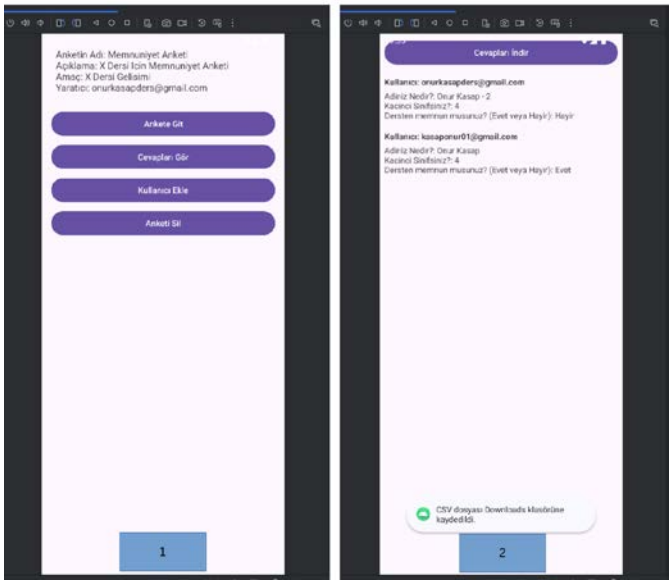
Geliştirilen uygulamanın Şekil 6'da görülen ekranında ise, yalnızca kendilerine anket yetkisi tanımlanmış olan kullanıcıların ilgili anketi tamamlayabilmeleri ve anket maddelerine verdikleri yanıtları görüntüleyebilmeleri mümkündür. "Kullanıcı Ekle" ve "Anketi Sil" gibi butonlar ise yalnızca anketi oluşturan kullanıcıya ait e-posta hesabı ile giriş yapıldığında görüntülenmektedir. Geliştirilen uygulamanın bu

özelliği anketin özgünlüğünü korumayı, katılımcıların verilerinin gizlilik ve güvenliğini sağlamayı amaçlamaktadır.



Şekil 7. Geliştirilen Anket Hazırlama Uygulaması Anket Ekranı-1

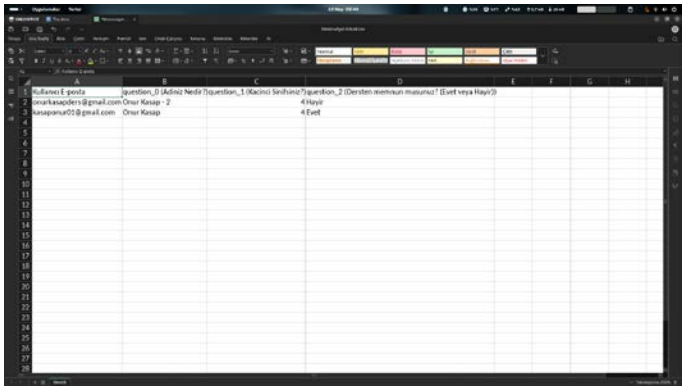
Şekil 7 (1)' de geliştirilen uygulama için bir anket giriş ekranı görünmektedir. Bu ekranda katılımcılar anket madde/sorularını yanıtlayabilmekte ve "Gönder" butonu ile anketi tamamlayabilmektedir. İşlem bitiminde Şekil 7 (2) 'de bulunan ekran görünmekte ve bu ekranda "Cevapları Gör" butonuna basan katılımcı, kendi yanıtladığı anket madde/ sorularının şıklarını görüntüleyebilmektedir. Kullanıcının Şekil 7 (2)' deki ekranda "Cevapları İndir" butonuna basması ile bir popup bildirimi ekrana gelmektedir. Bu da cevapları sadece anketi oluşturan kullanıcının indirebileceği anlamını taşımaktadır.



Şekil 8. Geliştirilen Anket Hazırlama Uygulaması Oluşturucu Ekranı

Şekil 8' de geliştirilen uygulamanın anket oluşturucu ekranının görüntüsü verilmektedir. Anketi oluşturan kullanıcı ekranında Şekil 8 (1)'de "Kullanıcı Ekle" ve "Anketi Sil" butonuna basılabilmektedir. Ayrıca Şekil 8 (2)'de görüldüğü üzere anketi oluşturan kullanıcı "Cevapları İndir" butonu ile ekranın altında popup bildirimi ile "CSV dosyası Downloads klasörüne kaydedildi" uyarısını almaktadır. Geliştirilen uygulamada bu işlemle anket yanıtları cihazın indirilenler klasörüne ".csv" formatında kaydedilebilmektedir.

Şekil 9' da ".csv" formatında indirilen anket yanıtları örnek olarak görülmektedir.



Şekil 9. Geliştirilen Anket Hazırlama Uygulaması .csv Formatı Anket Yanıtları Ekranı

4. SONUÇ

Bu çalışmada, Android işletim sistemi kullanılarak bir Anket Hazırlama Uygulaması geliştirilmiş, mobil uygulama geliştirme süreci teorik ve pratik yönleriyle değerlendirilerek, uygulamanın aşamaları hakkında bilgi verilmiştir. Geliştirilen Anket Hazırlama Uygulamasının kullanıcı arayüzü, kodlama süreci ve platform seçimi detaylı biçimde sunulmuştur.

Çalışma uygulaması, Android' in açık kaynak yapısı, düşük yayın maliyetleri ve esnek geliştirme araçları sayesinde bireysel geliştiriciler için uygun bir platform olduğunu destekler niteliktedir. Geliştirilen Anket Hazırlama Uygulaması ile ayrıca, ".csv" formatında anket sonuçlarını indirilerek kolaylıkla Python, R, SPSS gibi programlama ve hazır yazılımlarla entegre veri analizi imkânı da sunabilmektedir. Böylece geliştirilen uygulama sadece kullanıcı deneyimi sağlamakla kalmayıp bilimsel anlamda veri toplama ve istatistiksel analizlerin de etkili biçimde kullanılabileceğini göstermektedir.

Mobil uygulamaların çeşitli güvenlik önlemleri sayesinde, herhangi bir dış yazılım şirketine ihtiyaç duyulmadan kurumlar içerisinde de güvenli ve esnek bir kullanım imkanı sunabilen bu tür anket uygulamalarının geliştirilmesi mümkündür. Günümüzde mobil cihazların yaygın olarak kullanılması, anketlere erişimi kolay ve hızlı hale getirmiştir. Geliştirilen Anket Hazırlama Uygulaması' nın bu anlamda bir örnek olduğunu ve Android tabanlı uygulamaların hem teknik beceri geliştirme açısından hem de veri bilimi süreçlerine katkı sağladığını söylemek mümkündür.

KAYNAKÇA

AppMaster. (2023). *Eksiksiz Bir Mobil Geliştirme Kılavuzu.* [Erişim: 18 Nisan 2023, https://appmaster.io/tr/blog/mobil-uygulama-gelistirme-rehberi]

Arista F. S. & Kuswanto H. (2018). Virtual Physics Laboratory Application Based on the Android Smartphone to Improve Learning Independence and Conceptual Understanding. *International Journal of Instruction,* 11(1), 1-16.

Büyükgöze, S. (2019). Mobil Uygulama Marketlerinin Güvenlik Modeli İncelemeleri, *Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi*, 12(1), 9-18.

Çınar, S. M. & Bilici, H. (2020). Mobil Cihazlarda Uygulama Geliştirmekte Kullanılan Platformların ve Dillerin Karşılaştırılması, *JournalMM,* 1(1), 42-54.

Çiloğlu, T., Özeren, E., & Ustun, A. B. (2021). Mobil Uygulama Geliştirme, Yayımlama ve Ekonomik Gelir Etme Aşamalarının İncelenmesi: İOS ve Android Sistemlerinin Karşılaştırması. Yeni Medya Elektronik Dergisi, 5(1), 60-77.

Ensonhaber. (2025). *Dünyada en çok kullanılan mobil işletim sistemleri belli oldu.* [Erişim: 16 Mart 2025, https://www.ensonhaber.com/teknoloji/dunyada-en-cok-kullanilan-mobil-isletim-sistemleri-belli-oldu]

Kopmaz B. & Arslanoğlu A. (2018). Mobil sağlık ve akıllı sağlık uygulamaları. *Sağlık Akademisyenleri Dergisi,* 5(4), 251-255.

MOBİSAD (Mobil İşletim Sektörü Raporu) 23. (2023). Türkiye'nin ve Dünyanın Dijital Karnesi. *Mobil İletişim Araçları ve Bilgi Teknolojileri İş İnsanları Derneği.*

Özdamar Keskin N. & Kılınç H. (2015). Mobil Öğrenme Uygulamalarına Yönelik Geliştirme Platformlarının Karşılaştırılması ve Örnek Uygulamalar. *Açıköğretim Uygulamaları ve Araştırmaları Dergisi*, 1(3), 68-90.

Regmi P. R. vd. (2016). Guide to The Design and Application of Online Questionnaire Surveys. *Nepal Journal of Epidemiology*, 6(4), 640-644.

Şencan Karakuş, B. (2021). Psikoterapide Mobil Uygulama Kullanımının Etik Kurallar Çerçevesinde Ele Alınması. *Yaşam Becerileri Psikoloji Dergisi*, 5(10), 133-140.

Türk, E. vd. (2014). Üniversite için Android Tabanlı Mobil Uygulaması ve Geliştirme Sürecinde Öğrenilenler. *Akademik Bilişim 2014.*

Uslu, B., Gür, Ş., Eren, T., Özcan, E. (2020). Mobil Uygulama Seçiminde Etkili Olan Kriterlerin Belirlenmesi ve Örnek Uygulama. İstanbul İktisat Dergisi, 70(1), 113-139. https://doi.org/10.26650/ISTJECON2019-0022

EKLER

EK-1: Veritabanına Kullanıcı Kaydetme

```
private fun saveUserToFirestore() {
    val auth = FirebaseAuth.getInstance()
    val db = FirebaseFirestore.getInstance()
    val currentUser = auth.currentUser
    if (currentUser != null) {
        val user = hashMapOf(
            "uid" to currentUser.uid,
            "email" to currentUser.email,
            "name" to (currentUser.displayName ?: "Anonim Kullanıcı")
        )
        db.collection("users").document(currentUser.uid)
            .set(user)
            .addOnSuccessListener {
                android.util.Log.d("FirestoreUser", "Kullanıcı Firestore'a kaydedildi.")
                Toast.makeText(this, "Kullanıcı kaydedildi.", Toast.LENGTH_SHORT).show()
            }
            .addOnFailureListener { e ->
                android.util.Log.e("FirestoreError", "Hata: ${e.message}")
                Toast.makeText(this, "Kullanıcı kaydedilemedi: ${e.message}", Toast.LENGTH_SHORT).show()
```

```
            }
        }
    }
```

EK-2: Veritabanına Okuma-Yazma Modeli

```
data class Survey(
    var id: String = "",
    val name: String = "",
    val description: String = "",
    val goal: String = "",
    val creator: String = "",
    val creatorEmail: String = "",
    val allowedEmails: List<String> = listOf(),
    val questions: List<Question> = emptyList(),
    val assignedUsers: List<String> = emptyList()
)

data class Question(
    val text: String = "",
    val options: List<String>? = null
)
```

EK-3: CSV Formatında Cihaza Kaydetme

```
try {
```

```
    val resolver = contentResolver
    val contentValues = android.content.ContentValues().apply {
        put(android.provider.MediaStore.MediaColumns.DISP-
LAY_NAME, fileName)
        put(android.provider.MediaStore.MediaCo-
lumns.MIME_TYPE, "text/csv")
        put(android.provider.MediaStore.MediaColumns.RELA-
TIVE_PATH, android.os.Environment.DIRECTORY_DOWN-
LOADS)
    }
    val uri = resolver.insert(android.provider.MediaStore.Down-
loads.EXTERNAL_CONTENT_URI, contentValues)
}
```

EK-4: Kullanıcı Cevapları Kaydetme

```
private fun loadQuestions() {

                layoutContainer.addView(textView)
                if (question.options != null && question.opti-
ons.isNotEmpty()) {
                    val radioGroup = RadioGroup(this)
                    question.options.forEach { option ->
                        val radioButton = RadioButton(this).apply {
                            text = option
                        }
```

```
                radioGroup.addView(radioButton)
            }
            radioGroup.setOnCheckedChangeListener {
group, checkedId ->
                val selected = group.findViewById<RadioBut-
ton>(checkedId)?.text.toString()
                userAnswers[question.text] = selected
            }
            layoutContainer.addView(radioGroup)
        }
            editText.setOnFocusChangeListener { _, hasFo-
cus ->
                if (!hasFocus) {
                    userAnswers[question.text] = edit-
Text.text.toString()
                }
            }
            layoutContainer.addView(editText)
        }
    }
    val submitButton = Button(this).apply {
        text = "Gönder"
        setOnClickListener { submitAnswers() }
    }
    layoutContainer.addView(submitButton)
}
```

EK-5: Sorulara Şık Ekleme

```
private fun createOptionField(parentLayout: LinearLayout): LinearLayout {
    val optionLayout = LinearLayout(this).apply {
        orientation = LinearLayout.HORIZONTAL
    }
    val editText = EditText(this).apply {
        hint = "Şık"
        layoutParams = LinearLayout.LayoutParams(0, LinearLayout.LayoutParams.WRAP_CONTENT, 1f)
    }
    val removeButton = Button(this).apply {
        text = "Sil"
        setOnClickListener {
            parentLayout.removeView(optionLayout)
        }
    }
    optionLayout.addView(editText)
    optionLayout.addView(removeButton)
    return optionLayout
}
```

EK-6: Cevapları Veritabanından Çekme

```
private fun fetchAnswersForSurvey(surveyId: String, onSuccess: (List<Map<String, Any>>) -> Unit) {
    db.collection("answers").whereEqualTo("surveyId", surveyId).get()
        .addOnSuccessListener { result ->
            val answers = result.map { it.data }
            onSuccess(answers)
        }
        .addOnFailureListener { e ->
            Toast.makeText(this, "Cevaplar alınamadı: ${e.message}", Toast.LENGTH_SHORT).show()
        }
}
```

EK-7: Kullanıcıya Atanmış Anketleri Veritabanından Çekme

```
private fun loadSurveys(userEmail: String) {
    Log.d("SurveyDebug", "Kullanıcının atanmış anketleri yükleniyor: $userEmail")
    db.collection("surveys")
        .whereArrayContains("allowedEmails", userEmail)
        .get()
        .addOnSuccessListener { result ->
            surveyList.clear()
            for (doc in result) {
                try {
```

```
                val survey = doc.toObject(Survey::class.java).apply
{
                    id = doc.id
                }
                surveyList.add(survey)
                Log.d("SurveyDebug", "Anket yüklendi: ${sur-
vey.name}")
            } catch (e: Exception) {
                Log.e("SurveyError", "Anket yüklenirken hata:
${e.message}")
            }
        }
        if (surveyList.isEmpty()) {
            Toast.makeText(this, "Size atanmış anket buluna-
madı.", Toast.LENGTH_SHORT).show()
        }
        adapter.notifyDataSetChanged()
    }
}
```

DATA AUGMENTATION IN IMAGE CLASSIFICATION USING DEEP LEARNING

Fatma ÇAKIROĞLU[1]

Rifat KURBAN[2]

Ali DURMUŞ[3]

Ercan KARAKÖSE[4]

1. INTRODUCTION

Analyzing data sets containing large amounts of data using traditional methods and algorithms is quite difficult [1]. Recent years have witnessed a sharp rise in data-science research, spurred by the need to extract insight from ever-expanding datasets. This growth has drawn renewed attention to allied notions data mining, machine learning (ML), artificial intelligence (AI), and deep learning (DL). At its broadest level, AI concerns the design of computational systems capable of mimicking human cognitive abilities. ML occupies the next tier, enabling those systems to infer patterns autonomously by harnessing computational power and algorithms. DL sits one

[1] Lecturer Fatma Çakıroğlu, Kayseri University, Collage of Information Technologies, Kayseri, Türkiye, fatmacakiroglu@kayseri.edu.tr, ORCID:0000-0001-9794-4996.

[2] Assoc. Prof. Dr. Rifat Kurban, Abdullah Gül University, School of Engineering, Department of Computer Engineering, Kayseri, Türkiye, rifat.kurban@agu.edu.tr, ORCID: 0000-0002-0277-2210.

[3] Assoc. Prof. Dr. Ali Durmuş, Kayseri University, Faculty of Engineering, Architecture and Design, Dept. of Electrical & Electronics Engineering, Kayseri, Türkiye, alidurmus@kayseri.edu.tr, ORCID: 0000-0001-8283-8496.

[4] Prof. Dr. Ercan Karaköse, Kayseri University, Faculty of Engineering, Architecture and Design, Dept. of Basic Sciences, Kayseri, Türkiye, ekarakose@kayseri.edu.tr, ORCID: 0000-0001-5586-3258.

level deeper as a specialised branch of ML, employing layered neural networks to learn hierarchical data representations. Thus, ML functions as both a subset of AI and a superset that subsumes DL. Figure 1 illustrates this nested hierarchy, positioning DL within ML and ML within the overarching domain of AI [2].

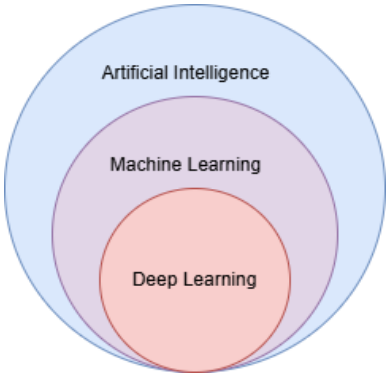


Figure 1. showing the connection between artificial intelligence, machine learning, and deep learning

Deep learning is a form of artificial intelligence built upon artificial neural networks that emulate the structure and function of the human brain. While it is technically a subset of machine learning, deep learning distinguishes itself through its use of multi-layered architectures that enable the automatic extraction of complex features from data. These deep neural networks can learn hierarchical representations without the need for manual feature engineering, making them particularly effective for tasks involving large-scale and high-dimensional data such as image recognition, natural language processing, and speech analysis. Deep learning eliminates some of the data preprocessing associated with typical machine learning and can take and process data such as text and images. As a result, it automates feature extraction, eliminating some of the dependence on experts. While deep learning is scalable with larger volumes of data, machine learning is limited to shallow learning after reaching a certain level, and adding more new data makes no difference. In recent

years, computer vision has achieved significant advancements in various fields such as image processing, face recognition, object recognition[3], autonomous vehicles; computer vision in unmanned vehicles, healthcare; disease diagnosis and feature extraction [4], robotics [5], agriculture, manufacturing, finance [6], speech recognition[7] and bioinformatics[8, 9]. Deep learning models have the ability to extract meaningful features from large data sets. [10]. In deep learning, algorithms can be supervised or unsupervised. [11, 12]. Deep learning algorithms are primarily based on a convolutional neural network model. Convolutional neural networks, which form the basis of deep learning architecture, combine in different ways to form the basis of modern deep learning architectures. The first deep learning architecture in the literature, "Neokognitron," was proposed by Fukushima in 1979. With the artificial neural network he named Neokognitron, the foundation was laid for convolutional artificial neural networks, which are now widely used in image analysis. [13]. Jürgen Schmidhuber also first proposed the long short-term memory (LSTM) model in 1997. The basic concept of LSTM is Cell State and the various gates it uses. It is one of the most important studies that contributed to the popularity of the field of "Deep Learning."[14]. Yann LeCun used the "LeNet" network to classify handwritten digits (MNIST) by applying convolutional networks together with backpropagation. Yann LeCun developed a gradient-based learning algorithm and combined it with the backpropagation algorithm. [15, 16]. Fei-Fei Li created the ImageNet database in 2009 for researchers, educators, and students. This system contains approximately 15 million data points, which users can utilize to design learning models. [17]. In the context of artificial neural networks, the term "deep learning" was first introduced in 2000 by Igor Aizenberg and his colleagues. [18]. In a seminal 2006 publication, Geoffrey Hinton introduced a groundbreaking approach for training deep feedforward neural networks. He demonstrated that each layer of

the network could be effectively pre-trained in an unsupervised manner using a restricted Boltzmann machine (RBM), allowing the network to learn meaningful feature representations layer by layer. Following this unsupervised pre-training phase, the entire network could then be fine-tuned using a supervised backpropagation algorithm. This method addressed several challenges associated with training deep architectures, particularly issues related to vanishing gradients and poor convergence, and marked a pivotal moment in the resurgence of deep learning research. [19]. With advancements in GPU (Graphics Processing Unit) technology, it has become feasible to train deep neural networks from scratch without the need for pre-training. GPUs are specifically designed to handle parallel processing tasks efficiently, making them well-suited for rapid computation of high-resolution images and videos. As specialized hardware for graphical computations, GPUs significantly accelerate the training process of deep learning models. Leveraging this capability, Ciresan and his collaborators applied deep learning architectures directly—without pre-training—to domains such as traffic sign recognition, medical image analysis, and handwritten character classification, achieving notable performance improvements across these tasks. [20, 21]. Deep learning is primarily based on training classified attributes related to data. Therefore, low-level attributes are combined to form an attribute hierarchy containing higher-level attributes. To summarize how deep learning works: [22];

1. The problem is defined
2. The daha set to be used is determined.
3. The Deep Learning algorithm is determined.
4. The selected algorithm is trained with the specified data.

5. The trained algorithm is tested with the data set aside for testing.

Effective deep learning modeling relies on both a substantial dataset and a capable algorithm to process it. In such applications, the size and variability of the dataset are key determinants of model performance. Generally, as the dataset grows, so does the model's ability to learn meaningful patterns. However, this growth also leads to increased training time and larger model sizes. Importantly, quantity alone is not enough—diversity within the data is equally critical. A more varied dataset enables the model to generalize better across different scenarios. Still, performance gains from additional data tend to plateau; after a certain threshold, improvements become marginal. In image-based tasks, a few dozen or even hundreds of samples per class are rarely sufficient. For reliable learning, at least several thousand samples per class are typically required. When working with a limited dataset, deep learning may not be the most appropriate solution unless measures are taken to enhance the data. In such cases, synthetic data generation through data augmentation becomes essential to increase both the size and diversity of the training set.

2. DATA AUGMENTATION IN DEEP LEARNING

In deep learning workflows, effective model training necessitates dividing the dataset into distinct subsets: a training set and a test set. The training set is used to iteratively pass data through the network, during which the model parameters are continuously adjusted to optimize performance toward the desired output. Upon completion of the training phase, the test set is employed to objectively assess the generalization capability of the trained model. Deep learning algorithms typically require large volumes of data to mitigate overfitting—a condition in

which the model becomes overly tailored to the training data and fails to perform adequately on unseen inputs. This issue is particularly pronounced in image analysis tasks, where acquiring sufficiently large and diverse datasets can be difficult. To address these limitations, data augmentation techniques are widely utilized to synthetically expand the training set, thereby enhancing its variability and enabling more robust and generalizable model learning [23]. Data augmentation refers to a suite of synthetic data–generation strategies that enlarge and enrich image datasets via intentional transformations. By broadening the variety of training samples, these techniques boost model robustness and generalisation, curbing overfitting—an especially critical benefit when data are scarce. Early deep-learning work such as LeNet-5 already employed image-distortion tricks to recognise handwritten digits. Since then, the field has grown: Mikołajczyk et al. tackled limited-data challenges by augmenting with traditional operations (zoom, crop, rotate, histogram tweaks) and by comparing them with a Style-Transfer-based approach. Their contribution introduced style-transfer augmentation to the literature, merging the source image's content with alternate visual styles to yield perceptually rich, synthetic examples. These high-quality composites can be used for network pre-training, ultimately improving learning efficiency and final performance. [24]. Jakub Nalepa has conducted a research study to synthesize high-quality synthetic brain tumor data that could improve the generalization capabilities of deep learning models [25]. Shorten and Khoshgoftaar present an extensive survey of image-based data-augmentation strategies for deep learning, encompassing geometric manipulations, image blending, colour-space adjustments, random erasure, feature-space perturbations, generative adversarial networks (GANs), kernel filters, neural style transfer, adversarial training schemes and meta-learning frameworks. Their analysis devotes particular attention to GAN-

driven augmentation, elucidating implementation details and benchmarking its impact across multiple computer-vision tasks. [26]. A different study on GAN applications in radiology is discussed by V Sorin [27]. Another study focusing on different data augmentation techniques to reduce the overfitting problem was done by Cherry Khosla [28]. In their study, Khosla et al. categorized data augmentation techniques into two primary groups: synthetic oversampling methods and data warping strategies. They evaluated the impact of various augmentation approaches—including GAN/WGAN-based generation, flipping, rotation, cropping, shifting, and noise injection—on image classification performance within a deep convolutional neural network framework. The experiments were conducted using AlexNet as a pre-trained backbone model, and assessments were carried out on subsets of the CIFAR-10 and ImageNet datasets, following the dataset configuration outlined by Jia Shijie et al. This investigation provided valuable insights into how different augmentation techniques influence model accuracy and robustness in image classification tasks [29]. The "Smart Augmentation" method, introduced to the literature by oseph Lemley and his colleagues, is an innovative data augmentation strategy proposed to reduce the overfitting problem and increase the accuracy of the target model. This approach is based on an auxiliary network architecture that works integrated with the training process of the target network and learns to produce augmented data samples in a way that minimizes the loss of the network directly. In other words, the augmentation process is transformed into a learnable process instead of being predefined with fixed transformations. Thus, the generated data samples are dynamically adapted to increase the generalization capacity of the target model. As a result of the experimental studies, the Smart Augmentation method has provided statistically significant accuracy increases on various data sets and has demonstrated the potential to improve the generalization performance of the model.

In this respect, the method, unlike classical data augmentation techniques, offers a framework that optimizes the augmentation process specific to the learning task [30]. In studies on data augmentation techniques, various innovative approaches have been proposed that aim to overcome the limitations of classical transformation-based methods. In this context, the Random Local Rotation method developed by Alomar et al. stands out as a remarkable strategy that contributes to the image-based data augmentation literature. The method in question offers an augmentation mechanism based on randomly selecting the positions and sizes of circular regions on the image and rotating these regions with random angles. Its parameter-free structure and high applicability allow the method to be evaluated as both a simple and effective augmentation strategy. Alomar and his team developed this approach specifically to prevent distortions and irregularities that traditional rotation methods may cause at image boundaries. Since the random local rotation strategy affects only local regions instead of rotating the entire image, it largely preserves the structural integrity of the original image while also making the model more robust against local transformations. In this respect, the method offers an alternative and effective data augmentation approach that can be used to increase the generalization ability of the model, especially under limited data conditions [31]. Two new solutions to the overfitting problem encountered in small-scale datasets have been proposed by Nanni et al. The first of these solution methods is based on the Discrete Wavelet Transform (DWT) that enables multi-resolution analysis of image features, while the other one uses the Constant-Q Gabor Transform (CQT) that provides finer control over frequency resolution. Both transformation methods aim to reduce the risk of overfitting by supporting CNN-based models to learn more robust and generalizable representations on small datasets [32]. Another recent study on data augmentation involves the use of different imaging methods (mammography, fundoscopy, MRI, and CT) in

different organs (breast, eye, brain, and lung) to investigate augmentation techniques aimed at improving the performance of deep learning-based disease diagnosis. [33]. Conventional data-augmentation still leans on hand-crafted transformations. Although they can markedly boost a model's predictive power, choosing the right mix of augmentations is a painstaking, trial-and-error chore: the space of possible parameter settings is effectively endless. Even when no hard limits are imposed on how far an image can be altered, achieving strong results without exhaustive experimentation is unlikely. To relieve this burden, several algorithmic, fully automated solutions have emerged in recent years [34]. Because the "best" augmentation often depends on the dataset's content, experts traditionally tailor the pipeline by eye—a subjective process prone to missteps. Automatic Data Augmentation (AutoDA) techniques [35, 36] sidestep that subjectivity by learning augmentation policies directly from the data. Their objective is simple: discover the transformation strategies that push model performance to its peak. The augmentation methods reviewed here are organised in Section 3.

3. CLASSIFICATION OF DATA AUGMENTATION METHODS

Image data augmentation methods aim to improve both the quality and quantity of data sets so that models can be trained more effectively [26]. Data augmentation methods can be broadly categorized into two main types: traditional data augmentation methods and automatic data augmentation methods. Figure 2 shows the classification of data augmentation methods.

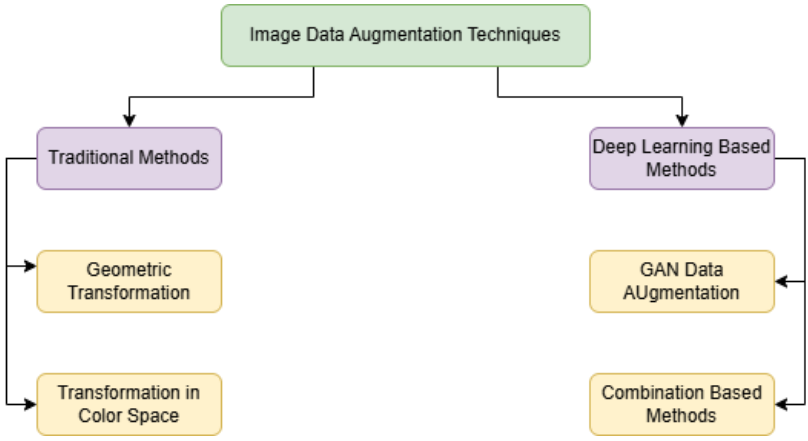


Figure 2. Classification of image data augmentation techniques

3.1. Traditional Data Augmentation Methods

Traditional data augmentation typically converts existing images into a new form while preserving the original label of the image. [26]. These techniques can be implemented through a variety of image processing strategies, such as geometric manipulations, color space alterations, or a blend of both. Figure 3 illustrates examples of classical data augmentation methods applied to a sample image from the Lytro dataset, showcasing how these transformations modify the original input while preserving its semantic content [37].

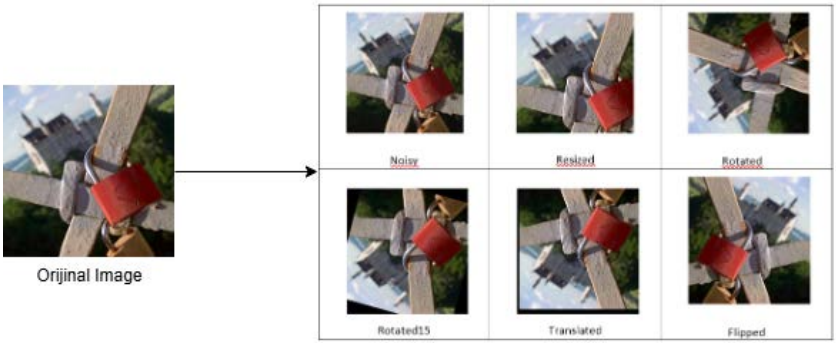


Figure 3. Some Traditional Data Augmentation Methods

3.1.1. Geometric Transformation Methods

The simplest way to start increasing image data is to use geometric transformation functions such as image rotation or scaling [35]. When starting to enhance image data, applying basic geometric transformations such as rotation, rescaling, and reflection are the most common and simplest methods [38]. Transformations that carry the risk of changing label information are not considered safe data augmentation methods. In this context, geometric transformations that preserve basic structural information are considered more favorable in terms of label consistency. However, applying the selected operation may not always be safe. Geometric data augmentation methods are described below.

3.1.1.1. Cropping

In particular, the cropping method applied to make images of different resolutions suitable for training is both an effective preprocessing step that normalizes spatial dimensions and a fundamental data augmentation strategy. Cropping reduces spatial information by removing a specific region from the image, unlike other augmentation methods such as geometric translation, which reduce image dimensions. This situation carries the risk of not preserving the class label after the transformation. Therefore, cropping does not always have the property of label-preservation. In the training process, fixing the dimensions of the input data is necessary for subsequent convolution and matrix operations. In this context, one of the important methods in the literature is RICAP (Random Image Cropping and Patching). The RICAP method, developed by Takahashi and colleagues, creates a new training example by combining patches randomly cropped from four different images. This technique not only increases example diversity but also integrates a soft labeling approach by proportionally blending the class labels of each patch. This

enhances the model's overall performance while reducing the potential for overfitting. RICAP has improved the model's generalization ability by creating a geometric transformation effect and distributing class representations within data augmentation strategies [39].

3.1.1.2. Rotation

It is a commonly used data augmentation technique that involves rotating the image to the right or left at a specific angle between 1° and 359° on an axis. The security of data augmentation methods based on transformation depends largely on the parameters that determine the degree of transformation applied. Operations with large rotation angles are generally unsafe, while a slight rotation between 1° and 20° can be useful for most image classification tasks. Low-degree rotations may be safe in digit recognition tasks such as MNIST and SVHN. However, as the rotation degree increases, it becomes unsafe. For example, in the MNIST and SVHN digit recognition tasks, if the digit 6 is rotated by 180 degrees, there is a possibility that it may be confused with the digit 9 [40].

3.1.1.3. Flipping

It is one of the easiest data augmentation techniques to implement. Horizontal and vertical flipping can be performed. Horizontal flipping is more commonly used. It is a different technique from rotation data augmentation because it produces mirror images. This method is an effective augmentation technique for CIFAR10-CIFAR100 and ImageNet datasets. However, it is not a label-preserving method for datasets containing text recognition, such as MNIST or SVHN [41].

3.1.1.4. Translation / Shifting

It is the displacement of all pixels in the image in a specific direction (right, left, up, or down). This transformation increases

the model's ability to recognize objects in different positions by changing the position of the object in the image [42].

3.1.1.5. Cutout

Cutting is a data augmentation strategy that involves masking one or more randomly selected regions within an image, typically by filling them with zeros or a constant color value. This approach reduces the model’s reliance on specific visual features, thereby enhancing its ability to generalize across varying conditions. It is particularly effective in scenarios involving object occlusion, where portions of the target object are obscured, partially blocked by other elements, or only partially visible. By simulating such occlusion during training, the model becomes more robust to real-world variations in visibility. Furthermore, this technique can be combined with other augmentation methods—such as geometric distortions or color-space modifications—to further enrich the training dataset and improve the model’s resilience to diverse visual contexts [43].

3.1.1.6. Scaling / Zooming

Scaling is the process of enlarging or reducing the dimensions of an image. This process creates an effect as if the distance between the object and the camera is changing. When used for data augmentation, it enables the model to learn to recognize objects at different scales [24].

3.1.2. Transformations in Color Space

Within the scope of data augmentation strategies, transformations performed in color space involve various changes made to the color components of images. These transformations are applied through the manipulation of visual qualities such as brightness, contrast, saturation, and tone. Since digital images are typically represented as three-dimensional tensors (height × width × color channels), enhancement operations performed on color

channels stand out as computationally efficient and highly applicable methods. Simple color enhancement techniques are typically performed by isolating a single color channel (e.g., red, green, or blue). This process can be easily implemented by replacing the other channels with a zero matrix. Additionally, the brightness of images can be adjusted through basic matrix operations on RGB values. More advanced approaches involve analyzing the color histograms of the image, thereby enabling a richer color variation. However, since color transformations can alter the structural content of the image, they cannot always be considered label-preserving transformations. Therefore, such enhancement methods must be applied with caution during the model training process [28].

3.1.2.1. Brightness

Brightness change is a color space-based data augmentation method that increases or decreases the brightness level of all pixels in an image. It ensures that the model performs well under different lighting conditions. The goal is to simulate different lighting conditions and allow the model to adapt to such variations. It affects all color channels equally [28].

3.1.2.2. Contrast

Contrast enhancement is a color space-based data augmentation method that increases or decreases the difference between colors and lights in an image. This process controls the difference between dark and light areas in the image and enables the model to perform better under different lighting conditions [26].

3.1.2.3. Saturation

Saturation adjustment is a color space-based data augmentation method that increases or decreases the vividness of colors in an image. Saturation indicates the intensity of a color;

colors with high saturation are very vivid and bright, while colors with low saturation are dull and grayish [26].

3.1.2.4. Grayscale

During data augmentation, it aims to increase the model's color-independent generalization ability by completely or partially removing the color information from the image. This is particularly useful in data sets with a lot of color variation [26].

3.1.2.5. RGB Channel Mixing

RGB Channel Shuffling is the process of mixing or randomly changing the color channels (red, green, blue) in an image during the data augmentation process. This technique reduces the model's dependence on the order of colors by breaking the dependency between color components in the image, thereby improving its generalization ability.

3.1.2.6. Kernel Filters

In image processing, instead of directly changing pixel values in the color space, these values are usually manipulated indirectly through kernel filters. Kernel filters are matrices consisting of specific numerical weights that are much smaller in size than the input image, and the values of these weights determine the function performed by the filter. These filters are applied to the image by performing a convolution operation on the entire image using a sliding window method. The new pixel values of the output image are calculated as a result of matrix multiplications performed on the obtained local regions. Among the most commonly used kernel filters are blurring and sharpening filters. For example, a Gaussian-based blurring filter reduces high-frequency details in the image, providing a lower-resolution and less noisy representation. This can enhance the model's ability to handle low-quality or low-resolution images. On the other hand, sharpening filters can support the model's

ability to learn more distinctive features by highlighting edges and details in the image. Such filter-based transformations play a significant role in increasing data diversity and making the model more resilient to different image conditions [44].

3.2. Deep Learning-Based Methods

Deep learning-based image data augmentation methods use more advanced and learning-based methods, in addition to traditional data augmentation techniques, to enable the model to learn in a more robust and generalizable manner. These methods generate more data by making feature-based changes to images and enable learning richer representations of the data. Deep learning-based methods are explained in the subheadings below.

3.2.1. GAN (Generative Adversarial Networks)

Generative Adversarial Networks (GANs) are deep generative architectures built around two mutually adversarial components: a generator, tasked with synthesising plausible samples that emulate the true data distribution, and a discriminator, charged with distinguishing these synthetic outputs from genuine observations. Through an iterative, game-theoretic training procedure, the discriminator's feedback progressively drives the generator toward producing increasingly realistic data. Owing to this capacity, GAN-based augmentation has become a prevalent strategy for expanding training sets and enhancing the generalisability of deep-learning models particularly when data are scarce or class distributions are imbalanced. By capturing intra-class variability, GANs can create high-fidelity synthetic images that mirror the statistical properties of the originals. Beyond data augmentation, GANs also excel in core probabilistic tasks such as approximate inference and maximum-likelihood estimation, underscoring their versatility as one of the foremost frameworks in deep generative modelling [45]. GANs offer certain advantages for efficiently generating desired samples

[27]. Among the notable advantages of Generative Adversarial Networks (GANs) is their capacity to generate more realistic outputs compared to autoregressive models. Furthermore, GAN architectures integrate seamlessly with deep neural networks and can be effectively combined with other deep learning frameworks. In this context, Hung has explored the application of data augmentation techniques—particularly the use of GANs—in generating high-quality and diverse synthetic imagery. His study offers a comprehensive analysis of advanced GAN architectures, with a focus on models that leverage one-to-one and many-to-many image mapping schemes. These innovations aim to enhance the flexibility and realism of synthetic data generation, further solidifying GANs as a powerful tool in data augmentation and representation learning [46]. The role of GANs in advancing artificial intelligence in the agriculture and food sectors has been researched by Yuzhen Lu [47]. Since acquiring large image datasets in medical imaging is costly, synthetic data generation is often preferred. Mantegna and colleagues used the GAN method to augment medical images. This method has been quite successful with small datasets [48]. Another study that created a synthetic data set by applying a GAN-based augmentation technique to a limited MRI data set was introduced to the literature by Chang Qi and his team

3.2.2. Data Augmentation with Self-Supervised Learning

Self-supervised learning is a training paradigm designed to enable models to learn informative representations by leveraging inherent structural patterns within the data itself, without relying on externally provided labels. This approach serves as a robust and adaptable alternative, particularly in contexts where labeled data are scarce. It plays a pivotal role in both data augmentation and representation learning, allowing models to extract semantically meaningful features through

pseudo-labels or pretext tasks derived directly from the raw input. Consequently, self-supervised methods significantly reduce dependency on manual annotation.

In contrast, semi-supervised learning seeks to enhance model performance by combining a small set of labeled data with a large corpus of unlabeled data. By exploiting the intrinsic structure present in the unlabeled portion, this approach facilitates the learning of more generalizable and robust feature representations. Semi-supervised learning is especially valuable in practical settings where labeling is costly or time-consuming, offering a pragmatic solution to bridge the gap between supervised and unsupervised learning [49]. In many real-world applications, obtaining large-scale, balanced labeled datasets is costly, time-consuming, and often impractical. In this context, while unlabeled data is a widely available resource, it typically requires expert knowledge and manual labeling to be used directly in supervised learning processes. Semi-supervised learning is a learning paradigm that aims to improve model performance by combining labeled and unlabeled data in situations where there are a limited number of labeled examples. This method is based on a combination of supervised and unsupervised learning, involving first training the model with labeled data, then having the trained model make predictions on unlabeled data to produce pseudo-labels, and finally incorporating this data back into the training process. Semi-supervised classification methods are particularly critical in scenarios where labeled data is insufficient. These approaches can enable the learning of more generalizable and discriminative representations by leveraging the structural information contained in unlabeled data. Additionally, large-scale unlabeled datasets can significantly improve classification performance when used as auxiliary representations. Semi-supervised learning techniques have become a fundamental component in many

modern machine learning systems aiming to achieve high accuracy rates despite low-labeled data costs [50]. A system that automatically learns the most appropriate data augmentation policies for self-supervised learning (SSL) methods working on unlabeled data has been introduced to the literature by CJ Reed and colleagues. In this study, transformation estimation is used as a self-supervised task. The image is rotated at certain angles. The model predicts the direction in which the image is rotated. The policies learned with SelfAugment have yielded better results in linear evaluation (linear classification) performance compared to classical augmentation methods [51].

3.2.3. Variational Autoencoders (VAE)

Variational Autoencoders (VAE) learn probabilistic representations of data and then generate new data samples from these representations. VAE generates different variations of images by using variations of the learned representations. Quentin Chadebec and colleagues proposed a geometry-based Variational Autoencoder (VAE) method for image data augmentation in high-dimensional but low-sample-size datasets (HDLSS – High Dimensional Low Sample Size), and this proposed method was tested on MRI images and proved to be quite successful in models working with limited data [52]. Yamini Madan and colleagues used MRI data to diagnose Parkinson's disease and created new synthetic MRI images using the Variational Autoencoder data augmentation method. With this method, they produced synthetic data and improved classifier performance by approximately 6% [53].

3.2.4. Style Transfer

Style Transfer combines the content of one image with the artistic style of another image. This method allows images to be diversified with different styles. Style Transfer typically works between two main components: Content Image: This is the

original data and the image to which style transfer will be applied. Style Image: This is the artistic style or visual style image that will be applied to the content image. Jackson and his colleague proposed a method that enhances robustness against domain shift in both classification and monocular depth estimation tasks by generating texture, color, and contrast variations through random style transfer. When combined with classical augmentations, the method yields better results [54]. A study aimed at performing image data augmentation using neural style transfer (NST) methods has been presented to the literature by Borijan Georgievski. This study has succeeded in improving classification performance by enriching the style variations of existing data sets. Training with augmented data has reliably improved CNN performance [55].

3.3. Combination-Based Methods

Combination-based image data augmentation techniques aim to create new and diversified data samples by combining different data manipulations. These methods can be more powerful than a single data augmentation technique because the combination of multiple transformations enables the model to learn more generally and robustly. Combination-based data augmentation offers significant advantages, especially when working with limited datasets. Chengtai Cao and colleagues conducted a comprehensive analysis of an important subset of data augmentation techniques called Mix-based Data Augmentation (MixDA), which generates new examples by combining multiple examples, and presented their findings in the literature [56]. Another study that comprehensively investigates combination-based data augmentation methods used in image processing and deep learning and their strategies was conducted by Lewy [57].

3.3.1. Mixup

Mixup creates new images by mixing images and labels in a linear fashion. This technique transforms both images and labels into a new image and label pair by mixing them. Mixup creates a mixture in both image data and labels. Two images and labels are selected. Both images are mixed at a random α (alpha) ratio. The resulting new image is paired with the mixed label [58].

3.3.2. CutMix

CutMix is similar to Mixup, but here part of the image is cut out and the corresponding part of another image is inserted. This provides a more unique approach to creating new images. A random area of an image is cut out. Another image of the same size is inserted into this area. Both images are matched with mixed labels.

3.3.3. Mixup and CutMix Combination

Mixup and CutMix can be jointly applied to develop more advanced data augmentation strategies. While Mixup generates synthetic samples by linearly interpolating two input images and their corresponding labels, CutMix enhances diversity by replacing a cropped region from one image with a patch from another. In a combined approach, an initial Mixup operation is followed by the application of CutMix to a specific region of the mixed image, resulting in richer and more varied training samples. Chanwoo Park and Sangdoo Yun conducted a unified theoretical analysis of such mixed-sample data augmentation techniques, including Mixup and CutMix. Their findings reveal that these methods act as forms of pixel-level regularization applied to the base training loss, as well as implicit regularization of the network's first-layer parameters. Notably, this regularization effect holds across different mixing strategies, highlighting the robustness and consistency of mixed-sample augmentation in improving generalization performance [59]. A

new method called MiAMix, which stands for Multi-Stage Augmented Mixing, has been introduced to the literature by W Liang [60]. MiAMix integrates image enhancement into the blending framework and uses multiple diversified blending methods simultaneously. This method improves the blending method by randomly selecting blending mask enhancement methods. MiAMix is also designed for computational efficiency.

4. IMAGE DATA SETS USED IN THE TRAINING OF DEEP LEARNING MODELS

4.1. CIFAR-10

CIFAR-10 is a dataset containing 10 classes of small color images. This dataset is widely used in the fields of computer vision and machine learning. CIFAR-10 contains 60,000 color images of 32x32 pixels in size. These images are divided into 10 different categories, with each category containing 6,000 images [61]. It includes ten different categories: car, bird, airplane, cat, deer, dog, frog, horse, ship, and truck. The CIFAR-10 dataset is a fundamental resource commonly used for testing and training deep learning algorithms. Each image consists of 32x32 pixels and three channels with RGB (red, green, blue) color channels [61, 62]. Figure 4 shows images from the CIFAR 10 dataset.

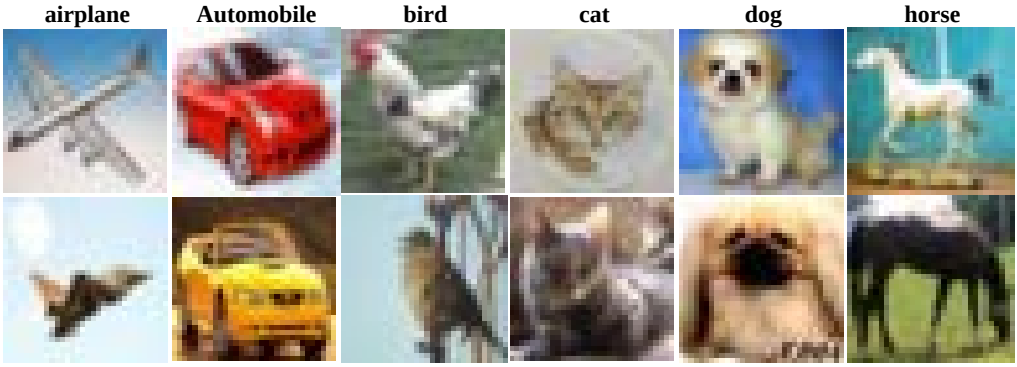


Figure 4. Some images from the CIFAR10 dataset

4.2. CIFAR-100

CIFAR-100 is an expanded version of the CIFAR-10 dataset and contains 100 different classes. The CIFAR-100 dataset, like CIFAR-10, contains 60,000 color images with a size of 32x32 pixels. These images are divided into 100 different categories, with each category containing 600 images. It includes 50,000 training images and 10,000 test images. The classes in CIFAR-100 exhibit greater diversity than those in CIFAR-10 and include more specific objects. These classes are divided into two main groups: “Main Classes” and “Subclasses.” When working with CIFAR-100, this dataset may require more careful and advanced models to ensure that the classification model is more precise and accurate due to the similarities between classes [61]. Figure 5 shows sample images from the CIFAR 100 dataset.

Figure 5. Sample images from the CIFAR 100 dataset

4.3. IMAGENET

ImageNet is a large and comprehensive visual dataset widely used in computer vision and deep learning research [63, 64]. ImageNet, which began in 2009 and was later expanded, contains millions of labeled images and is considered an important reference dataset for tasks such as object recognition and classification. Total Number of Images: ImageNet has over 14 million images. Number of Classes: It includes 1,000 different object categories. These categories cover living things, vehicles, food, clothing, locations, and many other items. Image Size: Each image is typically 256x256 pixels in size, but is often cropped to 224x224 pixels for use. ImageNet has a wide variety of categories, such as:

- **Animals:** cats, dogs, birds, horses, whales, etc.
- **Objects:** computers, tables, telephones, bags, chairs, etc.
- **Food:** apples, bananas, watermelons, pizza, etc.
- **Natural objects:** flowers, trees, stones, etc.

Figure 6. Some images from the ImageNet dataset

4.4. MNIST

The MNIST dataset (Modified National Institute of Standards and Technology) is a benchmark dataset consisting of images of handwritten digits (0-9) that is widely used in machine learning and especially in the field of image recognition [65]. Contains black-and-white images of handwritten digits from 0 to 9. Each image is 28x28 pixels (784-dimensional vector) in size. There are a total of 70,000 data points. The training set contains 60,000 data points, and the test set contains 10,000 data points. Each image is labeled with the digit it represents (0-9).

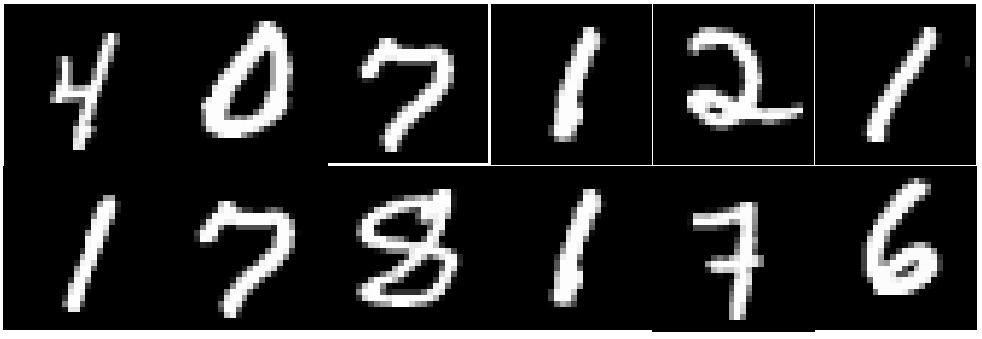


Figure 7. Images from the MNIST dataset

4.5. COCO

The COCO dataset (Common Objects in Context) is a comprehensive and richly labeled dataset developed for tasks such as object detection, image segmentation, image captioning, and keypoint detection in the field of computer vision [66]. It has 80 object categories. It contains more than 330,000 objects. It contains more than 1,500,000 tagged objects.

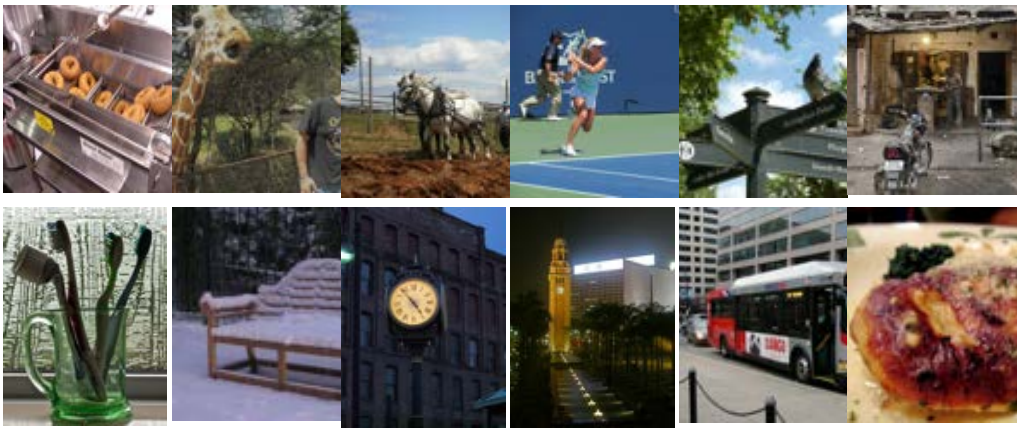


Figure 8. Images from the COCO dataset

4.6. PASCAL VOC (PASCAL Visual Object Classes)

The PASCAL VOC (Visual Object Classes) dataset is a classic and effective benchmark dataset widely used in computer vision and especially in tasks such as object detection, classification, and segmentation [67]. It has 20 subcategories. It contains approximately 11,000 images for the VOC 2012 version.

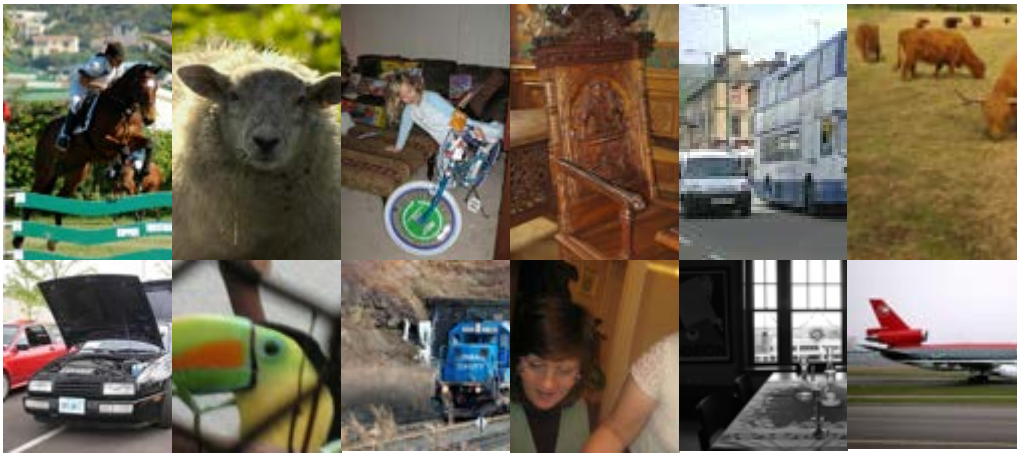


Figure 9. Sample images from the PASCAL VOC dataset

5. CONCLUSION

In this study, the effect of data augmentation methods on model performance in the field of image processing and data augmentation techniques have been comprehensively investigated. It is well known that deep learning-based algorithms require large and diverse data sets to achieve high success. However, collecting data in the real world can be costly, time-consuming, or ethically restrictive. In such cases, data augmentation methods offer an effective and practical solution. Research shows that while classical data augmentation methods enhance the model's generalization capacity, mix-based, GAN-based, and self-supervised methods, when used in conjunction with deeper artificial neural networks, yield notable improvements in accuracy. GAN-based VAE augmentation methods have achieved successful results in synthetic data augmentation on MRI datasets with limited data. Especially in cases of class imbalance, few-shot learning, and domain adaptation, data augmentation strategies significantly enhance the model's flexibility and robustness. In conclusion, data

augmentation is not merely a technique that enhances performance but also a fundamental approach that enables data-driven artificial intelligence systems to become more equitable, sustainable, and accessible. In future studies, adaptive combinations of different augmentation methods and learning-based augmentation strategies will contribute to making image processing systems more reliable.

REFERENCES

1. Najafabadi, M.M., et al., *Deep learning applications and challenges in big data analytics.* Journal of big data, 2015. 2: p. 1-21.
2. Metlek, S. and H. Çetiner, *Matlab Ortamında Derin Öğrenme Uygulamaları.* Ankara/Turkey, 2021.
3. Haris, M. and A. Glowacz, *RETRACTED: Road Object Detection: A Comparative Study of Deep Learning-Based Algorithms.* Electronics, 2021. 10(16): p. 1932.
4. Yu, Z., et al., *Popular deep learning algorithms for disease prediction: a review.* Cluster Computing, 2023. 26(2): p. 1231-1251.
5. Joseph, K., et al. *Deep Learning based Beach Cleaning Robot*. in *2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*. 2023. IEEE.
6. Jing, N., Z. Wu, and H. Wang, *A hybrid model integrating deep learning with investor sentiment analysis for stock price prediction.* Expert Systems with Applications, 2021. 178: p. 115019.
7. Yin, F., et al., *Depression detection in speech using transformer and parallel convolutional neural networks.* Electronics, 2023. 12(2): p. 328.
8. Kayaalp, K. and A. Süzen, *Derin Öğrenme.* Derin Öğrenme ve Türkiye'deki Uygulamaları, Adıyaman, Türkiye: İKSAD Yayınevi, 2018: p. 25-28.
9. Özyildiz, A.G.H. and D.D.G. Sonugür, *Derin Öğrenme Mimarileri Kullanilarak Cilt Kanseri Teşhisi.*
10. Druzhkov, P.N. and V.D. Kustikova, *A survey of deep learning methods and software tools for image*

classification and object detection. Pattern Recognition and Image Analysis, 2016. 26: p. 9-15.

11. Şeker, A., B. Diri, and H.H. Balık, *Derin öğrenme yöntemleri ve uygulamaları hakkında bir inceleme.* Gazi Mühendislik Bilimleri Dergisi, 2017. 3(3): p. 47-64.
12. Yu, D. and L. Deng, *Deep learning and its applications to signal and information processing [exploratory dsp]*. IEEE Signal Processing Magazine, 2010. 28(1): p. 145-154.
13. Fukushima, K., *Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position.* Biological cybernetics, 1980. 36(4): p. 193-202.
14. Hochreiter, S. and J. Schmidhuber, *Long short-term memory.* Neural computation, 1997. 9(8): p. 1735-1780.
15. LeCun, Y., et al., *Handwritten digit recognition with a back-propagation network.* Advances in neural information processing systems, 1989. 2.
16. LeCun, Y., et al., *Gradient-based learning applied to document recognition.* Proceedings of the IEEE, 1998. 86(11): p. 2278-2324.
17. Fei-Fei, L., J. Deng, and K. Li, *ImageNet: Constructing a large-scale image database.* Journal of vision, 2009. 9(8): p. 1037-1037.
18. Aizenberg, I.N., et al., *Multiple-Valued threshold logic and multi-valued neurons.* Multi-Valued and Universal Binary Neurons: Theory, Learning and Applications, 2000: p. 25-80.
19. Hinton, G.E., *Learning multiple layers of representation.* Trends in cognitive sciences, 2007. 11(10): p. 428-434.

20. Ciresan, D., et al., *Deep neural networks segment neuronal membranes in electron microscopy images.* Advances in neural information processing systems, 2012. 25.

21. Ciresan, D.C., et al. *Convolutional neural network committees for handwritten character classification*. in *2011 International conference on document analysis and recognition*. 2011. IEEE.

22. Akın, E. and M.E. Şahin, *Derin öğrenme ve yapay sinir ağı modelleri üzerine bir inceleme.* EMO Bilimsel Dergi, 2024. 14(1): p. 27-38.

23. Chlap, P., et al., *A review of medical image data augmentation techniques for deep learning applications.* Journal of medical imaging and radiation oncology, 2021. 65(5): p. 545-563.

24. Mikołajczyk, A. and M. Grochowski. *Data augmentation for improving deep learning in image classification problem*. in *2018 international interdisciplinary PhD workshop (IIPhDW)*. 2018. IEEE.

25. Nalepa, J., M. Marcinkiewicz, and M. Kawulok, *Data augmentation for brain-tumor segmentation: a review.* Frontiers in computational neuroscience, 2019. 13: p. 83.

26. Shorten, C. and T.M. Khoshgoftaar, *A survey on image data augmentation for deep learning.* Journal of big data, 2019. 6(1): p. 1-48.

27. Sorin, V., et al., *Creating artificial images for radiology applications using generative adversarial networks (GANs)–a systematic review.* Academic radiology, 2020. 27(8): p. 1175-1185.

28. Khosla, C. and B.S. Saini. *Enhancing performance of deep learning models with different data augmentation techniques: A survey*. in *2020 international conference on*

intelligent engineering and management (ICIEM). 2020. IEEE.

29. Shijie, J., et al. *Research on data augmentation for image classification based on convolution neural networks*. in *2017 Chinese automation congress (CAC)*. 2017. IEEE.
30. Lemley, J., S. Bazrafkan, and P. Corcoran, *Smart augmentation learning an optimal data augmentation strategy*. Ieee Access, 2017. 5: p. 5858-5869.
31. Alomar, K., H.I. Aysel, and X. Cai, *Data augmentation in classification and segmentation: A survey and new strategies.* Journal of Imaging, 2023. 9(2): p. 46.
32. Nanni, L., et al., *Comparison of different image data augmentation approaches.* Journal of imaging, 2021. 7(12): p. 254.
33. Goceri, E., *Medical image data augmentation: techniques, comparisons and interpretations.* Artificial Intelligence Review, 2023. 56(11): p. 12561-12605.
34. Mumuni, A. and F. Mumuni, *Data augmentation with automated machine learning: approaches and performance comparison with classical data augmentation methods.* Knowledge and Information Systems, 2025: p. 1-51.
35. Yang, Z., et al., *A survey of automated data augmentation algorithms for deep learning-based image classification tasks.* Knowledge and Information Systems, 2023. 65(7): p. 2805-2861.
36. Cheung, T.-H. and D.-Y. Yeung, *A survey of automated data augmentation for image classification: Learning to compose, mix, and generate.* IEEE transactions on neural networks and learning systems, 2023.

37. Nejati, M., S. Samavi, and S. Shirani, *Multi-focus image fusion using dictionary-based sparse representation.* Information fusion, 2015. 25: p. 72-84.

38. Bagherinezhad, H., et al., *Label refinery: Improving imagenet classification through label progression.* arXiv preprint arXiv:1805.02641, 2018.

39. Takahashi, R., T. Matsubara, and K. Uehara, *Data augmentation using random image cropping and patching for deep CNNs.* IEEE Transactions on Circuits and Systems for Video Technology, 2019. 30(9): p. 2917-2931.

40. Krell, M.M. and S.K. Kim. *Rotational data augmentation for electroencephalographic data*. in *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. 2017. IEEE.

41. Zhong, Z., et al. *Random erasing data augmentation*. in *Proceedings of the AAAI conference on artificial intelligence*. 2020.

42. Lin, C.-Y., et al., *Rotation, scale, and translation resilient watermarking for images.* IEEE Transactions on image processing, 2001. 10(5): p. 767-782.

43. Gong, C., et al. *Keepaugment: A simple information-preserving data augmentation approach*. in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021.

44. Wang, Y., et al., *CNN tracking based on data augmentation.* Knowledge-Based Systems, 2020. 194: p. 105594.

45. Borji, A., *Pros and cons of GAN evaluation measures.* Computer vision and image understanding, 2019. 179: p. 41-65.

46. Hung, S.-K., *Image data augmentation from small training datasets using generative adversarial networks (GANs)*. 2023, University of Essex.

47. Lu, Y., et al., *Generative adversarial networks (GANs) for image augmentation in agriculture: A systematic review.* Computers and Electronics in Agriculture, 2022. 200: p. 107208.

48. Mantegna, M., et al. *Benchmarking GAN-Based vs Classical Data Augmentation on Biomedical Images*. in *International Conference on Pattern Recognition*. 2024. Springer.

49. Wu, H. and S. Prasad, *Semi-supervised deep learning using pseudo labels for hyperspectral image classification.* IEEE Transactions on Image Processing, 2017. 27(3): p. 1259-1270.

50. Van Engelen, J.E. and H.H. Hoos, *A survey on semi-supervised learning.* Machine learning, 2020. 109(2): p. 373-440.

51. Reed, C.J., et al. *Selfaugment: Automatic augmentation policies for self-supervised learning*. in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021.

52. Chadebec, C., et al., *Data augmentation in high dimensional low sample size setting using a geometry-based variational autoencoder.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022. 45(3): p. 2879-2896.

53. Madan, Y., et al. *Synthetic data augmentation of MRI using generative variational autoencoder for Parkinson's disease detection*. in *Evolution In Computational Intelligence: Proceedings Of The 9th International*

Conference On Frontiers In Intelligent Computing: Theory And Applications (FICTA 2021). 2022. Springer.

54. Jackson, P.T., et al. *Style augmentation: data augmentation via style randomization*. in *CVPR workshops*. 2019.
55. Georgievski, B. *Image augmentation with neural style transfer*. in *International Conference on ICT Innovations*. 2019. Springer.
56. Cao, C., et al., *A survey of mix-based data augmentation: Taxonomy, methods, applications, and explainability.* ACM Computing Surveys, 2024. 57(2): p. 1-38.
57. Lewy, D. and J. Mańdziuk, *An overview of mixing augmentation methods and augmentation strategies.* Artificial Intelligence Review, 2023. 56(3): p. 2111-2169.
58. Zhang, H., et al., *mixup: Beyond empirical risk minimization.* arXiv preprint arXiv:1710.09412, 2017.
59. Park, C., S. Yun, and S. Chun, *A unified analysis of mixed sample data augmentation: A loss function perspective.* Advances in neural information processing systems, 2022. 35: p. 35504-35518.
60. Liang, W., Y. Liang, and J. Jia, *MiAMix: enhancing image classification through a multi-stage augmented mixed sample data augmentation method.* Processes, 2023. 11(12): p. 3284.
61. Krizhevsky, A. and G. Hinton, *Learning multiple layers of features from tiny images.* 2009.
62. Krizhevsky, A., V. Nair, and G. Hinton, *Cifar-10 (canadian institute for advanced research).* URL http://www. cs. toronto. edu/kriz/cifar. html, 2010. 5(4): p. 1.

63. Deng, J., et al. *Imagenet: A large-scale hierarchical image database*. in *2009 IEEE conference on computer vision and pattern recognition*. 2009. Ieee.

64. Le, Y. and X. Yang, *Tiny imagenet visual recognition challenge.* CS 231N, 2015. 7(7): p. 3.

65. LeCun, Y., *The MNIST database of handwritten digits.* http://yann. lecun. com/exdb/mnist/, 1998.

66. Lin, T.-Y., et al. *Microsoft coco: Common objects in context*. in *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*. 2014. Springer.

67. Everingham, M., et al., *The pascal visual object classes (voc) challenge.* International journal of computer vision, 2010. 88: p. 303-338.

UNDERSTANDING 3D ROTATIONS: EULER ANGLE CONVENTIONS FOR ESTIMATION, NAVIGATION, AND CONTROL

Tolga ÖZASLAN[1]

1. INTRODUCTION

There are several widely used conventions for representing rotations in 3D space. These include *rotation matrices*, *Euler angles*, *unit quaternions*, the *exponential map*, and the *axis-angle representation* (Diebel et al., 2006). Each of these representations offers distinct advantages depending on the application context.

Among these alternatives, Euler angles are commonly used in the estimation (Marins, Yun, Bachmann, McGhee, & Zyda, 2001), control (Mokhtari & Benallegue, 2004), and navigation of aerial systems such as quadcopters (Özaslan et al., 2017) and fixed-wing platforms, owing to their intuitive construction and ease of interpretation .

Euler angles represent a rotation in 3D as a sequence of elementary rotations about non-parallel axes. These axes correspond to the basis vectors of either the current or fixed frame depending on the chosen convention. For example, roll-pitch-yaw angles, widely used in the avionics literature, parameterize the orientation of an aerial platform through its relative angle about $\hat{\mathbf{x}} - \hat{\mathbf{y}} - \hat{\mathbf{z}}$ basis vectors of a fixed frame (Fig. 1) (Mellinger & Kumar, 2011).

[1]Dr. Öğr. Üyesi, Ankara Yıldırım Beyazıt Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi, Makina Mühendisliği Bölümü, tozaslan@aybu.edu.tr, ORCID: 0000-0002-5002-0670

Although Euler angles representation is easy to interpret, it is deficient in several important respects. Euler angles suffer from gimbal lock a phenomenon that occurs when two of the rotation axes align (Hemingway & O'Reilly, 2018). In such a configuration, one degree-of-freedom (DoF) is lost, making the platform instantaneously uncontrollable along the lost dimension. Secondly, composing multiple rotations using Euler angles is not possible except for a few corner cases. In general configurations, Euler angles must be converted into rotation matrices or quaternions for composition, and then converted back to Euler angles. Also there are twelve distinct conventions for composing elementary rotations which might be a source of confusion if adopted conventions are not defined clearly.

This work focuses on the use of *Euler and Tait-Bryan angles* for representing 3D rotations, with particular emphasis on clearly stating the conventions used in the literature and enabling the reader to competently interpret technical documentation and software implementations involving rotational formulations.

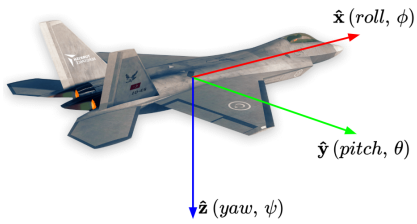


Figure 1: Roll-pitch-yaw (RPY) angles are widely used in avionics literature. This convention parametrizes the platform orientation as a sequence of elementary rotations applied in the `xyz` order with respect to a fixed (inertial) frame.

2. ELEMENTARY ROTATIONS

A reference frame is define through its origin and basis vectors. In Fig. 1 a reference frame, $\mathcal{B}$, is attached to an airplane body, with the basis vector $\hat{\mathbf{x}} - \hat{\mathbf{y}} - \hat{\mathbf{z}}$. In kinematic and dynamic formulations, the three basis vectors are chosen to be mutually orthogonal, *i.e.* $\hat{\mathbf{x}} \cdot \hat{\mathbf{y}} = 0, \hat{\mathbf{y}} \cdot \hat{\mathbf{z}} = 0$ and $\hat{\mathbf{x}} \cdot \hat{\mathbf{z}} = 0$, and have unit norms, *i.e.* $\hat{\mathbf{x}} \cdot \hat{\mathbf{x}} = \hat{\mathbf{y}} \cdot \hat{\mathbf{y}} = \hat{\mathbf{z}} \cdot \hat{\mathbf{z}} = 1$. Lastly, the basis vectors form a right-handed triad, *i.e.* $\hat{\mathbf{x}} \times \hat{\mathbf{y}} = \hat{\mathbf{z}}$, $\hat{\mathbf{y}} \times \hat{\mathbf{z}} = \hat{\mathbf{x}}$ and $\hat{\mathbf{z}} \times \hat{\mathbf{x}} = \hat{\mathbf{y}}$. Elementary rotations occur about one of the basis vectors of a given reference frame.

The axis about which the rotation occurs is unaffected, while the other two basis vectors rotate as shown in Fig. 2. We can obtain the respective rotation matrices through observing the coordinates of these basis vectors after rotation. For example, the vectors $\hat{\mathbf{y}}$ and $\hat{\mathbf{z}}$ after rotating about $\hat{\mathbf{x}}$ by γ radians can be written, in terms of their initial values, as

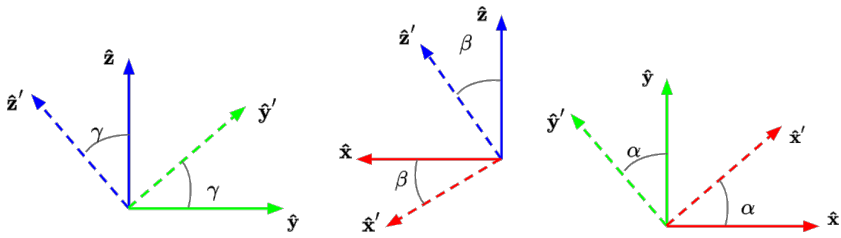
$$\hat{\mathbf{x}}' = \Big[1, 0, 0\Big]^\top \ , \ \hat{\mathbf{y}}' = \Big[0, c(\gamma), s(\gamma)\Big]^\top \ , \ \hat{\mathbf{z}}' = \Big[0, -s(\gamma), c(\gamma)\Big]^\top . \tag{1}$$

This relation can easily be confirmed by observing Fig. 2a. Stacking these vectors side-by-side one can obtain the elementary rotation matrix about the respective axis as given below

$$\mathbf{R}_{\hat{\mathbf{x}}}(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\gamma) & -s(\gamma) \\ 0 & s(\gamma) & c(\gamma) \end{bmatrix} \quad , \quad \mathbf{R}_{\hat{\mathbf{y}}}(\beta) = \begin{bmatrix} c(\beta) & 0 & s(\beta) \\ 0 & 1 & 0 \\ -s(\beta) & 0 & c(\beta) \end{bmatrix}$$

$$\mathbf{R}_{\hat{\mathbf{z}}}(\alpha) = \begin{bmatrix} c(\alpha) & -s(\alpha) & 0 \\ s(\alpha) & c(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

where the subscripts denote the axis of rotation.



(a) $\hat{\mathbf{x}}$-axis (b) $\hat{\mathbf{y}}$-axis (c) $\hat{\mathbf{z}}$-axis

Figure 2: The effect of elementary rotations on the other two basis vectors: from left to right, rotations about the $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{z}}$ axes.

3. COMPOSING ROTATIONS

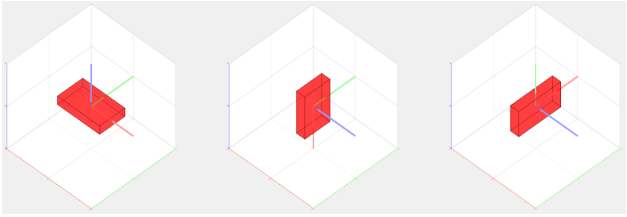
Rotations can be combined through matrix multiplication. It should be noted that the order in which the rotation matrices are multiplied is important since matrix multiplication is non-commutative except for special cases, *i.e.* $\mathbf{A} \cdot \mathbf{B} \neq \mathbf{B} \cdot \mathbf{A}$ for most $\mathbf{A}$ and $\mathbf{B}$. As an example, consider the two elementary rotations $\mathbf{R}_{\hat{\mathbf{y}}}\left(\frac{\pi}{2}\right)$ and $\mathbf{R}_{\hat{\mathbf{z}}}\left(\frac{\pi}{2}\right)$ applied at different orders on an a reference frame initially at identity orientation as shown in Fig. 3. The elementary rotation matrices are

$$\mathbf{R}_{\hat{\mathbf{y}}}\left(\frac{\pi}{2}\right) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \quad , \quad \mathbf{R}_{\hat{\mathbf{z}}}\left(\frac{\pi}{2}\right) = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

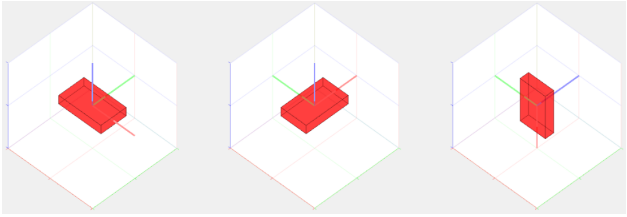
which results in the following orientations

$$\mathbf{R}_{\hat{\mathbf{y}}}\left(\pi/2\right)\mathbf{R}_{\hat{\mathbf{z}}}\left(\pi/2\right) = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \mathbf{R}_{\hat{\mathbf{z}}}\left(\pi/2\right)\mathbf{R}_{\hat{\mathbf{y}}}\left(\pi/2\right) = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \end{bmatrix}. \tag{4}$$

It is evident from both geometric and algebraic perspectives that the order in which rotations are applied is critical and may lead to different final orientations.



(a) Rotations about $\hat{\mathbf{y}}$ and then $\hat{\mathbf{z}}$.



(b) Rotations about $\hat{\mathbf{z}}$ and then $\hat{\mathbf{y}}$.

Figure 3: Consecutive elementary rotations applied on a rectangular prism about $\hat{\mathbf{y}}$ and $\hat{\mathbf{z}}$ at different orders to demonstrate that rotations are not commutative. It should be noted that the second rotation is applied with respect to the *current frame*.

3.1. Intrinsic and Extrinsic Rotations

In the above discussion, although the resulting orientations differ, both compositions produce valid rotation matrices. This underscores a concept of fundamental importance in rotational kinematics: the outcome of successive rotations depends critically on their order. Equally important is the *reference frame* with respect to which each rotation is applied. Specifically, a rotation can be performed either relative to the *fixed (inertial) frame*, referred to as an *extrinsic rotation*, or relative to the *current (rotating) frame*, which is known as an *intrinsic rotation*. This distinction leads to different interpretations and implementations, particularly in applications such as robotics, aerospace, and computer graphics.

In the case of *intrinsic rotations*, where each new rotation is applied relative to the current (rotating) frame, composition is

performed by multiplying the existing rotation matrix from the *right* by the new rotation. Conversely, for *extrinsic rotations*, where rotations are applied relative to the fixed (inertial) frame, composition is performed by multiplying the new rotation from the *left*.

Proof: Consider a rotation $\mathbf{R}_i$ applied to an initial identity orientation, where i denotes the axis of rotation aligned with a basis vector of the fixed (initial) frame. To apply a new rotation, $\mathbf{R}_j$ where j is a basis vector, with respect to the *fixed frame*, we must virtually align the current (rotated) frame with the fixed frame before applying the new rotation.

This alignment is achieved by *undoing* the current orientation, which can be performed by right-multiplying the current rotation matrix by its inverse, $\mathbf{R}_i^{-1}$. Although this may seem trivial, it allows us to apply the new elementary rotation matrix, $\mathbf{R}_j$, with respect to the fixed frame (which, at that moment, is effectively the current frame).

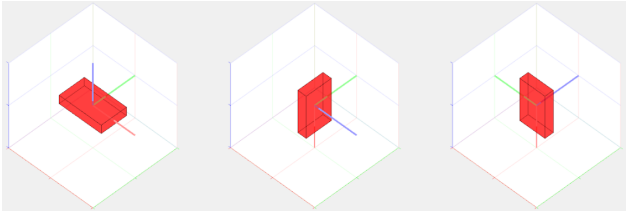
After applying the new rotation, $\mathbf{R}_j$, we reapply the original orientation, $\mathbf{R}_i$, by right-multiplying the result with the original rotation matrix. In total, this composition corresponds to *left-multiplication* of the new rotation matrix onto the current orientation, thereby validating the composition rule for extrinsic rotations. These steps are summarized in Table 1

In order to compare the effect of order of rotations, we will again consider the two elementary rotations $\mathbf{R}_{\hat{\mathbf{y}}}\left(\frac{\pi}{2}\right)$ and $\mathbf{R}_{\hat{\mathbf{z}}}\left(\frac{\pi}{2}\right)$ applied at different orders, but applied with respect to the fixed frame. The resultant rotation matrices write

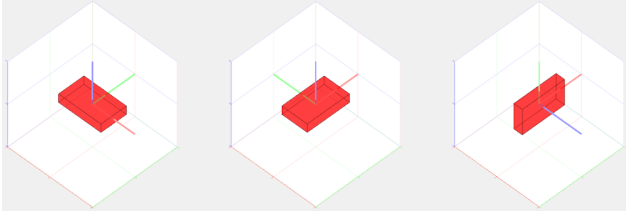
$$\mathbf{R}_{\hat{\mathbf{y}}}\left(\pi/2\right)\mathbf{R}_{\hat{\mathbf{z}}}\left(\pi/2\right)=\begin{bmatrix}0&0&1\\1&0&0\\0&1&0\end{bmatrix},\mathbf{R}_{\hat{\mathbf{z}}}\left(\pi/2\right)\mathbf{R}_{\hat{\mathbf{y}}}\left(\pi/2\right)=\begin{bmatrix}0&-1&0\\0&0&1\\-1&0&0\end{bmatrix}. \tag{5}$$

Step	Current Orientation	Explanation
1	$\mathbf{R}^{(1)} = \mathbf{R}_i$	Initial rotation about axis i.
2	$\mathbf{R}^{(2)} = \mathbf{R}_i \cdot \mathbf{R}_i^{-1}$	Undo the effect of the initial rotation. This temporarily aligns the current frame with the fixed (initial) frame.
3	$\mathbf{R}^{(3)} = \mathbf{R}_i \cdot \mathbf{R}_i^{-1} \cdot \mathbf{R}_j$	Apply a new rotation about axis j, one of the axes of the fixed (initial) frame.
4	$\mathbf{R}^{(4)} = \left(\mathbf{R}_i \cdot \mathbf{R}_i^{-1}\right) \cdot \mathbf{R}_j \cdot \mathbf{R}_i$	Reapply the undone rotation to restore the frame to its original orientation before the new rotation.
5	$\mathbf{R}^{(4)} = \mathbf{R}_j \mathbf{R}_i$	Since $\mathbf{R}_i \cdot \mathbf{R}_i^{-1} = \mathbf{I}$, we simplify to obtain the standard extrinsic rotation composition.

Table 1: Steps illustrating extrinsic rotation using the logic of undoing and reapplying previously applied rotations. Orientation at each step is denoted by $\mathbf{R}^{(n)}$ where n is the proof step.



(a) Rotations about fixed $\hat{\mathbf{y}}$ and then fixed $\hat{\mathbf{z}}$.



(b) Rotations about fixed $\hat{\mathbf{z}}$ and then fixed $\hat{\mathbf{y}}$.

Figure 4: Consecutive elementary rotations applied about fixed $\hat{\mathbf{y}}$ and fixed $\hat{\mathbf{z}}$ axes demonstrating *extrinsic rotation*.

The first equations is obtained by first applying a rotation about fixed-$\hat{\mathbf{z}}$ and then about fixed-$\hat{\mathbf{y}}$; and the second one rotating about the fixed-$\hat{\mathbf{y}}$ and then fixed-$\hat{\mathbf{z}}$. As these examples illustrate, applying rotations in reverse order under the opposite convention yields the same final orientation.

Remark 1. Applying a sequence of rotations *all* in the intrinsic convention is equivalent to applying the same sequence in reverse order using the extrinsic convention, and vice versa. This equivalence yields the same final orientation despite the differing frame of reference.

Example: Now let's work out a more comprehensive example involving five sequential elementary rotations, mixing intrinsic (current frame) and extrinsic (fixed frame) conventions. The superscript on each rotation axis indicates the index of the intermediate frame with respect to which the rotation is applied, and the subscript indicates the rotation axis. The initial orientation is de-

noted as $\mathbf{R}^{(0)} = \mathbf{I}$ and omitted in the following calculations since the identity rotation has no effect.

Step 1: Rotate by α about the current $\hat{\mathbf{x}}$-axis:

$$\begin{aligned}\mathbf{R}^{(1)} &= \mathbf{R}^{(0)} \cdot \mathbf{R}_{\hat{\mathbf{x}}^{(0)}}(\alpha) \\ &= \mathbf{R}_{\hat{\mathbf{x}}^{(0)}}(\alpha)\end{aligned} \tag{6}$$

It should be noted that $\hat{\mathbf{x}}^{(0)} = \hat{\mathbf{x}}^{(1)}$ since the axis of rotation is an eigenvector with identity eigenvalue of the respective rotation matrix, thus is not affected. In the subsequent steps, we retain the index from the previous step as a matter of notational consistency, even though the axis itself does not change.

Step 2: Rotate by γ about the current $\hat{\mathbf{y}}$-axis (intrinsic rotation):

$$\begin{aligned}\mathbf{R}^{(2)} &= \mathbf{R}^{(1)} \cdot \mathbf{R}_{\hat{\mathbf{y}}^{(1)}}(\gamma) \\ &= \mathbf{R}_{\hat{\mathbf{x}}^{(0)}}(\alpha) \cdot \mathbf{R}_{\hat{\mathbf{y}}^{(1)}}(\gamma)\end{aligned} \tag{7}$$

which is equivalent to writing

$$\begin{aligned}\mathbf{R}^{(2)} &= \mathbf{R}_{\hat{\mathbf{x}}^{(1)}}(\alpha) \cdot \mathbf{R}_{\hat{\mathbf{y}}^{(1)}}(\gamma) \quad \text{and} \\ &= \mathbf{R}_{\hat{\mathbf{x}}^{(1)}}(\alpha) \cdot \mathbf{R}_{\hat{\mathbf{y}}^{(2)}}(\gamma)\end{aligned} \tag{8}$$

due to the discussion in the previous step.

Step 3: Rotate by θ about the fixed $\hat{\mathbf{z}}$-axis (extrinsic rotation): We first undo the effect of the previously applied rotations, through right-multiplication by $\left(\mathbf{R}^{(2)}\right)^{-1}$, apply the new elementary rotation about the fixed $\hat{\mathbf{z}}$-axis, and then reapply the undone transformations, $\mathbf{R}^{(2)}$. This process is formalized as follows:

$$\begin{aligned}\mathbf{R}^{(3)} &= \mathbf{R}^{(2)} \cdot \left[\left(\mathbf{R}^{(2)}\right)^{-1} \cdot \mathbf{R}_{\hat{\mathbf{z}}^{(0)}}(\theta) \cdot \left(\mathbf{R}^{(2)}\right)\right] \\ &= \mathbf{R}_{\hat{\mathbf{z}}^{(0)}}(\theta) \cdot \left(\mathbf{R}^{(2)}\right) \\ &= \mathbf{R}_{\hat{\mathbf{z}}^{(0)}}(\theta) \cdot \mathbf{R}_{\hat{\mathbf{x}}^{(0)}}(\alpha) \cdot \mathbf{R}_{\hat{\mathbf{y}}^{(1)}}(\gamma)\end{aligned} \tag{9}$$

Observe that the first pair of matrices evaluated to identity and are excluded. Thus, we conclude that applying a rotation about a fixed axis is equivalent to pre-multiplying the corresponding elementary rotation matrix with the current orientation.

Step 4: Rotate by β about the current $\hat{\mathbf{y}}$-axis (intrinsic rotation):

$$\begin{aligned}\mathbf{R}^{(4)} &= \mathbf{R}^{(3)} \cdot \mathbf{R}_{\hat{\mathbf{y}}^{(3)}}(\beta) \\ &= \mathbf{R}_{\hat{\mathbf{z}}^{(0)}}(\theta) \cdot \mathbf{R}_{\hat{\mathbf{x}}^{(0)}}(\alpha) \cdot \mathbf{R}_{\hat{\mathbf{y}}^{(1)}}(\gamma) \cdot \mathbf{R}_{\hat{\mathbf{y}}^{(3)}}(\beta) \end{aligned} \tag{10}$$

Step 5: Rotate by δ about the fixed $\hat{\mathbf{x}}$-axis (intrinsic rotation):

$$\begin{aligned}\mathbf{R}^{(5)} &= \mathbf{R}^{(4)} \cdot \left[\left(\mathbf{R}^{(4)}\right)^{-1} \cdot \mathbf{R}_{\hat{\mathbf{x}}^{(0)}}(\delta) \cdot \left(\mathbf{R}^{(4)}\right) \right] \\ &= \mathbf{R}_{\hat{\mathbf{x}}^{(0)}}(\delta) \cdot \mathbf{R}_{\hat{\mathbf{z}}^{(0)}}(\theta) \cdot \mathbf{R}_{\hat{\mathbf{x}}^{(0)}}(\alpha) \cdot \mathbf{R}_{\hat{\mathbf{y}}^{(1)}}(\gamma) \cdot \mathbf{R}_{\hat{\mathbf{y}}^{(3)}}(\beta) \end{aligned} \tag{11}$$

The resultant rotation matrix in Eqn. 11 could have been obtained through other rotation sequences. One possible such sequence is composition of intrinsic elementary rotations about the axes and angles as they appear in Eqn. 11. That is, rotation about $\hat{\mathbf{x}}$ by δ, then $\hat{\mathbf{z}}$ by θ, then $\hat{\mathbf{x}}$ by α, then $\hat{\mathbf{y}}$ by γ and finally about $\hat{\mathbf{y}}$ by β radians, all with respect to the current (rotating) frame gives the same orientation. These observations will be useful when we discuss different Euler angle conventions in the subsequent sections.

4. EULER AND TAIT-BRYAN ANGLES

Leonhard Euler, in his seminal work in 1776, states that orientations of rigid bodies in three dimensional space can be represented using successively applied three elementary rotations about coordinate axes (basis vectors) (Pio, 1966). Euler primarily studied the `zxz` sequence where the angles about each axes are called yaw (ϕ), nutation (θ) and precession (ψ). Structures with the first and the last axes being equal are commonly referred to as a *proper Euler angle sequence*, and the elementary rotations are applied with

respect to the current frame unless otherwise is explicitly stated. Proper Euler angle are widely used in application areas such as classical and celestial mechanics, and analytical dynamics.

Later in the late 19^{th} and early 20^{th} centuries, researchers Peter Guthrie Tait and George Hartley Bryan proposed variations with distinct axes of rotations. Tait and Bryan extensively studied the xyz sequence which is now referred to as *roll-pitch-yaw* angles. Angles are usually represented with symbols $\phi - \theta - \psi$. Sequences with distinct axes of rotations became particularly useful in the emerging fields of aviation and navigation (Fig. 1).

4.1. Possible Sequences and Conventions

There are a total of $3^3 = 27$ possible axis sequences of three rotations, such as xyx, xyz, xzx, and xzy. However, only twelve of these constitute valid Euler or Tait-Bryan angle sequences. The remaining fifteen sequences involve repeated adjacent axes, such as xxx, xxy, or xyy, and are not used in practice due to lack of independent rotational degrees of freedom hence such sequences have either 1 or 2 DoF). Among the twelve valid configurations, six are classified as *proper Euler angle* sequences , and the other six as *Tait-Bryan angle* sequences. The former class has the first and third rotation axes the same and the latter have all three rotation axes distinct Table 2.

Each class of sequences, Euler and Tait-Bryan, can be formulated using either intrinsic or extrinsic composition conventions. However due to various reasons such as terminological consistency, conventions used in the literature and software packages and interpretability, intrinsic composition convention is preferred unless otherwise is clearly stated. As explained in the previous sections, it is always possible to express any given sequence in the reverse order thereby switching from intrinsic to extrinsic con-

vention, and vice versa, when required. In summary, while an extrinsic interpretation of Euler and Tait-Bryan angles can be defined consistently from a mathematical standpoint, its use should be stated clearly avoid misinterpretation.

Irrespective of the conventions used, only three angles along with the axes of rotations, suffice to represent any rotation in $\mathcal{SO}(3)$ making Euler/Tait-Bryan angles minimal in terms of representation.

Category	**Axis Sequences**
Improper Sequences	`xxx`, `xxy`, `xxz`, `yyx`, `yyy`, `yyz`, `zzx`, `zzy`, `zzz`, `xyy`, `yxx`, `yzz`, `zyy`, `xxz`
Proper Euler Sequences	`zxz`, `xyx`, `yzy`, `zyz`, `xzx`, `yxy`
Tait–Bryan Sequences	`xyz`, `xzy`, `yxz`, `yzx`, `zxy`, `zyx`

Table 2: Categorization of 3-element axis sequences into improper, proper Euler, and Tait-Bryan categories.

4.2. `zyz` Euler Angle Sequence:

In this section we will derive the rotation matrix for the `zyz` Euler angle sequence. Embracing the general convention we apply the elementary rotations with respect to the current frame and the angles represented as yaw (ϕ), nutation (θ) and precession (ψ). The

```
double phi = ..., theta = ..., psi = ...;

Eigen::AngleAxisd Rz1(phi, Eigen::Vector3d::UnitZ());
Eigen::AngleAxisd Ry(theta, Eigen::Vector3d::UnitY());
Eigen::AngleAxisd Rz2(psi, Eigen::Vector3d::UnitZ());

Eigen::Matrix3d R = Rz1 * Ry * Rz2;
```

Listing 1: c++ code snippet for ZYZ intrinsic rotation using Eigen library

elementary rotation matrices are written as

$$\mathbf{R}_{\hat{\mathbf{z}}^{(0)}}(\phi) = \begin{bmatrix} c(\phi) & -s(\phi) & 0 \\ s(\phi) & c(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad , \quad \mathbf{R}_{\hat{\mathbf{y}}^{(1)}}(\theta) = \begin{bmatrix} c(\theta) & 0 & s(\theta) \\ 0 & 1 & 0 \\ -s(\theta) & 0 & c(\theta) \end{bmatrix}$$
$$\mathbf{R}_{\hat{\mathbf{z}}^{(2)}}(\psi) = \begin{bmatrix} c(\psi) & -s(\psi) & 0 \\ s(\psi) & c(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{12}$$

The resultant rotation matrix is obtained as

$$\mathbf{R}_{\mathtt{zyz}}(\phi,\theta,\psi) = \mathbf{R}_{\hat{\mathbf{z}}^{(0)}}(\phi) \cdot \mathbf{R}_{\hat{\mathbf{y}}^{(1)}}(\theta) \cdot \mathbf{R}_{\hat{\mathbf{z}}^{(2)}}(\psi) \tag{13}$$
$$= \begin{bmatrix} c(\phi)c(\theta)c(\psi) - s(\phi)s(\psi) & -c(\phi)c(\theta)s(\psi) - s(\phi)c(\psi) & c(\phi)s(\theta) \\ s(\phi)c(\theta)c(\psi) + c(\phi)s(\psi) & -s(\phi)c(\theta)s(\psi) + c(\phi)c(\psi) & s(\phi)s(\theta) \\ -s(\theta)c(\psi) & s(\theta)s(\psi) & c(\theta) \end{bmatrix}$$

The same resultant matrix could have been obtain by using the ZYZ sequence with extrinsic convention and angles ψ, θ, ϕ which would write

$$\mathbf{R}^{(ext)}_{\mathtt{zyz}}(\psi,\phi,\theta) = \mathbf{R}_{\hat{\mathbf{z}}^{(0)}}(\phi) \cdot \mathbf{R}_{\hat{\mathbf{y}}^{(0)}}(\theta) \cdot \mathbf{R}_{\hat{\mathbf{z}}^{(0)}}(\psi) \tag{14}$$

We provide a c++ code snippet in Eigen (Guennebaud, Jacob, et al., n.d.) for this sequence with intrinsic convention in Listing 1.

4.2.1. Solving for zyz Euler Angles:

We have shown how to obtain the rotation matrix given the Euler angles for the zyz sequence in Equation 13. The inverse problem, however, requires extracting the angles from a given rotation matrix. Upon inspecting the structure of Equation 13, it becomes evident that the top-left 2×2 submatrix is heavily coupled, making it unsuitable for direct analytical inversion. In contrast, other elements of the matrix exhibit clearer patterns that can be exploited to solve for the individual Euler angles.

Before we proceed further, we enumerate the elements of the matrix as

$$\mathbf{R} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}. \tag{15}$$

Next we observe the elements R_{11}, R_{21} and R_{31}, R_{32} all of which include $s(\theta)$. Thus, for $\theta = \frac{(2n+1)\pi}{2}$ for $n \in \mathbb{Z}$ all of these terms become 0 which is a special case. Depending on the sign of $s(\theta)$, we can determine the angles using different formula. If $s(\theta) > 0$, or equivalently $\theta \in (0, \pi)$, the solution take the following form

$$\begin{aligned} \phi = \text{atan2}(R_{23}, R_{13}) \quad , \quad \psi = \text{atan2}(R_{32}, -R_{31}) \\ \theta = \text{atan2}\left(\sqrt{R_{13}^2 + R_{23}^2}, R_{33}\right). \end{aligned} \tag{16}$$

On the other hand if $s(\theta) < 0$, or equivalently $\theta \in (\pi, 2\pi)$, we get

$$\begin{aligned} \phi = \text{atan2}(-R_{23}, -R_{13}) \quad , \quad \psi = \text{atan2}(-R_{32}, R_{31}) \\ \theta = \text{atan2}\left(-\sqrt{R_{13}^2 + R_{23}^2}, R_{33}\right) \end{aligned} \tag{17}$$

In the special case that $s(\theta) = 0$, either $\theta = 2n\pi$ or $\theta = (2n+1)\pi$

for $n \in \mathbb{Z}$. In the former case Eqn. 13 takes the following form

$$\begin{aligned}
&\mathbf{R}_{\mathtt{zyz}}(\phi, 2n\pi, \psi) \\
&\quad = \begin{bmatrix} c(\phi)c(\psi) - s(\phi)s(\psi) & -c(\phi)s(\psi) - s(\phi)c(\psi) & 0 \\ s(\phi)c(\psi) + c(\phi)s(\psi) & -s(\phi)s(\psi) + c(\phi)c(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
&\quad = \begin{bmatrix} \cos(\phi+\psi) & -\sin(\phi+\psi) & 0 \\ \sin(\phi+\psi) & \cos(\phi+\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (18)
\end{aligned}$$

and the latter case gives

$$\begin{aligned}
&\mathbf{R}_{\mathtt{zyz}}(\phi, (2n+1)\pi, \psi) \\
&\quad = \begin{bmatrix} -c(\phi)c(\psi) - s(\phi)s(\psi) & c(\phi)s(\psi) - s(\phi)c(\psi) & 0 \\ -s(\phi)c(\psi) + c(\phi)s(\psi) & s(\phi)s(\psi) + c(\phi)c(\psi) & 0 \\ 0 & 0 & -1 \end{bmatrix} \\
&\quad = \begin{bmatrix} -\cos(\phi-\psi) & -\sin(\phi-\psi) & 0 \\ -\sin(\phi-\psi) & \cos(\phi-\psi) & 0 \\ 0 & 0 & -1 \end{bmatrix}. \qquad (19)
\end{aligned}$$

It clearly observed in the special case $s(\theta) = 0$, one degree of freedom is lost since for a given rotation matrix, one can only solve for the two quantities, θ and $\phi \pm \psi$. Individual values of ϕ and ψ cannot be determined and this is called a *gimbal lock*.

4.2.2. Gimbal Lock for zyz Euler Sequence

A *gimbal lock* occurs when two of the rotation axes align. In the case of zyz Euler sequence, this happened when the first and the last z axes align. We have shown in the previous section that $\theta = 2n\pi$ and $\theta = (2n+1)\pi$ result in two different gimbal locks. In the former case the intermediate rotation never happens since $\mathbf{R}^{(1)}_{\hat{\mathbf{y}}}(2n\pi) = \mathbf{I}$ causing $\hat{\mathbf{z}}$ to remain unrotated, *i.e.* $\hat{\mathbf{z}}^{(0)} = \hat{\mathbf{z}}^{(2)}$. In

the latter case $\mathbf{R}_{\hat{\mathbf{y}}}^{(1)}(2n\pi)$ results in $\hat{\mathbf{z}}^{(0)} = -\hat{\mathbf{z}}^{(2)}$, *i.e.* the first and the last axes of rotations become anti-parallel. In both cases only one rotation about the same (or anti-parallel) axis.

5. EULER ANGLE KINEMATICS

In a typical attitude estimation scenario which employs an Inertial Measurement Unit (IMU), an onboard gyroscope provides instantaneous angular velocity measurements which can be integrated over time to estimate the platform orientation. While there is huge literature on attitude estimation using accelerometer, gyroscope and magnetometer which are usually packed into a single IMU, the theoretical fundamentals and vast technical nuances are beyond the scope of the this work. However, we present the relation between angular rates measured in sensor/body frame ($\mathcal{B}$) and the Euler angles. We will consider the zyz sequence to study this.

In order to obtain such a relation we need to write each of the intermediate rotation axes with respect to the sensor/body frame, $\mathcal{B}$. We will use the notation introduced in the rest of the work and enumerate axes with the index of the intermediate reference frames. For example, the first, second and last rotations occur about $\hat{\mathbf{z}}^{(0)}$, $\hat{\mathbf{y}}^{(1)}$ and $\hat{\mathbf{z}}^{(2)}$. Since these are the axes of rotation of the respective step, they, respectively, are equal to $\hat{\mathbf{z}}^{(1)}$, $\hat{\mathbf{y}}^{(2)}$ and $\hat{\mathbf{z}}^{(3)}$. Let's express each of these with respect to the body frame basis vectors which are $\hat{\mathbf{x}}^{(3)}$, $\hat{\mathbf{y}}^{(3)}$ and $\hat{\mathbf{z}}^{(3)}$ as the final frame is by definition the body frame, $\mathcal{B}$. The basis vectors at the intermediate steps 1, 2 and 3 are related as

$$\begin{bmatrix} \hat{\mathbf{x}}^{(2)} \\ \hat{\mathbf{y}}^{(2)} \\ \hat{\mathbf{z}}^{(2)} \end{bmatrix} = \begin{bmatrix} c(\psi) & -s(\psi) & 0 \\ s(\psi) & c(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}^{\top} \begin{bmatrix} \hat{\mathbf{x}}^{(3)} \\ \hat{\mathbf{y}}^{(3)} \\ \hat{\mathbf{z}}^{(3)} \end{bmatrix} \tag{20}$$

$$\begin{bmatrix} \hat{\mathbf{x}}^{(1)} \\ \hat{\mathbf{y}}^{(1)} \\ \hat{\mathbf{z}}^{(1)} \end{bmatrix} = \begin{bmatrix} c(\theta) & 0 & s(\theta) \\ 0 & 1 & 0 \\ -s(\theta) & 0 & c(\theta) \end{bmatrix}^{\top} \begin{bmatrix} \hat{\mathbf{x}}^{(2)} \\ \hat{\mathbf{y}}^{(2)} \\ \hat{\mathbf{z}}^{(2)} \end{bmatrix} \tag{21}$$

where the rotation matrices, respectively, are $\mathbf{R}^{\top}_{\hat{\mathbf{z}}^{(2)}}$ and $\mathbf{R}^{\top}_{\hat{\mathbf{y}}^{(1)}}$. The body angular velocity in terms of the Euler angular rates is

$$\omega = \dot{\phi}\hat{\mathbf{z}}^{(1)} + \dot{\theta}\hat{\mathbf{y}}^{(2)} + \dot{\psi}\hat{\mathbf{z}}^{(3)}. \tag{22}$$

At this point we need to express all vectors in the above expression in terms of basis vectors of the final, 3^{rd}, frame as

$$\hat{\mathbf{z}}^{(1)} = s(\theta)\hat{\mathbf{x}}^{(2)} + c(\theta)\hat{\mathbf{z}}^{(2)} \tag{23}$$

$$\hat{\mathbf{y}}^{(2)} = -s(\psi)\hat{\mathbf{x}}^{(3)} + c(\psi)\hat{\mathbf{y}}^{(3)}. \tag{24}$$

But this requires us to write the following relations as well

$$\hat{\mathbf{x}}^{(2)} = c(\psi)\hat{\mathbf{x}}^{(3)} + s(\psi)\hat{\mathbf{y}}^{(3)} \tag{25}$$

$$\hat{\mathbf{z}}^{(2)} = \hat{\mathbf{z}}^{(3)} \tag{26}$$

which gives

$$\begin{aligned} \hat{\mathbf{z}}^{(1)} &= s(\theta)\Big(c(\psi)\hat{\mathbf{x}}^{(3)} + s(\psi)\hat{\mathbf{y}}^{(3)}\Big) + c(\theta)\hat{\mathbf{z}}^{(3)} \\ &= s(\theta)c(\psi)\hat{\mathbf{x}}^{(3)} + s(\theta)s(\psi)\hat{\mathbf{y}}^{(3)} + c(\theta)\hat{\mathbf{z}}^{(3)}. \end{aligned} \tag{27}$$

Thus the angular velocity in Euler angle rates can be written as

$$\begin{aligned} \omega &= \dot{\phi}\Big(s(\theta)c(\psi)\hat{\mathbf{x}}^{(3)} + s(\theta)s(\psi)\hat{\mathbf{y}}^{(3)} + c(\theta)\hat{\mathbf{z}}^{(3)}\Big) \\ &\quad + \dot{\theta}\Big(- s(\psi)\hat{\mathbf{x}}^{(3)} + c(\psi)\hat{\mathbf{y}}^{(3)}\Big) + \dot{\psi}\hat{\mathbf{z}}^{(3)} \end{aligned} \tag{28}$$

$$\begin{aligned} &= \hat{\mathbf{x}}^{(3)}\Big(\dot{\phi}s(\theta)c(\psi) - \dot{\theta}s(\psi)\Big) + \\ &\quad \hat{\mathbf{y}}^{(3)}\Big(\dot{\phi}s(\theta)s(\psi) + \dot{\theta}c(\psi)\Big) + \hat{\mathbf{z}}^{(3)}\Big(\dot{\phi}\,c(\theta) + \dot{\psi}\Big). \end{aligned} \tag{29}$$

In the robotics literature the angular rates as measured by a gyroscope are usually denoted $[p, q, r]^{\top}$ and defined in the 3^{rd} frame

(also $\mathcal{B}$). Hence these and the Euler angular rates are now defined in the same frame and can be related as

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} s(\theta)c(\psi) & -s(\psi) & 0 \\ s(\theta)s(\psi) & c(\psi) & 0 \\ c(\theta) & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}. \tag{30}$$

Having this relation established, gyroscope measurements can be transformed into Euler angular rates which then can be numerically integrated to directly update the orientation estimate in Euler domain.

6. CONCLUSION

Euler angles are widely used in the robotics and computer vision communities as a means of representing orientation in three-dimensional space. Their appeal lies in their intuitive geometric interpretation and ease of visualization. However, for algorithmic and computational purposes, Euler angles are often converted into more tractable representations such as rotation matrices or unit quaternions.

In this work, we focused on the zyz Euler angle convention. We derived the associated rotation matrix, examined the inverse problem of recovering Euler angles from a given rotation matrix, and analyzed singular configurations and corner cases that arise in practical applications. Furthermore, we established a relationship between angular velocity, typically measured by a gyroscope, and the time derivatives of the zyz Euler angles. This derivation provides a useful bridge between sensor measurements and orientation parameters.

The methodology and illustrative examples presented in this work can be extended to other Euler sequences. We hope this work serves as a foundational reference for researchers and practi-

tioners seeking to understand and employ Euler angle conventions in 3D orientation estimation tasks.

REFERENCES

Diebel, J., et al. (2006). Representing attitude: Euler angles, unit quaternions, and rotation vectors. *Matrix*, *58*(15-16), 1–35.

Guennebaud, G., Jacob, B., et al. (n.d.). *Eigen v3*. `https://eigen.tuxfamily.org`. (Accessed: 2025-06-23)

Hemingway, E. G., & O'Reilly, O. M. (2018). Perspectives on euler angle singularities, gimbal lock, and the orthogonality of applied forces and applied moments. *Multibody system dynamics*, *44*, 31–56.

Marins, J. L., Yun, X., Bachmann, E. R., McGhee, R. B., & Zyda, M. J. (2001). An extended kalman filter for quaternion-based orientation estimation using marg sensors. In *Proceedings 2001 ieee/rsj international conference on intelligent robots and systems. expanding the societal role of robotics in the the next millennium (cat. no. 01ch37180)* (Vol. 4, pp. 2003–2011).

Mellinger, D., & Kumar, V. (2011). Minimum snap trajectory generation and control for quadrotors. In *2011 ieee international conference on robotics and automation* (pp. 2520–2525).

Mokhtari, A., & Benallegue, A. (2004). Dynamic feedback controller of euler angles and wind parameters estimation for a quadrotor unmanned aerial vehicle. In *Ieee international conference on robotics and automation, 2004. proceedings. icra'04. 2004* (Vol. 3, pp. 2359–2366).

Özaslan, T., Loianno, G., Keller, J., Taylor, C. J., Kumar, V., Wozencraft, J. M., & Hood, T. (2017). Autonomous navi-

gation and mapping for inspection of penstocks and tunnels with mavs. *IEEE Robotics and Automation Letters*, *2*(3), 1740–1747.

Pio, R. (1966). Euler angle transformations. *IEEE Transactions on automatic control*, *11*(4), 707–715.

AKADEMİK PERSPEKTİFTEN

BİLGİSAYAR BİLİMLERİ VE MÜHENDİSLİĞİ

yaz
yayınları

YAZ Yayınları
M.İhtisas OSB Mah. 4A Cad. No:3/3
İscehisar / AFYONKARAHİSAR
Tel : (0 531) 880 92 99
yazyayinlari@gmail.com • www.yazyayinlari.com