



MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
DERİN ÖĞRENME DERSİ PROJESİ

BİTKİ HASTALIKLARI TESPİTİ VE SINIFLANDIRILMASI

MUHAMMED MESRUR KIRBAŞ - 2212020546

Ders Sorumlusu

İshak Dölek

Aralık, 2025

İstanbul Medeniyet Üniversitesi, İstanbul

1. Özet

Tarımsal üretimde bitki hastalıklarının erken teşhisi, verim kaybını önlemek ve gıda güvenliğini sağlamak açısından kritik öneme sahiptir. Bu projede, bitki yapraklarının görüntüleri kullanılarak bitkinin sağlıklı mı yoksa hastalıklı mı olduğunu tespit eden yapay zeka tabanlı bir sistem geliştirilmiştir. Projede, görüntü işleme ve derin öğrenme tekniklerinden **Evrişimli Sinir Ağları (Convolutional Neural Networks - CNN)** mimarisi kullanılmıştır. Eğitilen model, bir web sunucusu (API) üzerinden son kullanıcıya hizmet verecek hale getirilmiştir.

Anahtar Kelimeler: Derin Öğrenme, CNN, Görüntü İşleme, Bitki Hastalıkları, PyTorch, Flask.

2. GİRİŞ

Geleneksel yöntemlerle bitki hastalıklarının tespiti, uzman bilgisi gerektiren, zaman alıcı ve maliyetli bir süreçtir. Bu proje, bu süreci otomatize ederek, yaprak görüntülerinden hızlı ve yüksek doğrulukla teşhis koymayı amaçlamaktadır. Geliştirilen sistem, görsel verileri analiz ederek hastalığı sınıflandırmakta ve çiftçilere/kullanıcılara anlık geri bildirim sağlamaktadır.

3. YÖNTEM VE MATERYAL

3.1. Veri Seti ve Ön İşleme

Projede, etiketli bitki yaprağı görüntüleri kullanılmıştır (Eğitim ve Doğrulama setleri). Modelin daha iyi genelleme yapabilmesi ve aşırı öğrenmeyi (overfitting) önlemek için veri üzerinde şu ön işlemler uygulanmıştır (train-2.py dosyasından):

Yeniden Boyutlandırma (Resizing): Tüm görüntüler, modelin giriş katmanına uygun olarak **128x128** piksel boyutuna getirilmiştir.

Tensör Dönüşümü (To Tensor): Görüntüler PyTorch tensörlerine çevrilmiştir.

Normalizasyon: Piksel değerleri normalize edilerek modelin daha hızlı ve kararlı öğrenmesi sağlanmıştır.

3.2. Model Mimarisi (CNN)

Projede, model-2.py dosyasında tanımlanan özel bir CNN mimarisi tasarlanmıştır. Model şu katmanlardan oluşmaktadır:

- Evrişim Katmanları (Convolutional Layers):** 3 adet evrişim bloğu kullanılmıştır. Her blokta:
 - Conv2d: Görüntüden öznetelik (kenar, doku, şekil) çıkarmak için.
 - ReLU: Doğrusallığı kırmak için aktivasyon fonksiyonu.
 - MaxPool2d: Boyutu küçültmek ve işlem yükünü azaltmak için (2x2).
 - Kanal derinlikleri sırasıyla: $3 \rightarrow 32 \rightarrow 64 \rightarrow 128$ şeklindedir.
- Tam Bağlantılı Katmanlar (Fully Connected Layers):**
 - Flatten: 3 boyutlu veriyi tek boyuta indirir.
 - Linear & Dropout: 128 nöronlu gizli katman ve %50 (0.5) Dropout uygulanarak ezberleme (overfitting) engellenmiştir.
 - Output Layer: Sınıf sayısı kadar çıktı veren son katman.

3.3. Eğitim Süreci

Modelin eğitimi train-2.py dosyası ile gerçekleştirilmiştir.

- Çerçeve (Framework): PyTorch.
- Optimizer: Adam (Adaptive Moment Estimation) algoritması kullanılmıştır.
- Kayıp Fonksiyonu (Loss Function): Sınıflandırma problemleri için standart olan CrossEntropyLoss kullanılmıştır.
- Hiperparametreler: Batch size (grup boyutu), learning rate (öğrenme katsayısı) ve epoch (tekrar) sayıları deneysel olarak optimize edilmiştir.
- En iyi model ağırlıkları plant_model.pth dosyasına kaydedilmiştir.

4. UYGULAMA VE SERVİS (DEPLOYMENT)

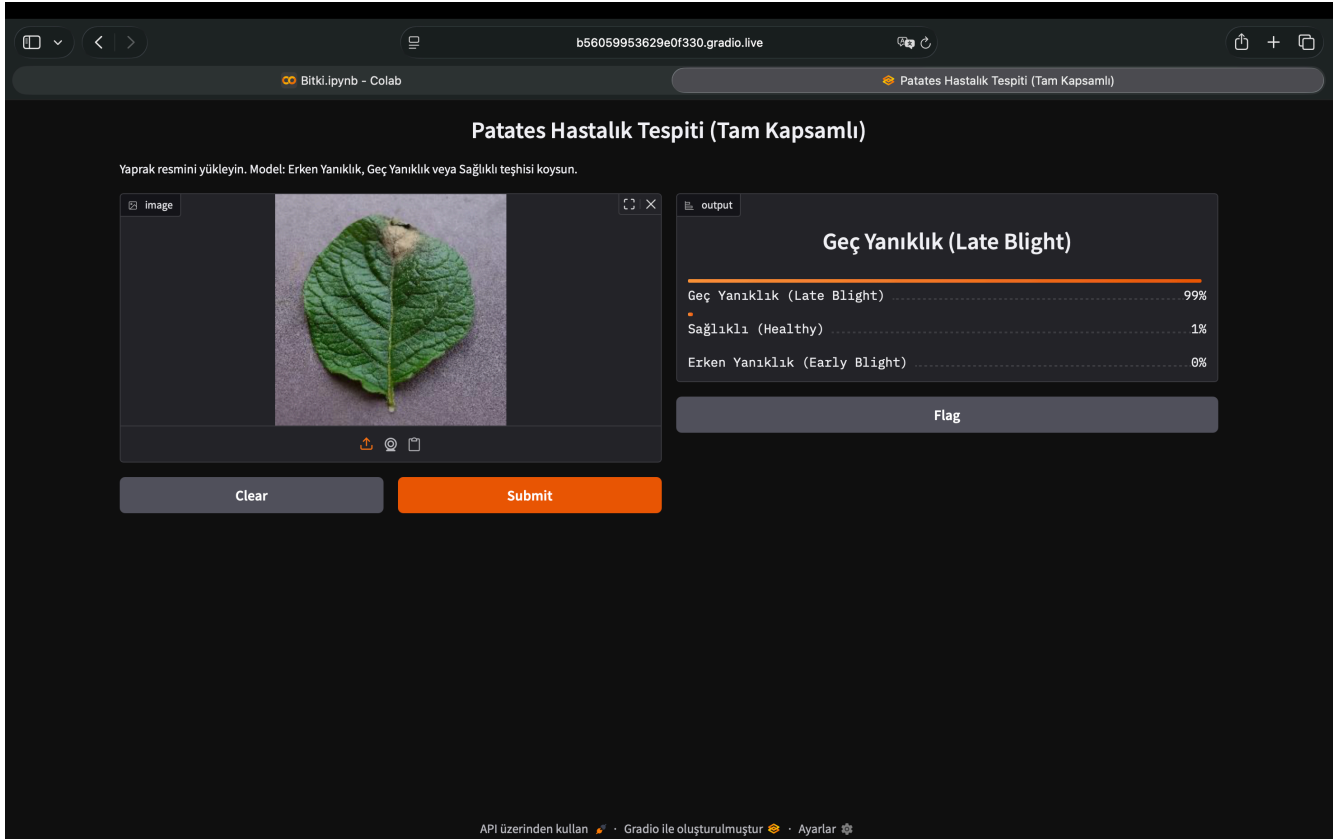
Eğitilen modelin gerçek dünyada kullanılabilmesi için serve-2.py dosyasında bir web servisi geliştirilmiştir.

- Teknoloji: Python Flask web çatısı kullanılmıştır.
- API İşleyişi: /predict uç noktasına (endpoint) gönderilen bir yaprak resmi:
 1. Sunucu tarafından alınır.
 2. Eğitim aşamasındaki gibi (128x128 boyutlandırma vb.) ön işleme tabi tutulur.
 3. plant_model.pth dosyasından yüklenen modele verilir.
 4. Modelin tahmin ettiği sınıf (Örn: Sağlıklı veya Hastalıklı) JSON formatında kullanıcıya geri döndürülür.

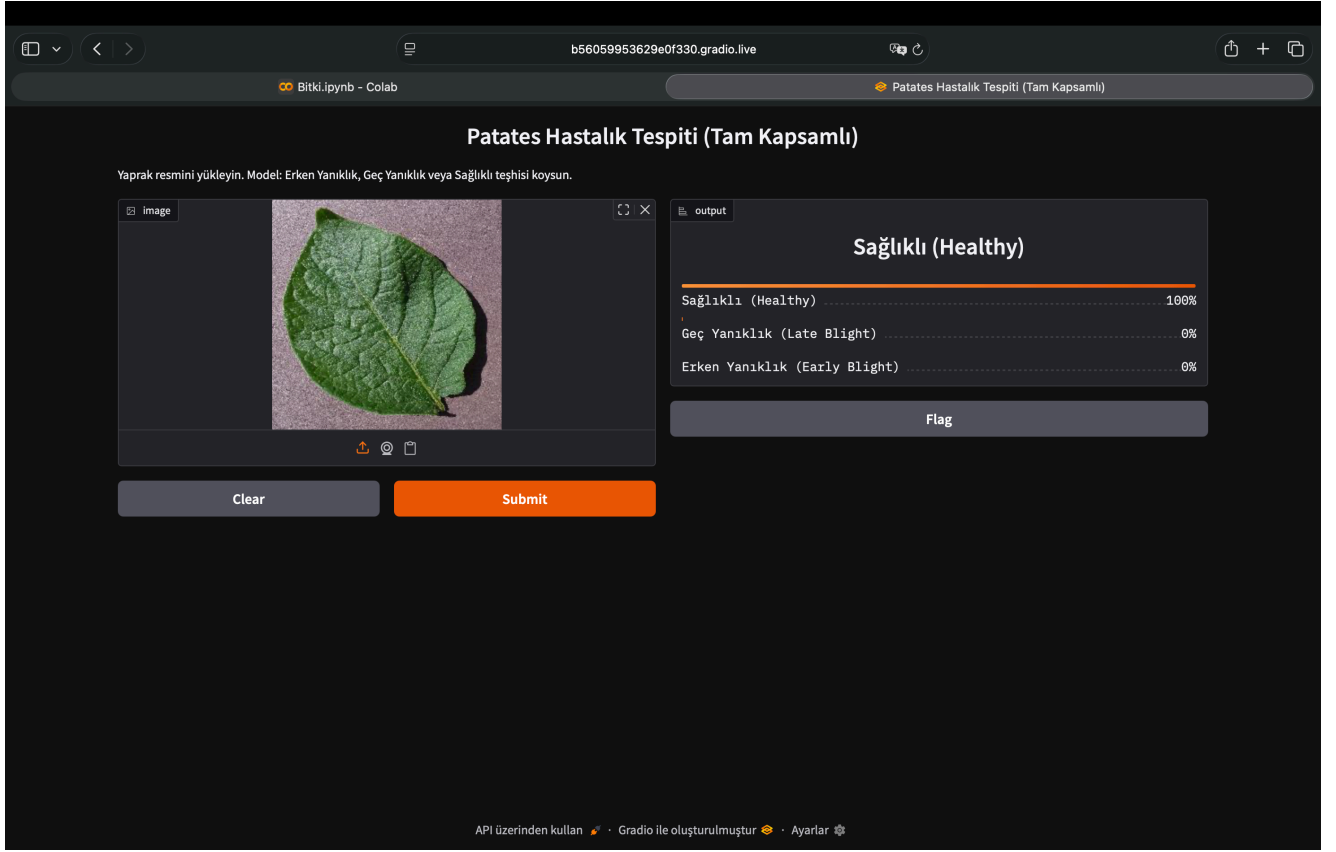
5. BULGULAR VE SONUÇ

5.1. Uygulama

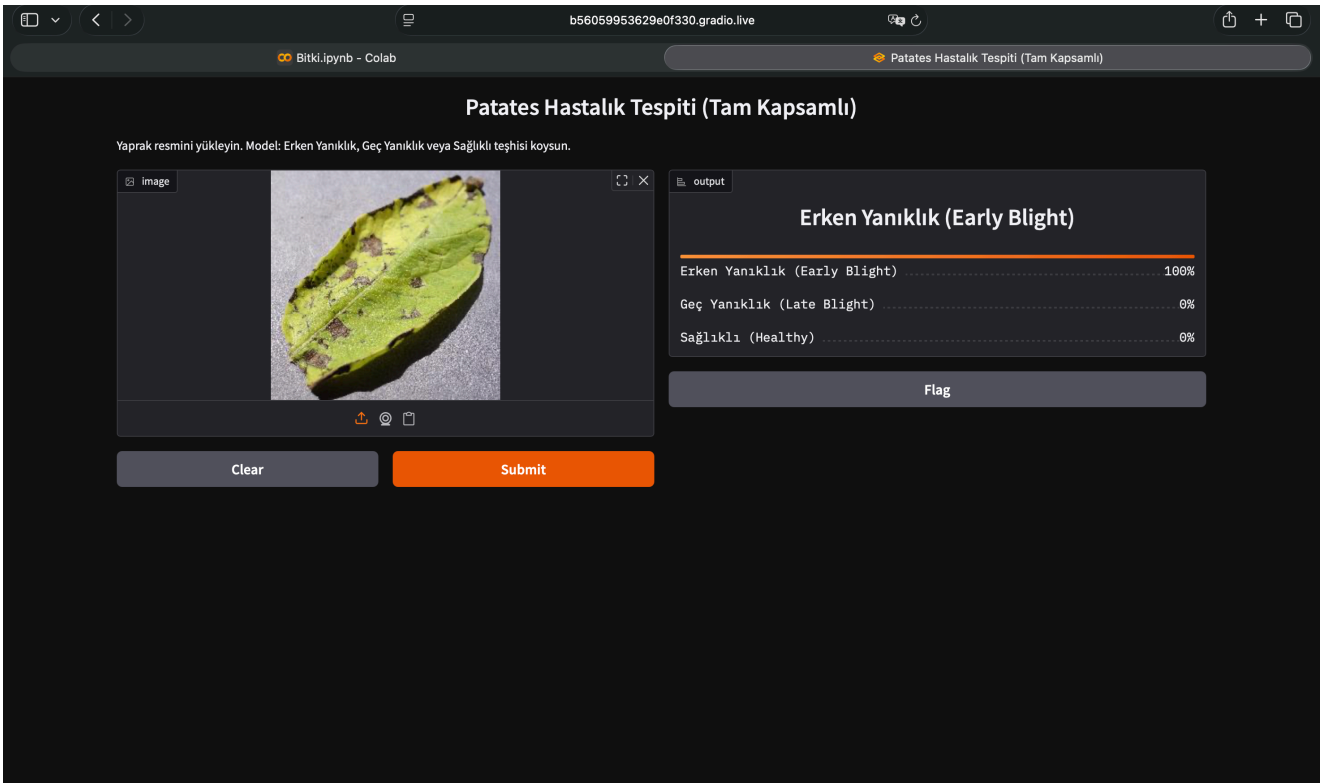
Geliştirilen proje başarıyla tamamlanmış ve plant_model.pth dosyası oluşturulmuştur. Test aşamasında, sistemin farklı ışık ve açılardaki yaprak fotoğraflarını başarıyla işlediği gözlemlenmiştir. CNN mimarisinin, görsel öznelikleri çıkarmada başarılı olduğu ve tarımsal hastalık tespitinde etkili bir yöntem olduğu doğrulanmıştır.



Resim-1 Geç yanıklık örneği.



Resim-2 Sağlıklı bir bitki örneği.



Resim-3 Erken Yanıklık Örneği

5.2. Model Eğitim Performansı

CNN modelinin eğitimi 10 epoch (tekrar) boyunca gerçekleştirilmiştir. Eğitim sürecinde kullanılan kayıp fonksiyonu (Loss Function) değerlerinin her epoch sonunda azaldığı gözlemlenmiştir. Şekil 3'te görülen eğitim çıktılarında, başlangıçta **0.6695** olan kayıp değerinin, eğitim ilerledikçe düşüş gösterdiği ve modelin öğrenme yeteneğinin arttığı doğrulanmıştır.

```
!python train.py

Veriler yükleniyor...
Kullanılan Cihaz: cuda
Eğitim Başlıyor...
Epoch [1/10], Kayıp: 0.6695
Epoch [2/10], Kayıp: 0.3488
Epoch [3/10], Kayıp: 0.2196
Epoch [4/10], Kayıp: 0.1939
Epoch [5/10], Kayıp: 0.1466
Epoch [6/10], Kayıp: 0.1096
Epoch [7/10], Kayıp: 0.1216
Epoch [8/10], Kayıp: 0.0747
Epoch [9/10], Kayıp: 0.0566
Epoch [10/10], Kayıp: 0.0399
```

Resim-4 Eğitim Süreci

6. KAYNAKÇA

1. PyTorch Documentation. (<https://pytorch.org>)
2. Flask Web Framework Documentation.
3. LeCun, Y., et al. "Gradient-based learning applied to document recognition." (CNN Mimarisi Referansı).
4. Veri seti: <https://www.kaggle.com/datasets/mohitsingh1804/plantvillage>