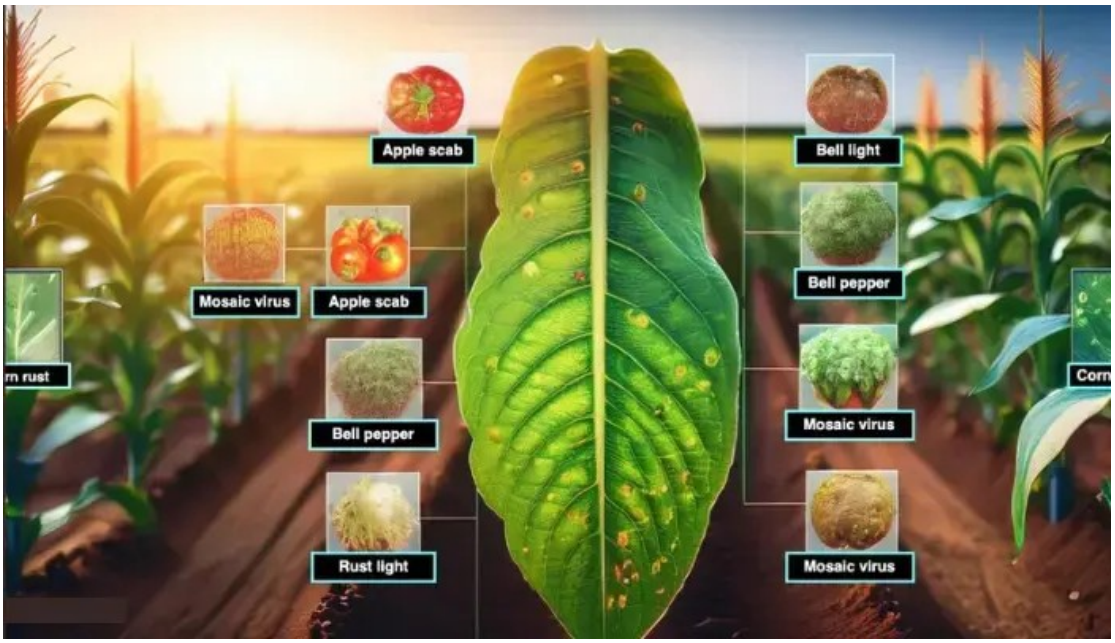


Bitki Hastalıkları Projesi

Genel Bilgilendirme Raporu

Ahmet efe Y-

27 Ocak 2026 Salı

Bitki hastalıkları projesi için geliştirmekte olduğum programın kabaca %65 i tamamlanmış durumdadır; bu projede ***Temelde hastalık tespiti-hasta/saglikli ayrımı*** *olmak üzere iki özellik bulunmakta bu özellikler için iki ayrı model bulunmakta bu modeller temel alt yapısında* ***CNN*** *yani* ***Convensional Neural Network'****leri kullanmaktadır* ***Hasta-Sağlıklı*** *ayrımı yapan model bizlere 0-1 arası bir değer verir bu değer* ***1 e ne kadar yakınsa ve 0 dan ne kadar uzak ise bitki o kadar sağlıklı dır*** *metriğine göre programdaki karar mekanizmaları sayesinde ölçülür ve*

0.5 ten büyük olan çıktılar sağlıklı ***(0.4) + 0*** *ve* ***(0.4) + 0,05*** *arası gelen değerler ise* ***Kısmen sağlıklı*** *yani* ***Half Healthy*** *kategorisinde değerlendirilir*
ve ***Hasta*** *yani* ***Diseased*** *olarak değerlendirilenler ise* ***(0.4) + 0*** *ve* ***(0.4) + 0,05*** *arasındaki değerlerinde altında olan çıktılardır;*

Model çıktılarının program tarafından yorumlanmasındaki metrikler bu şekilde dile getirilebilir ve teknik olarak açıklanabilir şimdi ise henüz eğitilmemiş olan Hastalık sınıflandırma modeline bir göz atalım;

BU modelde bitki hasta ise ve bu hastalık bilinmiyor ise gelen görseli X hastalık sayısınca tahmin eder ve X tane modele gösterilen hastalık türleri ile en uyumlu eşleşen sınıfa 0-1 arası en 1 e yakın değeri verir örneğin ***4*** *sınıflı bir yapı olsa çıktı şu şekilde olur*

[[0.05616~ , 0.03043~, 0.20510~, 0.50331]]

Şeklinde bir çıktı gelir burada aslında sınıflar şu şekildedir

{'X hastalığı': 0 , 'Y hastalığı': 1, 'Z hastalığı': 2, 'Q hastalığı': 3}

yani burada

X hastalığı : 0.05616~
Y hastalığı : 0.03043~
Z hastalığı : 0.20510~
Q hastalığı : 0.50331~

Şeklinde bir karar verilir ve ***Q hastalığı*** *tahminler yani* ***Prediction****'lar arasından en maksimum olanı olarak seçilir yani kısaca karar verilirken çok sınıflı yapılarda model bu şekilde her sınıfa* ***skorlar*** *biçer ve programda bunları metriklere göre analiz eder*

Bu projedeki 0-1 arası sınıflandırma yapan model ile çok sınıflı bir şekilde hastalık tespiti yapan modeli karıştırmayın ikisi farklı modeller ve şimdi ise 0-1 arası sınıflandırma yapan modelin girdi olarak neler aldığına bakalım;

Temel olarak bu model bizden girdi olarak 0-255 arası normalize edilmiş RGB uzay boyutlu görselin 0. axisine batch dimension ***Batch boyutu*** *eklenmiş ve 224x224 olarak yeniden boyutlandırılmış RGB renk uzayına sahip bir görsel ister görseller genel olarak farklı*

boyutlarda ve çoğunlukla BGR2 renk uzaylarında gelirler ayrıca görüntü matrisleri içinde INT yani tam sayılar bulunur 0-256 arası yani 0 dan 255 e kadar rakamlardan oluşan büyük matrislerdir ve modele bunları vermeden önce ilk başta belirttiğim işlermleri uygulamamız gerekir bu işlemleri pythonda istersek ***Tensorflow*** *un kendi fonksiyonları ile halledebilir veya istersekte harici olarak* ***opencv-python*** *ile halledebiliriz bu işleme* ***preprocess*** *yani* ***ön işleme*** *denir ön işleme esnasında modelin isteyeceği formatlara görüntü çevirilir.*
ve bunun bir farklısı ise ***Augomention****'dır bu ise hedef dizindeki klasörlerin hepsini bir sınıf olarak kabul eden ve o sınıfların içinden bir akış ile görselleri çekmek ve bir* ***generator*** *oluşturmaktır bu generatore görselleri depolar akış esnasında istenilen görüntü işleme işlemleri uygulanıp modelin besleme verisini daha gerçek dünya verisine benzetip modeli zorlar ve öğrenmesini & genelleme yapmasına dahada olanak tanır modelimizin eğitimi ise tam olarak aşşağıdaki anlattığım gibi gerçekleşiyor..*

Görüntüler *generator'de depolanmışken modele bu generator verilir ve model ise bu generator'deki veriler ile kendini besler modele verilir ve besleme aşaması gerçekleşmiş olur ve modelin tahmin yapacağında ise*

bu aşama tekrarlanır ve tahmin edeceği görsel modele verilmeden bu aşamalardan geçirilir ve modele o şekilde veriliki model sapıtmasın saçma tahminler yapmasın veri tutarlı olsun. Ancak bu aşamada yani model eğitildikten sonra tahmin yapması gerektiğinde tek bir görseli modele vermek için bir generatore sokmak yerine daha çok ***skimage opencv*** *ve* ***tensorflow*** *un görüntü fonksiyonları kullanılarak* ***Augomention*** *da hangi preprocess işlemleri uygulandı ise aynen yine o işlemler uygulanır*

Temelde bu programda ***ResNet50*** *modeli kullanıldı her iki modelde aslında* ***ResNet*** *yani* ***Residual Network 50*** *olarak programda mevcuttur burada bu modeli kullanma sebebim ise şu; normalde Sequential gibi modeller sıralı modellerdir ve önceden eğitilmemiş 0 yani hazır modellerdir gerçek dünya problemlerinde Sequential genelde pek tercih edilmez Sequential daha çok kısa vadeli modellerde ve bazı benzer küçük datasetlerde güzel sonuçlar veren* ***Tensorflow.Keras*** *ın güzel hazır modellerinden birisidir ancak* ***ResNet*** *önceden yaklaşık* ***2.3 MİLYON*** *görsel ile beslenmiştir bu görseller* ***Araba Bisiklet çiçek bardak vb. vb.*** *bir cok görselden oluşur buda modeli çok ağır ve büyük bir model yapıyor ve buda şunu getiriyor* ***Sizin modeli eğiteceğiniz verileri model zaten önceden gördü***

*ancak sizin verdiğiniz kadar detaylı görmedi sizde burada modele **Fine Tuning** yani **ince ayar** yaparak **layerları** donduruyorsunuz ve trainableını false yapıyorsunuz buda tek seferde 50 layeri birden(50 layer içine weight + b vb değerler dahil edilmedi) eğitmiyoruz sadece sondan n tane layeri trainable yapıp onların üzerine eğitimi gerçekleştiriyoruz ve modeli kompile ediyoruz kompile edilmiş modeli sonraısnda tekrardan eğitiyoruz ve bunu comp - fit -comp - fit -comp -fit şeklinde tekrarlıyoruz ve buda modelin ilk başta kabaca öğrenemsini sonrasında ise detaylıca öğrenebilmesini sağlayan bir mimariyi doğuruyor kısacası ResNet i kullanmamın temel mutlak sebebi budur daha fazla anlatmaya pekte gerek yok zaten bu model ile anlatılabilecek çok şey var*

Bu aşamalarıda temel olarak anlattığıma göre artık programa geçebilirim;

Programda temel olarak model ile alakalı bir özgürlük mevcut; bahsettiğim şey şu; Yukarıda bahsettiğim 2 tane model ile çalışmak zorunda değil program eğer programın girdisini destekleyen türde bir model bulur iseniz o modeli indirip programa attığınızda program modeli tanıyor ve model ile senkronize çalışabiliyor bu özellik daha geliştirme aşamasındadır bilindiği üzere

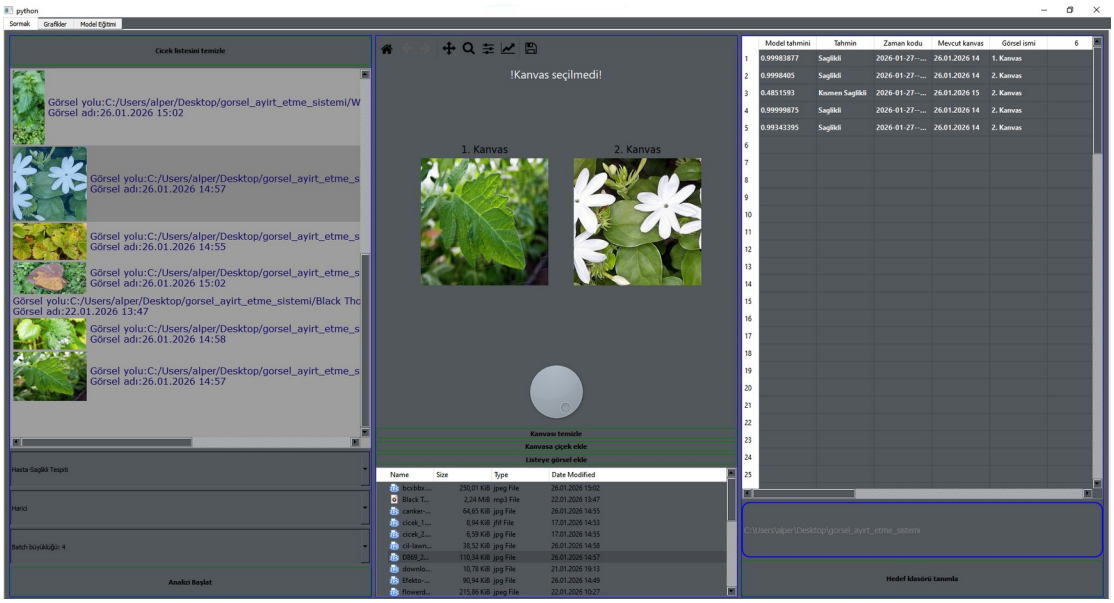
her modelin farklı girdi boyutları farklı girdi metrikleri ve farklı farklı istekleri bağımlılıkları mevcuttur şu anlık sadece programın girdisine göre tahmin yapabilen modeller program tarafından destekleniyor ancak bunları kullanmak zorunda değilsiniz bu sadece bir alternatif zaten temelde sağlam iki tane model mevcut ancak modellerde hala geliştirme aşamasında;

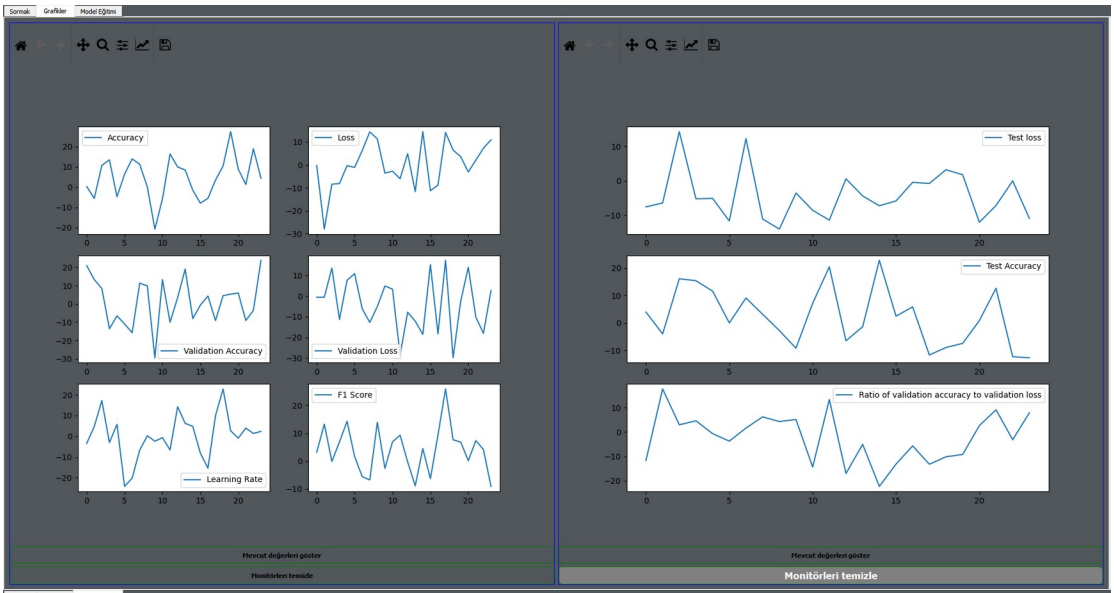
*Ve şimdi ise programdaki asıl özelliğe değinelim; Programda **Transfer Learning** mevcut yani programı kendiniz eğitebiliyorsunuz ve yaptığı hataları tekrarlamasını azaltabiliyorsunuz Şu anda bu özellik geliştirme aşamasında olsada programın bünyesi bunu kaldıracak mimariye sahip yani bu özellik eklenecek bir özellik ve teknik aksaklıklar çıkmaz ise programa kesin olarak eklenecek özelliklerden birisi bariz olarak budur; Progamda seçtiğiniz modelin **Summary** yani katman bilgilerini parametre sayılarını VB. bilgilerine erişmenizi sağlayan bir sistemde şu anda mevcut*

Ayrıca programdaki bir özellik ise model oluşturma sekmesi; evet bu sekme daha programda teknik olarak mevcut değil ancak en kısa zamanda eklencek özelliklerden birisi ve teknik mimari bu özelliğe uygun tasarlandı

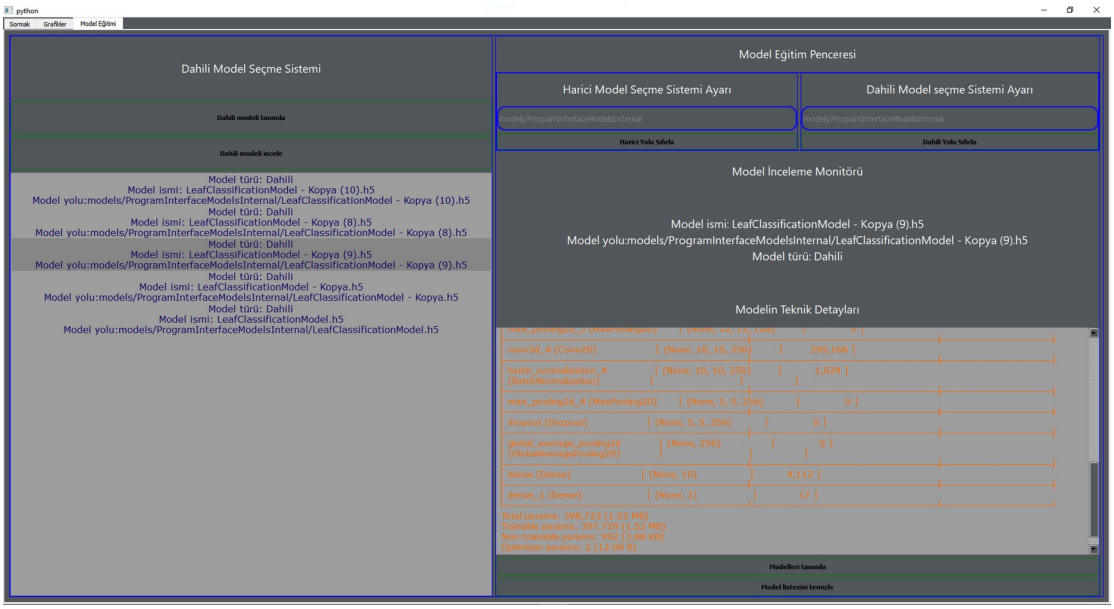
Ayrıca programdaki log sistemi sayesinde programı yazarken hazırladığım modeller dahil sizin eğiteceğiniz modellerinde tüm eğitim bilgilerinin loglandığı ve bu logların bir sistem aracılıgı ile parse edilip sayısal formata dönüştükten sonra grafiklerinin çıkarlıldığı bir sistemde programda Mevcuttur. Yani programda bir log + plotting sistemi mevcut olarak bulunmakta

Şimdi ise artık programın model ile olan bağlantıları ve özelliklerini anlatmayı bir kenara bırakıp programın kendisini sunayım;




Sormak
Grafikler
Model Eğitimi
Accuracy
Loss
Validation Accuracy
Validation Loss
F1 Score
Learning Rate
Mevcut değerleri göster
Monitörleri temizle
Test loss
Test Accuracy
Ratio of validation accuracy to validation loss
Mevcut değerleri göster
Monitörleri temizle

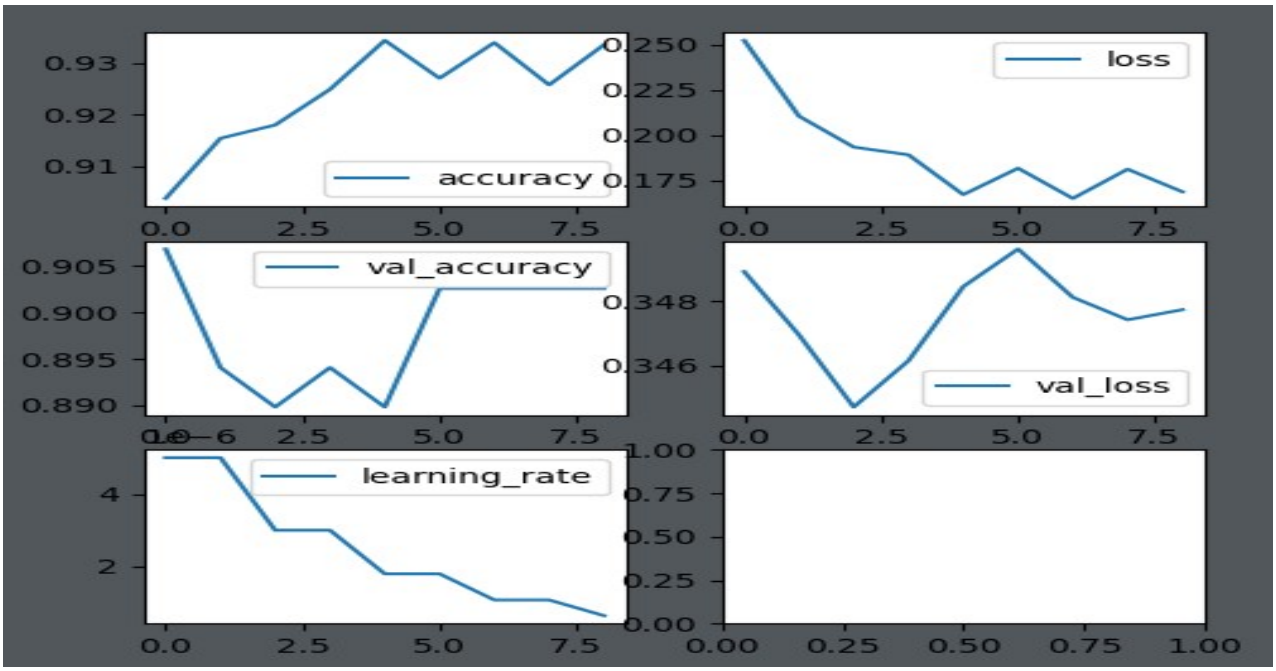

Sormak
Grafikler
Model Eğitimi
Dahili Model Seçme Sistemi
Dahili modeli tanımla
Dahili modeli incele
Model Eğitim Penceresi
Harici Model Seçme Sistemi Ayarı
Dahili Model seçme Sistemi Ayarı
models/ProgramInterfaceModelsExternal
models/ProgramInterfaceModelsInternal
Harici Yolu Sıfırla
Dahili Yolu Sıfırla
Model İnceleme Monitörü
Model ismi: LeafClassificationModel - Kopya (9).h5
Model yolu:models/ProgramInterfaceModelsInternal/LeafClassificationModel - Kopya (9).h5
Model türü: Dahili
Modelin Teknik Detayları
Model yolu:models/ProgramInterfaceModelsExternal/model.h5
Model ismi: model.h5
Model türü: Harici
Model: "sequential"
Layer (type)
Output Shape
Param #
conv2d (Conv2D)
(None, 222, 222, 16)
448
batch_normalization (BatchNormalization)
(None, 222, 222, 16)
64
max_pooling2d (MaxPooling2D)
(None, 111, 111, 16)
0
conv2d_1 (Conv2D)
(None, 109, 109, 32)
4,640
batch_normalization_1 (BatchNormalization)
(None, 109, 109, 32)
128
Modelleri tanımla
Model listesini temizle
Harici Model Seçme Sistemi
Harici modeli tanımla
Harici modeli incele



Görmüş olduğunuz üzere programda olduğunu belirttiğim özellikler bu şekilde gözüküyor ayrıca eklemeyi unuttuğum şey ise şu;

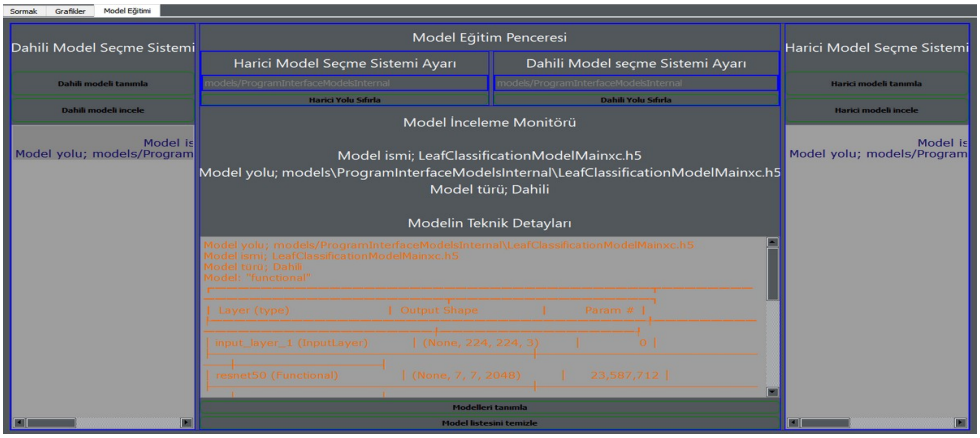
Modele tahmin yaptırmadan önce ***Batch büyüklüğü****,* ***Model türü****,* ***Model*** *gibi etkenleri seçip o yönde bir çıktı alabiliyorsunuz*

Şimdi ise modelin mevcut öğrenme grafiklerine gelelim

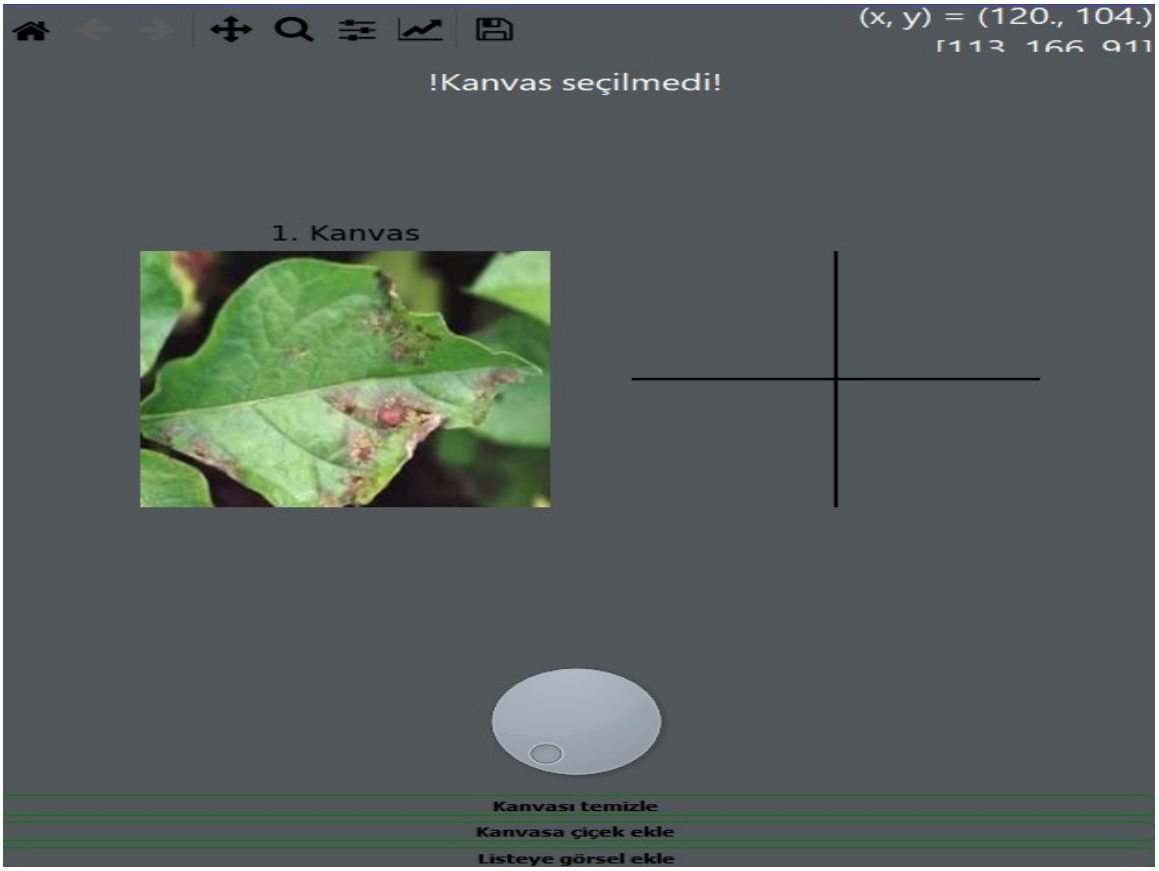


*Grafiklerdede görmüş olduğunuz üzere **accuracy ile loss** birbirine ters orantılı ancak göze çarpan bir etken var **Validation Accuracy** ve **Validation Loss** grafikleri bu grafikler normalden farklı olarak bir birine yakın değerlerde başlamış(ki bu iç açıcı bir durum değil) validation accuracy bu grafikte başlarda düşen tarafta ancak accuracy düşerken validation loss ta düşüyor buda grafiğe tekrar bakıldığında iyi bir etken olarak göze çarpıyor; işte tamda burada devreye learning rate giriyor burada learning rate 2. epoks civarlarnda devreye girmiş ve 2. epokstan sonrasında accuracy ufak bir zıplama gerçekleştirmiş aynı şekilde loss ta düşüşte kalmış 4. epoksa yaklaşılırken learning rate tekrardan düşüşe geçmiş ve bu seferde validation*

accuracy artık yükseliş göstermeye başlamış ve validation accuracy yükselirken validation loss ta onunla beraber yükselmeye başlamışki tamda 5. epoksta learning rate tekrar düşmüş ve loss dalgalı bir şekilde düşüş göstermiş accuracy ve loss dengesiz gibi gözüksede teknik olarak şu anlık düşük bir overfit riski taşıyor model bu test verilerine bakıldığında genellemesi orta seviyelerde duruyor model son epoksta EarlyStopping callback'i tarafından durdurulmuştur daha kısa kesmek gerekirse ***val loss maksimum 0.348*** *göstermiş iken* ***val accuracy maksimum olarak 0.900*** *civarları göstermiş aradaki fark gözle görülür şekilde belirgin kısacası model şu anki hali ile* ***mükemmel seviyede*** *değil ancak* ***karar verme aşamalarında*** *%70* ***doğruluk*** *ile çalışıyor örnek olarak bir test yapalım*



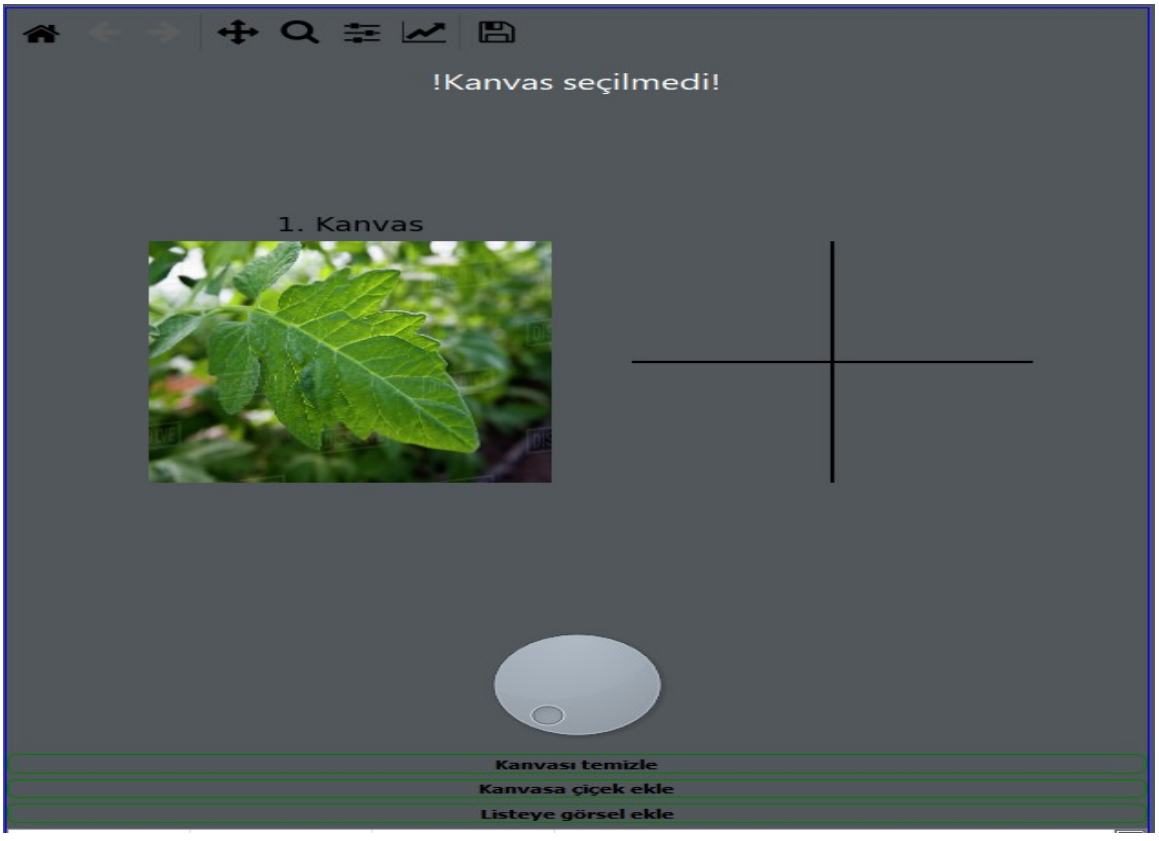
modelimizi tanımladık şimdi ise görselimizi tanımlayalım



*görselimizide tanımladık şimdi ise **analizi başlat** butonuna tıklayalım ve modelin yapacağı tahmine bakalım*

	Model tahmini	Tahmin	Zaman kodu	Mevcut kanvas	Görsel ismi	6
1	0.00031713338	Bitki hasta	2026-02-06--...	indir (1).jfif	1. Kanvas	
2						
3						
4						
5						
6						
7						

*Modelin belirtilen kanvasta görmüş olduğunuz görsele verdiği **sağlıklılık olasılığı** veya diğer bir değiş ile **sağlıklılık skoru** yaklaşık olarak **0.000317** civarında buda bitkinin hasta olduğunu anlayabildiğini gösteriyor şimdi ise sağlıklı bir çiçek ile testimizi tekrarlayalım*



Şimdi ise görselimiz ekranda göründüğü gibidir

ve modelimizin bu görsele verdiği tahmin ise aynen şu şekildedir

	Model tahmini	Tahmin	Zaman kodu	Mevcut kanvas	Görsel ismi	6
1	0.00031713338	Bitki hasta	2026-02-06--...	indir (1).jfif	1. Kanvas	
2	0.5132431	Saglikli	2026-02-06--...	26.01.2026 ...	1. Kanvas	
3						
4						
5						
6						
7						
8						

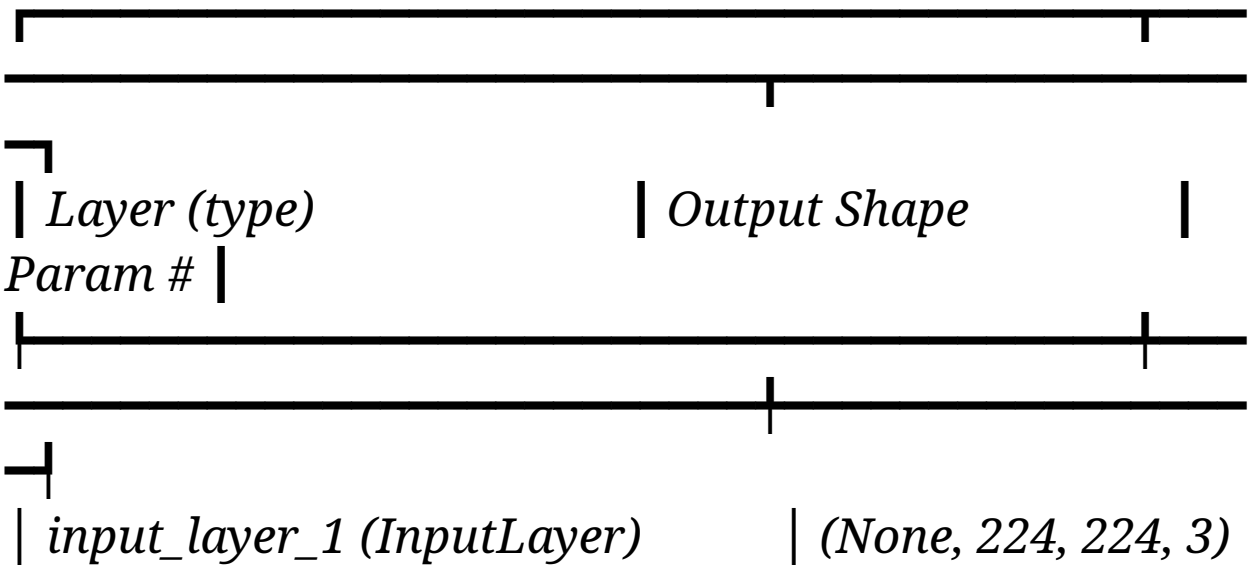
*Yani 1. kanvastaki görselimize modelimiz **0.5132** civarında bir skor biçti ve sağlıklı etiketini belirtti ancak burada bir husus var*

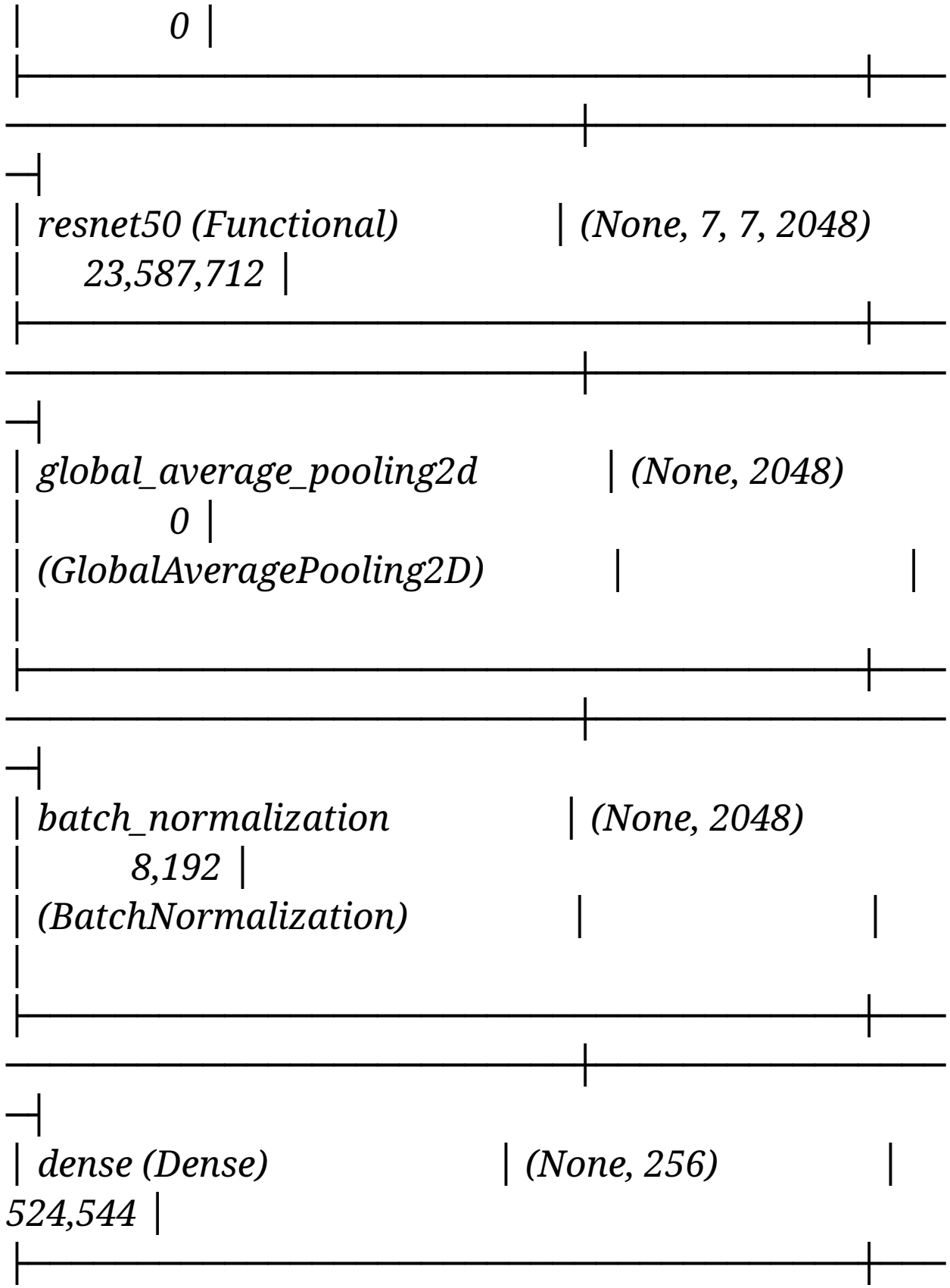
*Fark etmiş olduğunuz gibi model **0.513** civarı bir değer verdi aslında model full sağlıklı olan bir çiçeğe kaba haesap en az **0.780** civarında bir skor vermeli işte burada o az önce yorumladığımız grafikler devreye giriyor*

O grafiklerdede görüldüğü üzere model ortalama bir başarı gösteriyor ve bunuda testlerimiz ile doğrulamış olduk

not: teknik yorumlardan ek olarak programda model ismi ve seçili kanvasın column etiketlerinde bir kayma olmuş o yüzden seçili kanvasta görsel isimleri görsel isimlerinde ise seçili kanvas yazıyor yani teknik bir aksaklık değil sadece küçük bir dikkatsizlik .)

Model yolu; models/ProgramInterfaceModelsInternal\LeafClassificationModelMainxc.h5
Model ismi; LeafClassificationModelMainxc.h5
Model türü; Dahili
Model: "functional"





```
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃ dropout (Dropout)               ┃ (None, 256)            ┃             0 ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ dense_1 (Dense)                 │ (None, 1)              │           257 │
└─────────────────────────────────┴────────────────────────┴───────────────┘
```

Total params: 24,120,707 (92.01 MB)
Trainable params: 16,360,705 (62.41 MB)
Non-trainable params: 7,760,000 (29.60 MB)
Optimizer params: 2 (12.00 B)

Hastalık Sınıflandırma Modeli(CNN)

ResNet50

BU modelin teknik olarak tespit edebildiği hastalıklar

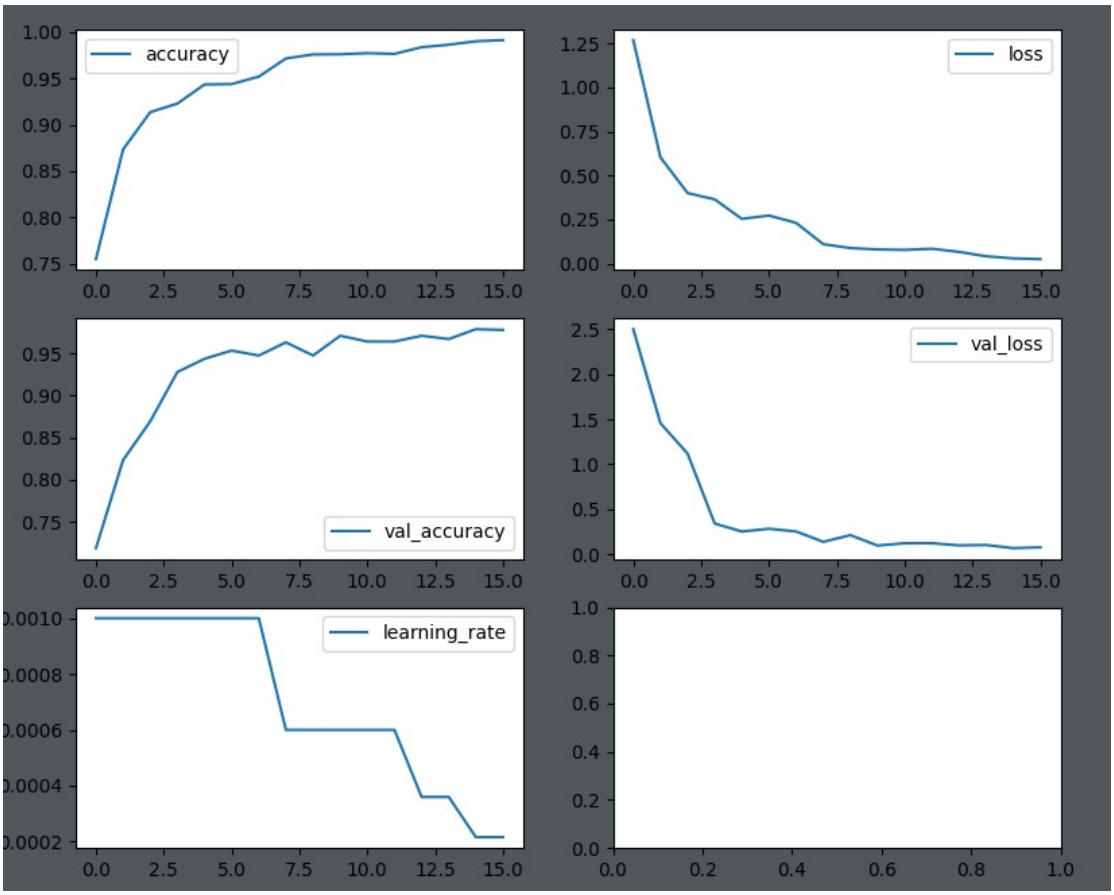
- *Biber bakteriyel hastalığı*
- *Patates erken yanıklığı*
- *Patates geç yanıklığı*
- *Domates bakteriyel leke hastalığı*
- *Domates erken yanıklığı*
- *Domates geç yanıklığı*
- *Domates yaprak deformasyonu*
- *Domates mozaik virüsü*
- *Domates septorya yaprak lekesi*
- *Domates kırmızı örümcek zararlısı*
- *Domates hedef lekesi*

Model yukarıda belirtmiş olduğum hastalıkları teknik olarak tespit edebiliyor 1. dökümanda belirttiğim gibi burada ***Softmax*** *aktivasyonu kullanıldı yani bize modelin yaptığı tahminden bir vektör geliyor ve sınıf sayısı kadar uzunlukta değerler geliyor maksimum argümanın anahtarı(dict veri yapısında) bize hangi sınıfın en çok olasılık ile model tarafından seçildiğini belirtir yani daha temel bir anlatış ile;* ***Bize gelen vektörde en büyük olasılığın denk geldiği sınıf ismi bize modelin görseli analiz ederek vermiş olduğu tahmini sınıfı verir ve buda modelin bize görsele dayalı yaptığı olasılık tahminini temsil eder...***
şeklinde temel olarakta açıkalanabilir

Şimdi ise Hangi modeli kullandığıma gelelim; Bunun için kullandığım model ***ResNet50*** *Modeli olarak geçen Görüntüler üzerinde temel olarak iyi sonuçlar veren bir CNN modelidir*

Bu modeli biraz daha anlatmam gerekirse; BU model önceden eğitilmş ***Pretrained*** *modeldir ve biz burada bu modelin katmanlarını yani* ***Layerlarını*** *dondurarak sondan 30 tane layer'ı tekrar eğittik yani bir sürü layeri tekrar tekrar eğitmek yerine sondan 30 tane layer'ı eğittik Buna* ***Transfer Learning*** *denir yani kısaca önceden eğitilmiş bir modeli kendi verilerin ile tekrardan eğitmektir*

Model *eğitilirken 16 epoks ve 16 batch büyüklüğü ile eğitilmiştir model toplamda net olarak* ***11 sınıflı 1024 görselli*** *Test verisi ve* ***11 sınıflı ve 4.843 görselli*** *bir Training yani* ***Eğitim*** *verisi ile eğitilmiştir Modelin eğitim grafikleri ise aşağıda verildiği gibidir*



Grafiklerdede görüldüğü üzere accuracy ve loss bir birine ters orantılı gitmişler loss ilk başlarda 1.25 gibi %125 küsür civarı değerlerde başlasada orantılı bir düşüş göstermiş validation acc ve validation loss grafikleri ise yine aynı şekilde validation loss başlarda 2.5 yani %250 civarında bir kayıp gösterirken gittikçe azalmış ayrıca validation acc burada yine artan tarafta yani grafiklere kabaca baktığımızda model şu an için mükemmel durmasada teknik olarak tahmin yapabilecek bir seviye Ayrıca söylemeden geçmemem gereken bir husus daha var Learning rate 7. epokstan sonra düşmeye başlamış 7. ve 15.0 arası büyük bir düşüş var buda şu demek; Callbacklerin izlediği değer olan val_loss learning_rate desteği ile kontrollü şekilde düşürülmüş

Şu anlık bu model test aşamasında olduğu için şu anlık belgeyi burada bitiriyorum ancak belge güncelleme alacaktır okuduğunuz için teşekkür ederim