**Yapay Zekaya Giriş — Derin Öğrenme Projesi Raporu**

Beyin MRI Görüntülerinden Tümör Tespiti (Binary Sınıflandırma)

|  |  |
| --- | --- |
| Öğrenci Ad Soyad: | Fatih AYIBASAN |
| Öğrenci No: | 220707070 |
| Bölüm / Sınıf: | Bilgisayar Mühendisliği 4. sınıf |
| Ders: | Yapay Zekaya Giriş |
| Tarih: | 01.12.2025 |

# 1. Amaç ve Kapsam

Bu projenin amacı, Yapay Zekaya Giriş dersi kapsamında verilen veri setleri ile derin öğrenme/transfer öğrenme yöntemleri kullanarak tahmin (sınıflandırma) yapmak; en az 10 farklı model denemek; yeni ve hibrit modeller geliştirmek; sonuçları grafikler ve metriklerle raporlamaktır.

Rapor; veri setinin kısa tanımı ve EDA bulguları, kullanılan yöntemlerin açıklaması, her model için performans grafikleri ve metrikler, tüm modelleri tek tabloda karşılaştırma, en iyi modelin kaydedilmesi ve örnek tahmin akışını içerir.

# 2. Veri Setleri

## 2.1. Eğitim Veri Seti (Ana Kaynak)

Mendeley Data: Brain Tumor MRI Dataset (Tumor / No Tumor).

Bağlantı: <https://data.mendeley.com/datasets/c9rt8d6zrf/1>

## 2.2. Test Veri Setleri (Dış Kaynak / External Test)

1) Kaggle: Brain MRI Images for Brain Tumor Detection

Bağlantı: <https://www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection>

2) Kaggle: Brain Tumor Detection

Bağlantı: <https://www.kaggle.com/datasets/ahmedhamada0/brain-tumor-detection>

## 2.3. Veri Setinin Çok Kısa Tanımı

Bu çalışma, beyin MRI görüntülerinden ikili sınıflandırma yapmayı hedefler. Görüntüler iki etikete ayrılmıştır: Tumor ve No Tumor. Amaç, bir MRI görüntüsünde tümör olup olmadığını otomatik olarak tahmin etmektir.

## 2.4. Split ve Veri Sızıntısı (Leakage) Problemi

Orijinal veri setinde eğitim ve test ayrımı bulunsa da, EDA sırasında aynı kişiye ait birden fazla MRI görüntüsünün farklı setlere (train/test) dağılabildiği gözlemlenmiştir. Bu durum, modelin eğitimde gördüğü kişiye çok benzeyen örneklerle testte değerlendirilmesine neden olur ve metrikleri yapay şekilde yükseltir (subject leakage).

Bu nedenle, Mendeley kaynağındaki tüm görüntüler eğitim havuzu olarak kullanılmış; test değerlendirmesi, farklı kaynaklardan alınan iki bağımsız veri seti üzerinde yapılmıştır. (Mete hocanın haberi var.) Bu yaklaşım, genelleme performansını daha güvenilir ölçmeyi amaçlar.

Not: Çalışma sonunda yaklaşık ~10.000+ eğitim görüntüsü ve ~3.500 test görüntüsü olacak şekilde veri havuzları oluşturulmuştur (nihai sayılar EDA bölümünde raporlandı).

# 3. Keşifsel Veri Analizi (EDA)

Bu projede EDA’nın kritik çıktısı, hasta/kişi bazlı tekrarların train-test ayrımını bozabildiği tespit edildi.

## 3.1. Veri Envanteri

* Train havuzu: 10.085
* Train sınıf başına dağılım: Tumor: 5.019, No Tumor: 5.066
* Test havuzu: 3.253
* Test sınıf başına dağılım: Tumor: 1.655, No Tumor: 1.598

## 3.2. Görüntü Boyutları ve Format

Görüntülerin piksel boyutları incelendiğinde, ana eğitim kaynağında örneklerin çoğunlukla 256×256 veya 512×512 boyutlarında olduğu gözlemlenmiştir. Model eğitiminde sabit giriş boyutu gereksinimi nedeniyle tüm görüntüler belirlenen input\_size değerine yeniden ölçeklendirilmiştir.

## 3.3. Kalite Kontrolleri

Bozuk/okunamayan, aşırı karanlık/parlak görseller bulunamamıştır.

## 3.4. Yakın Kopya / Subject Leakage Analizi

Aynı kişiye ait birden fazla çekimin farklı setlere dağılması, modelin gerçek genelleme yerine kişiye özgü ipuçlarını öğrenmesine yol açabiliceği tespit edilmiştir. Bu nedenle subject leakage riski EDA kapsamında analiz edilmiş ve dış kaynak test ile değerlendirme tasarımına geçilmiştir.

# 4. Yöntem

## 4.1. Ön İşleme (Preprocessing)

Bu çalışmada görüntüler klasör tabanlı olarak yüklenmiştir. Veri dizin yapısı Dataset/Training ve Dataset/Testing alt klasörleri olacak şekilde düzenlenmiş; sınıf isimleri no\_tumor ve tumor olarak tanımlanmıştır. Etiket sırası sabitlenmiştir: **0 = no\_tumor**, **1 = tumor**. Eğitim–doğrulama ayrımı, Training klasörü içinde **stratified** (sınıf oranlarını koruyan) biçimde yapılmıştır. Bu amaçla StratifiedShuffleSplit kullanılarak verinin **%10’u doğrulama (val)** olarak ayrılmıştır (val\_ratio = 0.1, random\_state = 42). Test verisi ise Testing klasöründen ayrı bir veri kümesi olarak değerlendirilmiştir.

Görüntü boyutları veri kümesinde değişken olduğu için, modele giriş boyutu sabitlenmiştir. Bu projede giriş boyutu **256×256** olarak seçilmiş ve tüm görüntüler bu boyuta yeniden ölçeklendirilmiştir. Yeniden boyutlandırma öncesinde görüntülerin en–boy oranını bozup gereksiz kırpmaya yol açmamak için **SquarePad** uygulanmıştır: görüntü kısa kenardan siyah (0) piksellerle doldurularak kare hale getirilir, ardından Resize(256,256) yapılır.

Görüntüler tensöre çevrildikten sonra normalizasyon uygulanmıştır. Normalizasyon parametreleri (mean/std) sabit elle verilmemiş; kullanılan backbone’un (timm model konfigürasyonu) varsayılan değerlerinden otomatik çekilmiştir. Seçilen özgün model custom\_msaf\_effb0 olduğu için mean/std değerleri **EfficientNet-B0** konfigürasyonundan alınmıştır (resolve\_data\_config). Bu sayede önceden eğitilmiş (pretrained) modellerin beklediği ölçeklendirme ile uyum sağlanmıştır.

Tekrar üretilebilirlik için rastgelelik kontrol altına alınmıştır: Python/NumPy/PyTorch seed ayarlanmış, DataLoader işçileri için seed\_worker kullanılmıştır. Eğitim GPU varsa CUDA üzerinde yürütülmüş, yoksa CPU kullanılmıştır.

## 4.2. Veri Artırma (Augmentation)

Bu çalışmada veri artırma yalnızca **eğitim (train)** aşamasında uygulanmış; doğrulama (val) ve test aşamalarında artırma kapatılmıştır. Artırma şiddeti tek bir parametre ile kontrol edilmiştir: aug\_strength. Bu parametre 0.0 olduğunda eğitim dönüşümleri doğrulama dönüşümleri ile aynı hale gelir ve augmentasyon **tamamen devre dışı** kalır. aug\_strength arttıkça augmentasyonun olasılıkları ve büyüklükleri kademeli olarak artacak şekilde tasarlanmıştır. Şiddet ölçeklemesi için parametre 0–1 aralığına taşınmış ve tüm artırmalar bu ölçek ile çarpılmıştır (şiddet arttıkça flip/rotasyon/perspektif vb. dönüşümlerin hem uygulanma ihtimali hem de etkisi artar).

Uygulanan artırmalar, tensöre çevirmeden önce (PIL seviyesinde) aşağıdaki dönüşümlerden oluşur:

* RandomHorizontalFlip (yatay çevirme)
* RandomVerticalFlip (dikey çevirme)
* RandomRotation (dönme)
* RandomAffine (çeviri + ölçekleme)
* RandomPerspective (perspektif bozulması)
* ColorJitter (parlaklık/kontrast/doygunluk/hue)
* GaussianBlur (bulanıklaştırma)

Ayrıca tensör seviyesinde, düzenlileştirme (regularization) amaçlı **RandomErasing** uygulanmıştır. Bu yöntem, görüntü üzerinde rastgele küçük bölgeleri silerek modelin tek bir lokal bölgeye aşırı bağımlı olmasını azaltmayı hedefler.

**Augmentation şiddeti için yapılan deneyler:** Bu projede aug\_strength değeri sistematik şekilde denenmiştir. Önce **0.0–0.3** aralığında denemeler yapılmış ve **0.3 değerinde yaklaşık %90 doğruluk** elde edilmiştir. Daha sonra **0.3–0.7** aralığında her değer tek tek denenerek eğitim tekrarlanmış; bu aralıktaki tüm denemelerde doğruluk **yaklaşık %90 seviyesinde kalmış**, yani doğrulukta anlamlı bir artış gözlenmemiştir. Buna karşılık daha düşük şiddetlerde performans düşmüştür: aug\_strength = 0.2 denendiğinde doğruluk **%89** seviyesine, aug\_strength = 0.1 denendiğinde ise **%85** seviyesine gerilemiştir. Bu sonuç, augmentasyonun belirli bir minimum şiddetin altında modele yeterli genelleme katkısı yapmadığını; 0.3 ve üzerindeki aralıkta ise doğruluk açısından **doygunluğa (plateau)** ulaşıldığını göstermektedir.

Bu nedenle nihai model eğitiminde augmentasyon, genelleme katkısı ile stabiliteyi birlikte sağlayan şiddet aralığında (özellikle **0.3 ve üzeri**) tercih edilmiştir.

## 4.3. Overfitting’i Engelleme Stratejileri

Overfitting’i azaltmak için birden fazla mekanizma birlikte kullanılmıştır:

1. **Pretrained backbone + kontrollü baş (head):** Özgün modelde EfficientNet-B0 backbone pretrained olarak kullanılmış, sınıflandırma için ayrı bir MLP head tasarlanmıştır.
2. **Düzenlileştirme (regularization):**

* Head katmanlarında **Dropout** kullanılmıştır (özgün modelde dropout = 0.35).
* Veri artırma (train-only) uygulanmıştır.
* Optimizasyonda **AdamW + weight decay** kullanılarak L2 benzeri düzenlileştirme sağlanmıştır.

1. **Early Stopping:** Eğitim sırasında doğrulama kaybı (val\_loss) izlenmiş, en iyi val\_loss görüldüğünde model ağırlıkları kaydedilmiş; val\_loss iyileşmezse sayaç artırılarak **patience = 7** sonunda eğitim durdurulmuştur. Böylece gereksiz epoch’larda ezberleme azaltılmıştır.
2. **LR Scheduler:** ReduceLROnPlateau kullanılmıştır (mode = “min”, factor = 0.5, patience = 3). Val\_loss plato yaptığında öğrenme oranı düşürülerek daha stabil yakınsama hedeflenmiştir.
3. **Sınıf dengesizliği (class imbalance) yönetimi:** Eğitim setinde sınıf sayıları farklı olabileceği için compute\_class\_weights ile ters frekans bazlı sınıf ağırlıkları hesaplanmış ve kayıp fonksiyonunda kullanılmıştır (CrossEntropyLoss(weight=class\_weights)).

## 4.4. Parametre Optimizasyonu

Hiperparametre seçimi için **Random Search** uygulanmıştır. Arama uzayı şu parametreleri içerir:

* Öğrenme oranı (lr): log-uniform dağılım, **1e-5 – 3e-3**
* Weight decay (weight\_decay): log-uniform dağılım, **1e-6 – 1e-3**
* Batch size (batch\_size): ayrık seçim, **[8, 16, 24, 32]**

Random search ayarları:

* Deneme sayısı: **N\_TRIALS = 15**
* Her deneme için epoch sayısı: **MAX\_EPOCHS = 10**
* Seçim metriği: **validation accuracy (val\_acc)**, büyük olan daha iyi kabul edilmiştir.

Her deneme, deterministikliği korumak amacıyla farklı ama kontrol edilen seed ile çalıştırılmıştır (seed = 42 + i). Random search sonucunda en iyi konfigürasyon seçilmiş ve nihai eğitim bu değerlerle daha uzun eğitim süresinde yapılmıştır.

## 4.5. Değerlendirme Metrikleri

Tüm modeller için Accuracy ve Loss eğrileri; Confusion Matrix; ROC eğrisi ve AUC; ayrıca tablo halinde precision, recall (sensitivity), F1-score ve Cohen’s kappa raporlanmıştır:

Eğitim sürecinde her epoch sonunda:

* **Train loss / Train accuracy**
* **Validation loss / Validation accuracy**  
  hesaplanmış ve kaydedilmiştir. Bu değerler kullanılarak Accuracy ve Loss eğrileri oluşturulmuştur.

Final değerlendirmede sınıflandırma performansı aşağıdaki metriklerle raporlanmıştır:

* **Confusion Matrix**
* **Accuracy**
* **Precision**
* **Recall / Sensitivity**
* **F1-score**
* **Cohen’s Kappa**
* **ROC eğrisi ve AUC**

Model çıktısı iki sınıf logitidir. Tümör olasılığı, softmax sonrası **pozitif sınıf (tumor=1)** olasılığı olarak alınmıştır. Sınıf kararı için sabit 0.5 eşiğine körü körüne bağlı kalınmamış; önce **yalnızca validation set** üzerinde 0–1 aralığında ince tarama yapılmıştır (1001 adet eşik) ve **F1-score’u maksimize eden threshold** seçilmiştir. Bu eşik seçimi test verisine dokunmadan yapılmış, daha sonra aynı threshold ile test seti değerlendirilmiştir. Böylece test seti “tuning” için kullanılmadığından değerlendirme daha adil kalır.

Çıktı olarak her model koşusunda şu dosyalar üretilmiştir:

* accuracy.png, loss.png (eğitim eğrileri)
* confusion\_matrix.png
* roc\_curve.png
* metrics\_table.png (precision/recall/f1/kappa/accuracy)
* best\_threshold.txt
* best.pt (model ağırlıkları)

## 4.6. Çalışma Akışı (Flowchart)

Bu projede uçtan uca işlem hattı aşağıdaki adımlardan oluşur. Flowchart’ta her madde ayrı kutu olacak şekilde gösterilmiştir:

1. **Veri kaynaklarının hazırlanması ve ayrımı**
   * Ana eğitim havuzu: Mendeley veri seti (tumor/no\_tumor).
   * Dış kaynak test: iki ayrı Kaggle veri seti.  
     (Amaç: aynı kişiye ait tekrar görüntülerden kaynaklı subject leakage riskini azaltmak ve genellemeyi dış testte ölçmek.)
2. **Çalışma ortamının kurulması ve tekrar üretilebilirlik**
   * CPU/GPU seçimi, seed ayarlama (Python/NumPy/PyTorch), DataLoader worker seed kontrolü.
3. **Transform / preprocessing pipeline’ının tanımlanması**
   * SquarePad → Resize (sabit input\_size) → ToTensor → Normalize
   * Train için opsiyonel: Augmentation (aug\_strength’e bağlı) + RandomErasing
   * Val/Test: augmentasyon kapalı, sadece preprocess.
4. **Eğitim–doğrulama ayrımı (stratified split)**
   * Training havuzundan sınıf oranlarını koruyacak şekilde validation ayrılır.
5. **DataLoader oluşturma**
   * Train / Val / Test için batch, shuffle ve num\_workers ayarları ile yükleyicilerin hazırlanması.
6. **Model kurulumunun yapılması ve 10 modelin denenmesi**  
   Bu projede toplam **10 farklı model** eğitilmiştir:
   * **7 adet normal (tek-backbone) model:** Standart transfer öğrenme yaklaşımıyla tek bir backbone üzerinden sınıflandırma.
   * **2 adet hibrit model:** İki farklı backbone’un özelliklerini birleştiren mimariler.
     + **Hibrit-1:** **DenseNet121 + EfficientNetB0**
     + **Hibrit-2:** **EfficientNetB0 + Swin-T**
   * **1 adet özgün (kendi geliştirdiğimiz) model:** **Multi-Scale feature extraction + attention/fusion** yaklaşımı içeren özel mimari (multiscale feature çıkarımı yapılan model).

Flowchart’ta bu adım, “Model Seçimi / Model Profili” kutusu altında **(Normal / Hibrit / Custom)** olarak üç dala ayrılacak şekilde gösterilmiştir.

1. **Kayıp fonksiyonu ve optimizasyon ayarları**
   * (Varsa) class weight ile ağırlıklı loss, optimizer (AdamW), weight decay, LR scheduler.
2. **Eğitim döngüsü**
   * Epoch bazlı train→val ilerleyişi, metriklerin her epoch sonunda kaydı (loss/accuracy).
   * (Varsa) mixed precision (AMP), gradient scaling.
3. **Early Stopping ve en iyi ağırlıkların seçimi**
   * Validation performansına göre en iyi model checkpoint’inin kaydedilmesi ve aşırı öğrenmenin durdurulması.
4. **Threshold seçimi (sadece validation üzerinde)**

* Pozitif sınıf olasılığı için farklı eşikler denenir, seçilen kriter (örn. F1 maksimum) ile en iyi threshold belirlenir.

1. **Dış test değerlendirmesi (external test)**

* Seçilen threshold ile test setinde Confusion Matrix, ROC/AUC ve diğer metrikler hesaplanır.

1. **Sonuçların kaydedilmesi ve raporlanması**

* Accuracy/Loss eğrileri, Confusion Matrix, ROC eğrisi, metrics table görselleri ve özet karşılaştırma tablosu üretilir.
* En iyi model ayrıca kaydedilir ve örnek tahmin akışı rapora eklenir.

**Flowchart çizimi için net şema (kutular böyle olsun)**

* **Data Sources** → **Preprocess & Augmentation** → **Split (Train/Val)** → **Dataloader** → **Model Branch**
  + **Normal Models (7)**
  + **Hybrid-1: DenseNet121 + EfficientNetB0**
  + **Hybrid-2: EfficientNetB0 + Swin-T**
  + **Custom: Multi-Scale Feature + Fusion**  
    → **Train + Scheduler + EarlyStopping** → **Threshold Selection (Val)** → **External Test Metrics** → **Save Results & Best Model**

# 5. Modeller ve Sonuçlar

Bu bölümde 13 farklı model raporlanır. Her model için: (i) kısa mimari/yöntem açıklaması, (ii) eğitim ayarları, (iii) Accuracy-Loss grafikleri, (iv) Confusion Matrix, (v) ROC eğrisi ve (vi) metrik tablosu verilerek yorumlanır.

## 5.1. Tüm Modellerin Karşılaştırma Tablosu

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model | Accuracy | Precision | Recall/Sensitivity | F1-Score | Kappa | AUC |
| my\_model\_with\_aug0.3.pt | 0.9084 | 0.9978 | 0.8218 | 0.9013 | 0.8173 | 0.9884 |
| hybrid\_dn121\_effb0\_with\_aug\_0.3.pt | 0.8607 | 1.0 | 0.7262 | 0.8414 | 0.7227 | 0.9665 |
| hybrid\_dn121\_effb0.pt | 0.8392 | 1.0 | 0.6839 | 0.8123 | 0.6801 | 0.9389 |
| my\_model.pt | 0.8054 | 0.9990 | 0.6181 | 0.7637 | 0.6133 | 0.9355 |
| hybrid\_swinT\_effb0\_with\_aug\_0.3.pt | 0.7949 | 0.9969 | 0.5987 | 0.7482 | 0.5926 | 0.9749 |
| resnet34.pt | 0.7943 | 0.9989 | 0.5963 | 0.7468 | 0.5914 | 0.9538 |
| Densenet121.pt | 0.7851 | 1.0 | 0.5776 | 0.7323 | 0.5733 | 0.9620 |
| convnext\_tiny.pt | 0.7746 | 1.0 | 0.5570 | 0.7155 | 0.5527 | 0.9603 |
| hybrid\_swinT\_effb0.pt | 0.7448 | 1.0 | 0.4984 | 0.6653 | 0.4940 | 0.9562 |
| resnet50.pt | 0.7193 | 1.0 | 0.4483 | 0.6191 | 0.4439 | 0.9620 |
| inception\_v3.pt | 0.7101 | 1.0 | 0.4302 | 0.6016 | 0.4258 | 0.9005 |
| efficientnet\_b0.pt | 0.6925 | 0.9969 | 0.3969 | 0.5678 | 0.3915 | 0.9027 |
| mobilenetv2\_100.pt | 0.6387 | 1.0 | 0.2900 | 0.4496 | 0.2864 | 0.8894 |

Tablo 1. Tüm modellerin metriklerle karşılaştırması.

## 5.2. Model 1: My\_model\_aug\_0.3

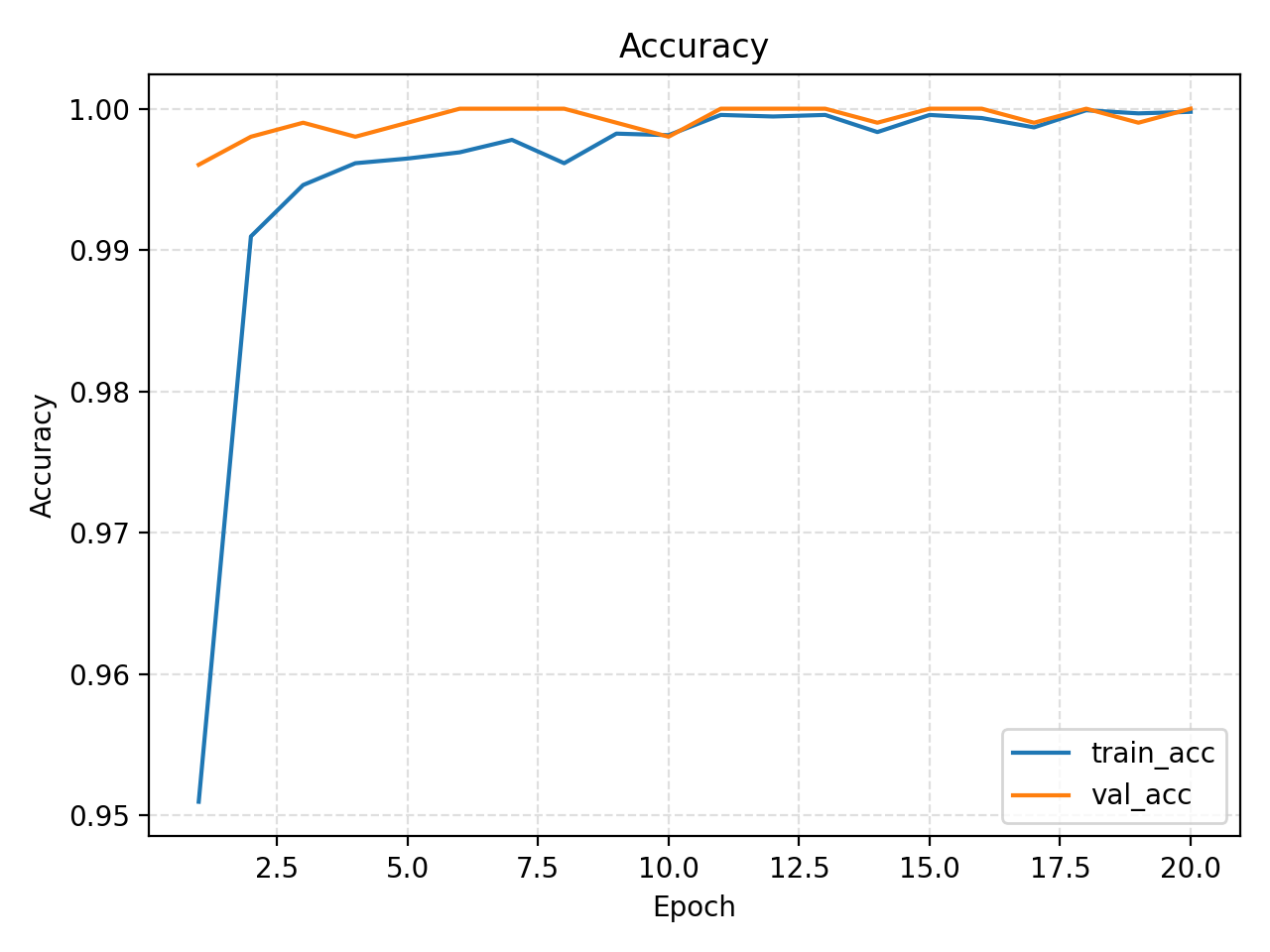
### Eğitim Ayarları

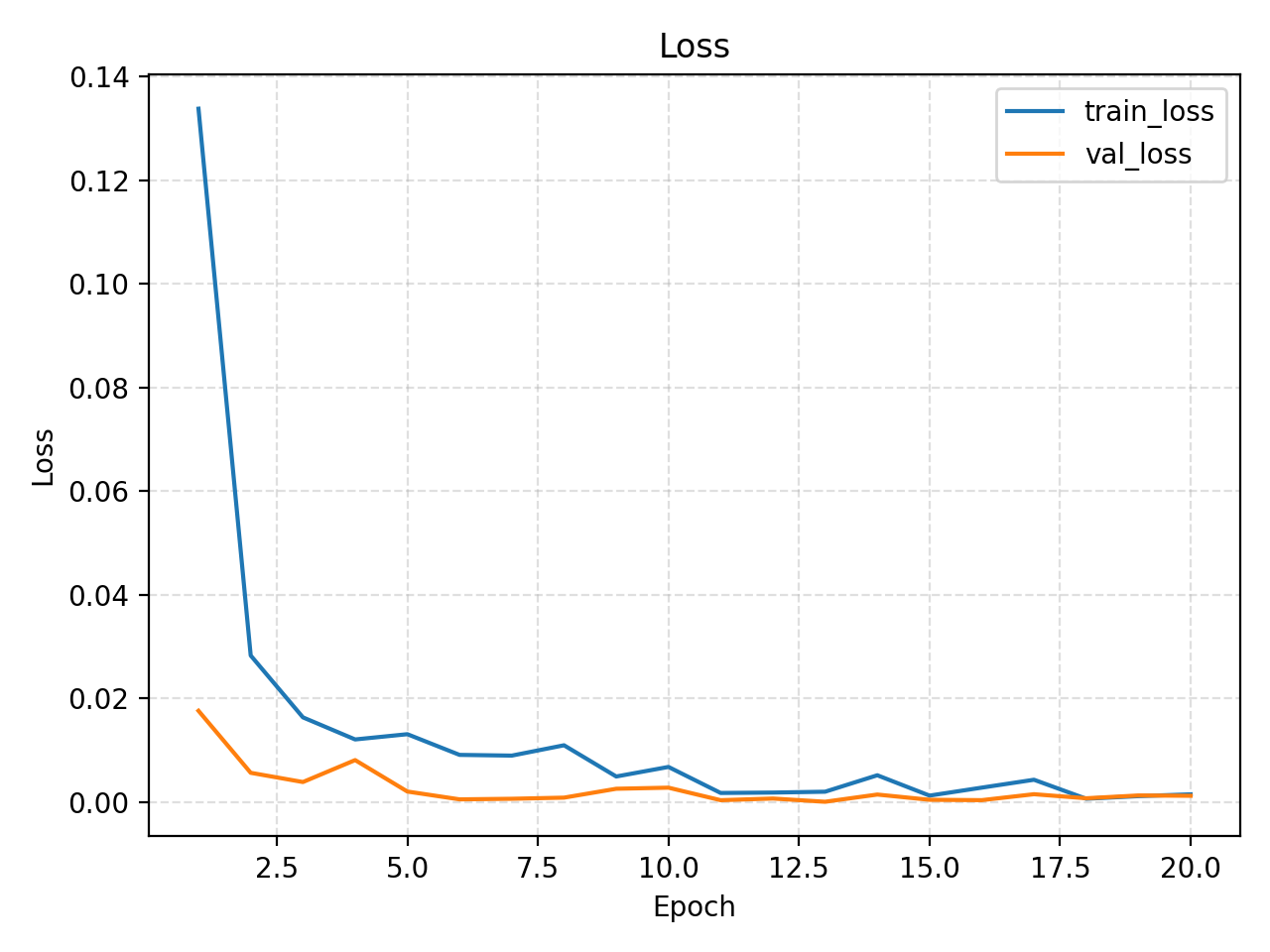
*Proje klasörlerinde model ismi içerisindeki her kod dosyası o modele özgüdür random search sonuçları içerisinde kayıtlıdır.*

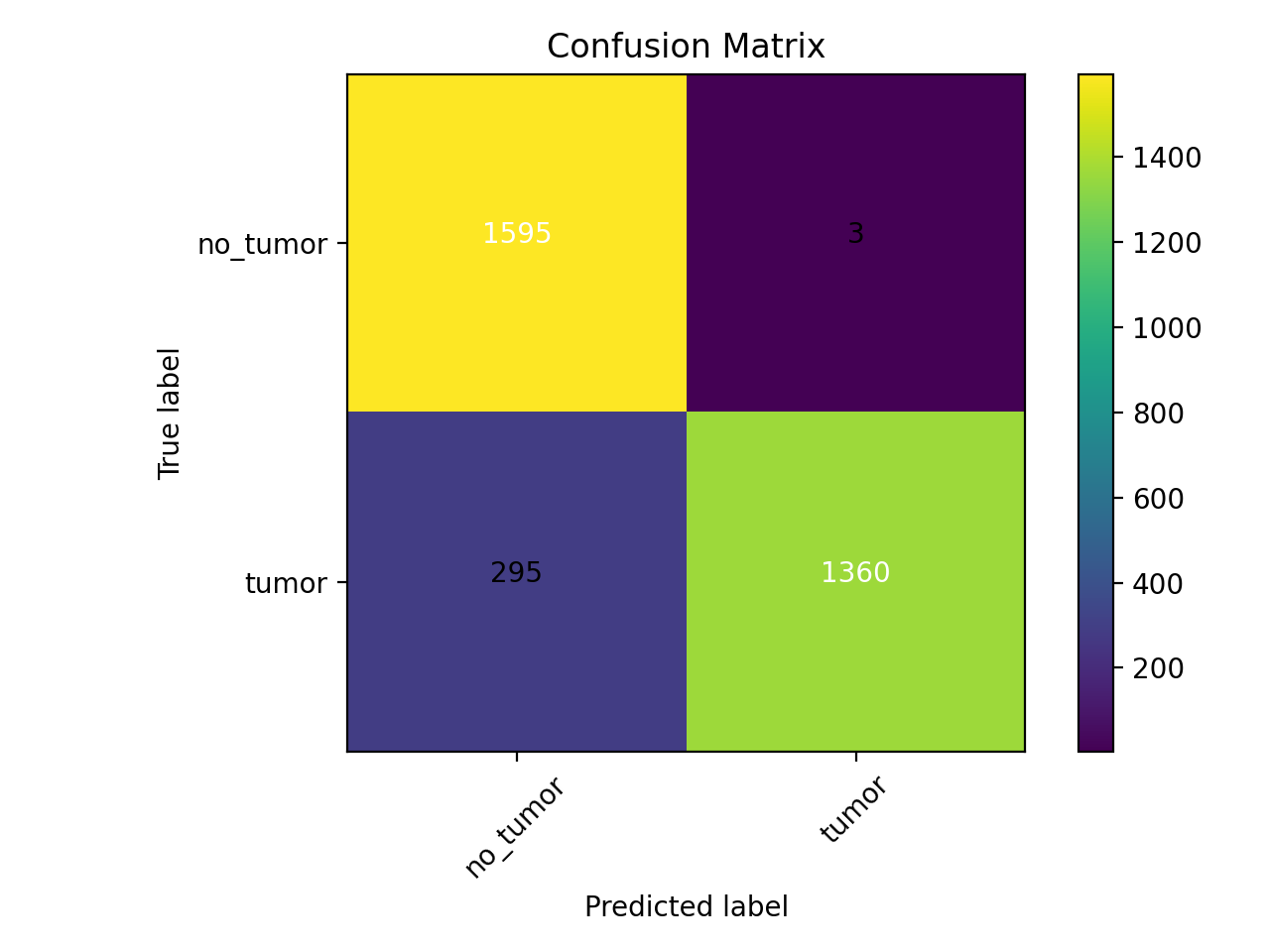
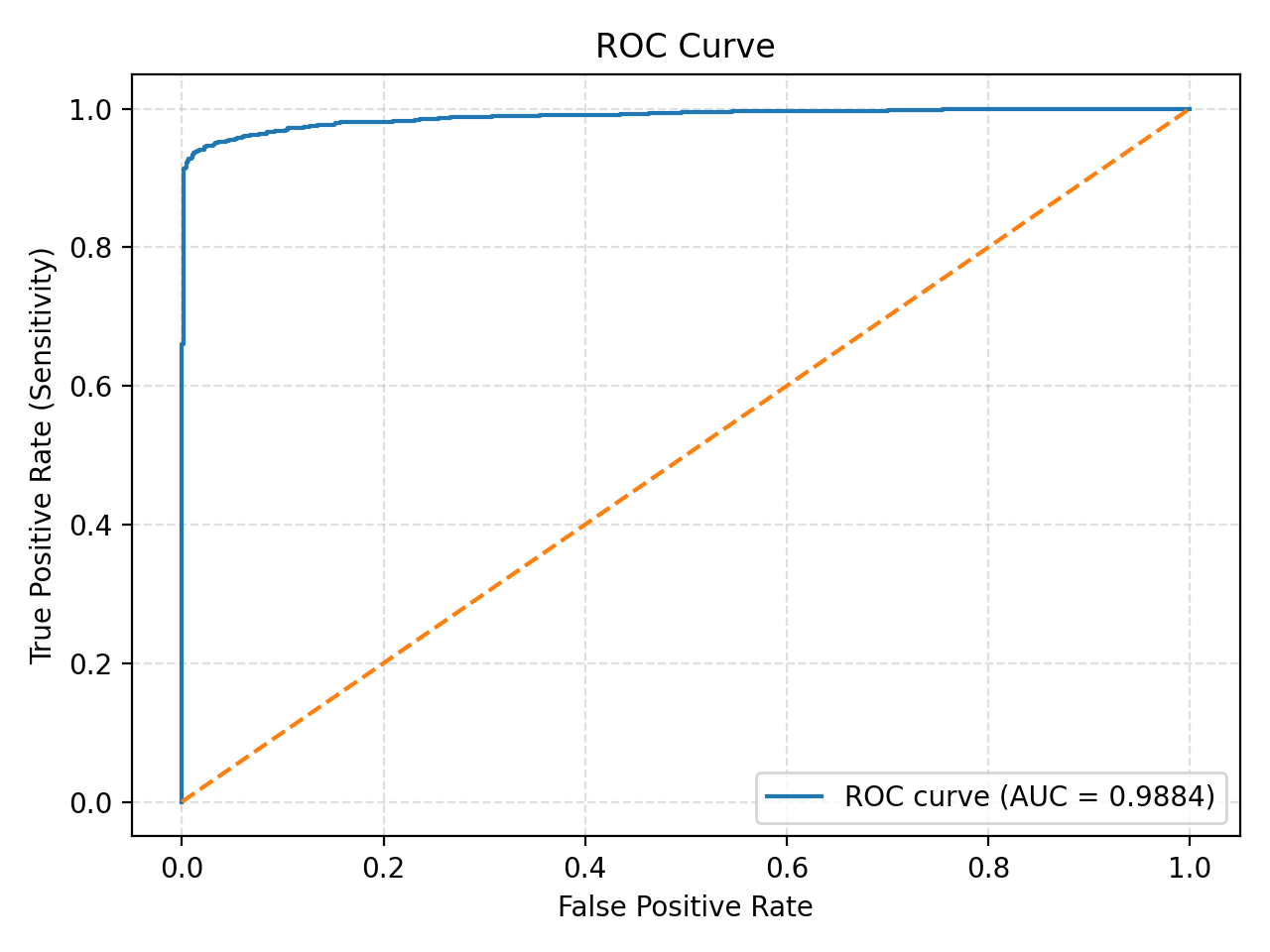
### Sonuç Grafikleri

metin, ekran görüntüsü, çizgi, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

**

**

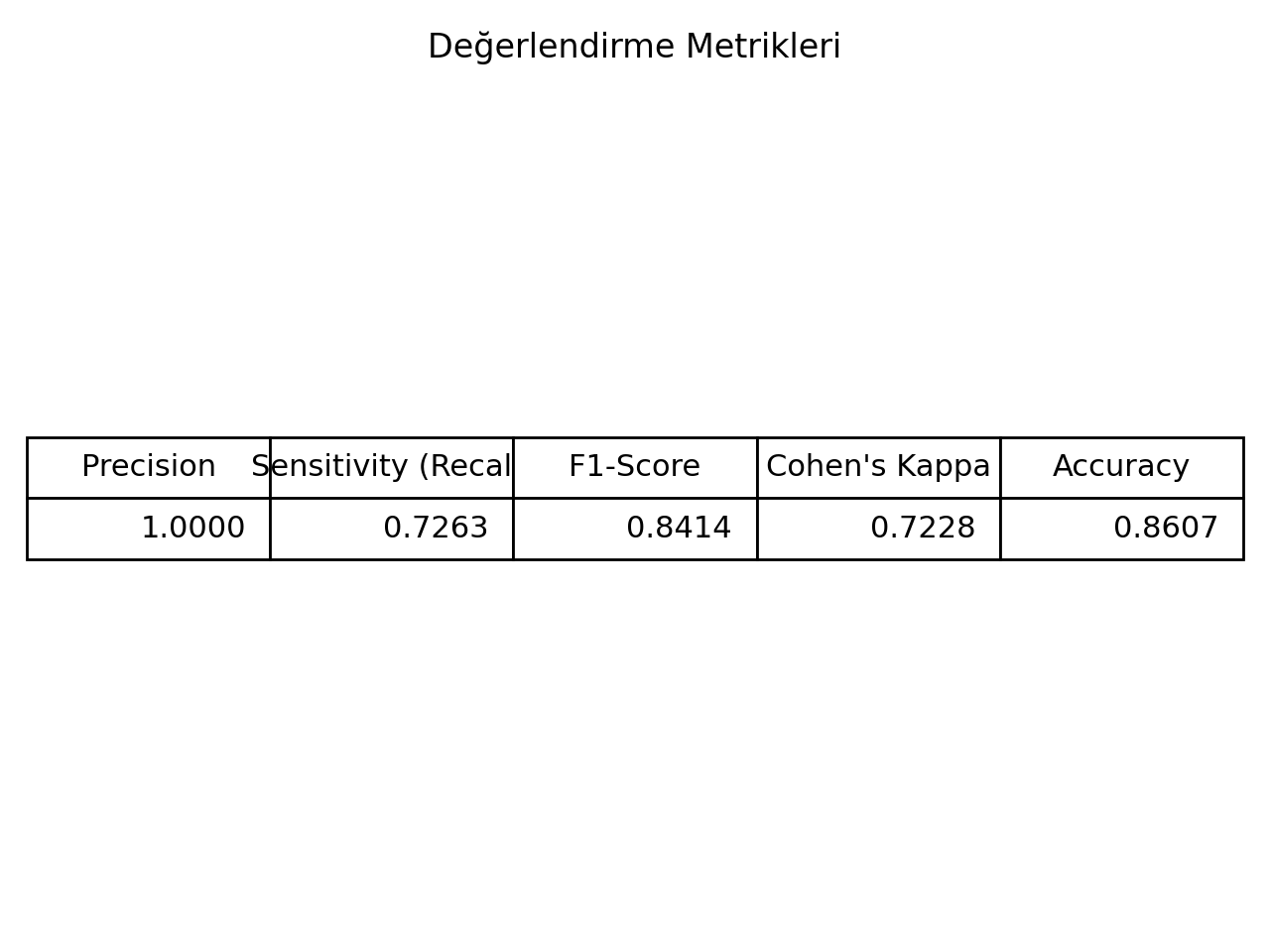
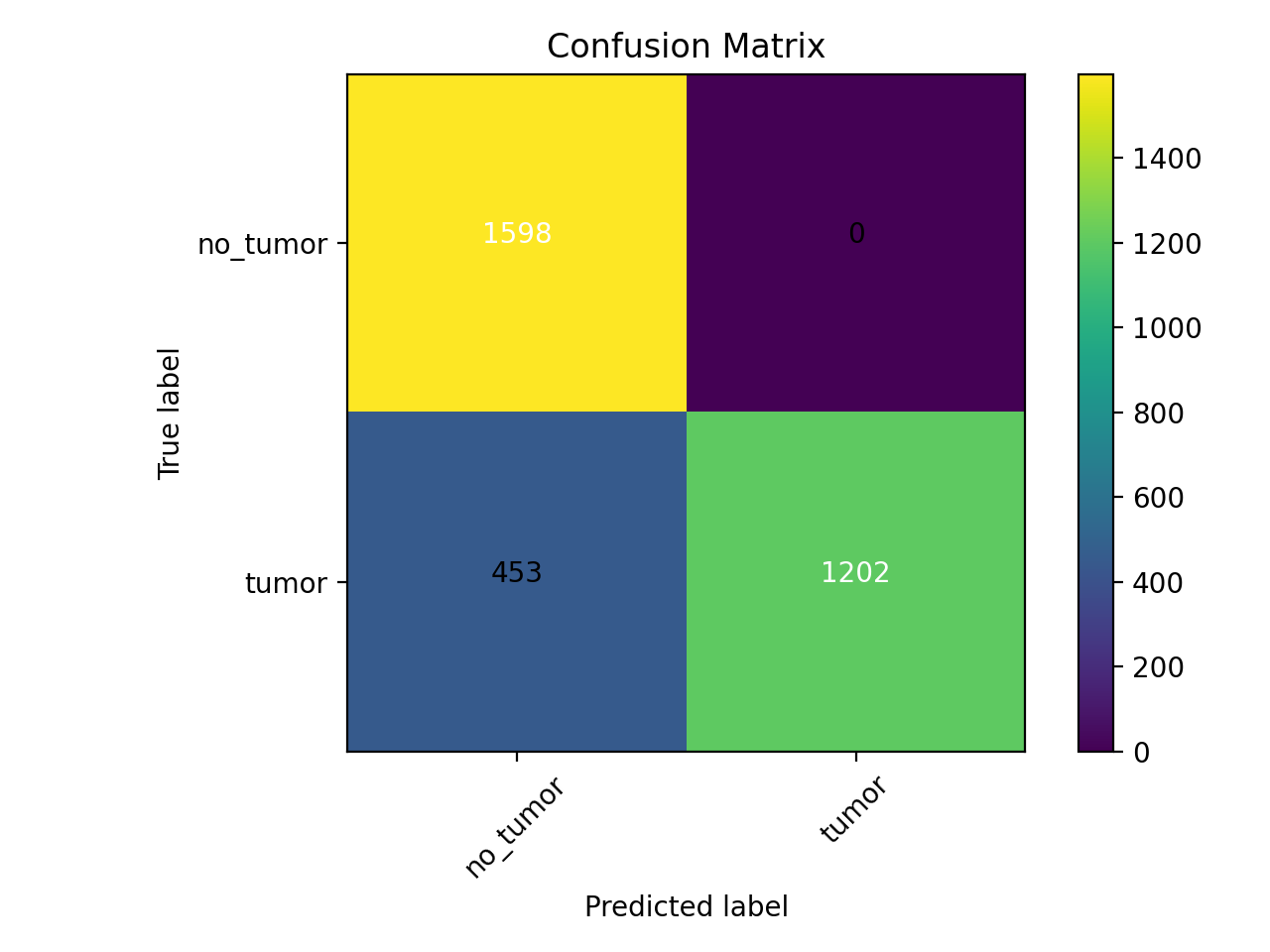
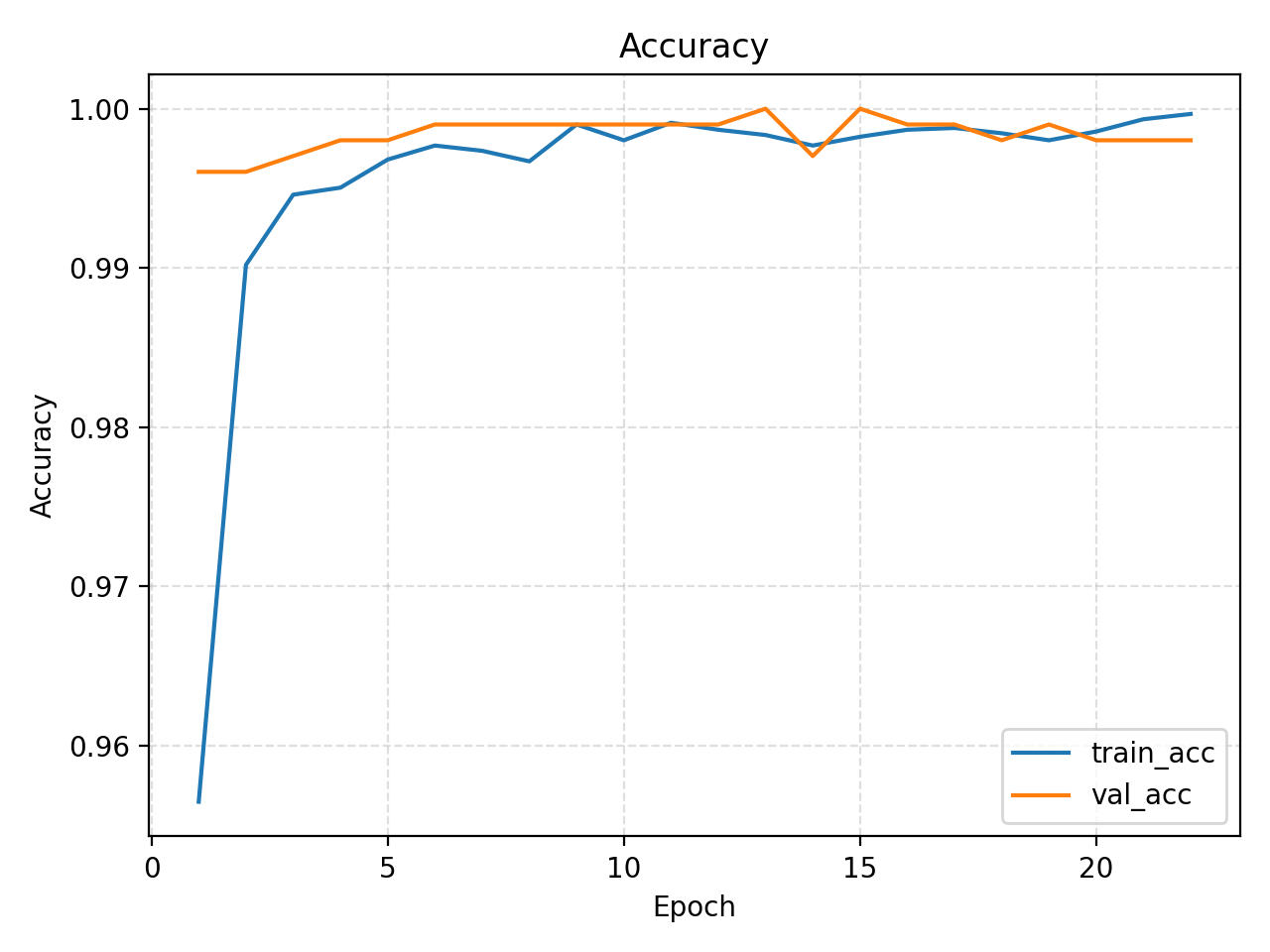


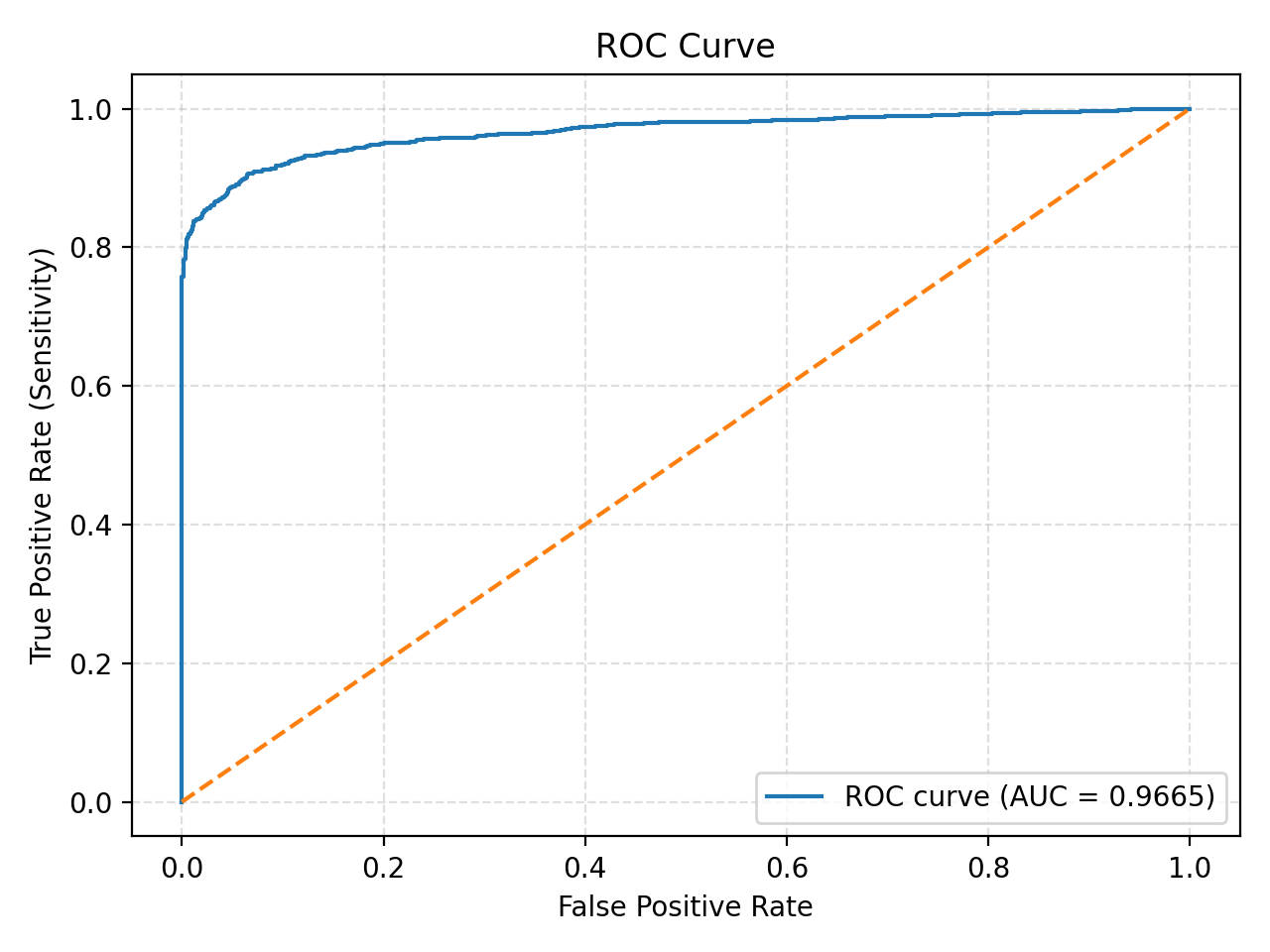
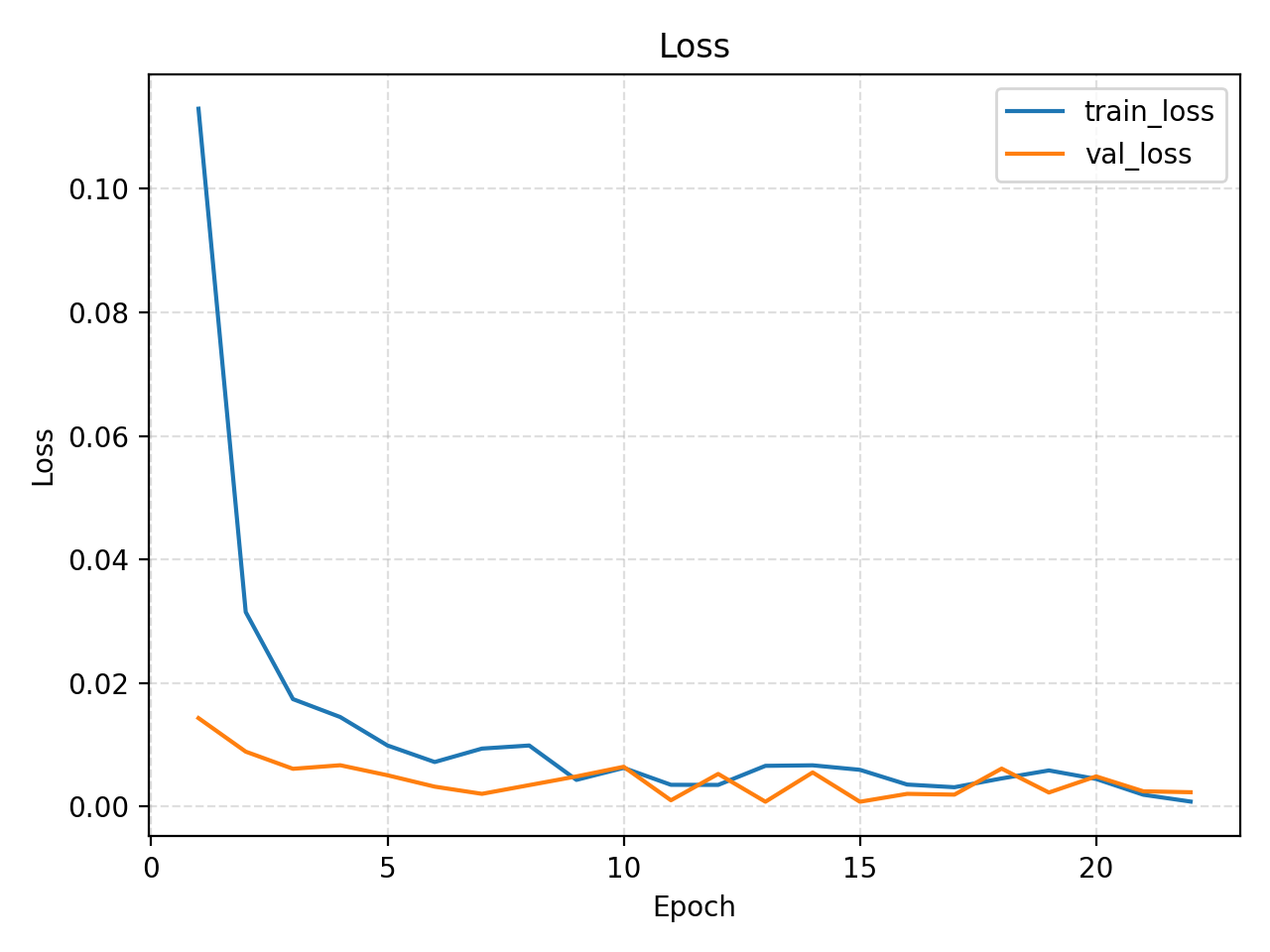
## 5.3. Model 2: hybrid\_dn121\_effb0\_with\_aug\_0.3.pt

### Eğitim Ayarları

*Proje klasörlerinde model ismi içerisindeki her kod dosyası o modele özgüdür random search sonuçları içerisinde kayıtlıdır.*

### Sonuç Grafikleri





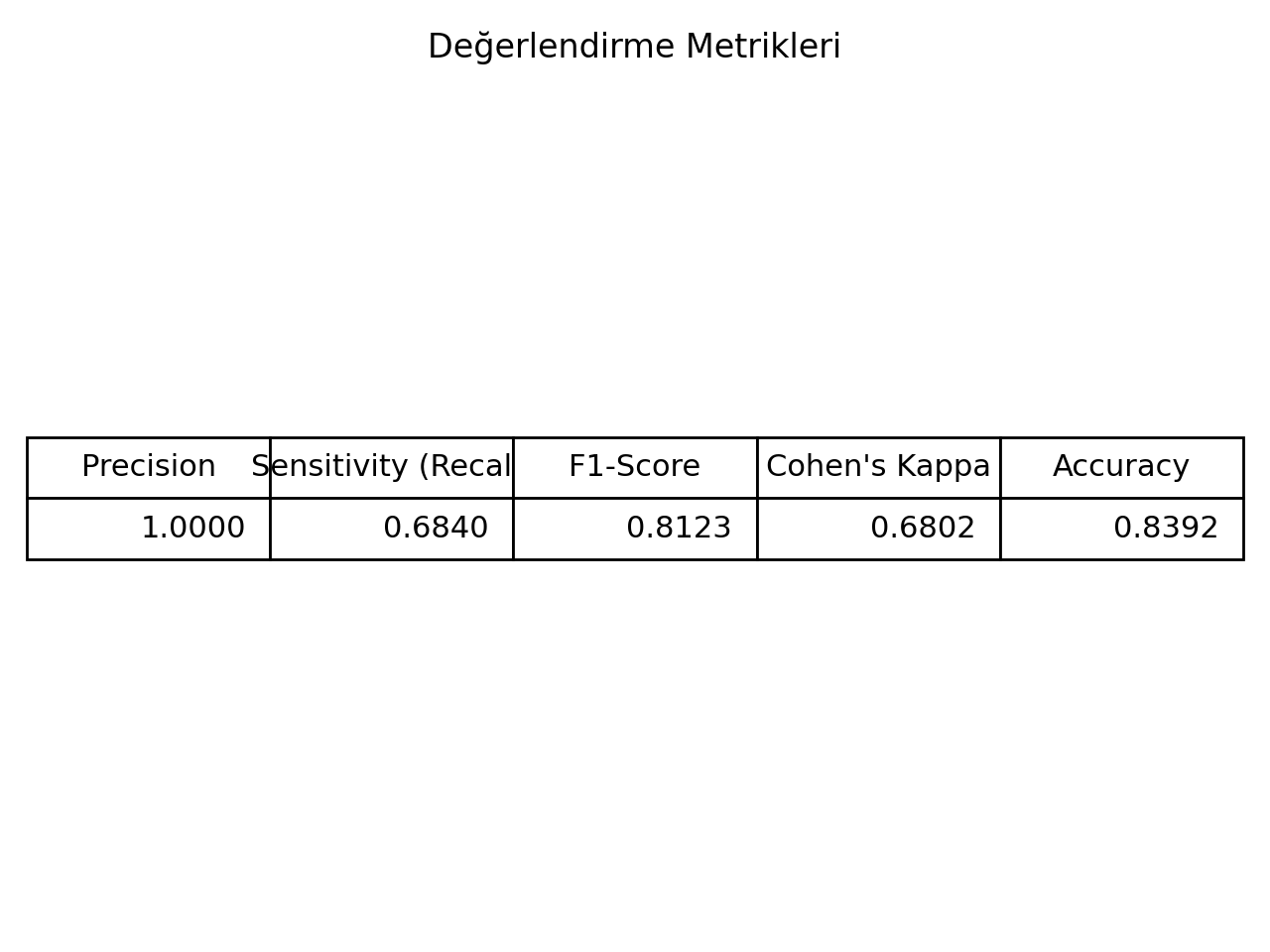
*.*

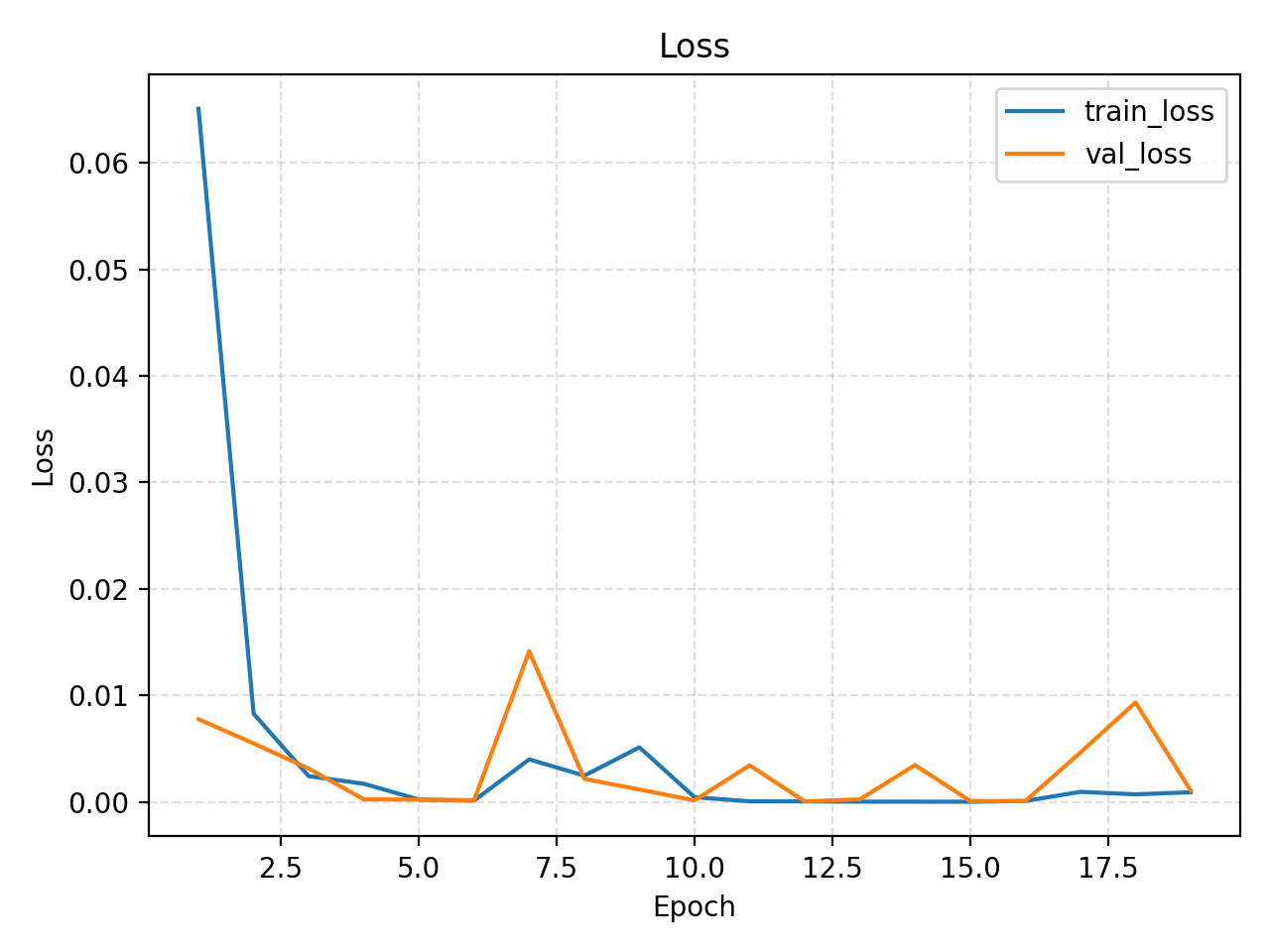
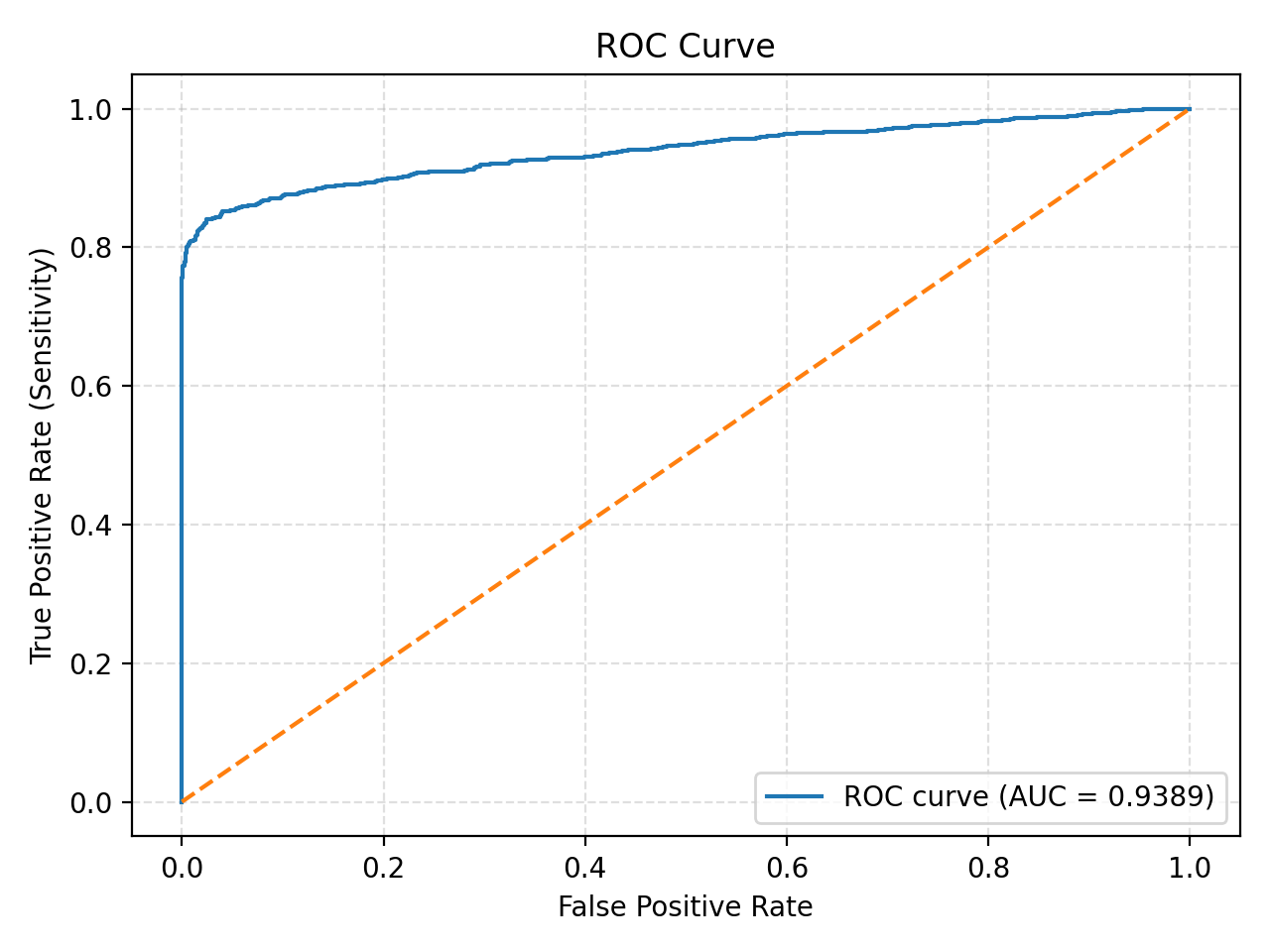
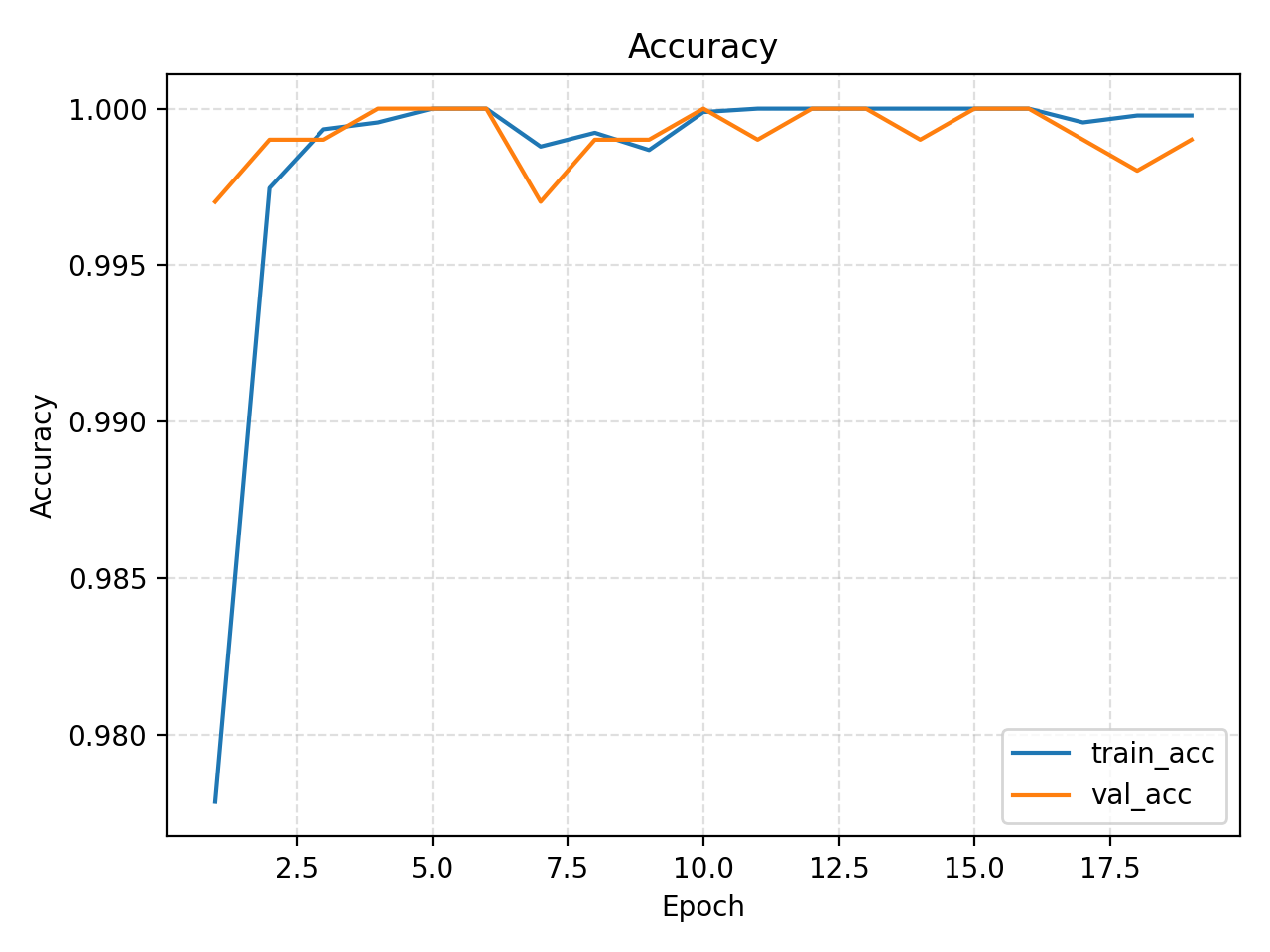
## 5.4. Model 3: hybrid\_dn121\_effb0.pt

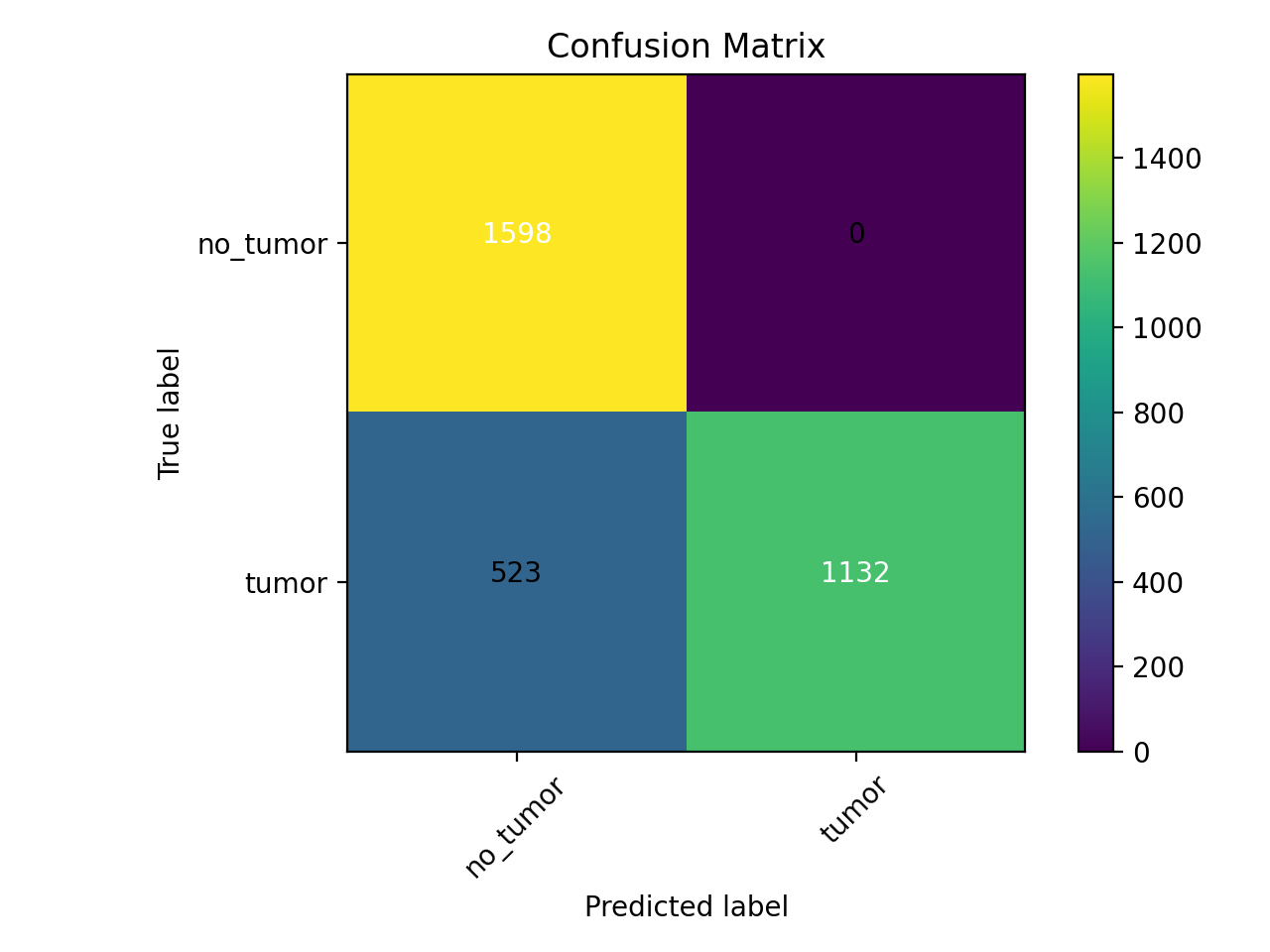
### Eğitim Ayarları

*Proje klasörlerinde model ismi içerisindeki her kod dosyası o modele özgüdür random search sonuçları içerisinde kayıtlıdır.*

### Sonuç Grafikleri

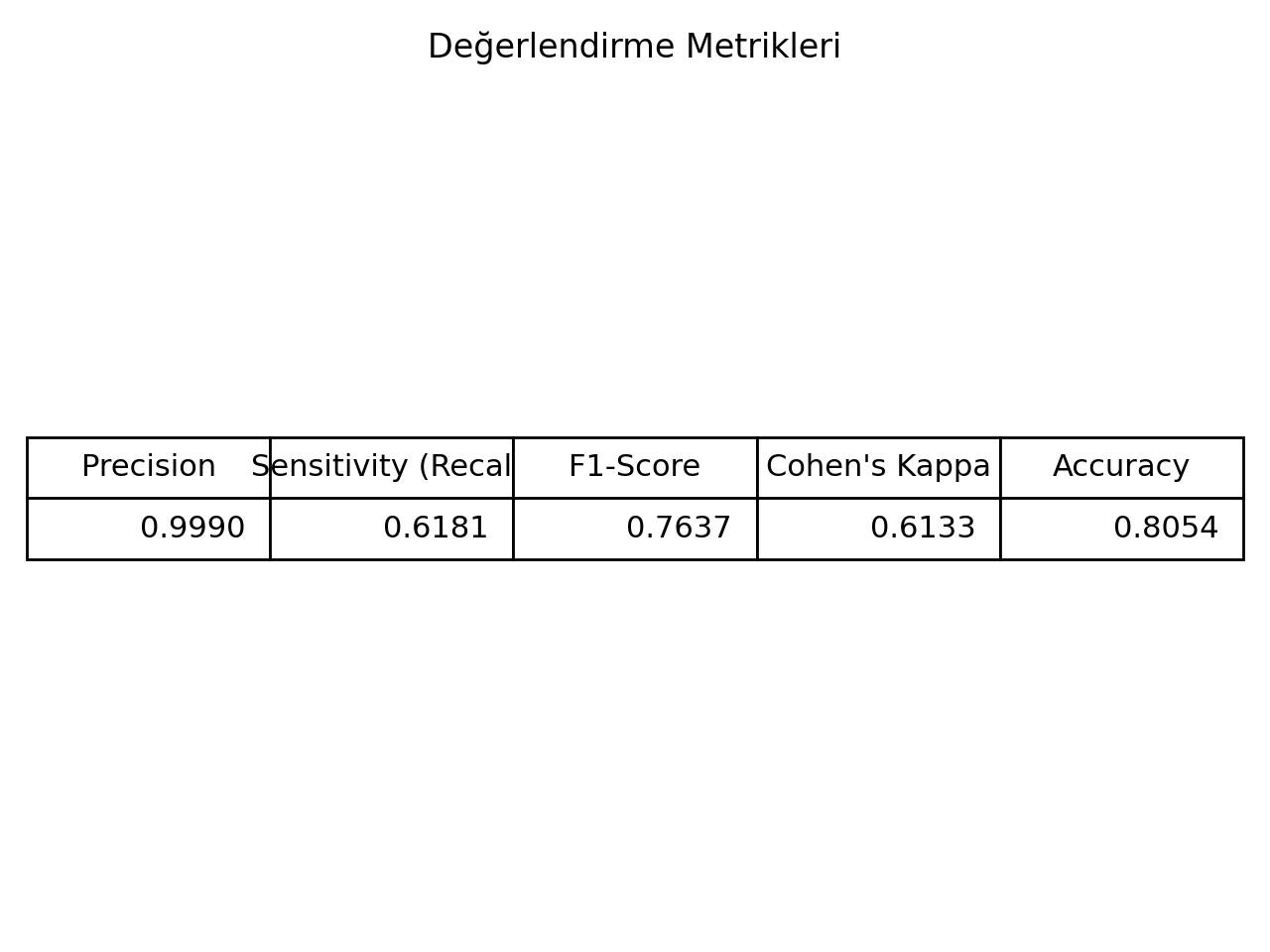




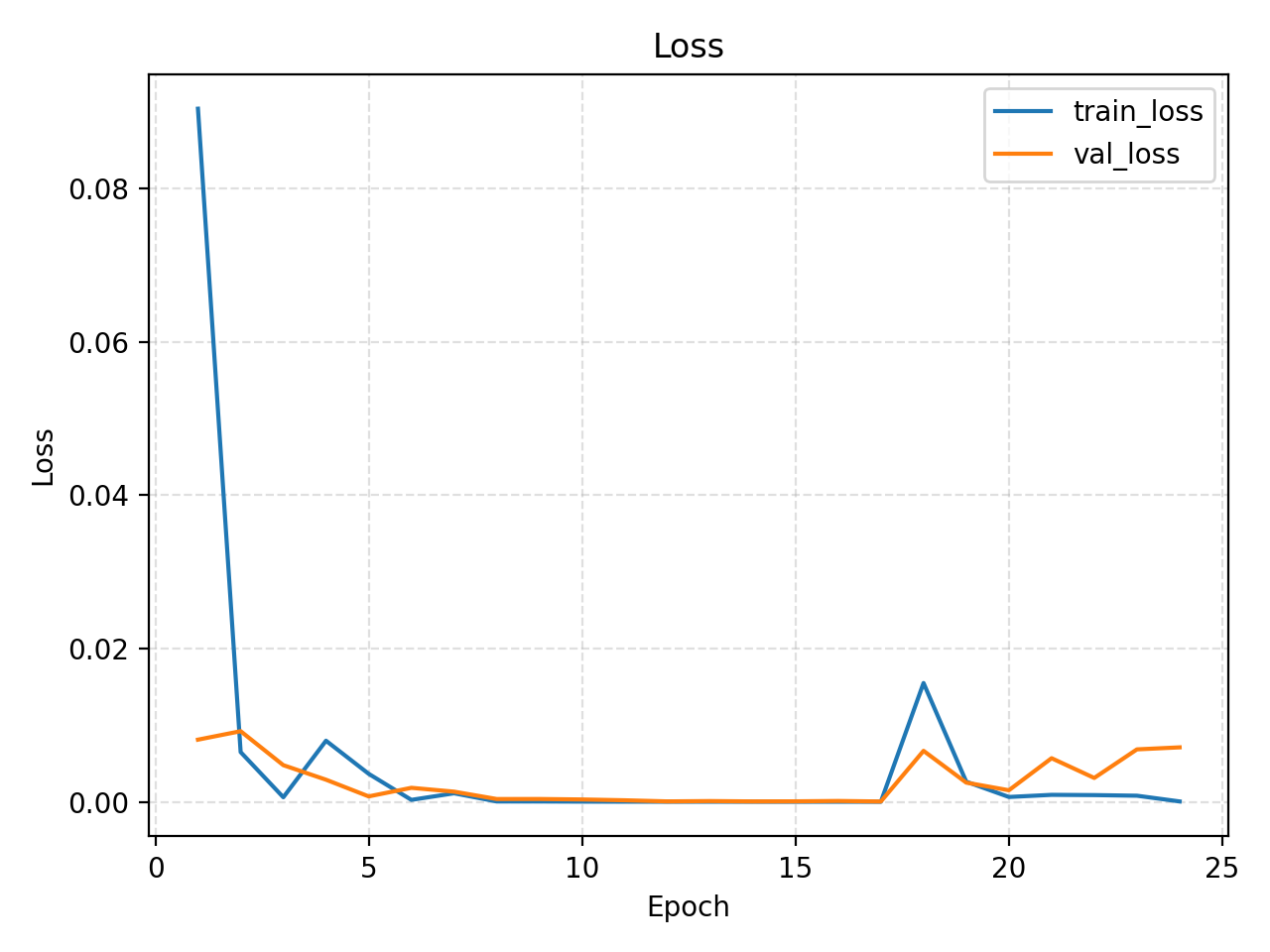


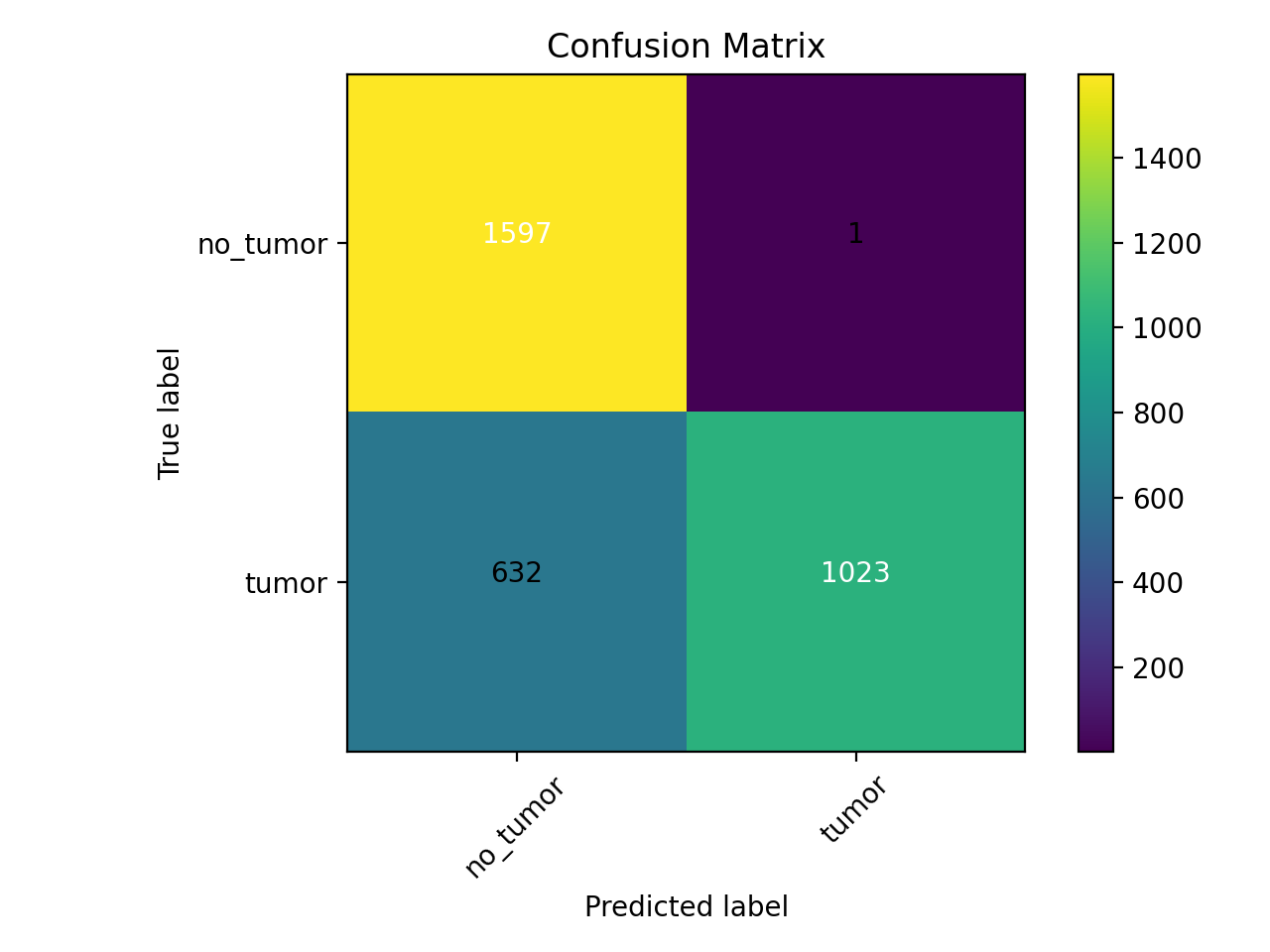
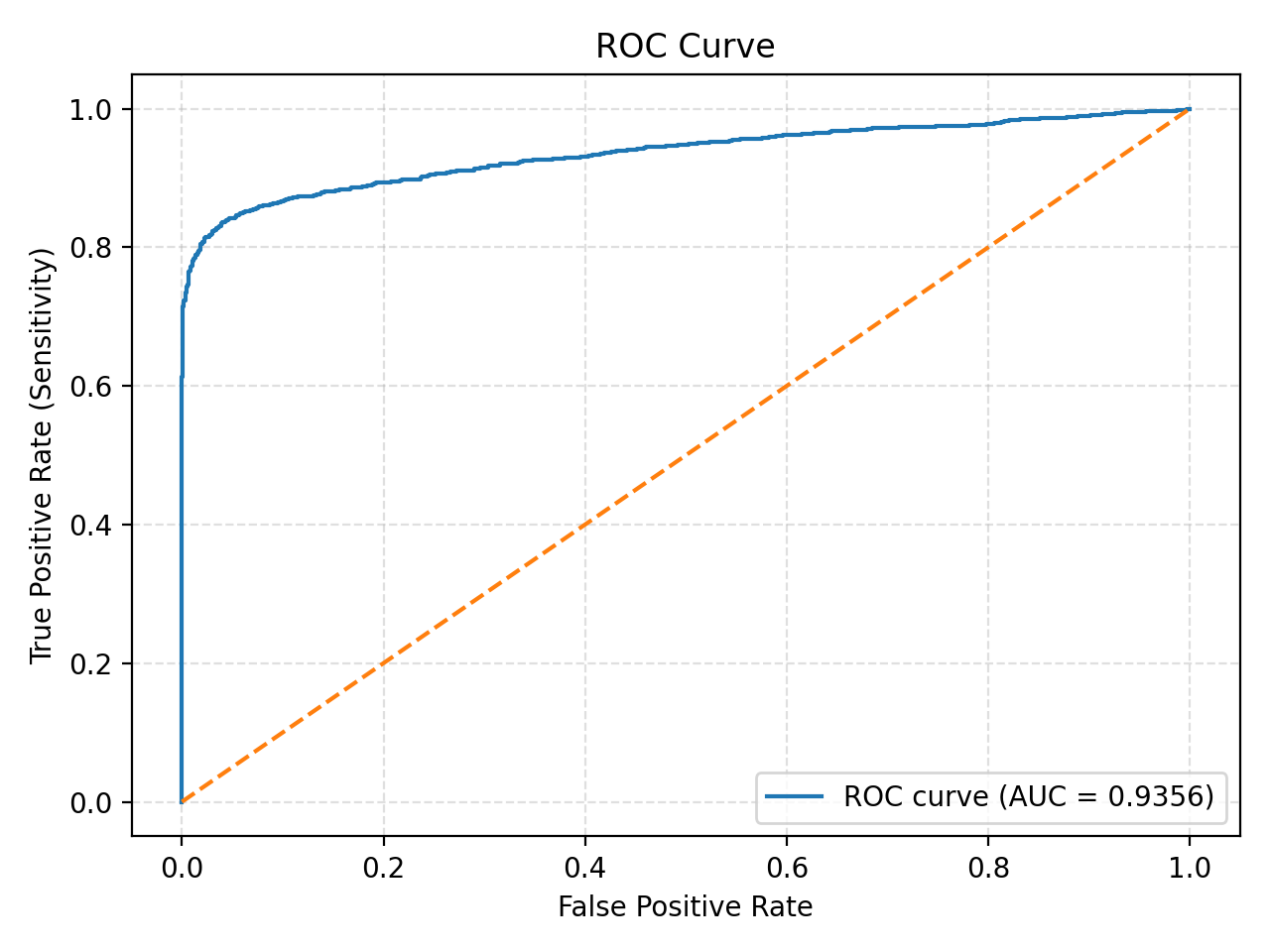
## 5.5. Model 4: my\_model.pt

### Sonuç Grafikleri

metin, diyagram, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, çizgi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.





### Yorum

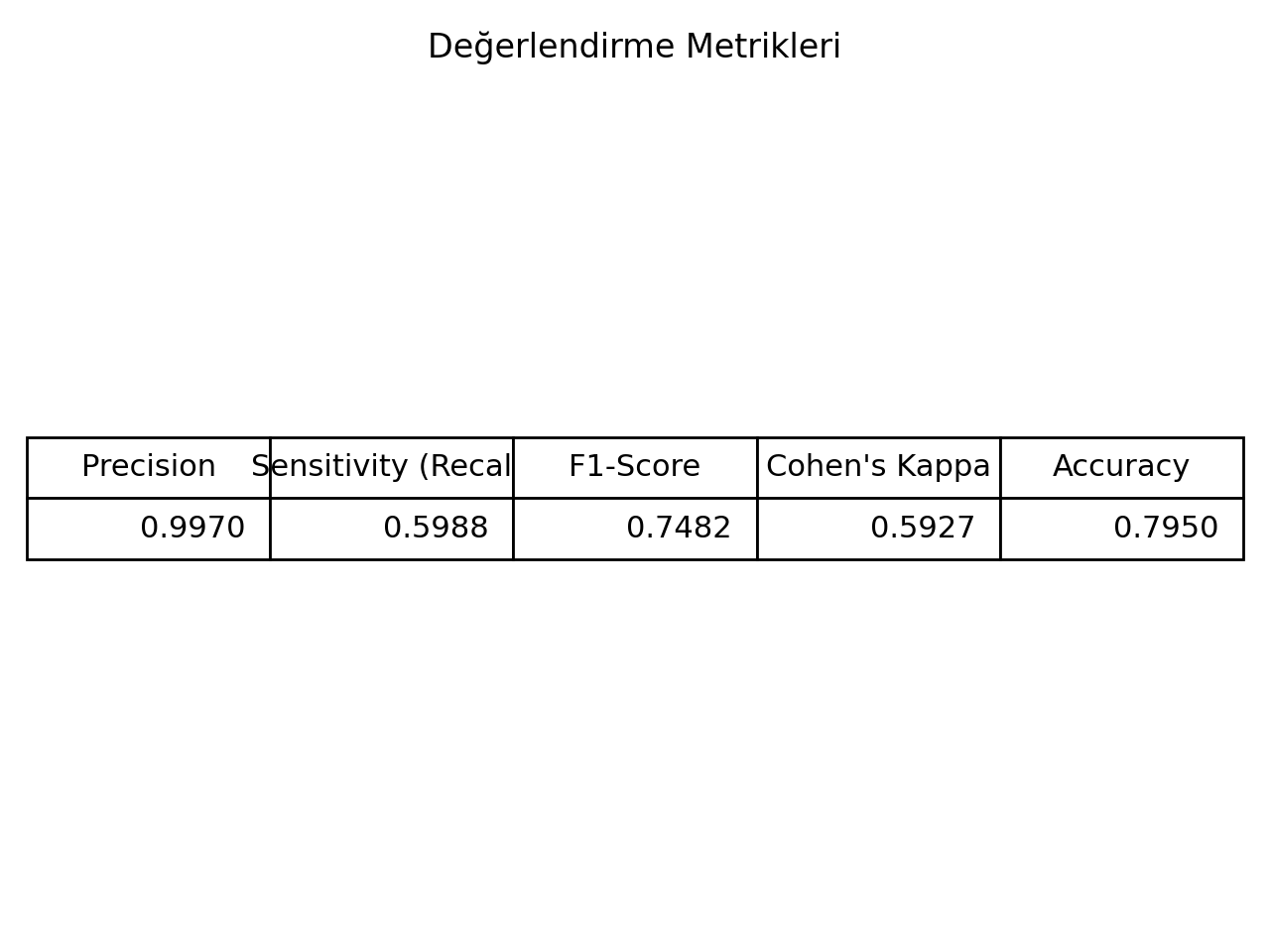
*Augmentasyon edildikten sonra ciddi doğruluk kazanıldığı tespit edildi.*

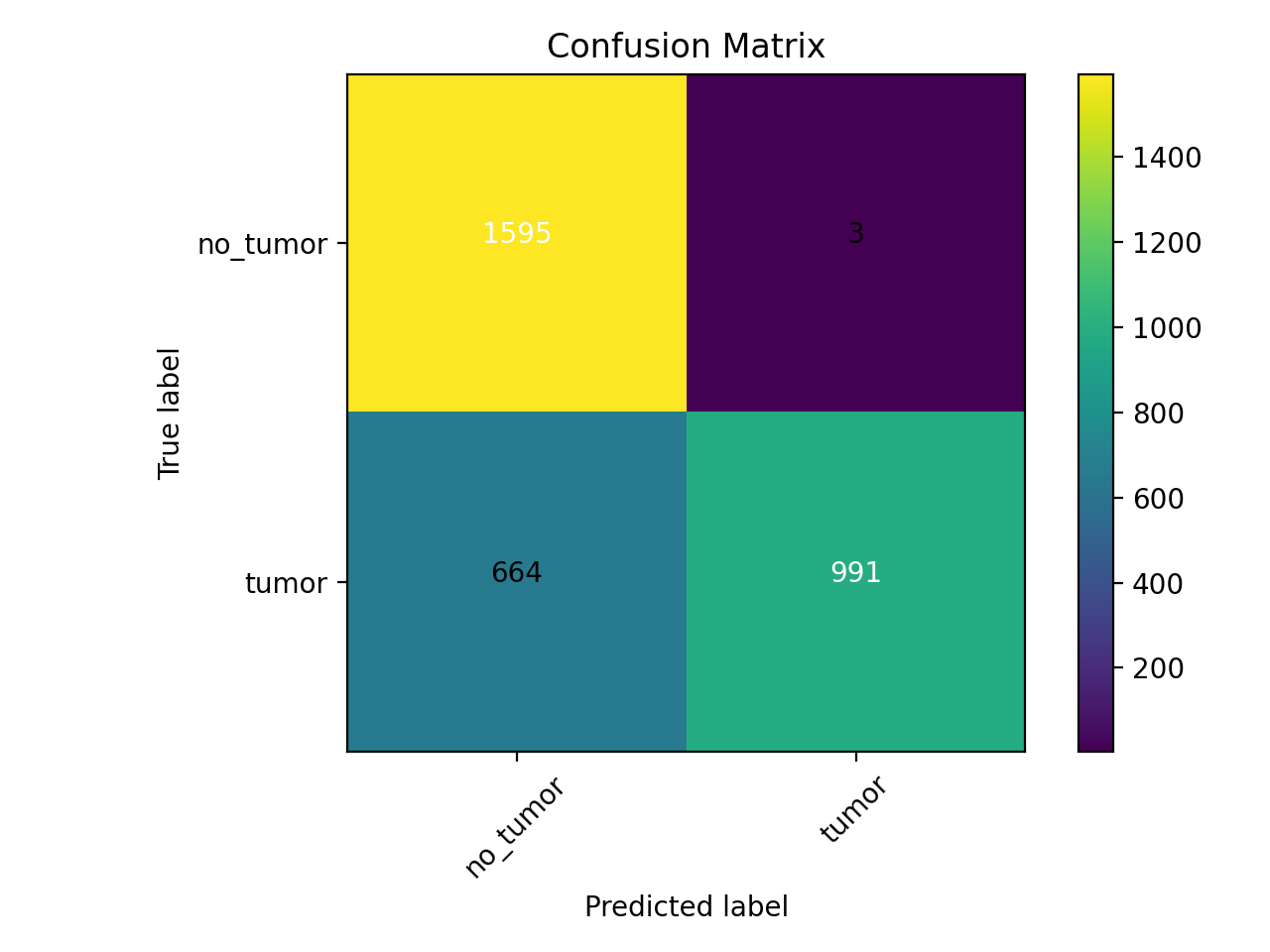
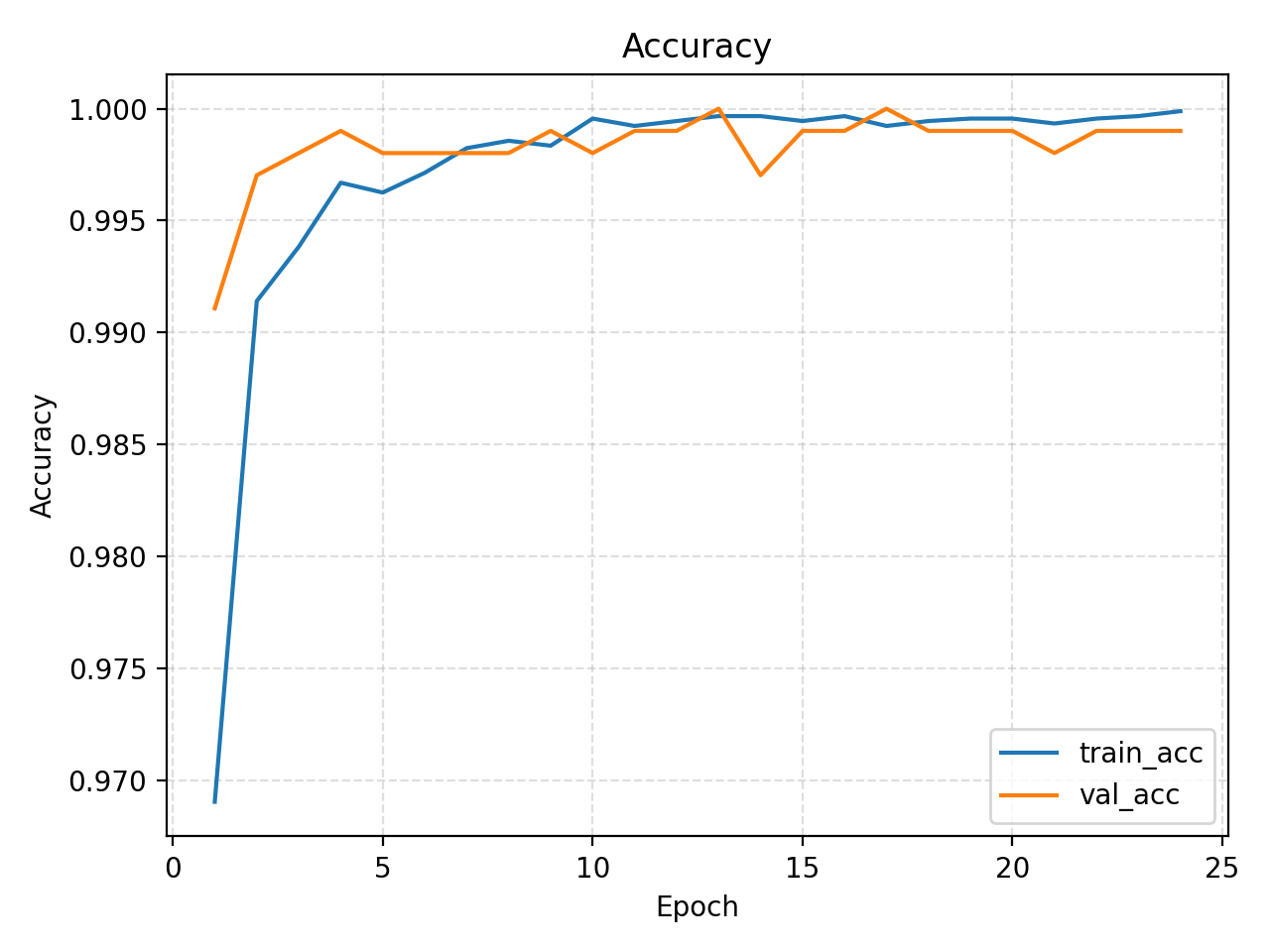
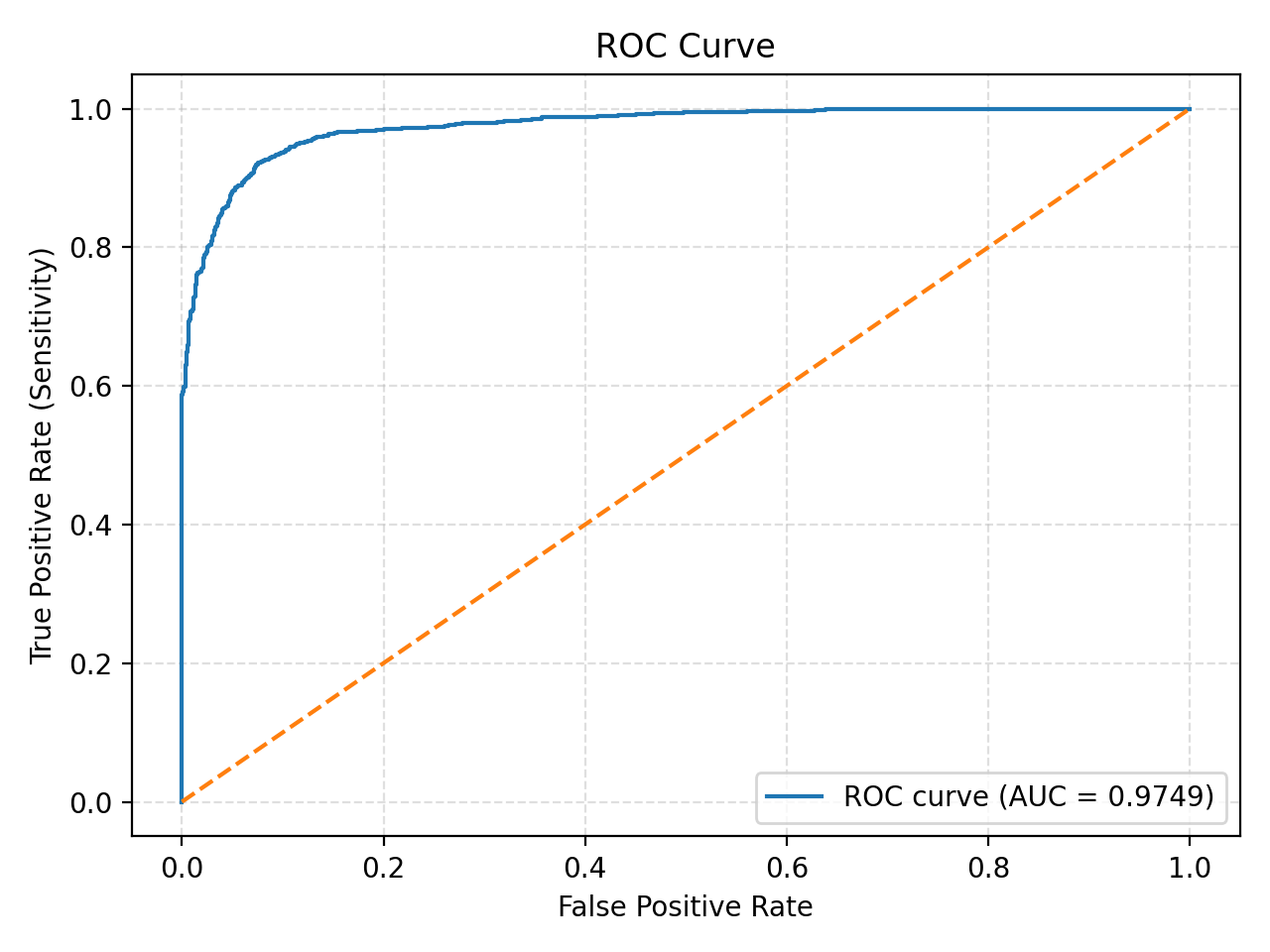
## 5.6. Model 5: hybrid\_swinT\_effb0\_with\_aug\_0.3.pt

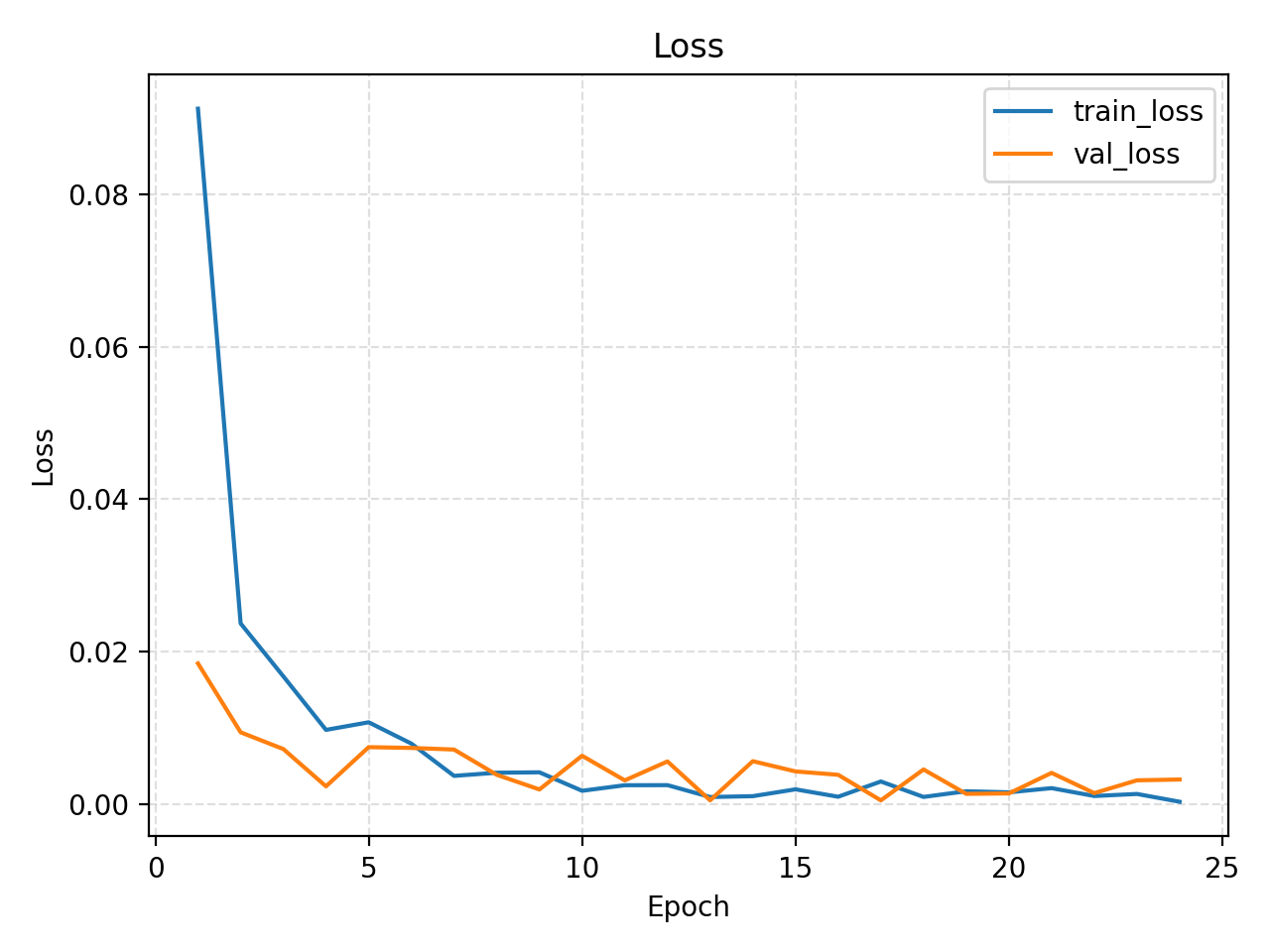
### Eğitim Ayarları

### *Proje klasörlerinde model ismi içerisindeki her kod dosyası o modele özgüdür random search sonuçları içerisinde kayıtlıdır.*

### Sonuç Grafikleri

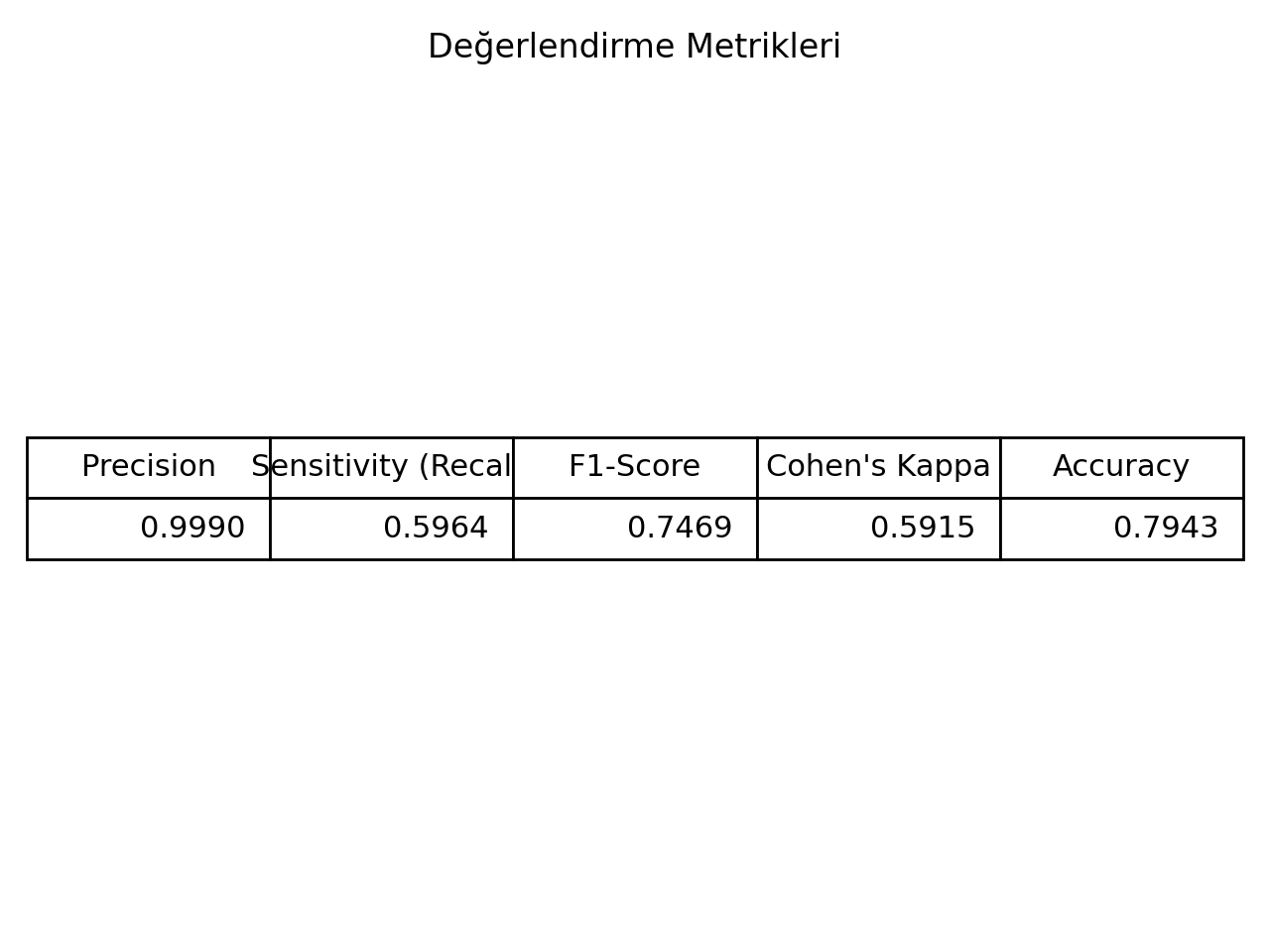


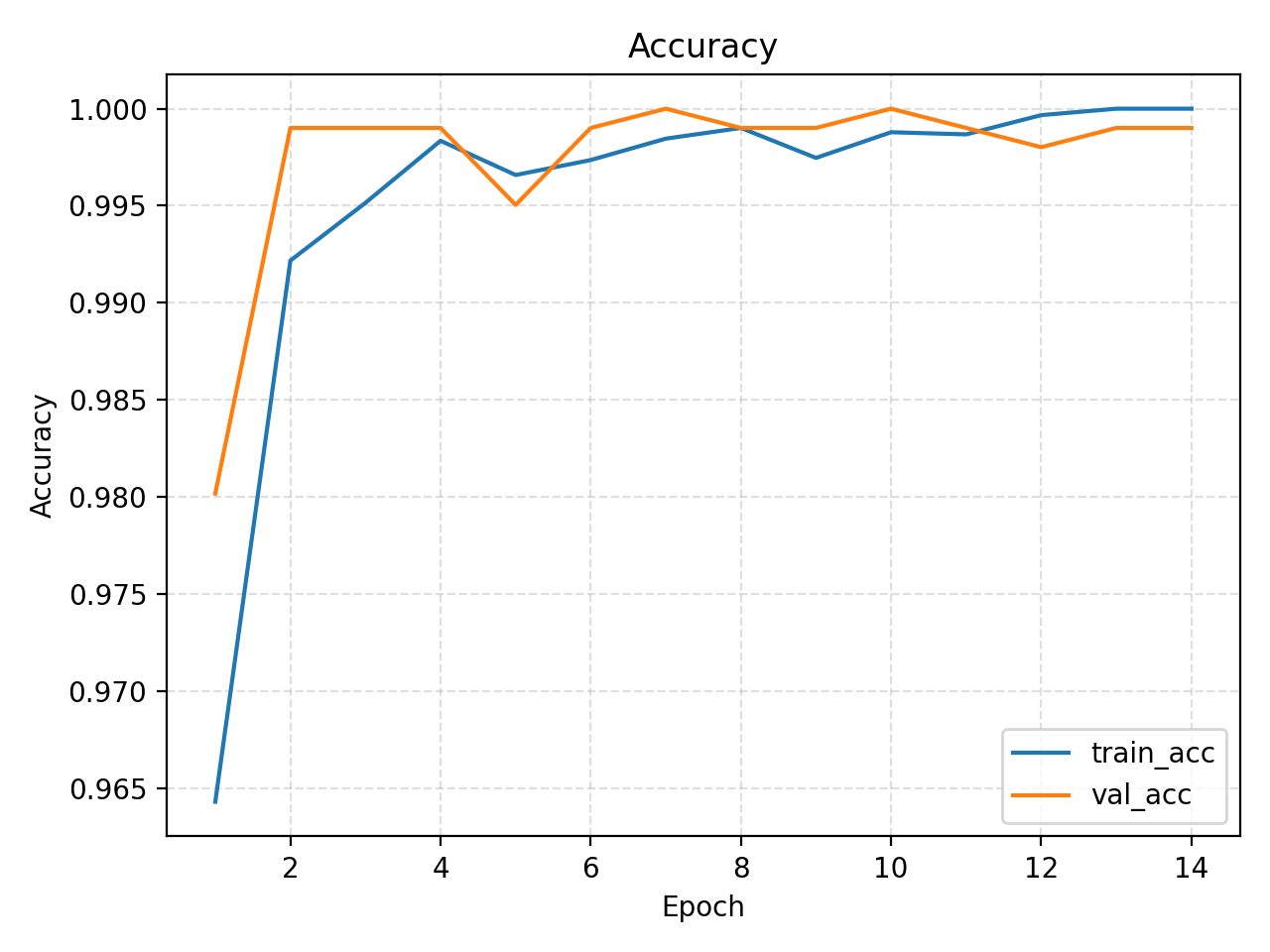




## 5.7. Model 6: resnet34.pt

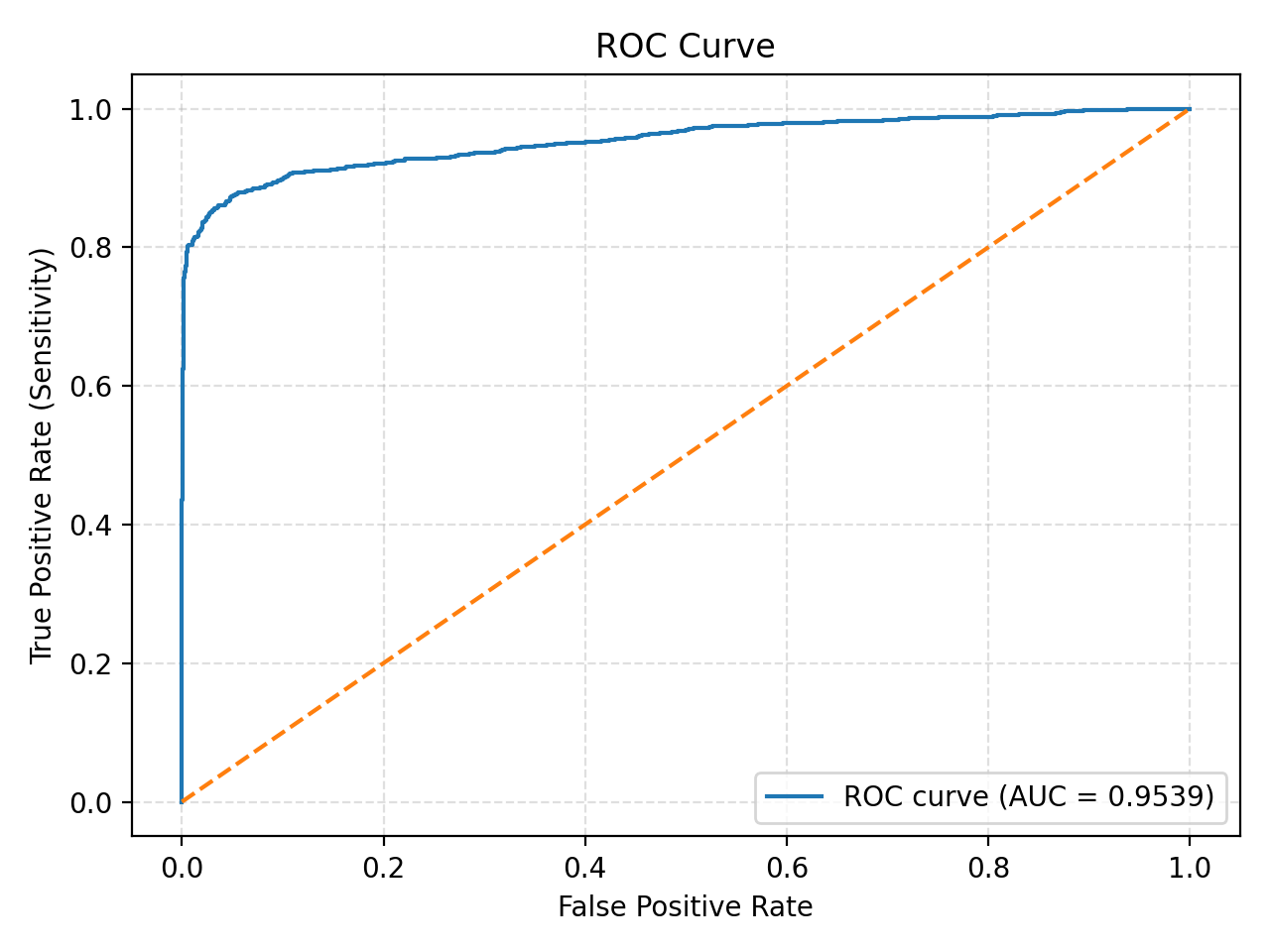
### Sonuç Grafikleri

**



metin, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, çizgi, diyagram içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.



metin, ekran görüntüsü, diyagram, renklilik içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

## 5.8. Model 7: Densenet121.pt

### Sonuç Grafikleri

metin, ekran görüntüsü, çizgi, yazı tipi içeren bir resim

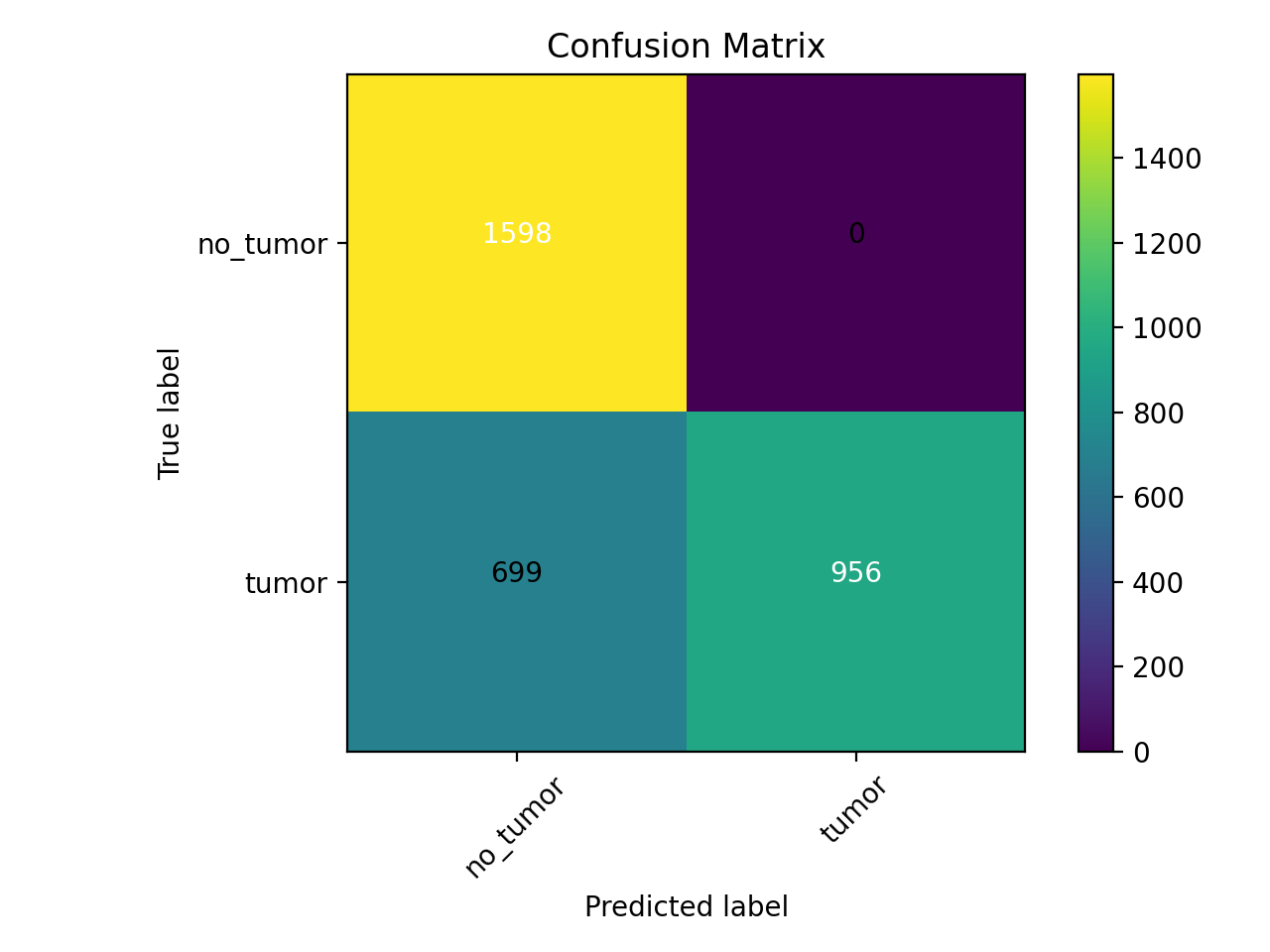
Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, çizgi, diyagram, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, diyagram, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, çizgi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.



metin, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

## 5.9. Model 8: convnext\_tiny.pt

### Sonuç Grafikleri

metin, ekran görüntüsü, yazı tipi, çizgi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, diyagram, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, çizgi, diyagram, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, ekran görüntüsü, diyagram, renklilik içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

## 5.10. Model 9: hybrid\_swinT\_effb0.pt

### Sonuç Grafikleri

metin, ekran görüntüsü, çizgi, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, çizgi, diyagram içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram, çizgi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, ekran görüntüsü, diyagram, çizgi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

## 5.11. Model 10: resnet50.pt

### Sonuç Grafikleri

metin, ekran görüntüsü, çizgi, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, diyagram, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, çizgi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, diyagram, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, çizgi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, ekran görüntüsü, diyagram, renklilik içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

## 5.12. Model 10: inception\_v3

### Sonuç Grafikleri

metin, ekran görüntüsü, çizgi, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, diyagram, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, diyagram, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, çizgi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, ekran görüntüsü, diyagram, renklilik içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

## 5.14. Model 10: efficientnet\_b0.pt

### Sonuç Grafikleri

metin, ekran görüntüsü, çizgi, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, çizgi, diyagram içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

diyagram, çizgi, metin, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, ekran görüntüsü, diyagram, renklilik içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

## 5.15. Model 10: mobilenetv2\_100.pt

### Sonuç Grafikleri

metin, ekran görüntüsü, çizgi, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram, çizgi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram, çizgi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

metin, ekran görüntüsü, diyagram, renklilik içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

# 6. Kaynaklar

Mendeley eğitim veri seti: https://data.mendeley.com/datasets/c9rt8d6zrf/1

Kaggle test veri seti 1: https://www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection

Kaggle test veri seti 2: https://www.kaggle.com/datasets/ahmedhamada0/brain-tumor-detection