MobileNetV2

1. Kullanılan Model Mimarisi

Model, önceden eğitilmiş MobileNetV2 mimarisi kullanılarak transfer öğrenme tekniğiyle oluşturulmuştur. Bu mimaride, son katmanlar çıkarılarak yerine:

- GlobalAveragePooling2D(): Özellik haritalarını sıkıştırmak için kullanılmış,
- Dropout(0.5): Aşırı öğrenmeyi önlemek için,
- Dense(2, activation='softmax'): İki sınıflı (maskeli/maskesiz) sınıflandırma için çıktı katmanı eklenmiştir.

2. Eğitim Süreci ve Kullanılan Metrikler

Model şu şekilde eğitilmiştir:

- Veri artırma (Augmentation) teknikleriyle eğitim verileri çeşitlendirilmiştir: rotation_range, zoom_range, width_shift_range, height_shift_range, shear_range, horizontal_flip ve fill_mode.
- Optimizasyon için Adam optimizasyon algoritması ve başlangıç öğrenme oranı 1e-4 kullanılmıştır.
- Kayıp fonksiyonu: categorical_crossentropy
- Değerlendirme metriği: accuracy

Eğitim süreci boyunca doğruluk ve kayıp değerleri izlenmiştir. Model 20 epoch boyunca eğitilmiştir. Kod sonunda bu metrikler grafiklerle görselleştirilmiştir.

3. Confusion Matrix ve Yorumu

Modelin test verisi üzerinde performansını ölçmek için confusion matrix (karışıklık matrisi) oluşturulmuştur:

Yorum:

- Modelin başarı oranı sadece %50'dir. Bu, modelin sınıflar arasında ayrım yapamadığını göstermektedir.
- Precision ve Recall oranlarının birbirine yakın olması modelin dengeli ama başarısız bir şekilde tahmin yaptığını gösterir.

4. Görsel Tahmin Sonuçları

Model eğitildikten sonra rastgele test görselleri üzerinde tahminler yapılarak görselleştirilmiştir:

- Görseller üzerine gerçek ve tahmin edilen sınıf etiketleri yazılmış.
- Ancak model %50 doğrulukta çalıştığı için birçok görselde yanlış sınıflandırmalar gözlenmiştir.

Bu görselleştirmeler, modelin davranışını analiz etmek için değerli olsa da, modelin düşük başarım oranı nedeniyle karar verme yetisinin zayıf olduğunu ortaya koyar.

5. Karşılaşılan Zorluklar ve Çözüm Yolları

Zorluklar:

- Düşük doğruluk (%50) → Bu, modelin sınıflar arasında hiç ayrım yapamadığını gösterir.
- Veri setinin dengesizliği, kalitesizliği veya azlığı bu durumun başlıca nedenlerinden olabilir.
- Eğitim sürecinde overfitting/underfitting gözlemlenmiş olabilir.

Çözüm Önerileri:

1. **Model Alternatifleri:** EfficientNet, ResNet50 gibi başka önceden eğitilmiş mimariler denenebilir.
2. **Hiperparametre Ayarlamaları:** Öğrenme oranı, dropout oranı, epoch sayısı optimize edilerek yeniden eğitim yapılabilir.
3. **Veri Artırma (Augmentation):** Eğitim verisinin çeşitliliğini artırmak için güçlü augmentasyonlar kullanılabilir

ResNet50

1. Kullanılan Model Mimarisi

Bu çalışmada, ResNet50 mimarisi transfer learning yaklaşımıyla kullanılmıştır. ResNet50, 50 katmandan oluşan ve "skip connection" yapısıyla bilinen güçlü bir konvolüsyonel sinir ağıdır. Modelin üst katmanları dondurularak, son katmanlara yeni fully-connected ve dropout katmanları eklenmiş, böylece önceden öğrenilmiş özellikler korunurken veri kümesine özgü öğrenme gerçekleştirilmiştir.

2. Eğitim Süreci ve Kullanılan Metrikler

Eğitim verisi: 6042 örnek
Test verisi: 1511 örnek
Sınıflar: with_mask, without_mask

Eğitim süreci boyunca kullanılan hiperparametreler:

- Epochs: 20
- Batch size (BS): 32
- Optimizasyon: Adam

Kullanılan metrikler:

- Doğruluk (Accuracy)
- Precision, Recall, F1-score
- Confusion Matrix

Model, eğitim süreci boyunca validation accuracy bakımından dalgalı ancak yaklaşık %50 doğruluk düzeyinde seyretmiş. Bu, modelin sınıflar arasında ayırt edici özellikleri yeterince öğrenemediğine işaret etmektedir.

3. Eğitim Sonuçları (Accuracy / Loss Eğrileri)

Gönderdiğiniz eğitim grafiğine göre:

- **Training accuracy** zamanla artış göstermektedir.
- **Validation accuracy** ise %50 civarında sabitlenmiş ve gelişme göstermemiştir.
- **Validation loss**, training loss'a göre yüksek ve dalgalı kalmıştır.

Bu durum aşırı öğrenme (overfitting) değil, daha çok öğrenememe (underfitting) ya da dengesiz veri/yanlış etiketleme gibi bir probleme işaret etmektedir.

4. Confusion Matrix ve Yorumu

- Her iki sınıf için precision, recall ve f1-score değerleri neredeyse eşit ve %50 civarında.
- Model, with_mask ve without_mask sınıflarını rastgele ayırt ediyor gibi görünmekte.
- Bu performans, modelin sınıflar arası anlamlı farkları öğrenemediğini göstermektedir.

Olası nedenler:

- Veri seti çok benzer görseller içeriyor olabilir.
- Görsellerde maske konumu/türü belirgin değilse, model ayırt etmekte zorlanır.
- Eğitim süreci yetersiz olabilir ya da yanlış katmanlar eğitiliyor olabilir.
- Data augmentation eksikliği.

5. Genel Değerlendirme ve Öneriler

- Modelin karmaşıklığına rağmen doğruluk %50 civarında kalmıştır. Bu, modelin öğrenme gerçekleştiremediği anlamına gelir.
- Eğitim verilerinin kalitesi ve çeşitliliği gözden geçirilmelidir.
- Veri ön işleme ve data augmentation yöntemleriyle modelin genelleme kabiliyeti artırılabilir.
- `rescale`, `rotation`, `zoom`, `horizontal_flip` gibi augmentation teknikleri kullanılmalı.
- Modelin daha fazla epoch ile ve erken durdurma (early stopping) kullanarak eğitilmesi faydalı olabilir.
- Son katmanlar daha karmaşık hale getirilebilir (örneğin: GlobalAveragePooling2D + Dropout + Dense katman kombinasyonu).

İyileştirilmiş MobileNetV2

1. Kullanılan Model Mimarisi

- Temel model: MobileNetV2 (imagenet ağırlıklı)
- include_top=False, trainable=False olarak ilk aşamada dondurulmuştur.

- **Üst Katmanlar:**
  - GlobalAveragePooling2D: Özellik haritasını vektöre dönüştürür.
  - Dense(256) + BatchNormalization + Dropout(0.6)
  - Dense(128) + BatchNormalization + Dropout(0.3)
  - Dense(1, activation='sigmoid'): İkili sınıflama için çıktı katmanı.

2. Eğitim Süreci ve Kullanılan Metrikler

Eğitim Aşamaları:

- İlk eğitimde base_model.trainable = False iken 40 epoch boyunca eğitim gerçekleştirilmiş.
- Daha sonra son 20 katman açılarak düşük öğrenme oranıyla (1e-5) fine-tuning yapılmıştır.

Callback'ler:

- EarlyStopping: Overfitting'i önlemek için kullanılmıştır.
- ReduceLROnPlateau: Validation kaybında ilerleme durduğunda öğrenme oranı azaltılmıştır.
- ModelCheckpoint: En iyi doğrulama performansı elde edilen model .keras formatında kaydedilmiştir.

Performans Metrikleri (Final Test Seti):

- **Test Accuracy:** 99.07%
- **Test Loss:** 0.0247

3. Confusion Matrix ve Sınıflandırma Raporu

	precision	recall	f1-score	support
with_mask	0.50	0.51	0.50	745
without_mask	0.52	0.51	0.52	766
accuracy			0.51	1511

- Elde edilen sonuç, modelin tahmin doğruluğunun aslında çok düşük olduğunu göstermektedir (%51 doğruluk). Bu durum, predict() ve validation_generator.classes'ın uyumsuzluğu ya da validation_generator.reset() sonrasında next() gibi bir işlem yapılmadan direkt kullanılmasından kaynaklanıyor olabilir.

4. Görsel Tahmin Sonuçları

a) test_single_image(...)

- Harici bir görsel (“resim.jpg”) için başarıyla tahmin yapıldı.
- Örnek çıktı:
  - **Tahmin:** without_mask
  - **Güven:** %99.97

b) display_random_test_results(...)

- Test setinden rastgele örnekler alınarak modelin görsel tahminleri sunulmuş.
- Renkli başlıklarla doğru (yeşil) **ve** yanlış (kırmızı) tahminler ayırt edilebilir şekilde görselleştirilmiştir.

5. Karşılaşılan Zorluklar

Zorluk 1: Eğitim Süresinin Uzunluğu

- Özellikle fine-tuning sırasında epoch başına 2-3 dakika eğitim süresi gözlemlenmiştir.
- **Çözüm Önerileri:**
  - Daha küçük batch size veya daha az layer fine-tune etmek.
  - tf.data pipeline ile veri yüklemesini optimize etmek.

Zorluk 2: Veri Seti Uyumsuzluğu veya Boş Klasörler

- Kod içindeki create_train_test_split fonksiyonu bazı durumlarda klasörleri silip yeniden oluşturduğu için dikkatli kullanılmalı.
- **Çözüm:**
  - Eğitim klasörlerinin önceden yedeklenmesi ya da işlem sonunda veri kaybı olmaması için .copy() yerine .move() tercihiyle veri taşımayı kontrol etmek.