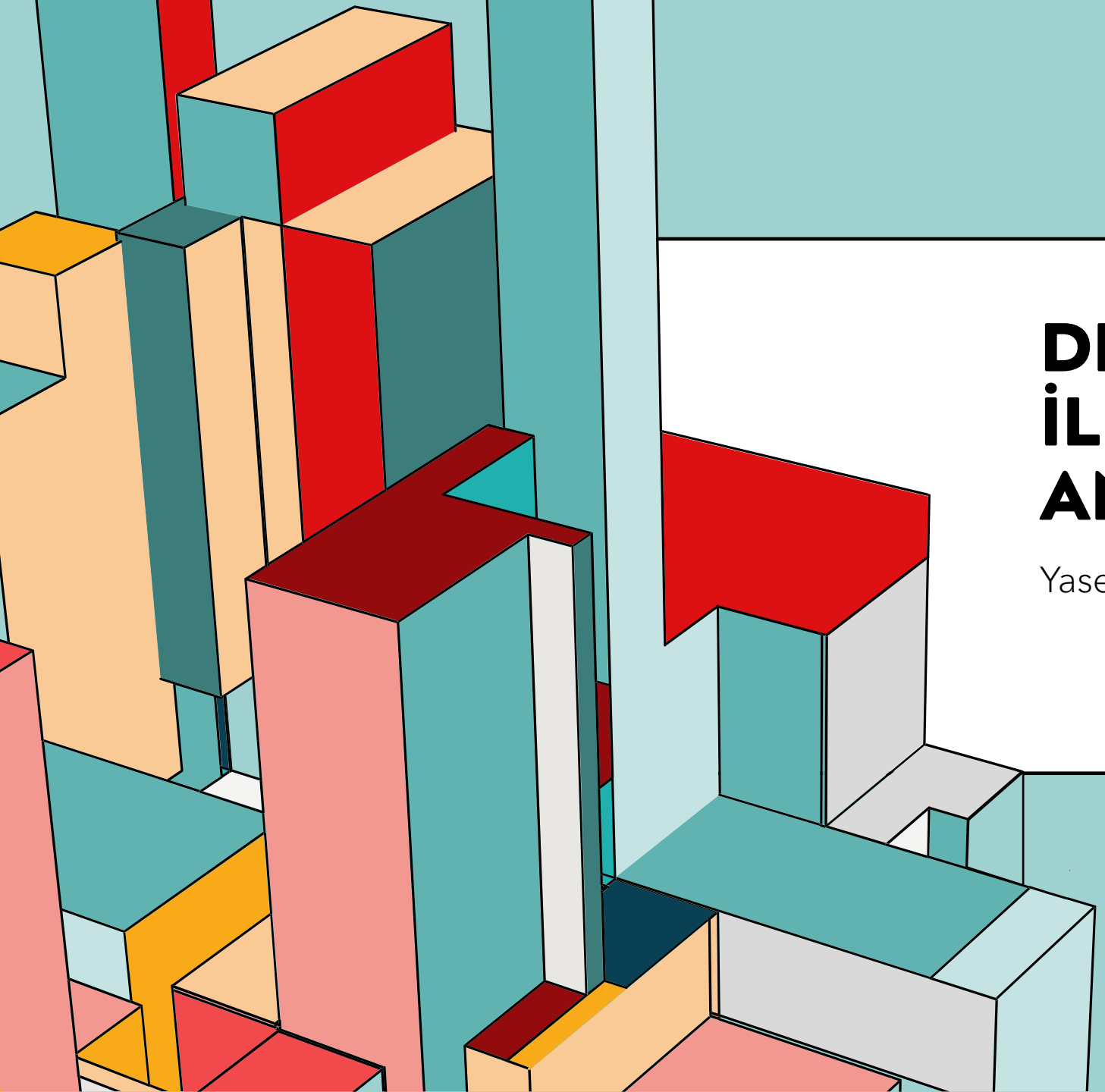
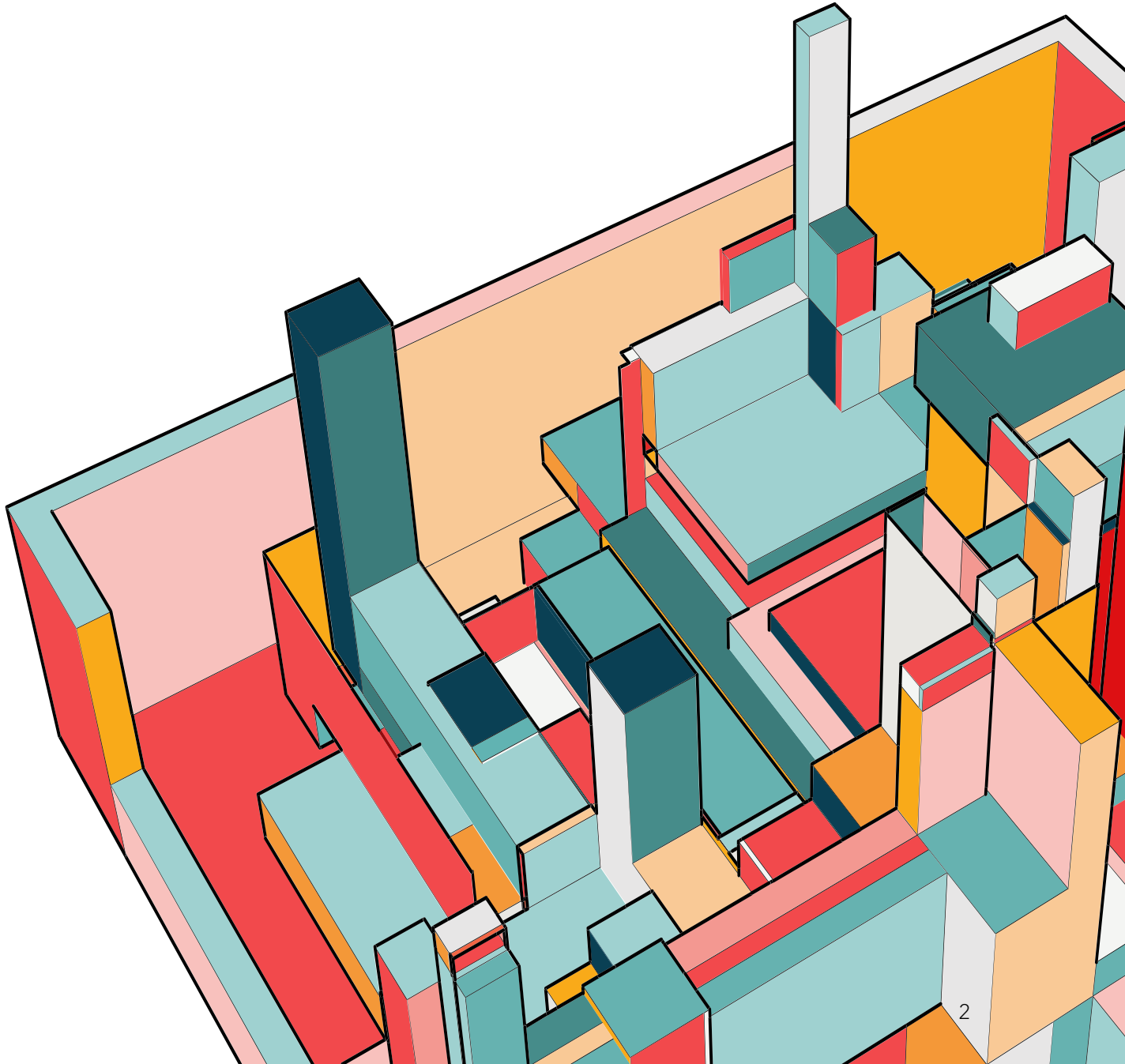
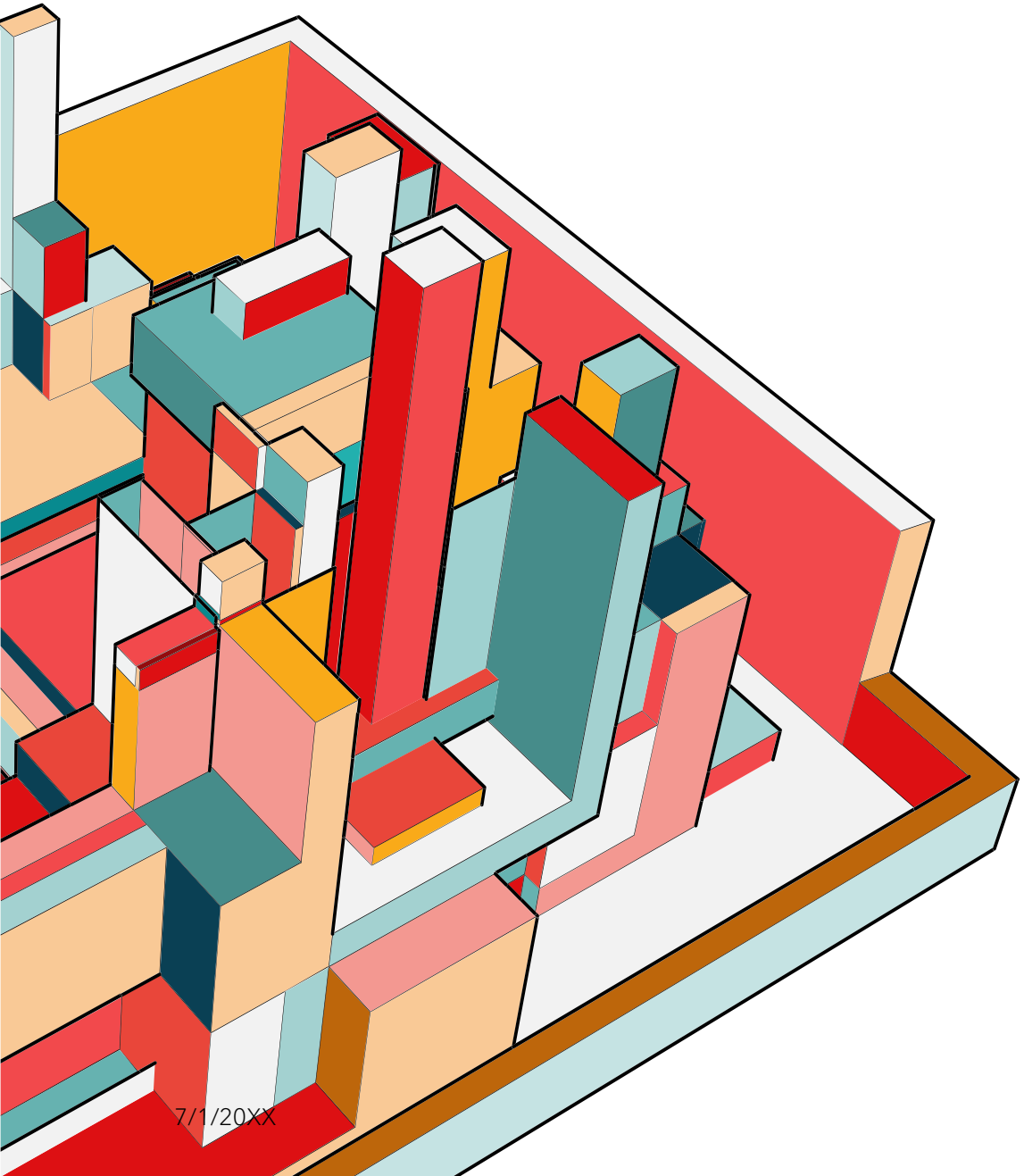


DERİN ÖĞRENME İLE DUYGU ANALİZİ PROJESİ

Yasemin Baskaya

İÇERİK





GİRİŞ

Bu proje, görüntülerdeki duygusal ifadeleri tanımlamak ve sınıflandırmak amacıyla geliştirilmiştir. Görüntülerdeki duygusal ifadeleri belirleme ve bu ifadeleri belirli duygusal kategorilere sınıflandırma süreci olan duygu analizi, bu projenin temel odak noktasıdır. Projemizin hedefleri arasında görüntülerdeki duygusal içerikleri tanımlama, duygu analizi yöntemlerinin doğruluğunu ve performansını değerlendirme, ve duygusal içeriğe sahip görüntülerin anlamını daha iyi anlamak ve kullanıcı deneyimini iyileştirmek yer almaktadır. Bu çalışma, duygu analizindeki güncel yöntemleri ve teknikleri kullanarak duygusal içeriklerin tespit edilmesi ve analiz edilmesi konusunda önemli bir adım oluşturmaktadır.

KULLANILAN ORTAMLAR VE YÖNTEMLER

Kullanılan ortamlar:

Python 3.12.3
Visual Studio Code 1.89.1
Jupyter 24.0

Kullanılan Yöntemler

Bu projede, insanların yüz ifadelerinden duygusal durumlarını tanımlamak amacıyla derin öğrenme yöntemleri kullanılmıştır. Projede kullanılan ana teknik, evrişimli sinir ağları (Convolutional Neural Networks - CNN) ile duygu tanıma modelinin oluşturulmasıdır. Kaggle platformundan elde edilen geniş bir veri seti kullanılmıştır. Bu veri seti, farklı duygusal durumları temsil eden yüz ifadelerinden oluşur. Veri seti önceden eğitim ve test kümelerine bölünmüştür, bu nedenle bölme işlemine gerek kalmamıştır. Her bir görüntü, duygusal durumu temsil eden bir etiketle eşleştirilmiştir. Veri setindeki görüntüler, öncelikle boyutlandırma ve ölçeklendirme işlemlerinden geçirilmiştir. Bu adımların amacı, görüntülerin modelin kabul edebileceği bir formata dönüştürülmesidir. Görüntüler ayrıca gri tonlamalı hale getirilmiş ve 48x48 piksel boyutlarına indirgenmiştir. Görüntülerden özelliklerin çıkarılması aşamasında, evrişimli sinir ağlarının özellik çıkarım yetenekleri kullanılmıştır. Bu adımda, her bir görüntüden elde edilen özelliklerin sayısal bir temsili oluşturulmuştur. Bu temsil, modelin duygusal durumları tanımasına yardımcı olacaktır. Oluşturulan özellikler, bir sinir ağı modeline girdi olarak verilmiştir. Model, evrişimli ve tam bağlantılı katmanlardan oluşan bir mimariye sahiptir. Evrişimli katmanlar, görüntülerdeki özellikleri tanımlamak ve öğrenmek için kullanılırken, tam bağlantılı katmanlar bu özellikleri sınıflandırmak için kullanılmıştır. Model, eğitim veri seti üzerinde eğitilmiş ve ardından test veri seti üzerinde değerlendirilmiştir. Modelin performansı, doğruluk (accuracy), hassasiyet (precision), geri çağırma (recall) ve F1 puanı gibi metriklerle değerlendirilmiştir. Bu metrikler, modelin duygusal ifadeleri doğru bir şekilde sınıflandırma yeteneğini gösterir.

VERİSETİ HAKKINDA

Veriseti Kaynağı

Veriseti, proje konusu belirlendikten sonra kaggle internet sitesinde yapılan tarama sonucu mutlu, kızgın, nötr ve üzgün duyguları projede kullanılmıştır. Ayrıca etiketler Türkçe olarak değiştirilmiştir.

https://www.kaggle.com/datasets/jonathanoheix/face-expression-recognition-dataset

Veriseti Özellikleri

Konu: Yüz ifadeleri tanıma.

Öznitelik Sayısı: Her görüntü, 48x48 piksel boyutlarında gri tonlamalı bir resim olarak temsil edilir, yani her görüntüde 2304 öznitelik bulunmaktadır.

Sınıf Sayısı: 4 (Mutlu, Üzgün, Kızgın, Nötr).

Örnek Sayısı:

Eğitim verisi: Toplamda 21077 görüntü.

Test verisi: Toplamda 5140 görüntü.

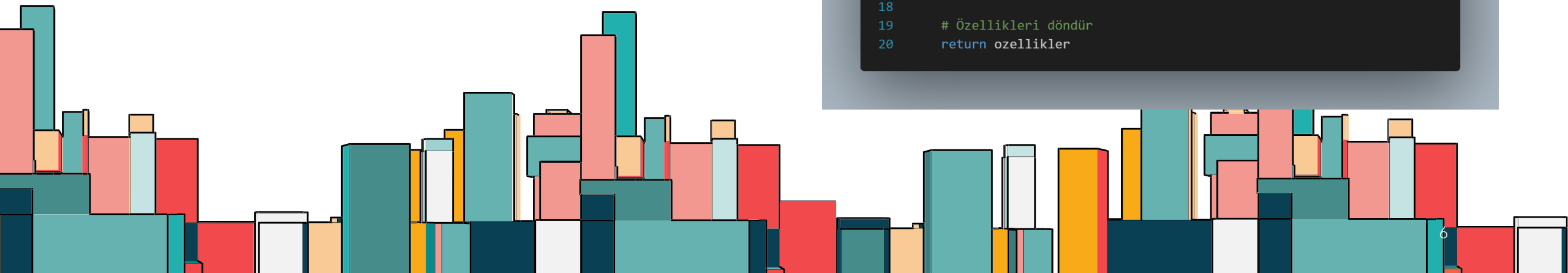
```
Eğitim Verisi Sayısı: 21077
Eğitim Verisi Boyutu: (48, 48, 1)
Eğitim Verisi Örnek Boyutu (Piksel): 48 x 48
Test Verisi Sayısı: 5140
Test Verisi Boyutu: (48, 48, 1)
Test Verisi Örnek Boyutu (Piksel): 48 x 48
```

DENEYDE KULLANILAN MODEL VE MİMARİLER

Veri Hazırlığı

Veri seti, eğitim ve test olarak ikiye ayrılmıştır. Her bir veri seti, duygulara göre etiketlenmiş resimlerden oluşmaktadır. Resimler gri tonlamalı (grayscale) olarak yüklenmiş ve 48x48 boyutunda işlenmiştir

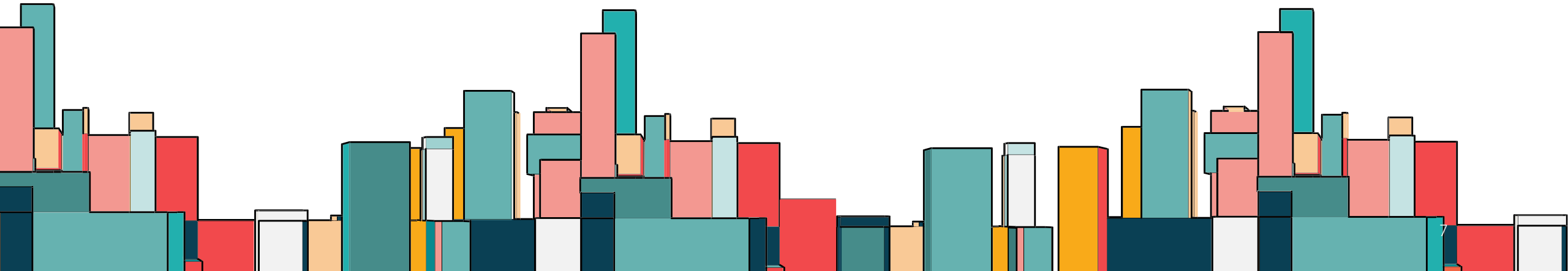
```python
def ozellikleri_cikar(resimler):
    # Özellikleri saklamak için boş bir liste oluştur
    ozellikler = []

    # Her bir resim için ilerleme çubuğunu göster
    for resim in tqdm(resimler):
        # Resmi yükle ve gri tonlamalı olarak al
        yuklenen_resim = load_img(resim, color_mode="grayscale")
        # Resmi numpy dizisine çevir
        yuklenen_resim = np.array(yuklenen_resim)
        # Özellikler listesine ekle
        ozellikler.append(yuklenen_resim)

    # Özellikleri numpy dizisine çevir
    ozellikler = np.array(ozellikler)
    # Özellikleri istenen şekle dönüştür (örneğin: (adet, 48, 48, 1))
    ozellikler = ozellikler.reshape(len(ozellikler), 48, 48, 1)

    # Özellikleri döndür
    return ozellikler
```



DENEYDE KULLANILAN MODEL VE MİMARİLER

Mimari

Giriş Katmanı: Görüntülerin boyutu 48x48 piksel ve tek kanallı (siyah beyaz) olarak belirlenmiştir.
Evrişimli Katmanlar (Conv2D): İlk evrişimli katman, 128 adet 3x3'lük filtre kullanarak görüntülerdeki temel özellikleri (kenarlar, desenler) öğrenir.Ardından gelen MaxPooling2D katmanı, veriyi örnekleyerek boyutunu küçültür ve önemli özellikleri vurgular.Dropout katmanları, aşırı uyumu önlemek için kullanılır. Her evrişimli katmanın ardından %40 oranında rastgele nöronlar devre dışı bırakılır.
Tam Bağlantılı Katmanlar (Dense):Düzleştirme katmanı, evrişimli katmanların çıktılarını düzleştirerek tam bağlantılı katmanlara geçişi sağlar.Modelde iki tam bağlantılı gizli katman bulunur: İlki 512 nöronlu, ikincisi ise 256 nöronlu. Her iki katmanda da aktivasyon fonksiyonu olarak ReLU kullanılır.Bu katmanlar, öğrenilen özellikleri sınıflandırma için kullanır ve aşırı uyumu kontrol etmek için Dropout kullanır.
Çıkış Katmanı:Modelin çıkışı, 4 sınıf için softmax aktivasyonu ile belirlenir. Bu aktivasyon, modelin her bir sınıfa ait olasılık değerlerini verir.



DENEYDE KULLANILAN MODEL VE MİMARİLER

Mimari

modelimiz.summary() ile mimarinin detaylı tablosu yanda verilmiştir.

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 46, 46, 128)	1,280
max_pooling2d (MaxPooling2D)	(None, 23, 23, 128)	0
dropout (Dropout)	(None, 23, 23, 128)	0
conv2d_1 (Conv2D)	(None, 21, 21, 256)	295,168
max_pooling2d_1 (MaxPooling2D)	(None, 10, 10, 256)	0
dropout_1 (Dropout)	(None, 10, 10, 256)	0
conv2d_2 (Conv2D)	(None, 8, 8, 512)	1,180,160
max_pooling2d_2 (MaxPooling2D)	(None, 4, 4, 512)	0
dropout_2 (Dropout)	(None, 4, 4, 512)	0
conv2d_3 (Conv2D)	(None, 2, 2, 512)	2,359,808
max_pooling2d_3 (MaxPooling2D)	(None, 1, 1, 512)	0
dropout_3 (Dropout)	(None, 1, 1, 512)	0
flatten (Flatten)	(None, 512)	0
dense (Dense)	(None, 512)	262,656
dropout_4 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 256)	131,328
dropout_5 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 4)	1,028

Total params: 12,694,286 (48.42 MB)

Trainable params: 4,231,428 (16.14 MB)

Non-trainable params: 0 (0.00 B)

Optimizer params: 8,462,858 (32.28 MB)

DENEYDE KULLANILAN MODEL VE MİMARİLER

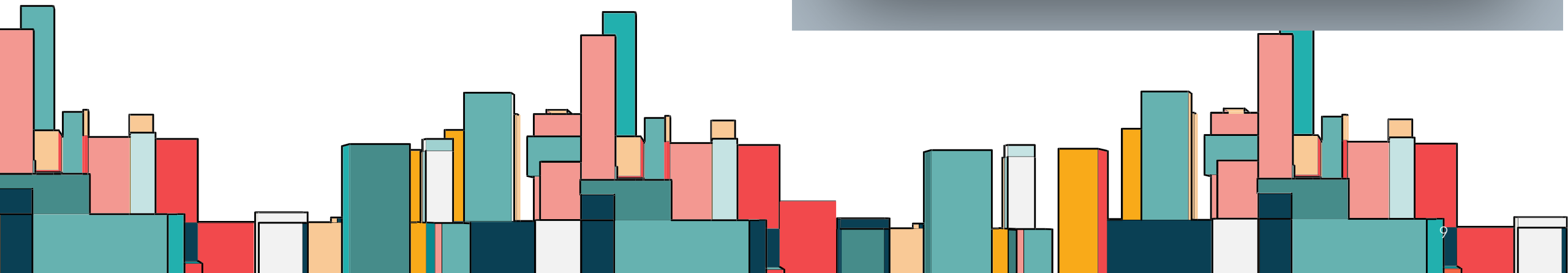
Model

Optimizasyon algoritması olarak Adam seçilir. Adam, adaptif moment tahmini kullanarak eğitimi hızlandırır.

Kayıp (loss) fonksiyonu olarak categorical_crossentropy seçilir. Bu, çok sınıflı sınıflandırma problemleri için yaygın olarak kullanılan bir kayıp fonksiyonudur.

Modelin performansını değerlendirmek için accuracy, recall, precision ve f1_score metrikleri kullanılır.

```
# Modeli Derleme (Compile) Etme
model.compile(
    optimizer='Adam',  # Optimizasyon algoritması olarak Adam'ı kullanır.
    loss='categorical_crossentropy',  # Kayıp fonksiyonu olarak categorical_crossentropy kullanır. Çok sınıflı sınıflandırma problemleri için uygun bir kayıp fonksiyonudur.
    metrics=['accuracy', 'recall', 'precision', 'f1_score']  # Metrikler olarak accuracy, recall, precision ve f1_score kullanır. Birden fazla metric kullanılması modelin değerlendirilebilmesi açısından daha sağlıklı olur.
)
```



DENEY SONUÇLARI

- Eğitim ve test setlerindeki modelin performansı aşağıdaki gibi ölçülmüştür:

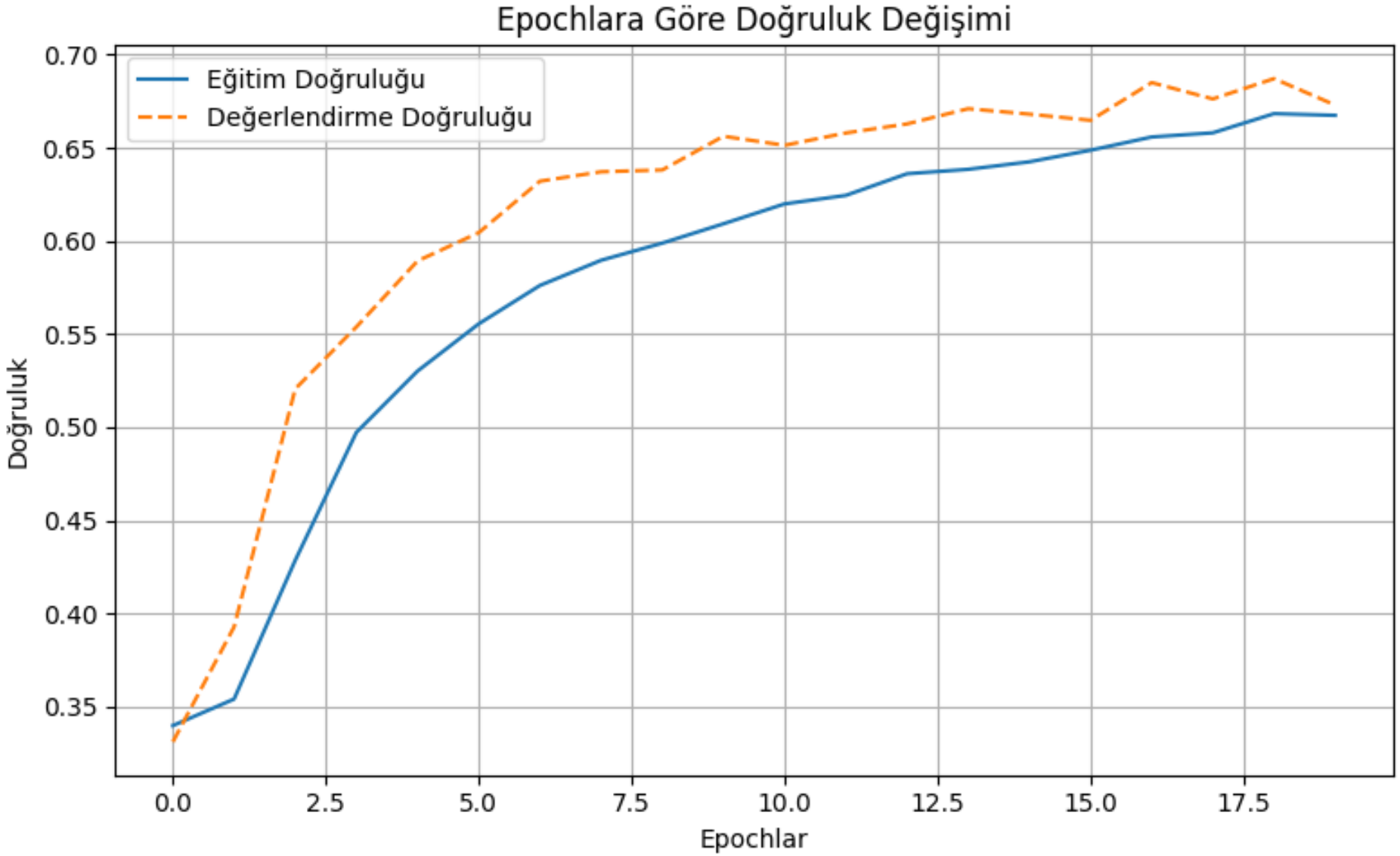
```
egitim_dogruluk = model.evaluate(x= x_train,y = y_train)
test_dogruluk = model.evaluate(x= x_test, y= y_test)
print('Eğitim: %.3f, Test: %.3f' % (egitim_dogruluk[0], test_dogruluk[0]))
```

- Ayrıca model eğitimi «history» değişkenine atandığı için her epoch sonucunda metriklerdeki değişim grafiklere yansıyabilmektedir.

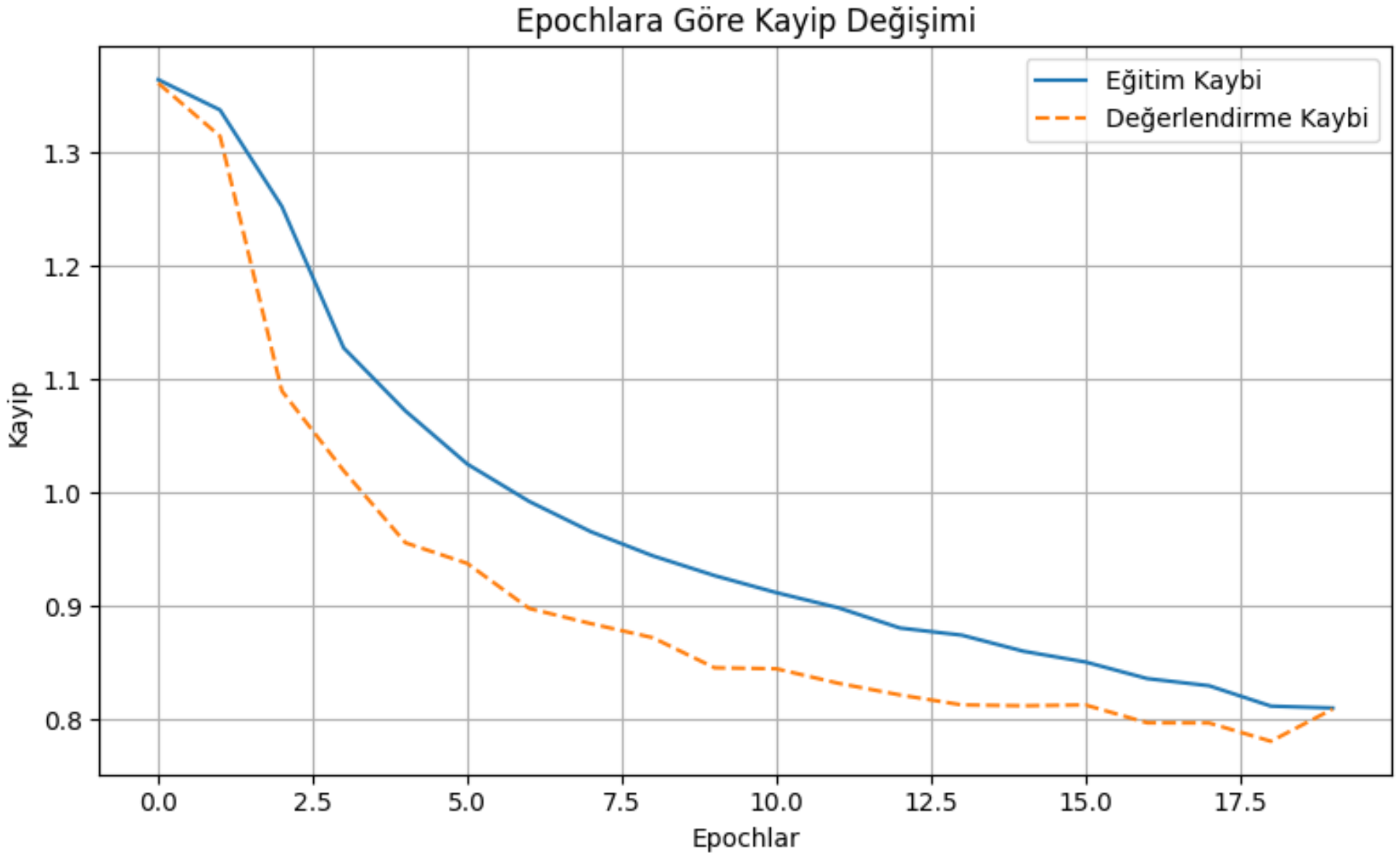
```
# Modelin Eğitimi
history = model.fit(
    x=x_train,  # Eğitim verileri
    y=y_train,  # Eğitim etiketleri
    batch_size=128,  # Mini-batch boyutu (her iterasyonda işlenecek örnek sayısı)
    epochs=20,  # Eğitim epoch sayısı (kaç kez tüm veriler model tarafından kullanılacak)
    validation_data=(x_test, y_test)  # Modelin doğruluğunu değerlendirmek için kullanılan doğrulama verileri
)

# Modelin eğitimine başlamak için x_train, y_train verilerini kullanır. Her iterasyonda 128 örnek işlenir ve toplamda 20 epoch boyunca eğitilir. Modelin doğruluğunu değerlendirmek için x_test, y_test verilerini kullanır.

```

```
659/659 ━━━━━━━━━━━━━━━━━━━━ 59s 90ms/step - accuracy: 0.6986 - f1_score: 0.4601 - loss: 0.7422 - precision: 0.8013 - recall: 0.5877
161/161 ━━━━━━━━━━━━━━━━━━━━ 15s 95ms/step - accuracy: 0.6186 - f1_score: 0.4106 - loss: 0.9371 - precision: 0.7094 - recall: 0.5197
Eğitim: 0.657, Test: 0.809
```

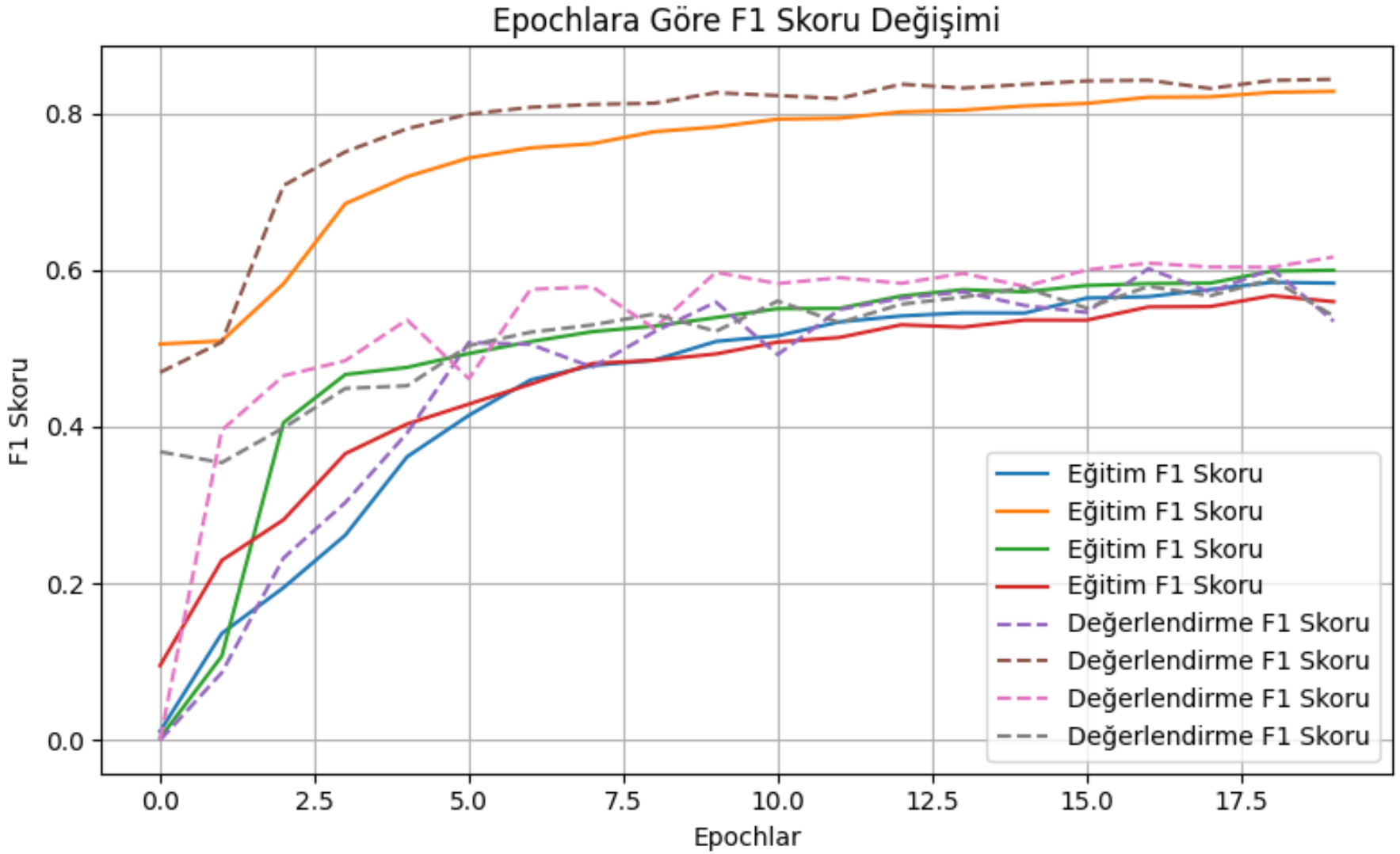
DENEY SONUÇLARI: DOĞRULUK



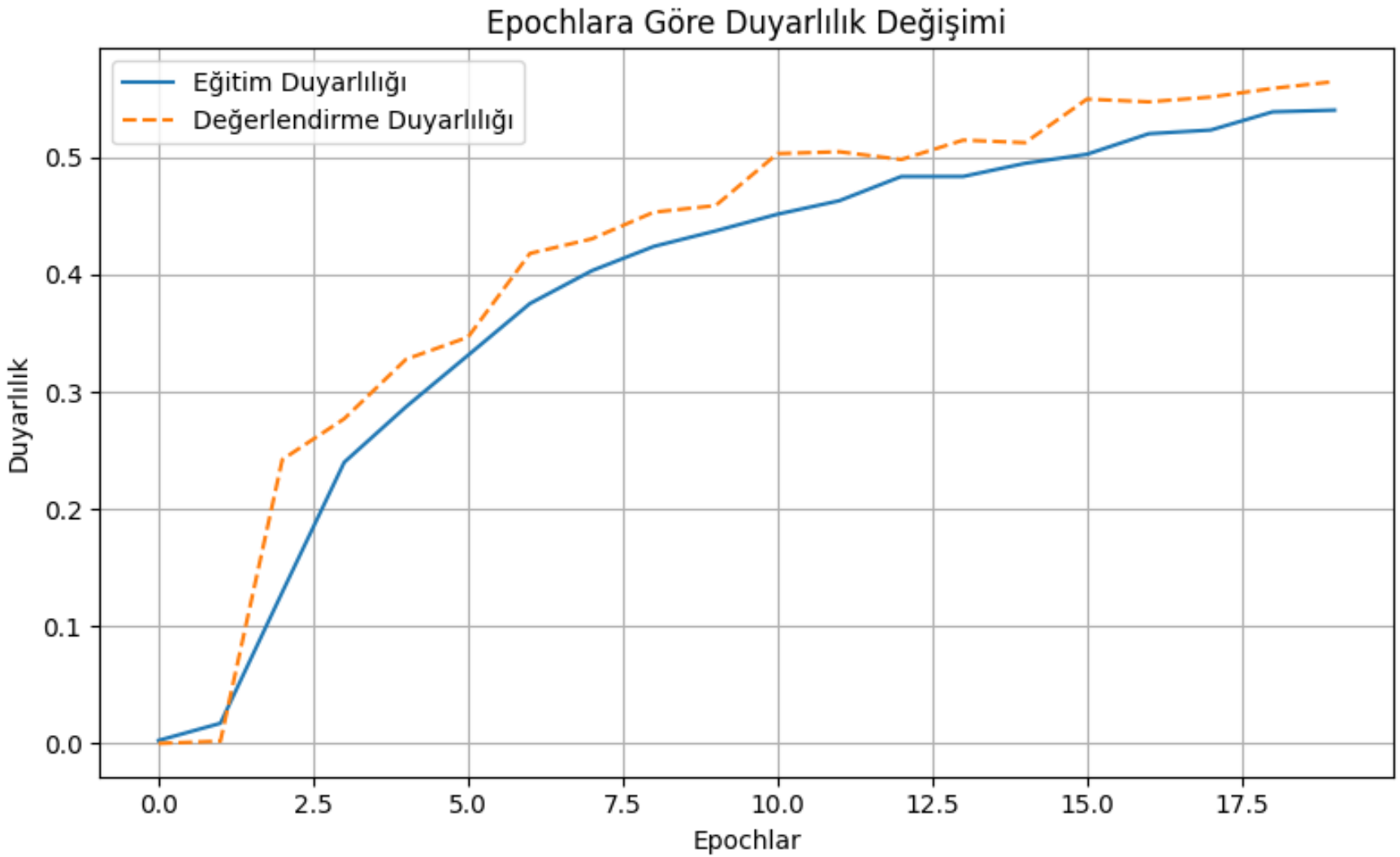
DENEY SONUÇLARI: KAYIP



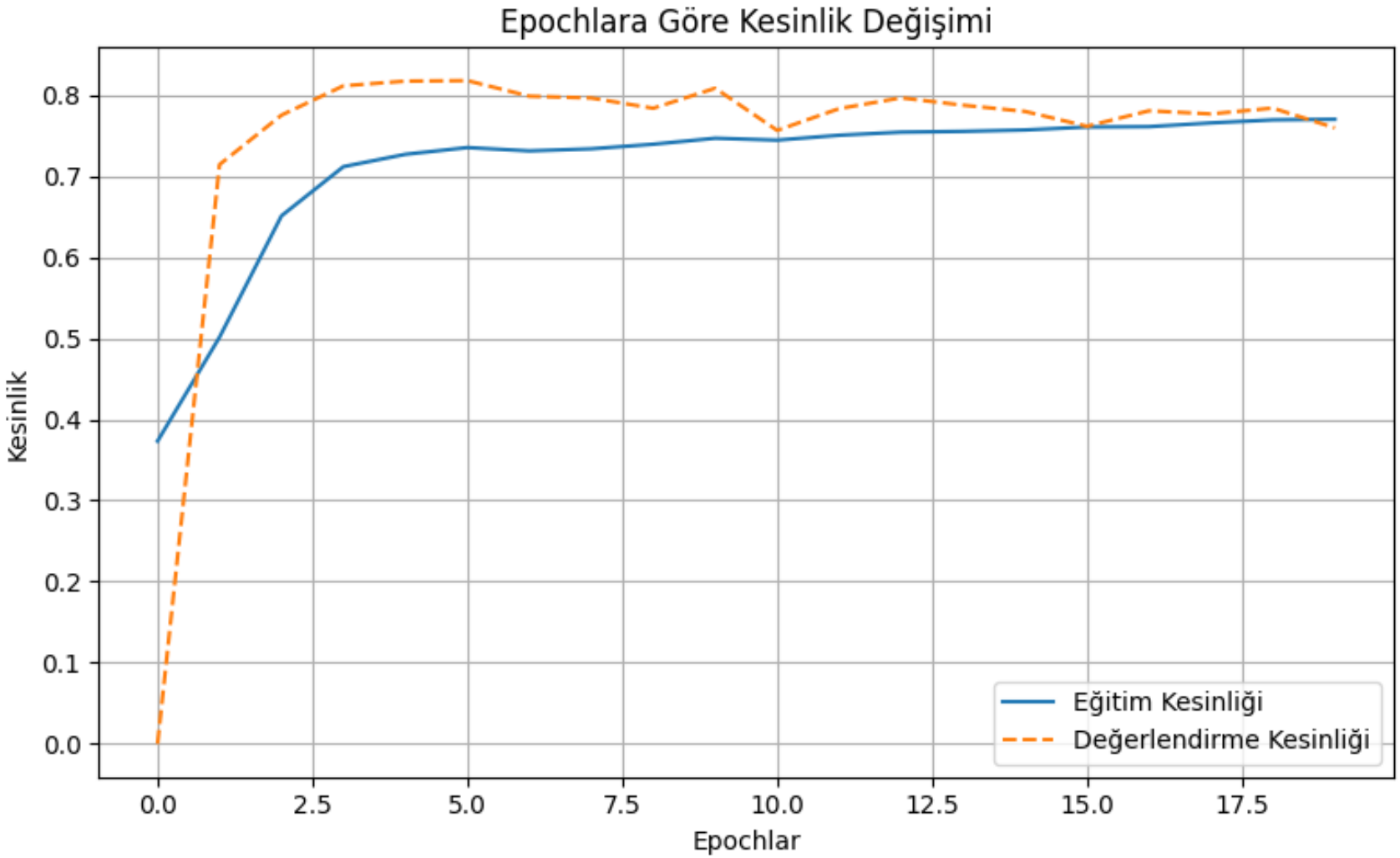
DENEY SONUÇLARI: F1 SKORU



DENEY SONUÇLARI: DUYARLILIK



DENEY SONUÇLARI: KESİNLİK



DENEY SONUÇLARI

1.Accuracy (Doğruluk):

- Eğitim doğruluğu (accuracy) ilk epochta %33.77'den başlayarak, son epochta %67.37'ye kadar yükselmiştir.
- Değerlendirme doğruluğu ise başlangıçta %33.11'den başlayıp, son epochta %67.30'a kadar yükselmiştir.

2.F1-Score:

1. Eğitim F1 skoru (f1_score) ilk epochta 0.1482 iken, son epochta 0.6466'ya yükselmiştir.
2. Değerlendirme F1 skoru başlangıçta 0.2094 olup, son epochta 0.6340 olmuştur.

3.Loss (Kayıp):

1. Eğitim kaybı (loss) başlangıçta 1.3725 olup, son epochta 0.7968'e düşmüştür.
2. Değerlendirme kaybı başlangıçta 1.3616 iken, son epochta 0.8090'a düşmüştür.

4.Precision (Kesinlik) ve Recall (Duyarlılık):

1. Eğitim kesinliği ve duyarlılığı genel olarak artış göstermiştir.
2. Değerlendirme kesinliği ve duyarlılığı da epochlar boyunca iyileşme göstermiştir.

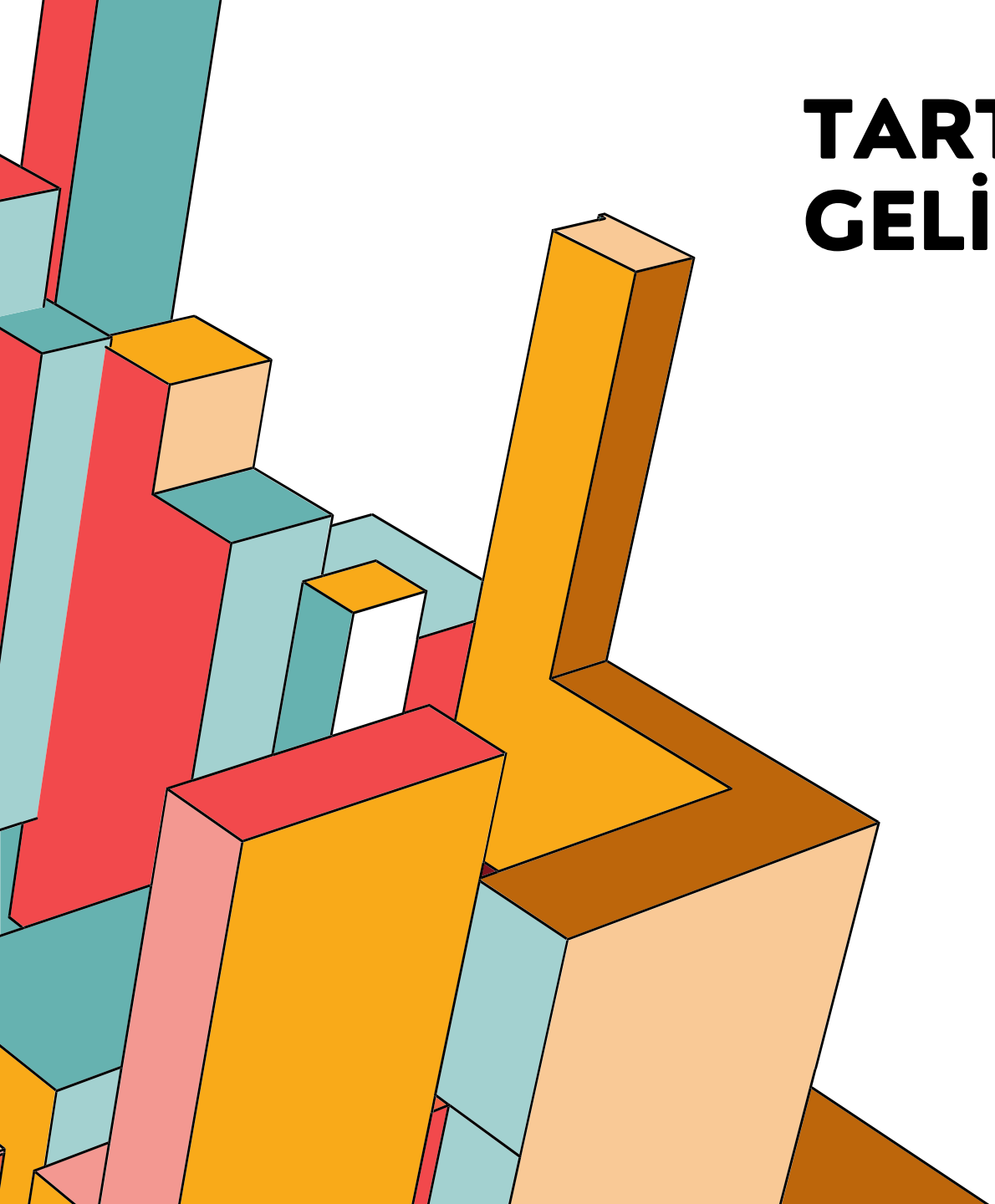
TARTIŞMA

İyi Yönler:

1. Genel İyileşme: Eğitim ve doğrulama metrikleri epoklar boyunca düzenli olarak iyileşmiştir. Bu, modelin öğrenme sürecinde başarıya ulaştığını göstermektedir.
2. Denge: Eğitim ve doğrulama metrikleri birbirine yakın değerler göstererek modelin aşırı uyum (overfitting) yapmadığını göstermektedir.
3. F1-Score: F1 skoru, hem kesinlik hem de duyarlılığı dikkate aldığı için daha dengeli bir performans ölçütüdür ve bu metrikteki artış, modelin genel olarak daha iyi bir performans sergilediğini göstermektedir.

Eksik Yönler:

1. Başlangıç Performansı: İlk birkaç epokta, modelin doğrulama metrikleri oldukça düşük kalmıştır. Bu, modelin başlangıçta veri dağılımını anlamakta zorlandığını göstermektedir.
2. Val Precision ve Recall: İlk epoklarda doğrulama kesinliği ve duyarlılığı sıfıra yakın değerler göstermiştir. Bu, modelin başlangıçta sınıfları doğru bir şekilde ayırt edemediğini göstermektedir.
3. Eğitim Süresi: Her bir epokun tamamlanma süresi oldukça uzundur. Bu, modelin eğitim sürecinin optimize edilmesi gerektiğini gösterebilir.



TARTIŞMA: GELİŞTİRİLEBİLECEK YÖNLER

Daha Fazla Veri:

Eğitim verisinin artırılması, modelin daha iyi genelleştirme yapmasına yardımcı olabilir.

Özellik Çıkarımını Geliştirme

Özelliklerin daha iyi çıkarılması ve dönüştürülmesi modelin performansını artırabilir.

Hiperparametre Optimizasyonu:

Öğrenme oranı, epok sayısı, mini-batch boyutu gibi hiperparametrelerin optimize edilmesi model performansını daha da artırabilir.

Veri Ön İşleme:

Veri ön işleme adımlarının gözden geçirilmesi, normalizasyon ve veri artırma (data augmentation) gibi tekniklerin uygulanması, modelin performansını artırabilir.

REFERANSLAR

- https://www.kaggle.com/datasets/jonathanoheix/face-expression-recognition-dataset
- https://www.geeksforgeeks.org/introduction-convolution-neural-network/
- https://www.datacamp.com/tutorial/introduction-to-convolutional-neural-networks-cnns

TEŞEKKÜRLER

Yasemin BASKAYA

yasemin.baskaya19@gmail.com

https://www.linkedin.com/in/yasemin-baskaya-38b132256/