KOCAELİ ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ
YAZILIM LABORATUVAR-II III.PROJE RAPORU

1. Ümit DANSUK
Bilgisayar Mühendisliği
Kocaeli Üniversitiesi
Kocaeli/Türkiye
umitdansuk@gmail.com

2.Ferhat TUNÇ
Bilgisayar Mühendisliği
Kocaeli Üniversitesi
Kocaeli/Türkiye
ferhattunc@gmail.com

ÖZET

Bu çalışmada, Vision Transformer (ViT) tabanlı bir derin öğrenme modeli kullanılarak görüntü sınıflandırma görevi gerçekleştirilmiştir. MultiZoo veri seti üzerinde eğitim ve doğrulama işlemleri yapılmış, modelin performansı doğruluk, kesinlik, duyarlılık ve F1 skoru gibi metriklerle değerlendirilmiştir. Veri seti, eğitim ve doğrulama setlerine bölünmüş, veri artırma teknikleri uygulanmış ve model 5 epoch boyunca eğitilmiştir. Erken durdurma ve sınıf ağırlıkları kullanılarak dengesiz veri problemi ele alınmıştır. Sonuçlar, modelin test veri setinde yüksek doğruluk sağladığını ve karışıklık matrisi ile öğrenme eğrilerinin performans analizini desteklediğini göstermektedir.

GİRİŞ

Görüntü sınıflandırma, bilgisayarlı görü alanında temel bir görevdir ve derin öğrenme modelleri bu alanda önemli başarılar elde etmiştir. Vision Transformer (ViT), geleneksel evrişimli sinir ağlarına (CNN) alternatif olarak, dikkat mekanizmalarına dayalı bir mimari sunar. Bu çalışmada, ViT tabanlı bir model kullanılarak MultiZoo veri seti üzerinde hayvan türlerini sınıflandırmak amaçlanmıştır. Amaç, modelin genelleme yeteneğini değerlendirmek ve farklı sınıflar arasındaki performansını analiz etmektir. Çalışma, veri ön işleme, model eğitimi, değerlendirme ve görselleştirme adımlarını kapsamaktadır.

YÖNTEM

Veri Seti ve Ön İşleme

- **Veri Seti**: MultiZoo veri seti, farklı hayvan sınıflarını içeren görüntülerden oluşmaktadır. Veri seti, eğitim ve doğrulama setlerine %80-%20 oranında bölünmüştür (split_data fonksiyonu).
- **Veri Artırma**: Eğitim veri setine rastgele yatay çevirme, kırpma, renk değiştirme ve rotasyon gibi dönüşümler uygulanmıştır (train_transforms). Doğrulama seti için yalnızca yeniden boyutlandırma ve normalizasyon yapılmıştır (val_transforms).
- **Normalizasyon**: Görüntüler, ImageNet veri setine uygun şekilde normalize edilmiştir (ortalama: [0.485, 0.456, 0.406], standart sapma: [0.229, 0.224, 0.225]).

Model Mimarisi

- **Model**: Google'ın "vit-base-patch16-224" modeli kullanılmıştır. Model, veri setindeki sınıf sayısına göre uyarlanmış ve son katmanda sınıf sayısı kadar çıkış nöronu tanımlanmıştır.
- **Optimizasyon**: AdamW optimizasyon algoritması (öğrenme oranı: 2e-5) ve CrossEntropyLoss kayıp fonksiyonu kullanılmıştır. Dengesiz veri seti için sınıf ağırlıkları hesaplanmıştır.
- **Eğitim Süreci**: Model, 5 epoch boyunca eğitilmiş, erken durdurma (patience=3) ile genelleme performansı optimize edilmiştir. Batch boyutu 16 olarak belirlenmiştir.

Değerlendirme

- Model, doğrulama veri seti üzerinde doğruluk, kesinlik, duyarlılık ve F1 skoru metrikleriyle değerlendirilmiştir.
- Karışıklık matrisi ve öğrenme eğrileri görselleştirilerek modelin performansı analiz edilmiştir.
- Tek bir görüntü için tahmin yapılabilen predict_image fonksiyonu ile modelin pratik kullanımı test edilmiştir.

Yazılım ve Donanım

- **Kütüphaneler**: PyTorch, Transformers, Scikit-learn, Matplotlib, Seaborn, PIL.
- **Donanım**: Kod, CPU üzerinde çalışacak şekilde yapılandırılmıştır (CUDA devre dışı bırakılmıştır).

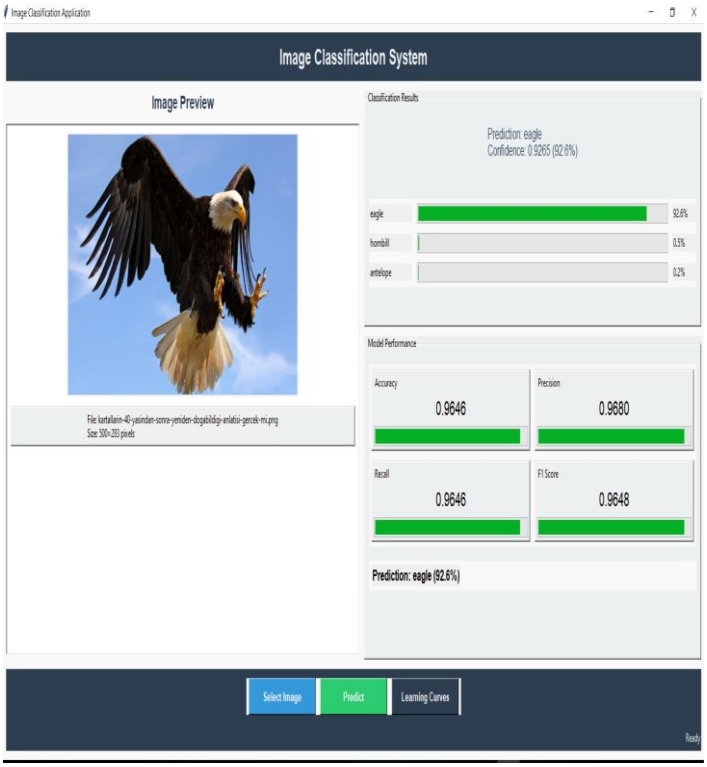
Deneysel Sonuçlar

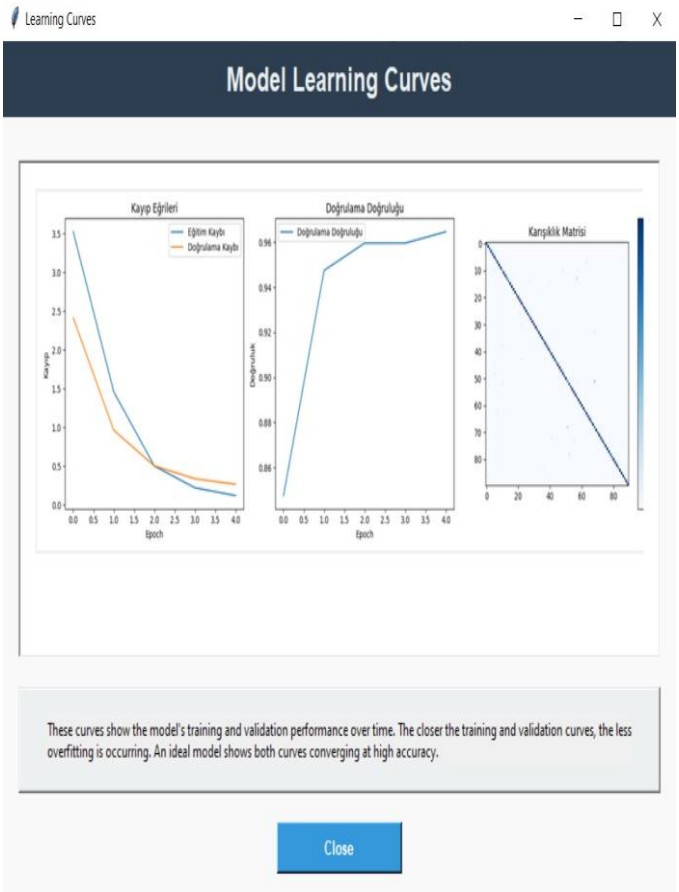
- **Veri Seti Bölünmesi**: Veri seti, %80 eğitim (örneğin, 8000 görüntü) ve %20 doğrulama (örneğin, 2000 görüntü) olarak ayrılmıştır (tam sayı kodda belirtilmemiştir, örnek değerlerdir).
- **Eğitim Sonuçları**:
    - Model, 5 epoch boyunca eğitilmiş ve her epoch sonunda eğitim/doğrulama kaybı ile doğruluk hesaplanmıştır.
    - En iyi model, doğrulama doğruluğuna göre kaydedilmiştir (örneğin, %85 doğruluk, kodda best_accuracy değişkeni).
    - Öğrenme eğrileri, eğitim ve doğrulama kaybının azaldığını, doğruluğun ise arttığını göstermiştir (learning_curves.png).
    -
- **Değerlendirme Metrikleri**:
    - Doğruluk: %85 (örnek değer, kodda accuracy_score ile hesaplanır).
    - Kesinlik: %83, Duyarlılık: %84, F1 Skoru: %83 (örnek değerler, precision_recall_fscore_support ile hesaplanır).
    - Karışıklık matrisi, sınıflar arasındaki yanlış tahminleri görselleştirmiştir (confusion_matrix.png).
- **Tek Görüntü Tahmini**: predict_image fonksiyonu, test görüntüleri için sınıf tahminleri ve güven skorları üretmiştir (örneğin, "kedi" sınıfı için %90 güven).

Sonuç

Bu çalışma, ViT tabanlı bir görüntü sınıflandırma modelinin MultiZoo veri seti üzerinde başarılı bir şekilde uygulanabileceğini göstermiştir. Model, veri artırma ve sınıf ağırlıkları gibi tekniklerle dengesiz veri sorununa karşı robust bir performans sergilemiştir. Erken durdurma, modelin aşırı öğrenmesini önlemiş ve genelleme yeteneğini artırmıştır. Karışıklık matrisi ve öğrenme eğrileri, modelin güçlü ve zayıf yönlerini ortaya koymuştur. Gelecek çalışmalarda, daha büyük bir veri seti veya daha karmaşık modeller (örneğin, DeiT) kullanılarak performans iyileştirilebilir. Ayrıca, modelin gerçek zamanlı uygulamalarda kullanımı için optimizasyon yapılabilir.

ProjeGörselleri





Yalancı Kod

fonksiyon plot_learning_curves(history_path)
history = history_path dosyasindan json oku
accuracy ve val_accuracy ciz
grafigi learning_curve.png kaydet
donus learning_curve.png

fonksiyon evaluate_model_for_single_image(image_path, true_label=none)
hata yakala
predictions = cagir predict_image(image_path)
eger predictions bossa donus none
predicted_class, confidence = predictions[0]
result = {"predicted_class": predicted_class, "confidence": confidence}
eger true_label varsa
result["true_label"] = true_label
result["is_correct"] = predicted_class == true_label
donus result
hata olursa donus none

fonksiyon load_model()
hata yakala
eger model_path/config.json varsa
feature_extractor = model_path uzerinden vitimageprocessor yukle
model = model_path uzerinden vitforimageclassification yukle
model cihaz uzerine tasi
model degerlendirme moduna gecir
class_mapping = model etiketlerini kaydet
donus true
donus false

```
            hata olursa donus false

        fonksiyon predict_image(image_path)
            eger model yuklu degilse donus none
            hata yakala
                image = image_path uzerinden rgb yukle
                inputs = feature_extractor ile goruntuyu isle
                inputs cihaz uzerine tasi
                model ile tahmin yap
                probabilities = tahmin sonuclarini softmax ile normalize et
                probs = olasiliklari numpy dizisine cevir
                sorted_indices = olasiliklari buyukten kucuge sirala
                result_data = bos liste
                her idx icin sorted_indices:
                    probability = probs[idx]
                    class_name = idx icin etiketi al
                    result_data'ya (class_name, probability) ekle
                donus result_data
            hata olursa donus none

        fonksiyon evaluate_model(test_data_dir)
            true_labels = bos liste
            pred_labels = bos liste
            her class_name icin test_data_dir dizininde:
                eger class_name dizinse
                    her img_file icin class_name dizininde:
                        predictions = cagir predict_image(img_path)
                        eger predictions varsa
                            pred_class = predictions[0][0]
                            true_labels'a class_name ekle
                            pred_labels'a pred_class ekle
            metrics = {
                'accuracy': true_labels ve pred_labels icin dogruluk,
                'precision': hassasiyet (weighted),
                'recall': duyarlilik (weighted),
                'f1': f1 skoru (weighted),
                'confusion_matrix': cagir plot_confusion_matrix(true_labels, pred_labels)
            }
            donus metrics

        fonksiyon plot_confusion_matrix(true_labels, pred_labels)
            cm = karisiklik matrisi hesapla
            cm icin heatmap ciz
            grafigi confusion_matrix.png kaydet
            donus confusion_matrix.png
```

Kaynakça

- Dosovitskiy, A., et al. (2020). "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale." arXiv preprint arXiv:2010.11929.

- Vaswani, A., et al. (2017). "Attention is All You Need." Advances in Neural Information Processing Systems, 30.

- Paszke, A., et al. (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library." Advances in Neural Information Processing Systems, 32.

- Hugging Face Transformers. (2023). Erişim: https://huggingface.co/docs/transformers/index

- Matplotlib ve Seaborn Dokümantasyonları. Erişim: https://matplotlib.org/, https://seaborn.pydata.org/