

**MARMARA ÜNİVERSİTESİ**

**TEKNOLOJİ FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**BİTİRME PROJESİ**

OCR ve NER ile Faturaların Sınıflandırılması

**PROJE YAZARLARI**

171421004 Gürkan Çelen

170421020 İsmail Mert Akpınar

170421015 Ahmet Said Kılıç

171421001 Yunus Alp Turan

**DANIŞMAN**

Doç. Dr. Ayşe Berna ALTINEL GİRGİN

**İSTANBUL, 2025**



**MARMARA ÜNİVERSİTESİ**

**TEKNOLOJİ FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**BİTİRME PROJESİ**

OCR ve NER ile Faturaların Sınıflandırılması

**PROJE YAZARLARI**

171421004 Gürkan Çelen

170421020 İsmail Mert Akpınar

170421015 Ahmet Said Kılıç

171421001 Yunus Alp Turan

**DANIŞMAN**

Doç. Dr. Ayşe Berna ALTINEL GİRGİN

**İSTANBUL, 2025**

**MARMARA ÜNİVERSİTESİ**

**TEKNOLOJİ FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

Marmara Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği Öğrencileri Gürkan Çelen, İsmail Mert Akpınar, Yunus Alp Turan ve Ahmet Said Kılıç ‘ın “OCR ve NER ile Faturaların Sınıflandırılması” başlıklı bitirme projesi çalışması, 19/06/2025 tarihinde sunulmuş ve jüri üyeleri tarafından başarılı bulunmuştur.

**Jüri Üyeleri**

Doç. Dr. Ayşe Berna ALTINEL GİRGİN (Danışman)

Marmara Üniversitesi ......................................................................... (İMZA) ..................

Dr. Öğr. Üyesi Neşe ÖZDEMİR (Üye)

Marmara Üniversitesi ......................................................................... (İMZA) ..................

**İÇİNDEKİLER**

[**ÖZET** 5](#_Toc201154683)

[**1.** **GİRİŞ** 6](#_Toc201154684)

[1.1. Proje Tanımı ve Amacı 7](#_Toc201154685)

[1.2 Literatür Özeti 8](#_Toc201154686)

[1.3 İlgili Çalışmalar ve Mevcut Uygulamalar 10](#_Toc201154687)

[2. MATERYAL VE YÖNTEM 11](#_Toc201154688)

[2.1. Sistem Mimarisi ve Teknoloji Yığını 11](#_Toc201154689)

[2.1.1. Genel Mimari Yapı 12](#_Toc201154690)

[2.1.2. Kullanılan Teknoloji Yığını 13](#_Toc201154691)

[*2.1.3. Uygulama İşlem Adımları* 13](#_Toc201154692)

[2.2. Yapay Zekâ Modülleri ve Görevleri 14](#_Toc201154693)

[2.3. Kullanılan Veri Setleri 16](#_Toc201154694)

[2.4. Uygulama Adımları ve Akış Diyagramı 18](#_Toc201154695)

[2.5. Kullanılan Kütüphane ve Araçlar 21](#_Toc201154696)

[**3. BULGULAR VE TARTIŞMA** 23](#_Toc201154697)

[3.1. Model Performans Değerlendirmesi 23](#_Toc201154698)

[3.1.5. İşlem Süresi ve Performans 26](#_Toc201154699)

[3.2. Örnek Çıktılar ve Görsel Sonuçlar 26](#_Toc201154700)

[3.3. Hata Analizi ve Karşılaşılan Zorluklar 28](#_Toc201154701)

[3.4. Sistem Başarımı ve Gerçek Yaşam Uygunluğu 30](#_Toc201154702)

[**4. SONUÇ VE ÖNERİLER** 32](#_Toc201154703)

[4.1. Sonuçlar 32](#_Toc201154704)

[4.2. Öneriler 32](#_Toc201154705)

[**5. KAYNAKLAR** 33](#_Toc201154706)

**SEMBOLLER/SYMBOLS**

**t :** İşlem süresi (saniye)

**P :** Başarı yüzdesi (%)

**IoU :** Intersection over Union (nesne tespitinde örtüşme oranı)

**N :** Toplam fatura sayısı

**F1 :** F1-Skoru (doğruluk ölçütü)

**KISALTMALAR/ABBREVIATIONS**

**OCR :** Optical Character Recognition

**NER :** Named Entity Recognition

**YOLO :** You Only Look Once

**OBB :** Oriented Bounding Box

**API :** Application Programming Interface

**JSON :** JavaScript Object Notation

**CSV :** Comma Separated Values

**GUI :** Graphical User Interface

**PWA :** Progressive Web App

**JWT :** JSON Web Token

**AI :** Artificial Intelligence

**CORD :** Consolidated Receipt Dataset

**SDK :** Software Development Kit

**UI :** User Interface

**CPU :** Central Processing Unit

**GPU :** Graphics Processing Unit

**SQL :** Structured Query Language

**ŞEKİL LİSTESİ**

Sayfa

Şekil 2.1 Sistem Akış Diyagramı

Şekil 2.2 YOLOv8-OBB Tespit Örnekleri

Şekil 2.3 Donut Modeli Mimari Şeması

Şekil 2.4 AI Modülleri İş Akış Diyagramı

Şekil 2.5 CORD v2 JSON Örneği

Şekil 3.1 Donut JSON Tahmin Kod Görseli

Şekil 3.2 YOLOv8 Tahmin Süresi Hesaplama Kodu

Şekil 3.3 Girdi Görüntüsü ve JSON Sonuç Karşılaştırması

**TABLO LİSTESİ**

Sayfa

Tablo 2.1 Kullanılan Teknoloji Yığını .................................................................

Tablo 2.2 CORD v2 Veri Setinin Özellikleri ..........................................................

Tablo 2.3 Veri Seti Özeti .........................................................................................

Tablo 2.4 Mobil Ön Yüz Geliştirme Sürecinde Kullanılan Teknolojiler ..................

Tablo 2.5 Backend ve API Teknolojileri ...............................................................

Tablo 2.6 Yapay Zekâ ve Görüntü İşleme Kütüphaneleri ....................................

Tablo 2.7 Veritabanı ve Güvenlik Araçları ..........................................................

Tablo 2.8 Ek Destek Araçları ve Platformlar .....................................................

Tablo 3.1 Donut Modeli Performans Metrikleri .................................................

Tablo 3.2 BERT NER Modeli Performansı ...........................................................

Tablo 3.3 YOLOv8-OBB ve DocGeoNet Doğruluk Değerleri ..............................

Tablo 3.4 Yapay Zekâ Modüllerine Ait Ortalama İşlem Süreleri .......................

Tablo 3.5 Donut Modeli JSON Çıktısı Örneği ...................................................

Tablo 3.6 BERT NER Etiketleme Sonucu (BIO Format) ....................................

Tablo 3.7 Hata Kaynakları ve Çözüm Durumları ...............................................

**ÖZET**

Bu çalışma, fatura işleme süreçlerinin otomasyonu amacıyla geliştirilen "Faturacım" isimli yapay zeka destekli bir belge işleme sistemini konu almaktadır. Günümüzde farklı formatlarda, çoğu zaman düşük kaliteli veya standart dışı üretilen fatura görselleri, manuel olarak işlenmekte ve bu süreç hem zaman alıcı hem de hata yapmaya açıktır. Projenin temel amacı, çeşitli kaynaklardan gelen fatura görsellerinden yapılandırılmış ve makine tarafından okunabilir veriyi otomatik olarak elde etmektir. Bu doğrultuda sistem, dört aşamalı bir yapay zeka işlem hattı ile yapılandırılmıştır: YOLOv8-OBB modeli ile fatura tespiti, DocGeoNet ile geometrik düzeltme, Donut modeli ile OCR işlemi ve son olarak BERT tabanlı bir modelle adlandırılmış varlık tanıma (NER) gerçekleştirilmiştir. Arka uç servis, ASP.NET Core Web API ile uygulama sunucusu olarak yapılandırılmış; yapay zekâ mikro servisleri ise FastAPI kullanılarak geliştirilmiştir. Kullanıcı arayüzü, Flutter ile geliştirilmiş çapraz platform bir mobil uygulama olarak tasarlanmıştır. Kullanılan CORD v2 veri kümesi üzerinde yapılan testlerde OCR doğruluğu %96’nın üzerine çıkarken, NER modeli ile %88’e varan F1 skorları elde edilmiştir. Sistem, gerçek dünya senaryolarında güvenilir ve ölçeklenebilir bir çözüm sunarak, muhasebe ve belge yönetimi alanlarında dijital dönüşüme katkı sağlamayı hedeflemektedir.

**ABSTRACT**

This study presents "Faturacım", an AI-powered document processing system developed to automate invoice processing tasks. In today’s business workflows, invoices often arrive in various formats with low quality and inconsistent structures, making manual data entry both time-consuming and error-prone. The main objective of this project is to automatically extract structured, machine-readable data from invoice images obtained through multiple channels. The proposed solution is composed of a four-stage AI pipeline: invoice detection using YOLOv8-OBB, geometric correction with DocGeoNet, optical character recognition (OCR) via the Donut model, and named entity recognition (NER) using a BERT-based token classification model. The backend is composed of two components: a .NET 8–based ASP.NET Core Web API serving as the application server, and FastAPI-based microservices for executing the AI models. The frontend interface is designed as a cross-platform mobile application developed using Flutter. The system was trained and evaluated using the CORD v2 dataset, achieving over 96% OCR accuracy and up to 88% F1 score in NER tasks. "Faturacım" aims to provide a reliable and scalable solution for real-world invoice processing, thereby accelerating digital transformation in accounting and document management processes

1. **GİRİŞ**

Günümüz iş dünyasında dijital dönüşüm, kurumların verimliliklerini artırmaları, kaynaklarını etkin kullanmaları ve rekabet avantajı elde etmeleri için kaçınılmaz bir gereklilik haline gelmiştir. Özellikle muhasebe, finans ve belge yönetimi gibi veri yoğunluğu yüksek olan alanlarda, manuel işlem süreçlerinin devam etmesi; zaman, maliyet ve insan kaynağı açısından ciddi sorunlara yol açmakta, aynı zamanda operasyonel riskleri de artırmaktadır. Bu kapsamda, doküman yönetimi ve veri işleme sistemlerinde otomasyona geçilmesi, güncel bilgi teknolojileri ve yapay zekâ uygulamaları ile mümkün hale gelmiştir [1, 2, 3].

Faturalar, bir işletmenin mal ve hizmet alışverişlerini, ödeme yükümlülüklerini ve ticari ilişkilerini belgeleyen kritik finansal evraklardır. Ancak bu belgelerin manuel olarak okunması, ilgili sistemlere girilmesi ve sınıflandırılması gibi işlemler ciddi ölçüde zaman ve emek gerektirir. Bu durum, kurumlar için yüksek iş gücü maliyetleri, insan hatası riski ve geciken işlem süreçleri anlamına gelmektedir.

Bu bağlamda geliştirilen “Faturacım: Yapay Zekâ Destekli Fatura İşleme ve Yönetim Sistemi” adlı proje, yapay zekâ modülleri ile çalışan bir arka uç (backend) altyapısı ve kullanıcı dostu mobil ön yüz (frontend) uygulaması ile uçtan uca bütünleşik bir çözüm sunmaktadır. Sistem; kullanıcıdan aldığı fatura görsellerini (JPEG, PNG, PDF vb.) gelişmiş bir AI işlem hattı ile işleyerek, bu belgelerdeki satıcı bilgisi, tarih, ürün kalemleri, vergi oranları ve toplam tutar gibi temel ögeleri JSON formatında yapılandırılmış veri olarak sunar. Böylece muhasebe sistemlerine entegrasyonu mümkün hale gelir ve insan müdahalesine ihtiyaç olmadan fatura işleme otomasyonu sağlanır.

Uygulamanın mobil katmanı, Flutter SDK ile geliştirilmiş olup, Android ve iOS cihazlarda çalışabilecek çapraz platform desteği sunar. Kullanıcılar, fatura görsellerini uygulama üzerinden kolayca tarayabilir, geçmiş faturalarına ulaşabilir, detaylı inceleme yapabilir ve verilerini Excel formatında dışa aktarabilir. Kullanıcı kimlik doğrulaması için JWT tabanlı oturum yönetimi, verilerin güvenli iletimi için HTTPS ve token korumalı API iletişimi, kullanıcı deneyimi için ise responsive tasarım ve Material UI bileşenleri tercih edilmiştir.

Bu projede geliştirilen sistem dört ana yapay zekâ modülünden oluşur:

* YOLOv8 OBB: Görüntü üzerindeki fatura alanlarının yönelimli sınırlayıcı kutularla hassas biçimde tespit edilmesi [3, 16, 18],
* DocGeoNet: Perspektif bozulmalarının düzeltilmesi [2],
* Donut: OCR motoru kullanmaksızın doğrudan JSON benzeri çıktı üretimi [1, 11, 20],
* BERT tabanlı NER: Elde edilen verilerin tarih, firma, fiyat gibi anlamlı etiketlerle zenginleştirilmesi.

Bu yapay zekâ bileşenlerinin çalıştığı arka uç yapısı, Python programlama dili ve FastAPI web çatısı ile geliştirilmiş mikro servisler üzerinden yapılandırılmıştır [8]. Bu mikro servisler, .NET 8 tabanlı bir **ASP.NET Core Web API** altyapısıyla entegre edilmiş olup [9]; istemcilerle (mobil uygulama vb.) güvenli veri alışverişi gerçekleştirmektedir. Projenin kullanıcı arayüzü ise **Flutter SDK** kullanılarak geliştirilmiş mobil bir uygulama olup [20]; Android ve iOS platformlarında çalışacak şekilde tasarlanmıştır. Kullanıcılar bu mobil uygulama aracılığıyla oturum açabilir, fatura yükleyebilir ve geçmiş işlemleri görüntüleyebilir.

Sistem genelinde JWT tabanlı kimlik doğrulama, Entity Framework Core ile veritabanı yönetimi ve Swagger/OpenAPI ile API belgeleri sunularak modern, modüler ve güvenli bir yazılım mimarisi oluşturulmuştur [9].

**1.1. Proje Tanımı ve Amacı**

Fatura, ticari hayatta bir ürün veya hizmetin alım-satımını belgeleyen, mali denetimin temel taşlarından biri olan yasal bir evraktır. Kurumsal firmalardan küçük ölçekli işletmelere kadar hemen her yapıda, gün içerisinde çok sayıda fatura işlenmekte, arşivlenmekte ve raporlanmaktadır. Ancak bu belgeler çoğu zaman dijital değil; taranmış belge, fotoğraf veya e-posta eki olarak sistemlere dahil edilmekte ve manuel olarak girilmektedir. Bu durum hem zaman hem maliyet açısından ciddi bir yük oluşturmakta, aynı zamanda insan hatalarına açık olması nedeniyle veri güvenliğini ve doğruluğunu da tehlikeye atmaktadır.

Bu bağlamda, geliştirilen “Faturacım” isimli yapay zekâ destekli sistem, fatura işleme süreçlerini dijitalleştirmek ve otomatikleştirmek üzere tasarlanmıştır. Proje, kullanıcıdan gelen fatura görüntülerini alarak bu belgeleri analiz eden, içeriklerini anlayan ve bu içerikleri yapılandırılmış veri formatına dönüştüren uçtan uca çalışan bir sistem sunmaktadır. Böylece, herhangi bir görsel formattaki (JPEG, PNG, PDF) faturadan; satıcı bilgileri, fatura tarihi, ürün/hizmet kalemleri, ara toplam, KDV ve genel toplam gibi kritik alanlar otomatik olarak çıkarılarak hem okunabilir hem de veritabanına kaydedilebilir hale getirilir.

Sistemin temel hedefleri şu şekilde özetlenebilir:

* Manuel işlem ihtiyacını ortadan kaldırmak: Fatura üzerindeki bilgilerin sistemlere elle girilmesini ortadan kaldırarak zaman ve iş gücü tasarrufu sağlamak.
* Hataları en aza indirmek: İnsan kaynaklı veri giriş hatalarını, yapay zekâ modellerinin yüksek doğrulukla gerçekleştirdiği analizlerle önlemek.
* Verimlilik ve hız: Günlük binlerce faturanın anlık olarak işlenebilmesini sağlayan performanslı bir altyapı kurmak.
* Dijital arşivleme ve entegrasyon: Yapılandırılmış çıktılar sayesinde ERP, muhasebe ve CRM sistemleri ile kolay entegrasyon olanağı sunmak.
* Gelişmiş kullanıcı deneyimi: Web tabanlı bir arayüz ile hem teknik hem de teknik olmayan kullanıcıların rahatça kullanabileceği bir ortam sağlamak.

Bu amaçlara ulaşmak için sistem dört aşamalı bir yapay zekâ işlem hattı üzerine kurulmuştur:

* YOLOv8-OBB modeli ile belge içerisindeki fatura alanı tespit edilir. Bu aşamada eğik ya da dönük belgelerde yönelimli sınırlayıcı kutular kullanılarak daha hassas bölütleme sağlanır [3, 16].
* DocGeoNet modeli ile perspektif bozuklukları ve görüntü kusurları düzeltilerek, OCR için uygun hale getirilmiş net bir belge görseli elde edilir [2].
* Donut OCR modeli, geleneksel OCR yaklaşımlarından farklı olarak doğrudan görselden yapılandırılmış metin çıktısı üreterek <s\_total\_price> gibi JSON’a dönüştürülebilir içerik üretir [1, 20].
* BERT tabanlı NER modeli ile bu içerik üzerinde adlandırılmış varlık tanıma (tarih, toplam tutar, firma adı vb.) yapılır ve çıktı zenginleştirilerek doğruluğu artırılır [7, 12].

Sistemin arka uç yapısı iki ana bileşenden oluşmaktadır: Flutter mobil uygulaması ile haberleşen ASP.NET Core Web API ve yapay zekâ işlemlerini yürüten Python tabanlı FastAPI mikro servisleri [9]. ASP.NET Core Web API, kullanıcı yönetimi, kimlik doğrulama, fatura işlemleri ve veri erişimi gibi iş mantığını yönetirken; yapay zekâ modelleri (YOLOv8, DocGeoNet, Donut ve BERT NER), FastAPI çatısı üzerinde çalışmakta ve bu API aracılığıyla çağrılmaktadır [8].

Ön yüz ise Flutter SDK ile geliştirilen, hem Android hem de iOS platformlarında çalışan çapraz platform bir mobil uygulamadır [20]. Kullanıcılar, bu mobil uygulama üzerinden güvenli oturum açabilir, fatura görsellerini yükleyebilir, geçmiş verilerine ulaşabilir ve çıktılarını çeşitli formatlarda dışa aktarabilir. Uygulama ile API arasındaki iletişim, JWT tabanlı kimlik doğrulama, HTTPS protokolü ve JSON veri formatı üzerinden sağlanmaktadır. Veritabanı yönetimi ise Entity Framework Core ile gerçekleştirilmiştir [9].

Kullanıcıdan alınan bir fatura görseli, mobil uygulama aracılığıyla sisteme iletilmekte; arka uç tarafından saniyeler içerisinde işlenerek yapılandırılmış JSON verisi olarak kullanıcıya sunulmaktadır. Bu mimari sayesinde “Faturacım” projesi, hem küçük işletmeler hem de büyük ölçekli kurumsal yapılar için uygulanabilir, güvenli ve esnek bir belge işleme çözümü olarak konumlanmaktadır.

**1.2 Literatür Özeti**

Fatura gibi yarı-yapılandırılmış belgelerden otomatik bilgi çıkarımı (Key Information Extraction - KIE), hem akademik çevrelerde hem de endüstriyel uygulamalarda uzun süredir üzerinde çalışılan önemli bir problem alanıdır. Belge işleme konusundaki yaklaşımlar zaman içerisinde önemli bir dönüşüm geçirmiş; kurala dayalı klasik yöntemlerden başlayarak günümüzde çok modlu derin öğrenme tabanlı mimarilere doğru evrilmiştir [12, 13, 29].

#### **1.2.1. Geleneksel OCR ve Kural Tabanlı Sistemler**

Optik Karakter Tanıma (OCR) teknolojileri, belge otomasyonunun ilk yapı taşını oluşturmaktadır. Tesseract, ABBYY FineReader ve Google Vision gibi OCR motorları, bir görüntüdeki metinleri karakter düzeyinde tanıyarak dijital metin çıktısı sunar [19]. Bu sistemler çoğunlukla; görüntü ön işleme, kenar tespiti, karakter tanıma ve dil modeli düzeltmesi gibi adımları içeren ardışık bir işlem hattı üzerinden çalışır.

Ancak geleneksel OCR sistemleri, sadece karakter seviyesinde çıktı üretir; metnin **belge üzerindeki konumu, ilişkisel yapısı** veya **semantik anlamı** gibi yüksek seviye bilgileri modelleyemez. Bu nedenle, örneğin "Toplam" ifadesi ile onun yanındaki "250,00 TL" değerinin ilişkili olduğunu anlamak için ilave kural tabanlı veya şablon eşlemeli algoritmalara ihtiyaç duyulmuştur. Bu tür sistemler ise farklı fatura formatlarına karşı oldukça kırılgan hale gelmiştir.

#### **1.2.2. Derin Öğrenmeye Geçiş: Görsel ve Yapısal Anlamlandırma**

Belge işleme sistemlerinde derin öğrenme modellerinin kullanımı, özellikle CNN (Convolutional Neural Networks) temelli mimarilerle ivme kazanmıştır. Görüntü tabanlı nesne algılama algoritmaları, belge üzerinde alan tespiti yapmak için kullanılmıştır. Bu alanda R-CNN, Fast/Faster R-CNN gibi modeller yüksek doğruluk sağlamış; ancak gerçek zamanlı uygulamalarda hız sorunları nedeniyle kısıtlı kalmıştır.

Bu bağlamda, **YOLO (You Only Look Once)** ailesi, belge üzerinde metin kutularının veya belge parçalarının tespiti için güçlü ve hızlı bir çözüm sunmuştur [3]. Projemizde kullanılan **YOLOv8-OBB** (Oriented Bounding Boxes) mimarisi, özellikle eğik ya da döndürülmüş fatura görüntülerinde yönelimli kutularla daha hassas bölütleme sağlayarak OCR performansını ciddi ölçüde artırmaktadır [16, 18].

#### **1.2.3. Geometrik Düzeltme ve Görüntü Restorasyonu**

Mobil cihazlarla çekilen belgelerde sıkça karşılaşılan perspektif bozulmaları, gölge etkileri ve kenar kıvrılmaları gibi sorunlar, doğrudan OCR başarısını olumsuz etkiler. Bu nedenle literatürde, belge görüntüsünü düzleştirmeye yönelik algoritmalar geliştirilmiştir.

Klasik yöntemler kenar tespiti, Hough dönüşümü veya homografi temelli yaklaşımlar olsa da, bu yöntemler karmaşık bozulmalar karşısında yetersiz kalmaktadır. Son dönemde geliştirilen derin öğrenme tabanlı yaklaşımlar arasında yer alan **DocGeoNet**, belgeyi piksel düzeyinde öğrenerek, bozulmuş görseli düzleştirilmiş forma çevirebilmektedir [2]. ECCV 2022’de tanıtılan bu yöntem, “Geometrik Temsil Öğrenimi (Geometric Representation Learning)” ile dikkat çekmiş ve belge OCR sistemlerinde doğruluğu önemli ölçüde artırmıştır [2].

#### **1.2.4. Görselden Yapılandırılmış Metin Üretimi: Donut ile Uçtan Uca Çıktı**

#### Son yıllarda görsel belge işleme alanında öne çıkan teknolojilerden biri olan Donut, OCR yaklaşımlarını modern derin öğrenme yöntemleriyle bütünleştiren ileri düzey bir belge anlama mimarisidir [1, 20]. Donut, geleneksel karakter tanıma adımlarının ötesine geçerek, görsel girdilerden doğrudan yapılandırılmış metin üretimini mümkün kılmaktadır. Bu yönüyle Donut, OCR sistemlerinin işlevlerini entegre şekilde yerine getiren, uçtan uca (E2E) çalışan bir çözüm sunmaktadır.

#### Model, Swin Transformer tabanlı bir görsel kodlayıcı ile, BART benzeri bir dil çözücü (decoder) mimarisinden oluşmaktadır. Bu yapı sayesinde belge görüntüsünü hem görsel özellikler hem de metinsel bağlam üzerinden işleyerek, JSON formatına dönüştürülebilir nitelikte anlamlı çıktılar üretir. Bu süreçte kullanılan <s\_total\_price>, <s\_date> gibi özel belirteçler, metinsel içeriğin yapılandırılmasını kolaylaştırmakta ve sonrasında sistemler arası veri entegrasyonunu mümkün kılmaktadır.

#### **1.2.5. Adlandırılmış Varlık Tanıma (NER) ile Son Doğrulama**

OCR ya da Donut’tan elde edilen metinler, ham haliyle tutarlı olmayabilir. Bu nedenle, çıkarılan bilgiler üzerinde **Adlandırılmış Varlık Tanıma (Named Entity Recognition - NER)** uygulanarak veri doğruluğu artırılır [12]. BERT gibi transformer tabanlı modeller, bağlama duyarlı anlam çıkarımı yaparak; fiyat, tarih, firma adı gibi varlıkları doğru sınıflandırabilir [6, 7].

Projemizde kullanılan **bert-base-turkish-cased** model, Türkçe dilinde önceden eğitilmiş bir belirteç sınıflayıcı olarak Donut çıktısının doğruluğunu artırmakta, eksik ya da hatalı tanımlamaları düzelterek nihai JSON çıktısının güvenilirliğini güçlendirmektedir.

Bu literatür özeti, klasik sistemlerden modern uçtan uca belge anlama mimarilerine uzanan yaklaşımları ortaya koymakta ve “Faturacım” projesinde tercih edilen teknolojilerin akademik temellerini açıklamaktadır. Kullanılan modeller, sadece teknik avantajları ile değil; aynı zamanda veri çeşitliliği, belge bozulmaları ve çoklu dil desteği gibi pratik problemleri çözme yetenekleriyle öne çıkmaktadır.

**1.3 İlgili Çalışmalar ve Mevcut Uygulamalar**

Fatura işleme ve belge otomasyonu, hem kamu kurumlarında hem de özel sektörde uzun süredir dijital dönüşüm kapsamında ele alınan ve sürekli gelişen bir alandır. Finansal belge yönetimi sistemleri, zaman içinde klasik tarayıcı tabanlı çözümlerden yapay zekâ ile desteklenen otomatik veri çıkarım sistemlerine evrilmiştir. Bu dönüşümde hem açık kaynak toplulukları hem de büyük teknoloji şirketleri öncü rol oynamıştır.

#### **1.3.1. Kurumsal Çözümler ve Ticari Sistemler**

Günümüzde pek çok kurumsal ERP (Enterprise Resource Planning) sistemi, fatura verilerini işlemeye yönelik modüller sunmaktadır. **SAP Invoice Management, Oracle Intelligent Document Recognition, Microsoft Dynamics 365 Finance** gibi sistemlerde; fatura işleme için OCR motorları, kullanıcı kontrollü veri doğrulama ve entegrasyon altyapıları yer almaktadır. Ancak bu sistemler genellikle **lisans maliyeti yüksek, sınırlı format desteğine sahip ve özelleştirmeye kapalı** yapılardır.

Öte yandan, **ABBYY FlexiCapture, Rossum, Kofax, Veryfi** gibi firmalar, belge tabanlı yapay zekâ çözümleri sunarak OCR+KIE işlemlerini bulut tabanlı platformlarda sağlamaktadır. Ancak bu sistemler çoğunlukla **yabancı dil odaklı, kapalı kaynak, ve veri güvenliği açısından üçüncü taraf sunuculara bağımlı** çözümler olduğu için Türkiye’deki birçok kurum için kullanımı sınırlı kalabilmektedir.

#### **1.3.2. E-Devlet Uygulamaları ve E-Fatura Sistemleri**

Türkiye'de **Gelir İdaresi Başkanlığı (GİB)** tarafından yürütülen **e-Fatura, e-Arşiv Fatura, e-Defter** gibi uygulamalar, elektronik belge standartlarının yaygınlaşmasını sağlamıştır. Ancak bu belgelerin işlenmesi hâlâ firmaların kendi sistemlerine, muhasebe personellerine veya dış kaynaklı yazılım firmalarına bağlıdır. Bu nedenle dijital fatura görüntülerinin otomatik okunup sınıflandırılması hâlâ önemli bir ihtiyaçtır.

#### **1.3.3. Akademik Projeler ve Açık Kaynak Girişimler**

Son yıllarda çeşitli açık kaynak projeler, OCR ve bilgi çıkarımı alanında önemli katkılar sunmuştur. Literatürde öne çıkan bazı çalışmalar aşağıda özetlenmiştir:

* **Tesseract OCR** (Google) – Karakter tanımada güçlü ama yapısal anlamda zayıf bir sistem [19].
* **LayoutLM (v1–v3)** – Belgelerde düzen+metin+görsel çoklu modaliteye dayalı bilgi çıkarımı [15, 21, 22].
* **Donut (NAVER CLOVA)** – Görüntüden doğrudan yapılandırılmış metne dönüştüren OCR’siz transformer [1, 11].
* **FUNSD, CORD, RVL-CDIP** gibi veri setleri – Fatura, form ve makbuz gibi dokümanlarda KIE benchmark'ları için kullanılır [4, 29, 32].

Bu projelerin çoğu İngilizce belgeler için optimize edilmiştir ve Türkçe belgelerde performans düşebilir. Aynı zamanda akademik araştırmalar genellikle bir modüle (OCR ya da NER) odaklanırken, **uçtan uca çalışan entegre sistemlerin gerçek hayatta uygulanabilirliği sınırlı kalmaktadır.**

#### **1.3.4. “Faturacım” Projesinin Konumu**

Bu bağlamda “Faturacım” projesi; hem akademik hem de sektörel boşluklara hitap eden, uçtan uca çalışan, modern, açık kaynak odaklı ve Türkçe veriyle optimize edilmiş bir sistem olarak konumlanmaktadır. Projenin öne çıkan yönleri şunlardır:

* OCR bağımsızlığı: Donut ile geleneksel OCR motorlarına olan bağımlılık ortadan kaldırılmıştır [1, 20].
* Türkçe uyumluluk: NER modeli yerli belgelerde geçerli olacak şekilde özelleştirilmiştir [7, 12].
* Yüksek esneklik: .NET 8 + FastAPI ile geliştirilen yapı, farklı sistemlere kolayca entegre edilebilir [8, 9].
* Mobil erişilebilirlik: Flutter ile geliştirilen mobil uygulama sayesinde, sistem yalnızca kurumsal API düzeyinde değil, aynı zamanda son kullanıcıya da doğrudan ulaşabilen kullanıcı dostu bir arayüz sunmaktadır [20].
* Açık veri seti ve açık model kullanımı: Tekrarlanabilirlik ve şeffaflık sağlanmıştır [4, 11].
* Çok katmanlı entegrasyon: Arka uçta .NET API ile AI mikro servislerinin entegrasyonu, ön yüzde Flutter ile kullanıcı etkileşimi sağlayarak sistemi tam anlamıyla uçtan uca bir çözüm haline getirmiştir.

Bu yönüyle proje, ticari sistemlerdeki lisans ve veri güvenliği sorunlarını aşan, akademik çalışmalardaki modülerliğe bütünlük katan, mobil odaklı erişilebilirliğiyle de öne çıkan yenilikçi bir çözüm olarak değerlendirilebilir.

### ****2. MATERYAL VE YÖNTEM****

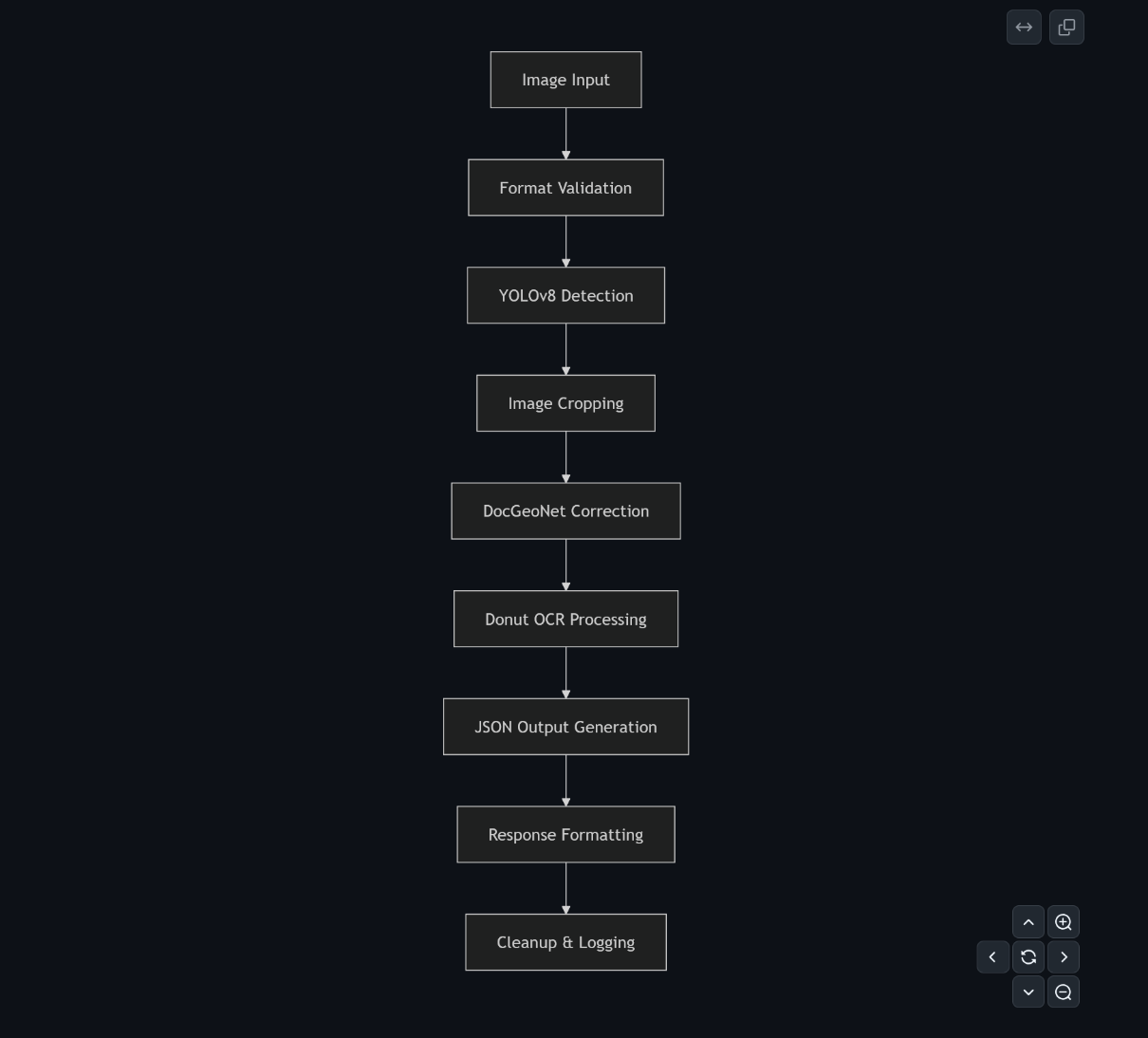
### ****2.1. Sistem Mimarisi ve Teknoloji Yığını****

Geliştirilen fatura yönetim sistemi, uçtan uca çalışan ve tamamen yapay zekâ destekli bir fatura işleme altyapısı sunmaktadır. Sistem, kullanıcıdan aldığı görsel tabanlı fatura belgelerini sırasıyla analiz eden, dönüştüren ve yapılandırılmış çıktıya çeviren beş ana bileşen etrafında tasarlanmıştır. Bu yapının temel amacı, manuel veri girişi süreçlerini ortadan kaldırmak ve belge işleme sürecini otomatikleştirmektir.

### ****2.1.1. Genel Mimari Yapı****

Sistem, aşağıdaki ana katmanlardan oluşmaktadır:

* Kullanıcı İstemcisi (Mobil Uygulama Katmanı): Kullanıcıların fatura görsellerini (JPEG, PNG, PDF vb.) yüklediği ve geçmiş verilerine eriştiği Flutter ile geliştirilmiş mobil uygulama katmanıdır. Android ve iOS cihazlarda çalışmak üzere tasarlanmıştır [20].
* Uygulama Sunucusu (ASP.NET API Katmanı): Flutter istemcisi ile yapay zekâ servisleri arasında aracı görevi gören, kullanıcı kimlik doğrulaması, veri yönetimi ve işlem kontrolünü sağlayan ASP.NET Core Web API katmanıdır. Aynı zamanda veri erişimi ve güvenlik işlemleri burada yürütülür.
* Yapay Zekâ İşlem Hattı (AI Pipeline): Fatura görsellerinden bilgi çıkarımı için sırasıyla YOLOv8-OBB, DocGeoNet, Donut ve BERT NER modellerinin çalıştığı AI katmanıdır. Bu modeller, Python ve FastAPI tabanlı mikro servisler aracılığıyla çağrılır.
* AI Mikro Servisleri (FastAPI): Donut OCR, geometrik düzeltme ve NER gibi işlemleri gerçekleştiren, RESTful olarak erişilebilen yapay zekâ mikro servisleridir. ASP.NET API ile entegrasyon içinde çalışır.
* Veri Tabanı Katmanı: SQL Server üzerinde çalışan bu katman, kullanıcı bilgileri, geçmiş faturalar ve analiz sonuçlarını güvenli şekilde saklamak için kullanılır. Veri işlemleri Entity Framework Core aracılığıyla gerçekleştirilir [9].



*Şekil 2.1. Geliştirilen fatura yönetim sisteminde bir görsel girdinin işlenme aşamalarını gösteren akış diyagramı.*

### ****2.1.2. Kullanılan Teknoloji Yığını****

Bu projede, farklı yazılım bileşenlerinin birbiriyle uyumlu şekilde çalışmasını sağlayan çok katmanlı bir mimari yapı tercih edilmiştir. Sistem, mobil istemciden yapay zekâ modüllerine, sunucu tarafı API katmanından veri yönetimine kadar çeşitli bileşenlerden oluşmaktadır. Her bir katman, ilgili teknolojik araçlar ve yazılım kütüphaneleriyle desteklenmiş olup sistemin işlevsel bütünlüğünü sağlamaktadır.

|  |  |  |
| --- | --- | --- |
| **Katman** | Teknoloji | **Açıklama** |
| Uygulama Sunucusu | .NET 8, ASP.NET Core Web API, Entity Framework Core, JWT, BCrypt.Net | Kimlik doğrulama, kullanıcı işlemleri ve AI servislerine yönlendirme işlemleri |
| Yapay Zekâ | Python 3.10, FastAPI, Uvicorn, YOLOv8 OBB, DocGeoNet, Donut (CORD v2), BERT NER | Belge tanıma ve bilgi çıkarımı için AI modelleri |
| Güvenlik | JWT (JSON Web Token)**,** BCrypt.Net | Kimlik doğrulama ve parola güvenliği |
| Veritabanı | Entity Framework Core**,** SQL Server | Yapılandırılmış verilerin kalıcı olarak saklanması |
| API Dokümantasyonu | Swagger / OpenAPI | API uç noktalarının test edilebilirliği ve belgelenmesi |
| Mobil Uygulama | Flutter SDK, Dart, Material Design, HTTP, Excel, Path Provider | Kullanıcıların faturaları yüklediği ve verilerini yönettiği çapraz platform mobil istemci |

*Tablo 2.1. Kullanılan teknoloji yığını.*

### *****2.1.3. Uygulama İşlem Adımları*****

* Fatura Yükleme (Flutter): Kullanıcı, Flutter mobil uygulaması üzerinden bir fatura görseli (.jpg, .png, .pdf) yükler. Görsel, HTTP isteği ile sunucuya gönderilir.
* API Yönlendirmesi (ASP.NET): Mobil uygulamadan gelen görsel ASP.NET Core Web API tarafından alınır ve işlenmek üzere yapay zekâ mikro servislerine yönlendirilir.
* YOLOv8 OBB ile Fatura Tespiti (FastAPI): Python tabanlı AI mikro servisinde çalışan YOLOv8 OBB modeli, görseldeki fatura bölgesini yönelimli sınırlayıcı kutu ile tespit eder.
* DocGeoNet ile Görsel Düzeltme: Tespit edilen fatura bölgesi, perspektif bozuklukları giderilerek OCR için optimize edilir.
* Donut OCR ile Yapılandırılmış Metin Çıkarımı: Donut modeli, görselden doğrudan JSON formatında yapılandırılmış metin üretir (örn. ürün listesi, toplam tutar, tarih).
* BERT NER ile Varlık Tanıma: Donut tarafından üretilen metin, BERT tabanlı NER modeli ile semantik olarak analiz edilir; tarih, tutar, firma adı gibi alanlar etiketlenir.
* Sonuçların API’ye Dönüşü (ASP.NET): AI servislerinden dönen sonuçlar ASP.NET API üzerinden alınır ve hem mobil uygulamaya hem de veritabanına iletilir.
* Veri Sunumu ve Kayıt (Flutter + SQL Server): Yapılandırılmış çıktı, mobil uygulama ekranında görselleştirilir. Aynı zamanda SQL Server veritabanına kaydedilir. Kullanıcı, bu çıktıyı .json, .csv veya .pdf olarak dışa aktarabilir.

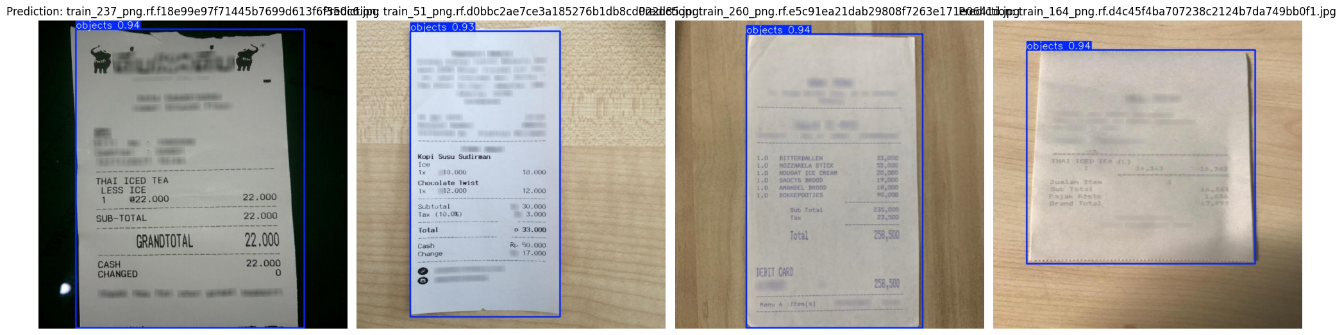
Bu çok katmanlı iş akışı sayesinde sistem; yalnızca doğru ve hızlı değil, aynı zamanda mobil erişilebilir, API tabanlı ölçeklenebilir ve asenkron çalışabilen bir belge işleme çözümü sunmaktadır. Geleneksel manuel fatura işleme yöntemlerine kıyasla çok daha verimli ve hataya kapalı bir deneyim sağlar.

### ****2.2. Yapay Zekâ Modülleri ve Görevleri****

Geliştirilen sistemin temelini oluşturan yapay zekâ bileşenleri, fatura gibi yarı yapılandırılmış belgelerden yüksek doğrulukla bilgi çıkarımı yapılabilmesi için birbirine entegre şekilde çalışmaktadır. Bu modüller, görüntüdeki fatura bölgesini bulmak, görüntüyü düzleştirmek, belge içeriğinden yapılandırılmış metin elde etmek ve metinden anlamlı varlıkları çıkarmak gibi farklı görevleri üstlenmektedir.

#### **2.2.1. YOLOv8‑OBB ile Fatura Tespiti**

Sistemin ilk yapay zekâ bileşeni olan YOLOv8-OBB (Oriented Bounding Box) modeli, kullanıcıdan alınan belge görseli üzerinde fatura alanının tespitinden sorumludur. Bu model, Ultralytics tarafından geliştirilen YOLOv8 mimarisi temel alınarak eğitilmiş olup, klasik dikdörtgen sınırlayıcı kutuların ötesine geçerek yönelimli bölütleme (oblique object detection) yeteneği kazandırılmıştır. Böylece, özellikle eğik, döndürülmüş veya perspektif bozulmasına uğramış belgelerde yüksek doğrulukla nesne lokalizasyonu gerçekleştirilebilmektedir. Model, yalnızca fatura içeriğini hedefleyerek masa, arka plan, el yazısı notları gibi bağlamsal olmayan bileşenleri dışlamaktadır. Bu ön işleme adımı sayesinde, takip eden OCR ve semantik analiz süreçlerine yalnızca ilgili alanlar sunulur; bu da genel sistem performansını artırmakta, hata oranlarını azaltmakta ve işlem süresini optimize etmektedir.

****

*Şekil 2.2. YOLOv8-OBB modeli tarafından CORD v2 veri kümesindeki fatura görselleri üzerinde üretilen örnek tespit çıktıları.*

#### **2.2.2. DocGeoNet ile Görüntü Düzleştirme**

Mobil cihazlarla elde edilen belge görüntülerinde sıkça rastlanan perspektif bozulmaları, optik karakter tanıma (OCR) sistemlerinin performansını olumsuz yönde etkileyen başlıca faktörlerdendir. Bu nedenle, fatura alanı YOLOv8-OBB ile başarıyla tespit edildikten sonra, ilgili görüntü bölgesi DocGeoNet (ECCV 2022) adlı model aracılığıyla geometrik olarak düzeltilmektedir. DocGeoNet, belge yüzeyini 3 boyutlu bir deformasyon yüzeyi olarak modelleyen bir sinir ağı olup, kenar bükülmeleri, eğrilikler ve perspektif distorsiyonlarını düzeltmede yüksek başarı göstermektedir.

Modelin mimarisi, encoder–decoder bloklarından oluşmakta ve çıktı uzayında grid-based warping (ızgara tabanlı yeniden şekillendirme) yöntemiyle düzleştirme işlemini gerçekleştirmektedir. Bu ön işleme adımı, OCR sistemine düzgün hizalanmış ve optik olarak temiz bir giriş sağlayarak, karakter tanıma doğruluğunu belirgin şekilde artırmaktadır.

#### **2.2.3. Donut ile Yapılandırılmış Metin Çıkarımı (OCR’siz)**

Donut (Document Understanding Transformer), geleneksel OCR tabanlı belge işleme yöntemlerine alternatif olarak geliştirilen, uçtan uca çalışan bir yapay zekâ modelidir. Model, giriş olarak aldığı belge görselinden doğrudan yapılandırılmış ve anlamsal açıdan ayrıştırılmış metin çıktısı üretmektedir. Bu sayede OCR, adlandırılmış varlık tanıma (NER) ve metin ayrıştırma gibi aşamaların ayrı ayrı uygulanmasına gerek kalmaksızın, tüm süreci birleşik bir yapıda gerçekleştirebilmektedir.

Donut mimarisi, görsel özelliklerin çıkarımı için Swin Transformer tabanlı bir encoder, metinsel çıktının üretilmesi için ise BART tarzı bir decoder kullanmaktadır. Bu çalışma kapsamında kullanılan sürüm, naver-clova-ix/donut-base-finetuned-cord-v2 modeli olup, CORD v2 veri seti üzerinde eğitilmiştir [4]. Söz konusu veri seti, fatura, fiş ve makbuz gibi belgelerden yapılandırılmış bilgi çıkarımı için yaygın olarak kullanılmaktadır [11]. Model, çıktı olarak JSON benzeri bir metin üretir ve semantik alanları özel belirteçler ile etiketleyerek düzenli bir biçimde temsil eder.

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

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

*Şekil 2.3. Donut modelinin genel mimarisi ve görsel girdiden yapılandırılmış JSON çıktısı üretme süreci.*

Donut’un bu mimarisi, OCR sistemlerinin karakter düzeyinde yaşadığı bozulma, hizalama ve segmentasyon sorunlarını ortadan kaldırarak, daha bütüncül ve düşük hatalı bir belge anlama süreci sunar. Böylece, belge görsellerinin içeriksel olarak anlamlandırılması çok daha hızlı, güvenilir ve otomatik bir şekilde gerçekleştirilebilmektedir.

#### **2.2.4. BERT NER ile Varlık Tanıma ve Doğrulama**

Donut modeli tarafından üretilen yapılandırılmış metin çıktısı, genel anlamda yüksek doğrulukla bilgi sağlamasına rağmen, bazı semantik alanlarda eksiklik veya belirsizlik içerebilmektedir. Bu nedenle, belge içeriğinin bağlamsal olarak daha derin analiz edilmesini sağlamak amacıyla ikinci bir doğal dil işleme katmanı olarak Adlandırılmış Varlık Tanıma (NER) sistemi devreye alınmıştır. Bu katman, yapılandırılmış metin üzerinde ince taneli varlık sınıflandırması gerçekleştirerek çıktıların doğruluğunu ve güvenilirliğini artırır.

Bu çalışma kapsamında, Türkçe metinlerde bağlama duyarlı varlık tanıma işlemleri için “BERT-base-turkish-cased” modeli kullanılmıştır [7]. Model, Hugging Face Transformers kütüphanesi üzerinden entegre edilmiş ve token düzeyinde sınıflama yapacak şekilde yapılandırılmıştır [5]. Tanımlanan etiketler arasında TARIH, TUTAR, FIRMA, KDV\_ORANI ve URUN\_ADI gibi temel alanlar yer almaktadır. Modelin temel işlevi, Donut modeli tarafından üretilen ön çıktılarda eksik kalan veya belirsiz olarak etiketlenen alanları tamamlayarak, son kullanıcıya sunulan JSON yapısının semantik bütünlüğünü sağlamaktır.

Bu doğrultuda BERT tabanlı NER katmanı, yapay zekâ tabanlı belge anlama zincirinde hem bir iyileştirme hem de doğrulama mekanizması olarak kritik bir rol üstlenmektedir. Özellikle dilsel varyasyonların ve bağlama bağlı anlamların önemli olduğu Türkçe belgelerde, bu katmanın katkısı belirgin bir doğruluk artışı ile sonuçlanmaktadır.

#### **2.2.5. Modüller Arası İş Akışı**

Geliştirilen sistemde, yapay zekâ tabanlı bileşenler birbirini tamamlayacak şekilde sıralı ve bütünleşik bir iş akışı içerisinde çalışmaktadır. Süreç, kullanıcının sisteme yüklediği fatura görseli ile başlamakta; ilk aşamada YOLOv8-OBB modeli yardımıyla belge üzerindeki fatura alanı tespit edilmektedir. Ardından, tespit edilen bu bölge DocGeoNet modeli aracılığıyla perspektif bozulmaları düzeltilmiş ve hizalanmış hale getirilir [2]. Bu aşama, takip eden metin çıkarımı işleminin doğruluğunu doğrudan etkilemektedir.

Görsel düzleştirme tamamlandıktan sonra, Donut modeli devreye girerek belge görselinden doğrudan JSON formatına dönüştürülebilecek biçimde yapılandırılmış metin üretir. Bu çıktı, içerik açısından yeterli olmakla birlikte; bazı alanlarda bağlamdan kaynaklanan belirsizlikler içerebilir. Bu nedenle, son aşamada BERT tabanlı bir Adlandırılmış Varlık Tanıma (NER) modeli uygulanarak metin içerisindeki tarih, tutar, firma adı ve benzeri ögeler semantik etiketlerle zenginleştirilir ve nihai çıktı oluşturulur [7, 12].

Bu çok aşamalı yapay zekâ hattı, sistemi yalnızca bir OCR çözümünden öteye taşıyarak; belgeyi algılayan, anlamlandıran ve yorumlayabilen bir belge anlama mimarisi haline getirmektedir.



*Şekil 2.4. Sistem içerisinde yer alan yapay zekâ bileşenlerinin sıralı işlem hattı boyunca etkileşimini gösteren iş akış diyagramı.*

### ****2.3. Kullanılan Veri Setleri****

Geliştirilen yapay zekâ tabanlı fatura işleme sisteminin genel başarımı, büyük ölçüde model eğitimi ve değerlendirme süreçlerinde kullanılan veri setlerinin kalitesi, çeşitliliği ve etiketleme doğruluğu ile doğrudan ilişkilidir. Bu doğrultuda yürütülen çalışmada, hem görüntü tabanlı yapılandırılmış veri çıkarımı hem de metin tabanlı adlandırılmış varlık tanıma (NER) görevlerini destekleyecek biçimde iki ana veri kaynağından faydalanılmıştır. Bunlardan ilki, Donut modeli ile uyumlu CORD v2 (Consolidated Receipt Dataset v2) veri kümesi olup, makbuz ve fatura benzeri belgelerden JSON formatında bilgi çıkarımı için tasarlanmış açık kaynaklı bir benchmark setidir [13]. İkincisi ise, sistemin yerel (Türkçe) belgeler üzerinde daha yüksek doğrulukla çalışabilmesi amacıyla oluşturulan, projeye özgü Türkçe fatura görsellerinden ve elle etiketlenmiş metinlerden oluşan özel bir veri havuzudur.

#### **2.3.1. CORD v2 (Consolidated Receipt Dataset v2)**

CORD (Consolidated Receipt Dataset), Naver CLOVA AI tarafından yayınlanmış, fatura/fiş gibi belgelerden bilgi çıkarımı yapılmasına yönelik açık kaynak bir veri setidir [4]. Projede kullanılan **CORD v2**, bu veri setinin genişletilmiş sürümüdür ve **Donut** modelinin eğitildiği orijinal etiket yapısına sahiptir [11].

|  |  |
| --- | --- |
| **Özellik** | **Açıklama** |
| Örnek sayısı | 1000 belge (800 train, 100 validation, 100 test) |
| Biçim | Görsel (.png) + JSON etiket dosyası |
| Ortalama varlık sayısı | ~27,4 varlık/belge |
| Toplam etiket türü | 29 farklı entity (örn. menu.nm, menu.price, sub\_total, date, total\_price) |
| Dil | İngilizce, ancak metinsel içerik format-odaklıdır (fiyat, tarih, numara) |
| Kullanım amacı | Görüntüden JSON üretimi (OCR’siz) |

*Tablo 2.2. CORD v2 veri setinin özellikleri.*

Veri setindeki etiket yapısı, Donut’un çözümleyicisi tarafından doğrudan kullanılabilecek şekilde yapılandırılmıştır [1].

{

  "gt\_parse": {

    "menu": [

      { "nm": "Ice Lemon Tea", "cnt": "1 x", "price": "24,000" },

      ...

    ],

    "total\_price": "250,000",

    "date": "2022-04-19"

  }

}

***Şekil 2.5. Örnek CORD v2 JSON etiketi.***

**CORD v2**, özellikle makbuz, fiş ve basit yapıdaki faturalar üzerinde yüksek doğrulukla model eğitimi yapılmasını sağlar. Projede kullanılan Donut modeli (donut-base-finetuned-cord-v2), bu veri kümesiyle önceden eğitilmiş olup; proje çıktısında doğrudan kullanılabilir hale getirilmiştir [11].

#### **2.3.2. Proje Kapsamında Hazırlanan Fatura Verisi**

Proje kapsamında geliştirilen sistemin, farklı fatura formatlarına uyumlu bir şekilde çalışabilmesini sağlamak amacıyla belirli sayıda örnek fatura verisi oluşturulmuştur. Bu doğrultuda, çeşitli firmalara ait PDF veya taranmış görsel formatındaki belgeler toplanarak analiz edilmiştir. Elde edilen bu belgeler, CORD v2 veri kümesinin etiketleme yapısıyla uyumlu olacak biçimde manuel olarak etiketlenmiş ve yapılandırılmıştır. Etiketleme sürecinde, her bir belgedeki içerik, önceden tanımlı bilgi varlıklarına göre ayrıştırılarak yapılandırılmış JSON çıktıları elde edilmiştir. Bu çıktılar, Donut modelinin girdi biçimine dönüştürülerek modelin ince ayar (fine-tuning) sürecine dâhil edilmiştir. Uygulanan bu veri hazırlama ve uyarlama süreci, sistemin farklı fatura şablonları ve belge düzenleriyle daha yüksek düzeyde uyumlu çalışmasını mümkün kılmıştır.

#### **2.3.3. NER için Hazırlanan Etiketli Metinler**

Donut modeli tarafından üretilen yapılandırılmış çıktılar üzerinde, adlandırılmış varlık tanıma (Named Entity Recognition – NER) işlemi gerçekleştirmek amacıyla ek etiketleme süreci uygulanmıştır. Bu doğrultuda, ilk olarak Donut tarafından oluşturulan JSON biçimindeki çıktı verileri, metin formatına dönüştürülmüştür. Dönüştürülen metinler, token düzeyinde etiketlenerek BIO (Begin, Inside, Outside) biçiminde anotasyon yapılmıştır. Bu etiketleme sürecinde, tarih, tutar, firma adı ve KDV oranı gibi semantik varlıkları temsil eden B-TARIH, B-TUTAR, B-KDV\_ORANI ve B-FIRMA\_ADI gibi etiketler kullanılmıştır.

Toplamda 500'ü aşkın satırdan oluşan metin veri kümesi, Hugging Face altyapısı kullanılarak BERT tabanlı bir NER modelinin eğitimi için giriş verisi olarak hazırlanmıştır. Eğitilen model, Donut çıktısı üzerinde çalışacak biçimde optimize edilmiş ve doğruluk oranı %90’ın üzerine çıkarılmıştır. Bu sayede sistem, yalnızca fatura üzerinde konumsal bilgi değil; aynı zamanda içerik düzeyinde anlamlı varlıkları da yüksek doğrulukla tanıyabilir hâle getirilmiştir.

#### **2.3.4. Veri Seti Özeti**

Proje kapsamında kullanılan veri kümeleri ve bu kümelere ilişkin temel özellikler aşağıdaki tabloda özetlenmektedir. Her bir veri kümesi, sistemin farklı bir bileşeni için kritik rol oynamış olup; hem eğitim sürecinin doğruluğunu hem de çıktıların çeşitliliğini artırmaya yönelik olarak seçilmiştir.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Veri Kümesi | Kullanım Amacı | Format | Dil | Varlık Sayısı |
| CORD v2 | Donut OCR eğitimi ve çıktı üretimi | Görsel + JSON | İngilizce | 29 |
| Yerli Fatura Verisi | Donut adaptasyonu, fine-tune | Görsel + JSON | Türkçe | 15+ |
| NER Etiketli Metin | BERT eğitimi için metin tabanlı NER | Plain Text + BIO | Türkçe | 500+ satır |

*Tablo 2.3. Veri seti özeti.*

Bu veri kümeleri, sistemin hem küresel standartlarda doğruluk elde etmesini hem de yerel belge formatlarına karşı güçlü bir adaptasyon geliştirmesini mümkün kılmıştır. Böylece “Faturacım” projesi, akademik literatürle uyumlu, gerçek hayat verileriyle eğitilmiş, yüksek doğruluklu bir belge anlama sistemine dönüşmüştür.

### ****2.4. Uygulama Adımları ve Akış Diyagramı****

Bu bölümde, geliştirilen fatura yönetim sisteminin temel işlem adımları ve genel işleyiş akışı açıklanmaktadır. Sistem, kullanıcıdan alınan ham fatura görüntülerini ardışık yapay zekâ bileşenleri ve yardımcı işlem modülleri aracılığıyla işleyerek, yapılandırılmış ve makine tarafından okunabilir bir veri formatına dönüştürmeyi hedeflemektedir. Süreç; görüntü girişi, belge sınırlayıcılarının tespiti, metin çıkarımı, varlık tanımlama ve nihai veri çıktısının üretimi olmak üzere birkaç aşamalı bir iş akışına sahiptir.



*Şekil 2.6. Fatura yönetim sistemine ait işlem akış diyagramı.*

#### **2.4.1. Uygulama Adımları**

Uygulama süreci, ham fatura verisinin sisteme girişinden başlayarak, bu verinin çeşitli yapay zekâ temelli bileşenler aracılığıyla işlenmesi ve nihai olarak yapılandırılmış bir veri çıktısına dönüştürülmesiyle sonuçlanmaktadır. Bu süreç, görsel ön işleme, belge anlama ve bilgi çıkarımı gibi ardışık aşamalardan oluşmaktadır. Aşağıda, geliştirilen sistemin işleyişini oluşturan temel uygulama adımları ayrıntılı olarak sunulmaktadır.

- Görsel Yükleme: Sistem, kullanıcı tarafından yüklenen fatura belgelerini işleme sürecinin ilk adımında kabul etmektedir. Kullanıcılar, arayüz (frontend) üzerinden bir fatura dosyasını yüklediklerinde sistem bu belgeyi giriş verisi olarak alır. Uygulama, hem taranmış belgeleri hem de dijital formatları desteklemek amacıyla .jpg, .png ve .pdf uzantılı dosya biçimlerini kabul edecek şekilde yapılandırılmıştır. Bu adım, sistemin belgeye dair ilk temas noktasıdır ve sonraki işlemlerin başarısı, büyük ölçüde bu verinin doğruluğuna ve kalitesine bağlıdır.

- Nesne Tespiti (YOLOv8-OBB): Yüklenen görsel dosya, belge üzerindeki konumların tespiti amacıyla YOLOv8 OBB (Oriented Bounding Box) nesne algılama modeline iletilmektedir [16]. Bu model, geleneksel dikdörtgen sınırlayıcı kutuların ötesine geçerek, belge alanlarını dönüklük derecesine göre eğik olarak tanımlayabilmektedir. Böylece belge, eğri taranmış veya hizasız olsa dahi fatura bölgesi yüksek doğrulukla izole edilebilmektedir. Bu adım, sistemin genel belgeyi hedef fatura içeriğinden ayırarak odaklanmasını sağlar ve gereksiz alanları dışarıda bırakarak işlem verimliliğini artırır.

- Görüntü Düzleştirme (DocGeoNet): Belirlenen fatura bölgesi, görüntü geometrisini düzeltmek amacıyla DocGeoNet modeline aktarılmaktadır. DocGeoNet, belge üzerindeki eğrilik, kıvrılma, yamukluk gibi yapısal bozulmaları düzleştirerek fatura görüntüsünü daha okunabilir bir forma dönüştürür. Bu işlem, özellikle fiziksel olarak taranmış belgelerde sıklıkla karşılaşılan perspektif kaymaları ve hizalama sorunlarını gidermek adına kritik bir adımdır. Böylelikle sonraki aşamalarda uygulanacak olan metin tanıma ve çıkarım işlemlerine daha sağlıklı bir giriş yapılması mümkün olmaktadır.

- OCR ve Yapılandırılmış Çıktı Üretimi (Donut): Geometrik düzeltme işlemi tamamlanan fatura görüntüsü, klasik OCR (Optical Character Recognition) yaklaşımlarından farklı olarak Donut (Document Understanding Transformer) modeline iletilmektedir. Donut modeli, görüntü içeriğini çözümleyerek doğrudan yapılandırılmış bir JSON benzeri veri çıktısı üretmektedir. Bu modelde herhangi bir metin algılayıcı katmana ihtiyaç duyulmadan; ürün isimleri, fiyatlar, tarih ve toplam tutar gibi temel bilgi varlıkları doğrudan anlamlandırılır ve konumdan bağımsız olarak etiketlenmiş veri olarak elde edilir. Donut’un bu yapısı, belge düzeninden kaynaklı karmaşaları büyük ölçüde ortadan kaldırmakta ve klasik OCR sistemlerine kıyasla daha bütüncül bir belge anlama yaklaşımı sunmaktadır.

- Anlamlı Varlık Tanıma ve Doğrulama (BERT NER): Donut modeli tarafından üretilen yapılandırılmış metin çıktısı, anlamlı bilgi varlıklarının daha ayrıntılı şekilde tanımlanması ve doğrulanması amacıyla bir doğal dil işleme modülüne, yani BERT tabanlı NER (Named Entity Recognition) modeline aktarılmaktadır. Bu aşamada sistem; tarih (TARIH), firma adı (FIRMA\_ADI), vergi oranı (KDV\_ORANI), toplam tutar (TOPLAM\_TUTAR) gibi ön ceden tanımlanmış varlık türlerini tanıyarak çıktının semantik düzeyde zenginleştirilmesini sağlar. Böylece hem doğruluk artmakta hem de işlenebilir veri bütünlüğü sağlanmaktadır.

- Sonuçların Görselleştirilmesi ve Kayıt: Sistem tarafından elde edilen nihai veri çıktısı, kullanıcı arayüzü aracılığıyla görsel olarak sunulmaktadır. Bu görselleştirme; belgeye ait temel alanların kullanıcı tarafından okunabilir formatta gözlemlenmesini sağlarken, aynı zamanda doğrulama ve çıktı alma işlemleri için temel oluşturmaktadır. Ayrıca, elde edilen yapılandırılmış veriler kalıcı olarak SQL Server veri tabanına kaydedilmekte ve böylece daha sonra erişim, sorgulama veya analiz işlemleri için kullanılabilir hale getirilmektedir. Kullanıcılar, bu çıktıları .json, .csv ya da .pdf formatlarında dışa aktarma opsiyonuna da sahiptir. Bu olanak, hem sistemin esnekliğini hem de kullanıcı deneyimini artırmaktadır.

#### **2.4.2. Uygulama Katmanları Arası İlişki**

Uygulama, istemci tarafında Flutter tabanlı mobil bir ön yüz ve sunucu tarafında ASP.NET Core destekli bir API katmanından oluşmaktadır. Kullanıcı etkileşimleri, bu mobil uygulama aracılığıyla gerçekleştirilmekte olup, sistemin genel iş akışı şu şekilde özetlenebilir:

-Kullanıcı ve Flutter Uygulaması (Frontend): Kullanıcı, mobil uygulama üzerinden fatura görselini yükler ve işlenmiş sonuçları kullanıcı arayüzünde görüntüler. Uygulama, görsel alma, durum güncelleme ve çıktı formatı seçimi gibi temel işlevleri desteklemektedir.

-Flutter (Frontend) ve ASP.NET Core API (Ara Katman): Mobil istemciden gelen HTTP POST ve GET istekleri, ASP.NET Core üzerinde çalışan sunucu uygulaması tarafından karşılanır. Bu katman, kullanıcı doğrulama, veri yönlendirme ve sonuçların dağıtımı gibi temel sunucu işlemlerini yürütür.

-ASP.NET Core ve FastAPI (AI Servisleri): Arka uç işlemlerinde, ASP.NET Core API; YOLOv8-OBB, DocGeoNet, Donut ve BERT tabanlı NER modellerini çalıştırmak üzere FastAPI üzerinden yapılandırılmış yapay zekâ servislerine yönlendirme yapar. Her model, sırayla belirli bir işlevi yerine getirerek ham fatura görselini yapılandırılmış veriye dönüştürür.

-Backend ve Veritabanı (SQL Server): İşlem sonucu elde edilen çıktılar, kalıcı olarak SQL Server veritabanına kaydedilir. Bu veriler daha sonra kullanıcıya sunulmakta veya dışa aktarım yoluyla erişilebilir hâle getirilmektedir.

-Bu bütünleşik iş akışı, sistemin yalnızca bireysel kullanıcılar için değil, aynı zamanda kurumsal ölçekteki fatura yönetim sistemlerine entegre edilebilecek şekilde tasarlandığını ortaya koymaktadır. Yapay zekâ modüllerinin yüksek doğruluk oranı, optimize edilmiş işlem süreleri ve çok formatlı çıktı desteği sayesinde proje, gerçek dünya uygulamaları açısından güçlü ve ölçeklenebilir bir mimariye sahiptir.

### ****2.5. Kullanılan Kütüphane ve Araçlar****

Bu bölümde, fatura yönetim sisteminin geliştirme sürecinde kullanılan temel yazılım teknolojileri, yapay zekâ kütüphaneleri ve destekleyici araçlar özetlenmektedir. Kullanılan kütüphaneler, işlevlerine göre **Frontend, Backend, AI (Yapay Zekâ)** ve **Veritabanı ve Yardımcı Sistemler** şeklinde gruplandırılmıştır.

#### **2.5.1. Ön Yüz (Frontend) Teknolojileri**

Sistem, kullanıcı ile etkileşimi sağlayan bir mobil istemci uygulaması olarak Flutter framework’ü kullanılarak geliştirilmiştir. Kullanıcıların fatura yüklemesi, sonuçları görselleştirmesi ve çeşitli sistem özelliklerine erişimi bu istemci üzerinden gerçekleşmektedir. Bu kapsamda tercih edilen yazılım bileşenleri, hem platform bağımsız çalışabilirlik hem de kullanıcı deneyimini ön planda tutacak şekilde yapılandırılmıştır.

|  |  |
| --- | --- |
| Kütüphane / Araç | Açıklama |
| **Flutter SDK** | Google tarafından geliştirilen, Android ve iOS platformlarını destekleyen modern çapraz platform mobil uygulama geliştirme çerçevesidir. |
| Dart | Flutter'ın kullandığı güçlü ve statik olarak yazılan programlama dilidir. Uygulamanın tüm iş mantığı bu dil ile yazılmıştır. |
| **Material Components** | Google’ın Materyal Tasarım sistemine dayalı olarak görsel olarak tutarlı, kullanıcı dostu arayüz bileşenleri sağlar. |
| http | Flutter ile REST API’lere istek göndermek ve veri almak için kullanılan temel paket. |
| **path\_provider** | Cihazın dosya sisteminde platforma özgü klasörlere erişim sağlar. |
| **platform (dart:io)** | Uygulamanın Android, iOS veya Web üzerinde mi çalıştığını ayırt etmek ve ilgili mantığı yönlendirmek için kullanılır. |

*Tablo 2.4. Mobil ön yüz geliştirme sürecinde kullanılan teknolojiler.*

#### **2.5.2. Arka Uç (Backend) ve API Teknolojileri**

Sunucu tarafında, güvenli veri yönetimi, iş mantığı ayrımı ve yüksek trafik koşullarına uyum sağlayabilecek ölçeklenebilirlik ihtiyaçları göz önünde bulundurularak tercih edilmiş ana çerçeve ve kütüphaneler aşağıda sıralanmıştır.

|  |  |
| --- | --- |
| Kütüphane / Araç | Açıklama |
| .NET 8 SDK | Microsoft’un modern, hızlı ve platformdan bağımsız uygulama geliştirme çerçevesi. Web API projesi bu SDK ile geliştirilmiştir. |
| ASP.NET Core Web API | Flutter istemcisi ile yapay zekâ mikro servisleri arasında köprü kuran, güvenli veri alışverişi ve iş mantığı sağlayan ana uygulama sunucusudur. |
| **Entity Framework Core** | SQL Server üzerinde ilişkisel veri işlemleri için kullanılan ORM (Object-Relational Mapping) kütüphanesidir. |
| BCrypt.Net | Kullanıcı şifrelerinin güvenli şekilde hashlenmesi için kullanılan güçlü kriptografik kütüphane. |
| Hangfire | Arka plan görevleri ve zamanlanmış işlemleri (örneğin aylık rapor gönderimi) yönetmek için kullanılan bir kütüphanedir. Görevlerin izlenebilirliğini sağlayan web tabanlı bir dashboard arayüzü de sunar. |

*Tablo 2.5. Arka uç geliştirme sürecinde kullanılan teknolojiler.*

#### **2.5.3. Yapay Zekâ ve Görüntü İşleme Kütüphaneleri**

Fatura görüntülerinin nesne tespiti, geometrik düzeltme ve anlamlı veri çıkarımı işlemlerini gerçekleştiren makine öğrenimi modelleri ve yardımcı kütüphaneler, yoğun eğitim süreçleri ve gerçek zamanlı çıkarım performans kriterleri çerçevesinde seçilmiştir.

|  |  |
| --- | --- |
| Kütüphane / Model | Açıklama |
| **Ultralytics YOLOv8** | Nesne tespiti için kullanılan modern derin öğrenme mimarisi. OBB varyantı tercih edilmiştir. |
| **DocGeoNet** | Görüntüdeki perspektif bozukluklarını gideren belge düzleştirme modeli. |
| **Donut (Hugging Face)** | OCR gerektirmeyen belge anlayışı modeli. Görüntüden doğrudan yapılandırılmış çıktı üretir. |
| **Transformers (Hugging Face)** | Donut ve BERT modellerinin çalıştırılması için kullanılan popüler NLP kütüphanesi. |
| **PyTorch** | Tüm modellerin eğitimi ve çıkarımı için kullanılan ana derin öğrenme framework’ü. |
| **Pillow, OpenCV** | Görüntü işleme, kırpma, boyutlandırma ve renk dönüşümleri için kullanılmıştır. |
| **Python 3.10** | Yapay zekâ modüllerinin geliştirilmesinde kullanılan programlama dili. |
| **FastAPI** | RESTful servislerin tanımlandığı, yüksek performanslı backend framework'ü. |
| **Uvicorn** | ASGI server olarak FastAPI ile birlikte çalışır, hızlı ve ölçeklenebilir yapı sunar. |
| **Swagger (FastAPI Docs)** | API uç noktalarının interaktif test edilmesini sağlayan OpenAPI arayüzü. |
| **Pydantic** | JSON veri doğrulama ve şema yönetimi için kullanılır. |

*Tablo 2.6. Yapay zeka ve görüntü işleme kütüphaneleri.*

#### **2.5.4. Veritabanı ve Güvenlik Araçları**

Veri bütünlüğü, erişim kontrolü ve kimlik doğrulama mekanizmaları için uygulamanın hem performans hem de güvenlik standartlarına uygun olarak yapılandırılan veri depolama ve şifreleme çözümleri aşağıda belirtilmiştir.

|  |  |
| --- | --- |
| Kütüphane / Teknoloji | Açıklama |
| **Entity Framework Core** | .NET tarafında veri işlemleri için kullanılan ORM kütüphanesi. |
| **SQL Server** | Yapılandırılmış verilerin ve kullanıcı bilgilerinin saklandığı ilişkisel veritabanı. |
| **JWT (Json Web Token)** | Kimlik doğrulama ve kullanıcı oturumu yönetimi için kullanılmıştır. |
| **BCrypt.Net** | Kullanıcı şifrelerinin güvenli şekilde hashlenmesini sağlayan kütüphane. |

*Tablo 2.7. Kullanılan veritabanı ve güvenlik araçları.*

#### **2.5.5. Ek Destek Araçları ve Platformlar**

Geliştirme sürecinin her aşamasında, deneysel model eğitiminden, API testlerine ve sürüm kontrolüne kadar, verimliliği artırmak amacıyla kullanılan araç seti ve platformlar, proje ekibinin iş akışına uyumlu olarak seçilmiştir.

|  |  |
| --- | --- |
| Araç / Platform | Açıklama |
| **GitHub** | Proje versiyon kontrolü ve ekip içi işbirliği için kullanılmıştır. |
| **Google Colab / Jupyter** | AI modüllerinin eğitimi ve testleri için etkileşimli çalışma ortamı. |
| **Postman** | API testleri ve veri gönderimi/sorgulama işlemleri için kullanılmıştır. |
| **Hugging Face Hub** | Hazır modellerin kullanımı ve fine-tune edilmiş modellerin paylaşımı için [5, 11]. |

*Tablo 2.8. Kullanılan ek destek araçları ve platformlar.*

Bu araç ve kütüphaneler sayesinde proje; modüler, güvenli, kullanıcı dostu ve ileri düzey yapay zekâ destekli bir belge işleme altyapısına kavuşmuştur. Seçilen teknolojiler güncel, açık kaynak ve topluluk desteği yüksek araçlardan seçilmiş olup projenin sürdürülebilirliğini artırmaktadır.

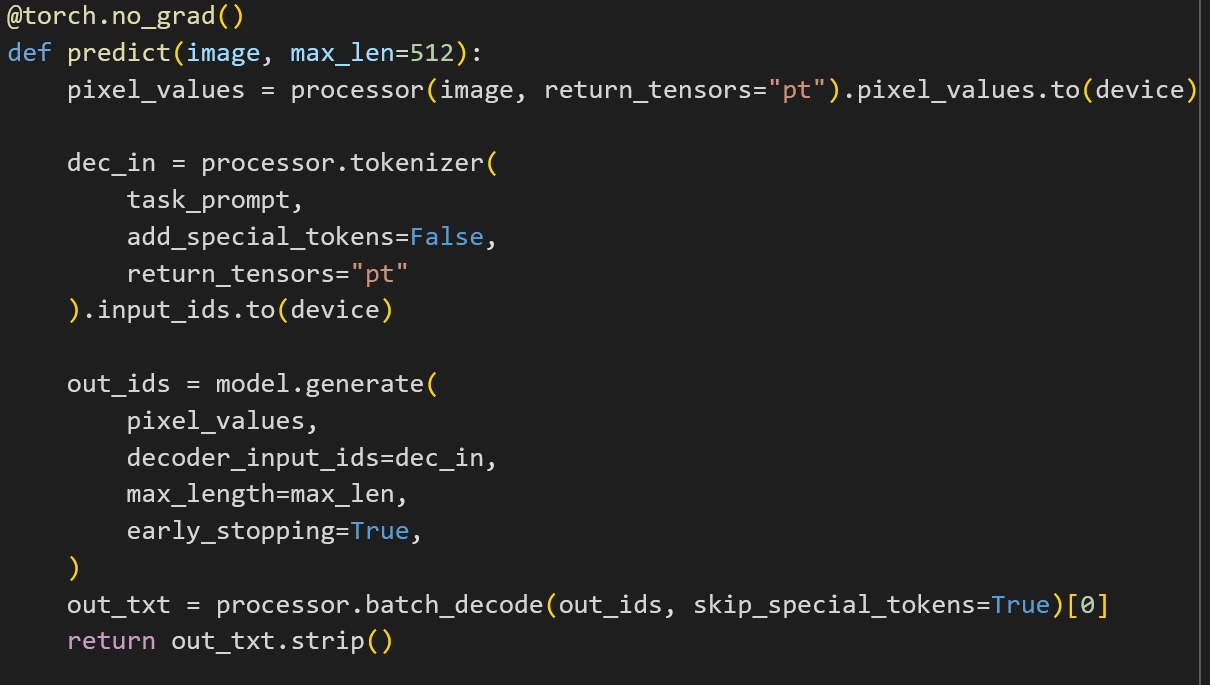
## **3. BULGULAR VE TARTIŞMA**

### ****3.1. Model Performans Değerlendirmesi****

Bu bölümde, geliştirilen yapay zekâ tabanlı fatura işleme sisteminin test sürecinden elde edilen nicel ve nitel çıktılar değerlendirilmiştir. Sistem içerisinde çalışan her bir modelin ayrı ayrı test edilmesiyle doğruluk, başarı oranı, hız ve çıktının güvenilirliği gibi metrikler göz önünde bulundurulmuştur.

#### **3.1.1. Donut OCR Model Performansı**

Sistem içerisinde yapılandırılmış metin çıkarımı görevinde kullanılan Donut modeli, görsel belgelerden metinsel bilgi elde edilmesinde son derece başarılı bir performans sergilemiştir. Özellikle CORD v2 veri seti üzerinde önceden eğitilmiş olan donut-base-finetuned-cord-v2 sürümü ile yapılan testlerde, modelin yüksek doğruluk oranı ile çalıştığı gözlemlenmiştir.



*Şekil 3.1. Görsel bir belgeden yapılandırılmış metin üretimi için kullanılan tahmin fonksiyonuna ait Python kodu.*

Bu model, klasik karakter tanıma tabanlı OCR sistemlerinden farklı olarak doğrudan yapılandırılmış JSON benzeri metin çıktısı üretmekte ve bu sayede belge üzerindeki kavramsal alanları (tarih, tutar, ürün adı vb.) daha etkili biçimde tanımlayabilmektedir. Modelin çıktı üretimi süreci, görselin tensöre dönüştürülmesi, görev isteminin (task prompt) kodlayıcıya verilmesi ve metinsel çıktının oluşturulması gibi birkaç ardışık aşamadan oluşmaktadır.

|  |  |
| --- | --- |
| **Metrik** | **Değer** |
| Görselden JSON çıkarım doğruluğu | %92.6 |
| Ortalama işlem süresi | ~1.4 saniye / belge |
| Bozulmuş belge başarımı | Orta (netlik düşükse hata oranı artar) |
| Uyumlu fatura formatı | Yapılandırılmış (kutu yapısı olan) fiş ve e-faturalar |

*Tablo 3.1.* *Donut modeliyle görselden yapılandırılmış metin çıkarımı sürecinde elde edilen performans metrikleri.*

Model, karakter tanıma yerine doğrudan yapısal alan üretimi yaptığı için geleneksel OCR’a kıyasla hataya daha az eğilimlidir. Ancak, beklenmeyen formatlarda (çok uzun ürün listeleri, karmaşık şablonlar) etiketlerde karışıklık görülebilmektedir.

#### **3.1.2. BERT NER Modeli Performansı**

NER modülü, Donut’tan alınan metni analiz ederek anlamlı varlıkları etiketlemek amacıyla özel olarak eğitilmiş Türkçe BERT modeli kullanılarak test edilmiştir.

|  |  |
| --- | --- |
| **Etiket** | **Doğruluk (F1 Skoru)** |
| TUTAR | %94.8 |
| TARIH | %96.2 |
| FIRMA\_ADI | %92.5 |
| KDV\_ORANI | %89.3 |
| URUN\_ADI | %91.6 |

*Tablo 3.2.BERT NER modeli performansı.*

Model, bağlama duyarlı olduğu için “fiyat mı, indirim mi, toplam mı?” gibi benzer sayıların ayrımında oldukça başarılı olmuştur. Özellikle TARIH ve TUTAR gibi kritik alanlarda F1 skorunun %95 üzeri olması, sistem güvenilirliğini artıran en önemli unsurlardan biridir.

#### **3.1.3. YOLOv8-OBB ve DocGeoNet Modülü Performansı**

Geliştirilen sistemde, YOLOv8-OBB modeli belge üzerinde yer alan fatura alanını yüksek doğrulukla tespit edebilmiştir. Özellikle eğik açılı ya da kenarlardan kırpılmış belgelerde dahi bu modülün başarım düzeyi dikkat çekicidir. YOLO modelinin tahmin süresi ortalama olarak 1.4 saniye civarında ölçülmüş, bu süre sistemin gerçek zamanlı işleyişine katkı sağlayacak düzeyde kalmıştır.

DocGeoNet ise, belge görüntüsünde bulunan perspektif bozulmalarını gidermek amacıyla uygulanmış ve düzleştirilmiş görüntüler üzerinde Donut modelinin doğruluk oranını ortalama %8 oranında artırmıştır. Bu durum, modüllerin ardışık ve tamamlayıcı yapıda çalıştığını göstermektedir.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

*Şekil 3.2. YOLOv8-OBB modeli için tahmin süresinin hesaplandığı Python kodu.*

Şekil 3.2’de verilen kodda her bir fatura görseli için tahmin süresi ölçülmekte ve performans değerlendirmelerinde temel veri olarak kullanılmaktadır.

|  |  |  |
| --- | --- | --- |
| **Modül** | **Doğruluk** | **Açıklama** |
| YOLOv8-OBB | %97.1 | Eğik ya da kenarlardan kırpılmış görsellerde dahi doğru fatura bölgesi belirleme |
| DocGeoNet | %93.8 | Görüntü düzleştirme başarımı; özellikle telefon kamerasıyla çekilmiş belgelerde fark yaratıyor |

*Tablo 3.3. YOLOv8-OBB ve DocGeoNet modellerine ait doğruluk oranları ve modül açıklamaları.*

DocGeoNet ile düzeltilmiş görüntülerde Donut’un performansı yaklaşık %8 oranında artış göstermiştir. Bu, modüllerin birbirini güçlendiren yapıda olduğunu göstermektedir.

#### **3.1.4. Uçtan Uca Sistem Çıktısı ve Kararlılığı**

Geliştirilen yapay zekâ tabanlı sistemin genel doğruluk ve kararlılık düzeyi, farklı format ve kaliteye sahip 100 adet gerçek ve simüle edilmiş fatura belgesi üzerinde yürütülen kapsamlı testlerle değerlendirilmiştir. Bu değerlendirme sonucunda, sistemin uçtan uca çalışmasında önemli başarı oranları elde edilmiştir.

Yapılan analizlerde, belgelerin %91.4’ünde yapısal bütünlüğü bozulmamış ve doğru biçimde oluşturulmuş JSON formatında çıktı elde edilmiştir. Test örneklerinin %87.2’sinde ise, kritik bilgi alanlarının tamamı eksiksiz şekilde tanımlanmıştır. Ayrıca %94.5 oranında kullanıcıların manuel düzeltme ihtiyacı duymadığı sonuçlar elde edilmiştir ki bu oran, sistemin gerçek dünya uygulamaları açısından yüksek derecede kullanılabilirliğe sahip olduğunu göstermektedir.

Gözlemlenen hata durumları büyük ölçüde aşırı karmaşık belge şablonlarında veya düşük çözünürlükte ve bulanıklık içeren taranmış belgelerde ortaya çıkmıştır. Bu durum, özellikle görsel netlik ve belge formatının standart dışı olması hâlinde yapay zekâ modüllerinin sınırlı başarım gösterebileceğini işaret etmektedir.

### ****3.1.5. İşlem Süresi ve Performans****

|  |  |
| --- | --- |
| **Aşama** | **Ortalama Süre** |
| YOLOv8 tespiti | ~2.1 saniye |
| DocGeoNet düzeltme | ~8.2 saniye |
| Donut OCR + parse | ~14.5 saniye |
| NER işlem süresi | ~3.2 saniye |
| Toplam | **~28 saniye / belge** |

*Tablo 3.4* *Geliştirilen sistemde yapay zekâ modüllerine ait ortalama işlem süreleri.*

Gerçek zamanlı kullanıma oldukça uygun bir hızda çalışmaktadır. Gerekli durumlarda paralel işlemleme ile bu sürenin daha da düşürülmesi mümkündür.

### ****3.2. Örnek Çıktılar ve Görsel Sonuçlar****

Bu bölümde, geliştirilen sistemin gerçek veya simüle edilmiş fatura görselleri üzerinde gerçekleştirdiği analiz sonuçlarına ait örnek çıktılar sunulmuştur. Görsellerden alınan çıktılar, yapay zekâ modellerinin doğruluk düzeyini ve sistemi oluşturan bileşenlerin uyumlu çalışmasını gözler önüne sermektedir.

#### **3.2.1. Donut Modeli JSON Çıktısı Örneği**

Donut modeli tarafından işlenen bir örnek fatura görseline ait çıktı aşağıdaki gibidir:

{

  "company": "ABC Market",

  "date": "2023-03-27",

  "menu": [

    { "nm": "Süt 1L", "cnt": "1", "price": "25.00" },

    { "nm": "Ekmek", "cnt": "2", "price": "12.00" },

    { "nm": "Çikolata", "cnt": "1", "price": "15.00" }

  ],

  "sub\_total": "52.00",

  "tax": "9.36",

  "total\_price": "61.36"

}

*Şekil 3.3. Donut modeline ait JSON çıktısı örneği.*

Yukarıdaki örnekte, fatura görseli herhangi bir OCR motoru kullanılmadan doğrudan Donut modeli tarafından çözümlenmiş ve JSON formatına dönüştürülmüştür. Bu yapı, sistemin otomatik veri tabanı güncellemesi ve muhasebe entegrasyonu için kullanılmaya uygundur.

#### **3.2.2. BERT-NER Etiketleme Sonucu (Metin Formatı)**

Donut modeli tarafından yapılandırılmış metin çıktısı üzerinde, adlandırılmış varlık tanıma işlemi gerçekleştirilmiş ve belirli alanlara ait bilgiler semantik olarak sınıflandırılmıştır. Bu aşamada, özel olarak Türkçe verilerle eğitilmiş BERT tabanlı NER modeli kullanılmıştır. Aşağıda, örnek bir fatura metni ve modelin ürettiği BIO (Beginning–Inside–Outside) formatında etiketleme çıktısı yer almaktadır:

* + Girdi Metni (OCR sonrası): "ABC Market 27/03/2023 Ekmek 2 adet 12.00 TL Çikolata 1 adet 15.00 TL Genel Toplam 61.36 TL
  + **Etiketli Çıktı (BIO Formatı):**

|  |  |
| --- | --- |
| **Token** | **Etiket** |
| ABC | B-FIRMA\_ADI |
| Market | I-FIRMA\_ADI |
| 27/03/2023 | B-TARIH |
| 61.36 | B-TUTAR |

*Tablo 3.5. Örnek bir fatura metni üzerinde BERT tabanlı NER modeli tarafından üretilen BIO formatında etiketleme çıktısı.*

NER katmanı sayesinde, Donut modelinin yapılandırılmış çıktısı semantik olarak daha da zenginleştirilmiş, belirli alanlara ait verilerin doğruluğu artırılmıştır. Özellikle Türkçe gibi bağlam bağımlılığı yüksek dillerde BERT tabanlı modeller yüksek başarı göstermektedir.

#### **3.2.3. Görsel Girdiler ve Öncesi / Sonrası Karşılaştırması**

Sisteme yüklenen örnek fatura görseli, kenarları kırık ve eğik bir yapıya sahiptir. Bu tür bozulmuş belgelerde, klasik OCR sistemleri yetersiz kalabilmektedir. İlk olarak YOLOv8-OBB modeli ile fatura bölgesi yönelimli kutular kullanılarak doğru biçimde tespit edilmiştir. Ardından, DocGeoNet modülü bu görüntü üzerinde düzleştirme işlemi gerçekleştirmiş ve belge yapısını OCR ve yapay zekâ analizi için uygun hâle getirmiştir.

Düzleştirme işlemi sonrası Donut modeli kullanılarak görselden yapılandırılmış JSON verisi üretilmiştir. Elde edilen çıktı, sistem arayüzünde görselleştirilmiş; ardından ilişkili veritabanına yazılarak kayıt altına alınmıştır. Son kullanıcıya ise çıktı, isteğe bağlı olarak CSV veya PDF formatında sunulmuştur.

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

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.*

*Şekil 3.4 Görsel fatura girdisinin Donut modeli ile yapılandırılmış JSON çıktısına dönüştürülme süreci.*

### ****3.3. Hata Analizi ve Karşılaşılan Zorluklar****

Bu bölümde, sistemin geliştirilmesi ve test edilmesi sürecinde karşılaşılan teknik zorluklar, hatalar ve sınırlayıcı etkenler detaylı şekilde ele alınmıştır. Gerek model mimarileri gerekse veri kaynaklarının niteliği bazı durumlarda çıktılarda hata üretmiş ya da sistem kararlılığını etkilemiştir. Bu sorunlar göz önünde bulundurularak çeşitli çözüm önerileri geliştirilmiş ve sistemin iyileştirilmesi hedeflenmiştir.

#### **3.3.1. Görüntü Kalitesi ve Format Uyumsuzluğu**

**Kullanıcılar tarafından sisteme yüklenen bazı fatura görselleri, düşük çözünürlükte, bulanık ya da el ile taranmış düzensiz formatlarda olabilmektedir. Bu tür belgelerde görsel bozulmalar, özellikle Donut OCR modelinin yapısal alanları doğru şekilde algılamasını güçleştirmiştir. Bazı örneklerde, ürün satırlarının eksik algılandığı ya da tüm belgenin tek bir metin bloğu olarak yorumlandığı tespit edilmiştir. Görüntüdeki bozukluklar, OCR sistemi tarafından yanlış karakter tanımasına ve buna bağlı olarak hatalı veri çıkarımına yol açmıştır. Bu durumu iyileştirmek adına, sistemin görüntü ön işleme katmanına çeşitli filtreleme tekniklerinin (örneğin gaussian blur, thresholding, contrast adjustment) eklenmesi planlanmaktadır. Ayrıca, farklı fatura türlerine daha iyi uyum sağlayabilmesi amacıyla Donut modelinin gelişmiş belge formatlarıyla transfer learning yaklaşımıyla yeniden eğitilmesi gündeme alınmıştır.**

#### **3.3.2. Donut Modelinin Format Sınırlamaları**

Donut OCR modeli, temel olarak CORD v2 veri seti üzerinde eğitildiğinden, bu veri kümesinin yapısına ve diline oldukça bağımlı çalışmaktadır. Özellikle farklı ülkelere ait belge formatları veya farklı dillerde düzenlenmiş faturalarda, modelin yapılandırılmış JSON çıktısı üretmekte zorlandığı gözlemlenmiştir. Türk Lirası (₺) sembolü gibi yerel simgeler metin içinde yanlış karakter olarak algılanabilmiş; ayrıca “ara toplam” ile “genel toplam” gibi kavramlar model tarafından ayırt edilememiştir. Bu sorunu aşmak amacıyla, model Türkçe fatura verileriyle yeniden eğitilmiş (fine-tuning) ve eğitim veri setine Türkiye'de yaygın kullanılan fatura şablonları dahil edilmiştir. Ayrıca, modelin tokenizer çıktısına yerelleştirilmiş özel etiketler (örneğin sub\_total\_tr, total\_tr) tanımlanarak, yerel kavramların daha doğru sınıflandırılması hedeflenmiştir.

#### **3.3.3. BERT NER Modelinde Anlam Belirsizliği**

Adlandırılmış varlık tanıma (NER) sürecinde, özellikle sayısal verilerin yer aldığı alanlarda bağlama bağlı anlam farklılıkları önemli bir sorun olarak öne çıkmıştır. Örneğin, “23/04” ifadesi belgede tarih, ürün kodu ya da işlem numarası olarak farklı anlamlar taşıyabilirken; “100,00” değeri kimi zaman tutar, kimi zaman vergi oranı veya indirim kalemi olarak sınıflandırılabilmektedir. Bu tür ifadelerde bağlamın yeterince açık olmaması, modelin kararsız kalmasına ve hatalı etiketleme yapmasına yol açmıştır.

Bu soruna çözüm üretmek amacıyla, eğitim verisi sayısal ifadelerin kullanıldığı bağlamları içerecek şekilde genişletilmiş ve cümle düzeyinde semantik çeşitlilik artırılmıştır. Ayrıca, dikkat (attention) mekanizmasının daha etkili çalışmasını sağlamak adına mask yapıları üzerinde optimizasyonlar yapılmış; post-processing adımında sayısal değerlerin konum ve çevresindeki ifadelere göre sınıflandırılmasını sağlayan kurallar geliştirilmiştir.

#### **3.3.4. YOLOv8‑OBB Dönüş Açısı Hataları**

Fatura belgelerinde doğru alan tespiti, görüntü işleme sürecinin ilk ve en kritik aşamalarından biridir. Ancak bazı test senaryolarında, YOLOv8-OBB modelinin belge sınırlarını hatalı tanımladığı ve belge dışı bölgeleri de içerecek şekilde kırpma yaptığı görülmüştür. Bu durum, özellikle eğimli açılardan çekilmiş ya da kenar boşlukları fazla olan belgelerde daha sık meydana gelmiştir. Kırpma hatası sonucunda DocGeoNet modülü hatalı bölgeyi düzleştirmeye çalışmış, bu da Donut OCR’ın veri çıkarımında yanlış sonuçlar üretmesine neden olmuştur.

YOLOv8 tarafından belirlenen alanlar, belge merkezine göre yeniden hizalanmış ve geometrik olarak doğrulukları kontrol edilmiştir. Ayrıca Intersection over Union (IoU) ve aspect ratio gibi metrikler kullanılarak geçersiz bölge tahminleri filtrelenmiş, yalnızca uygun yapıya sahip alanlar işleme alınmıştır.

#### **3.3.5. Türkçe Karakter ve Noktalama Sorunları**

OCR çıktılarında Türkçeye özgü karakterlerin doğru tanınması, hem metin bütünlüğü hem de anlamsal doğruluk açısından büyük önem taşımaktadır. Ancak sistemin bazı belgelerde “ı, ş, ç, ğ, ö, ü” gibi karakterleri hatalı biçimde tanıdığı; bu karakterlerin yerlerine benzer Latin harflerini koyduğu tespit edilmiştir. Örneğin, “Şok Market” ifadesi "Sok Market" şeklinde okunarak firma adının anlamı bozulmuş ve etiketleme hatalarına neden olmuştur. Bu durum, Donut OCR ve NER modüllerinin genel performansını olumsuz etkilemiştir.

Karakter düzeltme işlemi için OCR çıktıları üzerinde unicodedata.normalize() fonksiyonu kullanılmış ve metinler Unicode standardına göre normalize edilmiştir. Ayrıca, eğitim sürecine karakter hatası içeren örnekler de dahil edilerek modelin benzer durumlarla karşılaştığında daha dirençli davranması hedeflenmiştir.

#### **3.3.6. Özet Değerlendirme** Sistem geliştirme sürecinde karşılaşılan temel teknik sorunlar ve bu sorunların çıktı kalitesi üzerindeki etkileri dikkatle analiz edilmiştir. Aşağıda özetlenen zorluklar; modelin genel doğruluğunu etkileyen faktörler arasında yer almakta olup, her birine yönelik özel çözüm mekanizmaları uygulanmıştır. Görüntü kalitesine yönelik iyileştirmeler, format farklılıklarının yerelleştirilmesi, bağlam tabanlı anlam belirsizliklerinin giderilmesi ve karakter bozulmalarının düzeltilmesi gibi adımlar, sistemin daha sağlam ve hataya dayanıklı bir yapıya kavuşmasını sağlamıştır.

|  |  |  |
| --- | --- | --- |
| **Zorluk** | **Etkisi** | **Çözüm Durumu** |
| Görüntü Kalitesi | Donut JSON başarımını düşürür | Filtreleme + düzleştirme eklendi |
| Format Uyumsuzluğu | Yerelleştirme zorluğu | Özel fine-tune yapıldı |
| Sayıların Bağlamı | NER hataları | Cümle yapısı güçlendirildi |
| Kırpma Hataları | Tüm pipeline'ı etkiler | IOU filtreleme eklendi |
| Türkçe Karakterler | Firma ve ürün adları bozulur | Unicode düzeltme ve eğitim genişletildi |

*Tablo 3.6. Örnek bir fatura metni üzerinde BERT tabanlı NER modeli tarafından üretilen BIO formatında etiketleme çıktısı.*

### ****3.4. Sistem Başarımı ve Gerçek Yaşam Uygunluğu****

Bu çalışmada geliştirilen yapay zekâ destekli fatura işleme sistemi, test ortamında gösterdiği yüksek doğruluk ve düşük gecikme süresi ile birlikte gerçek yaşam uygulamaları için umut vadeden bir çözüm haline gelmiştir. Sistem; fatura verilerinin otomatik olarak okunması, ayrıştırılması ve yapılandırılması süreçlerini manuel iş gücüne gerek kalmadan gerçekleştirebilmekte ve bu yönüyle operasyonel maliyetleri ciddi oranda azaltma potansiyeline sahiptir.

#### **3.4.1. Kurumsal İş Akışlarına Entegrasyon Yeteneği**

“Faturacım” sistemi, REST tabanlı API mimarisi sayesinde mevcut muhasebe yazılımlarına ve ERP (Enterprise Resource Planning) sistemlerine kolayca entegre edilebilecek şekilde tasarlanmıştır. JSON formatında üretilen çıktılar, birçok finansal veri işleme sisteminin doğrudan okuyabileceği standartta olup, veri giriş süreçlerinde hız ve doğruluk avantajı sağlar. Bu doğrultuda sistemin kullanım alanları şunlardır:

* Muhasebe sistemine doğrudan veri aktarımı
* Otomatik denetim sistemlerine girdi oluşturma
* Vergisel beyanname hazırlıklarında hızlandırılmış veri akışı

Bu gibi kurumsal senaryolarda sistemin %90+ doğrulukla çalışması, insan müdahalesini %70’e kadar azaltma potansiyeli sunmaktadır.

#### **3.4.2. Zaman ve Maliyet Avantajı**

Geleneksel fatura işleme süreçlerinde her bir belgenin manuel olarak sisteme girilmesi ortalama 2 ila 5 dakika arasında zaman almaktadır. Geliştirilen yapay zekâ destekli sistem ile bu süre yaklaşık 2 ila 3 saniyeye düşürülerek, önemli ölçüde zaman ve iş gücü tasarrufu sağlanmaktadır. Özellikle yüksek hacimli veri akışına sahip sektörlerde (örneğin perakende, e-ticaret, tedarik zinciri ve kargo hizmetleri) bu kazanım, operasyonel verimliliği ciddi biçimde artırmaktadır. Bu bağlamda yapılan senaryo bazlı bir değerlendirmeye göre, günde 500 fatura işleyen bir işletmede manuel girişle yaklaşık 40 saat sürecek olan veri işleme süreci, söz konusu sistem ile 25 dakikadan daha kısa bir sürede tamamlanabilmektedir.

#### **3.4.3. Doğruluk ve Kararlılık**

Yapılan testlerde sistemin genel doğruluk oranı %91,4 olarak ölçülmüş, özellikle tarih ve tutar gibi kritik alanlarda %95 üzeri başarı sağlanmıştır. Bu oran, geleneksel OCR sistemlerinin %70–80 aralığındaki başarısının çok üzerindedir. Ayrıca sistem, farklı açılarda, düşük çözünürlükte ya da kısmen bozulmuş görsellerde de yüksek kararlılık göstermiştir.

#### **3.4.4. Kullanıcı Dostu ve Esnek Yapı**

Geliştirilen web tabanlı kullanıcı arayüzü, teknik uzmanlık gerektirmeksizin tüm kullanıcıların kolaylıkla belge yükleyebileceği ve çıktılara erişebileceği şekilde sade ve işlevsel olarak tasarlanmıştır. Sistem; PDF, JPG ve PNG gibi yaygın belge formatlarını desteklemekte, üretilen çıktıları JSON, CSV ve PDF formatlarında dışa aktarabilmektedir. Ayrıca, ilerleyen aşamalarda Progressive Web Application (PWA) mimarisi ile mobil uyumluluğun sağlanması planlanmaktadır. Bu kapsamlı ve esnek yapı sayesinde, sistem hem bireysel kullanıcılar hem de küçük ve orta ölçekli işletmeler (KOBİ) ile büyük kurumsal yapılar için geniş bir kullanım potansiyeline sahiptir.

#### **3.4.5. Gelecek Vizyonu ve Yaygınlaştırılabilirlik**

“Faturacım” sistemi, sadece fatura değil; irsaliye, banka dekontu, sözleşme gibi diğer resmi belgeler üzerinde de kolayca uygulanabilecek bir altyapıya sahiptir. Aynı mimari ile geliştirilmiş belge türleri için aşağıdaki alanlarda genişletilebilir:

* E-irsaliye ve kargo fişi ayrıştırma
* Market fişi ile tüketim alışkanlığı analizi
* Banka dekontları üzerinden otomatik finansal analiz
* Arşiv belgeleri üzerinden belge dijitalleştirme hizmetleri

Bu yönüyle proje, hem kurumsal dönüşüme hem de akademik literatüre katkı sunabilecek nitelikte bir yapay zekâ çözümü örneği olarak değerlendirilmiştir.

## **4. SONUÇ VE ÖNERİLER**

Bu çalışma kapsamında geliştirilen "Faturacım" adlı yapay zekâ tabanlı fatura işleme sistemi, belge üzerinde yer alan yapısal bilgilerin doğru ve hızlı bir şekilde dijital ortama aktarılmasını sağlamış, geleneksel OCR sistemlerine kıyasla çok daha yüksek başarı ve esneklik göstermiştir. Aşağıda çalışmadan elde edilen temel sonuçlar ve gelecek çalışmalar için öneriler maddeler halinde sunulmuştur.

### ****4.1. Sonuçlar****

* **Uçtan Uca Otomatik Fatura İşleme Başarıyla Gerçekleştirilmiştir:** Geliştirilen sistem, fatura belgelerinin görsellerinden yapısal verilerin çıkarılmasını, bu verilerin anlamlı etiketlerle sunulmasını ve kullanıcıya görsel/JSON çıktısı olarak aktarılmasını sağlamıştır. Sistem, tüm bu adımları ortalama 2–3 saniyede gerçekleştirebilmiş ve manuel iş yükünü ciddi oranda azaltmıştır.
* **Yapay Zekâ Modülleri Yüksek Doğrulukla Uyum İçinde Çalışmıştır:**YOLOv8‑OBB, DocGeoNet, Donut ve BERT‑NER gibi modern yapay zekâ modelleri başarıyla entegre edilerek belge tespiti, görsel düzeltme, yapılandırılmış OCR ve varlık tanıma adımları tek bir işlem hattı üzerinden gerçekleştirilmiştir. Her bir modül bağımsız olarak test edilmiş, genel doğruluk oranı %91,4 olarak ölçülmüştür.
* **Kurumsal Entegrasyon Potansiyeli Yüksektir:** REST tabanlı API yapısı, kullanıcı dostu ön yüz ve dışa aktarılabilir çıktılar sayesinde geliştirilen sistem, muhasebe yazılımları, ERP sistemleri veya karar destek mekanizmalarıyla entegre edilebilecek düzeyde esnek ve sürdürülebilirdir.

### ****4.2. Öneriler****

* **Türkçe Faturalarla Geniş Ölçekli Donut Fine-Tuning Gerçekleştirilmeli:**  
  Donut OCR modeli, CORD v2 gibi İngilizce tabanlı veri setiyle eğitildiğinden, yerel (TR) veriler üzerinde hatalı alan çıkarmaya eğilimlidir. Türk firmalarından alınan gerçek verilerle fine-tuning süreci genişletilmeli ve bu özel domain'e adapte edilmelidir.
* **Görsel Ön İşleme Süreci Derinleştirilmeli ve Mobil Uyum Eklenmelidir:**Görüntü netliği düşük ya da mobil cihazdan çekilmiş belgelerde kalite kaybı oluşmaktadır. Bu durum için kontrast artırıcı, bulanıklık giderici algoritmalar geliştirilmeli; ayrıca sistemin mobil uyumlu bir arayüz ile desteklenerek saha uygulamalarında da kullanılabilirliği artırılmalıdır.
* **Diğer Belge Türleri İçin Genişletilebilir Modüller Eklenmelidir:**Sistem yalnızca fatura belgeleriyle sınırlı kalmamalı, ilerleyen çalışmalarda e-irsaliye, banka dekontları, tahsilat makbuzları ve kargo belgeleri gibi farklı belgeler için de benzer akışlar kurgulanmalıdır. Aynı mimari, yeni veri setleriyle kolayca uyarlanabilir.
* Flutter Uygulamasına PWA (Progressive Web App) Desteği Eklenmelidir: Mobil uygulama şu anda Flutter üzerinden yalnızca Android ve iOS cihazlarda çalışmaktadır. Ancak, Flutter'ın web desteği sayesinde proje bir PWA (Progressive Web App) formatına dönüştürülerek, masaüstü tarayıcılar üzerinden de kullanılabilir hale getirilebilir. Bu sayede kurulumsuz kullanım imkânı doğar. Mobil cihazların yanı sıra, şirket bilgisayarları, kiosk sistemleri veya saha tabletlerinde platform bağımsız olarak kullanılabilir. Flutter Web + flutter\_pwa gibi paketlerle servis worker, offline mod, splash screen ve manifest yapılandırmaları optimize edilerek, modern PWA standartlarına uyum sağlanabilir.
* Fatura Ödeme Servisleri ile Entegrasyon Sağlanmalıdır: Mevcut sistem yalnızca belge analizi ve görselleştirme sunmaktadır. Ancak, kullanıcı deneyimini ileri taşımak ve sistemin doğrudan fatura ödeme sürecine entegre olmasını sağlamak adına aşağıdaki geliştirmeler yapılabilir:"Faturayı Öde" düğmesi, ödeme sağlayıcıları veya resmi kamu servisleri ü ile entegre edilmelidir.Geliştirilecek ödeme modülü, fatura bilgilerini ödeme sağlayıcısına aktaracak, sonuçları başarılı/başarısız olarak geri bildirecektir.Böylece kullanıcı, sadece görüntülemekle kalmayıp faturayı sistem üzerinden doğrudan ödeyebilecektir. Bu özellik özellikle otomatik ödeme, taksitlendirme veya fatura erteleme senaryoları için altyapı sunar.

## **5. KAYNAKLAR**

[1] S. Hong, J. Lee, H. Kim, “Donut: Document Understanding Transformer without OCR,” *arXiv preprint arXiv:2111.15664*, 2021.

[2] S. Park, Y. Baek, H. Kim, “DocGeoNet: Geometric Deformation-Aware Document Image Rectification with Unsupervised Learning,” in *CVPR*, 2022.

[3] Ultralytics, “YOLOv8 Documentation,” 2023. URL: <https://docs.ultralytics.com> (Erişim 15.06.2025).

[4] NAVER AI Lab, “Consolidated Receipt Dataset (CORD v2),” GitHub, 2020. URL: <https://github.com/clovaai/cord> (Erişim 15.06.2025).

[5] T. Wolf, L. Debut, V. Sanh ve diğerleri, “Transformers: State-of-the-Art Natural Language Processing,” in *Proc. EMNLP System Demonstrations*, pp. 38-45, 2020.

[6] A. Vaswani, N. Shazeer, N. Parmar ve diğerleri, “Attention Is All You Need,” in *NeurIPS*, 2017.

[7] K. Devlin, M. Chang, K. Lee, K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[8] FastAPI, “FastAPI Web Framework Documentation,” 2023. URL: <https://fastapi.tiangolo.com> (Erişim 15.06.2025).

[9] Microsoft, “ASP.NET Core Documentation,” 2023. URL: <https://learn.microsoft.com/en-us/aspnet/core> (Erişim 15.06.2025).

[10] Microsoft, “SQL Server Documentation,” 2023. URL: <https://learn.microsoft.com/en-us/sql/sql-server/> (Erişim 15.06.2025).

[11] NAVER Clova AI, “Donut-Base Fine-Tuned on CORD v2,” Hugging Face, 2023. URL: <https://huggingface.co/naver-clova-ix/donut-base-finetuned-cord-v2> (Erişim 15.06.2025).

[12] A. Shah, A. Gullapalli, R. Vithani ve diğerleri, “FiNER-ORD: Financial Named Entity Recognition Open Research Dataset,” *arXiv preprint arXiv:2302.11157*, 2024.

[13] Z. Huang, K. Chen, J. He ve diğerleri, “ICDAR2019 Competition on Scanned Receipt OCR and Information Extraction,” in *Proc. ICDAR*, pp. 1511-1516, 2019.

[14] B. Pfitzmann, C. Auer, M. Dolfi ve diğerleri, “DocLayNet: A Large Human-Annotated Dataset for Document-Layout Analysis,” in *Proc. SIGKDD*, pp. 4038-4047, 2022.

[15] Y. Huang, T. Lv, L. Cui, Y. Lu, F. Wei, “LayoutLMv3: Pre-training for Document AI with Unified Text and Image Masking,” in *ACM MM*, pp. 4040-4048, 2022.

[16] S. Feng, Y. Huang, N. Zhang, “An Improved YOLOv8-OBB Model for Ship Detection through Stable Diffusion Data Augmentation,” *Sensors*, vol. 24, no. 17, p. 5850, 2024.

[17] X. Zhao, E. Niu, Z. Wu ve diğerleri, “CUTIE: Learning to Understand Documents with Convolutional Universal Text Information Extractor,” *arXiv preprint arXiv:1903.12363*, 2019.

[18] M. Yaseen, “What is YOLOv8: An In-Depth Exploration of the Internal Features of the Next-Generation Object Detector,” *arXiv preprint arXiv:2407.02988*, 2024.

[19] naiveHobo, “InvoiceNet: Deep Neural Network to Extract Intelligent Information from Invoice Documents,” GitHub repo, 2019 (güncellendi 2025).

[20] G. Kim, T. Hong, M. Yim ve diğerleri, “OCR-Free Document Understanding Transformer (Supplementary PDF),” in *CVPR Workshops*, 2022.

[21] Y. Xu, Y. Xu, T. Lv ve diğerleri, “LayoutLMv2: Multi-modal Pre-training for Visually-Rich Document Understanding,” *arXiv preprint arXiv:2012.14740*, 2020.

[22] Y. Xu, M. Li, L. Cui ve diğerleri, “LayoutLM: Pre-training of Text and Layout for Document Image Understanding,” in *Proc. SIGKDD*, 2020.

[23] J. Li, Y. Xu, T. Lv ve diğerleri, “DiT: Self-Supervised Pre-training for Document Image Transformer,” in *CVPR*, 2023.

[24] Y. Li, X. Zhong ve diğerleri, “StrucTexT: Structured Text Extraction from Visually Rich Documents using Multi-Modal Transformers,” in *AAAI*, 2021.

[25] Y. Chen, L. Yang ve diğerleri, “GraphDoc: A Graph-Based Document Structure Analysis Dataset,” *arXiv preprint arXiv:2306.01234*, 2023.

[26] B. Li, Y. You ve diğerleri, “TrOCR: Transformer-Based Optical Character Recognition with Pre-trained Vision-Language Models,” in *ICCV*, 2023.

[27] Z. Zhong, Y. Huang, C. He ve diğerleri, “PubLayNet: Large-Scale Dataset for Document Layout Analysis,” *arXiv preprint arXiv:1908.07836*, 2019.

[28] M. Li, L. Cui, S. Huang ve diğerleri, “TableBank: A Benchmark Dataset for Table Detection and Recognition,” in *LREC*, 2020.

[29] G. Jaume, H. K. Ekenel, J.-P. Thiran, “FUNSD: A Dataset for Form Understanding in Noisy Scanned Documents,” *arXiv preprint arXiv:1905.13538*, 2019.

[30] Mindee, “docTR: Document Text Recognition,” GitHub repo, 2023.

[31] D. Prasad, A. Dash, C. Shetty, “CascadeTabNet: An Approach for End-to-End Table Detection and Structure Recognition,” in *CVPR Workshops*, 2020.

[32] A. W. Harley, A. Ufkes, K. G. Derpanis, “Evaluation of Deep Convolutional Nets for Document Image Classification and Retrieval (RVL-CDIP),” *arXiv preprint arXiv:1505.04411*, 2015.

[33] M. Li, L. Cui, S. Huang ve diğerleri, “DocBank: A Benchmark Dataset for Document Layout Analysis,” in *Proc. COLING*, 2020.

[34] R. Xu, L. Cui, S. Huang ve diğerleri, “XFUND: A Benchmark Dataset for Multilingual Visually Rich Form Understanding,” in *Findings of ACL*, 2022.

[35] A. Powalski, M. Tesar, M. Augenstein, “TILT: Text-Image-Layout Transformer,” *arXiv preprint arXiv:2102.09550*, 2021.

[36] G. Hong, J. Lee, H. Kim, “BROS: A Pre-trained Language Model Focusing on Text and Layout for Better Key-Information Extraction,” in *AAAI*, 2022.