Kapak Bilgileri

**Üniversite/Fakülte/Bölüm**: [Doldurulacak]

**Proje Başlığı**: [Doldurulacak]

**Öğrenci**: [Ad Soyad] — [ID]

**Danışman**: [Ad Soyad, Ünvan]

**Tarih**: [AA YYYY]

**Anahtar Kelimeler**: [LLM, Chatbot, Flask, Streamlit, Llama 3, Değerlendirme, Docker, Prometheus, Grafana]

──────────────────────────────────────────────────

# Özet

Bu çalışma, farklı boyutlardaki Llama 3 modellerinin (Llama 3.1-8b-instant ve Llama 3.3-70b-versatile) performans ve doğruluk açısından karşılaştırmalı analizini amaçlamaktadır. Proje kapsamında, matematik, Türkçe dil becerileri, kodlama, alan özgü sorular ve çeşitli özel senaryolar (uzun bağlam, tutarlılık, kısıtlı üretim vb.) kullanılarak kapsamlı bir değerlendirme yapılmıştır. Sonuçlar, 70B modelinin matematik ve Türkçe görevlerinde daha yüksek doğruluk oranına sahip olduğunu, ancak 8B modelinin daha deterministik ve format uyumlu çıktılar ürettiğini göstermiştir. Sistem mimarisi olarak Flask (backend) ve Streamlit (frontend) kullanılmış, dağıtım ve izleme için Docker, Prometheus ve Grafana entegre edilmiştir. Bu çalışma, LLM tabanlı chatbot sistemlerinde model seçimine yönelik kritik bir rehber sunmayı hedeflemektedir.

# Abstract

This study aims to conduct a comparative analysis of Llama 3 models (Llama 3.1-8b-instant and Llama 3.3-70b-versatile) in terms of performance and accuracy. The project evaluates these models across various domains, including mathematics, Turkish language skills, coding, domain-specific questions, and specialized scenarios (long-context, consistency, constrained generation, etc.). Results indicate that the 70B model achieves higher accuracy in mathematical and Turkish language tasks, while the 8B model produces more deterministic and format-compliant outputs. The system architecture utilizes Flask (backend) and Streamlit (frontend), with Docker, Prometheus, and Grafana integrated for deployment and monitoring. This work provides a critical guide for model selection in LLM-based chatbot systems.

# Kısaltmalar ve Semboller

**LLM**: Large Language Model

**API**: Application Programming Interface

- [Diğerleri]

──────────────────────────────────────────────────

# Giriş

**Problem Tanımı**: Büyük dil modellerinin performans ve doğruluk açısından karşılaştırılmasına yönelik sistematik bir çalışma eksikliği.

**Motivasyon**: Farklı model boyutlarının pratik uygulamalardaki etkinliğini anlamak ve kullanıcılar için en uygun model seçimine rehberlik etmek

**Amaç ve Katkılar**: Llama 3 modellerinin çok yönlü bir değerlendirmesini yaparak, performans, doğruluk ve maliyet açısından karşılaştırmalı sonuçlar sunmak.

**Kapsam ve Sınırlar**: Çalışma, Llama 3 modelleri ile sınırlıdır ve özel donanım gerektiren senaryoları kapsamaz.

**Bölüm Organizasyonu**: Çalışma, ilgili çalışmalar, sistem mimarisi, deneyler ve sonuçlar bölümlerinden oluşmaktadır.

# İlgili Çalışmalar ve Arka Plan

**LLM ve Sohbet Sistemleri**: Büyük dil modellerinin gelişimi ve sohbet sistemlerindeki uygulamaları.

**Model Değerlendirme Yöntemleri**: Matematiksel problemler, dil becerileri ve kodlama görevleri üzerinden değerlendirme.

**Ön Uç/Arka Uç Teknolojileri (Streamlit/Flask)**: Kullanıcı arayüzü ve API geliştirme için kullanılan teknolojiler.

**Gözlemlenebilirlik (Prometheus/Grafana)**: Sistem performansının izlenmesi ve görselleştirilmesi.

──────────────────────────────────────────────────

# Sistem Mimarisi

**Genel Bakış**: Sistem, kullanıcı etkileşimi, model çağrıları, değerlendirme ve izleme bileşenlerinden oluşur.

**Bileşenler**:

- Frontend: `frontend/streamlit\_app.py`

- Backend API: `backend/app.py`

- Değerlendirme: `backend/evals/` (ör. `suites/`, `results/`)

- Dağıtım: `docker-compose.yml`, `Dockerfile`, `nginx/nginx.conf`

- İzleme: `monitoring/prometheus.yml`, `monitoring/grafana/`

- Depolama/Günlükler: `uploads/`, `logs/`

## Mimari Diyagram

flowchart LR

U["Kullanıcı (Tarayıcı)"] --> S["Streamlit UI (frontend/streamlit\_app.py)"]

S --> A["Flask API (backend/app.py)"]

A -->|Model Çağrıları| M["LLM Sağlayıcıları (Llama 3.x)"]

A --> E["Değerlendirme Suite (backend/evals/)"]

A --> ST["Depolama (uploads/, logs/)"]

A --> P["Prometheus Exporter"]

P --> G["Grafana Dashboard"]

S <-->|Reverse Proxy| N["Nginx"]

mermaid

flowchart LR

U["Kullanıcı (Tarayıcı)"] --> S["Streamlit UI (frontend/streamlit\_app.py)"]

S --> A["Flask API (backend/app.py)"]

A -->|Model Çağrıları| M["LLM Sağlayıcıları (Llama 3.x)"]

A --> E["Değerlendirme Suite (backend/evals/)"]

A --> ST["Depolama (uploads/, logs/)"]

A --> P["Prometheus Exporter"]

P --> G["Grafana Dashboard"]

S <-->|Reverse Proxy| N["Nginx"]

### Veri Akışları  
  
- İstek Yanıt Akışı: Kullanıcıdan gelen istekler Streamlit arayüzü üzerinden Flask API'ye iletilir ve model yanıtları kullanıcıya döndürülür.  
- Oturum ve Bağlam Yönetimi: Kullanıcı oturumları ve bağlam bilgileri oturum boyunca saklanır.  
- Dosya/URL/TTS İşleme: Kullanıcı tarafından sağlanan dosya ve URL'ler işlenerek modele iletilir.  
  
---  
  
## Tasarım Kararları  
  
- \*\*Model Seçimi\*\*: llama-3.1-8b-instant (hafif ve hızlı) ve llama-3.3-70b-versatile (yüksek doğruluk) modelleri seçilmiştir.  
- \*\*Token Limit ve Guard Mekanizmaları\*\*: Token sınırlamaları ve güvenlik mekanizmaları uygulanmıştır.  
- \*\*Güvenlik ve Gizlilik\*\*: API anahtarları ve içerik filtreleri kullanılmıştır.  
- \*\*Performans Optimizasyonları\*\*: Önbellekleme ve akış (streaming) teknikleri uygulanmıştır.  
  
---  
  
## Uygulama Ayrıntıları  
  
### Backend (Flask API)  
  
- \*\*Temel Uçlar\*\*: GET/POST endpoint'leri ile sohbet ve değerlendirme işlevleri sağlanır.  
- \*\*İş Mantığı\*\*: Sohbet akışı, hata yönetimi ve loglama işlemleri gerçekleştirilir.  
- \*\*Dış Servis Entegrasyonları\*\*: LLM ve TTS servisleri ile entegrasyon sağlanmıştır.  
  
### Frontend (Streamlit)  
  
- \*\*Arayüz Bileşenleri\*\*: Mesaj listesi, model seçici ve ayarlar paneli bulunur.  
- \*\*Durum Yönetimi\*\*: Oturum durumu (session state) kullanılarak kullanıcı etkileşimi yönetilir.  
- \*\*Kullanıcı Deneyimi Detayları\*\*: Kod blokları ve temalar ile kullanıcı deneyimi geliştirilmiştir.  
  
### Değerlendirme (Evals)  
  
- \*\*Suite’ler\*\*: `backend/evals/suites/\*.jsonl` dosyaları ile test senaryoları tanımlanır.  
- \*\*Çalıştırma\*\*:

python backend/evals/run\_eval.py [opsiyonlar]

- \*\*Çıktılar\*\*: `backend/evals/results\*/` ve `overview.json` dosyaları ile sonuçlar raporlanır.   
  
### İzleme ve Loglama  
  
- \*\*Prometheus Metrikleri\*\*: Sistem performansı ve model yanıt süreleri izlenir.  
- \*\*Grafana Dashboard’ları\*\*: Metrikler görselleştirilerek analiz edilir.  
- \*\*Log Formatı ve Tutma Politikası\*\*: Loglar belirli bir formatta saklanır ve belirli bir süre sonra temizlenir.  
  
### Dağıtım  
  
- \*\*Yerel Çalıştırma\*\*:

python -m venv .venv && source .venv/bin/activate

pip install -r requirements.txt

pip install -r backend/requirements.txt -r frontend/requirements.txt

streamlit run frontend/streamlit\_app.py

- \*\*Docker Compose\*\*:

docker compose up --build

- \*\*Nginx ve SSL (Varsa)\*\*:   
  
---  
  
## Deneyler ve Değerlendirme  
  
- \*\*Araştırma Soruları\*\*: Hangi model daha yüksek doğruluk ve performans sunmaktadır?  
- \*\*Deney Tasarımı\*\*: Matematik, Türkçe, kodlama ve özel senaryolarda testler yapılmıştır.  
- \*\*Veri/Kriterler\*\*: Kolay ve zor seviyede sorular, uzun bağlam ve kısıtlı üretim senaryoları.  
- \*\*Metrikler\*\*: Doğruluk, tutarlılık, gecikme ve token kullanımı.  
- \*\*Sonuçların Sunumu\*\*:  
  
 | Model | Doğruluk (Matematik) | Doğruluk (Türkçe) | Doğruluk (Kodlama) | Ortalama Gecikme (ms)|  
 | ----------------| ---------------------| ------------------| -------------------| -------------------- |  
 | [Llama 3.1-8B] | [ 2/10] | [ 9/10] | [ 8/10] | [ 352] |  
 | [Llama 3.3-70B] | [4+/10] | [10/10] | [ 8/10] | [ 240] |  
  
 - Şekiller/Grafikler: [Ekran görüntüleri, Grafana panoları]  
 - Hata Analizi: [Yanlış örnekler ve nedenleri]  
  
---  
  
## Tartışma  
  
- \*\*Bulguların Yorumu\*\*: 70B modeli matematik ve Türkçe görevlerinde daha başarılıdır, ancak 8B modeli daha deterministiktir.  
- \*\*Güçlü Yönler/Zayıflıklar\*\*: 70B'nin yüksek doğruluğu, 8B'nin düşük maliyeti ve deterministik çıktıları.  
- \*\*Tehditler ve Sınırlar\*\*: Model boyutunun artmasıyla maliyet ve kaynak kullanımı artmaktadır.  
  
## Sonuç ve Gelecek Çalışmalar  
  
- \*\*Sonuç\*\*: 70B modeli yüksek doğruluk gerektiren uygulamalar için uygunken, 8B modeli düşük maliyetli ve deterministik çözümler için tercih edilebilir.  
- \*\*Gelecek Çalışmalar\*\*: Daha fazla modelin karşılaştırılması ve özel senaryoların genişletilmesi.  
  
---  
  
## Kurulum ve Kullanım (Özet)  
  
- \*\*Bağımlılıklar\*\*: `requirements.txt`, `backend/requirements.txt`, `frontend/requirements.txt`  
- \*\*Çevre Değişkenleri\*\*: [API anahtarları, yapılandırmalar]  
- \*\*Komutlar\*\*:

# Geliştirme

streamlit run frontend/streamlit\_app.py

# Değerlendirme

python backend/evals/run\_eval.py

# Dağıtım

docker compose up --build

---  
  
## Etik ve Güvenlik  
  
- \*\*Gizlilik ve Veri Koruma\*\*: Kullanıcı verileri şifrelenerek saklanır.  
- \*\*İçerik Güvenliği\*\*: Zararlı içerik filtreleme mekanizmaları uygulanmıştır.  
- \*\*Model Yanlılığı ve Riskler\*\*: Modellerin yanlılık potansiyeli değerlendirilmiştir.  
  
## Kaynakça  
  
[Standart atıf formatı kullanın. Örn. IEEE/APA.]  
  
## Ekler  
  
- \*\*Yapılandırma Dosyaları\*\*: `docker-compose.yml`, `nginx/nginx.conf`, `monitoring/\*`  
- \*\*Komut Çıktıları\*\*: Değerlendirme sonuçları ve rapor ekran görüntüleri.  
- \*\*Ek Tablolar/Grafikler\*\*: Performans ve doğruluk grafikleri.  
  
---  
  
## Revizyon Geçmişi (İsteğe Bağlı)  
  
- v0.1 – [Tarih] – İlk taslak  
- v0.2 – [Tarih] – [Değişiklik]