VERİ BİLİMİ DERSİ

Generative AI ve Prompt Mühendisliği

Hafta 14 · Modül 14
Büyük Dil Modelleri, Prompt Teknikleri ve RAG

6
Saat

4
Notebook

∞
Olasılık

Dr. Murat Altun

Veri Bilimi ve Yapay Zekâ Eğitimi · 2026

İçindekiler

Büyük Dil Modelleri (LLM)

LLM Nedir?

Milyarlarca parametreyle eğitilmiş, insan benzeri metin üreten devasa sinir ağları. Dil'i anlayan, çeviren, özetleyen ve yeni içerik üreten yapılar.

175B
GPT-3
Parametre

1.8T
GPT-4
Parametre (tahmin)

2T+
Gemini Ultra
Parametre

GPT (OpenAI)

ChatGPT'in arkasındaki model ailesi

Gemini (Google)

Multimodal: metin + görüntü + kod

Claude (Anthropic)

Güvenli, anayasal AI yaklaşımı

LLaMA (Meta)

Açık kaynak, topluluk odaklı

İpucu: Parametre sayısı = modelin öğrenme kapasitesi. Daha fazla parametre ≠ her zaman daha iyi sonuç. Verimlilik (efficiency) de önemli!

Transformer Mimarisi

"Attention Is All You Need" (2017)

Google'ın çığır açan makalesi. RNN/LSTM'nin sıralı işleme sorununu çözdü. Tüm kelimeleri aynı anda işleyerek paralel eğitim mümkün kıldı.

Temel fikir: Her kelime, cümledeki diğer tüm kelimelere "bakarak" bağlamını anlar.

Encoder

Girdi metnini anlamlı vektörlere dönüştürür

Decoder

Çıktı metnini token token üretir

RNN vs Transformer Karşılaştırması

	RNN / LSTM	Transformer
İşleme	Sıralı (yavaş)	Paralel (hızlı)
Uzun bağlam	Unutma sorunu var	Attention ile çözülür
Eğitim süresi	Uzun	Kısa (GPU paralelliği)

Self-Attention Mekanizması

Sorgu (Q) — Anahtar (K) — Değer (V)

Her kelime üç vektöre dönüştürülür: Sorgu ("neyi arıyorum?"), Anahtar ("ben neyim?"), Değer ("içeriğim ne?"). Sorgu ile Anahtar çarpılır → attention skoru → Değerler ağırlıklı toplanır. Böylece her kelime, hangi diğer kelimelere "dikkat etmesi" gerektiğini öğrenir.

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^T / \sqrt{d_k}) \times V$$

d_k = anahtar boyutu (normalizasyon için)

Multi-Head Attention

Birden fazla attention başı paralel çalışır. Her baş farklı bir ilişki türünü (sözdizimi, anlam, referans) öğrenir.

Paralel işleme

Tüm kelimeler aynı anda

Uzun mesafe

1000+ token bağlamı

Ölçeklenebilirlik

GPU'larda verimli

Transfer öğrenme

Pre-train + fine-tune

Prompt Engineering Nedir?

Prompt = LLM'e verilen talimat

LLM'lerden en iyi sonucu almak için talimatları (prompt'ları) sistematik olarak tasarlama sanatı ve bilimi. Küçük değişiklikler çıktıda dev farklar yaratabilir.

✗ Kötü Prompt

"Python'da bir şey yaz"

"Makine öğrenimi anlat"

"Veri analizi yap"

Belirsiz, bağlamsız, format belirtilmemiş

✓ İyi Prompt

"Python'da Pandas ile CSV oku, eksik verileri doldur, gruplama yap"

"Lojistik regresyonu, 5 maddelik liste halinde, örneklerle açıkla"

Spesifik, bağlamlı, format ve çıktı belirli

Zero-shot vs Few-shot Prompting

Zero-shot (Sıfır Örnek)

Modele hiç örnek vermeden doğrudan soru sorma. Model, önceden öğrendiği bilgilerle yanıt üretir.

Örnek: "Bu yorum olumlu mu olumsuz mu?
Yorum: Harika bir ürün!"

Few-shot (Birkaç Örnek)

Modele 2–5 örnek göstererek pattern öğretme. Model, örüntüyü kavrayıp devam eder.

Örnek: "Harika! → Olumlu
Berbat! → Olumsuz
Fırsat kaçmaz! → ?"

```
# Few-shot Prompt Örneğiprompt = """Metin sınıflandırma yapıyorsun. Örnekler:"Çok beğendim" → Olumlu"Kötü hizmet" → Olumsuz"İdare eder" → NötrŞimdi sınıflandır: "Mükemmel kalite, teşekkürler!" →"""
```

Chain-of-Thought (CoT) Prompting

Adım Adım Düşünme

"Adım adım düşün" eklediğinizde model, cevaba atlamak yerine mantık zinciri kurar. Matematik, kodlama ve mantıksal çıkarım görevlerinde %40–70 doğruluk artışı sağlar.

+40%
Matematik

+70%
Mantık

```
# Chain-of-Thought Prompt Örneğiprompt = """Soru: Bir mağazada 15 elma var. 8 tanesini sattılar,sonra 12 tane daha geldi. Kaç elma kaldı?Adım adım düşünerek çöz:1. Başlangıç: 15 elma2. Satılan: 15 - 8 = 7 elma3. Gelen: 7 + 12 = 19 elmaCevap: 19 elma"""
```

Sistem Prompt’u ve Persona

Sistem Prompt’u Nedir?

Chatbot’un kişiliğini, davranış kurallarını ve sınırlarını belirleyen gizli talimat. Kullanıcı görmez ama her yanıtı etkiler.

Veri Bilimci Asistan
Teknik, ölçümlü, kod odaklı

Türkçe Öğretmen
Sabrlı, teşvik edici, basit dil

SEO Uzmanı
Anahtar kelime odaklı, pragmatik

```
# Sistem Prompt Örneğisystem_instruction = """Sen bir veri bilimi eğitmenisin.Kuralların:1. Her açıklamaya bir Python örneği ekle2. Türkçe yanıt ver3. Bilmediğin konularda 'bilmiyorum' de4. Zararlı içerik üretme"""
```

Google Gemini API

1 API Key Al

Google AI Studio'dan
ücretsiz API anahtarı

2 SDK Kur

pip install
google-generativeai

3 Kullan

genai.configure() +
GenerativeModel()

1M+
Token
Kontext Penceresi

Ücretsiz
Başlangıç
Kotası

Multi
Metin+Görüntü
+Kod+Ses

2.5
Flash/Pro
Model Seçimi

⚠ **GÜVENLİK**

API key'i ASLA frontend koduna (VITE_, NEXT_PUBLIC_) veya GitHub'a eklemeyin! Supabase Edge Function veya backend .env dosyasında saklayın.

Gemini API — Temel Kullanım Kodu


```
# Gemini API — Temel Kullanımimport google.generativeai as genai# 1) API key
yapılandırgenai.configure(api_key="YOUR_API_KEY")# 2) Model seçmodel = genai.GenerativeModel("gemini-2.5-
flash")# 3) İçerik üretresponse = model.generate_content(    "Python'da liste comprehension'ı açıkla")# 4)
Sonucu yazdırprint(response.text)
```

Yapısal Çıktı — JSON Üretme

Neden Yapısal Çıktı?

LLM'den düz metin yerine JSON, CSV veya tablo formatında çıktı almak, veriyi programatik olarak işlemek için kritiktir. Gemini API'da response_schema ile garanti edilir.

```
// Beklenen JSON çıktı
{
  "konu": "Makine Öğrenimi",
  "özet": "Veriden öğrenen...",
  "seviye": "başlangıç"
}
```

```
# JSON çıktı almak için prompt tasarımıimport jsonprompt = """Aşağıdaki metni analiz et ve JSON döndür:{"duygu":
"olumlu/olumsuz/nötr", "güven": 0.0-1.0}Metin: 'Bu ürün gerçekten harika!'"""response =
model.generate_content(prompt)result = json.loads(response.text)print(result["duygu"])  # → olumlu
```

Kişisel Asistan Chatbot

BÖLÜM 3

Persona + System Instruction + Çok Turlu Konuşma

Chatbot'a kişilik verin, kural koyun, geçmiş mesajları hatırlasın. Gemini'nin chat() metodu çok turlu konuşmayı otomatik yönetir.

```
# Kişisel Asistan Chatbotimport google.generativeai as genaigenai.configure(api_key=API_KEY)model =
genai.GenerativeModel(    "gemini-2.5-flash",    system_instruction="Sen yardımcı bir veri bilimi asistanısın.")chat =
model.start_chat(history=[])while True:    user_input = input("Sen: ")    response = chat.send_message(user_input)
print(f"Asistan: {response.text}")
```

Blog Yazarı Uygulaması

SEO Uyumlu Blog Üretimi

LLM'e detaylı talimat vererek otomatik blog yazısı ürettirme:

- Konu ve hedef kitle belirle
- SEO anahtar kelimeleri listele
- Başlık, giriş, gelişme, sonuç yapısı
- Meta description ve etiketler

1 **Konu Girdisi**
"Yapay zekâ eğitimde"

2 **Prompt Şablonu**
SEO talimatları + format

3 **LLM Üretimi**
Blog yazısı + meta

Son Çıktı

```
# Blog Yazarı Prompt Tasarımıblog_prompt = f"""Konu: {konu} | Hedef Kitle: {hedef_kitle}SEO Anahtar Kelimeler: {anahtar_kelimeler}Görev: 800-1200 kelimelik, SEO uyumlu blog yaz.Format: Markdown, H2/H3 başlıklar, meta description ekle."""
```

RAG Nedir? (Retrieval-Augmented Generation)

Bilgi Tabanı + LLM = Doğru ve Güncel Yanıtlar

LLM’ler eğitim verisiyle sınırlıdır ve "halüsinasyon" yapabilir. RAG, önce ilgili dokümanları arar (Retrieval), sonra bu dokümanları prompt’a ekleyerek (Augmented) LLM’in doğru cevap üretmesini (Generation) sağlar.

1. Retrieval

Sorguyu vektör DB’de ara, en benzer dokümanları bul

2. Augmentation

Bulunan dokümanları prompt’a context olarak ekle

3. Generation

LLM, context + soru ile doğru cevap üretir

Sade LLM

Eski bilgi, halüsinasyon riski yüksek

RAG + LLM

Güncel, kaynak gösterebilen, doğru

Embedding ve Vektör Veritabanı

Metin → Vektör Dönüşümü

Her metin parçası, yüzlerce boyutlu bir sayı dizisine (vektör) dönüştürülür. Anlamca yakın metinlerin vektörleri de yakındır.

"kral" - "erkek" + "kadın" ≈ "kraliçe"

Bu ilişkiler, cosine similarity ile ölçülür.

ChromaDB	Açık kaynak, Python-native, kolay başlangıç
Pinecone	Bulut tabanlı, yönetilen hizmet, ölçeklenebilir
FAISS	Meta'nın kütüphanesi, çok hızlı, lokal
Weaviate	GraphQL API, hibrit arama

Cosine Similarity — Benzerlik Ölçümü

```
from sklearn.metrics.pairwise import cosine_similarity# embedding1 ve embedding2: iki metnin vektör
temsilleribenzerlik = cosine_similarity([emb1], [emb2])  # 0.0 - 1.0
```

Basit RAG Sistemi — Kod Örneği


```
# Basit RAG Sistemi — ChromaDB + Geminiimport chromadbimport google.generativeai as genai# 1) Dokümanları yükle ve embedding oluşturclient = chromadb.Client()collection = client.create_collection("dersler")collection.add(    documents=["Pandas ile veri analizi...", "Scikit-learn ile ML..."],    ids=["doc1", "doc2"])# 2) Kullanıcı sorusuyla en benzer dokümanı bulresults = collection.query(query_texts=["Veri nasıl analiz edilir?"], n_results=2)# 3) Bulunan dokümanları prompt'a ekleyip LLM'e göndercontext = "\n".join(results["documents"][0])prompt = f"Bağlam: {context}\n\nSoru: Veri nasıl analiz edilir?"response = model.generate_content(prompt)
```

Haftalık Notebook’lar

NB

gemini_api.ipynb

Gemini API’ye bağlanma, metin üretme, parametre ayarlama (temperature, top_p)

API · SDK · Temel

NB

kisisel_asistan.ipynb

Çok turlu chatbot oluşturma, sistem prompt’u, konuşma hafızası

Chat · Persona · Memory

NB

blog_yazari.ipynb

SEO uyumlu blog üretimi, prompt şablonu, Markdown çıktı, meta tag’ler

NLP · SEO · İçerik

NB

rag_giris.ipynb

ChromaDB kurulumu, embedding oluşturma, basit RAG pipeline, sorgulama

RAG · ChromaDB · Vektör

Ödev ve Kaynaklar

🎯 Hafta 14 Ödevi

1. Gemini API ile kişisel chatbot oluşturun (persona + 3 kural)
2. 10 farklı prompt tekniğini karşılaştırın (tablo formatında)
3. Basit RAG sistemi kurun: 5 doküman + soru-cevap

Teslim: Jupyter Notebook (.ipynb) formatında

📚 Kaynaklar

- Google AI Studio: aistudio.google.com
- Prompt Engineering Guide: promptingguide.ai
- LangChain Docs: python.langchain.com
- ChromaDB Docs: docs.trychroma.com
- Attention Is All You Need (2017 makalesi)
- Google Generative AI Cookbook
- OpenAI Prompt Best Practices
- Hugging Face NLP Course

Hafta 14 — Özet

1. LLM’ler Transformer mimarisi üzerine inşa edilir; Self-Attention her şeyi değiştirdi
2. Prompt mühendisliği: Zero-shot, Few-shot ve Chain-of-Thought ile sonuçlar %40–70 iyileşir
3. Gemini API ile Python’dan metin üretme, chatbot ve JSON çıktı almak 5 satır kod
4. Sistem prompt’u ile chatbot’a kişilik, kural ve sınır verilebilir
5. RAG = Retrieval + Augmentation + Generation — halüsinasyonu azaltır, doğruluğu artırır

“Yapay zekâ soru sormayı bilene cevap verir; prompt mühendisliği, doğru soruyu sorma sanatıdır.”

Dr. Murat Altun · Veri Bilimi ve Yapay Zekâ Eğitimi · 2026