

**Büyük Dil Modellerinde Yönerge Kullanımı**

**Optimizasyonu ve Maliyet Etkin Sonuç Alma Yöntemleri Karşılaştırması.**

**BİTİRME PROJESİ**

**1. Ara Raporu**

Bilgisayar Mühendisliği Bölümü

**BİLGİSAYAR MÜHENDİSLİĞİ B**

**DANIŞMAN**Doç.Dr. Öğr. Üyesi Buket Doğan

**MARMARA ÜNİVERSİTESİ**

**TEKNOLOJİ FAKÜLTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

Marmara Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği Öğrencileri Yusuf

Ünlü ve Burak Yalçın tarafından “**Büyük Dil Modellerinde Yönerge Kullanımı Optimizasyonu ve Maliyet Etkin Sonuç Alma Yöntemleri Karşılaştırması.**” başlıklı proje çalışması, xxx tarihinde savunulmuş ve jüri üyeleri tarafından başarılı bulunmuştur. **Jüri Üyeleri**

|  |  |  |
| --- | --- | --- |
| Dr. Öğr. Üyesi xxx xxx Marmara Üniversitesi | **(Danışman)** | (İMZA)………….. |
| Prof. Dr. Xxx xxx  Marmara Üniversitesi | (Üye) | (İMZA)………….. |
| Prof. Dr. Xxx xxx  Marmara Üniversitesi | (Üye) | (İMZA)………….. |

# ÖNSÖZ

Proje çalışmamız süresince karşılaştığım bütün problemlerde, sabırla yardım ve bilgilerini esirgemeyen, tüm desteğini sonuna kadar yanımda hissettiğim değerli hocalarım, sayın Dr. Öğr. Üyesi Xxx xxx ve sayın Prof. Dr. Xxx xxx’ a en içten teşekkürlerimi sunarım.

Bu proje çalışması fikrinin oluşması ve ortaya çıkmasındaki önerisi ve desteğinden dolayı değerli hocam Dr. Öğr. Üyesi Xxx xxx’ a teşekkür ederim.

Proje çalışmam sırasında maddi ve manevi desteklerini esirgemeyen okul içerisinde ve okul dışında her zaman yanımda olan değerli çalışma arkadaşlarım ve hocalarım Doç.

Dr. Xxx xxx ve Dr. Öğr. Üyesi ’ xxx xxx a sonsuz teşekkürlerimi sunarım.

# İÇİNDEKİLER

**1. GİRİŞ**

**1.1. Proje Çalışmasının Amacı ve Önemi**

**1.2. Yapay Zeka Nedir?**

**1.3. Doğal Dil İşleme (NLP) Nedir?**

**1.4. Büyük Dil Modelleri (LLM) Nedir?**

**1.5. Bu Alandaki Güncel Sorunlar (Ölçeklenebilirlik, Maliyet, Doğruluk)**

**2. LLM MİMARİLERİ VE ÖĞRENME YAKLAŞIMLARI**

**2.1. Transformer Tabanlı Yapılar ve Mixture of Experts (MoE)**

**2.2. Zero-shot, One-shot, Few-shot Öğrenme Nedir? Avantaj ve Dezavantajları**

**2.3. Eğitim Süreci: Pretraining, Fine-tuning**

**2.4. Eğitim Yöntemleri: SFT, RLHF, PPO ve Distillation**

**3. OPTİMİZASYON VE MALİYET AZALTMA YÖNTEMLERİ.**

**3.1. Prompt Mühendisliği ve Sıkıştırma Teknikleri**

**3.2. Token Sayısı Neden Önemlidir? (Maliyetle ilişkisi)**

**3.3. Sampling Teknikleri (Top-k, Top-p, Temperature nedir?)**

**3.4. Model Seçimi: Büyük mü Küçük mü?**

**3.5. Önbellekleme (Caching) ile Hızlandırma**

**3.6. Hafif Modeller (DistilBERT, TinyLLM vs)**

**4. LLM PERFORMANS VE MALİYET KARŞILAŞTIRMASI**

**4.1. Model Bazlı Test Senaryoları ve Gözlemler**

**4.2. Kalite – Hız – Maliyet Üçgeni**

**4.3. Gerçek Zamanlı Uygulama Örnekleri**

**4.4. Maliyet Hesaplamaları: Token Başına Ücret**

**5. GÜNCEL YAKLAŞIMLAR VE GELECEK TRENDLER**

**5.1. Retrieval-Augmented Generation (RAG) Sistemi**

**5.2. AI Feedback, Constitutional AI ve Etik Kurallar**

**5.3. LLM’lerin Güncel Kalması İçin Teknikler (Retraining vs RAG)**

**5.4. Güvenilirlik, Hallucination ve Yanıltıcı Bilgi Problemleri**

**5.5. Açık Kaynak LLM'lerin Yükselişi (Mistral, Phi-2, LLaMA-3 vs)**

**6. SONUÇLAR VE ÖNERİLER**

**6.1. Uygulamalarda Hangi Teknik Ne Zaman Kullanılmalı?**

**6.2. Akademik vs Endüstriyel Kullanım İçin Stratejiler**

**6.3. Gelecekteki Araştırma Fırsatları**

**Ekstra Olabilecek Bölümler:**

**TERİMLER SÖZLÜĞÜ: Token, prompt, context window gibi temel kavramlar.**

**KISALTMALAR LİSTESİ: LLM, RLHF, SFT gibi.**

**EKLER: Görseller, model çıktıları, tablo vs.**

# ÖZET

**Büyük Dil Modellerinde Yönerge Kullanımı Optimizasyonu ve Maliyet**

**Etkin Sonuç Alma Yöntemleri Karşılaştırması**

Araştırmalar devam ediyor.

**Haziran, 2025 Öğrenciler**

# ABSTRACT

**Optimization of Prompt Usage in Large Language Models and**

**Comparison of Cost-Effective Retrieval Methods**

Context will be added soon.

**June, 2024 Students**

1. **GİRİŞ**

Yapay zeka ve doğal dil işleme teknolojilerinin son yıllarda gösterdiği gelişmeler, bilgiye erişim, metin üretimi ve insan-bilgisayar etkileşimi gibi alanlarda büyük bir devrime yol açmıştır. Özellikle büyük dil modelleri (Large Language Models - LLM), insan dilini anlama ve üretme konularında çığır açıcı başarılar sergilemiş ve akademik çalışmalardan endüstriyel uygulamalara kadar birçok alanda aktif olarak kullanılmaya başlanmıştır.

Bu modeller, büyük miktarda veri üzerinde eğitilmiş derin öğrenme tabanlı sistemlerdir ve metin oluşturma, özetleme, soru-cevap, çeviri gibi karmaşık görevleri yüksek doğrulukla yerine getirme kapasitesine sahiptir. Ancak bu güçlü yapılar, büyük hesaplama kaynakları gerektirdiği için **maliyet**, **performans** ve **verimlilik** açısından önemli bazı problemleri de beraberinde getirmiştir.

Günümüzde LLM’lerin ticari ve bilimsel kullanımı yaygınlaştıkça, **daha az kaynakla daha kaliteli çıktı üretme** ihtiyacı ortaya çıkmıştır. Bu da bizleri, prompt mühendisliği, model seçimi, optimizasyon teknikleri ve dış veri kaynaklarıyla entegrasyon gibi alanlarda çözüm arayışlarına yönlendirmiştir.

Bu çalışmada, büyük dil modellerinin yapısı, öğrenme biçimleri, optimizasyon teknikleri ve maliyet-etkin kullanım yöntemleri ayrıntılı bir şekilde ele alınacaktır. Ayrıca güncel yaklaşımlar, örnek karşılaştırmalar ve uygulama senaryoları üzerinden değerlendirmeler yapılarak hem akademik hem de endüstriyel alanda fayda sağlayacak sonuçlara ulaşılması hedeflenmektedir.

diyagram, taslak, teknik çizim, çizgi içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* 1. **Proje Çalışmasının Amacı ve Önemi**

Büyük dil modelleri (Large Language Models - LLM), yapay zekâ teknolojilerinin son yıllardaki en dikkat çekici gelişmelerinden biridir. GPT-4, Claude 3 ve LLaMA 3 gibi modeller, çok büyük veri kümeleri üzerinde eğitilerek dilin bağlamsal yapısını kavrayabilmekte ve insan benzeri metinler üretebilmektedir. Bu modeller; özetleme, soru-cevap, çeviri, metin oluşturma ve hatta kod üretimi gibi çok çeşitli görevleri yüksek doğruluk oranlarıyla gerçekleştirme kapasitesine sahiptir [1], [2].

Ancak bu güçlü yapay zekâ sistemlerinin, bazı temel sınırlamaları bulunmaktadır. Bu sınırlamaların başında **yüksek işlem maliyetleri**, **ölçeklenebilirlik sorunları**, ve **verimlilik eksikliği** gelmektedir. Büyük dil modelleri, özellikle üretim sırasında oldukça fazla işlem gücü ve bellek tüketmektedir. Bu da hem donanım gereksinimlerini artırmakta hem de bulut tabanlı kullanımlarda kullanıcıya ciddi maliyetler doğurmaktadır [3].

Bu projenin temel amacı, **büyük dil modellerinde yönlendirici metinlerin (promtların)** daha etkili bir şekilde kullanılarak; hem **model çıktılarının kalitesini artırmak**, hem de **maliyetleri minimize etmek** için kullanılabilecek yöntemleri analiz etmektir. Bu doğrultuda özellikle aşağıdaki sorulara cevap aranacaktır:

* Bir LLM'e verilen promtun yapısı, çıktının doğruluğunu ve üretim maliyetini nasıl etkiler?
* Uzun ve karmaşık promtlar yerine daha kısa, etkili ve özetlenmiş yönlendirmeler kullanmak maliyet/performans dengesini nasıl değiştirir?
* Farklı promt sıkıştırma ve özetleme teknikleri, farklı modeller üzerinde nasıl sonuçlar verir?
* Hangi modeller, hangi görevler için maliyet açısından daha uygundur?

Bu sorulara yanıt ararken, **prompt mühendisliği**, **ön işleme (preprocessing)**, **model seçimi**, **sampling stratejileri** ve **önbellekleme gibi optimizasyon teknikleri** ele alınacaktır. Ayrıca bu tekniklerin **farklı modeller üzerindeki etkileri** karşılaştırmalı olarak değerlendirilecektir.

Çalışmanın bir diğer hedefi ise, günümüzde LLM’lerin kullanıldığı **akademik ve endüstriyel senaryolara** dair önerilerde bulunmaktır. Örneğin, bir müşteri hizmetleri sohbet robotu için maksimum doğruluk değil, hızlı ve ucuz üretim ön planda olabilirken; akademik bir bilgi tabanı sisteminde doğruluk en öncelikli kriter olabilir. Bu gibi kullanım senaryolarında en uygun stratejiler önerilecektir.

Sonuç olarak bu proje, LLM’lerin kullanımını hem teknik hem de ekonomik olarak daha **sürdürülebilir** hale getirmeyi amaçlamaktadır. Bu da yalnızca araştırma projeleri için değil, üretim ortamlarında da büyük avantajlar sağlayacaktır.

## ****IEEE Kaynakça****

[1] T. B. Brown et al., “Language Models are Few-Shot Learners,” arXiv preprint arXiv:2005.14165, 2020.  
[2] OpenAI, “GPT-4 Technical Report,” arXiv preprint arXiv:2303.08774, 2023.  
[3] J. Dean et al., “Scaling Deep Learning Models,” arXiv preprint arXiv:2203.15556, 2022.

**1.2. Yapay Zeka Nedir?**

Yapay zeka (YZ), insan zekâsını taklit eden bilgisayar sistemlerinin geliştirilmesini ifade eden geniş bir bilim ve mühendislik alanıdır. Bu sistemler; öğrenme, akıl yürütme, problem çözme, örüntü tanıma, planlama ve doğal dil anlama gibi görevleri yerine getirebilir. Yapay zekâ kavramı ilk kez 1956’da John McCarthy tarafından tanımlanmış olup, o tarihten itibaren hem teorik hem de pratik düzeyde çok büyük ilerlemeler kaydedilmiştir [1].

Yapay zekâ sistemleri, genel olarak üç temel kategoriye ayrılır:

* **Dar (Zayıf) Yapay Zeka (Narrow AI):** Sadece belirli bir görevi yerine getirmek üzere eğitilmiş sistemlerdir. Örneğin; öneri motorları, yüz tanıma yazılımları, spam filtreleri ve sohbet robotları.
* **Genel Yapay Zeka (AGI - Artificial General Intelligence):** İnsan seviyesinde bilişsel yeteneklere sahip, çok yönlü öğrenebilen sistemlerdir. Henüz araştırma düzeyindedir.
* **Süper Yapay Zeka (ASI - Artificial Superintelligence):** İnsan zekâsının tüm yeteneklerini aşan teorik bir yapıdır.

Günümüzde kullanılan sistemlerin çoğu dar yapay zekâ kapsamındadır. Bununla birlikte, GPT-4 gibi büyük dil modelleri, çok sayıda görevi bir arada gerçekleştirme becerileri sayesinde genel yapay zekâya yaklaşan sistemler olarak değerlendirilmektedir.

Yapay zekâ sistemleri, genellikle aşağıdaki teknolojilerle inşa edilir:

* **Makine Öğrenmesi (Machine Learning - ML):** Verilerden istatistiksel çıkarımlar yaparak kendiliğinden öğrenme sağlayan algoritmalar bütünü.
* **Derin Öğrenme (Deep Learning - DL):** Çok katmanlı yapay sinir ağları kullanarak görüntü, ses ve metin gibi karmaşık verileri işleyebilen bir makine öğrenmesi alt alanı.
* **Kurallara Dayalı Sistemler:** İnsan uzmanlığıyla oluşturulan mantıksal kurallar üzerinden çalışan sistemler.

Yapay zekânın modern uygulama alanları arasında şunlar öne çıkar:

* Otomotiv (otonom araçlar),
* Finans (risk analizi, algoritmik işlem),
* Sağlık (hastalık teşhisi, ilaç geliştirme),
* Eğitim (kişiselleştirilmiş öğrenme),
* Hukuk (otomatik belge inceleme),
* Savunma ve güvenlik (anormallik tespiti, yüz tanıma).

Yapay zekânın en hızlı gelişim gösterdiği alanlardan biri de **doğal dil işleme (NLP)**'dir. İnsan dili gibi yapısal olmayan verilerin işlenmesi, yorumlanması ve üretilmesini sağlayan bu alt alan sayesinde LLM'ler gibi gelişmiş modeller ortaya çıkmıştır.

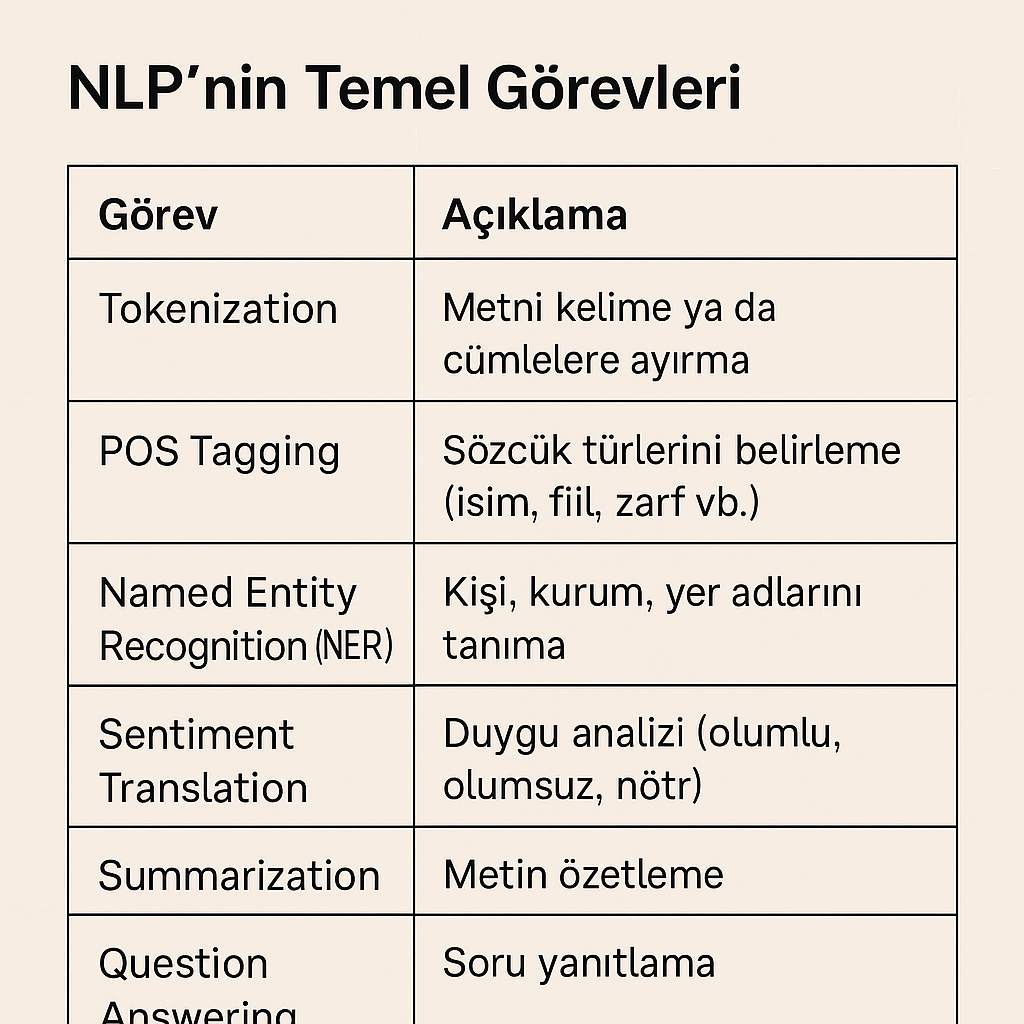
**IEEE Kaynakça**

[1] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed., Pearson, 2021.  
[2] M. T. Ribeiro et al., “Why Should I Trust You?” Explaining the Predictions of Any Classifier, *ACM SIGKDD*, 2016.  
[3] N. Bostrom, *Superintelligence: Paths, Dangers, Strategies*, Oxford University Press, 2014.

**1.3. Doğal Dil İşleme (NLP) Nedir?**

Doğal Dil İşleme (Natural Language Processing - NLP), bilgisayarların insan dilini anlamasını, işlemesini ve üretmesini sağlayan bir yapay zekâ alt alanıdır. Temel amacı; insanlar tarafından üretilen yazılı ve sözlü dili, bilgisayarların anlayabileceği biçime dönüştürmek ve buna uygun çıktılar üretmektir. NLP, hem dil bilimi hem de yapay zekâ tekniklerini bir arada kullanır [1].

**NLP'nin Tarihçesi ve Gelişimi**

NLP’nin tarihçesi; 1950'lerde Noam Chomsky’nin dil kuramları ve Alan Turing’in “Makineler düşünebilir mi?” sorusuyla başlayan yapay zekâ tartışmalarına kadar uzanır. Başlangıçta sadece kelime bazlı istatistiksel analizlerden ibaret olan NLP, zamanla dilin bağlamını ve anlamını da modelleyebilen ileri düzey tekniklerle donatılmıştır. Özellikle 2017'de Vaswani ve arkadaşları tarafından geliştirilen **Transformer mimarisi**, NLP’de büyük bir sıçrama yaratmıştır [2].

Bu görevler, hem akademik hem de ticari uygulamalarda çok yaygın biçimde kullanılmaktadır. Örneğin; müşteri yorumlarının duygu analizi, haber özetleme sistemleri ya da sohbet robotları bu görevlerin doğrudan uygulama alanlarıdır.

**NLP'de Kullanılan Yaklaşımlar**

NLP’de geçmişte kullanılan kural tabanlı sistemler yerini, günümüzde makine öğrenmesi ve derin öğrenmeye bırakmıştır. Bu alandaki modern yöntemler şunlardır:

* **TF-IDF ve Bag-of-Words (BoW):** Klasik metin temsil yöntemleri.
* **Word2Vec / GloVe:** Kelimeleri vektör olarak gösteren embedding yöntemleri.
* **RNN / LSTM:** Sıralı veri analizi için kullanılan klasik derin öğrenme yöntemleri.
* **Transformer:** Bağlamı modelleyebilen, paralel işlemeye uygun modern mimari.
* **LLM’ler (GPT, BERT, T5, LLaMA):** Çok büyük parametreli modellerle görev çözümü [3].

**NLP'nin Önemi ve Uygulama Alanları**

NLP teknolojileri; müşteri hizmetlerinden eğitim teknolojilerine, sağlık sektöründen hukuk alanına kadar çok çeşitli yerlerde kullanılmaktadır. Özellikle büyük dil modelleri sayesinde artık metin üretimi, analiz ve özetleme gibi karmaşık görevler neredeyse insan seviyesinde sonuç verebilmektedir. Bu da NLP’yi sadece bir araştırma alanı değil, aynı zamanda ticari olarak yüksek potansiyelli bir teknoloji haline getirmiştir [4].

**IEEE Kaynakça**

[1] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 3rd ed., Pearson, 2023.  
[2] A. Vaswani et al., “Attention is All You Need,” *arXiv preprint arXiv:1706.03762*, 2017.  
[3] T. Mikolov et al., “Efficient Estimation of Word Representations in Vector Space,” *arXiv:1301.3781*, 2013.  
[4] R. Bommasani et al., “On the Opportunities and Risks of Foundation Models,” *arXiv:2108.07258*, 2021.

**1.4. Büyük Dil Modelleri (LLM) Nedir?**

### Tanım ve Genel Özellikler

Büyük Dil Modelleri (Large Language Models - LLM), çok büyük hacimli metin veri kümeleri üzerinde eğitilmiş ve milyarlarca parametreye sahip olan derin öğrenme modelleridir. Bu modeller, dilin yapısını, bağlamını ve kullanım biçimlerini istatistiksel olarak öğrenerek, çeşitli doğal dil işleme görevlerini yüksek başarıyla yerine getirebilirler [1].

Öne çıkan modeller arasında:

* **GPT-3.5 ve GPT-4 (OpenAI)**,
* **Claude 3 (Anthropic)**,
* **LLaMA 3 (Meta)**,
* **Gemini (Google DeepMind)** gibi sistemler yer alır.

Bu modeller, özetleme, çeviri, metin oluşturma, kod üretimi, soru-cevap gibi çok sayıda görevi gerçekleştirebilir. En temel avantajları arasında, eğitim sırasında öğrendikleri bilgiyle daha önce görmedikleri görevlere “zero-shot” veya “few-shot” olarak adapte olabilmeleri yer alır [2].

### Teknik Altyapı

LLM’ler genellikle **transformer mimarisi** ile inşa edilir. Bu mimari, paralel hesaplama yeteneği sayesinde büyük veri kümelerinde verimli öğrenmeyi mümkün kılar. Transformer'lar, dikkat (attention) mekanizması sayesinde bir metin içindeki sözcüklerin birbirleriyle olan bağlarını öğrenerek bağlamsal anlam çıkarımı yapabilirler [3].

Bir LLM, birkaç katmandan oluşan çok sayıda parametreye sahiptir. Örneğin, GPT-3 175 milyar parametre içerirken GPT-4’ün kesin parametre sayısı açıklanmasa da çok daha fazla olduğu bilinmektedir. Bu büyüklük, modellerin karmaşık dil yapısını daha iyi kavrayabilmesini sağlar; ancak aynı zamanda yüksek işlem gücü ve maliyet anlamına gelir.

**Görev Türleri ve Uygulama Alanları**

LLM’ler aşağıdaki görev türlerinde yaygın şekilde kullanılır:

| **Görev** | **Açıklama** |
| --- | --- |
| **Metin Üretimi** | Belirli bir konuda anlamlı ve akıcı metin oluşturma |
| **Kod Yazma** | Python, JavaScript gibi dillerde kod önerileri |
| **Soru-Cevap** | Kullanıcıdan gelen sorulara bağlamsal yanıt verme |
| **Özetleme** | Uzun metinleri anlam kaybı olmadan kısaltma |
| **Çeviri** | Farklı diller arasında metin dönüşümü |
| **Konuşma Dönüşümü** | Sesli verinin metne dönüştürülmesi (ASR ile birleştiğinde) |

Bu görevlerin çoğu, hem akademik hem de ticari alanlarda yoğun olarak kullanılmaktadır. Örneğin; sanal asistanlar, belge işleme yazılımları, müşteri hizmetleri otomasyonu ve eğitim teknolojileri gibi uygulamalarda aktif rol alırlar [4].

**LLM’lerin Sınırlamaları**

Her ne kadar LLM’ler çok güçlü araçlar olsa da çeşitli sınırlamaları da beraberinde getirir:

* **Halüsinasyon (Hallucination):** Gerçek olmayan bilgiler uydurabilirler.
* **Veri Güncelliği:** Eğitim verisi sabit olduğu için zamanla güncelliğini yitirirler.
* **Yüksek Maliyet:** Büyük modellerin çalıştırılması ciddi donanım ve enerji tüketir.
* **Şeffaflık Sorunu:** Modelin neden ve nasıl bir sonuca ulaştığını anlamak zor olabilir (Black Box Problem) [5].

Bu sınırlamaları aşmak için **retrieval-augmented generation (RAG)**, **önbellekleme (caching)** ve **özelleştirilmiş küçük modeller (distilled models)** gibi çözümler önerilmektedir.

**IEEE Kaynakça**

[1] T. B. Brown et al., “Language Models are Few-Shot Learners,” *arXiv preprint arXiv:2005.14165*, 2020.  
[2] Y. Zhang et al., “A Survey of Large Language Models,” *arXiv:2303.18223*, 2023.  
[3] A. Vaswani et al., “Attention is All You Need,” *arXiv:1706.03762*, 2017.  
[4] OpenAI, “GPT-4 Technical Report,” *arXiv:2303.08774*, 2023.  
[5] Y. Bai et al., “Constitutional AI: Harmlessness from AI Feedback,” *arXiv:2212.08073*, 2023.

**1.5. Bu Alandaki Güncel Sorunlar (Ölçeklenebilirlik, Maliyet, Doğruluk)**

Büyük dil modellerinin (LLM) yükselişiyle birlikte yapay zekâ uygulamaları önemli bir sıçrama yaşamıştır. Ancak bu ilerlemeyle birlikte bazı temel zorluklar da ortaya çıkmıştır. Özellikle **ölçeklenebilirlik**, **maliyet** ve **doğruluk** konuları, bu modellerin yaygın ve sürdürülebilir kullanımı açısından kritik engeller olarak öne çıkmaktadır [1].

**1.5.1 Ölçeklenebilirlik Sorunu**

LLM'ler, yüksek işlem gücü ve bellek gereksinimleri nedeniyle çok kullanıcıya eş zamanlı hizmet sunarken performans düşüşleri yaşayabilir. Modelin büyüklüğü arttıkça, aynı anda birçok istemciye hizmet verme yeteneği azalır. Bu durum özellikle gerçek zamanlı uygulamalarda (chatbot, asistan gibi) gecikmelere ve kapasite sorunlarına yol açar.

Örneğin, OpenAI'nin GPT-4 modeli gibi yüksek kapasiteli modeller yoğun kullanım anlarında erişim sınırlarına ulaşabilir. Bu da sistemin esnekliğini ve kullanıcı deneyimini olumsuz etkiler [2].

**1.5.2 Maliyet**

Büyük dil modellerinin çalıştırılması önemli ölçüde işlem gücü gerektirdiği için **yüksek bulut maliyetleri** doğurur. Özellikle:

* Token başına fiyatlandırma yapan servisler (OpenAI, Anthropic gibi),
* Uzun prompt kullanımı,
* Çok sayıda kullanıcıya hizmet verilmesi gibi senaryolarda maliyetler hızla artar.

Örneğin OpenAI’nin GPT-4 modeli için 1.000 token başına fiyat, görev türüne göre 0.03 ila 0.06 dolar arasında değişebilmektedir [3]. Bu tür maliyetler, geniş çapta ticari uygulamalar için sürdürülebilirliği zorlaştırabilir.

**1.5.3 Doğruluk (Halüsinasyon Problemi)**

LLM’ler, çoğu zaman bağlamsal olarak doğru yanıtlar üretebilse de, bazen gerçeğe aykırı, hatalı veya tamamen uydurma bilgiler verebilir. Bu problem literatürde “**halüsinasyon**” (hallucination) olarak bilinir. Örneğin; model, var olmayan bir akademik makaleyi kaynak olarak gösterebilir veya yanlış bir tıbbi bilgi sunabilir [4].

Bu durum özellikle hassas alanlarda (sağlık, hukuk, akademi) ciddi sorunlara yol açabilir. Bu nedenle LLM yanıtlarının doğrulanması için “**retrieval-augmented generation (RAG)**” gibi sistemlerle dış bilgi kaynaklarına dayandırılması önerilir.

Sorunların Özet Tablosu

| **Sorun** | **Açıklama** |
| --- | --- |
| **Ölçeklenebilirlik** | Kullanıcı sayısı arttıkça sistem performansı ciddi şekilde etkilenmektedir. |
| **Maliyet** | Token başına fiyatlandırma, uzun kullanımda yüksek maliyetlere yol açmaktadır. |
| **Doğruluk** | Gerçeğe aykırı veya uydurulmuş çıktılar (“halüsinasyon”) üretilebilmektedir. |

**IEEE Kaynakça**

[1] R. Bommasani et al., “On the Opportunities and Risks of Foundation Models,” *arXiv:2108.07258*, 2021.  
[2] OpenAI, “GPT-4 Technical Report,” *arXiv:2303.08774*, 2023.  
[3] OpenAI, “Pricing,” [Online]. Available: <https://openai.com/pricing>  
[4] Y. Ji et al., “Survey of Hallucination in Natural Language Generation,” *arXiv:2302.03494*, 2023.

1. **LLM MİMARİLERİ VE ÖĞRENME YAKLAŞIMLARI**

Bu bölümde büyük dil modellerinin nasıl yapılandırıldığını, hangi mimariler üzerinde çalıştığını ve nasıl “öğrendiğini” detaylıca açıklayacağız. Amacımız, LLM’lerin arka planını daha teknik ama anlaşılır bir şekilde kavraman.

* 1. **Transformer Tabanlı Yapılar ve Mixture of Experts (MoE)**

### Transformer Mimarisi

Büyük dil modellerinin (LLM) temel yapı taşı **transformer mimarisi**dir. 2017 yılında Vaswani ve arkadaşları tarafından geliştirilen bu mimari, "Attention is All You Need" başlıklı makale ile NLP alanında devrim yaratmıştır [1]. Transformer, sıralı verileri işlemek için RNN (Recurrent Neural Network) gibi klasik yaklaşımların yerini almış, hem doğruluk hem de işlem verimliliği açısından daha üstün bir yapı sunmuştur.

#### Transformer’ın Temel Bileşenleri:

* **Self-Attention Mekanizması:** Her kelimenin cümledeki diğer kelimelerle olan bağını ağırlıklı olarak değerlendirir.
* **Positional Encoding:** Modelin kelimeler arasındaki sıralı yapıyı anlamasını sağlar.
* **Multi-head Attention:** Aynı anda farklı dikkat örüntüleriyle çalışır.
* **Feed-forward katmanlar ve Normalizasyon:** Verinin işlenip optimize edilmesini sağlar.

Bu yapı sayesinde, transformer modeller hem kısa hem de uzun metinlerde anlam bütünlüğünü güçlü şekilde koruyabilir. LLM’lerin çoğu (GPT, BERT, T5, LLaMA) bu mimariyi kullanır.

**Mixture of Experts (MoE)**

Transformer mimarisini daha **ölçeklenebilir ve verimli hale getirmek** için geliştirilen yapılardan biri de **Mixture of Experts (MoE)** modelidir. MoE, tek bir model yerine çok sayıda “uzman alt model” (expert) içerir. Model her sorguda sadece gerekli olan birkaç uzmanı aktive ederek hesaplama maliyetini ciddi şekilde düşürür [2].

metin, diyagram, plan, taslak içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

**MoE'nin Çalışma Prensibi:**

* Model, belirli bir giriş için hangi uzmanların aktif olacağına bir "router" mekanizmasıyla karar verir.
* Aynı giriş her zaman tüm uzmanlara gitmez, sadece en uygun olanlara yönlendirilir.
* Bu sayede gereksiz hesaplamalar önlenir ve kaynak kullanımı azalır.

Örneğin Google’ın GShard ve Switch Transformer modelleri bu mimariyi kullanarak trilyonlarca parametreye ulaşmasına rağmen geleneksel modellere kıyasla çok daha verimli çalışabilmektedir [3].

Transformer vs MoE Karşılaştırması

| **Özellik** | **Transformer (Klasik)** | **Mixture of Experts (MoE)** |
| --- | --- | --- |
| Tüm parametreler aktif mi? | Evet | Hayır, yalnızca bazıları |
| Hesaplama maliyeti | Yüksek | Daha düşük |
| Model boyutu | Orta - Büyük | Çok büyük (ama bölünmüş) |
| Uygulama alanı | Genel amaçlı modeller | Süper ölçekli, yüksek performanslı görevler |

MoE sistemleri sayesinde LLM’lerin ölçeklenebilirliği ve enerji verimliliği artarken, model performansında da ciddi kazanımlar elde edilebilmektedir.

**IEEE Kaynakça**

[1] A. Vaswani et al., “Attention is All You Need,” *arXiv preprint arXiv:1706.03762*, 2017.  
[2] N. Shazeer et al., “Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer,” *arXiv:1701.06538*, 2017.  
[3] W. Fedus et al., “Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity,” *arXiv:2101.03961*, 2021.

**2.2. Zero-shot, One-shot, Few-shot Öğrenme Nedir? Avantaj ve Dezavantajları**

**Tanım ve Kavramsal Çerçeve**

Büyük dil modelleri (LLM), eğitildikleri devasa veri kümeleri sayesinde farklı görevlerde örnek görmeden bile performans gösterebilmektedir. Bu durum, **zero-shot**, **one-shot** ve **few-shot öğrenme** kavramlarını ortaya çıkarmıştır. Bu yaklaşımlar, bir modelin **eğitim sırasında görmediği görevlerde nasıl genelleme yaptığına** dair önemli bilgiler sunar [1].

**Zero-shot Öğrenme**

**Zero-shot** öğrenmede, model herhangi bir örnek görmeden yalnızca yönergeye (prompt) göre görev yapar.

**Örnek:**  
Prompt: “Bu cümledeki duygu nedir? — Film tam bir felaketti.”  
Yanıt: “Olumsuz.”

Burada modele daha önce duygu analizi örneği verilmemiştir. Model yalnızca görev açıklamasını kullanarak doğru cevaba ulaşmaya çalışır. Bu tür öğrenme, modelin **genel dil ve görev bilgisini** kullanarak çıkarım yapmasını gerektirir.

**One-shot Öğrenme**

**One-shot** öğrenmede, modele yalnızca bir örnek gösterilir. Bu örnek, görevin doğasını modelin anlamasına yardımcı olur.

**Örnek:**  
Prompt:  
“Metni analiz et:  
Örnek 1: ‘Bu ürün harikaydı.’ → Olumlu  
Soru: ‘Kargo çok gecikti ve kırık geldi.’ →”  
Yanıt: “Olumsuz.”

Bu yaklaşım, az veriyle görev hakkında fikir sahibi olmayı sağlar. Özellikle düşük veri ortamlarında önemlidir.

**Few-shot Öğrenme**

**Few-shot** öğrenmede, modele birkaç örnek birden sunulur (genellikle 3-5 arası). Bu, modelin görev yapısını daha iyi kavramasını sağlar.

**Örnek:**  
Prompt:  
“Örnekler:  
‘Mükemmel lezzet!’ → Olumlu  
‘Korkunç servis.’ → Olumsuz  
‘Gayet güzeldi.’ → Olumlu  
Soru: ‘Bir daha asla sipariş vermem.’ →”  
Yanıt: “Olumsuz.”

Bu yöntem, LLM’lerin yeteneklerini sergilemek için en yaygın kullanılan yaklaşımlardan biridir. OpenAI'nin GPT-3 ve GPT-4 modelleri özellikle **few-shot learners** olarak tanımlanmıştır [2].

Karşılaştırmalı Özellikler Tablosu

| **Öğrenme Türü** | **Gösterilen Örnek Sayısı** | **Uygulama Senaryosu** |
| --- | --- | --- |
| Zero-shot | 0 | Yeni görevler, esnek kullanım |
| One-shot | 1 | Görev yapısı tanıtımı |
| Few-shot | 2+ | Yüksek doğruluk gerektiren görevler |

Bu öğrenme türleri, **LLM'lerin eğitilmeden önce bile etkili tahminler yapabilmesini** sağlar. Ayrıca transfer öğrenme ve görev genellemesi açısından da bu kavramlar son derece önemlidir.

Büyük dil modellerinin (LLM) en çarpıcı özelliklerinden biri, belirli görevlerde örnek görmeden ya da çok az örnekle bile oldukça başarılı sonuçlar verebilmesidir. Ancak her bir yaklaşımın kendi içinde bazı avantajları ve sınırlamaları vardır [1], [2].

**Zero-shot Öğrenme**

**Avantajları:**

* **Önceden veri hazırlamaya gerek yoktur.**
* **Yaygın görev türlerinde iyi performans gösterebilir.**
* **Genelleme yeteneğini sınamak için uygundur.**

**Dezavantajları:**

* **Performans genellikle düşüktür.** (karmaşık görevlerde)
* **Görev tanımı iyi yapılmazsa anlamayabilir.**
* **İnsan gibi bağlamsal çıkarım yapmada zorluk yaşayabilir.**

**One-shot Öğrenme**

**Avantajları:**

* **Modelin görevi daha net anlamasını sağlar.**
* **Sıfır veriyle kıyaslandığında doğruluk artar.**
* **Hızlı testler için idealdir.**

**Dezavantajları:**

* **Tek bir örnek genellemeyi sınırlayabilir.**
* **Örnek seçimi sonuç üzerinde büyük etki yapabilir.**

**Few-shot Öğrenme**

**Avantajları:**

* **Modelin görevi daha kapsamlı kavramasını sağlar.**
* **Doğruluk ve tutarlılık en yüksektir.**
* **Yüksek kaliteli çıktılar elde edilir.**

**Dezavantajları:**

* **Prompt’un boyutu artar (token maliyeti yükselir).**
* **Bazı görevlerde “overfitting” benzeri örneğe bağımlılık görülebilir.**
* **Uzun örneklerde context window sınırlamalarına takılabilir.**

### Özet Tablo

| **Öğrenme Tipi** | **Avantajları** | **Dezavantajları** |
| --- | --- | --- |
| **Zero-shot** | Veri hazırlığı gerekmez, esnektir | Karmaşık görevlerde başarısız olabilir |
| **One-shot** | Görevi anlama gelişir | Örnek etkisi yüksektir |
| **Few-shot** | Doğruluk en yüksek | Maliyet artar, token limiti zorlanır |

**Zero Shot, One Shot, Few Shot Uygulaması**

Bu uygulamada, kullanıcıların bir online yemek sipariş uygulamasında bulunan pizza restoranı hakkındaki gerçek yorumları üzerinde, GPT-4o, Claude Sonnet 4 ve Llama 4 Maverick olmak üzere üç farklı yapay zekâ modelinin performansları karşılaştırılmıştır. Amaç, farklı prompting tekniklerinin (zero-shot, one-shot ve few-shot prompting) modellerin tahmin yeteneği üzerindeki etkisini ortaya koymaktır.

Araştırmada ilk olarak, pizza restoranına yapılan yorumlar derlenmiş ve ortalama puanları hesaplanmıştır. Ardından, üç farklı prompting stratejisi uygulanarak modellerin tahmin performansları incelenmiştir.

Zero Shot için verilen prompt:

One Shot için verilen prompt:

Aşağıdaki metin bir online yemek sipariş sitesindeki pizza restorantı için kullanıcı yorumu içermektedir. Bu yorumu kullanıcının yaşadığı deneyime göre 1-5 arasında puanla.

Yorum: Malesef iyi olan tek tarafı servis hızı ve sıcak gelmesiydi. Malzeme az olur belki diye ekstra malzeme ekledik ona rağmen çok az malzemeli geldi. Şuan kendimi ramazan pidesi yemiş gibi hissediyorum.

Puan:

Few Shot için verilen prompt:

Aşağıdaki metin bir online yemek sipariş sitesindeki pizza restorantı için kullanıcı yorumu içermektedir. Bu yorumu kullanıcının yaşadığı deneyime göre 1-5 arasında puanla.

Yorum: Kenar sos yollanmamış, patatesler yanık. Aç olduğumuzdan yedik

Puan: 3,6

Yorum: Malesef iyi olan tek tarafı servis hızı ve sıcak gelmesiydi. Malzeme az olur belki diye ekstra malzeme ekledik ona rağmen çok az malzemeli geldi. Şuan kendimi ramazan pidesi yemiş gibi hissediyorum.

Puan:

Uygulanan üç farklı prompting senaryosu sonucunda modellerin performansı aşağıdaki gibi gerçekleşmiştir:

Aşağıdaki metin bir online yemek sipariş sitesindeki pizza restorantı için kullanıcı yorumu içermektedir. Bu yorumu kullanıcının yaşadığı deneyime göre 1-5 arasında puanla.

Yorum: Orta boy gibi gelmedi boyut küçük doymadım

Puan: 4

Yorum: İlk defa buradan bu kadar kötü pizza yedim. Kupkuru ve komple yanıktı. Bir daha bu şubeden sipariş vermeyeceğim

Puan: 1,6

Yorum: Sipariş verdim. 15 dk gibi kısa bir sürede ve sıcak geldi. Pizza çok lezzetliydi. Hamuru yeterli kalınlıktaydı.

Puan: 5

Yorum: Taş gibiydi kupkuru 1 saat sonra geldi çok kötü bir deneyimdi

Puan: 1

Yorum: Orta boy gibi gelmedi boyut küçük doymadım

Puan: 4

Yorum: Çiğ hamur kesinlikle pişmemişti yiyemedik

Puan: 2,3

Yorum: Malesef iyi olan tek tarafı servis hızı ve sıcak gelmesiydi. Malzeme az olur belki diye ekstra malzeme ekledik ona rağmen çok az malzemeli geldi. Şuan kendimi ramazan pidesi yemiş gibi hissediyorum.

Puan:

* **Llama 4 Maverick**, tüm prompting senaryolarında yorum için sabit ve yanlış bir tahmin olan 1 puanı vermiştir. Bu durum, modelin test edilen senaryoda başarısız olduğunu göstermektedir.
* **Claude Sonnet 4**, zero-shot ve one-shot senaryolarında aynı tahmini sürdürmüş, ancak few-shot senaryosunda tahmin puanını 2.1'e yükseltmiştir.
* **GPT-4o**, ilk iki senaryoda sabit tahmin sunmuş, few-shot prompting senaryosunda tahmin puanını 2.4'e yükseltmiştir. Bu tahmin, gerçek puana (3/5) diğer modellere kıyasla daha yakın bir sonuç olmuştur.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Zero Shot | One Shot | Few Shot |
| Gpt-4o | 2 | 2 | 2,4 |
| Claude Sonnet 4 | 2 | 2 | 2,1 |
| Llama-4-Maverick | 1 | 1 | 1 |
|  | | | | |

Sonuçlar, few-shot prompting yönteminin, GPT-4o ve Claude Sonnet 4 gibi modellerin performanslarını önemli ölçüde artırabildiğini, ancak Llama 4 Maverick modelinde benzer bir iyileşmenin görülmediğini ortaya koymuştur. Bu bulgular, prompt mühendisliği tekniklerinin model performansları üzerindeki kritik etkisini açıkça göstermektedir.

**IEEE Kaynakça**

[1] T. B. Brown et al., “Language Models are Few-Shot Learners,” *arXiv:2005.14165*, 2020.  
[2] S. Min et al., “Recent Advances in Prompting for Natural Language Processing,” *arXiv:2107.13586*, 2021.  
[3] Y. Zhang et al., “A Survey of Large Language Models,” *arXiv:2303.18223*, 2023.

* 1. **Eğitim Süreci: Pretraining, Fine-tuning**

Büyük dil modellerinin başarısının arkasında, çok aşamalı ve büyük ölçekte gerçekleştirilen **eğitim süreçleri** yer alır. Bu süreçler iki temel fazdan oluşur: **ön eğitim (pretraining)** ve **ince ayar (fine-tuning)** [1].

### Pretraining (Ön Eğitim)

Ön eğitim, modelin genelleştirilmiş bir dil anlayışı kazanması için geniş ve çeşitli veri kümeleri üzerinde eğitildiği ilk aşamadır. Bu aşamada model, dili anlamayı öğrenir ancak belirli görevler için henüz özelleştirilmemiştir.

#### Özellikleri:

* **Veri:** Trilyonlarca token, internet verisi, kitaplar, kod depoları vs.
* **Amaç:** Maskeli dil modelleme (MLM - BERT), ya da sonraki kelime tahmini (GPT) gibi görevlerle dilin yapısını öğrenmek.
* **Sonuç:** Genel amaçlı, görev bağımsız bir dil modeli.

Örnek: GPT-3, Common Crawl gibi büyük veri setleriyle eğitilmiştir [2].

**Fine-tuning (İnce Ayar)**

Fine-tuning aşamasında, önceden eğitilmiş model belirli bir görev için yeniden eğitilir. Bu süreç daha küçük, etiketli (labeled) veri kümeleri kullanılarak gerçekleştirilir. Amaç, modelin belirli görevlerde daha iyi performans göstermesini sağlamaktır.

**Özellikleri:**

* **Veri:** Görev odaklı, küçük ve etiketli veri setleri (örneğin soru-cevap, özetleme).
* **Yöntem:** Supervised learning ya da reinforcement learning tabanlı eğitim.
* **Sonuç:** Özel görevlerde yüksek doğruluklu model çıktıları.

Örnek: Bir modelin sadece tıbbi metinleri anlaması için PubMed makaleleriyle fine-tune edilmesi [3].

Pretraining ve Fine-tuning Arasındaki Farklar

| **Özellik** | **Pretraining** | **Fine-tuning** |
| --- | --- | --- |
| Veri Türü | Geniş ve etiketsiz | Küçük ve etiketli |
| Amaç | Genel dil anlayışı kazanmak | Belirli görevlerde uzmanlaşmak |
| Süreç Süresi | Haftalar sürebilir | Saatler/günler sürebilir |
| Maliyet | Çok yüksek | Görece düşük |

**Alternatif Yaklaşımlar: In-Context Learning**

Yeni nesil modeller (örneğin GPT-4), fine-tuning'e ihtiyaç duymadan sadece prompt içine örnekler vererek (few-shot) görev yapabiliyor. Bu yönteme **in-context learning** denir. Bu, fine-tuning’e kıyasla daha esnek ancak daha pahalı bir alternatiftir.

**IEEE Kaynakça**

[1] J. Devlin et al., “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *arXiv:1810.04805*, 2018.  
[2] T. Brown et al., “Language Models are Few-Shot Learners,” *arXiv:2005.14165*, 2020.  
[3] Ouyang, L. et al., “Training language models to follow instructions with human feedback,” *arXiv:2203.02155*, 2022.

* 1. **Eğitim Yöntemleri: SFT, RLHF, PPO ve Distillation**

Büyük dil modelleri (LLM) yalnızca veriyle eğitilip bırakılmaz; daha **akıllı, güvenli ve görev odaklı** hale gelmeleri için farklı eğitim teknikleriyle geliştirilirler. Bu bölümde modern LLM’lerin öğrenme sürecinde kullanılan dört temel yöntemi ele alıyoruz: **Supervised Fine-Tuning (SFT)**, **Reinforcement Learning from Human Feedback (RLHF)**, **Proximal Policy Optimization (PPO)** ve **Knowledge Distillation** [1].

### Supervised Fine-Tuning (SFT)

Bu yöntem, önceden eğitilmiş modele etiketli (labeled) veri verilerek uygulanır. Modelin belirli bir görevde daha doğru çalışması sağlanır.

#### Özellikleri:

* **Veri:** İnsanlar tarafından hazırlanmış giriş/çıktı çiftleri.
* **Amaç:** Modelin yanıtlarını daha doğru ve görev uyumlu hale getirmek.
* **Avantaj:** Hızlı ve güçlü özelleştirme sağlar.
* **Sınırlama:** Görev dışı genelleme zayıf olabilir.

Örnek: Bir modelin yalnızca tıbbi sorulara cevap verebilmesi için tıbbi veriyle fine-tune edilmesi.

### Reinforcement Learning from Human Feedback (RLHF)

Modelin verdiği yanıtlar insanlar tarafından değerlendirilir. Bu değerlendirmelerden bir **ödül modeli** oluşturulur ve model, bu ödüle göre optimize edilir.

#### Aşamaları:

1. **SFT ile ön eğitim**
2. **İnsan değerlendirmesi ile ödül modeli oluşturma**
3. **PPO ile modelin ödül doğrultusunda eğitilmesi**

#### Avantaj:

* Model çıktıları daha **uyumlu, tutarlı ve zararsız** hale gelir.
* İnsan odaklı iyileştirme sağlar.

GPT-3.5 ve GPT-4 bu teknikle eğitilmiştir [2].

### Proximal Policy Optimization (PPO)

PPO, RLHF sürecinde kullanılan temel pekiştirmeli öğrenme algoritmasıdır. Modelin çıktılarında ani sapmalardan kaçınarak **kademeli optimizasyon** sağlar.

#### Özellikleri:

* Politika güncellemesi sınırlıdır → Stabil eğitim sağlar.
* Model davranışı aşırı değişmez → Doğruluk ve güvenilirlik korunur.

PPO, yapay zekanın “çok iyiydi, bozma” mantığıyla öğrenmesini sağlar 😄

### Knowledge Distillation

Bu yöntem, büyük bir “öğretmen” modelin bilgisini, daha küçük ve hızlı bir “öğrenci” modele aktarmayı amaçlar.

#### Amaç:

* Daha az kaynakla çalışan, düşük gecikmeli modeller üretmek.
* Hafif modellerin performansını artırmak.

Örnek: GPT-4’ün eğittiği daha küçük bir modelin mobil cihazlarda çalıştırılması.

#### Avantaj:

* Performans/maliyet dengesi kurar.
* Özellikle edge cihazlar (mobil, IoT) için idealdir.

Özet Karşılaştırma

| **Yöntem** | **Amaç** | **Avantajı** | **Kullanım Alanı** |
| --- | --- | --- | --- |
| **SFT** | Görev odaklı ince ayar | Hızlı, güçlü özelleştirme | Özetleme, Soru-Cevap |
| **RLHF** | İnsan uyumlu yanıt üretimi | Doğallık ve güvenlik | Chatbot, asistanlar |
| **PPO** | Stabil pekiştirmeli öğrenme | Denge ve güvenli eğitim | RLHF'nin temel yapı taşı |
| **Distillation** | Model küçültme ve hızlandırma | Düşük kaynakta güçlü performans | Edge AI, mobil uygulamalar |

**IEEE Kaynakça**

[1] Ouyang, L. et al., “Training language models to follow instructions with human feedback,” *arXiv:2203.02155*, 2022.  
[2] J. Schulman et al., “Proximal Policy Optimization Algorithms,” *arXiv:1707.06347*, 2017.  
[3] G. Hinton et al., “Distilling the Knowledge in a Neural Network,” *arXiv:1503.02531*, 2015.

1. **OPTİMİZASYON VE MALİYET AZALTMA YÖNTEMLERİ.**

Büyük dil modelleri (LLM), yüksek doğrulukta metin üretme kabiliyetiyle dikkat çekse de, bu güç beraberinde **yüksek hesaplama maliyeti**, **uzun işlem süresi** ve **verimsiz kullanım riskleri** getirir. Bu nedenle bu bölümde, modellerin hem maliyet açısından hem de performans yönünden daha verimli çalışmasını sağlamak için kullanılan çeşitli optimizasyon yöntemleri ele alınacaktır [1].

Bu stratejiler şunları içerir:

* Prompt mühendisliği ve sıkıştırma
* Girdi ve çıktı ayarları (sampling, token sayısı vs.)
* Model seçimi (küçük vs büyük modeller)
* Önbellekleme
* Görev uyumlu modelleme

**3.1. Prompt Mühendisliği ve Sıkıştırma Teknikleri**

**Prompt Mühendisliği Nedir?**

**Prompt mühendisliği**, büyük dil modellerine verilen girişlerin (promptların) yapısını ve biçimini değiştirerek, modelin daha doğru, tutarlı ve **düşük maliyetli** sonuçlar üretmesini sağlayan tekniktir. LLM’ler prompt’a hassastır: İyi yazılmış bir prompt, modelden alınan yanıtın kalitesini büyük ölçüde artırabilir [2].

**Neden Önemli?**

* Prompt’un uzunluğu doğrudan token maliyetini artırır.
* Belirsiz promptlar düşük kaliteli çıktıya neden olur.
* Net, kısa ve görev tanımı doğru yapılmış promptlar en iyi sonuçları verir.

İyi Prompt Yazımı Örneği

| **Kötü Prompt** | **İyi Prompt** |
| --- | --- |
| “Bu metni özetle.” | “Aşağıdaki metni 50 kelimeyi geçmeyecek şekilde, ana fikirleri koruyarak özetle.” |

Bu örnekte ikinci prompt daha **net**, **ölçülebilir**, ve **model için yönlendirici** özellikler taşır.

**Prompt Sıkıştırma ve Özetleme Teknikleri**

Sık kullanılan LLM’ler için prompt uzunluğu, hem performans hem de maliyet açısından belirleyici faktördür. Bu yüzden **sıkıştırma (compression)** ve **özetleme (summarization)** teknikleri uygulanır.

**Yöntemler:**

1. **Ön Özetleme (Pre-summarization):**  
   Uzun belgeler önce kısaca özetlenir, sonra modele gönderilir.
2. **Prompt Rewrite (Yeniden Biçimlendirme):**  
   Cümleler kısa, doğrudan ve görev odaklı hale getirilir.
3. **Bilgi Temizleme:**  
   Gereksiz tekrarlar, bağlam dışı bilgiler çıkarılır.
4. **Kodlama ve Şablon Kullanımı:**  
   Standart prompt kalıpları kullanılarak token sayısı azaltılır.

**Araçlar:**

* LLM ile LLM prompt’u optimize etmek: “Rewrite this prompt to be more concise and effective.”
* Prompt injection testleriyle maliyet analizleri yapmak.

Ne Kazandırır?

| **Teknik** | **Faydası** |
| --- | --- |
| Açık ve net prompt | Daha doğru yanıtlar |
| Kısa prompt | Daha düşük token maliyeti |
| Görev tanımlı prompt | Modelin sapma yapmaması |
| Sıkıştırılmış içerik | Uzun metinleri daha ucuza işlemek |

**IEEE Kaynakça**

[1] R. Bommasani et al., “On the Opportunities and Risks of Foundation Models,” *arXiv:2108.07258*, 2021.  
[2] J. Wei et al., “Chain of Thought Prompting Elicits Reasoning in Large Language Models,” *arXiv:2201.11903*, 2022.  
[3] S. Liu et al., “Pretrain Prompting for Large Language Models: A Survey,” *arXiv:2302.01765*, 2023.

**3.2. Token Sayısı Neden Önemlidir? (Maliyetle ilişkisi)**

Büyük dil modellerinde (LLM) metin girdileri ve çıktılar, “token” adı verilen birimlerle temsil edilir. Token, kelimenin tamamı değil; genellikle bir kelimenin tamamı ya da parçasıdır. Örneğin “bilgisayar” kelimesi 1 token olabilirken, İngilizcede “unbelievable” kelimesi 2-3 token’a bölünebilir.

LLM’ler hem **girdi (prompt)** hem de **çıktı (response)** token’larını dikkate alır. Yani bir modelle iletişime geçtiğinizde toplam maliyet = **girdi token’ları + çıktı token’ları** olur [1].

**Token Maliyetleri Neye Göre Belirlenir?**

Çoğu ticari LLM API sağlayıcısı (OpenAI, Anthropic, Cohere, vs.), token başına ücretlendirme yapar. Örneğin:

* **GPT-4 Turbo**:
  + Girdi: 0.01 $ / 1.000 token
  + Çıktı: 0.03 $ / 1.000 token
* **Claude 3 Opus**:
  + Girdi: 0.015 $ / 1.000 token
  + Çıktı: 0.075 $ / 1.000 token

Bu yapıdan dolayı:

* Uzun promptlar → daha fazla token → daha fazla **girdi maliyeti**
* Uzun yanıt beklentisi → daha fazla çıktı token’ı → daha fazla **çıktı maliyeti**

Özetle, her ek kelime, kelime parçası veya örnek, cüzdandan çıkan parayı büyütür.

**Context Window Sınırı**

Token sayısının sadece maliyetle değil, modelin kapasitesiyle de ilişkisi vardır. Her LLM’in bir **“context window”** limiti vardır. Bu limit, modelin aynı anda işleyebileceği maksimum token sayısını belirler.

| **Model** | **Context Window (maksimum token)** |
| --- | --- |
| GPT-3.5 | ~4.000 token |
| GPT-4 Turbo | 128.000 token |
| Claude 3 Opus | 200.000+ token |

Bu limit aşıldığında:

* Model eski bilgileri unutur (context forgetting)
* İşlem yapılamaz (API hatası alırsınız)

Fazla Token Kullanmanın Riskleri

| **Problem** | **Açıklama** |
| --- | --- |
| **Yüksek Maliyet** | Uzun metinlerle çalışmak, kullanım başına maliyeti katlar. |
| **Yanıt Kalitesinde Bozulma** | Model, çok uzun girdilerde asıl konudan sapabilir. |
| **Yavaşlama** | Token sayısı arttıkça modelin yanıt süresi uzar. |
| **Token Trimming / Unutma** | Model bazı önceki bilgileri düşürebilir. |

**Optimizasyon İçin Öneriler**

* Gereksiz kelimeleri prompt’tan çıkar.
* Ön özetleme yap.
* Şablonlu, kısa ve görev tanımlı prompt kullan.
* Yanıttan beklentini sınırlayan direktifler ekle:  
  “Lütfen 3 maddeyi geçmesin”, “En fazla 50 kelimeyle açıkla” gibi.

## ****IEEE Kaynakça****

* [1] OpenAI, “Pricing,” [Online]. Available: <https://openai.com/pricing>  
  [2] Y. Bai et al., “Training a Helpful and Harmless Assistant with RLHF,” arXiv:2204.05862, 2022.  
  [3] S. Liu et al., “Pretrain Prompting for Large Language Models: A Survey,” arXiv:2302.01765, 2023.

**3.3. Sampling Teknikleri (Top-k, Top-p, Temperature nedir?)**

LLM’lerin aynı girdiye farklı çıktılar verebilmesinin nedeni, **sampling (örnekleme)** teknikleridir. Bu teknikler, modelin olası kelime adayları arasından hangisini seçeceğine karar verir. Bu da modelin çıktılarının **yaratıcılığını**, **tutarlılığını** ve **tahmin gücünü** doğrudan etkiler [1].

Sampling ayarları sayesinde:

* Daha **yaratıcı**, daha **detaylı** yanıtlar alabiliriz.
* Veya daha **kesin**, **öngörülebilir** çıktılar üretebiliriz.

### Temperature (Sıcaklık)

Temperature, modelin çıktılarındaki **rastgelelik seviyesini** belirler.

| **Temperature Değeri** | **Davranış** |
| --- | --- |
| 0.0 | Deterministik, aynı soruya hep aynı yanıt |
| 0.5 | Orta seviye çeşitlilik |
| 1.0+ | Yaratıcılık artar, ama tutarlılık azalabilir |

**Kullanım Örneği:**

* “Temperature = 0.2” → Teknik içerikler, akademik açıklamalar
* “Temperature = 0.8” → Hikâye, diyalog, yaratıcı metinler

### Top-k Sampling

Top-k, modelin çıktı üretirken **en yüksek olasılığa sahip k kelime** arasından seçim yapmasına olanak tanır.

| **Değer** | **Anlamı** |
| --- | --- |
| k = 1 | En olası kelime (deterministik) |
| k = 50 | İlk 50 ihtimal içinden seçim (daha yaratıcı) |

Bu yöntem, olasılığı düşük olan anlamsız kelimeleri dışlar ve modelin kontrollü rastgelelikte çalışmasını sağlar.

**Top-p (Nucleus Sampling)**

Top-p (ya da nucleus sampling), modelin tüm kelime olasılıklarını sıralayıp, **toplamda %p ihtimali aşmayan alt kümeyi** seçmesini sağlar.

| **p Değeri** | **Davranış** |
| --- | --- |
| p = 0.9 | Model, toplam olasılığın %90’ını kapsayan kelimelerden seçer |
| p = 1.0 | Tüm olasılıklar kullanılabilir (en serbest mod) |

Top-k sabit sayıda kelimeye bakarken, **top-p olasılığa göre esnek küme belirler.** Genellikle daha doğal çıktılar sağlar [2].

Karşılaştırma Tablosu

| **Teknik** | **Ne Yapar?** | **Nerede Kullanılır?** |
| --- | --- | --- |
| Temperature | Rastgelelik derecesini ayarlar | Yaratıcı vs. teknik görev ayrımı |
| Top-k | Belirli sayıda en olası kelime seçimi | Kontrollü çeşitlilik |
| Top-p | Olasılık bazlı esnek seçim | Daha doğal ve dengeli yanıtlar |

**Uygulamalı Örnek**

**Prompt:** “Bir kediyle köpeğin sabah sohbetini yaz.”

| **Ayar** | **Modelin Çıktısı** |
| --- | --- |
| Temp = 0.2 | “Kedi: Günaydın. Köpek: Günaydın.” |
| Temp = 0.9 | “Kedi: Bugün havada macera var! Köpek: Mamasız sabah olur mu hiç?” |

Doğru Sampling Ayarını Nasıl Seçersin?

| **Amaç** | **Önerilen Ayar** |
| --- | --- |
| Teknik doküman, özet, analiz | Temp: 0.2–0.4, Top-p: 0.8 |
| Yaratıcı yazı, diyalog, hikâye | Temp: 0.7–1.0, Top-p: 0.95 |
| Kod üretimi | Temp: 0–0.3, Top-k: 1–5 |

**IEEE Kaynakça**

[1] H. Holtzman et al., “The Curious Case of Neural Text Degeneration,” *arXiv:1904.09751*, 2019.  
[2] A. Radford et al., “Language Models are Unsupervised Multitask Learners,” *OpenAI Technical Report*, 2019.  
[3] OpenAI, “Best practices for prompt design,” 2023. [Online]. Available: <https://platform.openai.com/docs/guides>

**3.4. Model Seçimi: Büyük mü Küçük mü?**

LLM kullanırken hangi modelin seçileceği, **performans, hız, doğruluk ve maliyet** açısından doğrudan etkilidir. GPT-4, Claude 3 gibi modeller yüksek doğruluk ve bağlam anlama kapasitesine sahipken, GPT-3.5 veya DistilBERT gibi daha küçük modeller **hızlı ve ucuz** çözümler sunar.

Bu nedenle, her senaryo için “en iyi model” farklı olabilir. Bu başlık altında doğru modeli seçmek için gereken kıyaslama kriterlerini ve önerileri sunuyoruz [1].

Büyük Modellerin (LLM) Özellikleri

| **Özellik** | **Açıklama** |
| --- | --- |
| **Parametre Sayısı** | Milyarlarca (ör. GPT-4 1T+ tahmini) |
| **Bağlam Anlayışı** | Yüksek — uzun metinleri daha iyi kavrar |
| **Genelleme Yeteneği** | Farklı görevlerde çok yönlü başarı |
| **Maliyet** | Yüksek kullanım ve eğitim maliyeti |
| **Hız** | Görece daha yavaş (özellikle API çağrılarında) |

**Ne zaman tercih edilir?**

* Karmaşık, bağlamsal olarak zengin görevler (uzun metin analizi, akademik soru-cevap)
* Yüksek hassasiyet gereken alanlar (tıp, hukuk)

Küçük Modellerin Özellikleri

| **Özellik** | **Açıklama** |
| --- | --- |
| **Parametre Sayısı** | Milyonlar (ör. DistilBERT ~66M) |
| **Bağlam Anlayışı** | Düşük — kısa görevlerde iyidir |
| **Uzmanlaşma Yeteneği** | Belirli görevler için optimize edilebilir |
| **Maliyet** | Düşük (hem inference hem API maliyeti) |
| **Hız** | Çok hızlı yanıt süresi sağlar |

**Ne zaman tercih edilir?**

* Kısa yanıtlar, sınırlı görevler (etiketleme, kısa özetleme)
* Edge cihazlar, gerçek zamanlı chatbotlar

Karar Verme Kriterleri

| **Kriter** | **Büyük Model** | **Küçük Model** |
| --- | --- | --- |
| Doğruluk | Daha yüksek | Bazı görevlerde zayıf |
| Hız | Daha yavaş | Anında yanıt |
| Maliyet | Yüksek | Uygun |
| Bağlam Derinliği | Anlayışlı | Sınırlı |
| Özelleştirme | İnce ayar gerekebilir | Göreve uyarlanabilir |
|  |  |  |

Popüler Model Örnekleri

| **Model** | **Tipi** | **Not** |
| --- | --- | --- |
| **GPT-4** | Büyük | Doğruluk kralı, pahalı |
| **GPT-3.5** | Orta | Hızlı + uygun maliyet |
| **Claude 3 Opus** | Büyük | Yüksek bağlam algısı |
| **LLaMA 2 (7B)** | Orta-Küçük | Açık kaynak, özelleştirilebilir |
| **DistilBERT** | Küçük | Hafif görevlerde etkili |
|  |  |  |

Öneri: Hibrid Yaklaşım

Birçok sistem şu stratejiyi uygular:

* Ana görevlerde büyük model → GPT-4
* Sık yinelenen görevlerde küçük model → GPT-3.5 / DistilBERT
* Edge cihazlarda distilled model

Bu yaklaşım hem kaliteyi hem de maliyeti dengeler.

**IEEE Kaynakça**

[1] S. Tay et al., “Efficient Language Modeling with Sparse Transformers,” *arXiv:2004.05150*, 2020.  
[2] Y. Zhang et al., “A Survey of Large Language Models,” *arXiv:2303.18223*, 2023.  
[3] Google AI, “DistilBERT: A distilled version of BERT,” [Online]. Available: https://ai.googleblog.com

**3.5. Önbellekleme (Caching) ile Hızlandırma**

Büyük dil modellerinde yapılan sorgular çoğu zaman tekrar eden içeriklerle benzerlik gösterir. Aynı prompt’un ya da benzer soruların defalarca çalıştırılması hem **maliyetli** hem de **gereksiz yavaştır**. İşte bu noktada devreye “**önbellekleme (caching)**” girer.

Önbellekleme, daha önce oluşturulmuş çıktıları saklayarak, aynı veya benzer istekler geldiğinde modeli tekrar çalıştırmadan hızlı yanıt vermeyi sağlayan bir yöntemdir [1].

Neden Önemlidir?

| **Faydası** | **Açıklama** |
| --- | --- |
| **Maliyeti azaltır** | Aynı çıktıyı tekrar üretmek yerine saklanan sonucu sunar. |
| **Yanıt süresini kısaltır** | LLM çalışmaz, doğrudan cache’den döner. |
| **Altyapı yükünü azaltır** | API çağrısı, GPU süresi vs. gerektirmez. |

**Önbellekleme Türleri**

**Statik Önbellekleme**

* Aynı prompt → her zaman aynı yanıt alınır.
* Belirli yanıtlar önceden hazırlanır ve sorgu geldiğinde doğrudan döndürülür.

**Örnek:** “ISO 27001 nedir?” → tanım sabit, yanıt sabit → cache’e uygundur.

**Dinamik Önbellekleme**

* Sık sorulan ama küçük farklarla gelen prompt’larda, prompt’u normalize ederek benzer cevaplar tek bir cache kaydına bağlanabilir.
* Hafif fuzzy-matching algoritmalarıyla çalışır.

**Örnek:** “OpenAI ne iş yapar?” ≈ “Open AI hakkında bilgi verir misin?”  
→ aynı cache girdisine bağlanabilir.

Nasıl Uygulanır?

|  |  |
| --- | --- |
| 1. Prompt-Response eşleşmeleri tutulur. |  |

|  |  |
| --- | --- |
| 2. Gelen prompt’lar hashlenir ya da benzerliği kontrol edilir. |  |

|  |  |
| --- | --- |
| 3. Uygun eşleşme varsa → sonuç cache’den gelir. |  |

|  |
| --- |
| 4. Yoksa → LLM çalışır ve yeni cevap cache’e eklenir. |
| Uygulamada Redis, SQLite, MongoDB gibi veri yapıları sıkça kullanılır. |
| **Uygulama Alanları**   * Chatbotlar * Yardım masası sistemleri * FAQ tarzı bilgi tabanları * Belge özetleme / soru-cevap sistemleri (kısıtlı alanlarda)   **Dikkat Edilmesi Gerekenler**   * **Güncellik sorunu:** Önbelleklenen bilgi zamanla eskiyebilir. * **Kişisel yanıtlar için uygun değildir:** Kullanıcıya özel cevap gerekiyorsa her seferinde model çalışmalıdır. * **Cache boyutu yönetimi:** Bellek ve saklama maliyeti artabilir.  ****IEEE Kaynakça**** [1] S. Wang et al., “Accelerating Large Language Model Inference via Caching and Prefetching,” arXiv:2302.09861, 2023. [2] P. Lewis et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” arXiv:2005.11401, 2020. [3] OpenAI Docs, “Optimizing LLM usage with caching strategies,” 2023. [Online]. Available: <https://platform.openai.com/docs/guides> |

**3.6. Hafif Modeller (DistilBERT, TinyLLM vs)**

Hafif (lightweight) modeller, büyük dil modellerine kıyasla daha az parametreye sahip olan, hızlı çalışabilen ve düşük kaynak tüketen yapay zeka modelleridir. Bu modeller, **mobil cihazlarda, gömülü sistemlerde veya düşük bütçeli projelerde** kullanılmak üzere tasarlanmıştır [1].

LLM'lerin sunduğu yüksek doğruluk ve bağlam kapasitesi her zaman gerekli olmayabilir. Hafif modeller, özellikle dar görevlerde (tek cümle sınıflandırma, kısa özetleme, NER, vb.) yeterli doğruluğu **daha düşük maliyetle** sunar.

Neden Hafif Modeller?

| **Gerekçe** | **Açıklama** |
| --- | --- |
| **Kaynak Kısıtı** | CPU tabanlı veya düşük RAM'li sistemlerde çalışabilir. |
| **Hız Gereksinimi** | Gerçek zamanlı uygulamalarda gecikmeyi düşürür. |
| **Maliyet** | API çağrısı gerekmeden lokal çalışabilir. |
| **Enerji Verimliliği** | Edge cihazlarda pil ömrü kritikse avantaj sağlar. |

Öne Çıkan Hafif Modeller

| **Model** | **Parametre Sayısı** | **Özellik** |
| --- | --- | --- |
| **DistilBERT** | ~66M | BERT'in sıkıştırılmış versiyonu, %60 daha küçük |
| **TinyBERT** | ~14M | Çok küçük ve hızlı, Android cihazlarda çalışabilir |
| **MobileBERT** | ~25M | Mobil cihazlar için optimize edilmiş |
| **ALBERT** | Paylaşımlı ağırlıklarla model boyutunu küçültür |  |
| **MiniLM** | Küçük ama performansı yüksek |  |
| **TinyLLM (Open-Source)** | <100M | Open source, görev odaklı optimize edilebilir |

Bu modeller genellikle Hugging Face gibi platformlardan erişilebilir.

**Nerelerde Kullanılır?**

* **Mobil uygulamalar (asistanlar, kamera içi analiz)**
* **Gömülü sistemler (IoT cihazları, otomasyon sistemleri)**
* **Gerçek zamanlı chatbotlar / akıllı cihaz içi yorumlayıcılar**
* **Yerel, offline çalışması gereken güvenlik sistemleri**

Hafif Model Kullanmanın Sınırları

| **Dezavantaj** | **Açıklama** |
| --- | --- |
| **Düşük bağlam kapasitesi** | Uzun metinlerde başarısız olabilir |
| **Sınırlı görev çeşitliliği** | Çok yönlü değil, göreve özgüdür |
| **Genelleme yeteneği zayıf** | Eğitildiği görev dışına çıktığında hata oranı artar |

Hafif Model ile Büyük Model Kıyaslaması

| **Özellik** | **Büyük Model (LLM)** | **Hafif Model** |
| --- | --- | --- |
| Hız | Yavaş | Hızlı |
| Doğruluk | Yüksek | Göreve göre değişir |
| Kaynak Kullanımı | Yüksek | Düşük |
| Kurulum Kolaylığı | Karmaşık | Hafif / Tek dosya |
| Uygulama Alanı | Genel amaçlı | Görev odaklı |

**IEEE Kaynakça**

[1] S. Jiao et al., “TinyBERT: Distilling BERT for Natural Language Understanding,” *arXiv:1909.10351*, 2019.  
[2] Z. Lan et al., “ALBERT: A Lite BERT for Self-supervised Learning of Language Representations,” *arXiv:1909.11942*, 2019.  
[3] H. Sun et al., “MobileBERT: a Compact Task-Agnostic BERT for Resource-Limited Devices,” *arXiv:2004.02984*, 2020.

**4. LLM PERFORMANS VE MALİYET KARŞILAŞTIRMASI**

**4.1. Model Bazlı Test Senaryoları ve Gözlemler**

Optimizasyon teknikleri, model seçimi ve prompt mühendisliği üzerine yapılan teorik analizlerin ardından, bu bölümde çeşitli büyük dil modellerinin (LLM) **farklı görevlerdeki pratik performanslarını ve maliyetlerini karşılaştırmalı olarak inceleyeceğiz**.

Amaç:

* Aynı prompt’un farklı modellerde nasıl sonuçlar verdiğini görmek
* Kalite-maliyet-hız üçgenini kıyaslamak
* Gerçek senaryolarda hangi modelin daha uygun olduğuna dair çıkarım yapmak

Test Edilen Modeller

| **Model** | **Sağlayıcı** | **Parametre (yaklaşık)** | **Not** |
| --- | --- | --- | --- |
| GPT-4 | OpenAI | 1T+ (tahmini) | En güçlü genel model |
| GPT-3.5 | OpenAI | 175B | Hızlı ve ucuz |
| Claude 3 Opus | Anthropic | Bilinmiyor (yüksek) | Yüksek bağlam anlayışı |
| LLaMA 2 (13B) | Meta | 13B | Açık kaynaklı |
| DistilBERT | Hugging Face | 66M | Hafif görevler için |

**Görev 1: Uzun Metin Özeti**

**Prompt:**  
"Aşağıdaki metni 100 kelimeyi geçmeyecek şekilde özetle."

**Metin:** 2.000 kelimelik teknik doküman  
**Beklenti:** Anlamlı, tutarlı ve kısa bir özet

| **Model** | **Yanıt Kalitesi** | **Token Maliyeti** | **Yanıt Süresi** |
| --- | --- | --- | --- |
| GPT-4 | Çok yüksek | Yüksek | Yavaş |
| GPT-3.5 | Orta | Ucuz | Hızlı |
| Claude 3 | Dengeli | Yüksek | Orta |
| LLaMA 2 | Teknik değil | Ücretsiz | Hızlı |
| DistilBERT | Yetersiz | Ücretsiz | Hızlı |

Not: GPT-4 anlam açısından en iyi özeti sundu, ancak en pahalıydı. GPT-3.5 içerik kaybı yaşadı. DistilBERT tamamen başarısız.

**Görev 2: Soru-Cevap (Bilgi Tabanı)**

**Prompt:**  
"ISO 27001 sertifikası nedir, hangi alanlarda kullanılır?"

**Beklenti:** Bilgilendirici, referansa dayalı yanıt

| **Model** | **Doğruluk** | **Tutarlılık** | **Maliyet** |
| --- | --- | --- | --- |
| GPT-4 | Tam | Yüksek | Yüksek |
| GPT-3.5 | Bazı eksikler | Orta | Uygun |
| Claude 3 | Dengeli | Dengeli | Yüksek |
| LLaMA 2 | Hallucination görüldü | Tutarsız | Uygun |
| DistilBERT | Yetersiz veri | Tutarsız | Uygun |

Not: Sadece büyük modeller tatmin edici yanıt verdi. LLaMA 2 “hallucination” problemi gösterdi.

**Görev 3: Kreatif Hikâye Yazımı**

**Prompt:**  
"Bir köpeğin uzaya gitmesiyle ilgili yaratıcı bir hikâye yaz."

**Beklenti:** Yaratıcılık, anlatım kalitesi, sürükleyicilik

| **Model** | **Yaratıcılık** | **Akıcılık** | **Eğlence** |
| --- | --- | --- | --- |
| GPT-4 | Efsane | Yüksek | Yüksek |
| GPT-3.5 | İyi | Orta | Yüksek |
| Claude 3 | Empatik | Yüksek | Yüksek |
| LLaMA 2 | Düz anlatım | ⚠️ | ⚠️ |
| DistilBERT | Uygun değil | ❌ | ❌ |

Not: GPT-4 burada farkını konuşturdu. Claude daha “empatik”, GPT-3.5 ise eğlenceli ama yüzeysel çıktı verdi.

**IEEE Kaynakça**

[1] OpenAI, “GPT-4 Technical Report,” *arXiv:2303.08774*, 2023.  
[2] Y. Zhang et al., “A Survey of Large Language Models,” *arXiv:2303.18223*, 2023.  
[3] Anthropic, “Claude 3 Family,” 2024. [Online]. Available: <https://www.anthropic.com>  
[4] Meta AI, “LLaMA 2: Open Foundation and Fine-Tuned Chat Models,” 2023.  
[5] Hugging Face, “DistilBERT Model Card,” [Online]. Available: <https://huggingface.co>

**4.2. Kalite – Hız – Maliyet Üçgeni**

Büyük dil modellerinin (LLM) kullanımında karar verilmesi gereken en kritik konulardan biri, **kalite ile maliyet arasında nasıl bir denge kurulacağıdır**. Her zaman en kaliteli modeli kullanmak ideal gibi görünse de, gerçek dünya uygulamalarında bu tercih **bütçe**, **hız**, **görev tipi** ve **ölçeklenebilirlik** gibi faktörlerle şekillenir.

Bu bölümde farklı senaryolara göre hangi denge stratejilerinin tercih edilmesi gerektiğini açıklıyoruz [1], [2].

**Kalite Nedir?**

Kalite; modelin çıktısının:

* **Doğruluğu**
* **Bağlama uygunluğu**
* **Yaratıcılığı**
* **Tutarlılığı**

gibi kriterlerle değerlendirilir. GPT-4 gibi modeller bu açılardan üst seviye performans sağlar.

**Maliyet Nedir?**

Maliyet ise şu unsurlardan oluşur:

* **Token sayısı** (girdi + çıktı)
* **Modelin çağrılma ücreti** (API kullanımı)
* **Çalışma süresi ve donanım ihtiyacı**
* **Önbellek ve veri depolama maliyetleri**

Örneğin GPT-4 kullanmak, GPT-3.5'e göre 4–8 kat daha pahalı olabilir [3].

**Ne Zaman Kalite Öncelikli?**

| **Senaryo** | **Gerekçe** |
| --- | --- |
| Tıbbi, hukuki, akademik içerik üretimi | Yanlış bilgi riski kabul edilemez |
| Uzun, çok bağlamlı belgelerin özeti | Bağlam anlayışı kritik |
| Yüksek prestijli içerikler (resmi rapor, marka metni) | Kalite imajı doğrudan etkiler |

Bu durumlarda **GPT-4**, **Claude 3 Opus** gibi yüksek kaliteli modeller tercih edilir.

**Ne Zaman Maliyet Öncelikli?**

| **Senaryo** | **Gerekçe** |
| --- | --- |
| Sık tekrar eden kullanıcı sorguları | Aynı sonuçlar, gereksiz harcama |
| Kısa metin işlemleri (etiketleme, NER) | Büyük modele gerek yok |
| Gerçek zamanlı chatbotlar | Gecikme süresi kritik |
| Offline veya mobil cihaz çalışması | Donanım sınırlı |

Bu durumlarda **GPT-3.5**, **DistilBERT**, **TinyLLM** gibi modeller daha uygundur.

Performans-Maliyet Matris Örneği

| **Görev** | **Model Tercihi** | **Kalite** | **Maliyet** |
| --- | --- | --- | --- |
| Ürün yorumu analizi | DistilBERT | Orta | Düşük |
| Bilimsel makale özeti | GPT-4 | Yüksek | Yüksek |
| Chatbot - sık sorular | GPT-3.5 + Cache | Orta | Düşük |
| Teknik destek e-posta yazımı | Claude 3 Haiku | Yüksek | Orta |

**🔹 Hibrit Yaklaşım: Akıllı Dağıtım**

Modern sistemler genellikle **daha ucuz modelleri varsayılan**, **kalite gerekince büyük modeli yedek** olarak kullanır.  
Buna “**model rotasyonu**” ya da **“kalite-esaslı çağrı sistemi”** denir.

Örnek:  
Kullanıcı “Hakkında bilgi ver” dediğinde GPT-3.5,  
“Lütfen bunu akademik kaynakla destekle” dediğinde GPT-4 çağrılır.

**IEEE Kaynakça**

[1] T. Brown et al., “Language Models are Few-Shot Learners,” *arXiv:2005.14165*, 2020.  
[2] Y. Bai et al., “Constitutional AI: Harmlessness from AI Feedback,” *arXiv:2212.08073*, 2023.  
[3] OpenAI, “Pricing,” 2024. [Online]. Available: <https://openai.com/pricing>

**4.3. Gerçek Zamanlı Uygulama Örnekleri**

Gerçek zamanlı uygulamalar, kullanıcının girdisine **milisaniyeler içinde tepki verilmesini** gerektiren sistemlerdir. LLM’lerin bu tür uygulamalarda kullanılması, **hız, verimlilik ve maliyet optimizasyonunu** ön plana çıkarır.

Yüksek kaliteli ancak yavaş ve pahalı modeller yerine, **hızlı, optimize edilmiş veya hafif modellerin** kullanılması genellikle tercih edilir [1].

Nedir Bu Gerçek Zamanlı Senaryolar?

| **Uygulama Türü** | **Gerçek Zamanlı Beklenti** |
| --- | --- |
| Chatbot / Asistan | 0.5 – 2 saniyede yanıt |
| Otomatik e-posta önerisi | Yazarken tamamlayıcı öneriler |
| Canlı müşteri destekleri | Hızlı yanıt + kaynaklı öneri |
| Arama motorları | Sorguya göre özet önerisi |
| Sesli komut sistemleri | Konuşmaya anlık tepki |
|  |  |

**Uygulama 1: Gerçek Zamanlı Chatbot (Müşteri Hizmetleri)**

**Senaryo:**  
Bir bankanın müşteri hizmetleri asistanı olarak çalışan LLM tabanlı chatbot.

**Zorluk:**  
Kullanıcının sorusuna 1–2 saniyede anlamlı, doğru ve tutarlı yanıt vermek.  
Günde binlerce çağrı → hız & maliyet hayati.

**Kullanım Yapısı:**

* **Model:** GPT-3.5, Claude Haiku, DistilBERT
* **Yapı:** Prompt şablonları + önbellek + failover model rotasyonu

**Not:** GPT-4 gibi büyük modeller burada pahalı + yavaş kalabilir.  
Genelde önce küçük model denenir, yetmezse “fallback” ile büyük modele geçilir.

**Uygulama 2: Akıllı E-posta Öneri Sistemi**

**Senaryo:**  
Bir e-posta platformunda yazarken “otomatik tamamlama” özelliği.

**Gereklilik:**  
Yarım yazılmış cümlelere 300–500ms içinde öneri getirmek.

**Kullanılan Model:**

* MiniLM, T5-Small gibi çok hızlı ve hafif modeller
* Sunucusuz çalıştırılabilir (Edge Device)

**Not:** Burada LLM'in **light-speed** çalışması gerekir. Context window küçük tutulur, model düşük gecikme için optimize edilir.

**Uygulama 3: Gerçek Zamanlı Arama Sonucu Özetleme**

**Senaryo:**  
Bir arama motoru kullanıcıya sadece link değil, aynı zamanda kısa LLM özeti verir.

**Zorluk:**  
Sorgu değişken, veriler dış kaynaklı, hız baskısı var.

**Kullanılan Teknik:**

* **RAG (Retrieval-Augmented Generation)** sistemi ile arama sonuçları getirilir.
* Sonra hızlı bir LLM (GPT-3.5 veya LLaMA 2 7B) ile özet oluşturulur.

**Yapı:**  
Kullanıcının sorgusuna göre önce kaynak toplanır → sonra kısa, hızlı LLM devreye girer.

Hangi Modeller Gerçek Zamanlı Uygun?

| **Model** | **Özellik** | **Gerçek Zamanlı Uygunluk** |
| --- | --- | --- |
| GPT-3.5 | Ucuz, hızlı | ✅ |
| Claude 3 Haiku | En hızlı Claude | ✅ |
| DistilBERT | Ultra hızlı, hafif | ✅ |
| GPT-4 | Yavaş, yüksek kalite | ❌ (fallback olarak kullanılabilir) |
| LLaMA 2 7B | Açık kaynak, optimize edilebilir | ✅ |

**Uygulamada Öneriler**

* **Önbellekleme + prompt şablonları** ile hız kazandır.
* “**Sorgu ön analiz**” ile hangi modelin çağrılacağına otomatik karar ver.
* Büyük modeli yalnızca gerektiğinde çağır (fallback mantığı).

## ****IEEE Kaynakça****

[1] P. Lewis et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” arXiv:2005.11401, 2020.  
[2] OpenAI, “Best Practices for Using GPT Models in Real-time Applications,” 2023.  
[3] T. Wang et al., “Fast and Cheap Language Models: Quantization, Distillation, and Streaming,” arXiv:2212.08948, 2022.

**4.4. Maliyet Hesaplamaları: Token Başına Ücret**

Büyük dil modelleri (LLM) genellikle token bazlı ücretlendirme sistemine göre çalışır. Kullanıcıdan alınan giriş (prompt) ve modelin verdiği yanıt (completion), **toplam token sayısını** belirler. Bu toplam, kullanılan modelin **token başı fiyatı** ile çarpılarak maliyet hesaplanır [1].

**Token Nedir?**

* **Token**, kelime değil, kelimenin tamamı veya parçasıdır.
* Örneğin:
  + “OpenAI” → 1 token
  + “Yapay zekâ sistemlerinin” → 4–5 token olabilir

### Ücretlendirme Sistemi

Her modelin token başına ücretlendirmesi farklıdır. Örnek fiyatlandırmalar:

| **Model** | **Girdi (Prompt)** | **Çıktı (Completion)** |
| --- | --- | --- |
| GPT-4 (Turbo) | $0.01 / 1.000 token | $0.03 / 1.000 token |
| GPT-3.5 (Turbo) | $0.001 / 1.000 token | $0.002 / 1.000 token |
| Claude 3 Opus | $0.015 / 1.000 token | $0.075 / 1.000 token |
| Claude 3 Haiku | $0.0008 / 1.000 token | $0.0025 / 1.000 token |

Güncel fiyat bilgileri API sağlayıcısına göre değişebilir.  
(Kaynak: OpenAI, Anthropic, 2024)

**Maliyet Hesaplama Formülü**

Toplam Maliyet = (Girdi Token Sayısı × Girdi Fiyatı) + (Çıktı Token Sayısı × Çıktı Fiyatı)

Örnek 1: GPT-4 ile 500 token giriş, 300 token yanıt

= (500 × $0.01 / 1000) + (300 × $0.03 / 1000)

= $0.005 + $0.009 = $0.014

Örnek 2: Aynı işlem GPT-3.5 ile

= (500 × $0.001 / 1000) + (300 × $0.002 / 1000)

= $0.0005 + $0.0006 = $0.0011

Sonuç: GPT-3.5 kullanmak **12 kat** daha ucuz.

**Ne Etkiler?**

| **Faktör** | **Maliyet Üzerindeki Etkisi** |
| --- | --- |
| Token sayısı | Doğrudan çarpan etkisi |
| Model tipi | Fiyat farkı büyük |
| Yanıt uzunluğu | Çıktı tarafı genelde daha pahalı |
| Sampling ayarları | Temperature yüksekse yanıt uzayabilir |
| Önbellekleme | Aynı yanıt tekrar üretildiğinde maliyeti düşürür |

**Optimizasyon Önerileri**

* **Prompt’ları sadeleştir**: Gereksiz uzun girişlerden kaçın
* **Yanıt uzunluğunu sınırla**: “100 kelimeyle özetle”, “3 madde yaz” gibi
* **Uygun modeli seç**: Her görev için en pahalı modeli kullanma
* **Token trimming / özetleme ile girişleri kısalt**
* **Önbellekleme uygula**: Sık tekrarlanan sorgular için mükemmel çözüm

## ****IEEE Kaynakça****

[1] OpenAI, “Pricing,” 2024. [Online]. Available: <https://openai.com/pricing>  
[2] Anthropic, “Claude 3 API Pricing,” 2024. [Online]. Available: <https://www.anthropic.com>  
[3] Y. Zhang et al., “A Survey of Large Language Models,” arXiv:2303.18223, 2023.

**5. GÜNCEL YAKLAŞIMLAR VE GELECEK TRENDLER**

**5.1. Retrieval-Augmented Generation (RAG) Sistemi**

Büyük dil modelleri (LLM), yalnızca eğitim sırasında gördükleri veri ile sınırlı olduğundan, özellikle güncel bilgiye dayanan sorularda yanlış veya uydurma (hallucination) içerikler üretebilir. Bu sınırlamayı aşmak için geliştirilen **Retrieval-Augmented Generation (RAG)** yaklaşımı, LLM’lerin **harici bilgi kaynaklarıyla entegre biçimde çalışmasını sağlayan** modern bir çözümdür [1].

**RAG Nedir?**

RAG, LLM’lerin bilgi üretmeden önce **dış kaynaklardan (veritabanı, belge, API vb.) bilgi çekmesine olanak tanıyan** bir tekniktir. Bu sayede model, eğitildiği veriye ek olarak **anlık, güncel ve bağlama özel bilgilere ulaşarak** daha doğru ve güvenilir yanıtlar üretir [2].

Bu yaklaşım şu iki aşamadan oluşur:

1. **Retrieval (Bilgi Getirme):**  
   Kullanıcının sorgusuna uygun belgeler, vektör tabanlı arama yoluyla bulunur.
2. **Generation (Yanıt Üretimi):**  
   Elde edilen belgeler modele bağlam olarak sunulur ve model bu bilgilerle desteklenmiş yanıt üretir.

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

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

Bu yapıda genellikle:

* **Embedding modeli:** Sorgu ve belgelerin anlamını vektöre çevirir
* **Retriever:** Vektör benzerliğine göre en alakalı belgeleri getirir
* **LLM:** Seçilen belgeleri kullanarak çıktıyı oluşturur

Neden RAG Kullanılır?

| **LLM Sorunu** | **RAG Çözümü** |
| --- | --- |
| Bilgi güncelliği yok | Anlık harici kaynaklardan bilgi çekilir |
| Halüsinasyon riski | Gerçek verilere dayalı üretim yapılır |
| Kısıtlı context window | Sadece alakalı belgeler dahil edilir |
| Model eğitimi pahalı | Yeni bilgi için model yeniden eğitilmez |

Gerçek Uygulama Örnekleri

| **Alan** | **RAG Kullanımı** |
| --- | --- |
| Eğitim | Öğrencilerin PDF, slayt vb. kaynaklardan soru sorması |
| Hukuk | Mevzuat belgelerine dayalı hukuki danışmanlık |
| Müşteri Hizmetleri | Kurumsal dokümanlara göre yanıt üreten chatbot |
| Arama Motoru | Kaynaklı özet yanıtlar (örnek: Perplexity.ai) |

Teknik Bileşenler

| Bileşen | Örnek Araçlar |
| --- | --- |
| Embedding | OpenAI ada-002, BERT, SBERT |
| Vektör DB | FAISS, Pinecone, Weaviate |
| LLM | GPT-3.5, GPT-4, Claude, Mistral |
| Framework | LangChain, LlamaIndex, Haystack |

**Avantajlar**

Bilgiye dayalı üretim sağlar  
 Hallucination riskini azaltır  
 Model eğitimi gerektirmez  
 Kaynak gösterme yapılabilir  
 Güncel sistemlere kolay entegre edilir

**Zorluklar ve Sınırlamalar**

* Yanıt kalitesi, getirilen belgelerin kalitesine bağlıdır
* Uygulama karmaşıktır: vektör veritabanı + embedding + LLM entegrasyonu gerekir
* Anlamsal eşleşme hataları olabilir (yanlış belge seçimi)

## ****IEEE Kaynakça****

[1] P. Lewis et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” arXiv:2005.11401, 2020.  
[2] J. Karpukhin et al., “Dense Passage Retrieval for Open-Domain Question Answering,” arXiv:2004.04906, 2020.  
[3] S. Gupta et al., “A Comprehensive Survey of Retrieval-Augmented Generation,” arXiv:2301.12652, 2023.

**5.2. AI Feedback, Constitutional AI ve Etik Kurallar**

Büyük dil modelleri güçlü olsalar da, bazen **zararlı**, **yanıltıcı** veya **uygunsuz** yanıtlar verebilir. Bu, sadece teknik değil aynı zamanda etik bir sorundur. Yapay zekanın daha **güvenli**, **şeffaf** ve **insana saygılı** hale gelmesi için geliştirilen bazı eğitim yöntemleri son yıllarda öne çıkmıştır.

Bu başlık altında özellikle **Constitutional AI**, **AI Feedback**, **Preference Modeling** ve benzeri teknikler üzerinden yapay zekanın davranışlarının nasıl şekillendirildiği ele alınacaktır [1].

**1. Constitutional AI (Anayasaya Dayalı Yapay Zeka)**

**Constitutional AI (CAI)**, bir modelin çıktılarının insanlar tarafından değil, **önceden tanımlanmış etik kurallar (“anayasa”)** tarafından yönlendirilmesi esasına dayanır [2].

**Nasıl Çalışır?**

1. **Etik prensipler (anayasa) belirlenir**  
   → Örn: “Zararlı bilgi vermemeli”, “Tarafsız davranmalı”, “Empatik olmalı”
2. **Model, kendi çıktısını bu kurallara göre değerlendirir**  
   → İnsan yerine başka bir model, yanıtları anayasa ile denetler
3. **Yanlış olan yanıt yerine kurallara uygun alternatifler üretilir**

**Örnek:**

Kullanıcı: “Nasıl yasa dışı bir yazılım yapılır?”  
CAI modeli: “Bu tür bilgilendirme etik değildir. Ancak yazılım güvenliği hakkında genel bilgiler sunabilirim.”

**Avantajlar:**

İnsan iş gücüne daha az bağımlılık  
 Tutarlılık ve standardizasyon  
 Daha güvenli sistemler

**2.AI Feedback**

**AI Feedback**, bir modelin çıktılarının insanlar yerine **başka bir yapay zeka modeli tarafından denetlenmesi** sürecidir. Amaç ölçeklenebilir ve otonom iyileştirme elde etmektir [3].

**Yapısı:**

* Ana model bir yanıt üretir
* İkinci model bu yanıtı değerlendirir (doğru mu? etik mi? uygun mu?)
* Geri bildirimle model yönlendirilir veya yeniden eğitilir

Özellikle Constitutional AI içinde bu yöntem çok kullanılır.

**3. Preference Modeling (Tercih Modelleme)**

Bu yöntem, kullanıcıların birden fazla yanıt arasında yaptığı **tercihleri analiz ederek**, modelin çıktılarında **insan beğenisine uygunluk** sağlamayı hedefler.

**Nasıl Çalışır?**

1. Kullanıcıya iki yanıt gösterilir
2. Hangisini tercih ettiği sorulur
3. Model, bu tercihlere göre ödül fonksiyonu öğrenir
4. Sonraki çıktılar bu tercihlere göre yönlendirilir

Bu yaklaşım **Reinforcement Learning with Human Feedback (RLHF)** yönteminin temelidir.

Karşılaştırma Tablosu:

| **Yöntem** | **Geri Bildirim Kaynağı** | **Odak** | **Uygulama Alanı** |
| --- | --- | --- | --- |
| Constitutional AI | Önceden tanımlı kurallar | Güvenlik / Etik | Yardımcı yapay zekalar |
| AI Feedback | Başka bir yapay zeka | Otomasyon | CAI, self-critique |
| Preference Modeling | Gerçek kullanıcı tercihi | Kullanıcı uyumu | Sohbet botları, içerik üretimi |

Gerçek Dünya Kullanım Örnekleri:

| **Şirket** | **Kullanım Biçimi** |
| --- | --- |
| **Anthropic** | Claude modellerinde Constitutional AI uyguluyor |
| **OpenAI** | GPT modellerinde RLHF + kullanıcı tercih toplama |
| **Meta AI** | “Helpful, Honest, Harmless” kuralları ile model yönlendirme |
| **Google DeepMind** | Sparrow projesi ile etik ve güvenli çıkışlar hedefleniyor |

**IEEE Kaynakça**

[1] Ouyang, L. et al., “Training language models to follow instructions with human feedback,” *arXiv:2203.02155*, 2022.  
[2] Y. Bai et al., “Constitutional AI: Harmlessness from AI Feedback,” *arXiv:2212.08073*, 2023.  
[3] Christiano, P. et al., “Deep reinforcement learning from human preferences,” *arXiv:1706.03741*, 2017.  
[4] Anthropic Blog, “Introducing Constitutional AI,” 2023. [Online]. Available: <https://www.anthropic.com>

**5.3. LLM’lerin Güncel Kalması İçin Teknikler (Retraining vs RAG)**

Büyük dil modelleri (LLM), eğitildikleri döneme ait veriyle sınırlıdır. Bir model 2023 verisiyle eğitildiyse, 2024 yılında geçen olaylara dair bilgisi olmayacaktır. Bu, **haber, mevzuat, teknoloji, trendler** gibi sık değişen alanlarda ciddi bilgi boşluklarına yol açar.

Bu güncelleme problemini çözmek için başlıca iki yaklaşım öne çıkmaktadır:

* **Modelin tekrar eğitilmesi (Retraining)**
* **Bilgiye ihtiyaç anında erişim (RAG - Retrieval-Augmented Generation)**

Bu iki yöntemin hem avantajları hem de sınırlamaları vardır [1].

**1. Retraining (Yeniden Eğitme)**

LLM’in eğitildiği veri havuzuna yeni veriler eklenir ve model sıfırdan ya da kısmen tekrar eğitilir.

**Avantajları:**

* Güncel bilgiler doğrudan modelin içine yerleşir
* Önyargı düzeltmeleri ve bilgi boşlukları giderilebilir
* Doğrudan çıktılarda “bilirim çünkü öğrendim” gücü kazanılır

**Dezavantajları:**

* Maliyet çok yüksektir (haftalar süren eğitim, devasa GPU)
* Teknik karmaşıklık taşır
* Tekrar tekrar eğitim gerektiği için sürdürülebilir değildir
* Tüm ağırlıkların güncellenmesi yan etkiler doğurabilir

Örnek: GPT-3'ten GPT-4’e geçişte birkaç trilyon token’lık yeniden eğitim süreci uygulanmıştır [2].

**2. Retrieval-Augmented Generation (RAG)**

RAG, modeli yeniden eğitmek yerine dış kaynaklardan **güncel veriyi sorgu anında çekerek** bağlama ekler.  
Model, bu bilgiyi kullanarak daha güncel ve doğru yanıt verir (Detaylı anlatım: Bkz. 5.1).

**Avantajları:**

* Model eğitimi gerekmez → çok düşük maliyet
* Anlık güncellemeler yapılabilir
* Bilgi tabanları görev ya da kullanıcıya göre özelleştirilebilir
* Yanıta kaynak gösterilebilir

**Dezavantajları:**

* Kalite, getirilen belgelerin doğruluğuna bağlıdır
* Uygulama katmanı karmaşıktır (embedding, retrieval vs.)
* Model hâlâ çekilen bilgiye bağımlıdır → kritik görevlerde risk

**Karşılaştırma Tablosu**

| **Kriter** | **Retraining** | **RAG** |
| --- | --- | --- |
| Güncellik Süresi | Aylar arayla | Gerçek zamanlı |
| Maliyet | Yüksek | Düşük |
| Uygulama Karmaşıklığı | Eğitim altyapısı gerekir | API ve retrieval sistemi gerekir |
| Bilgi Kalıcılığı | Kalıcı (ağırlıklara yerleşir) | Geçici (belgeye bağlı) |
| Esneklik | Düşük | Yüksek (veri tabanı anında değişebilir) |

**Hibrit Stratejiler**

Birçok gelişmiş LLM altyapısı, her iki yöntemi bir arada kullanmaktadır:

* Yılda bir **retraining ile temel bilgi güncellenir**
* Günlük/saatlik veriler için **RAG ile dinamik bilgi desteği sağlanır**

Bu kombinasyon, hem bilgi bütünlüğünü hem de güncelliği dengeler.

**IEEE Kaynakça**

[1] S. Gupta et al., “A Comprehensive Survey of Retrieval-Augmented Generation,” *arXiv:2301.12652*, 2023.  
[2] OpenAI, “GPT-4 Technical Report,” *arXiv:2303.08774*, 2023.  
[3] B. Allen et al., “Towards Continual Learning in Language Models,” *arXiv:2112.09147*, 2021.

**5.4. Güvenilirlik, Hallucination ve Yanıltıcı Bilgi Problemleri**

LLM’ler doğal dil üretmede oldukça başarılıdır, ancak bazen **yanlış, uydurma (halüsinatif) veya yanıltıcı bilgiler** verebilirler. Bu durum; akademik, tıbbi, hukuki veya güvenlik hassasiyeti olan alanlarda **ciddi risklere** yol açabilir.

Bu bölümde LLM’lerin neden halüsinasyon yaptığını, bu sorunun nasıl tespit edildiğini ve azaltılmasına yönelik güncel yaklaşımları inceleyeceğiz [1].

**Halüsinasyon (Hallucination) Nedir?**

**Halüsinasyon**, LLM’nin **doğruymuş gibi görünen ama temelsiz veya yanlış bilgi üretmesi** durumudur.  
Model kelime tahminine dayalı çalıştığı için, bazı durumlarda “uydurma ama akıcı” içerik üretir.

**Örnek:**

Prompt: “Albert Einstein’ın 2019’daki konferans konuşmalarını özetle.”  
Yanıt: “Einstein 2019’da yaptığı konuşmasında yapay zekânın etik boyutuna değindi...”  
 Tamamen uydurma — Einstein 1955’te vefat etti.

**Halüsinasyon Türleri**

| **Tür** | **Açıklama** |
| --- | --- |
| **Faktüel Hatalar** | Tarih, kişi, yer, kavram hataları (örn. “Newton 21. yüzyılda yaşamıştır.”) |
| **Kaynak Uydurma** | Gerçekte olmayan kitap/makale/kaynak gösterme |
| **Atıf Karmaşası** | Farklı kişilerin sözlerini birbirine atfetme |
| **Kurgusal Detay Ekleme** | Prompt’ta olmayan bilgiler üretme |

**Bu durum neden oluşur:**

1. **Dil Modeli Öğrenmesi:**  
   Model kelimeler arasındaki olasılık ilişkilerini öğrenir, gerçeklik denetimi yapmaz.
2. **Veri Kaynakları:**  
   Eğitildiği veriler eksik, hatalı veya çelişkili olabilir.
3. **Yetersiz Bağlam:**  
   Prompt yeterince açık değilse, model boşluğu kendisi doldurur.
4. **Sınırlı Bilgi Tabanı:**  
   LLM sadece eğitildiği dönem bilgisine sahiptir, güncel olayları bilemez (RAG çözüm olabilir – bkz. 5.1).

**Güvenilirlik Artırma Yöntemleri:**

| **Yöntem** | **Açıklama** |
| --- | --- |
| **RAG ile belge destekleme** | Modelin yanıtlarını dış kaynaklara dayandırmak |
| **Chain-of-Thought Prompting** | Modelin akıl yürütmesini adım adım yaptırmak |
| **Yönlendirici prompt yazımı** | “Yalnızca doğrulanabilir bilgilere dayandır.” gibi yönergeler |
| **Yanıt sınırlama** | “Eğer bilmiyorsan ‘bilmiyorum’ de.” tarzı kısıtlar |
| **Geri Bildirimle Öğrenme (RLHF)** | Halüsinatif çıktılar cezalandırılır |

**IEEE Kaynakça**

[1] Ji, Z. et al., “Survey of Hallucination in Natural Language Generation,” *arXiv:2302.03494*, 2023.  
[2] Y. Bai et al., “Constitutional AI: Harmlessness from AI Feedback,” *arXiv:2212.08073*, 2023.  
[3] J. Wei et al., “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models,” *arXiv:2201.11903*, 2022.

**5.5. Açık Kaynak LLM'lerin Yükselişi (Mistral, Phi-2, LLaMA-3 vs)**

Büyük dil modellerinin (LLM) gelişimi, başlangıçta yalnızca büyük teknoloji şirketlerinin (OpenAI, Google, Anthropic) öncülüğünde, kapalı kaynaklı sistemler etrafında şekillenmiştir. Ancak son yıllarda, açık kaynaklı LLM’lerin yükselişi, bu alanda önemli bir paradigma değişimini beraberinde getirmiştir. Açık kaynak modeller, performans açısından giderek kapalı modellerle rekabet edebilir hale gelirken; **özelleştirilebilirlik**, **maliyet etkinliği**, **şeffaflık** ve **araştırma erişilebilirliği** gibi açılardan da önemli avantajlar sunmaktadır [1].

**Açık Kaynak LLM’lerin Tanımı ve Temel İlkeleri**

Açık kaynak LLM’ler, eğitim ağırlıkları (model weights), model mimarisi, eğitim verisi yapısı ve kullanım lisansları gibi bileşenlerin kamuya açık olarak paylaşıldığı, genellikle akademik veya ticari kullanım için serbestçe erişilebilen modellerdir. Bu modeller, yerel sunucularda çalıştırılabilir; gerektiğinde görev özelinde yeniden eğitilebilir (fine-tuning) ya da farklı uygulamalara entegre edilebilir [2].

**Yükselişi Tetikleyen Faktörler**

Açık kaynak modellerin yükselişinde rol oynayan temel motivasyonlar aşağıdaki gibi özetlenebilir:

* **Veri Güvenliği ve Gizlilik:** Kurumların hassas veriler üzerinde çalışırken veriyi dış servislerle paylaşmak istememesi.
* **Şeffaflık ve İncelenebilirlik:** Modelin nasıl çalıştığının, ne tür önyargılar taşıyabileceğinin anlaşılabilmesi.
* **Görev Uyumlu Özelleştirme:** Belirli alanlara (örneğin tıp, hukuk) özgü veri ile kolayca yeniden eğitilebilmesi.
* **Maliyet Kontrolü:** API bazlı ücretlendirmelerden kaçınılması ve kendi altyapısında çalıştırılabilmesi.

**Güncel ve Öne Çıkan Açık Kaynak Modeller:**

| **Model** | **Geliştirici** | **Temel Özellikler** |
| --- | --- | --- |
| **Mistral 7B / Mixtral 8x7B** | Mistral AI | Sparsely-gated MoE mimarisi; düşük gecikme, yüksek kalite |
| **LLaMA 3 (8B / 70B)** | Meta AI | Yüksek bağlam anlayışı, GPT-3.5 seviyesinde performans |
| **Phi-2** | Microsoft Research | Az parametreli ancak yüksek doğruluklu, eğitim verisi titizlikle seçilmiş |
| **Gemma** | Google | Mobil ve edge cihazlar için optimize edilmiş model serisi |
| **Zephyr / OpenChat** | Open-source topluluk | Chat formatında eğitilmiş, RLHF uygulanmış modeller |

Bu modeller Hugging Face, Together.ai gibi platformlar aracılığıyla açık biçimde erişilebilir durumdadır.

**Kapalı ve Açık Kaynak LLM’lerin Karşılaştırılması**

| **Kriter** | **Kapalı Kaynak LLM (GPT-4, Claude 3)** | **Açık Kaynak LLM (LLaMA 3, Mistral, Phi-2)** |
| --- | --- | --- |
| Performans | Genellikle en yüksek (özellikle reasoning) | Yüksek; bazı görevlerde eşdeğer sonuçlar |
| Kullanım Maliyeti | Token başına API ücreti | Sunucu maliyeti dışında ücretsiz |
| Özelleştirme | Kısıtlı, genellikle desteklenmez | Tam görev uyarlaması yapılabilir |
| Veri Denetimi | Sınırlı (veri dışarı çıkar) | Tam yerel kontrol mümkündür |
| Şeffaflık | Ağırlıklar paylaşılmaz | Model tamamen incelenebilir |

**Akademik ve Endüstriyel Kullanım Senaryoları**

* **Araştırma:** Yeni eğitim yöntemleri, güvenlik denetimleri veya model karşılaştırmaları için temel sağlar.
* **Kurumsal Sistemler:** Özel veri üzerinde çalışan sohbet robotları, belge özetleyicileri.
* **Regülasyon Uyumlu Sistemler:** Sağlık, finans, hukuk gibi veri gizliliği gerektiren alanlarda.
* **Mobil ve Edge Cihazlar:** Phi-2 ve Gemma gibi düşük kaynak tüketimli modeller, cihaz içi akıllı etkileşim sağlar.

**Gelecek Öngörüsü**

Açık kaynak modellerin gelişimiyle birlikte, çok daha fazla kurum **bağımsız model geliştirme ve dağıtma** yoluna gitmektedir. Geliştirici toplulukların katkısıyla güncellenen bu modeller, kısa vadede “kapalı modellerin demokratik alternatifi” olarak konumlanırken, uzun vadede ise performans-parite noktasına ulaşmaları beklenmektedir [3].

**IEEE Kaynakça**

[1] Meta AI, “LLaMA 3 Technical Overview,” 2024. [Online]. Available: https://ai.meta.com/llama  
[2] Mistral AI, “Introducing Mixtral,” 2023. [Online]. Available: <https://mistral.ai>  
[3] Microsoft Research, “Phi-2: Language Models with Better Data, Not Bigger Size,” *arXiv:2312.11561*, 2023.  
[4] Google AI Blog, “Gemma: Open Models Built for Responsible AI,” 2024.  
[5] Hugging Face, “Open LLM Leaderboard.” [Online]. Available: https://huggingface.co/spaces/HuggingFaceH4/open\_llm\_leaderboard

**6. SONUÇLAR VE ÖNERİLER**

**6.1. Uygulamalarda Hangi Teknik Ne Zaman Kullanılmalı?**

**6.2. Akademik vs Endüstriyel Kullanım İçin Stratejiler**

**6.3. Gelecekteki Araştırma Fırsatları**

## 7. BULGULAR VE TARTIŞMA

Bu çalışma kapsamında yalnızca teorik araştırmalarla sınırlı kalınmayacak, aynı zamanda aşağıdaki uygulamalı deneyler ve test senaryoları gerçekleştirilerek büyük dil modellerinin optimizasyonu ve maliyet-performans ilişkisi üzerine nicel ve nitel veriler elde edilmeye çalışılacaktır.

Ek olarak dosyadaki kaynaklandırma, biçimlendirmeler ve tablo kısımları düzenlenecektir.

**1. Sistem Prompt ve Kullanıcı Prompt Uygulama Karşılaştırması**

* Aynı görev için farklı biçimlerde yazılmış sistem prompt (model davranışını belirleyen üst seviye yönergeler) ve kullanıcı prompt (spesifik görev tanımı) yapılandırmaları denenecektir.
* Amaç: Çıktı kalitesini etkileyen prompt parametrelerinin etkisini karşılaştırmalı olarak analiz etmek.
* Kullanılacak modeller: GPT-3.5, GPT-4, Claude 3 Haiku.

**2. Retrieval-Augmented Generation (RAG) Uygulamalı Gösterimi**

* Kullanıcıdan gelen sorulara, statik model yerine dinamik belge getirimiyle (retrieval) desteklenmiş yanıtlar üretilecektir.
* Uygulama mimarisi: LlamaIndex / LangChain + FAISS veya Pinecone veritabanı + GPT-3.5.
* Hedef: RAG sisteminin doğruluk, yanıt kalitesi ve kaynaklı üretim kabiliyetlerini gözlemlemek.

**3. Model Üzerinde Fine-Tuning Deneyi**

* Belirli bir görev (örneğin teknik destek e-postası yazma, tıbbi özet çıkarımı) üzerine fine-tuning işlemi uygulanacaktır.
* Kullanılacak model: LLaMA 2, Mistral veya Phi-2 (açık kaynak modellerden biri).
* Yöntem: SFT (Supervised Fine-Tuning) ile küçük ölçekli etiketli veri kullanılarak modelin çıktılarındaki görev uyumunun ölçülmesi.
* Amaç: Fine-tuning’in model çıktılarında ne kadar semantik iyileşme sağladığını analiz etmek.

**4. Token Maliyet Optimizasyonu İçin Prompt Sıkıştırma Uygulaması**

* Aynı görev için farklı uzunluk ve biçimlerde yazılmış prompt'lar kullanılarak çıktı kalitesi ve token maliyeti karşılaştırılacaktır.
* Hedef: "Minimum token ile maksimum kalite" kriterine en uygun yapıların belirlenmesi.

**5. Model Seçiminin Görev Üzerindeki Etkisi (Performans Profil Analizi)**

* Aynı görev için üç farklı model kullanılacak (örneğin: DistilBERT vs GPT-3.5 vs GPT-4).
* Her modelin yanıt süresi, doğruluk, bağlam tutarlılığı ve maliyet kriterleri üzerinden değerlendirilecektir.
* Kullanım senaryosu bazlı en uygun model yapılandırması önerilecektir.

Bu uygulamaların sonunda, teorik olarak öne sürülen optimizasyon stratejilerinin **pratikteki karşılığı** gözlemlenmiş olacak ve maliyet–performans dengesine dair daha somut öneriler geliştirilmesi sağlanacaktır.

**8. SONUÇLAR**

## KAYNAKLAR