**GENAI Kavramlar ve Bilgi Notu**

**1. LLM (Large Language Model)**

Büyük dil modelleri; çok büyük metin veri kümeleriyle eğitilmiş, insan benzeri metin üretebilen , soruları yanıtlayan, çeviri yapabilen, sohbet edebilen ve kod yazan modellerdir.

LLM’ler, çok büyük metin koleksiyonları (kitaplar, internet siteleri, belgeler...) üzerinde eğitilir. Bu sayede bir kelimenin veya cümlenin ne geleceğini “tahmin edebilir”.

Örnekler aşağıdakş tabloda yer almaktadır:

| **Model Adı** | **Geliştirici** | **Öne Çıkan Özellik** |
| --- | --- | --- |
| **GPT-4** | OpenAI | Çok yönlü, çok dil bilen |
| **Claude** | Anthropic | Güvenliğe odaklı |
| **LLaMA** | Meta | Açık kaynaklı |
| **Gemini** | Google DeepMind | Multimodal destekli |

**LLM’lerin Sınırlamaları**

**Halisülasyonlar**: Gerçekte olmayan bilgileri gerçekmiş gibi üretebilir.

**Güncel veri eksikliği**: Eğitildiği tarihten sonra olan olayları bilemeyebilir (eğer harici kaynaklara bağlı değilse).

**Yanlılık (Bias)** içerebilir. Bu kapsamda toplum tarafından genel olarak kabul gören ırkçı, ayrımcılığa neden olan söylemler içerbilir.

**2. Prompt Engineering**

Bir yapay zeka modelinden en iyi çıktıyı alabilmek için **nasıl bir komut (prompt)** verilmesi gerektiğini planlama, test etme ve iyileştirme sanatıdır.

Örn: "Explain quantum physics like I'm 5."

**Prompt Türleri:**

| **Tür** | **Açıklama** |
| --- | --- |
| **Zero-shot** | Hiç örnek verilmeden görev istenir. |
| **Few-shot** | Görevle ilgili 1-2 örnek verilir. |
| **Chain-of-thought** | Modelden düşünerek adım adım yanıt vermesi istenir. |
| **Role-based** | “Sen bir avukatsın, şu dilekçeyi yaz” gibi roller tanımlanır. |
| **Instructional** | “X yap, sonra Y yap. Sonucu listele.” gibi adım adım talimatlar verilir. |

**3. Temperature**

Modelin yanıtlarında ne kadar *“yaratıcı”* ya da *“tutarlı”* olacağını belirleyen bir parametre.

**Nasıl Çalışır?**

* **Temperature değeri 0’a yakınsa (örneğin 0 - 0.3):**  
  → Model daha **tutarlı**, **kesin** ve **tekrar eden** yanıtlar verir.  
  → Genelde daha **güvenilir** ve **gerçekçi** sonuçlar alınır.  
  → Hallucination riski düşer ama yaratıcılık sınırlıdır.
* **Temperature değeri yüksekse (örneğin 0.8 - 1.0):**  
  → Model daha **yaratıcı**, **sürprizli** ve bazen **absürt** yanıtlar üretir.  
  → Farklı alternatifler sunabilir ama doğruluk azalabilir.

| **Temperature** | **Özellikleri** | **Kullanım Alanı** |
| --- | --- | --- |
| 0 - 0.3 | Güvenilir, tekrar eden, tutarlı | Kodlama, teknik özet, akademik içerik |
| 0.4 - 0.7 | Dengeli, hem doğru hem yaratıcı | Blog yazısı, açıklamalar, e-posta |
| 0.8 - 1.0 | Yaratıcı, rastgele, riskli ama ilginç | Hikaye, şiir, reklam metni, ilham verici |

**4. Fine-tuning**

Genel amaçlı bir LLM’i, özel bir görev veya sektöre uygun hâle getirmek için yeniden eğitme süreci. Farklı bir ifade ile; **Fine-tuning**, önceden eğitilmiş (pretrained) bir büyük dil modelinin, **belirli bir görev** veya **alan** için **yeniden ve özel olarak eğitilmesidir**.

Modelin **daha isabetli ve özelleşmiş** cevaplar vermesi için Fine-Tuning yapılır.

Belirli bir jargon, yazım dili veya bilgiye adapte olması ve alanın terimlerine hakimiyeti için de Fine-Tuning yapılır.

Örneğin hukuk, tıp, finans gibi alanlara özel içerik üretmesi için Fine-Tuning yapılması daha isabetli sonuçlar almamızı sağlar.

**Fine-Tuning vs Prompt Engineering**

| **Özellik** | **Fine-Tuning** | **Prompt Engineering** |
| --- | --- | --- |
| **Uzun vadeli öğrenme** | ✅ Modelin kendisini değiştirir | ❌ Model sabit kalır |
| **Gereken veri** | Çok sayıda örnek gerekir | Genelde tek bir prompt yeterlidir |
| **Maliyet** | Yüksek (donanım, zaman) | Düşük |
| **Esneklik** | Belirli görevlerde güçlü | Genel kullanımda daha esnek |

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

RAG, büyük dil modellerinin (LLM'lerin) dış veri kaynaklarından bilgi araştırarak ve ardından bu bilgiyi kullanarak doğru, güncel ve bağlama uygun yanıtlar üretmesini sağlayan bir yöntemdir.

*“Retrieval” = Bilgi çekme*

*“Generation” = Cevap üretme*

*İkisi birleşince: Bilgiyi getir + üzerine mantıklı yanıt oluştur.*

**RAG Geliştirilmesinin Nedeni:**

* LLM'ler sadece eğitim aldıkları verilerle sınırlıdır (cut-off tarihleri var)
* Bilgi **güncellenemez** veya **sonradan eklenemez**
* “Hallucination” riski yani yanlış/uydurma bilgi üretme olasılığı yüksektir

**RAG ise bu sorunlara çözüm getirir.**

**RAG Nasıl Çalışır?**

RAG sistemleri genelde şu adımlarla çalışır:

1. **Kullanıcı bir soru sorar**
2. Sistem, **önceden indekslenmiş bir bilgi kaynağında** (veritabanı, belge koleksiyonu, knowledge base) bu soruyla **ilgili parçaları (passage, snippet) arar**
3. Bulduğu bu bilgileri, LLM’e **bağlamsal bilgi** olarak verir
4. LLM, sadece kendi "hafızasıyla" değil, aynı zamanda bu getirilen bilgilerle **yanıt oluşturur**

**RAG Avantajları**

| **Özellik** | **Açıklama** |
| --- | --- |
| ✅**Güncellik** | API, veritabanı veya belge ile her zaman en güncel bilgiye ulaşır |
| ✅ **Daha az hallucination** | Model, uydurmak yerine gerçek veriye dayanır |
| ✅ **Şeffaflık** | Kaynak gösterilebilir (“Bu cevabı şu belgede buldum” gibi) |
| ✅ **Veri ile öğrenme gerekmez** | Modeli yeniden eğitmeden bilgiye ulaşmak mümkündür |

**Nerelerde Kullanılır?**

* **Kurumsal chatbotlar:** Belgelerden bilgi çeksin ama cevapları doğal dilde verir.
* **Arama motorları:** Cevapları klasik sonuç listesi yerine akıllı metinlerle gösterir.
* **Eğitim asistanları:** Kaynaklı, sağlam ve referanslı cevaplar sunar.
* **Yapay zeka hukuk danışmanları:** Yönetmelik, yasa, karar gibi metinlerden bilgi alır.

**Not:** LLM yaratıcı ama unutkandır. Arama sistemleri bilgili ama sıkıcıdır. RAG ikisini birleştirerek hem bilgili hem doğal yanıtlar sağlar.

**6. Embedding (Gömme)**

Embedding, kelimeleri, cümleleri veya hatta belgeleri sayılardan oluşan vektörlere (yani matematiksel temsilcilere) dönüştürme yöntemidir.

**Amaç:** Anlamları sayısal bir forma çevirmek ve benzer olanları matematiksel olarak yakınlaştırmak.

**Embedding yapılmasının nedeni:** Bilgisayarlar kelimeleri “elma, masa, adalet” gibi anlayamaz. Onlara bir anlam verebilmek için her birini matematiksel bir uzayda konumlandırırız.

**Embedding Ne İşe Yarar?**

* **Semantik Arama:** Aynı anlama gelen farklı kelimeleri yakalar

"telefon kablosu" ≈ "şarj kablosu"

* **Chatbotlar:** Kullanıcının ne demek istediğini anlar

“Bugün hava nasıl?” ≈ “Bugünkü hava durumu nedir?”

* **RAG sistemleri:** Soru ile ilgili en yakın belge parçasını bulmak için kullanılır
* **Benzerlik Tespiti:** Cümleler veya belgeler benzer mi?

**Nerelerde kullanılır?**

| **Uygulama** | **Açıklama** |
| --- | --- |
| **Google Arama** | Kelimelerin anlamını anlar, tam eşleşmeye bağlı kalmaz |
| **Chatbotlar** | Kullanıcının niyetini çözümler |
| **LLM’ler (ChatGPT)** | Hafızasını yönetmek için vektörlerle çalışır |
| **Bilgi grafikleri** | Benzer düğüm/ilişki tespiti |

**Embedding üretmek için kullanılan bazı yöntemler:**

* **Word2Vec**
* **GloVe**
* **FastText**
* **BERT / Sentence-BERT**
* **OpenAI Embeddings (text-embedding-ada-002 gibi)**

Embedding, yapay zekanın kelimeleri ve kavramları “anlamca yakınlık” üzerinden sayılarla anlamasıdır.  
Bu sayede makineler artık sadece “ne söylendiğini” değil, “ne demek istendiğini” de anlayabilir.

**Özetle,** Embedding Kelimelerin veya cümlelerin matematiksel olarak vektörlere dönüştürülmesi.  
🡪 Semantic search ve vektör veritabanı sorgularında kullanılır.

**7. Token**

**Token**, bir dil modeli (örneğin ChatGPT, GPT-4, BERT vb.) tarafından **işlenmek üzere parçalara ayrılan kelime, kelime parçası ya da işarettir**. Yani model kelimeleri tam olarak değil, onları daha küçük **"token"** adlı birimlere çevirerek işler.

Token her zaman bir kelime değildir. Bazen kelimenin bir kısmı da token olabilmektedir.

Örnek: “Merhaba Dünya!” = Bu cümle model tarafından örneğin şöyle token’lara ayrılabilir: “Mer” - “haba” - “dün” - “ya”-“!”🡪 Yani bu 2 kelime + 1 noktalama = **5 token** olabilir.

Örnek 2: Cümle: *"Artificial intelligence is amazing."* Şu şekilde Token’lar olabilir: “Artificial” - “ intelligence” -“ is” -“ amazing”- “.” 🡪 Bu da yine 5 token.

**Token sayısının Önemi**

1. **Maliyet**: OpenAI gibi servisler, token sayısına göre ücret alır.  
   Örn: 1000 token ≈ 750 kelime
2. **Kapasite**: Bir modelin işlem yapabileceği maksimum token sayısı vardır.  
   Örn: GPT-4 Turbo = 128,000 token kapasiteye kadar çıkabilir.
3. **Bellek**: Bir önceki konuşma geçmişi, token bazında hatırlanır.

**Özetle,** Token, bir modelin "anlayabilmesi" için metnin parçalara ayrılmış halidir.  
Tüm metinler, modelin anlayacağı dijital kelime parçacıklarına dönüştürülür.

**8. Hallucination (Halüsinasyon)**

Modelin, gerçekte var olmayan ama inandırıcı görünen yanlış bilgi üretmesi.

**Neden Hallucination Olur?**

1. Modelin eğitildiği verilerde eksik/yanıltıcı bilgi olabilir.
2. Model, cevabı bilmese bile boşluk doldurmak ister.
3. Model, tahmin yaparak çalışır. “Bilmek” gibi anlamaz.
4. Gerçek kaynaklara bağlı değildir (örneğin bir veritabanı ya da güncel web).

**Nasıl Önlenir?**

1. **Retrieval-Augmented Generation (RAG)** **kullanılarak:** Modelin güncel bilgiye erişmesi sağlanır (örneğin: veritabanı, doküman, arama motoru).
2. **Bilgi grafikleri (Knowledge Graph)** ile desteklenerek.
3. **Daha iyi prompt engineering** ve doğrulama adımları ile.
4. **Kaynak gösterimi** zorunlu tutularak.

**9. Knowledge Graph (Bilgi Grafiği)**

**Gerçek dünya varlıklarını** (kişi, yer, nesne, kavram) ve **bunlar arasındaki ilişkileri** gösteren **graf yapısında bir veritabanıdır.** Yani: Nesneleri **düğümler (nodes)** olarak, aralarındaki bağları ise **ilişkiler (edges)** olarak gösterir.

**Temel Unsurlar:**

1. **Düğüm (Node):** Kişi, yer, şey, kavram  
   Örn: “Einstein”, “Fizik”, “Almanya”
2. **İlişki (Edge):** Aralarındaki bağlantı  
   Örn: “doğdu”, “çalıştı”, “uzmanlık alanı”
3. **Özellik (Property):** Varlıkların detayları  
   Örn: “doğum yılı: 1879”

metin, ekran görüntüsü, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

**Ne İşe Yarar?**

* Bilgileri daha **anlamlı** ve **bağlantılı** tutar.
* LLM’leri **gerçek bilgilerle desteklemek** için kullanılır.
* Karmaşık sorulara **daha doğru cevaplar** üretir.
* Arama motorlarında (Google, Bing) **daha iyi sonuçlar sağlar.**

**LLM ile İlişkisi:**

Bir LLM (örneğin GPT-4) bazen “halüsinasyon” yapar. Ama Knowledge Graph kullanılırsa, model gerçek bilgiye dayalı cevaplar üretir. Yani: “Bilgi grafiği = Yapay zekâya referans verebileceği sağlam bir bilgi kaynağı.”

**Nerelerde Kullanılır?**

Google arama sonuçları – Chatbotlar - Akıllı asistanlar (Siri, Alexa) - Medikal bilgi sistemleri - Finansal analiz sistemleri

**10. Zero-shot / Few-shot Learning**

**Zero-shot learning** (sıfır-şut öğrenme), **modelin daha önce hiç görmediği bir görev veya örnek üzerinde doğru sonuç üretmesidir**. Başka bir deyişle, model, verilen görevle ilgili eğitim verilmeden **doğrudan** çözüm üretmeye çalışır.

**Temel Özellik:**

* **Model, eğitim verisinde olmayan yeni örneklerle çalışır.**
* Modelin, görevle ilgili bilgi **doğrudan** verilmez, ancak **genel bilgi** ve **daha önce öğrendikleri** kullanılarak doğru cevaplar üretmeye çalışır.

**Örnek:** Bir model hiç "kendi fotoğrafını çekme" göreviyle eğitilmemiştir, fakat “Bir kişinin fotoğrafı nasıl çekilir?” gibi genel bilgilerle eğitilmiştir. Bu model, birinin fotoğrafını çekme görevini hiç görmeden doğru şekilde yapabilir.

**Zero-shot Kullanım Alanı:**

* **Doğal dil işleme (NLP)**: Eğer bir model daha önce belirli bir soru-cevap tipini görmemişse, sıfır-şut öğrenme sayesinde bu yeni sorulara doğru cevap verebilir.
* **Görüntü sınıflandırma**: Model, daha önce hiç görmediği bir nesnenin fotoğrafına bakarak o nesneyi sınıflandırabilir.
* **Çoklu dil modelleme**: Model, hiç eğitilmediği bir dilde bile metin anlayışı yapabilir.

**Few-shot Learning (Az-şut Öğrenme)**

**Few-shot learning**, **modelin belirli bir görevi çok az örnekle öğrenmesi** anlamına gelir. Bu durumda model, birkaç örnek üzerinden görev öğrenir ve doğru sonuçlar üretmeye çalışır.

**Temel Özellik:**

* **Model sadece birkaç örnekle eğitilir**.
* Model, bu örnekleri kullanarak, benzer yeni verilerde doğru sonuçlar vermeye çalışır.

**Örnek:** Bir model, 5 farklı matematiksel problem örneğiyle eğitildiğinde, bu örnekleri kullanarak benzer sorulara cevap verebilir. 5 örnekle doğru cevaplar verebilir.

**Few-shot Kullanım Alanı:**

* **Doğal dil işleme (NLP)**: Model, yalnızca birkaç örnekle yeni bir metin türünü anlayabilir veya benzer dil görevlerini yerine getirebilir.
* **Kişisel asistanlar**: Bir dil modeli, kullanıcının alışkanlıklarını birkaç etkileşimle öğrenebilir.
* **Görüntü sınıflandırma**: Model, yalnızca birkaç örnekle yeni bir kategori öğesini tanıyabilir.

**Farklar:**

* **Zero-shot: Model eğitim verisi olmadan ve hiç görmediği bir görevde doğru cevaplar üretir.**
* **Few-shot: Model, yalnızca çok az örnekle eğitilir ve yeni görevde başarılı olur.**

**Kısaca Özet:**

* **Zero-shot: Model hiç örnek görmeden doğru tahminler yapar.**
* **Few-shot: Model sadece birkaç örnekle eğitilir ve yeni görevlerde doğru sonuçlar üretir.**

**Kullanım Örnekleri:**

* **Zero-shot learning: “Bugün hava nasıl?” sorusunu model hiç görmediyse de, genel bilgiyle doğru bir yanıt verir.**
* **Few-shot learning: “Hangi filmler bana hitap eder?” sorusunda model, 3-4 öneriye dayalı olarak daha fazla öneri sunar.**