Büyük Dil Modellerinin Çalışma Prensipleri - I

Eğitmen
Kubilay Tuna

Kubilay Tuna

Machine Learning Engineer | Senior Data Scientist
@ EPAM Systems

Eğitim

- **Konya Teknik Üniversitesi** – Yüksek Lisans, Elektrik-Elektronik Mühendisliği
  - Araştırmacı
  - Endüstriyel Danışmalık
- **Selçuk Üniversitesi** – Lisans, Elektrik-Elektronik Mühendisliği
  - Bölüm ve fakülte ikinciliği
  - Öğrenci Asistan
  - Yüksek Onur Öğrencisi

Deneyim

- **EPAM Systems (11 Ay)**
  - Senior Data Scientist (11 Ay)
  - MLOps Mühendisi
  - RAG Mühendisi
  - Bilgisayarlı Görü Uzmanı
- **Huawei (4 Yıl)**
  - Technical Lead
  - Senior AI Engineer
  - AI Application Engineer
- **BNBC International** – Computure Vision Engineer (9 Ay)
- **InfoDif** – Computer Vision Engineer (1 year)

kubilaytuna26@hotmail.com

https://www.linkedin.com/in/ktuna/

Ders İçeriği

1. LLM'lerin önemi ve yükselişi
2. Doğal Dil İşleme Evrimi
3. Transformer Mimarisi
4. Dil Modelleri Nasıl Öğrenir?

LLM'LERİN ÖNEMİ VE YÜKSELİŞİ

Küresel Trendler ve Katalizörler

- 2017'de yayınlanan "**Attention is All You Need**" makalesiyle başlayan devrim.
- 2020 sonrası GPT-3 ve benzeri modellerin yeteneklerinin görünür hale gelmesi.
- Moore Yasası + Verinin artışı + Cloud altyapısı = LLM ivmesi
- Otomasyon ihtiyacı, bilgiye erişim hızı, çok dilli iletişim gibi kurumsal talebi hızlandıran faktörler.

Yükseliş Yolcuğu

- 2023'te ChatGPT'nin 2 ayda 100M kullanıcıya ulaşması
- Microsoft'un $10B OpenAI'a yatırımı, Google'un PaLM, Gemini modellerini yayınlaması, Meta'nın LLaMA 2, LLaMA 3 modellerini çıkarması gibi kurumsal yatırımlar.
- Finans: risk modelleme, müşteri destek otomasyonu, sağlık: klinik karar desteği, özetleme ve hukuk: sözleşme inceleme, belge analizi gibi her sektörde yaygınlaşma.

Anahtar Noktalar

- Bilgiye doğal dil ile erişim: Hiç kodlama bilmeyen birinin basit yazılımlar geliştirebilmesi.
- Verimlilik artışı: Günlerce belki aylarca sürecek işin 5 dakikada yapılabilmesi.
- Yenilikçi ürünler: Chat destekli arayüzler, özelleştirilmiş LLM ajanları.

DOĞAL DİL İŞLEMENİN EVRİMİ

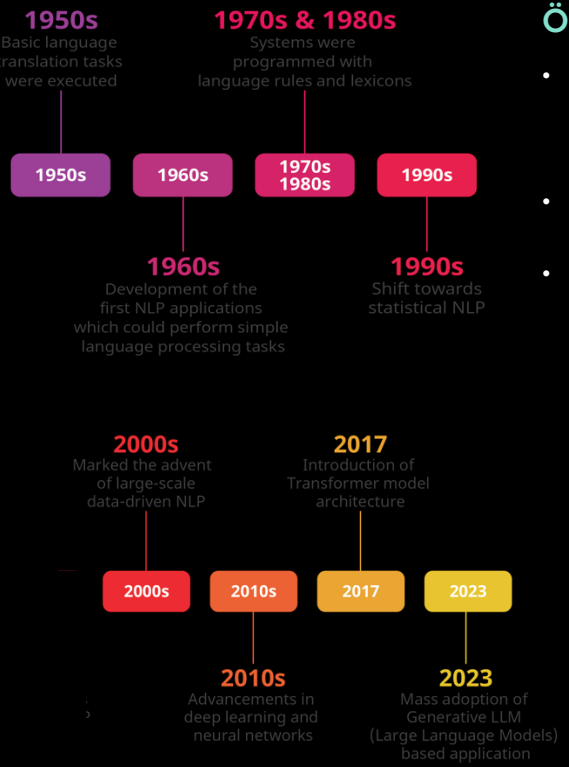
🏛️ 1. Klasik NLP Dönemi (1950–1990)

- El yazımı kurallar: Gramer kuralları ve dil bilgisi üzerinden kural tabanlı sistemler
- Makine çevirisinin ilk adımları (IBM, Georgetown deneyi)
- Sınırlamalar: düşük esneklik, manuel müdahale zorunluluğu, dar kapsam

📊 2. İstatistiksel NLP (1990–2010)

- Veriye dayalı yaklaşımlar: n-gram modelleri, Hidden Markov Models (HMM), TF-IDF
- Dil modellemede istatistiksel yaklaşımlar: Hangi kelime hangi kelimeden sonra gelir?

🤔 Anlamsal bağlam yok, sözdizimsel yapı ihmal ediliyor



🧠 3. Temsil Öğrenimi ve Derin Öğrenme (2010–2017)

- **Word embeddings (Word2Vec, GloVe):** Kelimeler vektör olarak modellenmeye başladı. Anlamsal yakınlık vektörlerde ölçülebilir hâle geldi.
- Bağlamı takip eden RNN, LSTM modelleri ortaya çıktı.
- Encoder-Decoder yapıları

🔁 4. Transformer Devrimi (2017–2020)

- **"Attention is All You Need" (2017):** Sıralı işlemeye gerek kalmadan tüm bağlama odaklanabilen yapı
- **BERT (2018):** çift yönlü bağlam öğrenme
- **GPT (2018-2020):** LLM devrimi

🚀 5. Foundation Model & LLM Çağı (2020–…)

- GPT-3, T5, PaLM, LLaMA gibi modellerle Zero-shot / Few-shot / Chain-of-thought kullanımları
- Artıkyalnızca metin analizi değil: Diyalog, kod üretimi, analiz, planlama, arama, sentez, araç kullanımı

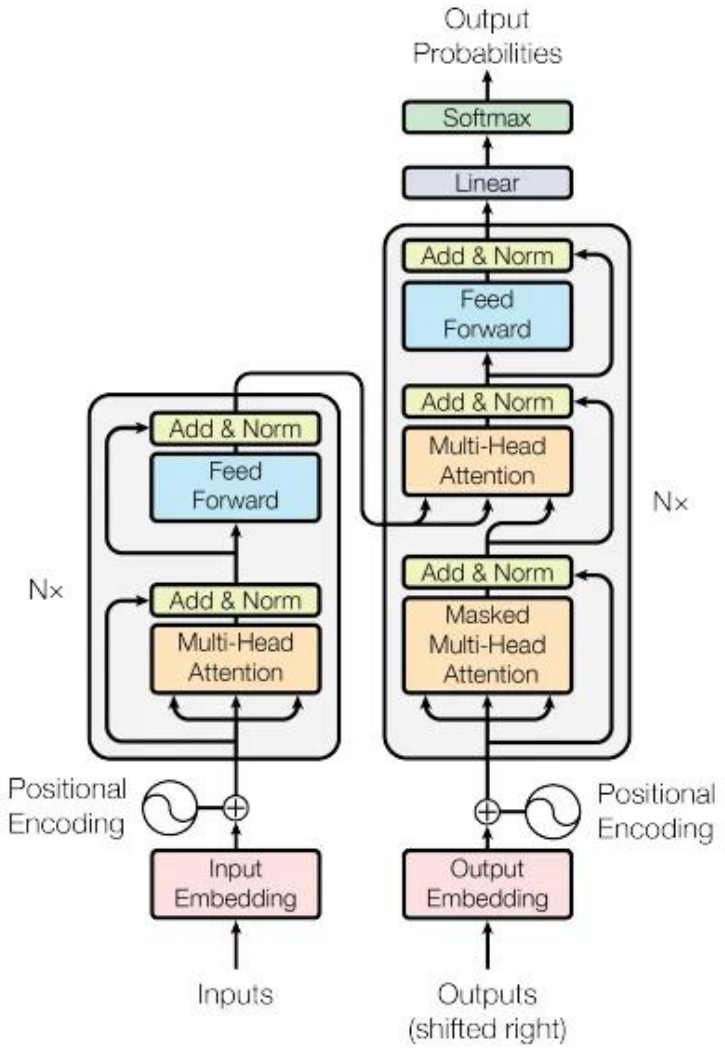
Transformer Mimarisi

Transformer mimarisi, LLM'lerin temelini oluşturan devrimsel bir yapıdır!

2017 yılında Google ve Toronto Üniversitesi tarafından “ Attention is all you need” başlıklı makalenin yayınlanmasının ardından her şey değişti ve günümüzün LLM'lerinin yapısını oluşturan transformer mimarisi ortaya çıktı. Bu yeni yaklaşım, bugün tanık olduğumuz Üretken Yapay Zekanı temellerini oluşturdu.

Bu mekanizma, dikkat mekanizmasının yardımıyla girdinin en kritik kısmına odaklanarak daha uzun dizileri işlemeyi mümkün kılmakta ve önceki modellerde karşılaşılan bellek sorunlarını çözmektedir. Ayrıca çok daha büyük eğitim veri kümelerini kullanarak girdi verilerini paralel olarak işleyebilmektedir.

Bu mekanizmanın ana bileşenleri; encoder-decorder yapısı, self-attention mekanizması, layer normalizasyonu ve residual connectionlardır.



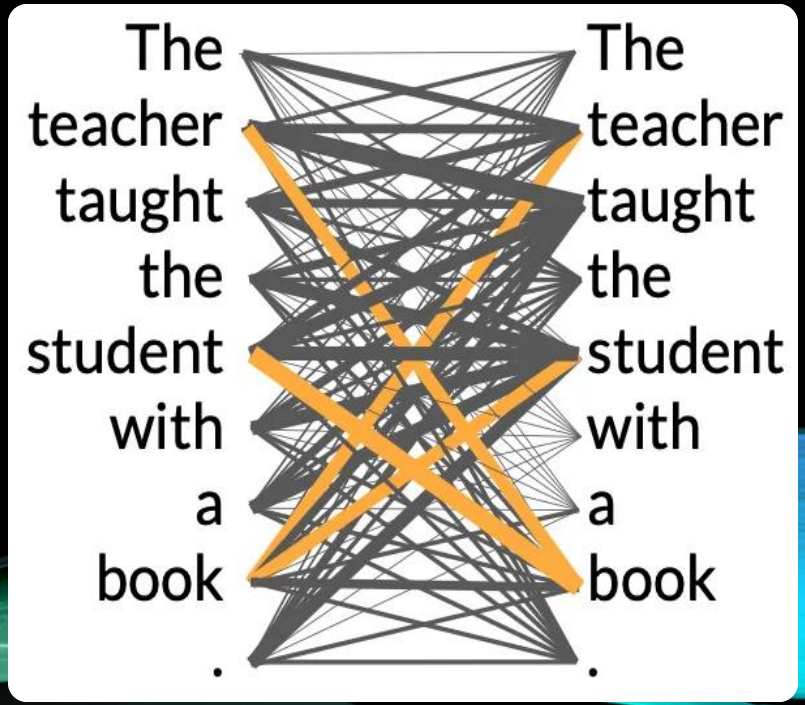
Transformer Mimarisi: Self-Atention Mekanizması

Self-attention mekanizması transformer yapısının önemli bileşenlerinden biridir.

Transformer mimarisinin gücü, bir cümle içindeki her kelimenin alaka düzeyini ve bağlamını anlama yeteneğinde yatar. Bu da girdi içindeki konumlarından bağımsız olarak her bir kelime arasındaki ilişkilerin öğrenilmesini kolaylaştıran self-attention mekanizması sayesinde gerçekleştirilir.

Bu ilişkiyi anlamak için her bir kelime ile diğerleri arasındaki dikkat ağırlıklarını etkili bir şekilde gösteren yandaki dikkat haritası (attention map) diyagramını inceleyelim.

Self-attention ağırlıkları tarafından oluşturulan bu sağlam bağlantılar, algoritmaya kitabın kimde olduğunu, kitabın kimde olabileceğini ve hatta belgenin daha geniş bağlamıyla ilgili olup olmadığını öğrenme yeteneği sağlar. Bu self-attention ağırlıkları LLM eğitimi sırasında öğrenilir.



Encoder-Decoder Yapısı: Sekanstan Sekansa Öğrenmenin Temeli

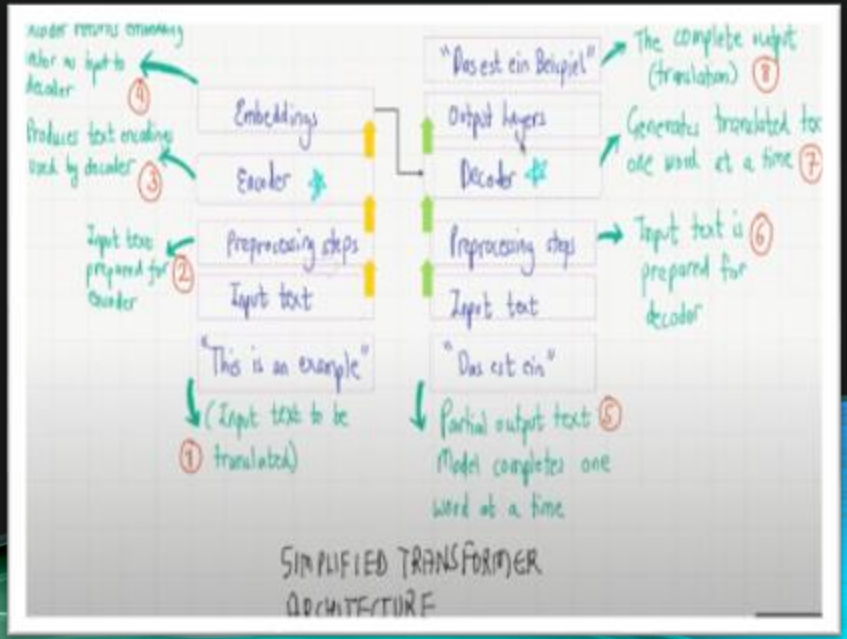
Encoder-Decoder mimarisi, özellikle sıralı verilerle çalışan görevler (makine çevirisi, konuşma tanıma, özetleme) için geliştirilmiş bir derin öğrenme mimarisidir.

Encoder, girdi dizisini işler (örneğin: İngilizce bir cümle) ve bağlamsal temsili bir context vector'e dönüştürür.

Decoder, bu bağlamsal vektörü kullanarak hedef çıktıyı üretir (örneğin: Almanca çevirisi). Her adımda önceki çıktıyı kullanarak sırayla kelimeleri üretir.

Genellikle **RNN**, **LSTM** ya da **GRU** gibi sıralı veriye uygun yapılar kullanılır.

Girdi dizisinin tamamı işlenmeden çıktı üretilemez. Bu da bütünsel anlamı kaçırma sorununu ortaya çıkarır. Ayrıca bu yapının performansı sabit boyuttaki context vector'e çok bağımlıdır.



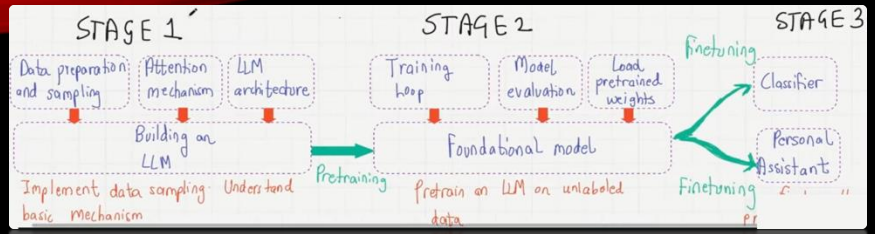
RNN-LSTM-Transformer Karşılaştırması

Özellik	RNN	LSTM	Transformer
Yapı	Tekrarlayan sinir ağı	Gated (kapılı) RNN	Self-attention temelli
Zaman karmaşıklığı	O(T) – sıralı işlem zorunlu	O(T) – sıralı işlem zorunlu	O(1) – paralelleştirilebilir
Uzun bağlam öğrenimi	Zayıf (vanishing gradient)	Orta – gated yapı sayesinde	Güçlü – her token tüm bağlama erişir
Paralel hesaplama	❌ Yok	❌ Yok	✅ Evet
Bellek gereksinimi	Düşük	Orta	Yüksek
Karmaşıklık	Düşük	Orta	Yüksek
Kullanım alanları	Erken NLP, dil modeli başlangıcı	Makine çeviri, konuşma tanıma	LLM'ler, özetleme, soru yanıtlama

🧭 Ne Zaman Hangisi?

Durum	Önerilen
Küçük veri / kaynak kısıtı	LSTM veya GRU
Gerçek zamanlı ses verisi	LSTM
Büyük veri / çoklu GPU	Transformer
Uzun bağlamlı metin işleme	Transformer (örneğin: >512 token)

DİL MODELLERİ NASIL ÇALIŞIR?



1) Giriş İstemi

» Hava bugün

Dil modeli bu girdiyi alır ve bu girdiyi tokenlara ayırarak işleme başlar. Her bir token kelime/kelime grubu/alt kelime ya da karakter olabilir.

2) Tokenization

Model girdiyi bireysel tokenlara ayırır:

["Hava", "bugün"]

Tokenizaiton modelin girdi metninin her bir parçasının ayrı ayrı anlaşılmasını ve işlenmesini sağlar.

3) Embedding

Her bir token vektör uzayında temsil edilebilmesi için embedding'e çevrilir. Bu sayısal vektörler her bir tokenın semantik anlamını yakalarlar.

Bu embeddingler, benzer anlamlara veya bağlamlara sahip kelimeleri yüksek boyutlu bir alanda birbirine yakın yerleştirir ve modelin ilişkileri anlamasını sağlar. Örneğin; "hava durumu", "iklim veya "tahmine" yakın olabilir.

4) Contextual Understanding, Attention

Model, istemdeki tokenlar arsındaki ilişkileri analiz etmek için dikkat mekanizması kullanır. Burada, her bir kelimenin cümledeki değeriyle nasıl ilişkili olduğu değerlendirilir. Örneğin ; "bugün"nün "hava durumu"nu değiştirdiğini ve mevcut hava durumu hakkında bir ifadeyi belirttiğini anlar.

DİL MODELLERİ NASIL ÇALIŞIR?

5) Sonraki Tokennın Tahmini

Model komut istemine dayanarak olası sonraki kelimeler veya tokenlar için bir olasılık dağılımı oluşturur. Bu tahmini yapmak için ön eğitim sırasında öğrenilen kalıplardan yararlanılır.

Bizim örneğimiz için muhtemel tahminlerden bazıları "**güneşli**", "**bulutlu**", "**yağmurlu**" veya "**rüzgarlı**" gibi kelimeler olabilir.

6) Örnekleme ve Üretme

Model tahmin edilen dağılımdan en olası belirteci seçer. Örneğin: "**güneşli**"

Ve cümle şu şekilde görünür:

» Hava bugün güneşli.

7) Adımların Tekrarı

Daha fazla metin talep edilirse, model işlemi tekrarlar ve yeni ifadeyi, güncellenmiş istem ("Bugün hava güneşli. ") olarak ele alır.

Daha sonra "**ve**", "**ile**", veya "**noktalama işaretleri**" arasında seçim yaparak bir sonraki tokenı tahmin eder.

Örneğin "**ve sıcak**" ekleyebilir ve aşağıdaki sonucu elde edebilir:

» Hava bugün güneşli ve sıcak.

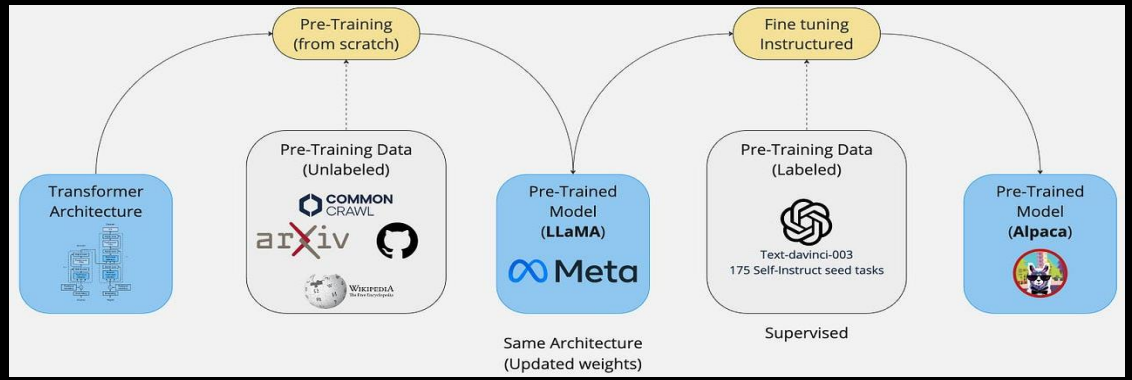
8) Tamamlama

Model, bir durdurma kriterine (örneğin bir süre, belirli bir token sayısı veya kullanıcı tarafından tanımlanan sınırlar) ulaşana kadar token üretmeye devam eder.

Son cümle şu olabilir:

» Hava bugün güneşli ve sıcak, hafif bir esinti var.

PRETRAİNİNG VS FİNETUNİNG



Pretraining

Modelin büyük, genel metin koleksiyonları (Wikipedia, Common Crawl, kitaplar, kodlar) üzerinde **gözetimsiz şekilde** eğitilmesi.

Amaç; bir sonraki kelimeyi tahmin etmektir

- **GPT:** Otoregresif
- **BERT:** Maskeli dil modelleme

Özelikler; devasa veri (~trilyon token), kendi kendini denetleyen (self-supervised), genel dil ve dünya bilgisi kazanımı.

Finetuning

Önceden eğitilen modelin belirli bir görev (sınıflandırma, özetleme, soru-cevap vb.) için **gözetimli veriyle yeniden** eğitilmesi.

Uygulamalar; sınıflandırma, özetleme, çeviri, kişisel asistant, vb.

Özelikler; küçük ama kaliteli veri (bin – milyon örnek), göreve özel optimize edilir, genelde daha kısa eğitim süresi gerekir, aşırı uyum (overfitting) ve felaket unutma (catastrophic forgetting) riski vardır


Q&A

TEŞEKKÜRLER!

Kubilay Tuna

Senior Data Scientist

kubilaytuna26@hotmail.com