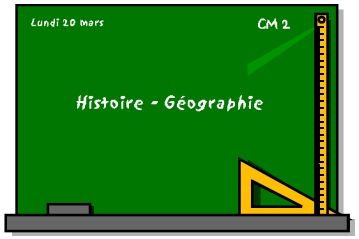
무지개
동산

Dil Modelleri

Prof.Dr. Banu Diri



Dil Modeli Nedir?

Bir dildeki kelimelerin sıralanışının olasılık dağılımı o dilin ***istatistiksel dil modeli*** olarak tanımlanır.

Lemma ?

Başsözcük→Sözlükteki madde başı olan kelimedir.

Başsözcükten türeyen aynı anlamdaki sözcüklere **sözcükbirim** denir.

read → reads, reading

ad soylu Türkçe bir kelimeye **24 çekim eki** eklenebilir
eylem soylu Türkçe bir kelimeye **46 çekim eki** eklenebilir
Bir dilin söz varlığı, o dildeki başsözcüklerin sayısı kadardır (**70 bin TR**)

Dilin modellenmesinin amacı

- Konuşma tanıma (Speech recognition)
- El yazısı tanıma (Handwriting recognition)
- İmla hatalarının düzeltilmesi (Spelling correction)
- Makine çeviri sistemleri (Machine translation systems)
- Optik karakter tanıma (Optical character recognizers)

El yazısı tanıma (Handwriting recognition)

Bankadaki veznedara bir not verildiğini düşünün, ve veznedar notu "I have a gub" olarak okusun. (cf. Woody Allen)

NLP burada yardımcı olur ....

gub ingilizcede anlamlı bir kelime değildir.

gun, gum, Gus, ve gull olabilir, fakat gun kelimesinin banka ile ilişki olasılığı daha fazla olduğundan "gub", "gun" olarak alınır.

İmla hatalarının kontrolünde

Birbirinin yerine sıklıkla geçebilen kelimeler

piece/peace, whether/weather, their/there …

Örnek:

"On Tuesday, the whether …"

"On Tuesday, the weather …"

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

W

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

Wh

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

Wha

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

What

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

What d

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

What do

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

What do you think the next letter is?

Letter-based Language Models

Shannon's Game

Guess the next letter:

What do you think the next letter is?

Guess the next word:

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

What do you think the next letter is?

Guess the next word:

What

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

What do you think the next letter is?

Guess the next word:

What do

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

What do you think the next letter is?

Guess the next word:

What do you

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

What do you think the next letter is?

Guess the next word:

What do you think

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

What do you think the next letter is?

Guess the next word:

What do you think the

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

What do you think the next letter is?

Guess the next word:

What do you think the next

Harf-tabanlı (Letter-based) dil modelleri

Shannon's Game

Guess the next letter:

What do you think the next letter is?

Guess the next word:

What do you think the next word is?

- zero-order approximation: harflerin sıraları birbirinden bağımsız
  - xfoml rxkhrjffjuj zlpwcwkcy ffeyvkcqsghyd

- first-order approximation: harfler birbirinden bağımsızdır, fakat dildeki (Ingilizce) harflerin dağılımlarına göre meydana gelir
  - ocro hli rgwr nmielwis eu ll nbnesebya th eei alhentppa oobttva nah

- second-order approximation: bir harfin görülme olasılığı bir önceki harfe bağlıdır
  - On ie antsoutinys are t inctore st bes deamy achin dilonasive tucoowe at teasonare fuzo tizin andy tobe seace ctisbe
- third-order approximation: bir harfin görülme olasılığı kendisinden önce gelen iki harfe bağlıdır
  - in no ist lat whey cratict froure birs grocid pondenome of demonstures of the reptagin is regoactiona of cre

Farklı diller için yüksek frekanslı trigram'lar:

İngilizce: THE, ING, ENT, ION
Almanca: EIN, ICH, DEN, DER
Fransızca: ENT, QUE, LES, ION
İtalyanca: CHE, ERE, ZIO, DEL
İspanyolca:QUE, EST, ARA, ADO

Dillerdeki hece benzerlikleri

Aynı aile içerisinde bulunan diller birbirlerine diğer dillere göre daha fazla benzer

Aynı aile içerisinde yer alan diller birbirlerine nasıl benzerler ?

- Hece tabanlı benzerlik

Aile içerisinde yer alan her bir dildeki en fazla kullanılan kelimeler çıkarılır;

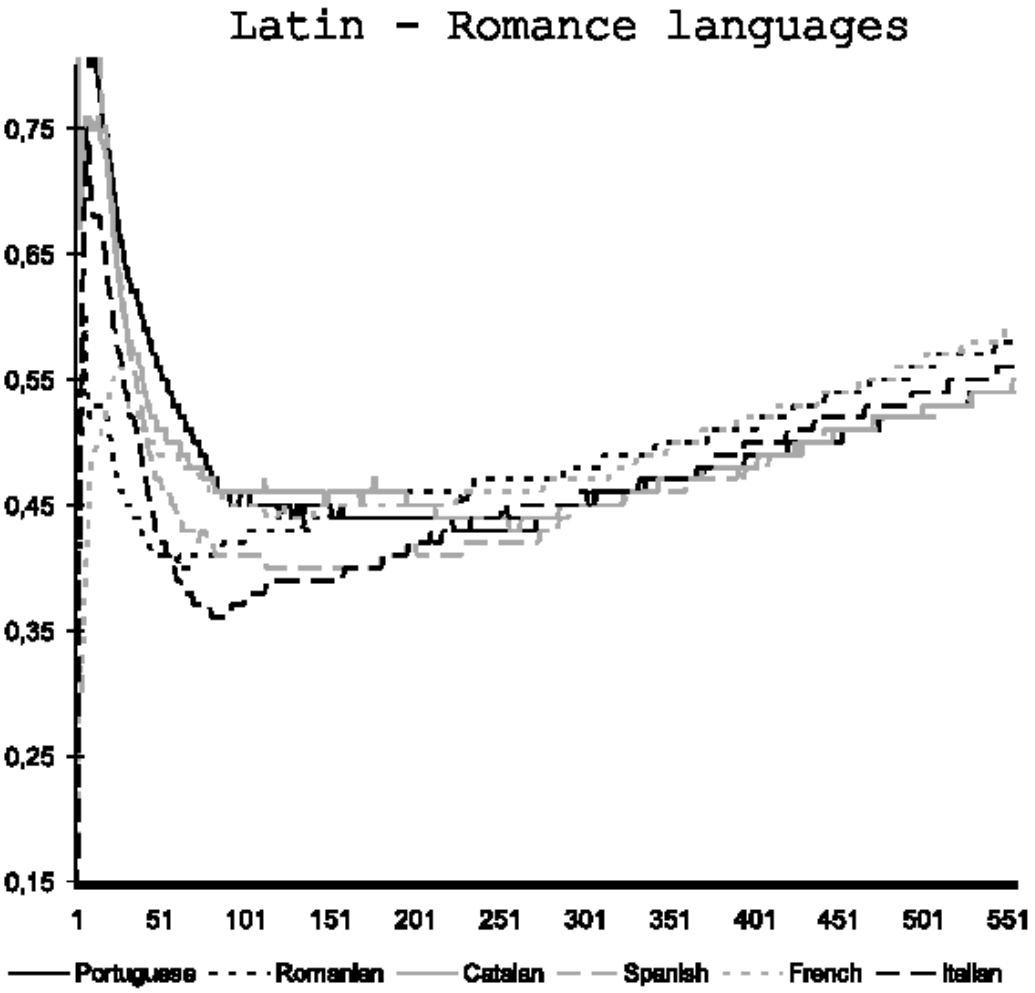
- Kelimeler hecelerine ayrılır
- Hecelerin frekansları hesaplanır
- Heceye dayalı dildeki benzerlik hesaplanır

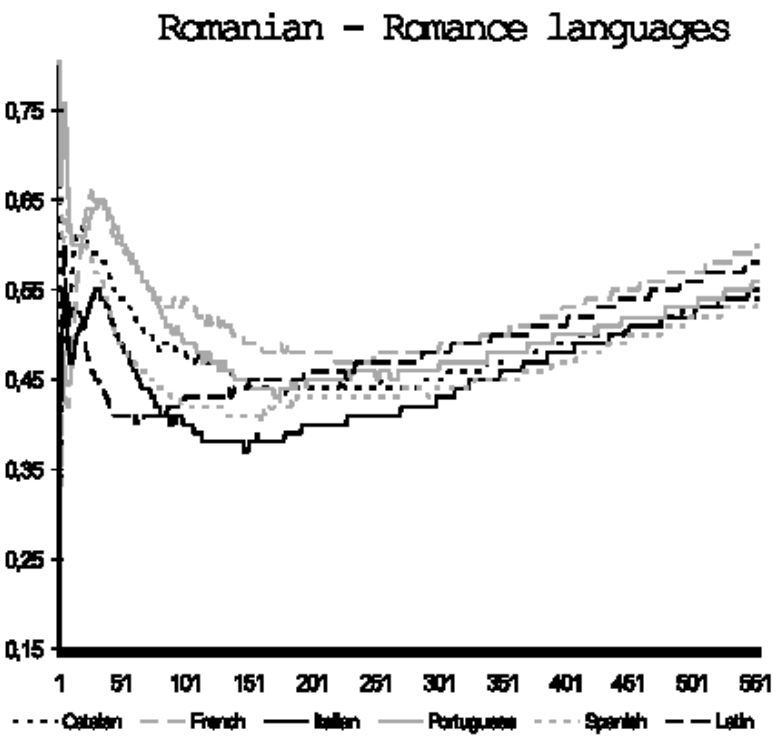
Örnek: Romance dilleri ailesi

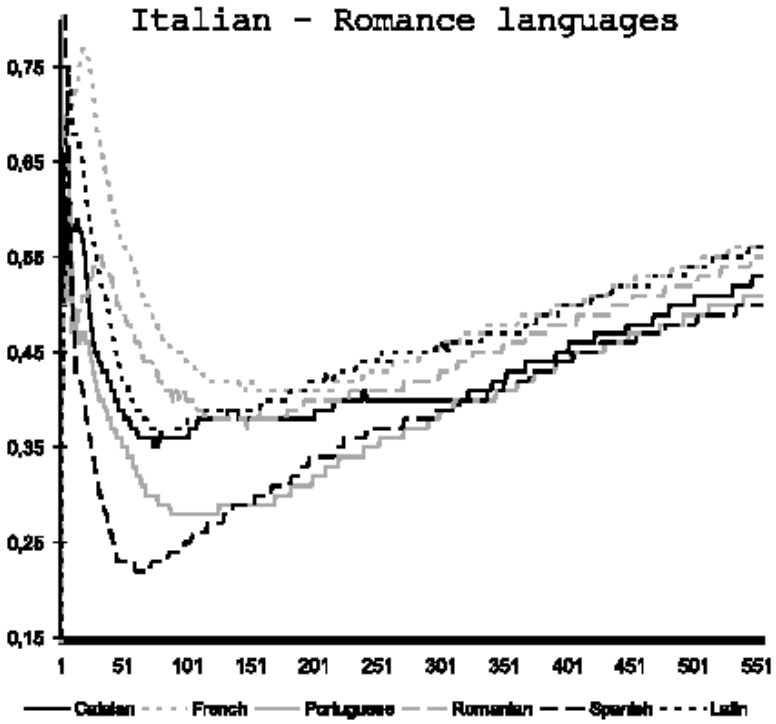
Romance dillerindeki heceler

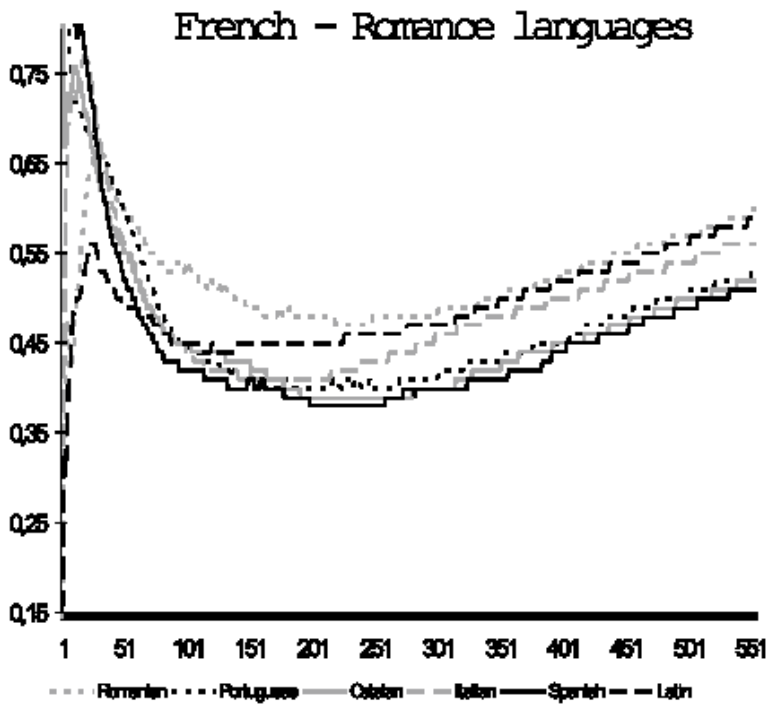
Language	The percentage covered by the first ··· syllables						No. syllables	
	100	200	300	400	500	561	type	token
Latin	72%	86%	92%	95%	98%	100%	561	3922
Romanian	63%	74%	80%	84%	87%	90%	1243	6591
Italian	75%	85%	91%	94%	96%	97%	803	7937
Portuguese	69%	84%	91%	95%	97%	98%	693	6152
Spanish	73%	87%	93%	96%	98%	99%	672	7477
Catalan	62%	77%	84%	88%	92%	93%	967	5624
French	48%	61%	67%	72%	76%	78%	1738	5691

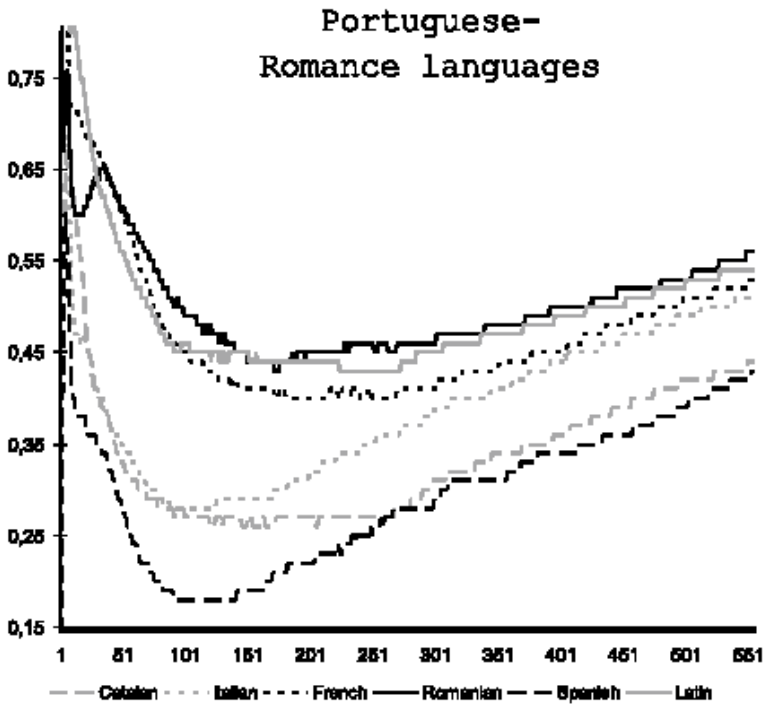
Latin-Romance Dillerinin Benzerliği

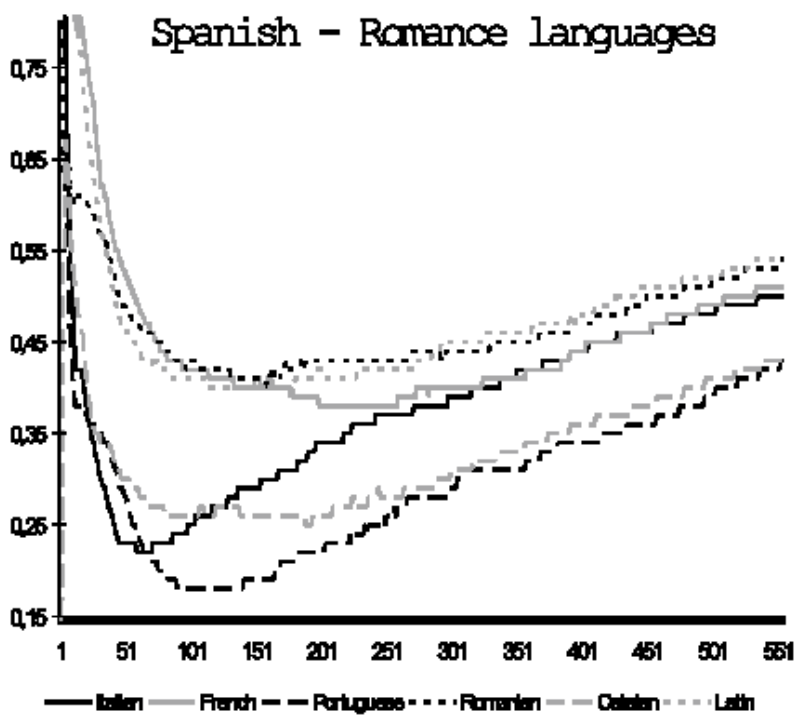


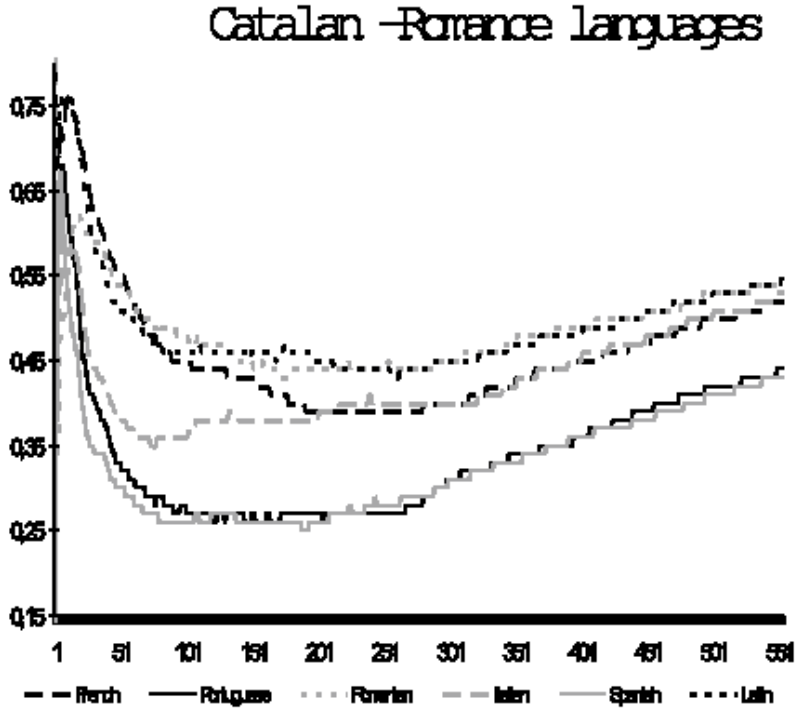

Romanian – Romance languages
0,75
0,65
0,55
0,45
0,35
0,25
0,15
1
51
101
151
201
251
301
351
401
451
501
551
Catalan
French
Italian
Portuguese
Spanish
Latin


Italian – Romance languages
0,75
0,65
0,55
0,45
0,35
0,25
0,15
1
51
101
151
201
251
301
351
401
451
501
551
Catalan
French
Portuguese
Romanian
Spanish
Latin


French – Romance languages
0,75
0,65
0,55
0,45
0,35
0,25
0,15
1
51
101
151
201
251
301
351
401
451
501
551
Romanian
Portuguese
Catalan
Italian
Spanish
Latin


Portuguese-
Romance languages
0,75
0,65
0,55
0,45
0,35
0,25
0,15
1
51
101
151
201
251
301
351
401
451
501
551
Catalan
Italian
French
Romanian
Spanish
Latin


Spanish - Romance languages
0,75
0,65
0,55
0,45
0,35
0,25
0,15
1
51
101
151
201
251
301
351
401
451
501
551
Italian
French
Portuguese
Romanian
Catalan
Latin


Catalan -Romance languages
0,75
0,65
0,55
0,45
0,35
0,25
0,15
1
51
101
151
201
251
301
351
401
451
501
551
French
Portuguese
Romanian
Italian
Spanish
Latin

Zipf Yasası

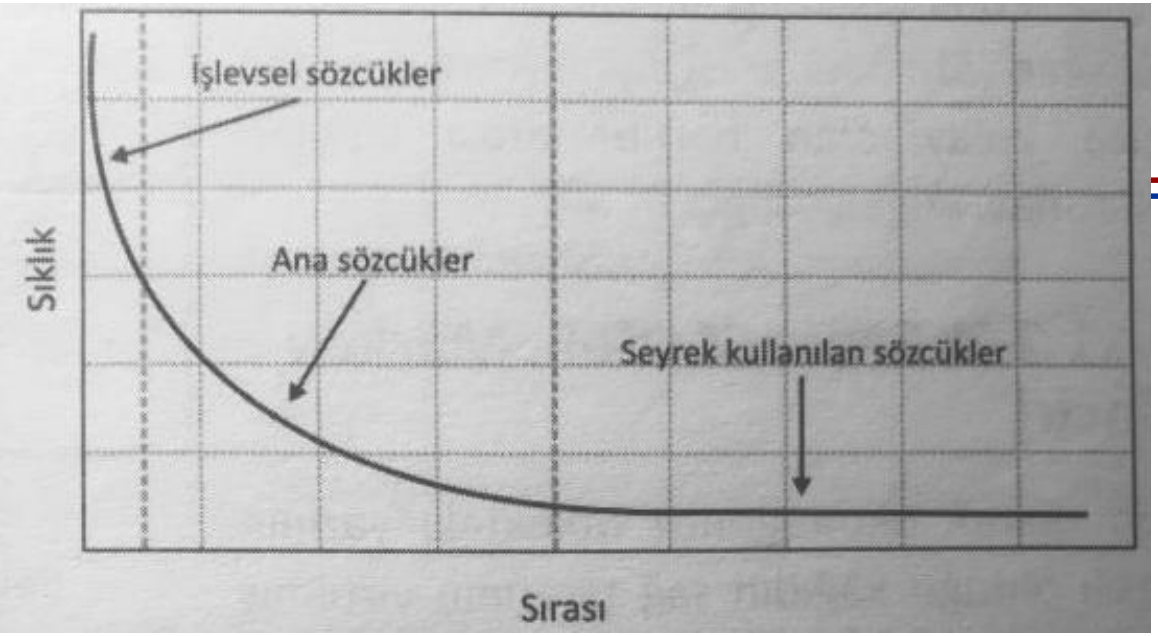
Bir dildeki kelimelerin kullanım sıraları ve sıklıkları arasındaki ilişkiyi tanımlar George Zipf'in (1902-1950)

Zipf'in bulgularına göre kelimeler kullanım sıklığına göre sıralandıklarında **ilk sıradaki kelime, yani en sık kullanılan kelime, ikinci sıradaki kelimenin iki katı kadar kullanılıyordu.**

10 sözcükten oluşan bir dil var ve bu dilde yazılmış bir metinde en sık kullanılan sözcük 100 defa kullanılmış. En sık kullanılan kelimeden en az kullanılan kelimeye göre yapılan sıralama listesi şöyle olacaktır:

1. sözcük => 100/1 = 100
2. sözcük => 100/2 = 50
3. sözcük => 100/3 = 33,3
4. sözcük => 100/4 = 25
5. sözcük => 100/5 = 20
6. sözcük => 100/6 = 16,6
7. sözcük => 100/7 = 14,3
8. sözcük => 100/8 = 12,5
9. sözcük => 100/9 = 11,1
10. sözcük => 100/10= 10

Zipf yasasının grafik yorumu



$$n(r) \propto \frac{1}{r^{\alpha}}$$

$$f(n) \propto \frac{1}{n^{\gamma}}$$

$$\gamma = 1 + \frac{1}{n^{\alpha}}$$

n→r.sıradaki kelimenin sıklık değeri
α → sabit bir değer
∝→yaklaşık değer

Türkçede sık kullanılan kelimelerin sıklık ve sıra ilişkisi

Kelime Tabanlı Dil Modelleri

Dil modeli, S cümlesinin olasılığını (likelihood/probability) hesaplamaya yardımcı olur, P(S).

En basit haliyle, herbir kelime bir sonraki w kelimesini eşit olasılıkla izler (0-gram).

V sözlüğünün boyunun |V| olduğunu farzedelim. Buna göre n uzunluğundaki S cümlesinin olasılığı (likelihood) = 1/|V| × 1/|V| … × 1/|V| olarak hesaplanır.

Eğer bir dilde 100,000 kelime varsa, gelecek olan herbir kelimenin olasılığı 1/100000 = .00001 dır.

- Kesin: gelecek olan her kelimenin olasılığı kelimenin frekansı ile ilişkilidir.
  - cümlenin olasılığı $S = P(w_1) \times P(w_2) \times \ldots \times P(w_n)$
  - herbir kelimenin olasılığı diğer kelimelerin olasılıklarından bağımsızdır.
- En kesin: daha önce verilmiş olan kelimenin olasılığına bakılır (n-gram).
  - S cümlesinin olasılığı = $P(w_1) \times P(w_2|w_1) \times \ldots \times P(w_n|w_{n-1})$
  - her kelimenin olasılığının diğer kelimelerin olasılıklarına bağlı olduğu farzedilir.

Bir string $w_1^n = w_1 \ldots w_n$ oluşsun.
Bu stringin olasılığı

$$P(w_1^n) = P(w_1)P(w_2|w_1)P(w_3|w_1..w_2)\ldots P(w_n|w_1\ldots w_{n-1})$$

$$= \prod_{k=1}^{n} P(w_k \mid w_1^{k-1})$$

Fakat bu yaklaşım genelde, bir kelime sırasının olasılığını belirlemek için çok yararlı değildir. Hesaplama maliyeti çok yüksektir.

Markov Yaklaşımı- Basit N-Grams

- N-gram model, gelecek kelimeyi tahmin edebilmek için önceki N-1 adet kelimeyi kullanır.

$$P(w_n \mid w_{n-N+1} w_{n-N+2} \ldots w_{n-1})$$

P(rabbit|I saw a) yerine, P(rabbit|a) kullanılabilir mi?

N=2 (bigram): $P(w_1^n) \approx \prod_{k=1}^{n} P(wk|wk_{-1});\ w_0 = \langle start \rangle$

unigrams: P(dog)
bigrams: P(dog | big)
trigrams: P(dog | the big)
quadrigrams: P(dog | chasing the big)

N-Grams kullanımı

Hatırla

N-gram: $P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-N+1}^{n-1})$

Bigram: $P(w_1^n) \approx \prod_{k=1}^{n} P(w_k \mid w_{k-1})$

Bigram grameri:

Cümlenin olasılığı P(cümle), cümle içerisinde yer alan bütün bigram'ların olasılıklarının çarpına yakın bir değerdir.

Örnek:

P(I want to eat Chinese food) =

P(I | <start>) P(want | I) P(to | want) P(eat | to)

P(Chinese | eat) P(food | Chinese)

Bigram Gramer Parçaları

Eat on	.16	Eat Thai	.03
Eat some	.06	Eat breakfast	.03
Eat lunch	.06	Eat in	.02
Eat dinner	.05	Eat Chinese	.02
Eat at	.04	Eat Mexican	.02
Eat a	.04	Eat tomorrow	.01
Eat Indian	.04	Eat dessert	.007
Eat today	.03	Eat British	.001

<start> I	.25	Want some	.04
<start> I'd	.06	Want Thai	.01
<start> Tell	.04	To eat	.26
<start> I'm	.02	To have	.14
I want	.32	To spend	.09
I would	.29	To be	.02
I don't	.08	British food	.60
I have	.04	British restaurant	.15
Want to	.65	British cuisine	.01
Want a	.05	British lunch	.01

Cümlenin olasılığının hesaplanması

P(I want to eat British food) =
P(I|<start>) P(want|I) P(to|want) P(eat|to)
P(British|eat) P(food|British) =
.25×.32×.65×.26×.001×.60 = .000080

P(I want to eat Chinese food) = .00015

Olasılıklar dünyanın bildiği bir gerçeği göstermektedir.

N-grams sonuçları

Sparse data

Eğitim seti içerisinde bütün N-gram'lar yer almayabilir ve bu n-gram'ların frekansı sıfır olarak alınır, bu yüzden yumuşatma (smoothing) tekniklerine ihtiyaç duyulur.

P("And nothing but the truth") ≈ 0.001

P("And nuts sing on the roof") ≈ 0

Örnek

Bigram grameri VxV boyutunda bir olasılıklar matrisidir.
V, sözlük boyutu

	I	Want	To	Eat	Chinese	Food	lunch
I	8	1087	0	13	0	0	0
Want	3	0	786	0	6	8	6
To	3	0	10	860	3	0	12
Eat	0	0	2	0	19	2	52
Chinese	2	0	0	0	0	120	1
Food	19	0	17	0	0	0	0
Lunch	4	0	0	0	0	1	0

Unigram değerleri

I	Want	To	Eat	Chinese	Food	Lunch
3437	1215	3256	938	213	1506	459

$$P(w_n|w_{n-1}) = C(w_{n-1}w_n)/C(w_{n-1})$$

Computing the probability of I I

$$P(I|I) = C(I\ I)/C(I) = 8 / 3437 = .0023$$

Bigram Olasılık

	I	want	to	eat	Chinese	food	lunch
I	.0023	.32	0	.0038	0	0	0
want	.0025	0	.65	0	.0049	.0066	.0049
to	.00092	0	.0031	.26	.00092	0	.0037
eat	0	0	.0021	0	.020	.0021	.055
Chinese	.0094	0	0	0	0	.56	.0047
food	.013	0	.011	0	0	0	0
lunch	.0087	0	0	0	0	.0022	0

Yumuşatma (Smoothing) Teknikleri

Çok geniş bir derleme sahip olsak bile N-gram eğitim matrisi sparse bir matrisdir (Zipf's law).

Çözüm: Gözükmeyen n-gram olasılıklarını tahmin etmek

Types (V): Derlem içerisinde yer alan ayrık kelime sayısı (vocabulary size)

Token (N_T): Derlem içerisindeki toplam kelime sayısı

Şimdiye kadar gözüken kelime sayısı (T): Derlemde görülen ayrık kelime sayısı ($T<<V$ ve N_T)

Add-one Smoothing

Her n-gram değerine 1 eklenir.

$N_T/(N_T+V)$ katsayısı ile normalize edilir

Yumuşatılmış toplam: $c_i' = (c_i+1)\frac{N_T}{N_T+V}$
(Smoothed count)

Yumuşatılmış olasılık
(Smoothed probability): $P'(w_i) = c_i'/N_T$

Add-one Smoothed Bigrams

$$P(w_n|w_{n-1}) = C(w_{n-1}w_n)/C(w_{n-1})$$

	I	want	to	eat	Chinese	food	lunch
I	8	1087	0	13	0	0	0
want	3	0	786	0	6	8	6
to	3	0	10	860	3	0	12
eat	0	0	2	0	19	2	52
Chinese	2	0	0	0	0	120	1
food	19	0	17	0	0	0	0
lunch	4	0	0	0	0	1	0

	I	want	to	eat	Chinese	food	lunch
I	.0023	.32	0	.0038	0	0	0
want	.0025	0	.65	0	.0049	.0066	.0049
to	.00092	0	.0031	.26	.00092	0	.0037
eat	0	0	.0021	0	.020	.0021	.055
Chinese	.0094	0	0	0	0	.56	.0047
food	.013	0	.011	0	0	0	0
lunch	.0087	0	0	0	0	.0022	0

$$P'(w_n|w_{n-1}) = [C(w_{n-1}w_n)+1]/[C(w_{n-1})+V]$$

	I	want	to	eat	Chinese	food	lunch
I	9	1088	1	14	1	1	1
want	4	1	787	1	7	9	7
to	4	1	11	861	4	1	13
eat	1	1	3	1	20	3	53
Chinese	3	1	1	1	1	121	2
food	20	1	18	1	1	1	1
lunch	5	1	1	1	1	2	1

	I	want	to	eat	Chinese	food	lunch
I	.0018	.22	.00020	.0028	.00020	.00020	.00020
want	.0014	.00035	.28	.00035	.0025	.0032	.0025
to	.00082	.00021	.0023	.18	.00082	.00021	.0027
eat	.00039	.00039	.0012	.00039	.0078	.0012	.021
Chinese	.0016	.00055	.00055	.00055	.00055	.066	.0011
food	.0064	.00032	.0058	.00032	.00032	.00032	.00032
lunch	.0024	.00048	.00048	.00048	.00048	.00096	.00048

Örnek

$$P_i^* = \frac{C_i + 1}{N + V}, \; i = 1, 2, \ldots, t$$

	Bugün	de	her	zamanki	gibi	eve	gidiyorum
Bugün	0	1	0	0	0	0	0
de	0	0	1	0	0	0	0
her	0	0	0	1	0	0	0
zamanki	0	0	0	0	1	0	0
gibi	0	0	0	0	0	1	0
eve	0	0	0	0	0	0	1
gidiyorum	0	0	0	0	0	0	0

$$P_{Bugün\ \text{de}} = \frac{1}{5}, \; P_{de\ \text{her}} = \frac{1}{5}, \; P_{her\ \text{zamanki}} = \frac{1}{5}, \; P_{\text{zamanki gibi}} = \frac{1}{5}, \; P_{gibi\ \text{gidiyorum}} = \frac{1}{5}$$

	Bugün	de	her	zamanki	gibi	eve	gidiyorum
Bugün	1	2	1	1	1	1	1
de	1	1	2	1	1	1	1
her	1	1	1	2	1	1	1
zamanki	1	1	1	1	2	1	1
gibi	1	1	1	1	1	2	1
eve	1	1	1	1	1	1	2
gidiyorum	1	1	1	1	1	1	1

$$P_{Bugün\ \text{de}} = \frac{2}{5+49} = \frac{2}{54}, \; P_{de\ \text{her}} = \frac{2}{54}, \; P_{her\ \text{zamanki}} = \frac{2}{54}, \; P_{\text{zamanki gibi}} = \frac{2}{54}, \; P_{gibi\ \text{gidiyorum}} = \frac{2}{54}$$

Diğer yumuşatma teknikleri: Good-Turing

Balık tutmaya çıktığınızı hayal edin...

Ve 10 tane aynalı sazan (carp), 3 tane morina (cod), 2 tane tuna, 1 tane alabalık (trout), 1 tane som balığı (salmon), 1 tane de yılan balığı (eel) yakalamış olun.

Bir sonraki yakalanacak olan balığın yeni bir tür olma olasılığı nedir ?

karşımıza 1 kez çıkan balık türü/toplam tutulan balık sayısı=3/18

Back-off Yöntemi

Hatırlatma :N-gram'lar her zaman (N-1) gram'a göre daha duyarlıdır.

Fakat, N-gram'lar (N-1) gram'a göre daha fazla sparse 'tır.

Bu ikisi nasıl birleştirilir ?

N-gram'ın frekans değeri uygun değilse (sıfır ise) vazgeçilir (back-off) ve (n-1) gram'a dönülür. Monogram'a kadar devam edilir. Recursive bir yapı sözkonusudur.

$$\hat{P}(w_i|w_{i-2}w_{i-1}) = \begin{cases} \tilde{P}(w_i|w_{i-2}w_{i-1}), & \text{if } C(w_{i-2}w_{i-1}w_i) > 0 \\ \alpha_1 \tilde{P}(w_i|w_{i-1}), & \text{if } C(w_{i-2}w_{i-1}w_i) = 0 \text{ and } C(w_{i-1}w_i) > 0 \\ \alpha_2 \tilde{P}(w_i), & \text{otherwise} \end{cases}$$

N-gram modelin Biçimbirimsel Belirsizliğin Giderilmesinde kullanımı

Bazı kelimelerin biçimbirimsel analizleri birden fazla olmaktadır. Doğru olanı seçebilmemiz için istatistiksel yaklaşımlardan *n-gram* modelini kullanabiliriz.

«Çoban bizim için sürüden bir **koyun** seçti»
«Oyuna devam etmek istiyorsanız beşer lira **koyun**»

- n değerini 2 alalım (n değeri büyüdükçe daha kesin sonuç alınır)
- Belirsizliği giderilecek olan kelime w_t olsun
- Derlem içerisinde w_t ve w_{t-1} peşi sıra gelme olasılığı çıkarılsın

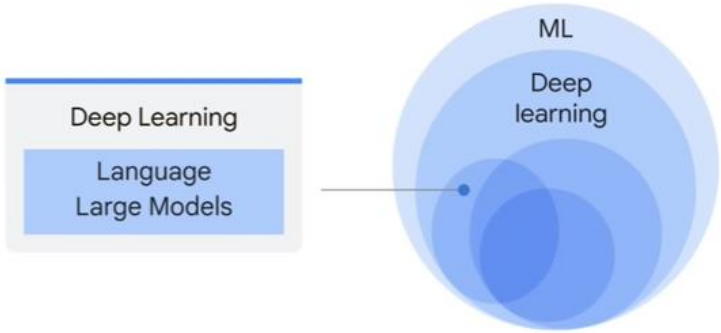
1. cümle için

Oran	Nitelik	Oran	Nitelik	Oran	Nitelik
0,01	Fiil	0	önad	0	Bağlaç
0,5	İsim	0	belirteç	0	Yansıma
0,01	Zamir	0,001	edat	0	Özel ad
0	sayı	0,007	ünlem	0,0001	kısaltma

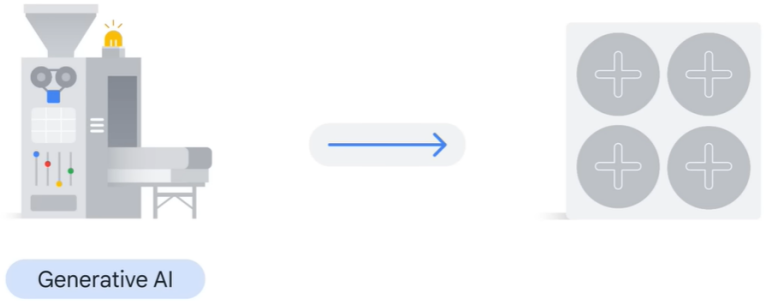
2.cümle için

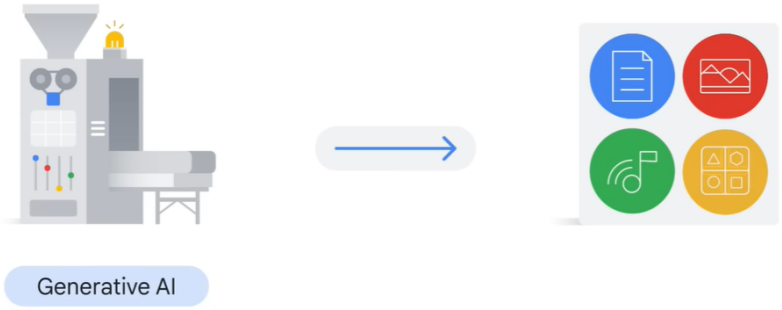
Oran	Nitelik	Oran	Nitelik	Oran	Nitelik
0,8	Fiil	0	önad	0	Bağlaç
0,01	İsim	0	belirteç	0	Yansıma
0,01	Zamir	0,002	edat	0	Özel ad
0	sayı	0,007	ünlem	0	kısaltma

Büyük Dil Modelleri-LLM



- Büyük Dil Modelleri / LLM derin öğrenmenin bir alt kümesidir
- Derin öğrenme tekniklerini ve özellikle de dönüştürücüleri (transformers) kullanan gelişmiş yapay zeka modelleridir
- Büyük Dil Modelleri dil çevirisi, metin sınıflandırma, duygu analizi, metin üretme, soru cevaplama, içerik oluşturma gibi doğal dil işleme (NLP) görevlerini gerçekleştirmek için dönüştürücüleri kullanır





Üretken Yapay Zeka (Generative AI)

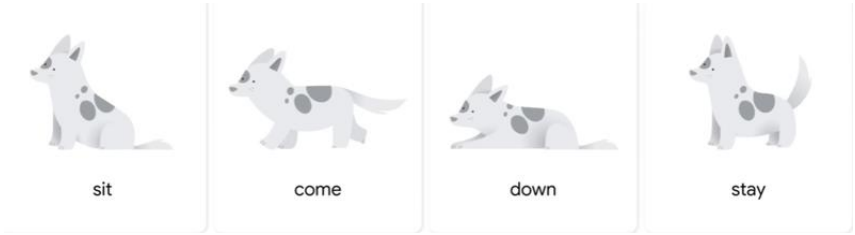
- Metin
- Resim
- Ses
- Sentetik veriler

Yeni içerikler üreten bir yapay zeka türü

Büyük Dil Modeli Nedir ?

Önceden çok büyük miktarda veriler ile eğitilmiş, sonra belli amaçlar için ayarlanabilen (fine tunning) model

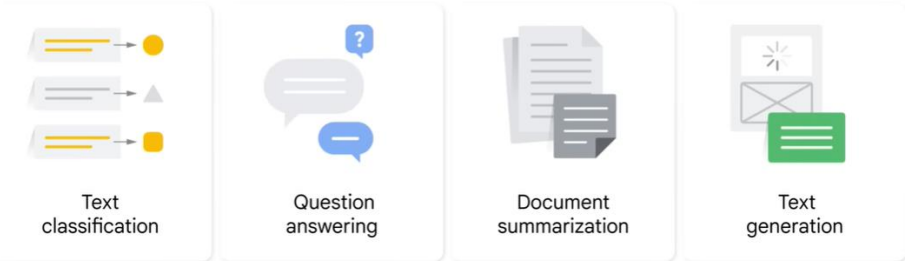
Ne demek ?



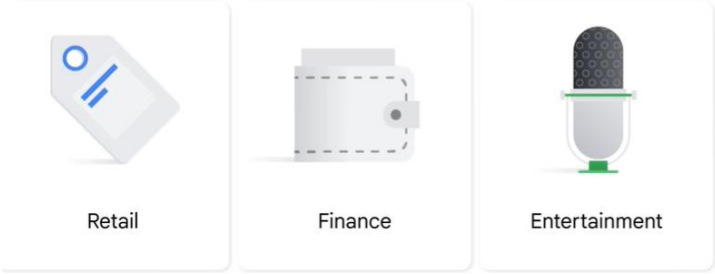
- Bir köpeği eğittiğimizi düşünelim
  - Otur, kalk, yat, gel gibi genel komutları öğretiriz

- Eğer, bu köpeğin bir polis, avcı, rehber köpeği olması gerekiyorsa
  - Bu eğitimlerin üzerine özel eğitimler eklenir

Büyük Dil Modelleri için de aynı şey geçerlidir



- Model genel amaçlar için eğitilir
  - Yaygın dil problemlerini çözmek için…
    Sınıflandırma, özetleme, soru cevaplama, …



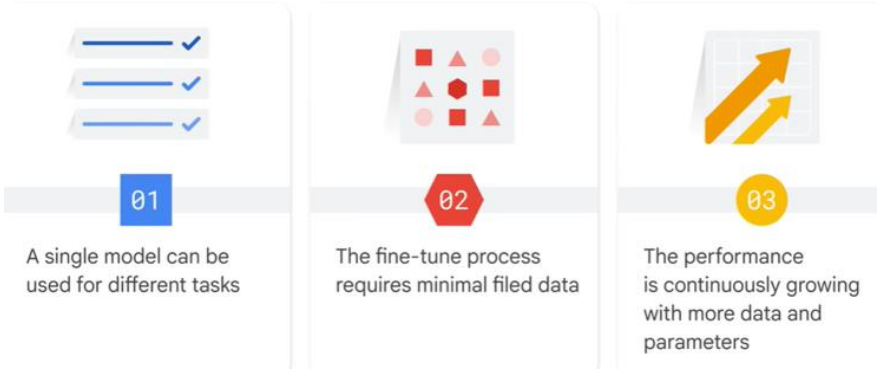
- Model özel amaçlar için ayarlanır (fine tuned)
  - Parakende, finans, eğlence, …
  - Daha küçük bir veri seti

Büyük Dil Modelleri

- Büyük
  - Petabyte seviyelerinde büyük bir veri seti
  - Çok sayıda parametre
- Genel Amaç
  - Yaygın problemleri çözmek
  - Kaynakların kısıtlı olması
- Pre-trained ve fine-tuned

➢ Parametre, model eğitiminde öğrenilen bilg
Modelin problem çözme becerisini

Büyük Dil Modellerini Kullanmanın Faydaları

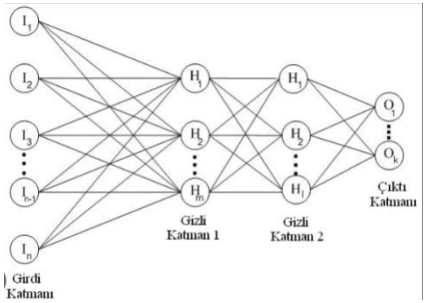


- ❖ Tek bir model farklı görevler için kullanılabilir
- ❖ Belirli problemleri çözmek için küçük bir veri seti ile uyarlama yapmak yeterli olur
- ❖ Daha fazla veri ve parametre eklendiğinde performansları sürekli artar

Geleneksel programlama

Kural tabanlı
Tür: hayvan
Bacak : 4
Kulak : 2
Kürk : evet
…….

Yapay Sinir Ağları



Ağı kedi, köpek resimleri ile eğitip, kedi resmini verip bu kedi mi diye sorduğumuzda kedi cevabını alabilmek

Üretken Yapay Zeka

Kedi nedir?

Kedi hakkında öğrendiği her şeyi veri

- Uzman olmak gerekmez
- Veri setine ihtiyaç yoktur
- Modeli eğitmek gerekmez
- En uygun prompt yazmak

Büyük Dil Modelleri (LLM)  **Üretken Yapay Zeka (Generative AI)**

- Üretken yapay zeka, görüntü, metin, müzik, video gibi yeni içerikler oluşturmaya odaklanır
- Eğitim verilerinden öğrendikleri kalıplara ve yapılara dayalı olarak çıktılar üretebilen algoritmalardır

  Popüler üretken yapay zeka teknikleri arasında *Generative Adversarial Networks* (GANs)
  Variational Autoencoders (VAEs)
  Recurrent Neural Networks (RNNs)

- Büyük dil modelleri (LLM'ler), insan benzeri metinleri anlama ve üretme konusunda uzmanlaşmış özel bir üretken yapay zeka türüdür.
- Bu modeller gramer, bağlam, stil varyasyonları ve diğer dilsel nüansları öğrenmek için büyük miktarda metin verisi kullanılarak eğitilir.
- LLM'ler, girdi dizilerini etkili bir şekilde işlemek için dikkat mekanizmalarına sahip dönüştürücü mimarileri gibi gelişmiş derin öğrenme tekniklerini kullanır.

Büyük Dil Modeli Türleri

1. Dil Temsil Modeli

- Birçok doğal dil işleme uygulaması, insan dilini anlama ve üretme yeteneğine sahip dil temsil modellerine dayanır. Örneğin, GPT (*Generative Pre-trained Transformer*), BERT (*Bidirectional Encoder Representations from Transformers*) ve RoBERTa (*Robustly Optimized BERT Pre-training Approach*) gibi modeller bu kategoriye örnektir.
- Bu modeller genellikle geniş metin verileri üzerinde önceden eğitilir ve daha sonra belirli görevler için özelleştirilebilir. Örneğin: metin sınıflandırma veya dil üretimi gibi

2. Sıfır Atış Modeli

- Belirli eğitim verileri olmadan görevleri yerine getirme yetenekleriyle bilinir. Bu modeller daha önce hiç görmedikleri görevler için genelleme yapabilir ve tahminlerde bulunabilir veya metin oluşturabilir.
- GPT-3 (12.888 embeding size) sıfır atışlı modele bir örnektir. Soruları yanıtlayabilir, dilleri çevirebilir ve minimum ince ayar ile çeşitli görevleri yerine getirebilir.

3. Multimodal Model

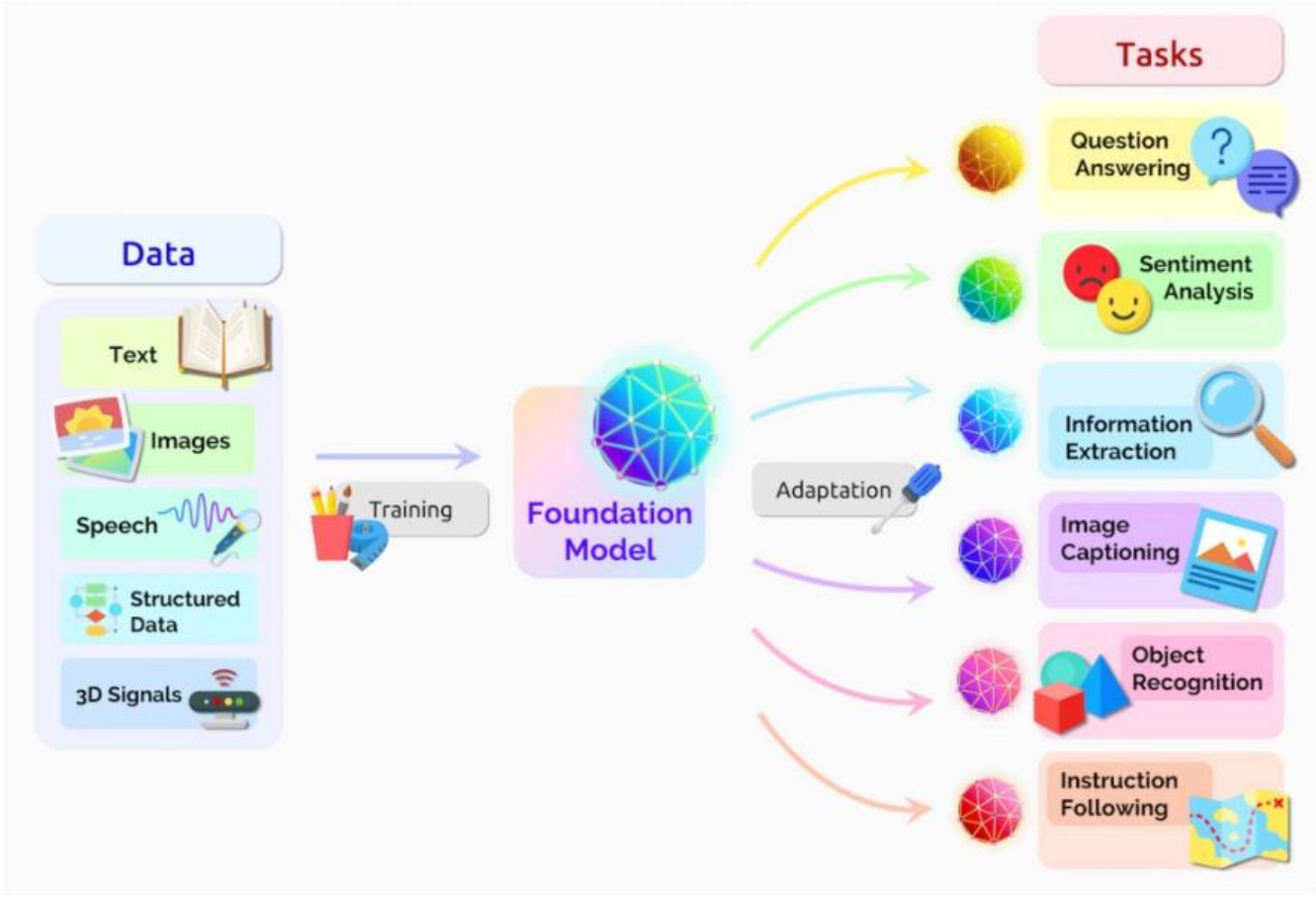
- LLM'ler başlangıçta metin için tasarlanmıştır. Ancak, multimodal modeller hem metin hem de görüntü verileriyle çalışır.
- Bu modeller, farklı modaliteler arasında içeriği anlamak ve oluşturmak için tasarlanmıştır.

OpenAI'nin **CLIP**'i, metni görüntülerle ilişkilendirebilen ve bunun tersini yapabilen çok modlu bir modeldir. Bu da onu görüntü altyazısı ve metin tabanlı görüntü oluşturma gibi görevler için kullanışlı hale getirir.

4. İnce Ayarlı veya Alana Özel Modeller

- Önceden eğitilmiş dil temsil modelleri çok yönlü olmakla birlikte, belirli görevler veya alanlar için her zaman en iyi performansı göstermeyebilir.
- İnce ayarlı modeller, belirli alanlardaki performanslarını artırmak için alana özgü veriler üzerinde ek eğitimden geçirilmelidir.

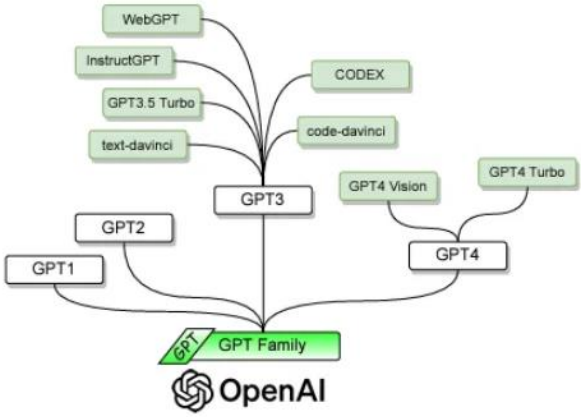
Örneğin: GPT-3 modeli, alana özgü bir tıbbi sohbet robotu oluşturmak veya tıbbi teşhise yardımcı olmak için tıbbi veriler üzerinde ince ayar yapılabilir.


Data
Text
Images
Speech
Structured Data
3D Signals
Training
Foundation Model
Adaptation
Tasks
Question Answering
Sentiment Analysis
Information Extraction
Image Captioning
Object Recognition
Instruction Following

ÖNE ÇIKAN DİL MODELLERİ

1. OpenAI GPT

- GPT ilk versiyonu 2018 yılında çıktı
- Farklı internet metinleri üzerinde eğitildi. Belirli bir kelime dizisinde bir sonraki kelimeyi tahmin ederek tutarlı ve bağlamsal olarak ilgili cümleler oluşturabiliyordu
- 2020 yılında 175 milyar parametreye sahip GPT-3 çıktı. Talimatlara (prompt) dayalı olarak insan benzeri metin ve kod üreten ilk model oldu
- GPT 3.5 temel alan ve RLHF (Reinforcement Learning from Human Feedback) ile geliştirilmiş ChatGPT piyasaya sürüldü (Kasım 2022)
- ChatGPT hizmetinin bir parçası olarak da Nisan 2023 de, API aracılığıyla kullanılabilen en yetenekli modeli olan GPT-4'ü çıkarıldı. Context length of 32k tokens. GPT-4 Turbo 128k tokens (about 240 pages)



2. Anthropic Claude

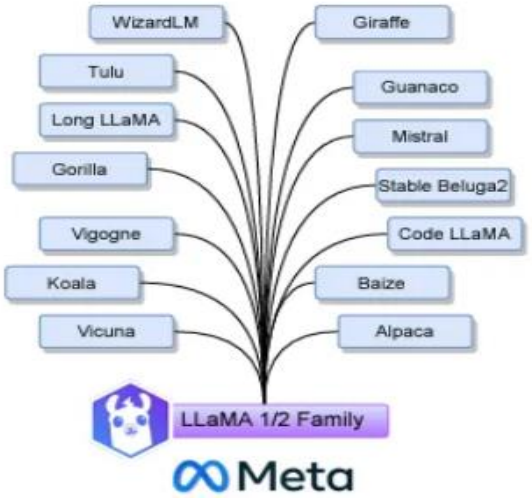
- Aynı anda 100 bin tokena kadar işlem yapabilen bir bağlam penceresine sahiptir
- Uzun belgeleri kolaylıkla işleyebilir, kapsamlı metin analizi ve anlama için idealdir

3. Microsoft Research ORCA

- 13 milyar parametreye sahip Meta LlaMA modelinin bir varyantına dayanan bir LLM'dir
- Modelin küçük boyutu basit bir dizüstü bilgisayarda çalışabilir
- Geleneksel büyük dil modelleri tarafından kullanılan mantık işleme yöntemlerini kopyalayarak mevcut açık kaynak modellerini aşmak üzere tasarlanmıştır
- Çok daha az parametre ile GPT-4'ün performans seviyeleri ile rekabet eder, çeşitli görevlerde GPT-3.5'e eşittir

4. Meta LLaMA (Large Language Model Meta AI)

- Meta ve Microsoft tarafından 2023 yılında LLaMa 2 çıkarıldı
- Llama 2, araştırma ve ticari kullanım için açık kaynaklı ve ücretsiz olarak açıldı
- Llama 2'ye Hugging Face, Amazon Web Services ve Microsoft Azure üzerinden ücretsiz olarak erişilebiliniyor
- Llama 2 önceden eğitilmiş üç farklı model boyutuna sahip: 7 milyar, 13 milyar ve 70 milyar parametre (Hepsi erişime açık)

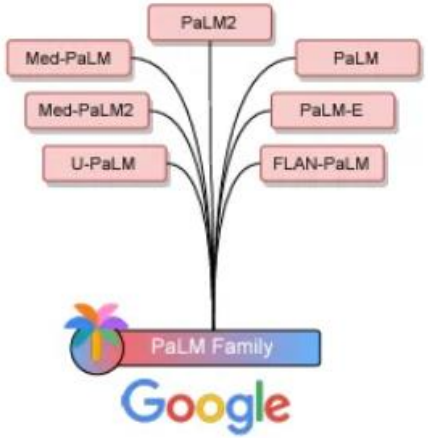


5. Cohere

- Kurumsal hizmetler geliştirmek için bir platform sunuyor
- Cohere'in LLM'lerine Google Cloud'un yapay zeka ve makine öğrenimi donanımı ve altyapısı destek veriyor

6. BERT

- BERT (Bidirectional Encoder Representations from Transformers) 2018 yılında piyasaya çıktı
- Bir kelimenin yalnızca solundaki bağlamı dikkate alan GPT'nin aksine BERT her kelimenin iki tarafına da bakar. Bu çift yönlü yaklaşım, modelin dili anlama ve üretme performansını artırır
- 3,3 milyar kelimeden oluşan veri seti, eğitimi için 64 TPU, 4 gün boyunca çalışmıştır
- Google Arama motorunun temel parçası olmuştur
- DistilBERT, Google BERT'in daha hafif bir versiyonunu sunmakta; BERT performansının %95'inden fazlasını korurken %60 daha hızlı çalışıyor.



7- PaLM (Pathways Language Model)

- Google AI tarafından geliştirilmiştir (Nisan 2022'de duyuruldu)
- 540 milyar parametreli dönüştürücü tabanlı büyük bir dil modelidir
  - Modelin başarısını test etmek için PaLM'ın daha küçük sürümleri 8 ve 62 milyar parametreli modelleri mevcuttur
- Sağduyulu ve matematiksel akıl yürütme, şaka, kod oluşturma, çeviri, vs. çeşitli görevleri yerine getirebilir
- Düşünce zinciri akıl yürütmesi (Chain of thought reasoning) ile birleştirildiğinde mantığa dayalı sorularda da iyi performans göstermektedir
- Api mart 2023 sonrası halka açıldı
- PaLM, çeşitli doğal dil görevlerini ve kullanım durumlarını içeren 780 milyar token ile önceden eğitilmiştir
  - filtrelenmiş web sayfaları, kitaplar, Wikipedia sayfaları, haber makaleleri, GitHub kaynak kodları ve sosyal medya içerikleri
- PaLM 540B, büyük TPU yapılandırması olan bir model ve veri paralelliği kombinasyonu kullanılarak bağlanan, 768 ana bilgisayara bağlı her bölmede 3.072 TPU v4 yongası bulunan iki TPU v4 bölmesi üzerinde eğitildi

8-Google BARD/Gemini/Gemma

- Bard güç veren dil modeli PaLM2 dir
- Bard Mayıs 2023'te 180'den fazla ülkede kullanıma sunuldu
- 2024 yılında Bard, Gemini adı ile değiştirildi
- Gemini dil, ses, kod ve video anlama yeteneklerine sahip, çoklu modlu bir LLM
- Gemma 2B ve Gemma 7B (Şubat 2024) lightweight model olarak çıktı
  (laptop/masaüstü bilgisayarda çalıştırabilir)
- Bir trilyona yakın parametre
- Gemma modeli sadece decoder yapısına sahiptir
- Ticari ve araştırma amaçlıdır
- Gemini, önyargı ve toksisite gibi riskler karşısında güvenlik testine tabi tutulmuş ve bir dereceye kadar LLM güvenliğini sağlamak için önlem alınmıştır

9-Mistral

- Mistral Large, OpenAI tarafından geliştirilen ve GPT-3'ten 10 kat daha büyük bir dil modeli
- Mistral Large, 600 milyar parametreyle metin üretme, dil çevirme ve kod yazmada başarılı
- 32K bağlam penceresine sahip

Diğerleri…

- Phind 70, Google AI'ın 70 milyar parametreli modeli. Diyalog kurmada ve açık uçlu sorulara cevap vermekte oldukça başarılı
- Samba 1, Microsoft'un 100 milyar parametreli modeli. Metin ve koddan oluşan büyük bir veri kümesi üzerinde eğitilmiş ve metin üretme, dil çevirme ve kod yazma gibi görevlerde Mistral Large ve Phind 70'e rakip olabilecek performansı var

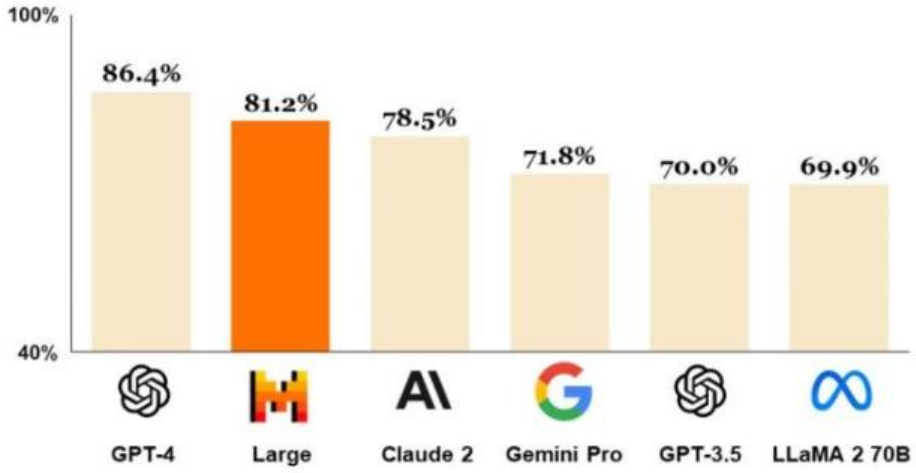
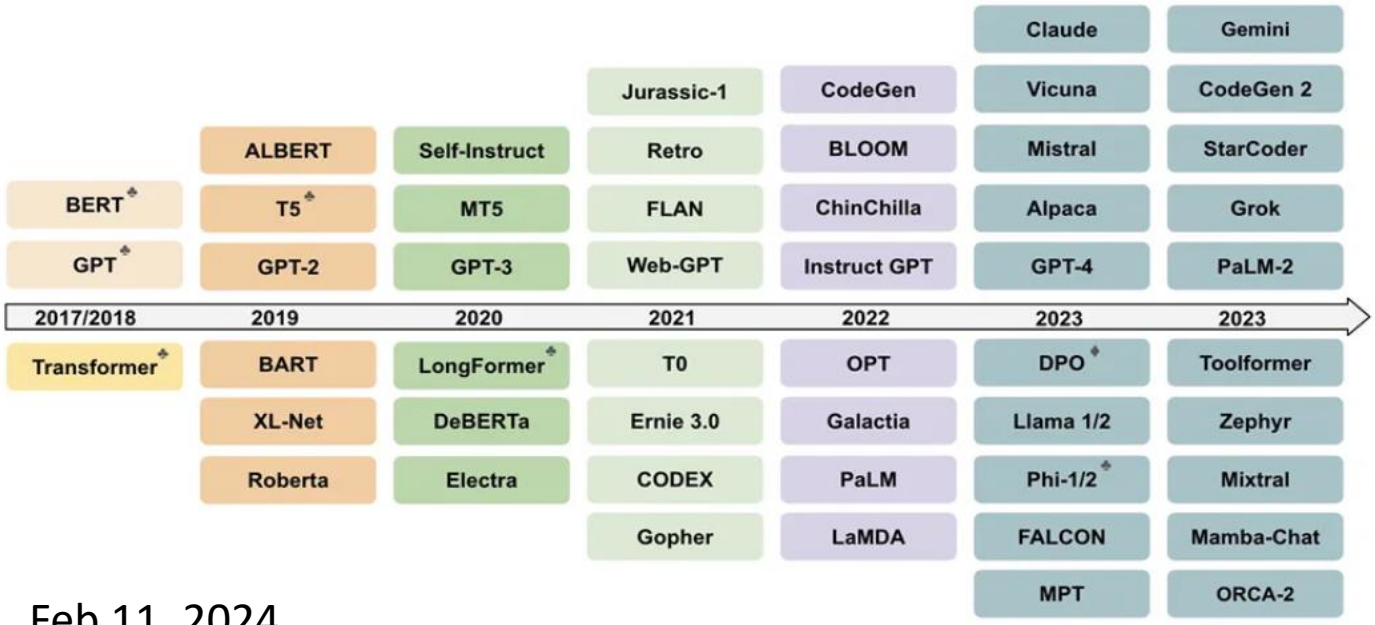


Figure 1: Comparison of GPT-4, Mistral Large (pre-trained), Claude 2, Gemini Pro 1.0, GPT 3.5 and LLaMA 2 70B on MMLU (Measuring massive multitask language understanding).


Claude
Gemini
Jurassic-1
CodeGen
Vicuna
CodeGen 2
ALBERT
Self-Instruct
Retro
BLOOM
Mistral
StarCoder
BERT
T5
MT5
FLAN
ChinChilla
Alpaca
Grok
GPT
GPT-2
GPT-3
Web-GPT
Instruct GPT
GPT-4
PaLM-2
2017/2018
2019
2020
2021
2022
2023
2023
Transformer
BART
LongFormer
T0
OPT
DPO
Toolformer
XL-Net
DeBERTa
Ernie 3.0
Galactia
Llama 1/2
Zephyr
Roberta
Electra
CODEX
PaLM
Phi-1/2
Mixtral
Gopher
LaMDA
FALCON
Mamba-Chat
MPT
ORCA-2

Feb 11, 2024

LLM leri neye göre seçmeliyiz?

❖ Hangi dil modelini seçeceğimiz çeşitli faktörlere ve özel kullanım alanlarına göre değişir

1. İçerik Oluşturma : Yüksek kaliteli metin oluşturmak istiyorsak GPT veya Cohere Technologies'in modeli

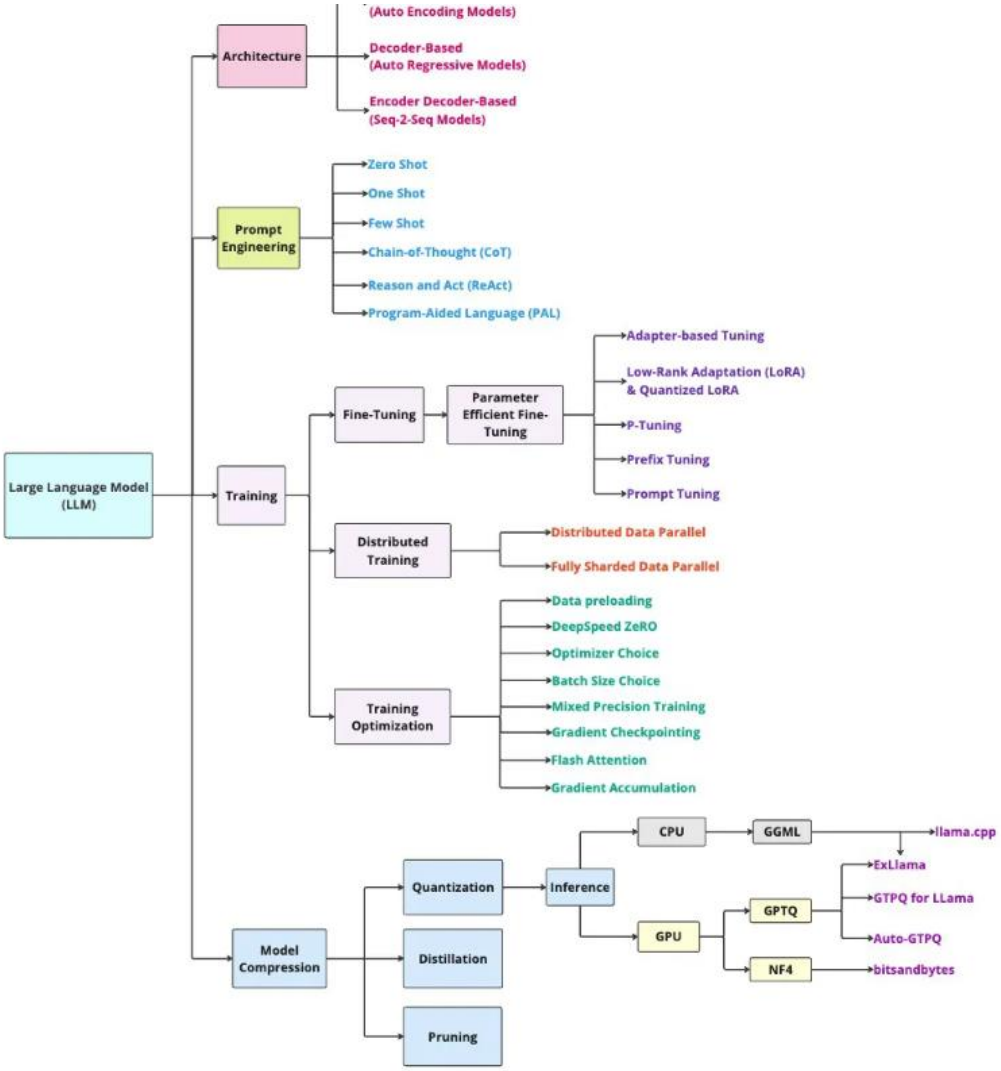
2. Müşteri Destek Sohbet Robotları : Müşteri sorularını etkili bir şekilde ele alan diyalog yapısında konuşmayı devam ettirmek için LaMDA veya OpenAI'nin ChatGPT'si

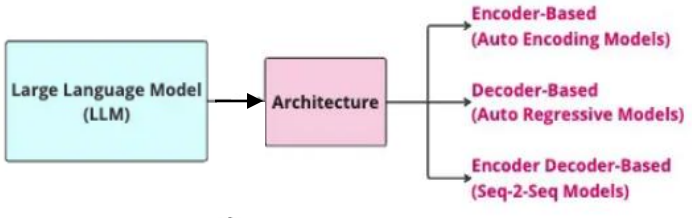
3. Pazarlamada Duygu Analizi : BERT tabanlı modeller, sosyal medya gönderilerinden veya ürün incelemelerinden duygu tespiti gibi görevlerde başarılı, işletmelerin veri odaklı kararlar almasına yardımcı olur

4. Yasal Belge Analizi : Yasal dil işleme konusunda uzmanlaşmış ince ayarlı LLM'ler sözleşmeler ve hukuki konularda daha iyi performans sunar

5. Tıbbi Bilgi Çıkarımı : Sağlık hizmetleri alanında, tıbbi dili anlamak için uyarlanmış bir LLM kullanmak, hasta kayıtlarını veya bilimsel literatürü analiz ederken daha doğru sonuçlar verir

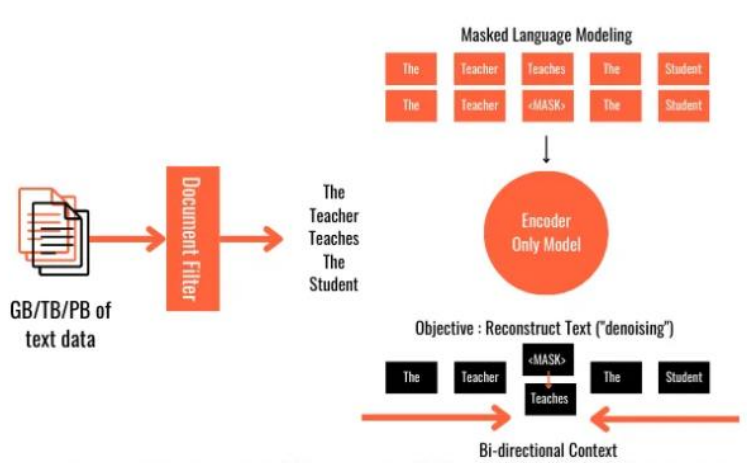
6. Kod Üretimi ve Programlama : GPT gibi modeller, doğal dil tanımlamalarına dayalı kod parçacıkları üretme konusunda yetenekli


Large Language Model (LLM)
Architecture
(Auto Encoding Models)
Decoder-Based (Auto Regressive Models)
Encoder Decoder-Based (Seq-2-Seq Models)
Prompt Engineering
Zero Shot
One Shot
Few Shot
Chain-of-Thought (CoT)
Reason and Act (ReAct)
Program-Aided Language (PAL)
Training
Fine-Tuning
Parameter Efficient Fine-Tuning
Adapter-based Tuning
Low-Rank Adaptation (LoRA) & Quantized LoRA
P-Tuning
Prefix Tuning
Prompt Tuning
Distributed Training
Distributed Data Parallel
Fully Sharded Data Parallel
Training Optimization
Data preloading
DeepSpeed ZeRO
Optimizer Choice
Batch Size Choice
Mixed Precision Training
Gradient Checkpointing
Flash Attention
Gradient Accumulation
Model Compression
Quantization
Distillation
Pruning
Inference
CPU
GGML
llama.cpp
GPU
GPTQ
ExLlama
GTPQ for LLama
Auto-GTPQ
NF4
bitsandbytes



Three categories of LLM architecture

AutoEncoding Models (Encoder Only)



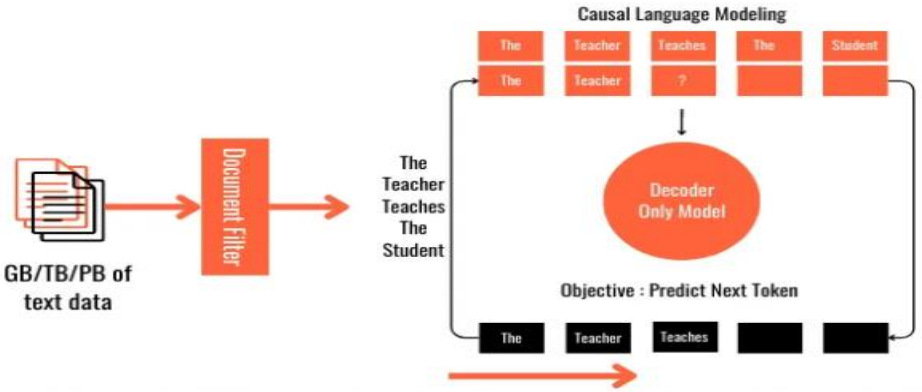
USE CASES

- Sentiment Analysis
- Named Entity Recognition
- Word Classification

EXAMPLES

- BERT
- ROBERTA

AutoRegressive Models (Decoder Only)



USE CASES

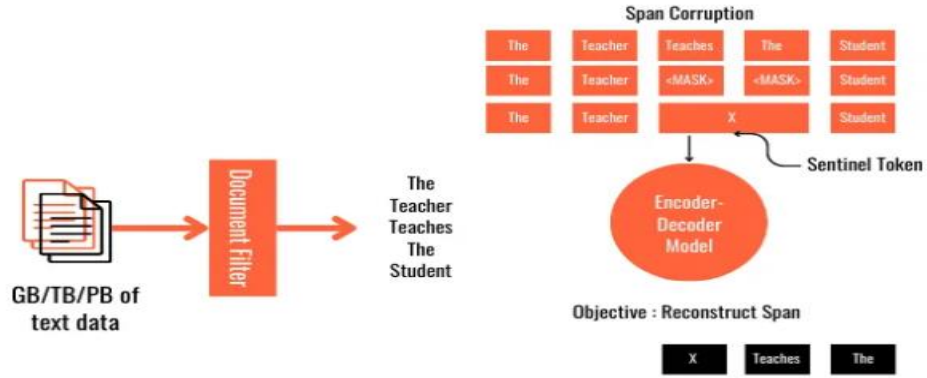
- Text Generation

  (Most common architecture now and larger models can perform a variety of tasks)

EXAMPLES

- GPT
- BLOOM

Sequence-to-Sequence Models (Encoder-Decoder)



USE CASES

- Translation
- Text Summarisation
- Question Answering

EXAMPLES

- T5
- BART

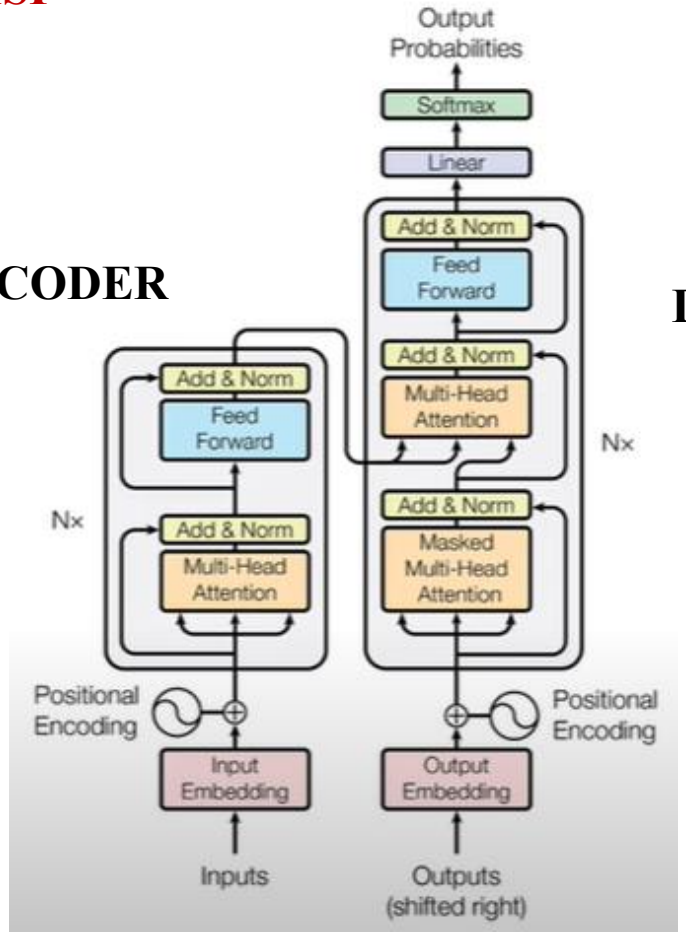
TRANSFORMER NEDİR?

- Transformer kavramı, Ashish Vaswani, Noam Shazeer, Niki Parmar ve diğer beş yazar tarafından Google'ın 2017 yılında yayınlanan "**Attention Is All You Need**" başlıklı makalesinde tanıtılmıştır
- Bu cümledeki kelimeler gibi sıralı verilerdeki ilişkileri izleyerek bağlamı ve dolayısıyla anlamı öğrenen bir sinir ağıdır
- Bir dizide birbirini etkileyen, birbirine bağlı olan ve birbirinden uzak veri öğelerinin ince bağlantılarını tespit etmek için, ilgi (attention) veya öz-ilgi (self-attention) adı verilen bir dizi matematiksel teknik kullanır
- Birçok kullanım alanında evrişimli ve tekrarlayan sinir ağlarının (CNN, RNN ve LSTM'lerin) yerini almıştır
- Geleneksel RNN ve LSTM aksine, metin içindeki uzun mesafeli bağlantıları daha etkin bir şekilde öğrenir
- Kullanıcılar için maliyetli ve zaman alıcı olan büyük etiketli veri kümeleriyle sinir ağlarını eğitmek yerine, trilyonlarca görüntü ve petabytelarca metin verisini web ve kurumsal veritabanlarında kullanılabilir hale getirip, tokenlar arasındaki örüntüleri matematiksel bir yaklaşım ile tespit ederler
- Kullandıkları matematiksel hesaplamalar paralel işlemeye elverişli olduğundan hızlı çalışırlar
- Bir kodlayıcı-kod çözücü (encoder-decoder) yapısını kullanır. Girdiyi kodlar ve bir çıktı tahmini üretmek için kodu çözer
- Multi-head self-attention mekanizması, modelin belirli bir token için tahminde bulunurken girdideki farklı tokenların önemini de tartmasına olanak tanır

TRANSFORMER MİMARİSİ

ENCODER

DECODER

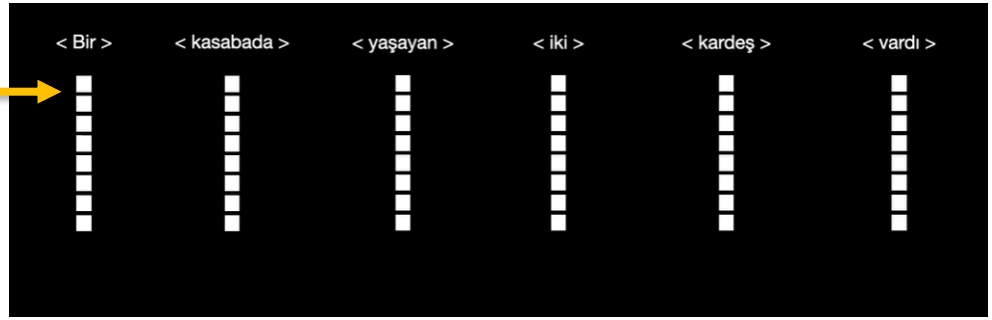


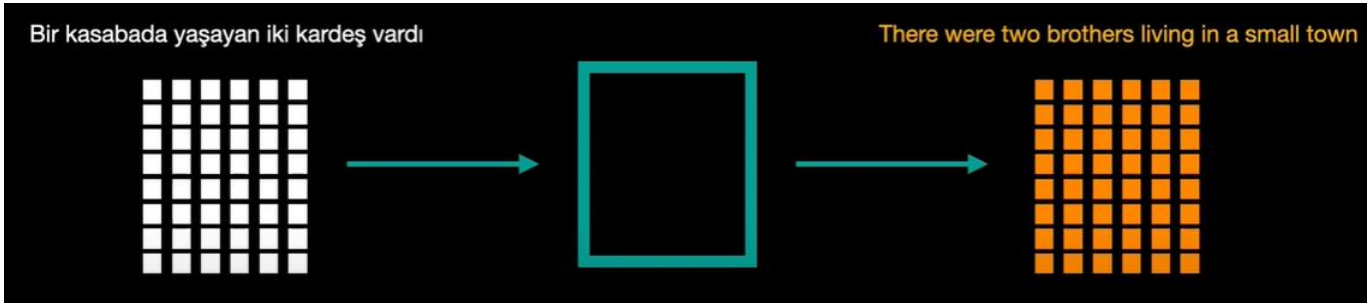
Transformerların anlatımına çeviri problemi örneği ile başlayalım.

TR Bir kasabada yaşayan iki kardeş vardı. Onların adları Ahmet ve Mehmet'ti. İkisi de babalarının lokantasında çalışıyorlardı. Ahmet, babasının en büyük oğluydu ve lokantanın başında duruyordu. Mehmet ise, küçük kardeşti ve mutfağa bakıyordu. İkisi de hayatlarının geri kalanını babalarının lokantasında geçirmeye karar vermişti. Ama bir gün, kasabaya yeni bir kadın gelir ve her şeyi değiştirir. Kadın, her erkeğin hayallerini süsleyen bir güzelliğe sahipti ve Ahmet ile Mehmet'in hayatlarını sonsuza dek değiştirirdi.

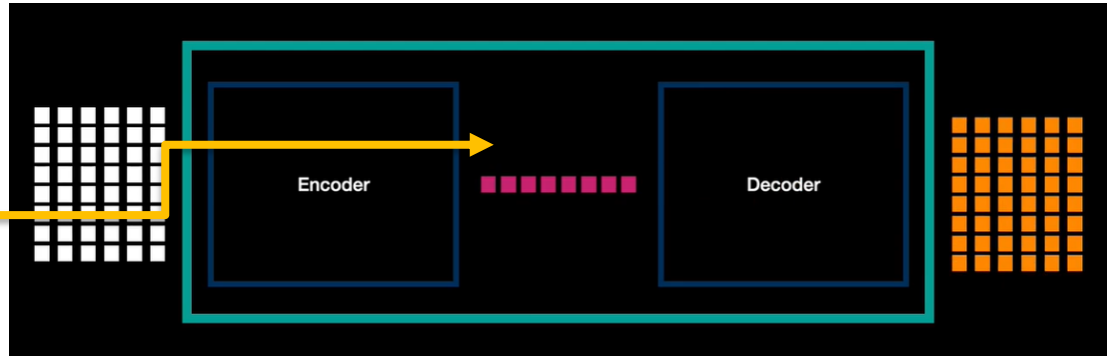
EN There were two brothers living in a small town. Their names were Ahmet and Mehmet. Both of them worked at their father's restaurant. Ahmet was the eldest son and he managed the restaurant while Mehmet looked after the kitchen. They had decided to spend the rest of their lives at their father's restaurant. But one day, a new woman came to town and changed everything. The woman had a beauty that every man dreamt of, and she would change Ahmet and Mehmet's lives forever

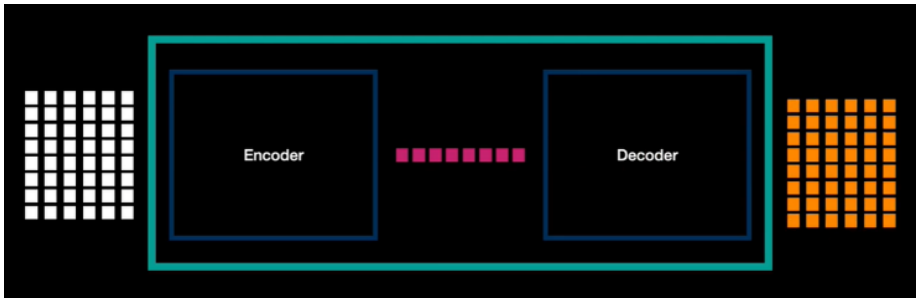
Kelime vektörleri



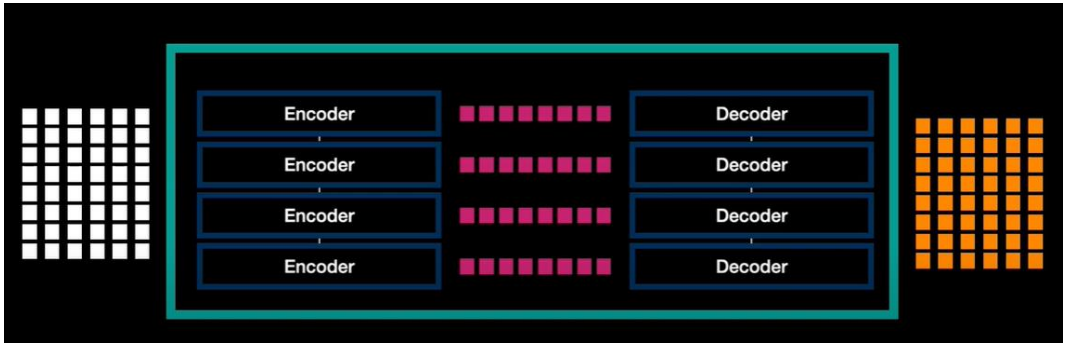


Sıkıştırılmış kodlayıcı vektör

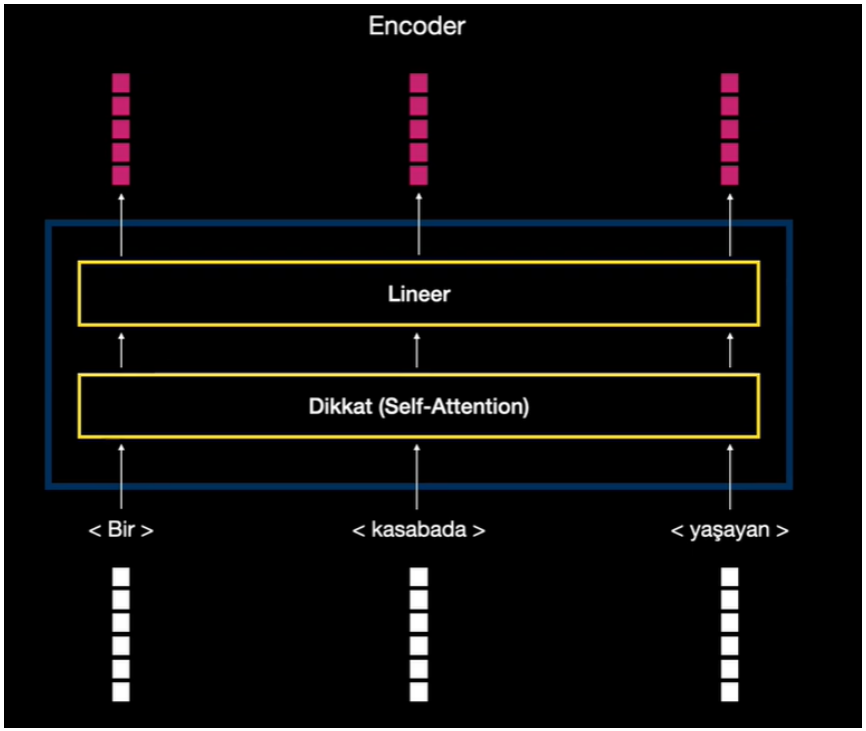




Birden fazla kodlayıcı ve kod çözücüyü zincirleme bağlayarak kullanırsak daha başarılı sonuçlar alabiliriz

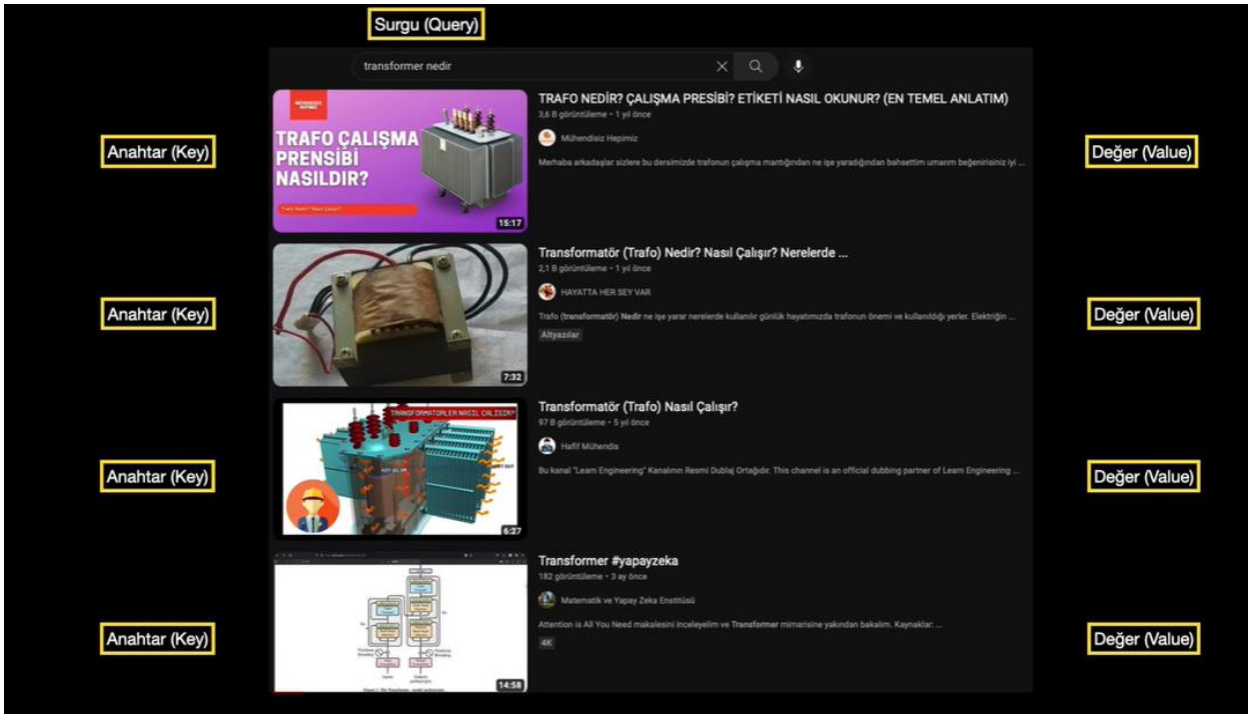


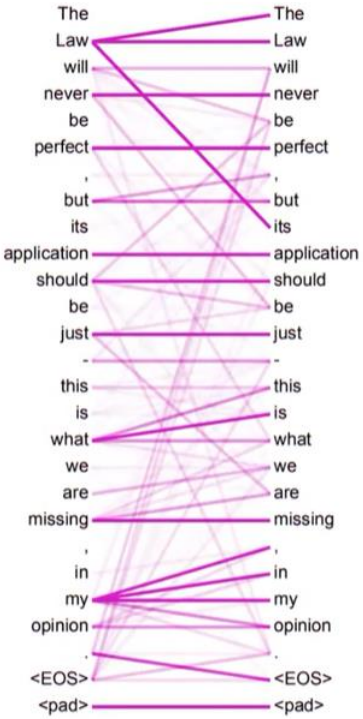
ENCODER-KODLAYICI



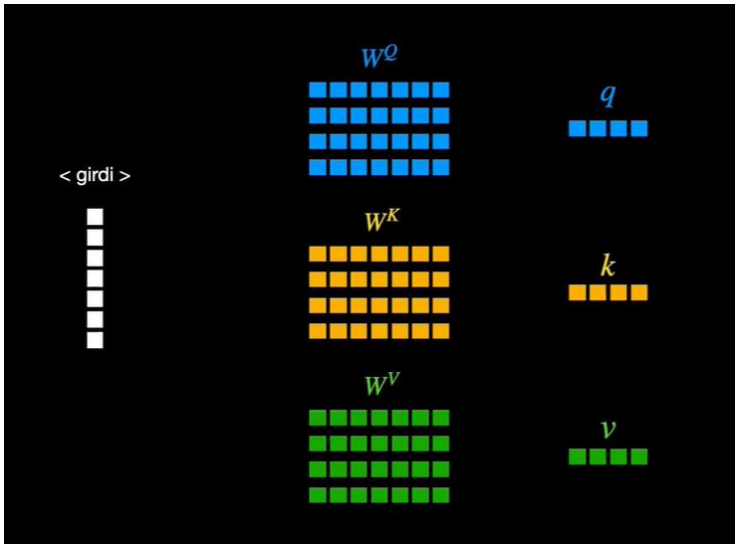
- Encoder, girdi metni alır ve her kelime/token için bir temsil oluşturur. Bu temsiller, metnin anlamını ve kelime ilişkilerini içerir. Encoder, metnin anlamını vektörler halinde kodlar, bu da decoder tarafından kullanılır.
- Attention mekanizması, encoder ve decoder içinde bulunur. Modelin metnin hangi kısımlarına "**dikkat etmesi**" gerektiğini belirler. Bu sayede, özellikle çeviri gibi görevlerde, model metnin belirli kısımları arasındaki bağlantıları daha etkili bir şekilde kurar.
- Transformerler bağlam bilgisini sınırsız bir şekilde taşıyabilir ve paralel olarak çalışırlar.

Self-attention mekanizmasını bir analoji ile anlatacak olursak

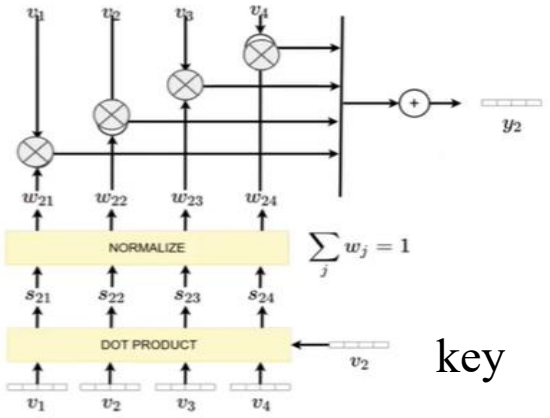




Self-attention mekanizmasının görevi bizim aradığımız videoya daha fazla **dikkat** değeri atamaktır

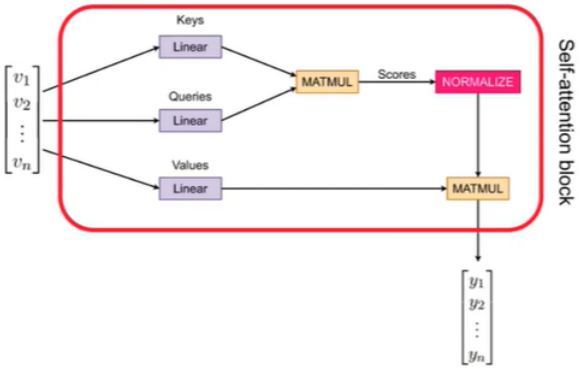


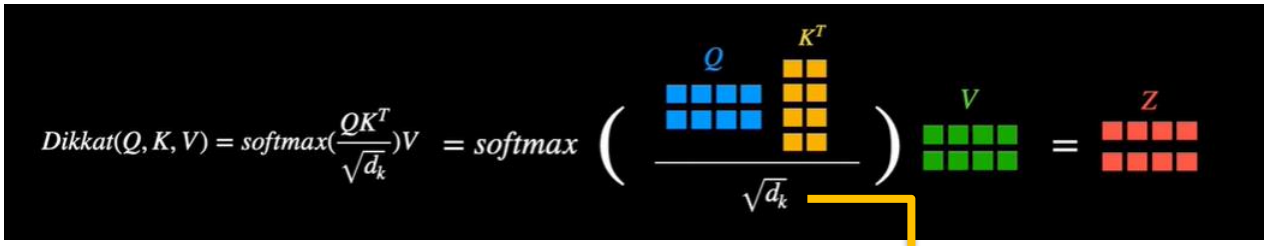
value



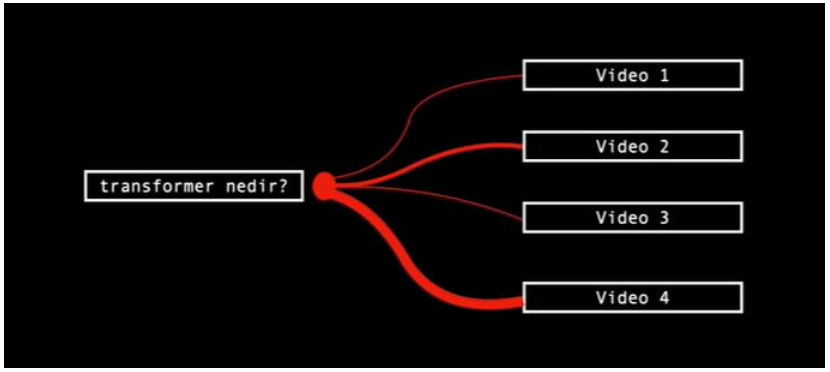
key

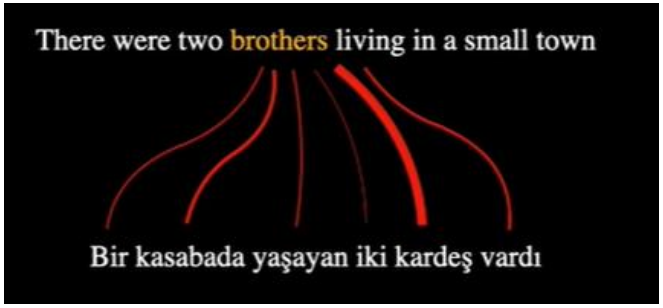
query





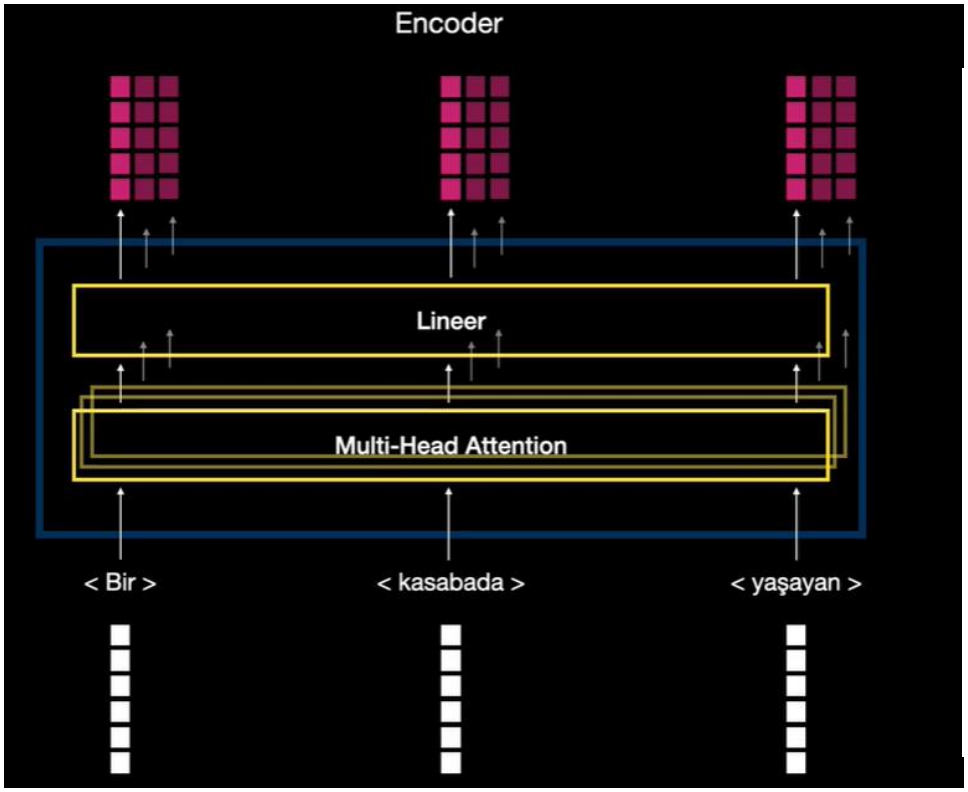
Normalize ediliyor. **d** vektörün boyutu

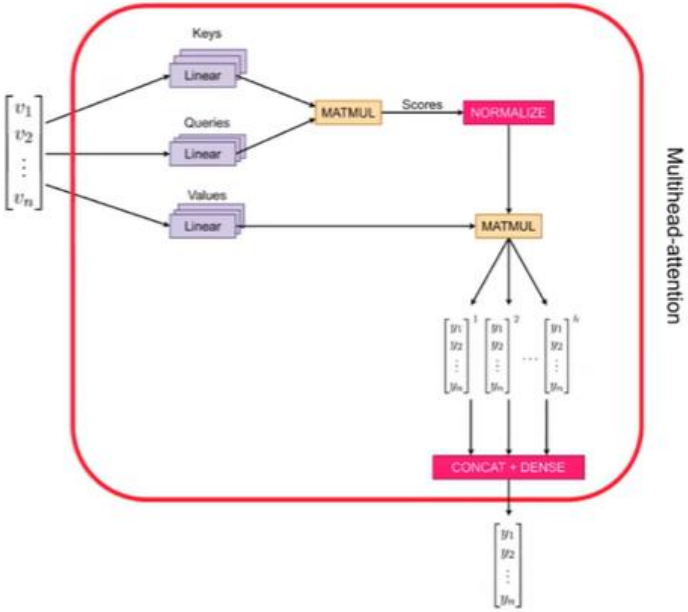




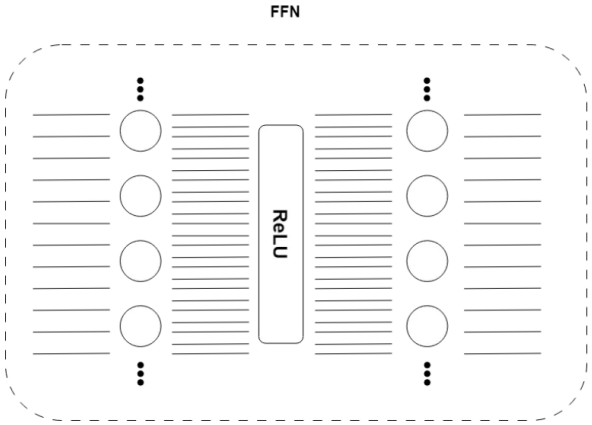
Cümle üzerinde örnek

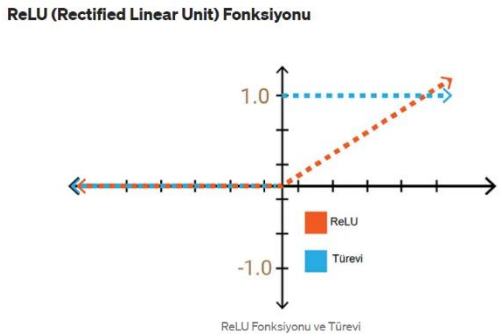
Multi-Head Attention Yapısı





Feed Forward Network

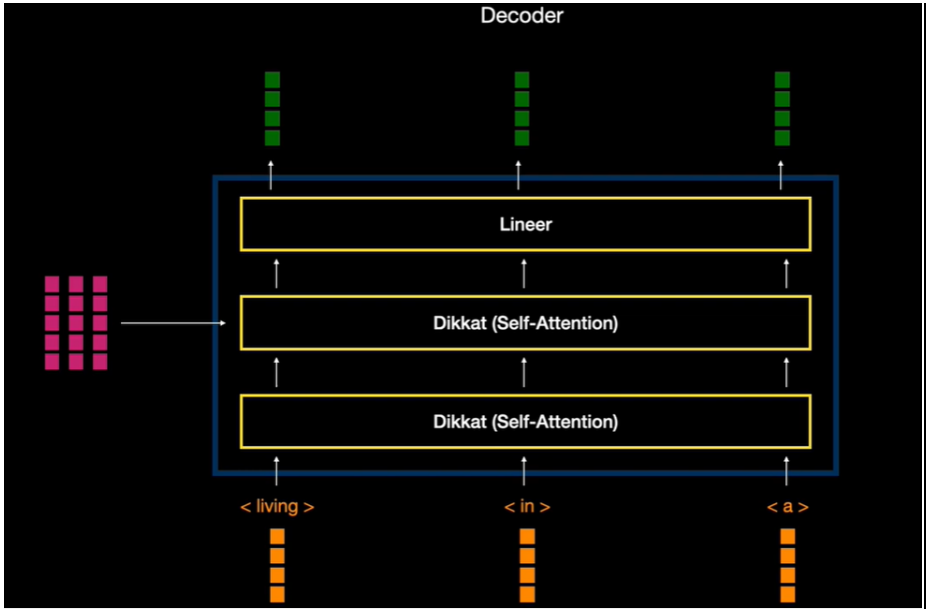


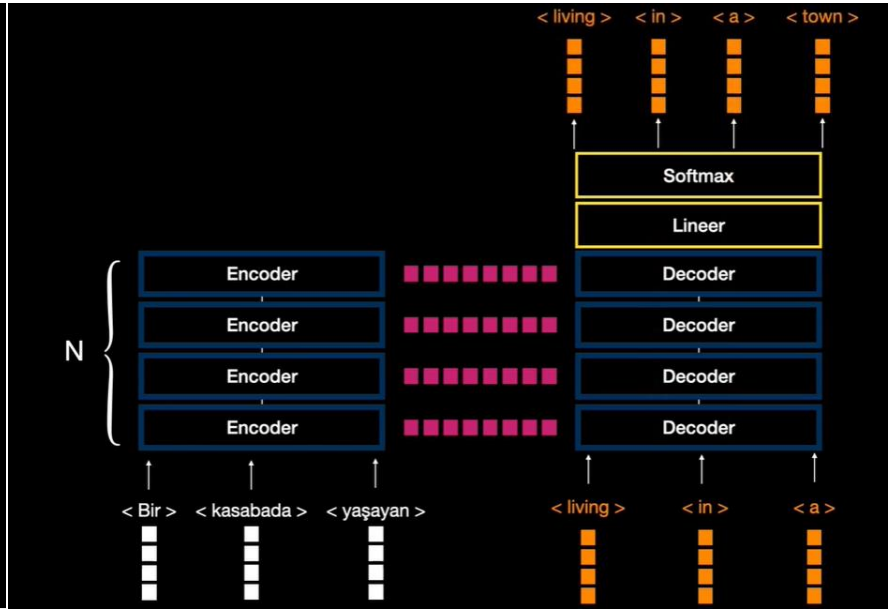


ReLU Fonksiyonu ve Türevi

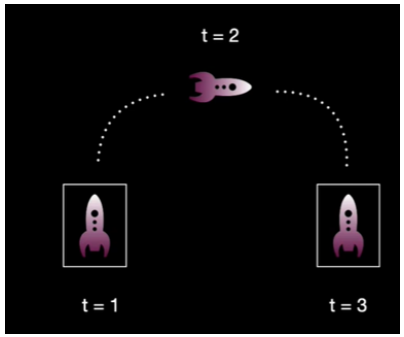
- İki normal yapay sinir ağı ve arasında bulunan *ReLU* aktivasyon fonksiyonundan oluşur
- Girdi olarak *Multi Head Attention* çıktısını alır (makalede 512 boyutunda bir dizi) ve aynı boyutta bir çıktı üretir
- ***FFN***' ler hakkında en önemli ayrıntı **Forward Expansion** kavramıdır
- ***FFN*** girdiyi alır, kendi içinde boyut yükseltmesi yapar (512*4=2048 lik bir çıktı üretir)
- Bu çıktı, ReLU aktivasyon fonksiyonundan geçer
- İkinci ağ, 2048 boyutunda ki girdiyi alır ve 2048/4=512 boyutunda bir çıktı üretir
- Bu yöntem ile girdi ve çıktı boyutu sabit tutulurken *Self-Attention* ile öğrenilen özelliklerin daha geniş parametrelere yayılarak öğrenilmesi sağlanır

DECODER-KOD ÇÖZÜCÜ

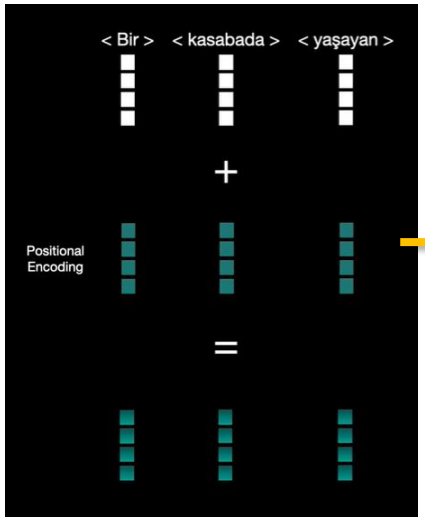




Positional Encoding Kavramı



❖Görüntü işleme yapacak olsak her kareyi bağımsız işleyecek olursak, iniş ve kalkış durumunu gösteren t=1 ve t=3 karesi birbiri ile karışır

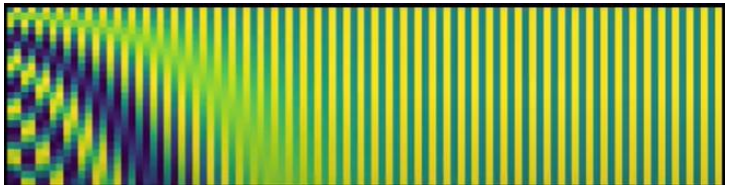


zaman bilgisi

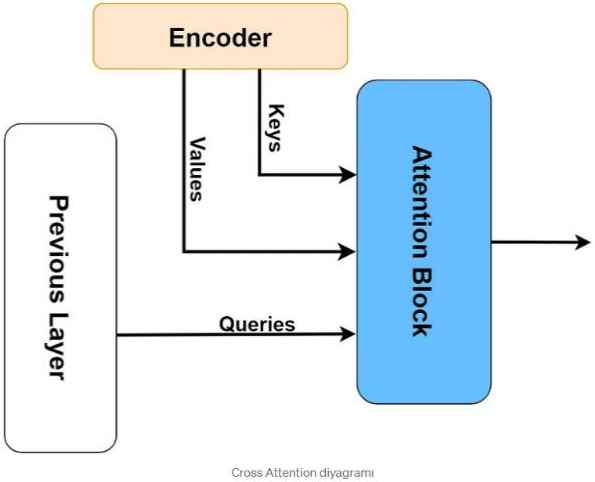
$$PE_{cift} = sin(pozisyon/10000^{2i/d_{model}})$$

$$PE_{tek} = cos(pozisyon/10000^{2i/d_{model}})$$

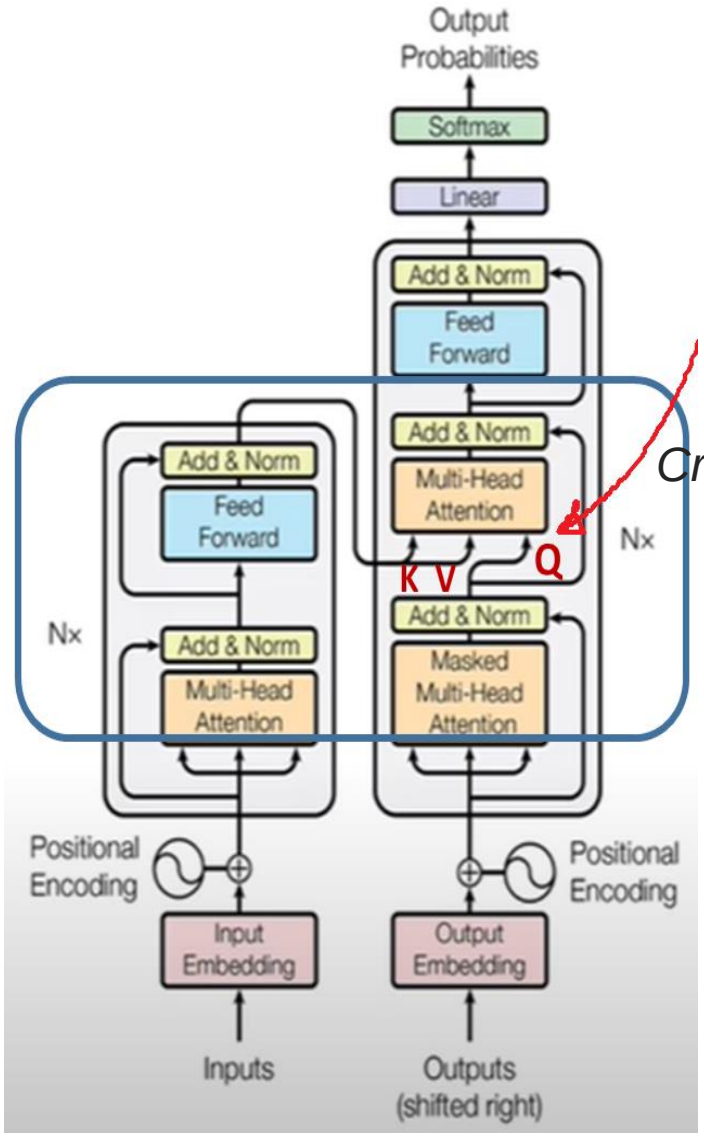
- Transformer makalesinde sinüzoidal kodlama olarak geçen bir teknik kullanır



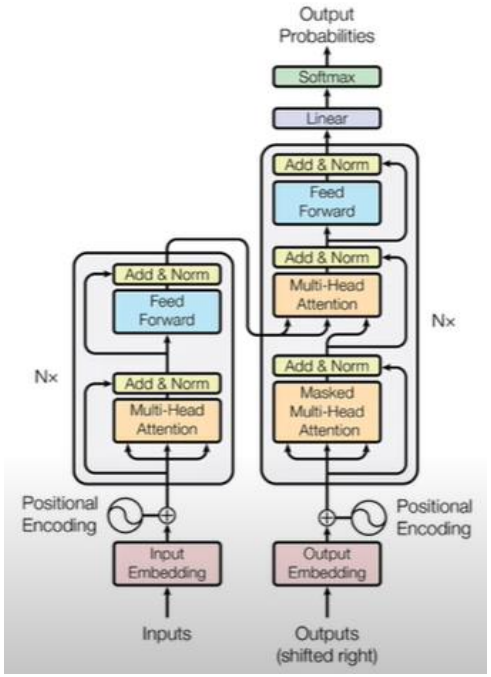
- Her satır değeri birbirinden farklı ve her token için indeks bilgisi tutulmuş oluyor



Cross Attention diyagramı



Cross Attention



- ✓ **Multi-Head Attention**, dikkat mekanizmasının aynı anda birden fazla kez paralel olarak uygulanması anlamına gelir; her uygulama farklı öğrenilmiş girişin doğrusal dönüşümlerini kullanır. Multi-Head yaklaşım, verideki farklı ilişki türlerini yakalamak için modele izin verir.
- ✓ **Feed Forward** ağlar, transformer modelinin kendi dikkat mekanizması tarafından üretilen temsilleri dönüştürür. Bu dönüşüm, modelin yalnızca dikkat mekanizmasıyla yakalanamayan verideki daha karmaşık ilişkileri de öğrenmesini sağlar.
- ✓ **Add & Norm** bağlantıları, eğitim sürecini istikrarlı hale getirmeye yardımcı olur. Modelin her katmana girişleri verirken standartlaştırarak, aşırı değerlerden veya istikrarsız gradyanlardan etkilenme şansını azaltır.

- ✓ **Output** doğrusal dönüşüm ve ardından olası çıkış kelimeleri üzerinde bir olasılık dağılımı üreten softmax fonksiyonundan oluşur. En yüksek olasılığa sahip kelime, her pozisyonda çıkış kelimesi olarak seçilir.

Mask Multi-Head Attenion

- *Encoder* yapısında cümle bir bütün olarak sisteme verildiği için maskeleme yapılmaz
- *Decoder* ise girdileri bir bütün olarak değil, parça parça alır, her zaman solundaki kelime ile ilgilenir ve sağındaki kelimeyi tahmin etmeye çalışır. Bu sebeple sağdaki tokenler maskelenir

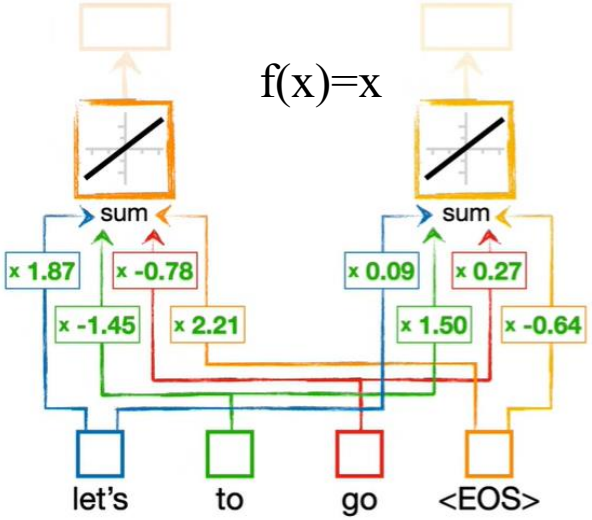
«Bugün hava güzel»

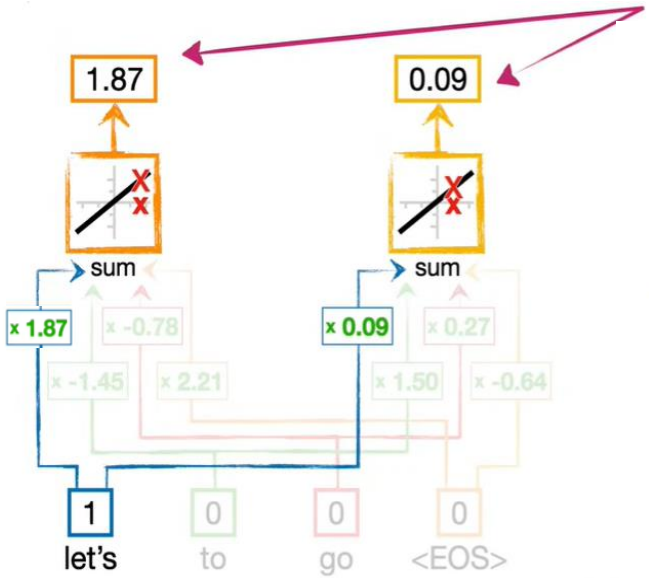
Örnek: *Decoder* girdi olarak "*bugün*" kelimesini alır ve "*hava*" kelimesini üretir, ardından bir sonraki kelimeyi üretmek için De*coder* "*bugün*" ve "hava" kelimesini girdi olarak alır ve çıktı olarak "*güzel*" kelimesini üretir

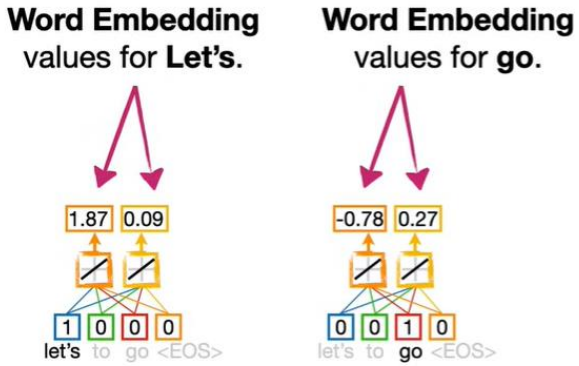
- Transformerlar her zaman encoder ve decoder yapısından oluşmaz.
- Transformer mimarisi, başlangıçta çift yönlü görevler için geliştirilmiştir. Encoder-decoder yapıları, özellikle dil çevirisi gibi belirli görevler için tasarlanmıştır.
- Bir transformer, metin sınıflandırma veya dil modelleri gibi tek yönlü görevler için sadece bir encoder kullanılabilir. Bu durumda, modelin sadece girdi metni üzerinde işlem yapması ve bir çıktı üretir.
- Bir transformer, metin oluşturma veya dil modelleriyle metin tamamlama gibi üretici görevler (generative tasks) için sadece decoder kısmını kullanır.

Word Embedding

Bu sayılar «let's» tokenını sembolize ediyor







Sinir ağları bağlamında her ağırlık en uygun değerler bulunana kadar rastgele bir sayı ile başlar ve eğitim aşamasıda geri yayılım algoritması ile bu değerler optimize edilir.

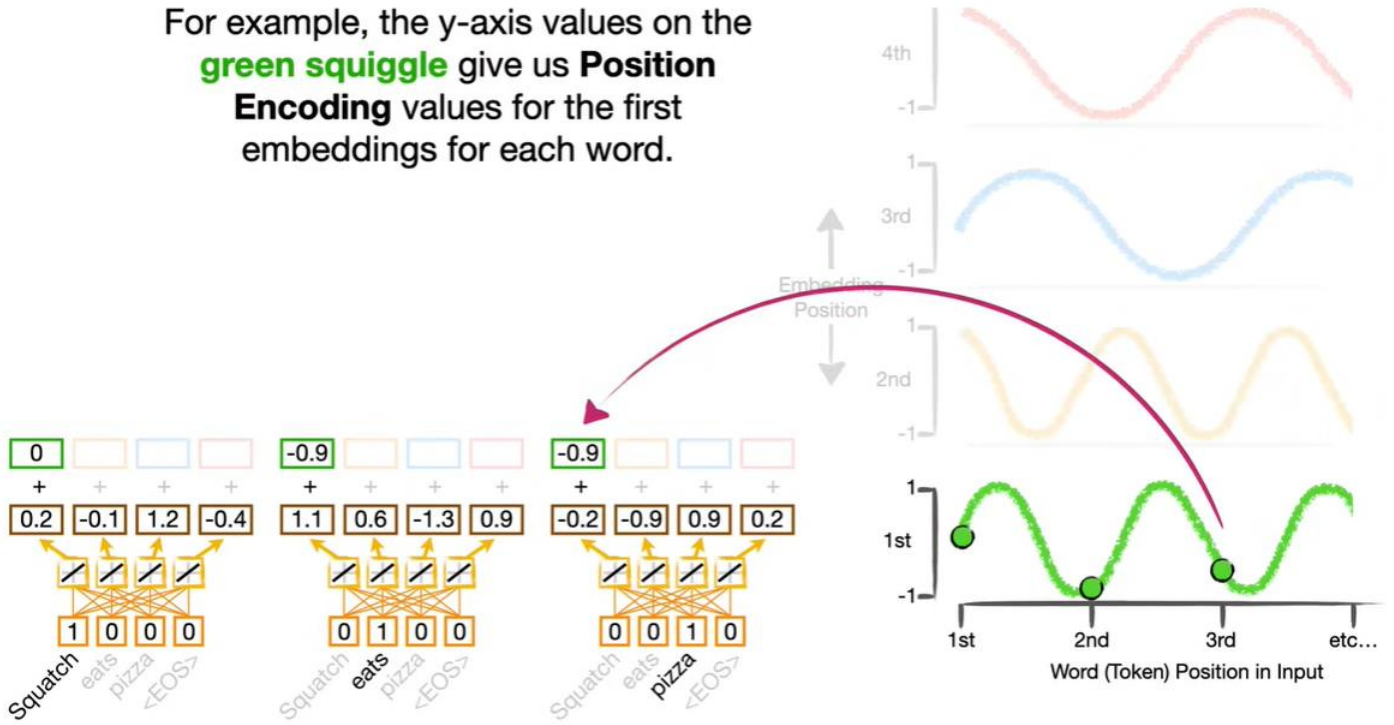
Positional Encoding

Anlamları farklı

'Squatch eats pizza!!!

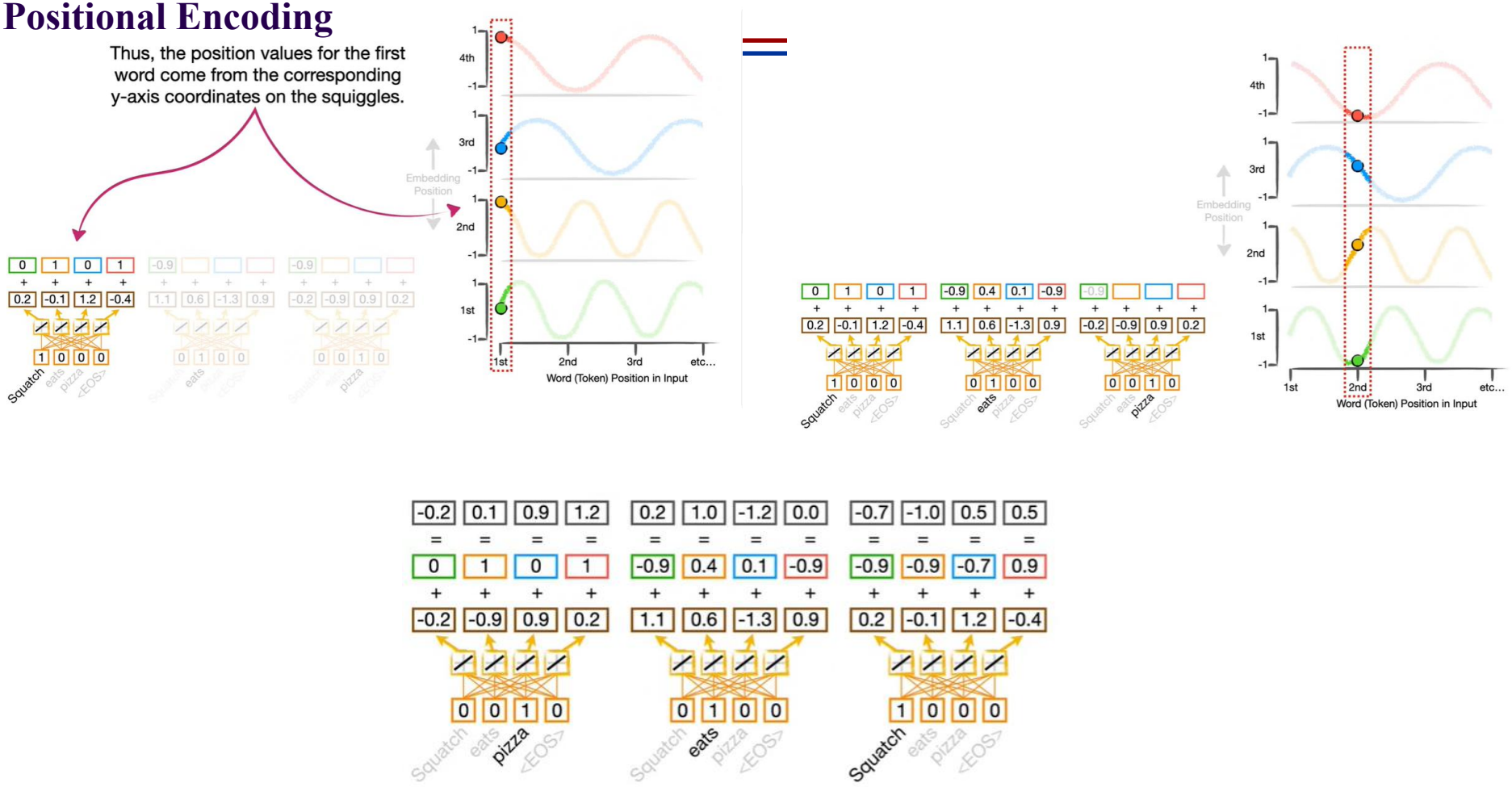
Pizza eats **'Squatch**!!!

For example, the y-axis values on the **green squiggle** give us **Position Encoding** values for the first embeddings for each word.

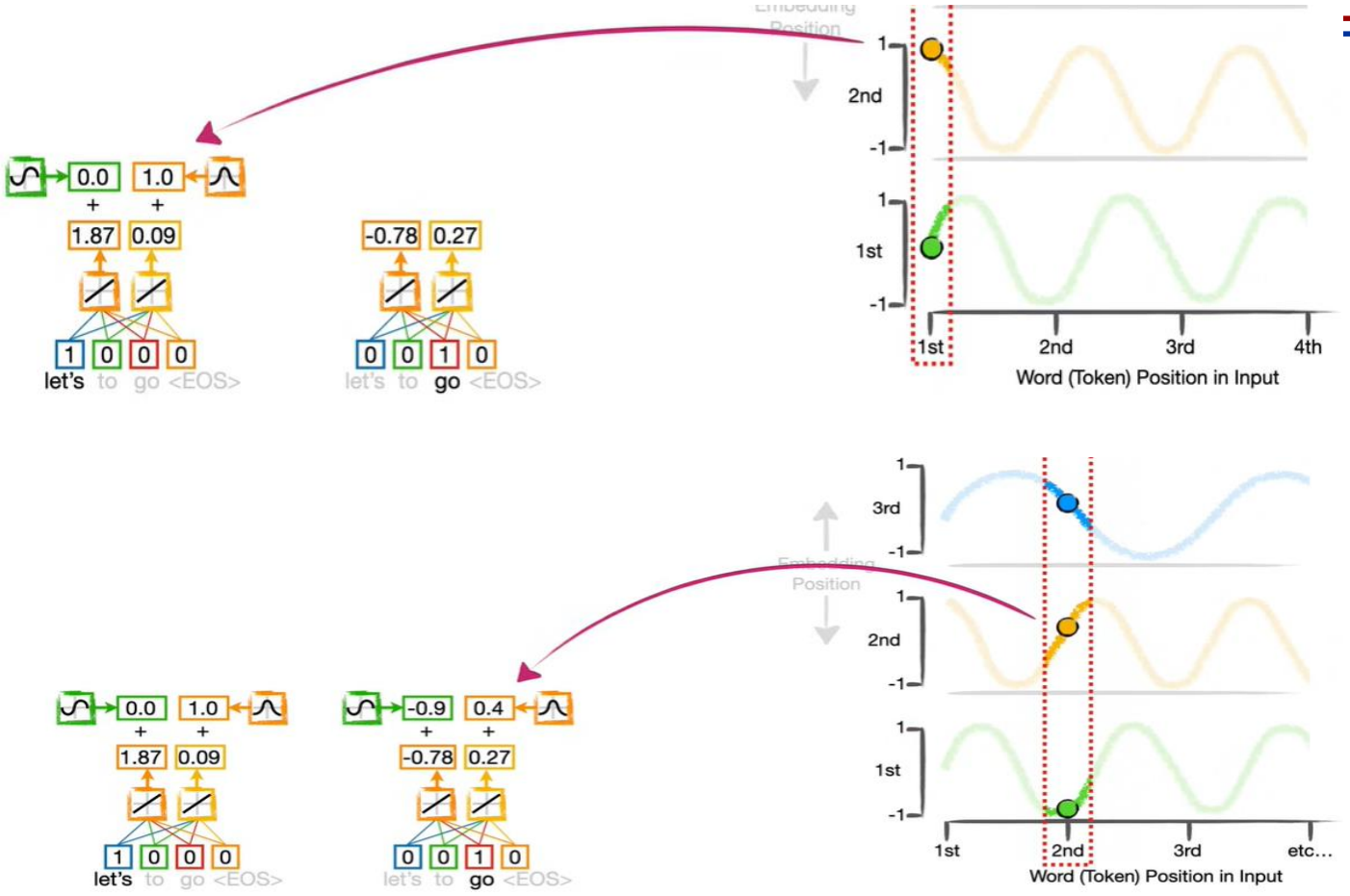


Positional Encoding

Thus, the position values for the first word come from the corresponding y-axis coordinates on the squiggles.

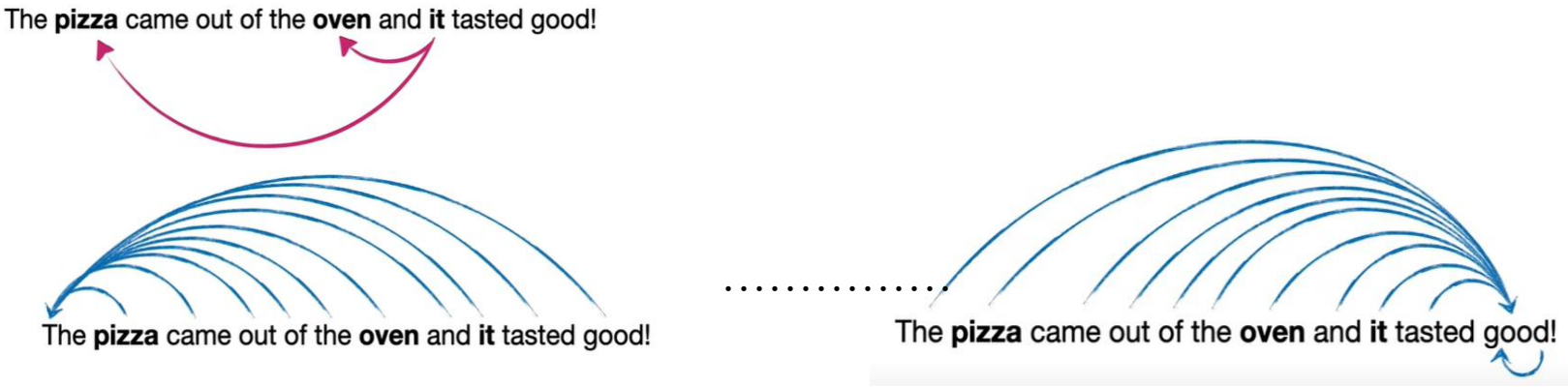


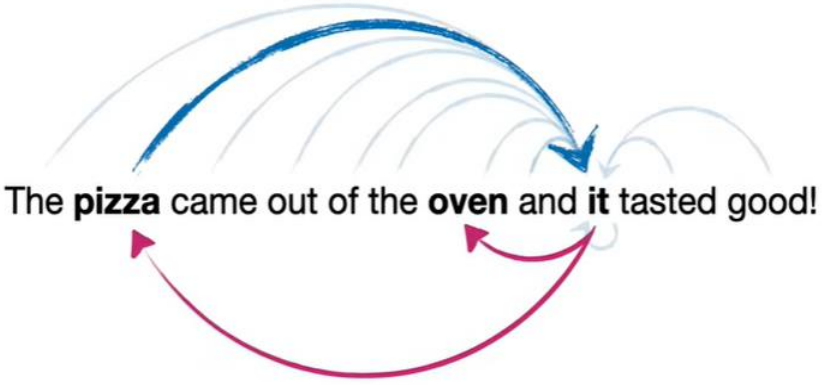
Let's go örneğimize dönelim…



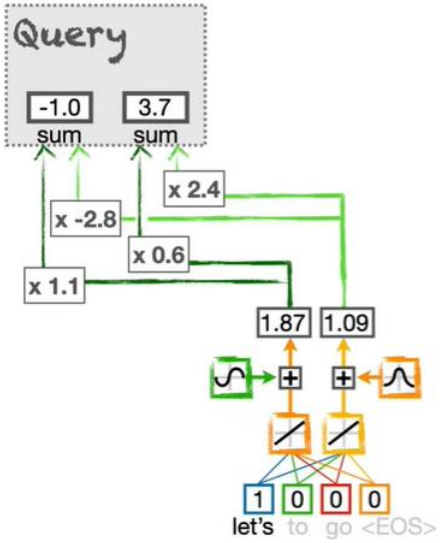
Self Attention-Öz Dikkat

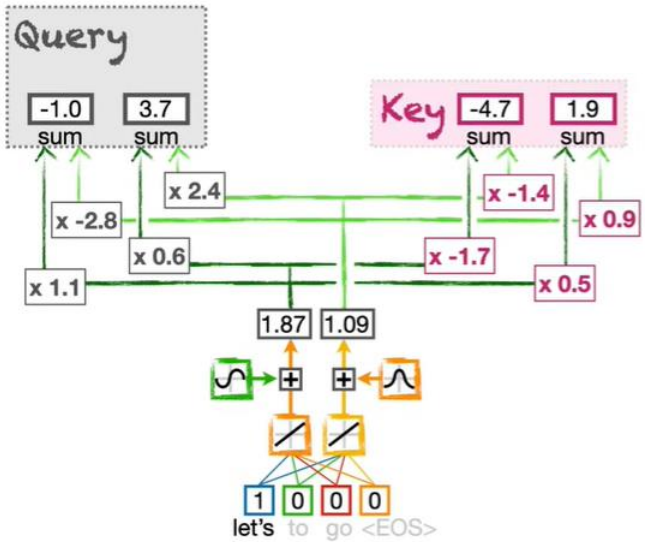
- Dikkat Mekanizması neyi bulmaya çalışıyor ?
  - Öz-dikkat, her bir kelimenin kendisi de dahil olmak üzere cümledeki tüm kelimelere ne kadar benzediğini görerek çalışır.

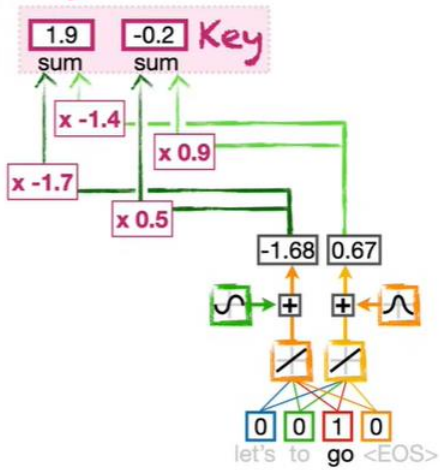


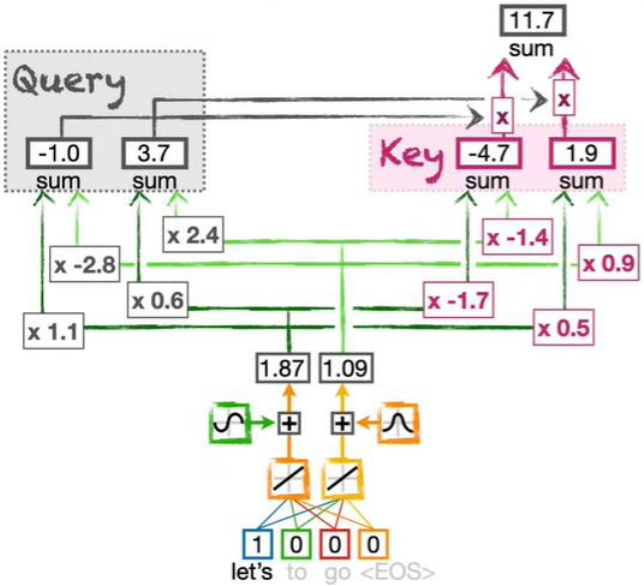


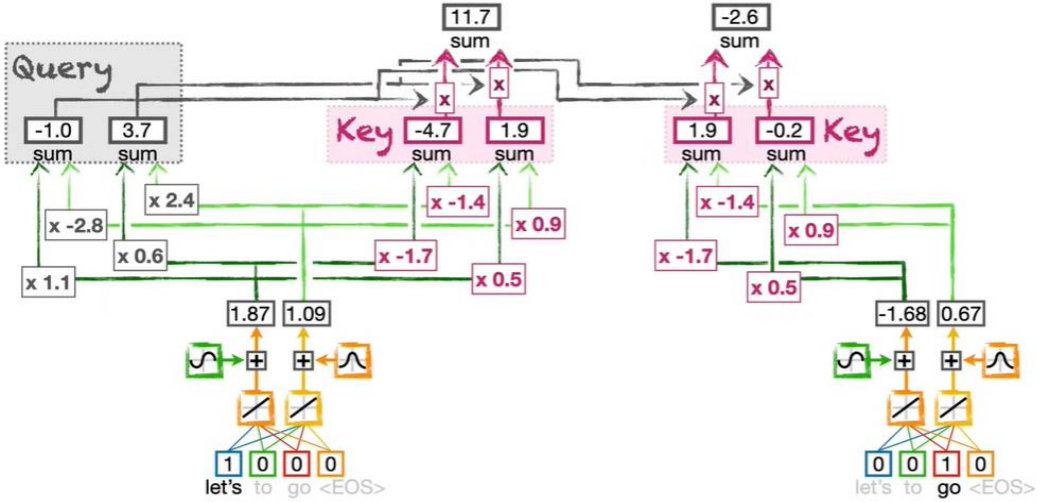
- Öz Dikkat mekanizması «pizza» ile «it» arasındaki ilişkiyi daha kuvvetli görür


Query
-1.0
3.7
sum
sum
x 2.4
x -2.8
x 0.6
x 1.1
1.87
1.09
1
0
0
0
let's
to
go
<EOS>

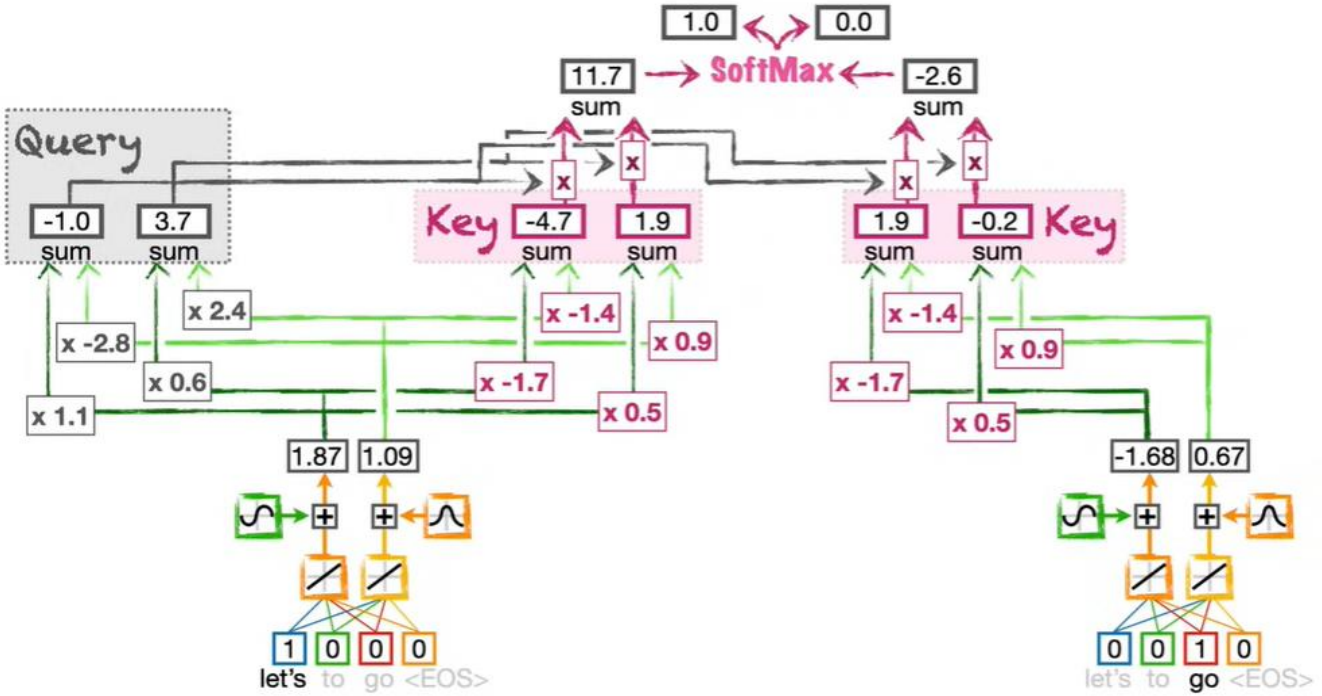

Query
-1.0
3.7
sum
sum
Key
-4.7
1.9
sum
sum
x 2.4
x -2.8
x 0.6
x 1.1
x -1.4
x 0.9
x -1.7
x 0.5
1.87
1.09
1
0
0
0
let's
to
go
<EOS>

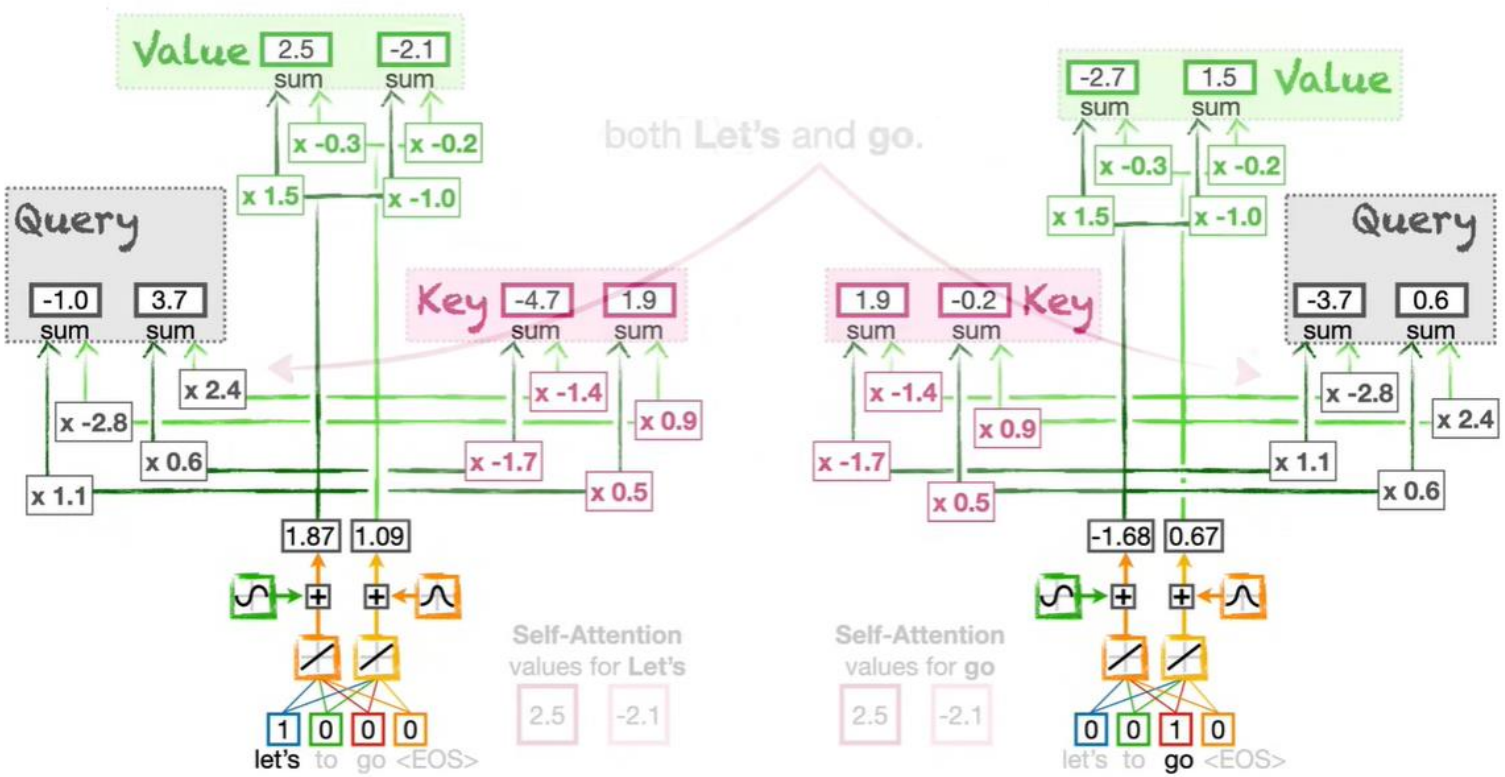

1.9
-0.2
Key
sum
sum
x -1.4
x 0.9
x -1.7
x 0.5
-1.68
0.67
0
0
1
0
let's
to
go
<EOS>

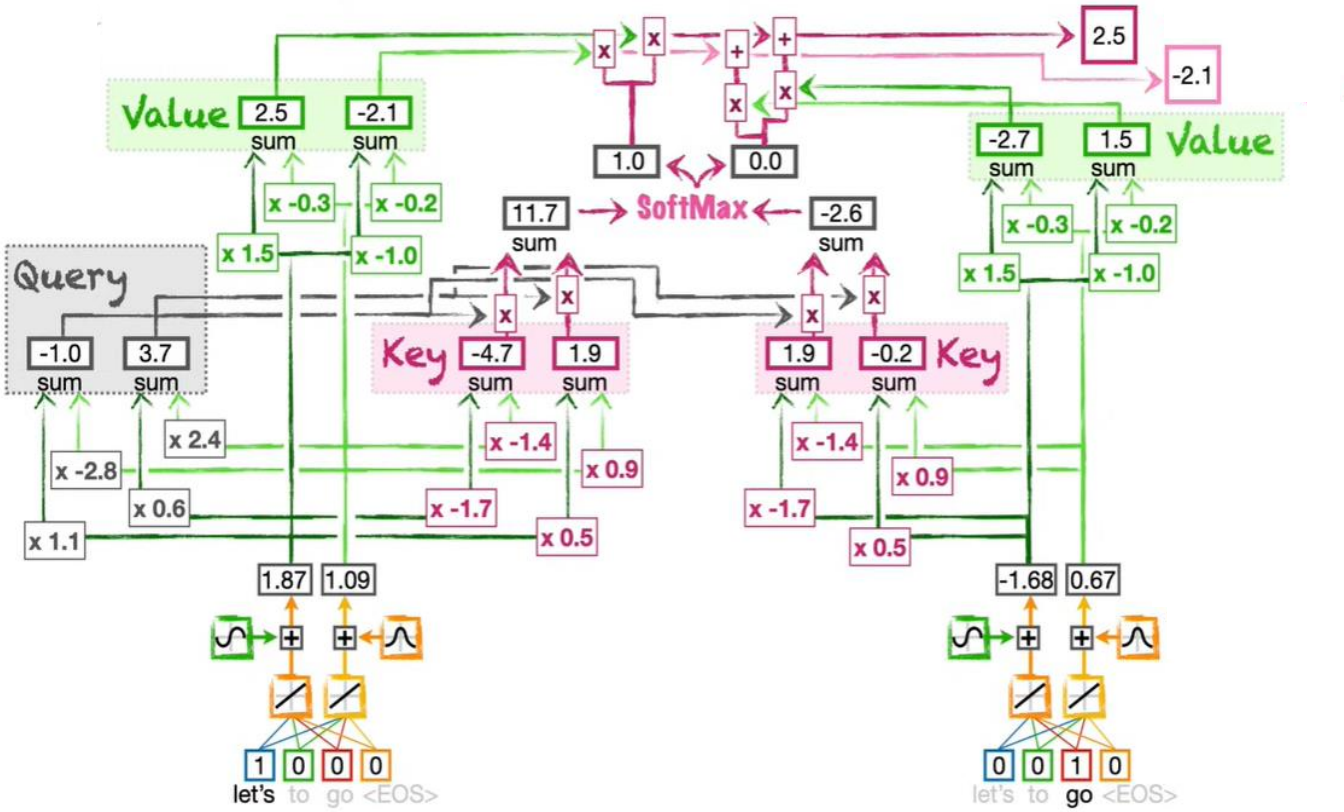

11.7
sum
Query
-1.0
sum
3.7
sum
Key
-4.7
sum
1.9
sum
x 2.4
x -1.4
x -2.8
x 0.9
x 0.6
x -1.7
x 1.1
x 0.5
1.87
1.09
1
0
0
0
let's
to
go
<EOS>

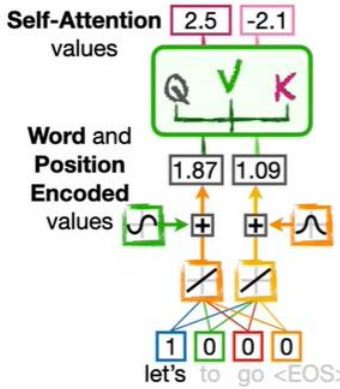

11.7
sum
-2.6
sum
Query
-1.0
sum
3.7
sum
Key
-4.7
sum
1.9
sum
1.9
sum
-0.2
sum
Key
x 2.4
x -1.4
x -2.8
x 0.9
x 0.6
x -1.7
x -1.4
x 0.9
x 1.1
x 0.5
x -1.7
x 0.5
1.87
1.09
-1.68
0.67
1
0
0
0
0
0
1
0
let's
to
go
<EOS>
let's
to
go
<EOS>

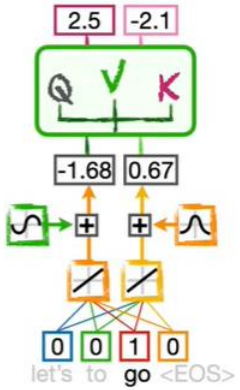
- Softmax değerleri sıralar 11.7, -2.6 ve 0 ile 1 arasına getirir

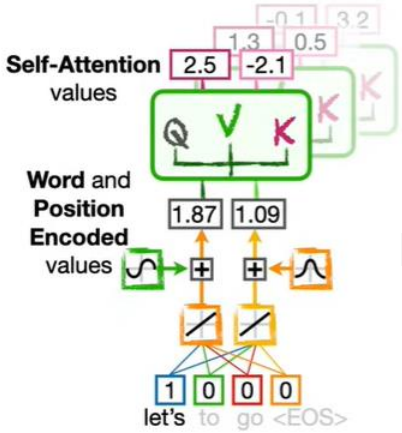



both Let's and go.
Value
2.5
-2.1
sum
sum
x -0.3
x -0.2
x 1.5
x -1.0
Query
-1.0
3.7
sum
sum
Key
-4.7
1.9
sum
sum
x 2.4
x -2.8
x -1.4
x 0.9
x 0.6
x -1.7
x 1.1
x 0.5
1.87
1.09
Self-Attention
values for Let's
2.5
-2.1
1
0
0
0
let's
to
go
<EOS>
-2.7
1.5
sum
sum
Value
x -0.3
x -0.2
x 1.5
x -1.0
Query
1.9
-0.2
sum
sum
Key
-3.7
0.6
sum
sum
x -1.4
x -2.8
x 0.9
x 2.4
x -1.7
x 1.1
x 0.5
x 0.6
-1.68
0.67
Self-Attention
values for go
2.5
-2.1
0
0
1
0
let's
to
go
<EOS>

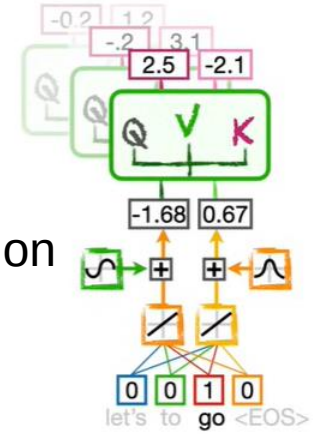

Value
2.5
-2.1
sum
sum
x -0.3
x -0.2
x 1.5
x -1.0
Query
-1.0
3.7
sum
sum
x 2.4
x -2.8
x 0.6
x 1.1
Key
-4.7
1.9
sum
sum
x -1.4
x 0.9
x -1.7
x 0.5
11.7
sum
SoftMax
-2.6
sum
1.0
0.0
2.5
-2.1
1.87
1.09
1
0
0
0
let's
to
go
<EOS>
Key
1.9
-0.2
sum
sum
x -1.4
x 0.9
x -1.7
x 0.5
Value
-2.7
1.5
sum
sum
x -0.3
x -0.2
x 1.5
x -1.0
-1.68
0.67
0
0
1
0
let's
to
go
<EOS>



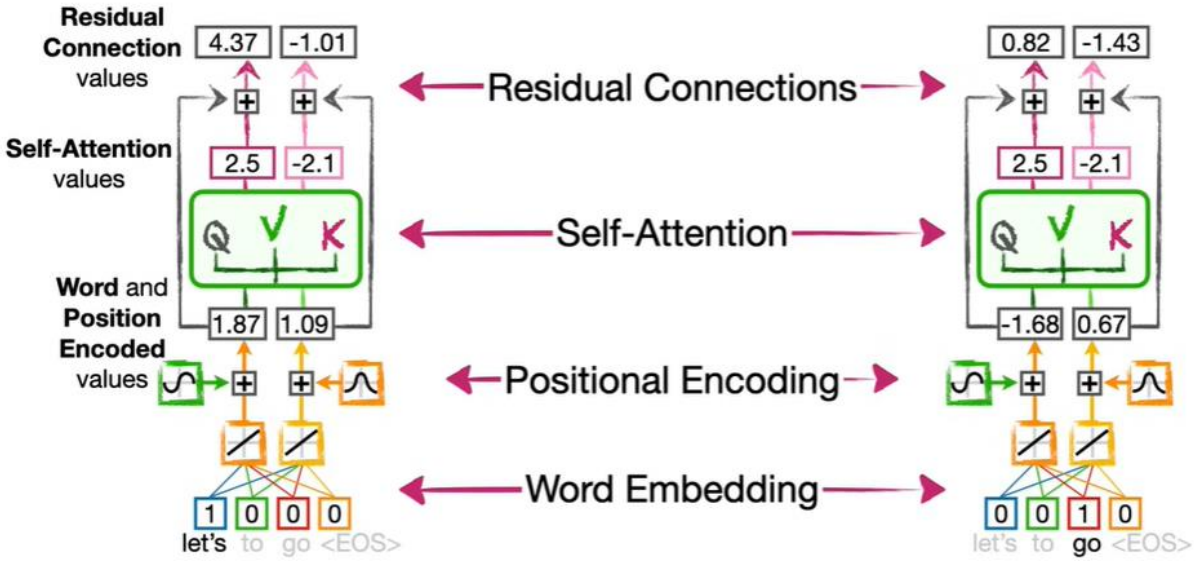


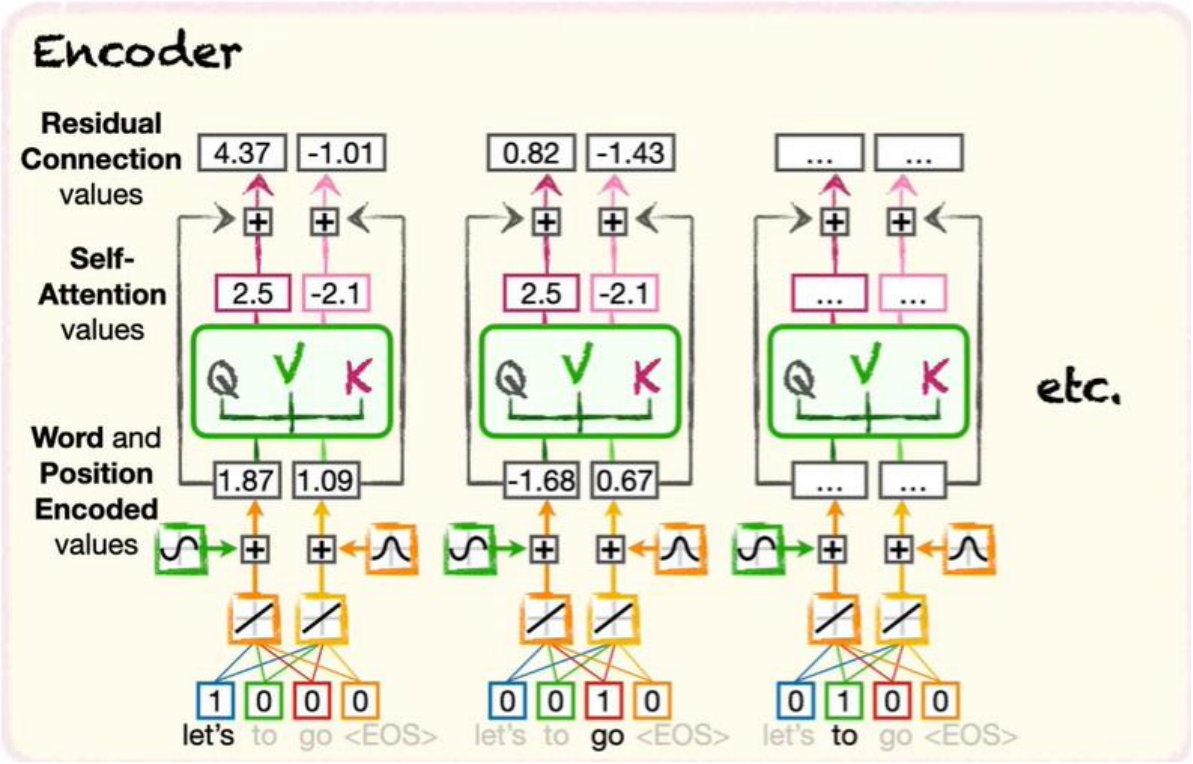


Multi-Head Attention



These **4** features allow the **Transformer** to: encode words into numbers, encode the positions of the words, encode the relationships among the words, and relatively easily and quickly train in parallel.

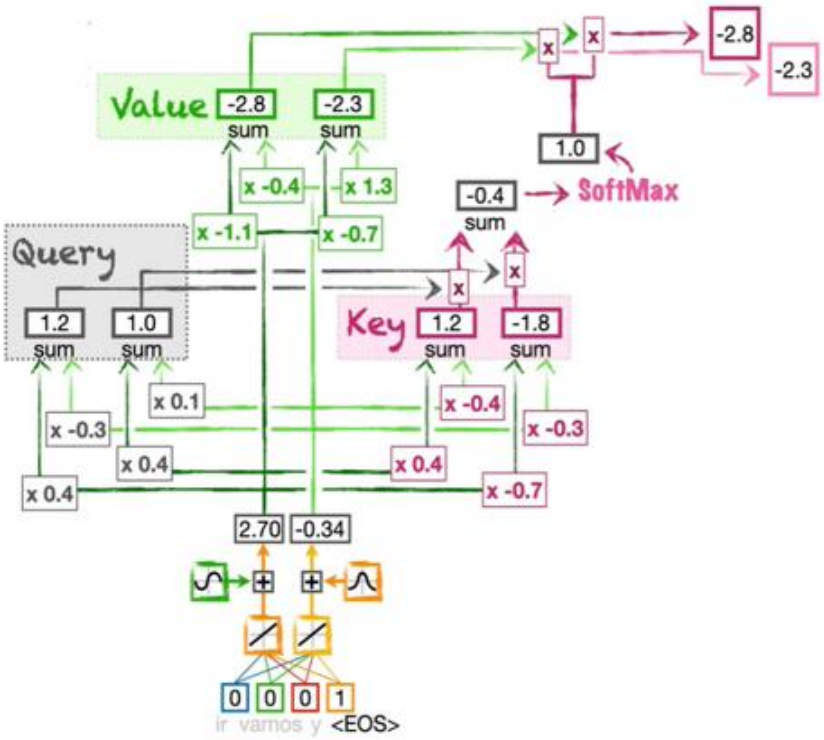




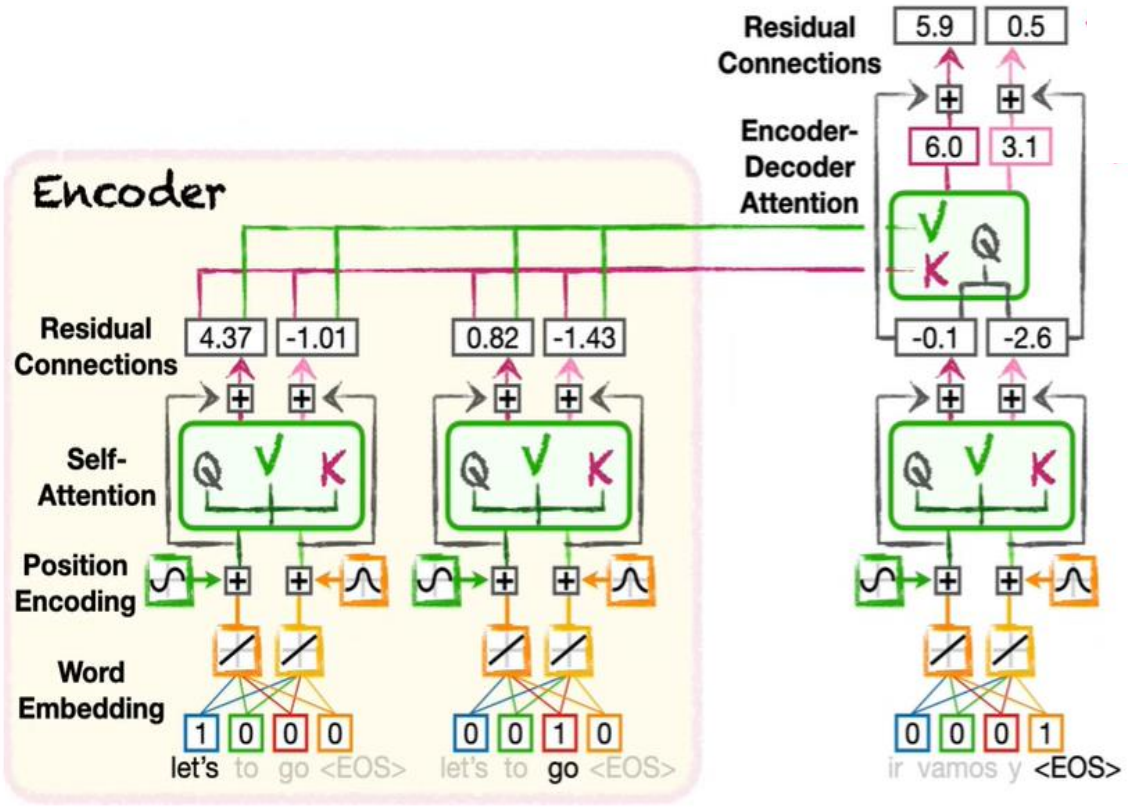
- Transformer, girişteki her kelimenin hesaplamasını aynı anda yapabilir (paralel olarak çalışır)

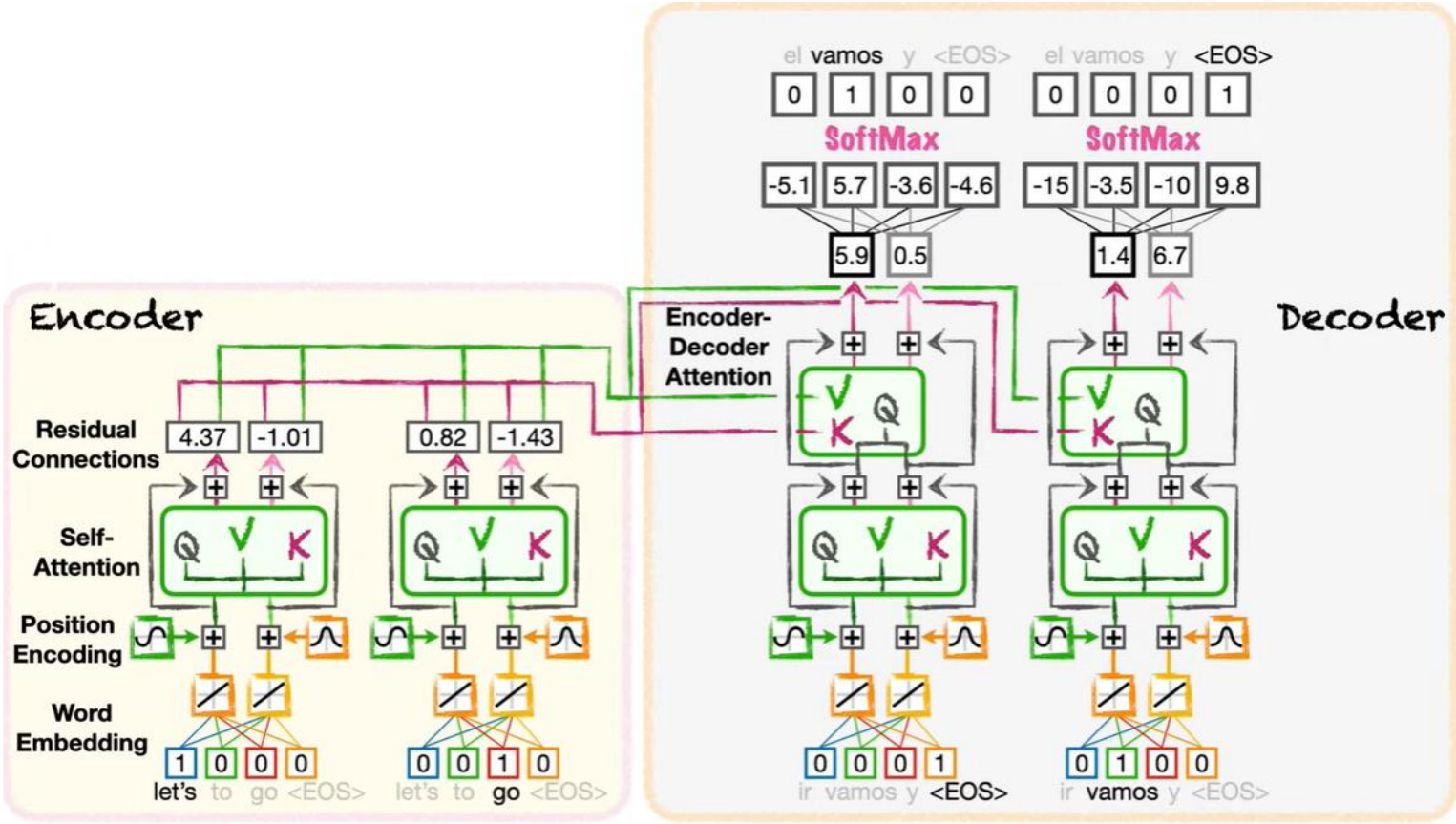
Decoder Word Embedding

- Kod çözme işlemini başlatmak için EOS (End of Sentence) kullanmak, kod çözme işlemini başlatmanın yaygın bir yoludur



Encoder-Decoder Attention




Encoder
Decoder
Encoder-Decoder Attention
Residual Connections
Self-Attention
Position Encoding
Word Embedding
4.37
-1.01
0.82
-1.43
let's to go <EOS>
let's to go <EOS>
el vamos y <EOS>
el vamos y <EOS>
0 1 0 0
0 0 0 1
SoftMax
SoftMax
-5.1 5.7 -3.6 -4.6
-15 -3.5 -10 9.8
5.9 0.5
1.4 6.7
0 0 0 1
ir vamos y <EOS>
0 1 0 0
ir vamos y <EOS>