ORTA DOĞU TEKNİK ÜNİVERSİTESİ
MIDDLE EAST TECHNICAL UNIVERSITY

CENG 463: Introduction to Natural Language Processing
Transformer and Tokenization

Asst. Prof. Çağrı Toraman
Computer Engineering Department
ctoraman@ceng.metu.edu.tr

25.11.2025

Transformer

Design neural network to encode and process text:

The restaurant refused to serve me a ham sandwich, because it only cooks vegetarian food. In the end, they just gave me two slices of bread. Their ambience was just as good as the food and service.

Transformer

Design neural network to encode and process text:

The restaurant refused to serve me a ham sandwich, because it only cooks vegetarian food. In the end, they just gave me two slices of bread. Their ambience was just as good as the food and service.

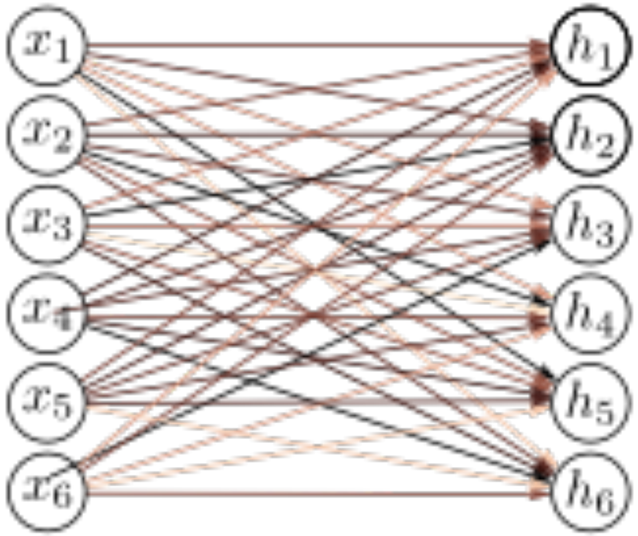
Transformer

Design neural network to encode and process text:

The restaurant refused to serve me a ham sandwich, because it only cooks vegetarian food. In the end, they just gave me two slices of bread. Their ambience was just as good as the food and service.

Transformer

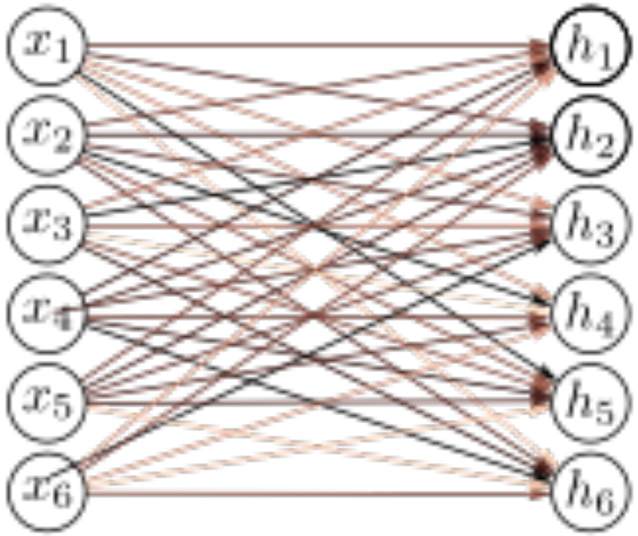
Standard fully-connected layer



$$\mathbf{h} = \mathbf{a}[\boldsymbol{\beta} + \boldsymbol{\Omega}\mathbf{x}]$$

Transformer

Standard fully-connected layer



D^2 connections

$$\mathbf{h} = \mathbf{a}[\boldsymbol{\beta} + \boldsymbol{\Omega}\mathbf{x}]$$

Transformer

Standard fully-connected layer

- A very large number of parameters
- Can’t cope with text of different lengths

Conclusion:

- We need a model where parameters don’t increase with input length

Transformer

Design neural network to encode and process text:

The restaurant refused to serve me a ham sandwich, because it only cooks vegetarian food. In the end, they just gave me two slices of bread. Their ambience was just as good as the food and service.

The word their must “attend to” the word restaurant.

Transformer

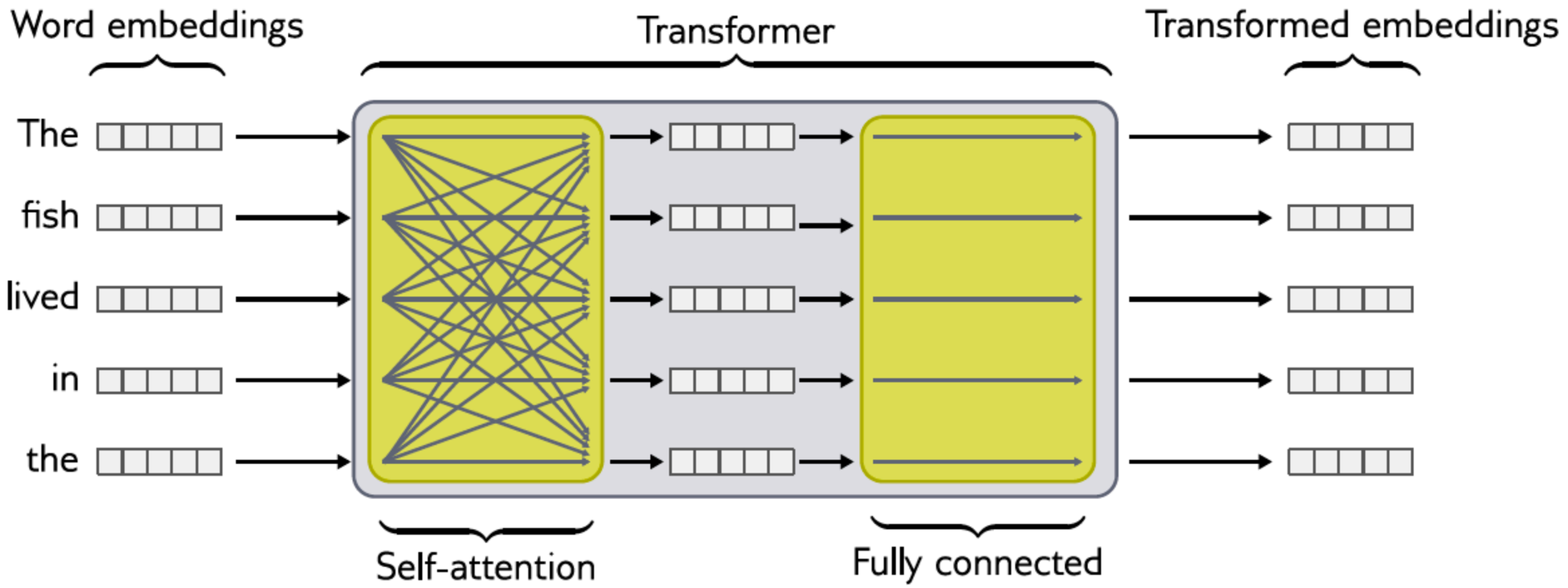
Design neural network to encode and process text:

The restaurant refused to serve me a ham sandwich, because it only cooks vegetarian food. In the end, they just gave me two slices of bread. Their ambience was just as good as the food and service.

The word their must “attend to” the word restaurant.

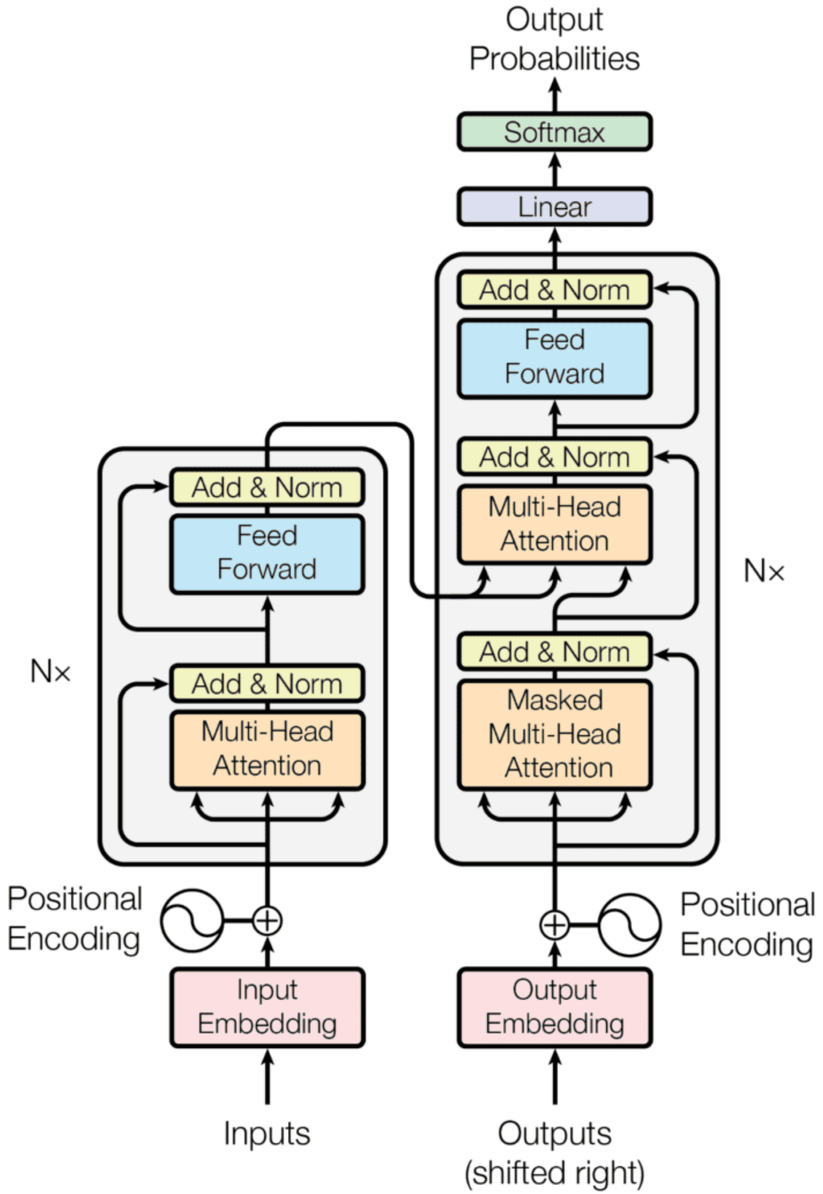
- There must be connections between the words.
- The strength of these connections will depend on the words themselves.

Transformer



Transformer

ENCODER



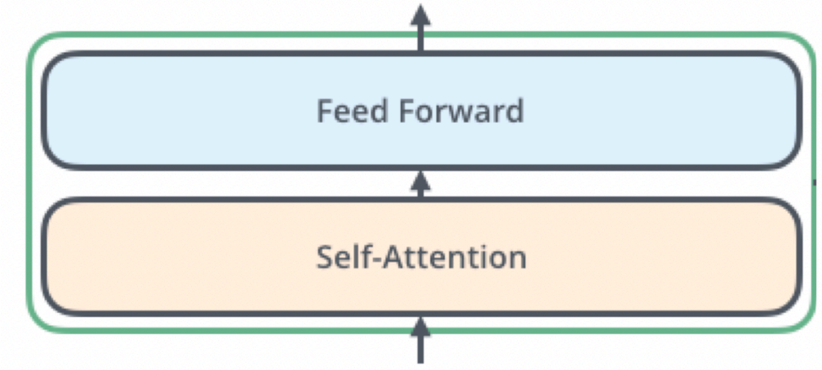
DECODER

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, NeurIPS 2017

Transformer

Build whole model out of self-attention

Uses only point-wise processing and attention (no recurrent units or convolutions)



A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, NeurIPS 2017

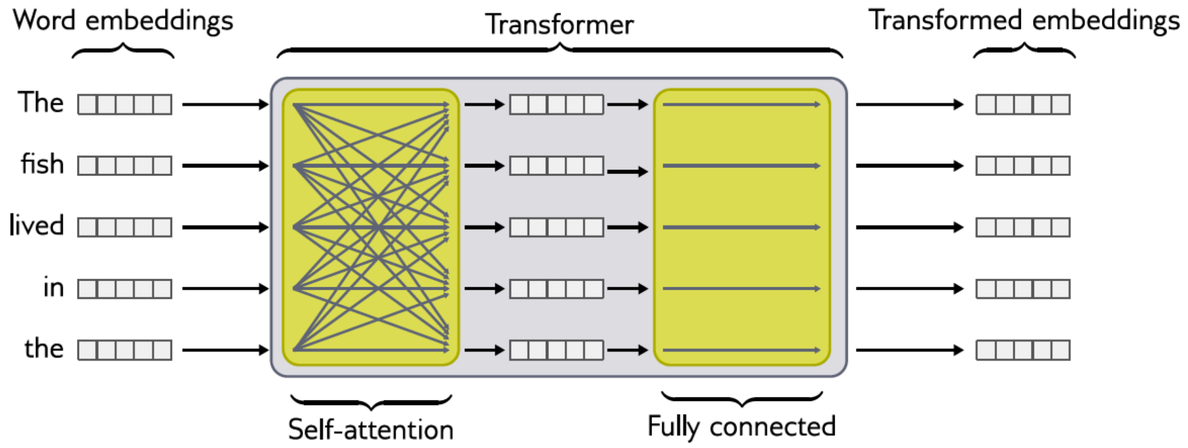
Transformer

Self-attention layer (in details next week..)

Similarities: scaled dot-product attention

Multi-head attention layers

Self-attention layer doesn’t care about the orders of the inputs!



Transformer

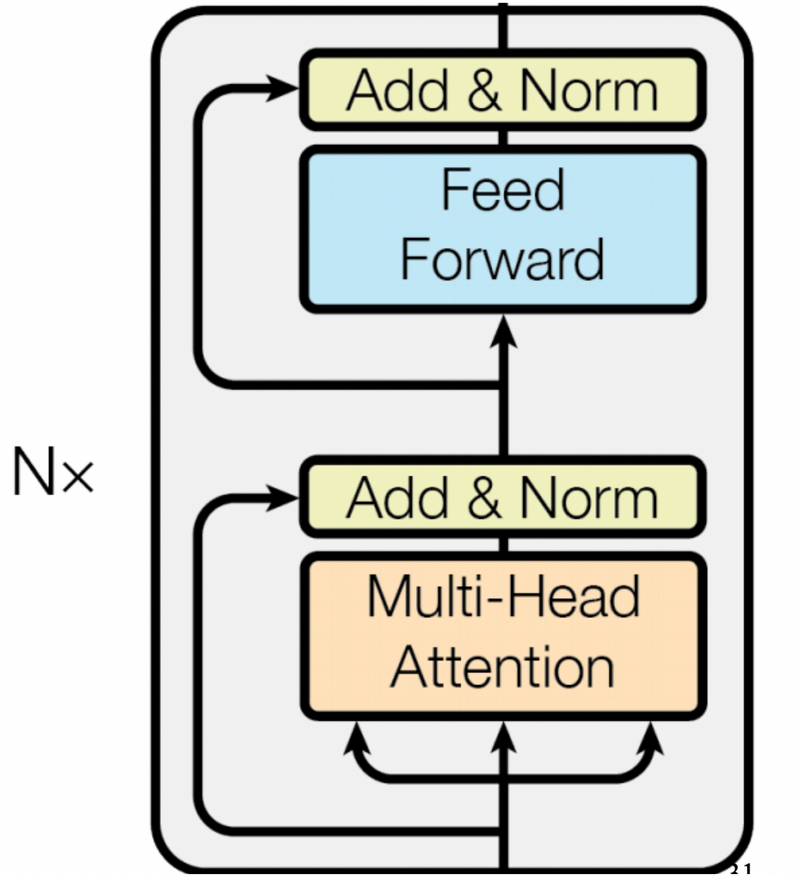
A **Transformer** is a sequence of transformer blocks

Vaswani et al.: N=12 blocks, embedding dimension = 512, 6 attention heads

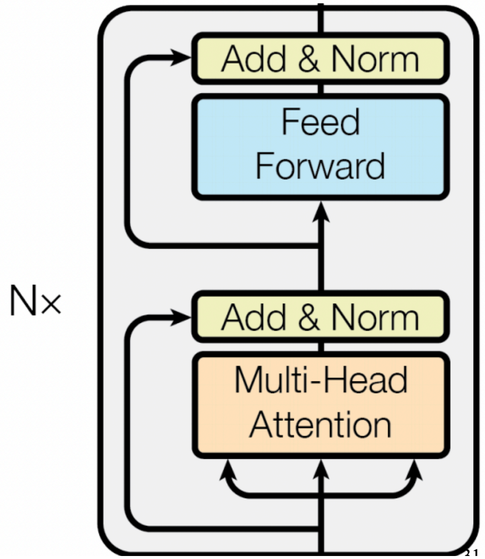
Add & Norm: residual connection followed by layer normalization (stabilizes training, regularization effect, residuals avoid vanishing gradients)

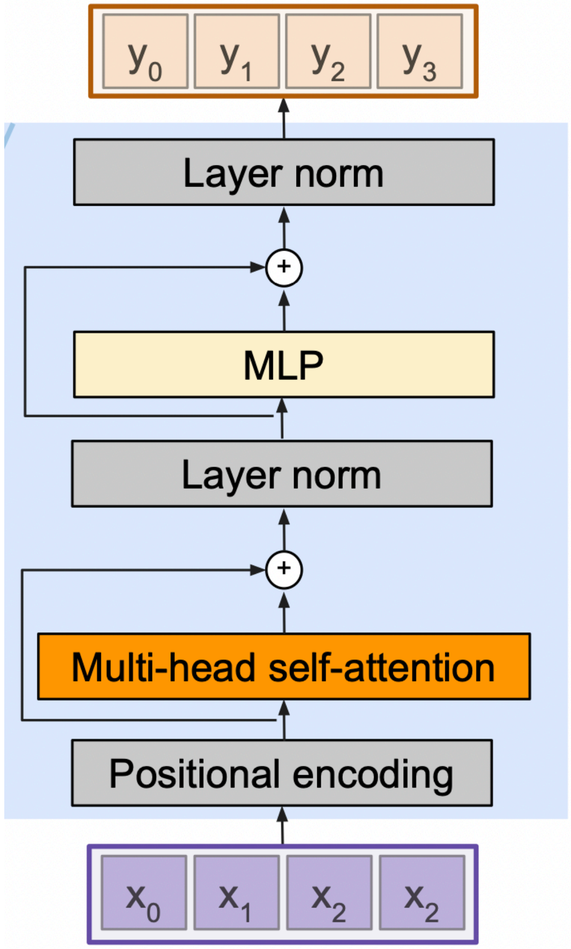
Feedforward: two linear layers with ReLUs (non linearity, learn from each other)

Attention is the only interaction between inputs!



Transformer (Encoder)





Transformer Encoder Block:

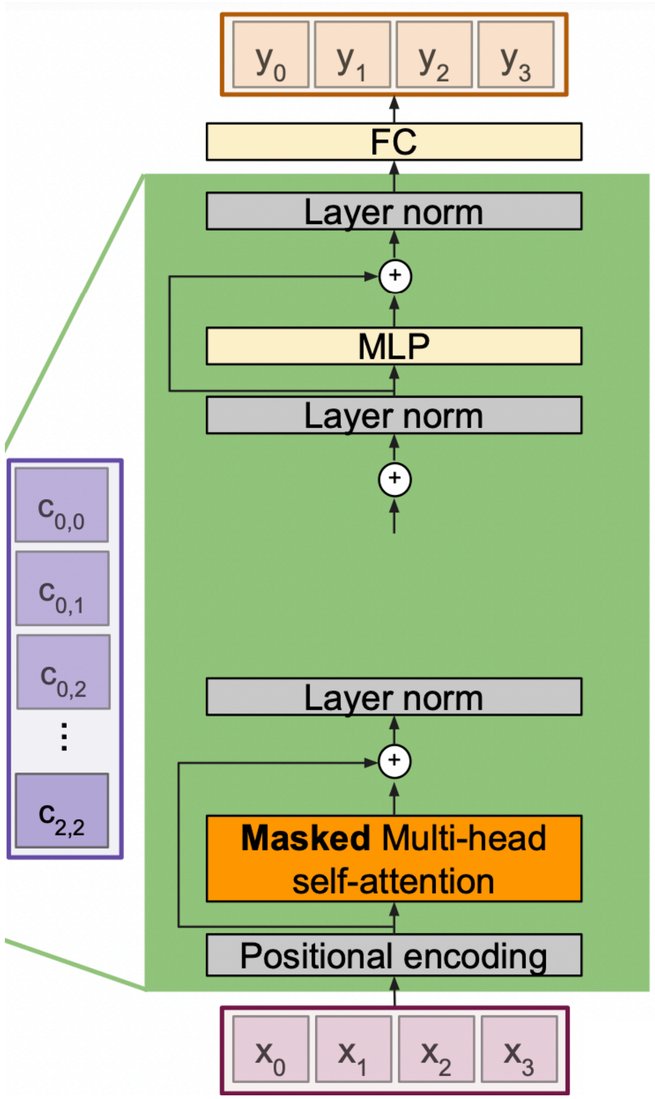
Inputs: Set of vectors **x**
Outputs: Set of vectors **y**

Self-attention is the only interaction between vectors.

Layer norm and MLP operate independently per vector.

Highly scalable, highly parallelizable, but high memory usage.

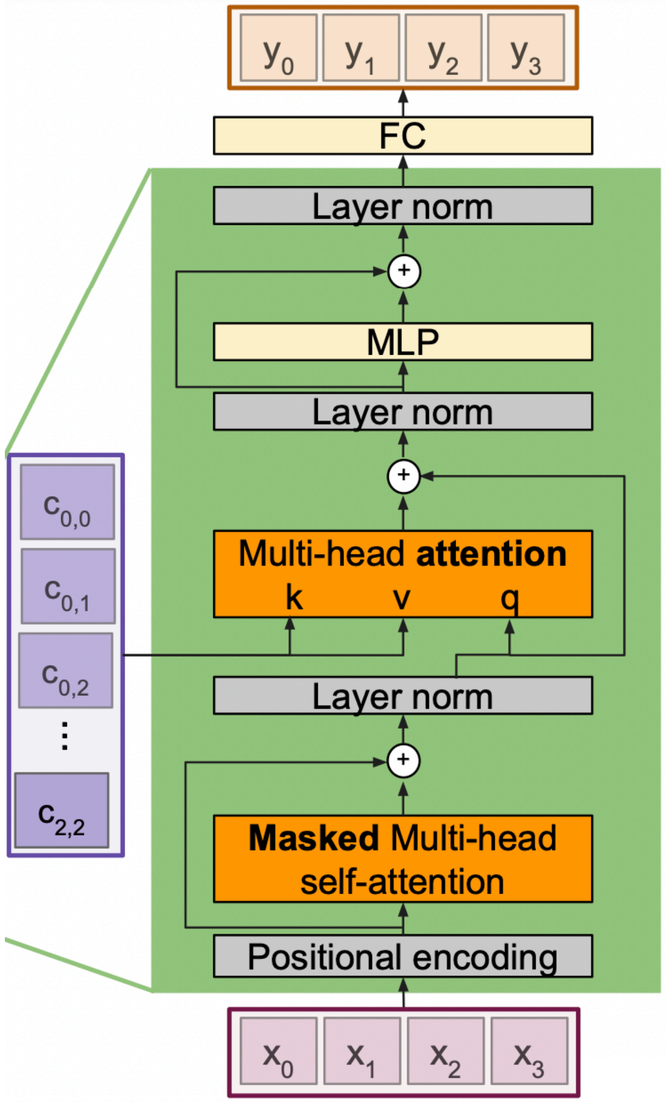
Transformer (Decoder)



Most of the network is the same the transformer encoder.

Transformer (Decoder)

The Transformer decoder is autoregressive at inference time and non-autoregressive at training time.



Transformer Decoder Block:

Inputs: Set of vectors **x** and Set of context vectors **c.**
Outputs: Set of vectors **y.**

Masked Self-attention only interacts with past inputs.

Multi-head attention block is NOT self-attention. It attends over encoder outputs.

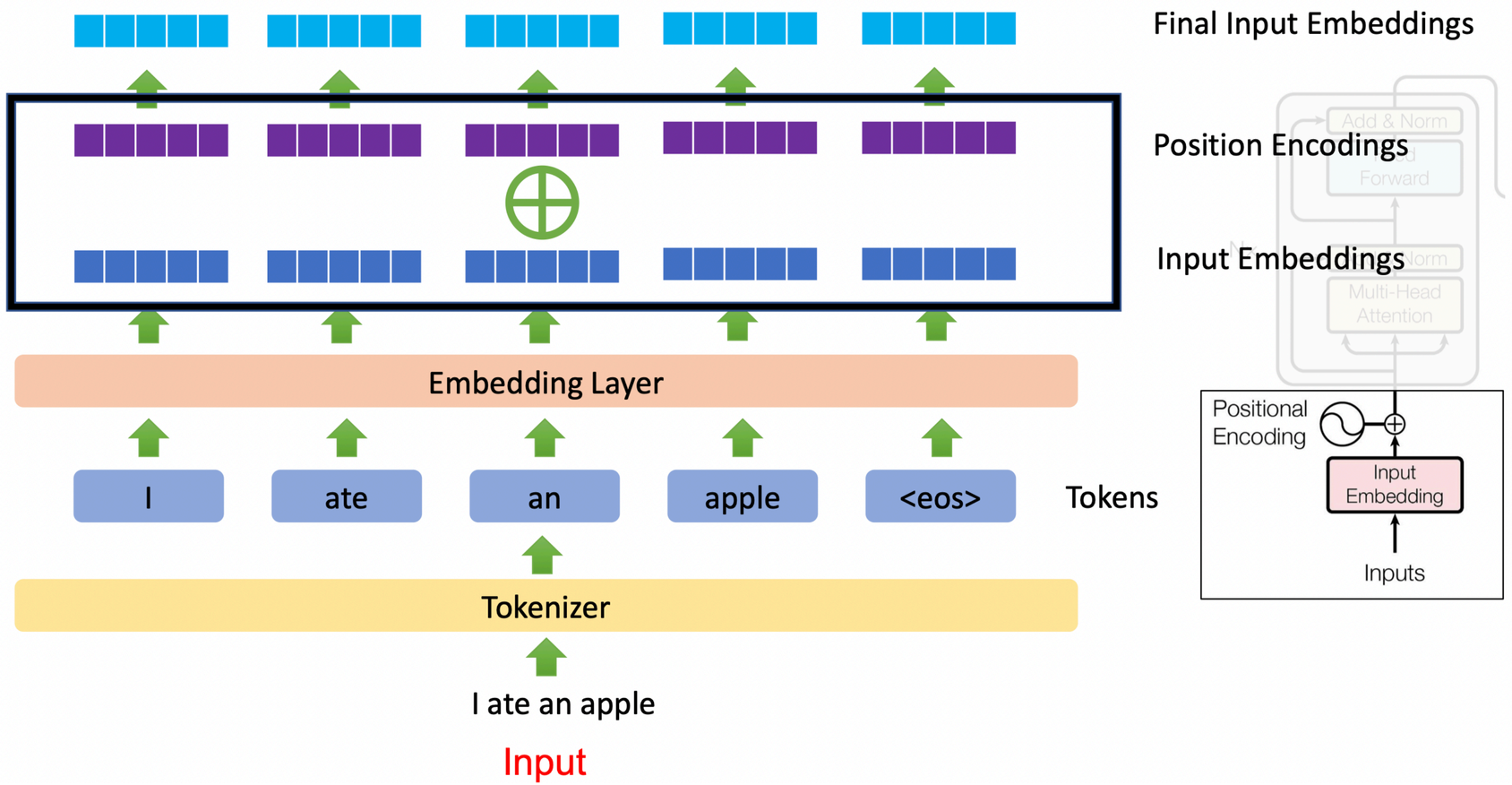
Highly scalable, highly parallelizable, but high memory usage.

Transformer

RNNs
(+) LSTMs work reasonably well for long sequences.
(-) Expects an ordered sequences of inputs
(-) Sequential computation: subsequent hidden states can only be computed after the previous ones are done.

Transformer:
(+) Good at long sequences. Each attention calculation looks at all inputs.
(+) Can operate over unordered sets or ordered sequences with positional encodings.
(+) Parallel computation: All alignment and attention scores for all inputs can be done in parallel.
(-) Requires a lot of memory: N x M alignment and attention scalers need to be calculated and stored for a single self-attention head. (but GPUs are getting bigger and better)

Tokenization



Tokenization

Goal: Tokenizer chooses input “units”

Inevitably, some words (e.g., names) will not be in the vocabulary.

The vocabulary would need different tokens for versions of the same word with different suffixes (e.g., walk, walks, walked, walking) and there is no way to clarify that these variations are related

Solution: Sub-word tokenization

Tokenization

Tokenization (Training) Algorithms (in ascending token-size order):

- Character-level (no training needed)
- BPE
- WordPiece
- Word-level (no training needed)

- Other methods:
  SentencePiece (can tokenize without pre-tokenization using white spaces)
  Morphologically aware tokenizers

Tokenization

Character-level

- Small vocabulary, just the number of unique characters in the training data (reduces the memory requirement in terms of model size)
- Enable diverse modeling (different languages)
- Spend more capacity to reach a higher-level representation (e.g. “t”, “h,” “th” in English)
- Potential loss when input sequence is too long

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, et al. 2021. ByT5: Towards a token-free future with pre-trained byte-to-byte models. arXiv preprint arXiv:2105.13626 (2021).

Tokenization

Byte Pair Encoding (BPE)

All unique words are first extracted.

A base vocabulary is then constructed from all symbols occurring in the unique words.

The final vocabulary is built by merging the symbols according to the frequencies of consecutive symbols or subwords.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Asso ciation for Computational Linguistics, Berlin, Germany, 1715–1725. https://doi.org/10.18653/v1/P16-1162

Tokenization

Byte Pair Encoding (BPE)

Base vocab: b, g, h, n, p, s, u

word	frequency
hug	10
pug	5
pun	12
bun	4
hugs	5

**Example from CS685 University of Massachusetts Amherst*

Tokenization

Byte Pair Encoding (BPE)

word	frequency
h+u+g	10
p+u+g	5
p+u+n	12
b+u+n	4
h+u+g+s	5

character pair	frequency
ug	20
pu	17
un	16
hu	15
gs	5

Tokenization

Byte Pair Encoding (BPE)

word	frequency
h+*ug*	10
p+*ug*	5
p+u+n	12
b+u+n	4
h+*ug*+s	5

character pair	frequency
un	16
h+*ug*	15
pu	12
p+*ug*	5
ug+*s*	5

Tokenization

Byte Pair Encoding (BPE)

word	frequency
hug	10
p+*ug*	5
p+*un*	12
b+*un*	4
hug + s	5

new vocab: b, g, h, n, p, s, u, ug, un, hug

Tokenization

Byte Pair Encoding (BPE)

GPT-2 uses bytes as the base vocabulary (size 256) and then applies BPE on top of this sequence (with some rules to prevent certain types of merges).

Commonly have vocabulary sizes of 32K to 64K

Tokenization

WordPiece

Main difference from BPE is that WordPiece merges symbols towards maximizing a likelihood score of language modeling.

$$\frac{\text{The probability of the merged symbol}}{\text{Individual probabilities (multiplication)}} > \text{The probability of any other symbol pair}$$

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and Korean voice search. In 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'12) (Kyoto, Japan). 5149–5152. https://doi.org/10.1109/ICASSP.2012. 6289079

Tokenization

Word-level

Splits text according to the spaces between words

Requires no vocabulary training

Requires more vocabulary size to properly tokenize the same amount of text

Out-of-vocabulary or unknown tokens are likely to be observed

Tokenization

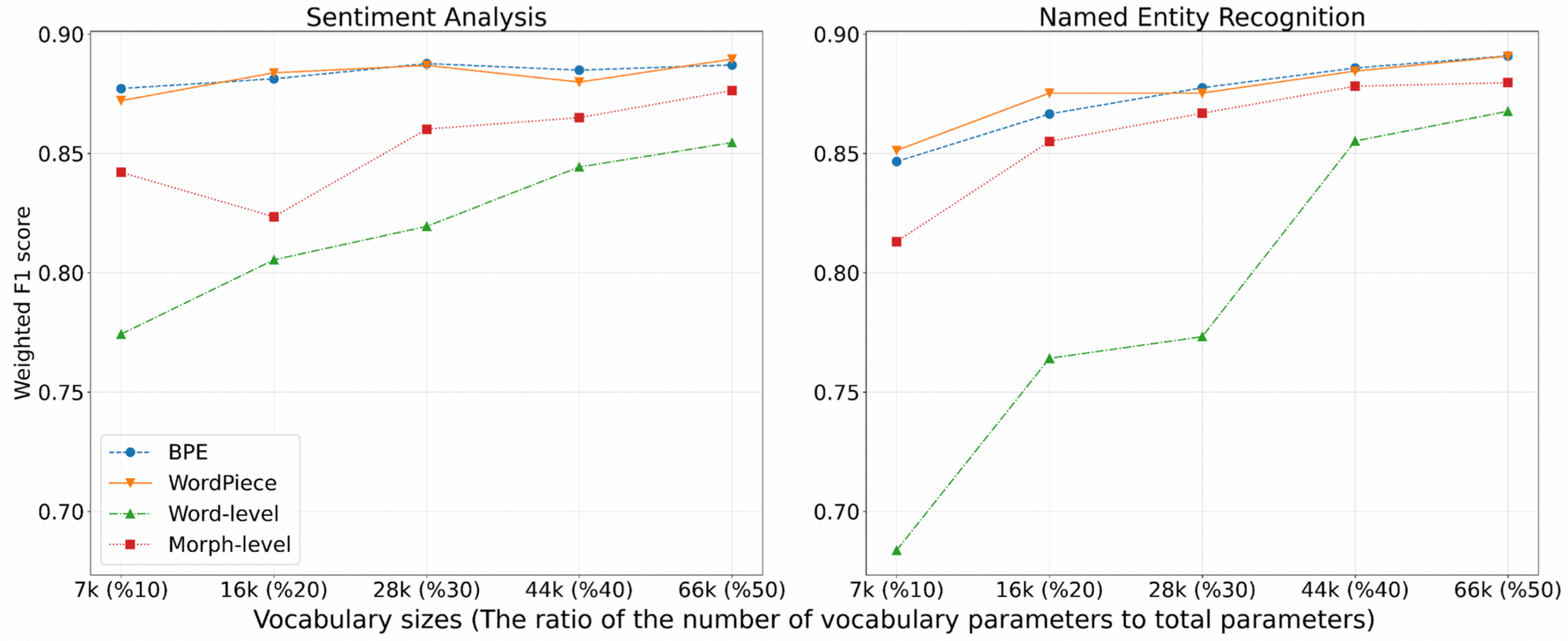
Table 3. Outputs of Different Tokenization Methods for a Sample Input, “Toplumsal barış sağlanır” (translated as “Social Peace Would be Achieved”)

Method	Tokenized text
Character-level	“t”, “o”, “p”, “l”, “u”, “m”, “s”, “a”, “l”, “ ”, “b”, “a”, “r”, “ı”, “ş”, “ ”, “s”, “a”, “ğ”, “l”, “a”, “n”, “ı”, “r”
BPE	“[CLS]”, “toplumsal”, “barış”, “sağ”, “##lanır”, “[SEP]”
WordPiece	“[CLS]”, “toplumsal”, “barış”, “sağlan”, “##ır”, “[SEP]”
Morphological-level	“[CLS]”, “toplum”, “##sal”, “barış”, “sağ”, “##lanır”, “[SEP]”
Word-level	“[CLS]”, “[UNK]”, “barış”, “[UNK]”, “[SEP]”

Cagri Toraman, Eyup Halit Yilmaz, Furkan Şahinuç, and Oguzhan Ozcelik. 2023. Impact of Tokenization on Language Models: An Analysis for Turkish. ACM Trans. Asian Low-Resour. Lang. Inf. Process. 22, 4, Article 116 (April 2023), 21 pages. https://doi.org/10.1145/3578707

Tokenization

Trade-off between vocabulary size and performance



Cagri Toraman, Eyup Halit Yilmaz, Furkan Şahinuç, and Oguzhan Ozcelik. 2023. Impact of Tokenization on Language Models: An Analysis for Turkish. ACM Trans. Asian Low-Resour. Lang. Inf. Process. 22, 4, Article 116 (April 2023), 21 pages. https://doi.org/10.1145/3578707

Tokenization

Unknown (out-of-vocabulary) tokens

What happens when a new word is observed during inference time?

Worst case: Word-level tokenization
Best case: Character-level tokenization

Solution: Assign <UNK> token (also during training!)

Is it the optimal solution?

ORTA DOĞU TEKNİK ÜNİVERSİTESİ
MIDDLE EAST TECHNICAL UNIVERSITY

Thanks for your participation!

Çağrı Toraman
25.11.2025