CS 395T: Foundations of Modern Generative AI Fall 2026

Lecture 3: Transformer Architecture

Prof. Noah Golowich

1 Transformer architecture

A transformer converts discrete tokens into vectors, repeatedly mixes information across token positions and transforms the vector at each position, and finally converts the resulting vectors back into distributions over tokens. Generally speaking, when presenting the transformer architecture, we follow the presentation of Phuong and Hutter [3].

1.1 Token embeddings

From text to tokens. A tokenizer maps a string to a sequence of elements of a finite vocabulary $\mathcal{V}$. Character tokenization keeps the vocabulary small but produces long sequences, while word tokenization produces shorter sequences but must handle an enormous vocabulary and unseen words. Modern language models therefore usually work with *subword tokens*: frequent words may be single tokens, whereas rare words are assembled from smaller pieces.

Byte-pair encoding (BPE) learns such pieces from a corpus. Begin by representing each word as a sequence of bytes. Count every adjacent pair, merge the most frequent pair into a new symbol, update the counts, and repeat until the desired number of merges has been learned. At test time, the learned merges are applied in their learned priority order. Thus common strings become short token sequences, while the base symbols ensure that an unfamiliar string can still be represented. The use of BPE to learn subword units for neural language processing was introduced by Sennrich, Haddow, and Birch [4].

Algorithm 1 Learning byte-pair encoding merge rules

Require: Corpus $\mathcal{C}$, represented as a multiset of words written as byte sequences, and merge budget M

Ensure: Ordered list $\mathcal{R}$ of at most M merge rules

$\mathcal{R} \leftarrow ()$
for $t = 1, \dots, M$ **do**
  Count every adjacent symbol pair in $\mathcal{C}$, without crossing word boundaries
  if no adjacent pair occurs **then**
    break
  end if
  Choose a most frequent pair (a_t, b_t), breaking ties by a fixed rule
  Introduce the new symbol $c_t := a_t b_t$
  for all word sequences w in $\mathcal{C}$ **do**
    Replace, from left to right, every nonoverlapping occurrence of (a_t, b_t) in w by c_t
  end for
  Append the rule $r_t := ((a_t, b_t) \mapsto c_t)$ to $\mathcal{R}$
end for
return $\mathcal{R}$

Embedding lookup. Let $N_{\mathcal{V}} := |\mathcal{V}|$ be the vocabulary size, let d be the model dimension, and identify the tokens with $[N_{\mathcal{V}}]$. For $v \in [N_{\mathcal{V}}]$, let $\delta_v \in \mathbb{R}^{N_{\mathcal{V}}}$ be the vth standard basis vector. The learned embedding matrix $W_E \in \mathbb{R}^{d \times N_{\mathcal{V}}}$ stores one vector per token: token v is represented by the vth column $e(v) := W_E \delta_v$.

Algorithm 2 Token embedding (bias-free version of Algorithm 1 in [3])

Require: Token ID $v \in [N_{\mathcal{V}}]$ and embedding matrix $W_E \in \mathbb{R}^{d \times N_{\mathcal{V}}}$
Ensure: Token representation $e \in \mathbb{R}^d$
1: **return** $e \leftarrow W_E \delta_v = W_E[:, v]$

Embedding geometry is learned jointly with the rest of the model.

1.2 Attention

Attention lets every query position form a content-dependent weighted average of context values. We first define the notion of a single *attention head*, which describes how each token of a *primary sequence* can attend to each token of a *context sequence* (in original implementations of attention, such primary and context sequences were different, though in modern decoder-only LMs, they are the same).

Let $X = [x_1, \ldots, x_{T_x}] \in \mathbb{R}^{d_x \times T_x}$ contain the primary sequence and let $Z = [z_1, \ldots, z_{T_z}] \in \mathbb{R}^{d_z \times T_z}$ contain the context sequence. Fix a query/key dimension d_k and value dimension d_v, and let

$$W_Q \in \mathbb{R}^{d_k \times d_x}, \qquad W_K \in \mathbb{R}^{d_k \times d_z}, \qquad W_V \in \mathbb{R}^{d_v \times d_z}$$

be learned projection matrices. A mask $M \in \{0,1\}^{T_z \times T_x}$ specifies which context positions are available to each query: $M_{ji} = 1$ means that query position i may use context position j. For a matrix $S \in \mathbb{R}^{T_z \times T_x}$, define the *column-wise softmax* by

$$\mathrm{softmax}(S)_{ji} := \frac{\exp(S_{ji})}{\sum_{u=1}^{T_z} \exp(S_{ui})}.$$

Algorithm 3 Single-head masked attention (Algorithm 4 of [3])

Require: Primary representations $X \in \mathbb{R}^{d_x \times T_x}$, context representations $Z \in \mathbb{R}^{d_z \times T_z}$, mask M, and matrices W_Q, W_K, W_V
Ensure: Updated representations $Y \in \mathbb{R}^{d_v \times T_x}$
1: $Q \leftarrow W_Q X$, $K \leftarrow W_K Z$, and $V \leftarrow W_V Z$
2: $S \leftarrow K^\top Q / \sqrt{d_k}$
3: **for all** $j \in [T_z]$ and $i \in [T_x]$ such that $M_{ji} = 0$ **do**
4: $\quad S_{ji} \leftarrow -\infty$
5: **end for**
6: $A \leftarrow \mathrm{softmax}(S)$ ▷ A_{ji} is the weight from context j to query i
7: **return** $Y \leftarrow VA$

For a fixed query i, the weights $A_{1i}, \ldots, A_{T_z i}$ are nonnegative and sum to one. The scale $1/\sqrt{d_k}$ prevents the dot products from growing with d_k under the usual roughly unit-variance initialization.

Unmasked and causal self-attention. For self-attention, set $Z = X$. With no mask, $M_{ji} = 1$ for every i, j, so every token may attend to every other token.

For language modeling, self-attention is inadequate since we need to predict each token as a function of only the previous ones; thus we introduce *causal attention*, where we use the *causal mask*

$$M_{ji} := \mathbf{1}\{j \le i\}. \tag{1}$$

Then output column $Y[:, i]$ depends only on $x_1, \dots, x_i$, so it can be used to predict token $i+1$ without seeing the future.

Multi-head attention. A single head produces only one attention distribution for each query. However, different linguistic or algorithmic relationships may require looking in different places: for example, consider predicting the last (boldfaced) word of the sentence "The French word for dog is **chien**."

Let H be the number of heads. Head $h \in [H]$ has its own matrices $W_Q^{(h)}, W_K^{(h)}, W_V^{(h)}$ and returns $Y^{(h)} \in \mathbb{R}^{d_v \times T_x}$. A learned output matrix $W_O \in \mathbb{R}^{d_{\text{out}} \times H d_v}$ mixes the vertically concatenated head outputs.

Algorithm 4 Multi-head attention (Algorithm 5 of [3])

Require: Sequences X, Z, mask M, head parameters $\{W_Q^{(h)}, W_K^{(h)}, W_V^{(h)}\}_{h=1}^{H}$, and output matrix W_O

1: **for** $h = 1, \dots, H$ **do**
2: $\quad Y^{(h)} \leftarrow \textsc{Attention}(X, Z, M; W_Q^{(h)}, W_K^{(h)}, W_V^{(h)})$
3: **end for**
4: $Y \leftarrow [Y^{(1)}; Y^{(2)}; \dots; Y^{(H)}]$ ▷ $Y \in \mathbb{R}^{H d_v \times T_x}$
5: **return** $W_O Y$

In the common balanced choice, every head has $d_k = d_v = d/H$, so concatenation restores model dimension d.

1.3 Additional components

1.3.1 Positional information

Attention by itself has no notion of token order: permuting the input columns permutes the output columns in the same way. Positional information breaks this symmetry.

Absolute sinusoidal embeddings. The original Transformer [7] added a fixed vector $p_t \in \mathbb{R}^d$ to the token embedding at absolute position ID $t \in \{0, 1, \dots\}$. For $r = 0, \dots, d/2 - 1$, its coordinates are

$$p_t[2r+1] = \sin\left(t/10000^{2r/d}\right), \qquad p_t[2r+2] = \cos\left(t/10000^{2r/d}\right). \tag{2}$$

The initial representation is $x_t^{(0)} = W_E \delta_{v_t} + p_t$. These are called *absolute* positional embeddings because the vector injected at position t is determined directly by t. The geometric range of frequencies allows linear operations to compare positions at several scales.

Rotary position embeddings. RoPE instead rotates queries and keys by angles determined by their absolute position IDs [6]. Let the head dimension d_k be even. For an angle ϕ, define

$$R(\phi) := \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix}.$$

For $r = 0, \ldots, d_k/2 - 1$, set $\theta_r := 10000^{-2r/d_k}$, and define the block-diagonal rotation

$$R_t := \mathrm{diag}(R(t\theta_0), \ldots, R(t\theta_{d_k/2-1})). \tag{3}$$

If $\bar{q}_i := W_Q x_i$ and $\bar{k}_j := W_K x_j$ are the content query and key, RoPE uses $q_i := R_i \bar{q}_i$ and $k_j := R_j \bar{k}_j$. To understand the effect this has on attention scores, note that the attention score S_{ji} is given by

$$\begin{aligned} q_i^\top k_j &= (R_i \bar{q}_i)^\top (R_j \bar{k}_j) = \bar{q}_i^\top R_i^\top R_j \bar{k}_j \\ &= \bar{q}_i^\top R_{j-i} \bar{k}_j. \end{aligned} \tag{4}$$

Indeed, $R(a)^\top R(b) = R(b-a)$ in every two-dimensional block. Consequently, the positional part of the query–key score depends on the position IDs only through the relative displacement $j - i$; thus RoPE is a *relative* positional embedding.

1.3.2 Layer normalization and RMSNorm

Normalization is applied separately to each token representation across its d features. For $x \in \mathbb{R}^d$, define

$$\mu(x) := \frac{1}{d} \sum_{r=1}^{d} x_r, \qquad \sigma^2(x) := \frac{1}{d} \sum_{r=1}^{d} (x_r - \mu(x))^2,$$

and let $\epsilon_{\mathrm{norm}} > 0$ be a fixed numerical stabilizer. With a learned scale $\gamma \in \mathbb{R}^d$ and bias $\beta \in \mathbb{R}^d$, layer normalization [1] performs

$$\mathrm{LN}_\gamma(x) := \gamma \odot \frac{x - \mu(x)\mathbf{1}_d}{\sqrt{\sigma^2(x) + \epsilon_{\mathrm{norm}}}} + \beta. \tag{5}$$

RMSNorm, which is also popular, omits the mean subtraction:

$$\mathrm{RMSNorm}_\gamma(x) := \gamma \odot \frac{x}{\sqrt{d^{-1} \sum_{r=1}^{d} x_r^2 + \epsilon_{\mathrm{norm}}}}. \tag{6}$$

It preserves rescaling invariance while avoiding the mean computation [8].

1.3.3 Position-wise MLPs

The original Transformer applies the same two-layer MLP independently at every position. For a hidden width d_{ff}, matrices $W_1 \in \mathbb{R}^{d_{\mathrm{ff}} \times d}$ and $W_2 \in \mathbb{R}^{d \times d_{\mathrm{ff}}}$ define

$$\mathrm{MLP}_{\mathrm{ReLU}}(x) := W_2 \, \mathrm{ReLU}(W_1 x). \tag{7}$$

Vaswani et al. used $d_{\mathrm{ff}} = 4d$ in their base model [7].

The origianl MLP has, over time, been replaced with variants which (a) change the activation and (b) add *gating*. In particular, define $\mathrm{SiLU}(a) := a/(1 + e^{-a})$ coordinatewise. With $W_g, W_u \in \mathbb{R}^{d_{\mathrm{ff}} \times d}$ and $W_2 \in \mathbb{R}^{d \times d_{\mathrm{ff}}}$,

$$\mathrm{MLP}_{\mathrm{SwiGLU}}(x) := W_2(\mathrm{SiLU}(W_g x) \odot W_u x). \tag{8}$$

Because SwiGLU has three matrices rather than two, Shazeer reduced its hidden width by a factor 2/3 when matching the parameter and operation count of the original MLP [5]. He closed the paper with the memorable explanation:

> "We offer no explanation as to why these architectures seem to work; we attribute their success, as all else, to divine benevolence."

1.3.4 Unembedding

The unembedding converts a contextual representation back into a distribution over the vocabulary. Let $W_U \in \mathbb{R}^{N_\mathcal{V} \times d}$ be the unembedding matrix.

Algorithm 5 Unembedding (bias-free Algorithm 7)

Require: Contextual token representation $x \in \mathbb{R}^d$ and $W_U \in \mathbb{R}^{N_\mathcal{V} \times d}$
Ensure: Distribution p over $\mathcal{V}$
1: $z \leftarrow W_U x$ ▷ one logit per vocabulary element
2: **return** $p \leftarrow \mathrm{softmax}(z)$

The matrix W_U may be learned independently, or its weights may be tied to the embedding matrix by setting $W_U = W_E^\top$.

1.4 Putting the components together

A decoder-only transformer stacks causal pre-normalized blocks and predicts the next token at every position. Let L be the number of layers. Starting from $X^{(0)} = [x_1^{(0)}, \ldots, x_T^{(0)}] \in \mathbb{R}^{d \times T}$, layer $r \in [L]$ computes

$$U^{(r)} = X^{(r-1)} + MHA_r\Big(\mathrm{Norm}^{(r)}_{\mathrm{attn}}(X^{(r-1)}); M_{\mathrm{causal}}\Big), \tag{9}$$

$$X^{(r)} = U^{(r)} + MLP_r\Big(\mathrm{Norm}^{(r)}_{\mathrm{mlp}}(U^{(r)})\Big). \tag{10}$$

Here normalization is applied to each column, and the additions are residual connections. These equations are *pre-norm*: normalization occurs before each attention or MLP sublayer.

With additive absolute position embeddings, $x_t^{(0)} = W_E \delta_{v_t} + p_t$; with RoPE, one typically sets $x_t^{(0)} = W_E \delta_{v_t}$ and applies the rotations inside every attention layer.

After the final layer, set

$$H = \mathrm{Norm}_{\mathrm{final}}(X^{(L)}), \qquad Z = W_U H, \qquad P = \mathrm{softmax}(Z), \tag{11}$$

where softmax is applied to each column. By the causal mask, column $P[:, t]$ depends only on $v_1, \ldots, v_t$ and represents the model's distribution for v_{t+1}.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. arXiv:1607.06450, 2016. arXiv.

[2] Huwan Peng. *Methodologies and Architectures for AI Inference Hardware: From Foundational Networks to Large Language Models*. PhD thesis, University of Washington, 2025. Thesis (PDF).

[3] Mary Phuong and Marcus Hutter. Formal algorithms for transformers. arXiv:2207.09238, 2022. arXiv.

[4] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1715–1725, 2016. doi:10.18653/v1/P16-1162.

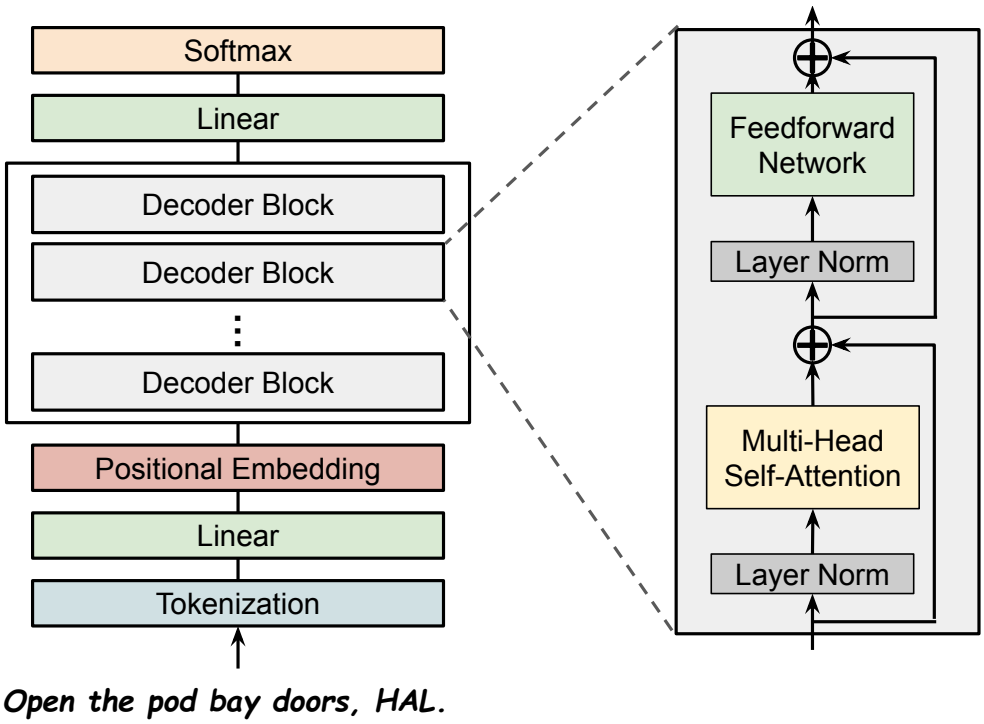

Figure 1: A decoder-only Transformer with a pre-norm block: each LayerNorm precedes its multi-head attention or feed-forward sublayer. Reproduced from Figure 2.2 of Peng [2]. See Equations (9)–(11).

[5] Noam Shazeer. GLU variants improve transformer. arXiv:2002.05202, 2020. arXiv.

[6] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. arXiv:2104.09864, 2021. arXiv.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008, 2017. NeurIPS.

[8] Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems 32*, 2019. NeurIPS.

AI Use. I prepared these lecture notes with the aid of AI assistance; I take full responsibility for all content in the notes.