MATH 565: Lecture 29 (04/28/2026)

Today: * Optimization in transformer networks

Transformer Networks

Motivation: Feedforward NNs map fixed size inputs to outputs. They do not have built-in notions of

1. variable length inputs
2. sequential structure in inputs
3. dependencies between positions in a sequence

Transformer networks: work with a set of tokens as input, and allow each token to interact with every other token.

The goal of this lecture is to give a brief overview of transformer networks and how the results for NNs generalize to their setting.

The Setup

* Vocabulary (V): set of all distinct tokens the model can process
  $|V|$: size of vocabulary $|V| \approx 30{,}000 - 50{,}000$ → for LLMs

* Embedding, positional encoding: each token is mapped to a point in $\mathbb{R}^d$ via a learned embedding matrix $E \in \mathbb{R}^{|V| \times d}$ where d is the model dimension, e.g., $d = 768$ in GPT_2, $d = 12{,}228$ in GPT-3, etc.

  Embedding a token t is equivalent to looking up row t of E.

For a given input of T tokens, the order/position of the tokens is important. We use position encodings $\bar{p}_1, \ldots, \bar{p}_T \in \mathbb{R}^d$ and add them to the token embeddings to get the token matrix $X \in \mathbb{R}^{T \times d}$. Thus, the rows of X are d-dimensional representations of the T input tokens.

→ $\bar{x}_t = \bar{e}_t + \bar{p}_t$, $\bar{e}_t = E_t$.
↳ t^{th} row of E

Self-Attention Mechanism (with one attention head)

Q: For a given set of T tokens, for each token $t \in T$, self attention asks "given what t is about, which other tokens in the sequence are most relevant, and what info should I retrieve from them?"

Answer in three steps

1. Compare t to all other tokens to get a relevance score.
2. Normalize these scores to a probability using the softmax function.
3. Retrieve a weighted average of information from all tokens using these probabilities as weights.

We project $\bar{x}_t$ (t^{th} row of X taken as a d-vector) in three ways:

* query $\bar{q}_t = W_Q^T \bar{x}_t$ (what token t is "looking for")
* key $\bar{k}_s = W_k^T \bar{x}_s$ (what token s "has to offer")
* value $\bar{v}_s = W_v^T \bar{x}_s$ (what token s contributes, if selected)

where $W_Q, W_k, W_v \in \mathbb{R}^{d \times d_h}$ (matrices to be learned)

The relevance of token s to token t is measured by $\bar{q}_t^T \bar{k}_s / \sqrt{d_h}$ for d_h = head dimension ($d_h = d$, by default)

Softmax then normalizes these scores across all s.

Def $\bar{a} = \text{softmax}(\bar{z})$ is defined by setting

$$a_i = \frac{e^{z_i}}{\sum_{k=1}^{d} e^{z_k}} \; ; \quad a_i\text{'s are probabilities, as } \sum_{i=1}^{d} a_i = 1.$$

For 2-d case with $z_1 = z$, $z_2 = 0$, the softmax function reduces to the sigmoid function: $\frac{e^z}{e^z + 1}$.

$$a_{ts} = \frac{e^{(\bar{q}_t^T \bar{k}_s / \sqrt{d_h})}}{\sum_{s'}^{T} e^{(\bar{q}_t^T \bar{k}_{s'} / \sqrt{d_h})}}$$

attention weights

The output for token t, called the **attention output**, is

$$\text{Attn}_t = \sum_{s=1}^{T} a_{ts} \bar{v}_s$$

We write the attention for all tokens together as follows:

query $Q = XW_Q$
key $K = XW_K$
value $V = XW_V$

Recall, $W_Q, W_K, W_V \in \mathbb{R}^{d \times d_h}$

The attention output is

$$\text{Attn}(Q,K,V) = \underbrace{\text{softmax}}_{\text{applied row-wise}}\left(\frac{QK^T}{\sqrt{d_h}}\right)V \in \mathbb{R}^{T \times d_h}$$

$QK^T \in \mathbb{R}^{T \times T}$ has all pairwise compatibility scores between tokens.

$\text{softmax}\left(\frac{QK^T}{\sqrt{d_h}}\right)$ applied row-wise gives the stochastic attention matrix $A \in \mathbb{R}^{T \times T}$, and AV gives the weighted combination of value vectors.

Q: Why divide by $\sqrt{d_h}$? (normal distribution)

If Q, K entries are $\mathcal{N}(0,1)$, then QK^T entries have variance d_h. Hence, the $\sqrt{d_h}$ scaling restores unit variance. Else, we could get ≈ 0 Jacobians (i.e., vanishing gradient problem).

Multi-head Attention

Run h attention heads in parallel with independent projections and concatenate:

$$\text{MHA}(X) = \text{cat}(\text{Attn}_1, \ldots, \text{Attn}_h) \cdot W_O \quad | \quad W_O \in \mathbb{R}^{d \times d}$$

$\sum_{i=1}^{h} d_{h_i} = d$ typically.

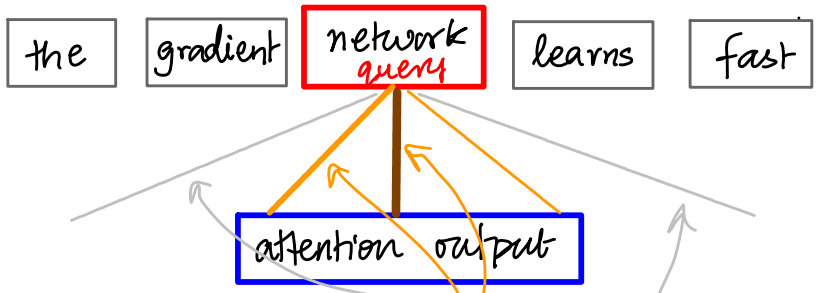
Why MHA?

"The bank by the river offers high interest rates."

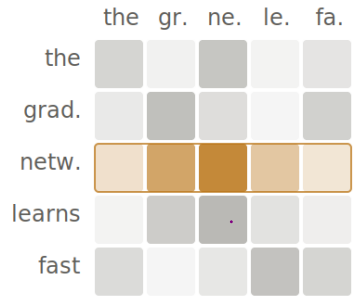
For t = "bank" needs attention from "river" and from "interest rates".

Illustration of Attention Block

$T=5$ $t=3$ ("network")



attention weights: high, low (thickness + dark color)



softmax matrix

The Transformer Block

One transformer block applies two operations in sequence, each with a residual (skip) connection and a layer normalization (LN).

$$\tilde{X} = LN\left(X + MHA(X)\right)$$

$$X' = LN\left(\tilde{X} + FFN(\tilde{X})\right)$$

Layer Normalization (LN):

$$LN(\bar{z}) = \frac{\bar{z} - \mu}{\sigma} \odot \bar{\gamma} + \bar{\beta} \quad \text{where}$$

μ, σ : mean and standard deviation of $\bar{z}$ (a single token)

$\bar{\gamma}, \bar{\beta}$: learned scale and shift parameter vectors.

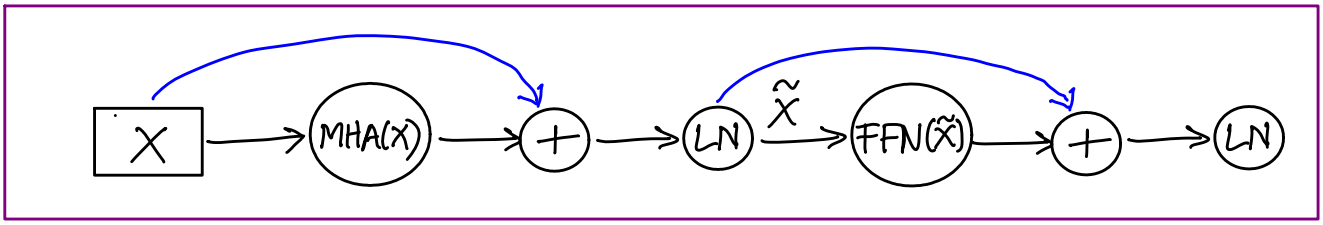
$\bar{z} \in \mathbb{R}^d$

Pointwise Feedforward Network (FFN) function:

$$FFN(\vec{z}) = W_2\, \Phi(W_1 \vec{z} + \vec{b}_1) + \vec{b}_2$$

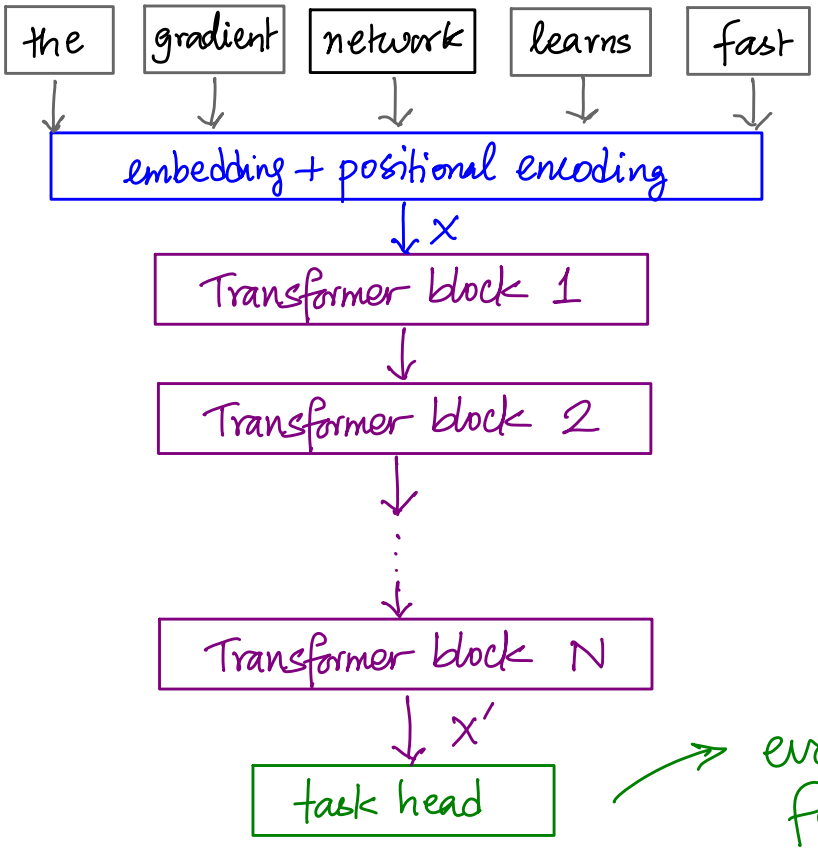
for $W_1 \in \mathbb{R}^{d_{ff} \times d}$, $W_2 \in \mathbb{R}^{d \times d_{ff}}$, $d_{ff} \approx 4d$ is the feedforward dimension.

→ compare to $\vec{s}^{(\ell)} = W^{(\ell)} \cdot \vec{y}^{(\ell-1)}$ in NNs, followed by $\Phi(\vec{s}^{(\ell)})$.



The Transformer Stack

A full transformer network consists of N identical transformer blocks stacked between the embedding and a task-specific output head. The same matrix X flows through each block, unchanged in shape.



Backpropagation through the transformer network works just like in Feedforward neural networks.

Theoretical Results

Theorem 17 (Jacobian of softmax)
Let $\bar{a} = \text{softmax}(\bar{z}) \in \mathbb{R}^T$ be defined by $a_i = \frac{e^{z_i}}{\sum_k e^{z_k}}$. Then, the softmax is differentiable and its Jacobian is given by

$$\mathcal{D}\bar{a} = \frac{\partial \bar{a}}{\partial \bar{z}} = \text{diag}(\bar{a}) - \bar{a}\bar{a}^T \in \mathbb{R}^{T \times T}$$

Proof

For $i=j$, $\frac{\partial a_i}{\partial z_i} = \frac{(\sum_k e^{z_k})e^{z_i} - e^{z_i}\cdot e^{z_i}}{(\sum_k e^{z_k})^2} = a_i - a_i^2 = a_i(1-a_i)$.

For $i \neq j$, $\frac{\partial a_i}{\partial z_j} = \frac{(\sum_k e^{z_k})\cdot 0 - e^{z_i}\cdot e^{z_j}}{(\sum_k e^{z_k})^2} = -a_i a_j$.

$$\Rightarrow \mathcal{D}\bar{a} = \frac{\partial \bar{a}}{\partial \bar{z}} = \text{diag}(\bar{a}) - \bar{a}\bar{a}^T$$

□

Corollary 18 (Gradient through softmax) If J is a scalar loss function and $\bar{g} = \frac{\partial J}{\partial \bar{a}}$ is the upstream gradient, then

$$\frac{\partial J}{\partial \bar{z}} = (\text{diag}(\bar{a}) - \bar{a}\bar{a}^T)\bar{g} = \bar{a} \odot \bar{g} - (\bar{a}^T\bar{g})\bar{a}$$

Remark $\operatorname{rank}(\mathcal{D}\bar{a}) = T-1$, since $(\mathcal{D}\bar{a})\bar{1} = \bar{0}$ as

$$(\operatorname{diag}(\bar{a}) - \bar{a}\bar{a}^T)\bar{1} = \bar{a} - \bar{a}\underbrace{(\bar{a}^T\bar{1})}_{=1} = \bar{0} \text{ as}$$

$\bar{a}^T\bar{1} = \sum a_k = 1$

In contrast, $\operatorname{diag}(\bar{\Phi}'(\bar{z}))$ has full rank in FFNNs.

Theorem 19 (gradient bound with residual connections) For a transformer network with N blocks, let

$$\bar{s}^{(\ell)} = \frac{\partial J}{\partial \operatorname{vec}(X^\ell)}$$ be the vectorized gradient of J

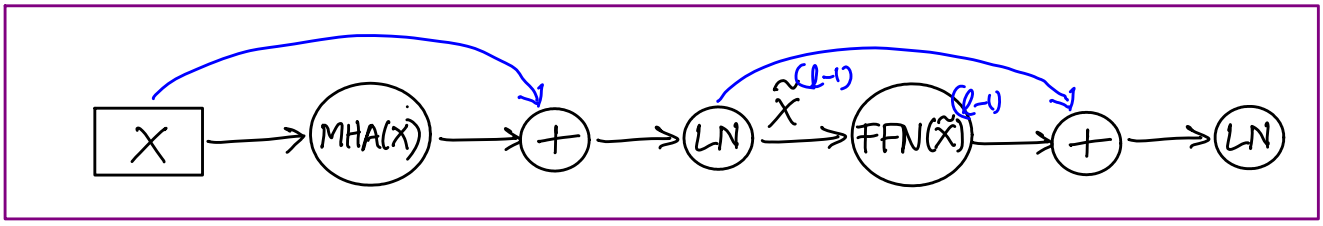
with respect to the token matrix after block ℓ (we ignore layer normalization for simplicity). Suppose for each block ℓ, the sublayer function $F^{(\ell)}$ (MHA or FFN) satisfies $\|\mathcal{D}_{F^{(\ell)}}\|_2 \le \rho$. Then

$$\|\bar{s}^{(\ell)}\|_2 \le (1+\rho)^{N-\ell} \|\bar{s}^{(N)}\|_2$$

Proof The residual connection acts as

$$X^{(\ell)} = \tilde{X}^{(\ell-1)} + F^{(\ell)}(\tilde{X}^{(\ell-1)}).$$

Recall:



Backward pass through this relation gives (by chain rule at $\oplus$ node) gives

$$\bar{s}^{(\ell-1)} = \frac{\partial J}{\partial \tilde{X}^{(\ell-1)}} = \frac{\partial J}{\partial X^{(\ell)}} \cdot \underbrace{\frac{\partial X^{(\ell)}}{\partial X^{(\ell-1)}}}_{=I} + \frac{\partial J}{\partial X^\ell} \cdot \underbrace{\frac{\partial X^\ell}{\partial F^{(\ell)}}}_{=I} \cdot \underbrace{\frac{\partial F^{(\ell)}}{\partial \tilde{X}^{(\ell-1)}}}_{=\mathcal{D}_{F^{(\ell)}}}$$

$\underbrace{\quad}_{\bar{s}^{(\ell)}}$ $\underbrace{\quad}_{\bar{s}^{(\ell)}}$

$$= \bar{\delta}^{(\ell)} + \bar{\delta}^{(\ell)} \cdot I \cdot \mathcal{D}_{F^{(\ell)}}$$

$$\Rightarrow \quad \bar{\delta}^{(\ell-1)} = \bar{\delta}^{(\ell)} + \mathcal{D}_{F^{(\ell)}}^{T} \bar{\delta}^{(\ell)}$$

Taking norms and applying submultiplicity of spectral norm gives

$$\| \bar{\delta}^{(\ell-1)} \|_2 \leq \| \bar{\delta}^{(\ell)} \|_2 + \| \mathcal{D}_{F^{(\ell)}} \|_2 \| \bar{\delta}^{(\ell)} \|_2$$

$$\leq (1+\rho) \| \bar{\delta}^{(\ell)} \|_2$$

Applying the bound inductively gives

$$\| \bar{\delta}^{(\ell-1)} \|_2 \leq (1+\rho)^{N-\ell} \| \bar{\delta}^{(N)} \|_2 . \qquad \square$$

Compare this result with that of Theorem 16, where $\sigma r < 1$ typically, resulting in exponential decay of gradients in early layers. But here, with $\rho \geq 0$, $(1+\rho)^{N-\ell}$ represent a polynomial growth, i.e., there is no decay!

Thus, the residual connections solve the vanishing gradient problem!

Optimization Challenges for Transformers

* Learning rate warmup: transformers are highly sensitive to learning rates. In practice, we increase them linearly for several thousand steps and then decay.

* Adam is mandatory: as plain SGD fails on transformers. The per-parameter adaptive scaling of Adam makes optimization tractable here.

* gradient clipping:

$$\text{if } \|\nabla J\|_2 > \tau, \quad \nabla J \leftarrow \tau \frac{\nabla J}{\|\nabla J\|_2}$$

as gradients could become large for several reasons.