An Alternate Formulation of Transformers

Residual Stream Perspective

Tanmoy Chakraborty
Associate Professor, IIT Delhi
https://tanmoychak.com/

Introduction to Large Language Models

Recall: Masked Self-Attention in Decoders

Self-Attention: Scaled dot-product attention

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

where, $Q = XW^Q, K = XW^K, V = XW^V$

Problem: While training autoregressive models (with next-word-prediction objective), Transformers 'can see the future'.

- For a current token x_i, the attention scores are computed with all tokens in the sequence including those which comes after x_i (as the whole sequence is available to us during training).

Solution: Masking

Recall: Masked Self-Attention in Decoders

Masking: 'Masked' scaled dot-product attention

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} + M\right)V$$

where, masking matrix M is defined as:

$$M_{ij} = \begin{cases} 0 & \text{if } j \leq i \\ -\infty & \text{if } j > i \end{cases}$$

For future tokens, the attention scores becomes zero after applying softmax [*softmax(-∞) = 0*].

- Effectively, **after masking**, the query is the current token x_i, and the keys and values comes from the tokens before it, including itself (i.e., $x_j,\ j \leq i$).

Re-writing the Masked Self-Attention Equation

Now let's re-write the masked attention equation for a current token x_i.

- Assume that we are considering the **attention head *h*** of **layer *l***.
- Let's denote the matrix with the **output hidden representation from layer *k* of previous tokens $x_j,\ j \leq i$** as $X^k_{\leq i}$.

Thus, for calculating attention scores for **attention head *h*** of **layer *l***, input to the attention sub-layer is the output representation from the previous layer *l-1*.

- **Query:** $x_i^{l-1}\ W_Q^{l,h}$
- **Keys:** $X_{\leq i}^{l-1}\ W_K^{l,h}$

$$a_i^{l,h} = \text{softmax}\left(\frac{\overbrace{x_i^{l-1} W_Q^{l,h}}^{\text{Query vector}} \underbrace{(X_{\leq i}^{l-1} W_K^{l,h})^\intercal}_{\text{Key vector}}}{\sqrt{d_k}}\right)$$

Ferrando et al., A Primer on the Inner Workings of Transformer-based Language Models

QK Circuit

$$\boldsymbol{a}_i^{l,h} = \text{softmax}\left(\frac{\underbrace{\boldsymbol{x}_i^{l-1}\boldsymbol{W}_Q^{l,h}}_{\text{Query vector}}\underbrace{(\boldsymbol{X}_{\leq i}^{l-1}\boldsymbol{W}_K^{l,h})^\intercal}_{\text{Key vector}}}{\sqrt{d_k}}\right)$$

$$= \text{softmax}\left(\frac{\boldsymbol{x}_i^{l-1}\boldsymbol{W}_{QK}^{h}\boldsymbol{X}_{\leq i}^{l-1\intercal}}{\sqrt{d_k}}\right),$$

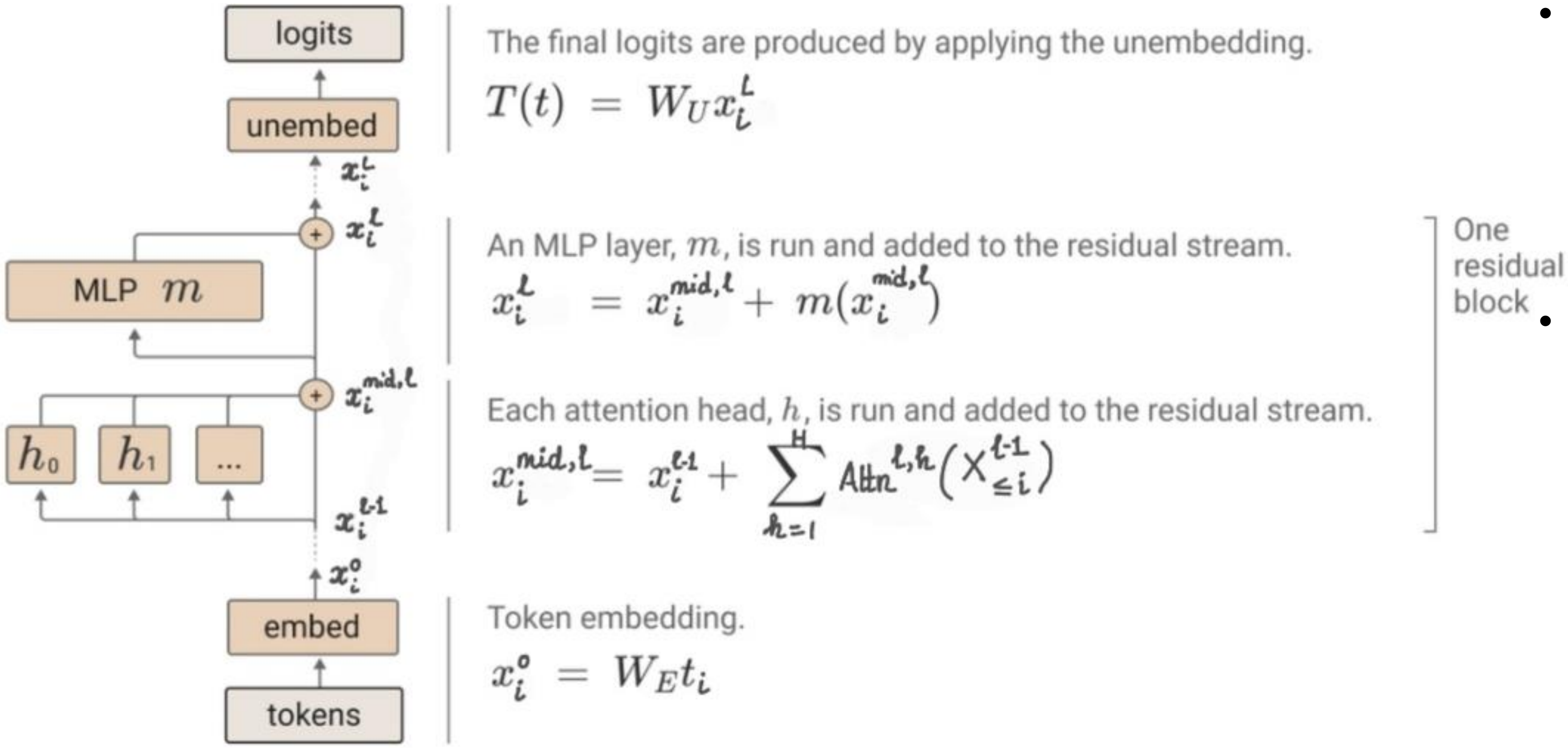
QK (query-key) circuit: $W_{QK}^h = W_Q^h W_K^{h\intercal}$

- QK circuits are responsible for reading from the **residual stream**.

Let's now look at the residual stream

Ferrando et al., A Primer on the Inner Workings of Transformer-based Language Models

Residual Stream Perspective



Elhage, et al., A Mathematical Framework for Transformer Circuits

- Each input embedding gets updated via vector additions from the attention and feed-forward blocks producing **residual stream states** (or intermediate representations).
- The final layer residual stream state is then projected into the vocabulary space via the unembedding matrix $W_U \epsilon\ R^{d \times |V|}$ and normalized via the *softmax*.

Combining the Output of Multiple Attention Heads

$$\boldsymbol{a}_i^{l,h} = \text{softmax}\left(\frac{\overbrace{\boldsymbol{x}_i^{l-1}\boldsymbol{W}_Q^{l,h}}^{\text{Query vector}}\underbrace{(\boldsymbol{X}_{\leq i}^{l-1}\boldsymbol{W}_K^{l,h})^\intercal}_{\text{Key vector}}}{\sqrt{d_k}}\right)$$

$$= \text{softmax}\left(\frac{\boldsymbol{x}_i^{l-1}\boldsymbol{W}_{QK}^h\boldsymbol{X}_{\leq i}^{l-1\intercal}}{\sqrt{d_k}}\right),$$

$$\text{Attn}^{l,h}(\boldsymbol{X}_{\leq i}^{l-1}) = \sum_{j\leq i} a_{i,j}^{l,h}\,\overbrace{\boldsymbol{x}_j^{l-1}\boldsymbol{W}_V^{l,h}}^{\text{Value vector}}\,\boldsymbol{W}_O^{l,h}$$

$$= \sum_{j\leq i} a_{i,j}^{l,h}\boldsymbol{x}_j^{l-1}\boldsymbol{W}_{OV}^{l,h},$$

Ferrando et al., A Primer on the Inner Workings of Transformer-based Language Models

OV Circuit

Value vector

$$\mathrm{Attn}^{l,h}(\boldsymbol{X}_{\leq i}^{l-1}) = \sum_{j \leq i} a_{i,j}^{l,h} \underbrace{\boldsymbol{x}_j^{l-1} \boldsymbol{W}_V^{l,h}}_{} \boldsymbol{W}_O^{l,h}$$

$$= \sum_{j \leq i} a_{i,j}^{l,h} \boldsymbol{x}_j^{l-1} \boldsymbol{W}_{OV}^{l,h},$$

OV (output-value) circuit: $W_{OV}^{l,h} = W_V^{l,h} W_O^{l,h}$

- OV circuits are responsible for writing to the **residual stream**.

Ferrando et al., A Primer on the Inner Workings of Transformer-based Language Models

Attention Block Output

The attention block output is the sum of individual attention heads, which is subsequently added back into the residual stream.

$$\mathrm{Attn}^{l,h}(\boldsymbol{X}_{\leq i}^{l-1}) = \sum_{j \leq i} a_{i,j}^{l,h} \underbrace{\boldsymbol{x}_j^{l-1} \boldsymbol{W}_V^{l,h}}_{\text{Value vector}} \boldsymbol{W}_O^{l,h} = \sum_{j \leq i} a_{i,j}^{l,h} \boldsymbol{x}_j^{l-1} \boldsymbol{W}_{OV}^{l,h},$$

$$\boldsymbol{a}_i^{l,h} = \mathrm{softmax}\left(\frac{\underbrace{\boldsymbol{x}_i^{l-1} \boldsymbol{W}_Q^{l,h}}_{\text{Query vector}} \underbrace{(\boldsymbol{X}_{\leq i}^{l-1} \boldsymbol{W}_K^{l,h})^{\intercal}}_{\text{Key vector}}}{\sqrt{d_k}} \right) = \mathrm{softmax}\left(\frac{\boldsymbol{x}_i^{l-1} \boldsymbol{W}_{QK}^{h} \boldsymbol{X}_{\leq i}^{l-1\intercal}}{\sqrt{d_k}} \right),$$

$$\boxed{\begin{aligned} \mathrm{Attn}^{l}(\boldsymbol{X}_{\leq i}^{l-1}) &= \sum_{h=1}^{H} \mathrm{Attn}^{l,h}(\boldsymbol{X}_{\leq i}^{l-1}) \\ \boldsymbol{x}_i^{\mathrm{mid},l} &= \boldsymbol{x}_i^{l-1} + \mathrm{Attn}^{l}(\boldsymbol{X}_{\leq i}^{l-1}). \end{aligned}}$$

Ferrando et al., A Primer on the Inner Workings of Transformer-based Language Models

Feed-Forward Network (FFN)

$$\boldsymbol{x}_i^{\text{mid},l} = \boldsymbol{x}_i^{l-1} + \text{Attn}^l(\boldsymbol{X}_{\leq i}^{l-1}).$$

$$\boldsymbol{W}_{\text{in}}^l \in \mathbb{R}^{d \times d_{\text{ffn}}}$$

$$\boldsymbol{W}_{\text{out}}^l \in \mathbb{R}^{d_{\text{ffn}} \times d}$$

$$\text{FFN}^l(\boldsymbol{x}_i^{\text{mid},l}) = g(\boldsymbol{x}_i^{\text{mid},l}\boldsymbol{W}_{\text{in}}^l)\boldsymbol{W}_{\text{out}}^l.$$

$$\boldsymbol{x}_i^l = \boldsymbol{x}_i^{\text{mid},l} + \text{FFN}^l(\boldsymbol{x}_i^{\text{mid},l}).$$

- W_{in}^l reads from the residual stream state $x_i^{mid,l}$.
- Its result is passed through an element-wise non-linear activation function g, producing the neuron activations.
- These get transformed by W_{out}^l to produce the output $FFN^l(x_i^{mid,l})$, which is then added back to the residual stream

Ferrando et al., A Primer on the Inner Workings of Transformer-based Language Models

Prediction as a Sum of Component Outputs

- Prediction head of a Transformer consists of an unembedding matrix: $\boldsymbol{W}_U \in \mathbb{R}^{d \times |\mathcal{V}|}$

We can rearrange the traditional forward pass formulation to separate the **contribution of each model component to the output logits**:

$$
\begin{aligned}
f(\mathbf{x}) &= \boldsymbol{x}_n^L \boldsymbol{W}_U \\
&= \Big(\sum_{l=1}^{L} \sum_{h=1}^{H} \mathrm{Attn}^{l,h}(\boldsymbol{X}_{\leq n}^{l-1}) + \sum_{l=1}^{L} \mathrm{FFN}^{l}(\boldsymbol{x}_n^{\mathrm{mid},l}) + \boldsymbol{x}_n \Big) \boldsymbol{W}_U \\
&= \underbrace{\sum_{l=1}^{L} \sum_{h=1}^{H} \mathrm{Attn}^{l,h}(\boldsymbol{X}_{\leq n}^{l-1}) \boldsymbol{W}_U}_{\text{Attention head logits update}} + \sum_{l=1}^{L} \underbrace{\mathrm{FFN}^{l}(\boldsymbol{x}_n^{\mathrm{mid},l}) \boldsymbol{W}_U}_{\text{FFN logits update}} + \boldsymbol{x}_n \boldsymbol{W}_U .
\end{aligned}
$$

Ferrando et al., A Primer on the Inner Workings of Transformer-based Language Models

Prediction as an Ensemble of Shallow Networks

- Residual networks work as ensembles of shallow networks, where **each subnetwork defines a path in the computational graph**.

Consider a two-layer attention-only Transformer, where each attention head is composed just by an OV matrix:

$$f(\boldsymbol{x}) = \boldsymbol{x}^1 + \boldsymbol{W}_{OV}^2(\boldsymbol{x}^1), \text{ with } \boldsymbol{x}^1 = \boldsymbol{x} + \boldsymbol{W}_{OV}^1(\boldsymbol{x})$$

We can decompose the forward pass as:

Direct path

Full OV circuits

$$f(\boldsymbol{x}) = \boldsymbol{x}\boldsymbol{W}_U + \boldsymbol{x}\boldsymbol{W}_{OV}^1\boldsymbol{W}_U + \boldsymbol{x}\boldsymbol{W}_{OV}^1\boldsymbol{W}_{OV}^2\boldsymbol{W}_U + \boldsymbol{x}\boldsymbol{W}_{OV}^2\boldsymbol{W}_U.$$

Virtual attention heads (V-composition)

Ferrando et al., A Primer on the Inner Workings of Transformer-based Language Models

Prediction as an Ensemble of Shallow Networks

Direct path

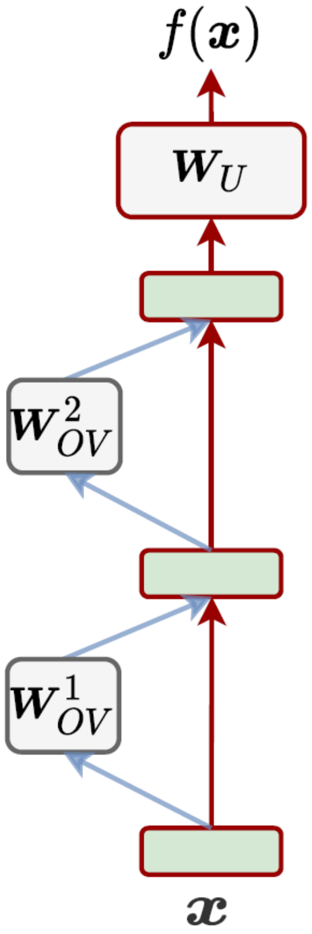
Full OV circuits

$$f(\boldsymbol{x}) = \boldsymbol{x}\boldsymbol{W}_U + \boldsymbol{x}\boldsymbol{W}_{OV}^{1}\boldsymbol{W}_U + \boldsymbol{x}\boldsymbol{W}_{OV}^{1}\boldsymbol{W}_{OV}^{2}\boldsymbol{W}_U + \boldsymbol{x}\boldsymbol{W}_{OV}^{2}\boldsymbol{W}_U\,.$$

Virtual attention heads (V-compositior

- This term links the input embedding to the unembedding matrix and is referred to as the **direct path**

- It shows the contribution of the input embedding towards the output logit of the next token to be predicted.



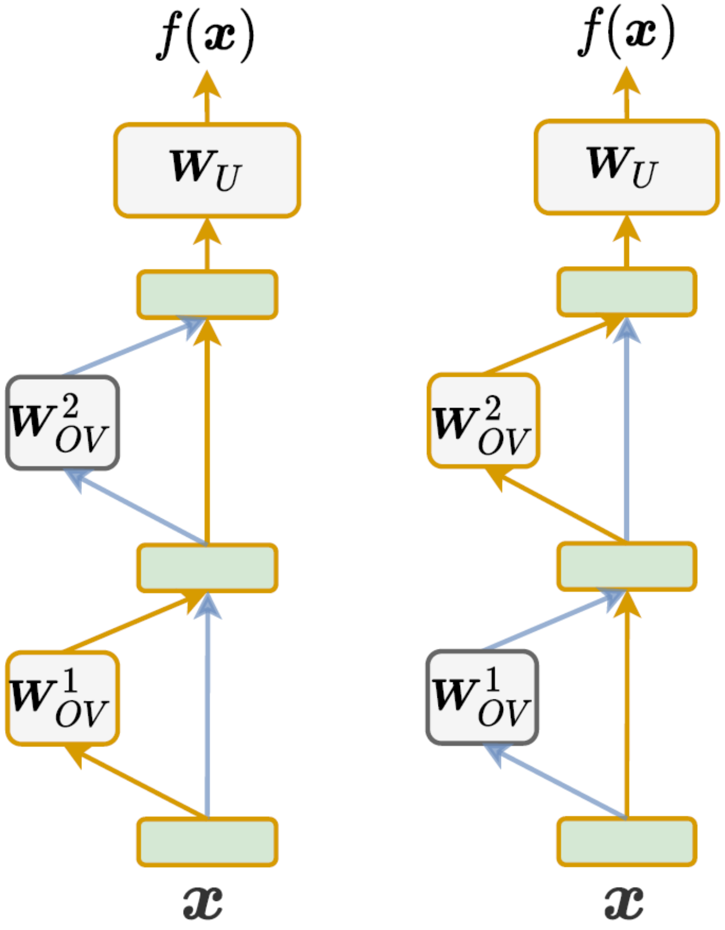
Ferrando et al., A Primer on the Inner Workings of Transformer-based Language Models

Prediction as an Ensemble of Shallow Networks

Direct path | Full OV circuits

$$f(\boldsymbol{x}) = \overbrace{\boldsymbol{x}\boldsymbol{W}_U}^{\text{Direct path}} + \boldsymbol{x}\boldsymbol{W}^1_{OV}\boldsymbol{W}_U + \underbrace{\boldsymbol{x}\boldsymbol{W}^1_{OV}\boldsymbol{W}^2_{OV}\boldsymbol{W}_U}_{\text{Virtual attention heads (V-composition)}} + \boldsymbol{x}\boldsymbol{W}^2_{OV}\boldsymbol{W}_U .$$

- These terms depicts paths traversing a single OV circuit, and are named **full OV circuits**
- They show the contribution of each OV circuit towards the output logit of the next token to be predicted.



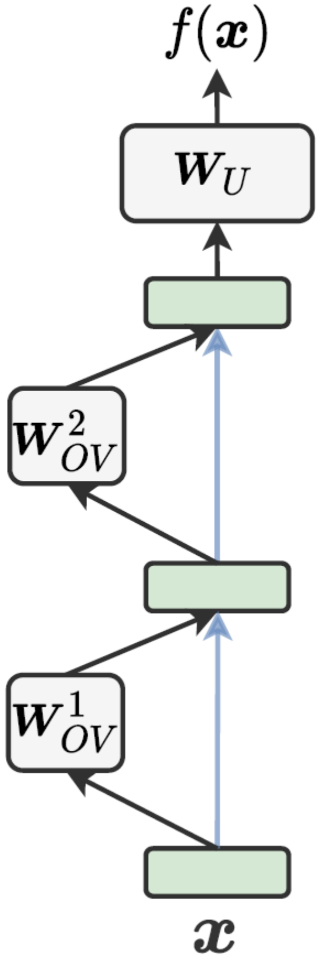
Ferrando et al., A Primer on the Inner Workings of Transformer-based Language Models

Prediction as an Ensemble of Shallow Networks

Direct path

Full OV circuits

$$f(\boldsymbol{x}) = \boldsymbol{x}\boldsymbol{W}_U + \boldsymbol{x}\boldsymbol{W}^1_{OV}\boldsymbol{W}_U + \underbrace{\boldsymbol{x}\boldsymbol{W}^1_{OV}\boldsymbol{W}^2_{OV}\boldsymbol{W}_U}_{\text{Virtual attention heads (V-composition)}} + \boldsymbol{x}\boldsymbol{W}^2_{OV}\boldsymbol{W}_U \, .$$



- This term depicts the path involving both attention heads, and is referred to as **virtual attention heads doing V-composition**
- This is called 'composition' since the sequential writing and reading of the two heads is seen as OV matrices composing together.
  - The amount of composition can be measured as:

$$\|W^1_{OV}W^2_{OV}\|_F \Big/ \|W^1_{OV}\|_F\|W^2_{OV}\|_F$$

Ferrando et al., A Primer on the Inner Workings of Transformer-based Language Models

Prediction as an Ensemble of Shallow Networks

- In full Transformer models, Q-composition and K-composition, i.e. compositions of W_Q and W_K with the W_{OV} output of previous layers, can also be found.

- Such decomposition enables us to localize the inputs or model components responsible for a particular prediction.

Why Do We Need Such a Formulation?

- By decomposing the Transformer into simpler components like the *query-key circuit* W_{QK} and the *output-value circuit* W_{OV}, we can better understand the information flow within Transformer-based LLMs.
- This formulation reveals how each layer incrementally transforms token representations.
  - Also shows how attention heads and feedforward networks contribute to language modeling.
- Breaking down the contributions of individual circuits allows us to interpret which aspects of the model influence specific predictions.

Thus, through this formulation, the behavior of attention heads, the interaction between tokens, and the role of the residual stream can be explored more clearly.