System for Artificial Intelligence

Introduction to LLMs and Optimizations

Siyuan Feng
Shanghai Innovation Institute

OUTLINE

01 Sequence Prediction

02 Transformers and Self-Attention

03 Recursive Attention

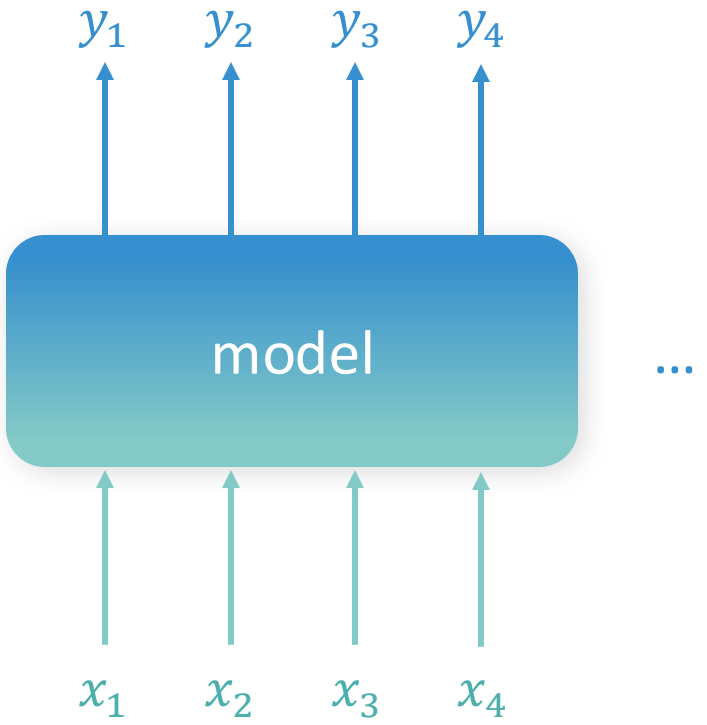
01

Sequence Prediction

Sequence Prediction

上海创智学院
Shanghai Innovation Institute

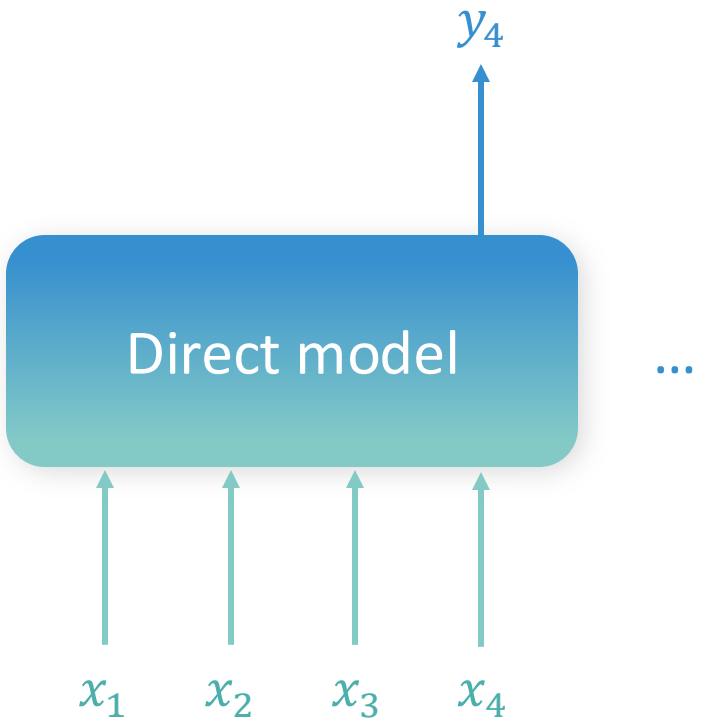
- Take a set of input sequence, predict the output sequence



- Predict each output based on history: $y_t = f_\theta(x_{1:t})$
- There are many ways to build up the predictive model

“Direct Prediction”

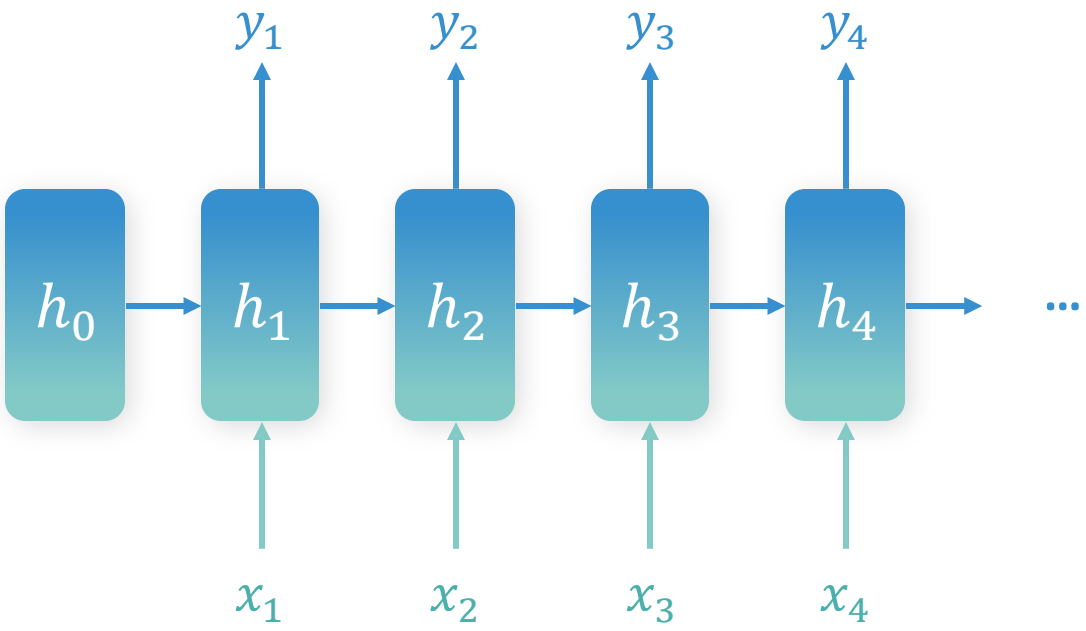
- One approach is we can do “direct prediction”



- Challenge: different size inputs.

RNN Approach

- Try to maintain a "latent state" that is derived from history.



- Challenge: The information is carried only through h_t

02

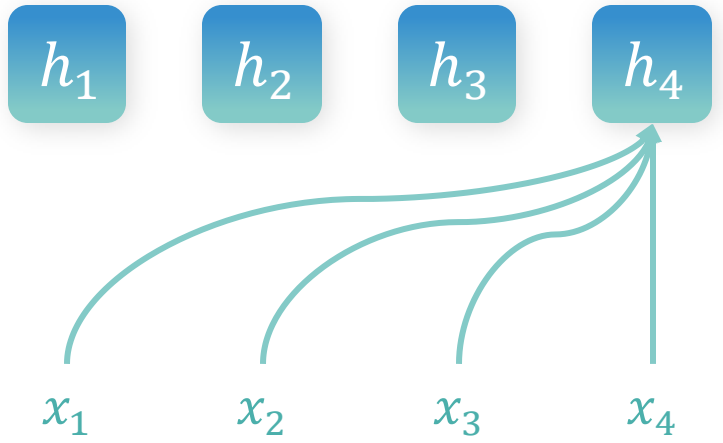
Transformers and Self-Attention

"Attention" Mechanism

上海创智学院
Shanghai Innovation Institute

- Generally refers to the approach that weighted combine individual states

Attention Output



Hidden states from previous layer

$$h_t = \sum_{i=1}^{t} s_i x_i$$

- Intuitively s_i is "attention score" that computes how relevant the position i's input is to this current hidden output There are different methods to decide how attention score is being computed

Self-Attention Operation

上海创智学院
Shanghai Innovation Institute

- Self attention refers to a particular form of attention mechanism. Given three inputs $Q, K, V \in \mathbb{R}^{T \times d}$ ("queries", "keys", "values")

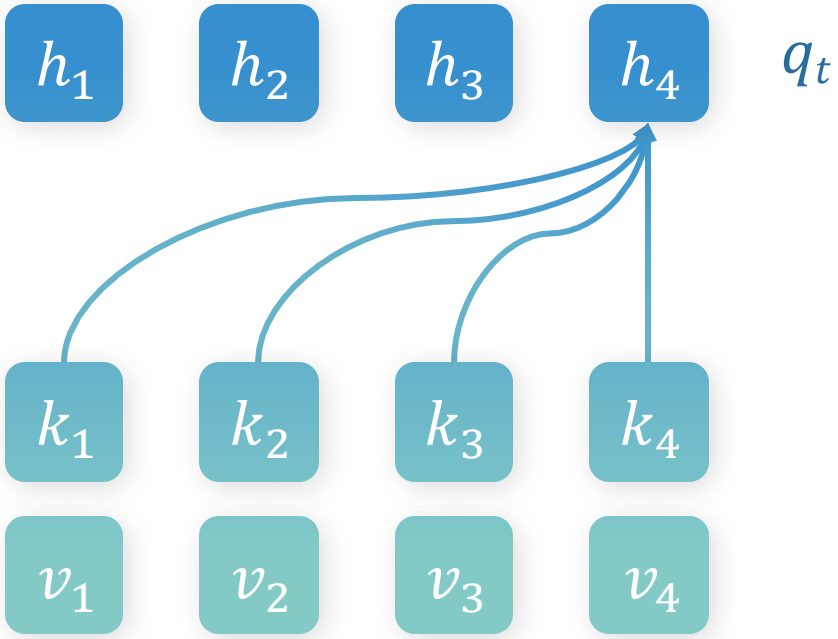
- Define the self-attention as:

$$\text{SelfAttention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{d^{1/2}}\right)V$$

A Closer Look at Self-Attention

上海创智学院
Shanghai Innovation Institute

- Use q_t, k_t, v_t to refers to row t of the K matrix

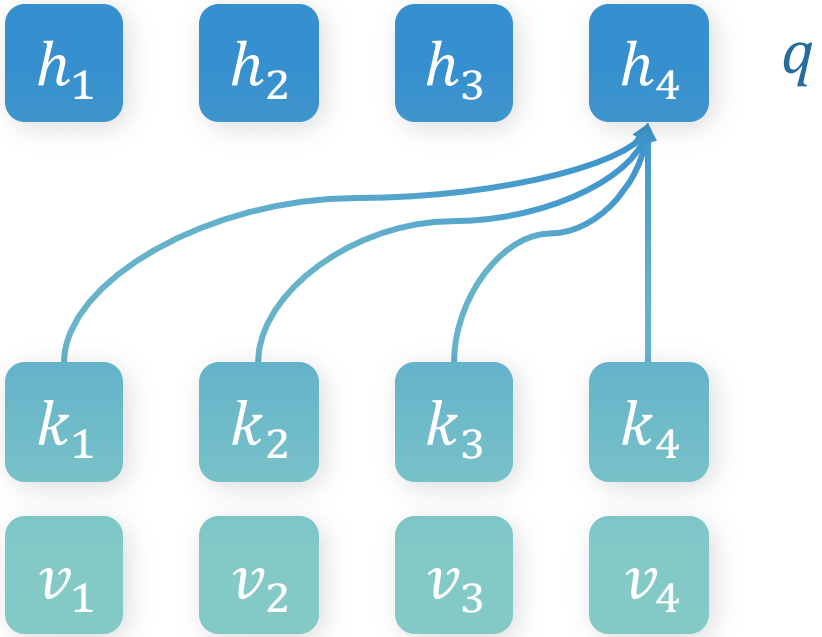


- Ask the following question:
- How to compute the output h_t, based on q_t, K, V one timestep t
- To keep presentation simple, we will drop suffix t and just use q to refer to q_t in next few slide

A Closer Look at Self-Attention

上海创智学院 Shanghai Innovation Institute

- Use q_t, k_t, v_t to refers to row t of the K matrix



- Conceptually, we compute the output in the following two steps:
- Pre-softmax "attention score"

$$s_i = \frac{1}{\sqrt{d}} q k_i^T$$

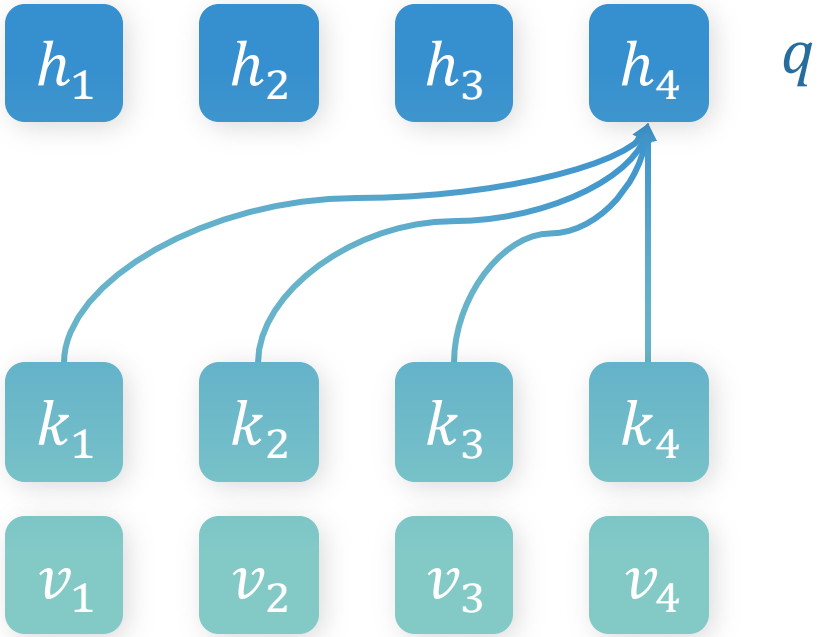
- Weighed average via softmax

$$h = \sum_i \text{softmax}(s)_i v_i = \frac{\sum_i \exp(s_i)\, v_i}{\sum_j \exp(sj)}$$

- Intuition: s_i computes the relevance of k_i to the query q,
- then we do weighted sum of values proportional to their relevance

Comparing the Matrix Form and the Decomposed Form

- Use q_t, k_t, v_t to refers to row t of the K matrix



$$\text{SelfAttention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{d^{1/2}}\right)V$$

- Pre-softmax "attention score"

$$s_{ti} = \frac{1}{\sqrt{d}} q k_i^T$$

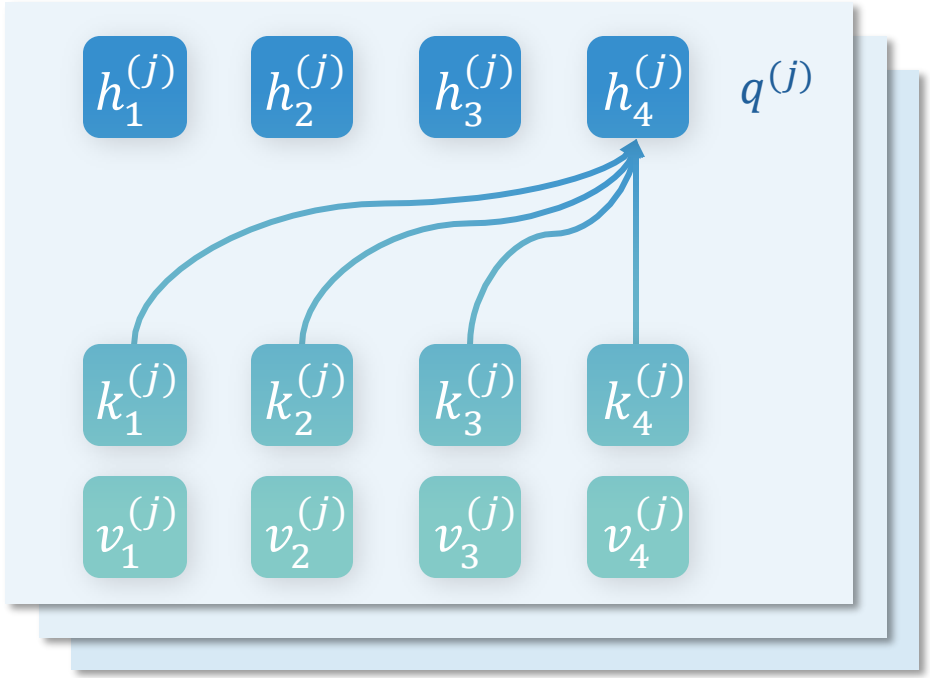
- Weighed average via softmax

$$h_t = \sum_i \text{softmax}(S_{t,:})_i v_i = \text{softmax}(S_{t,:})V$$

- Intuition: s_i computes the relevance of k_i to the query q,
- then we do weighted sum of values proportional to their relevance

Multi-Head Attention

- Have multiple "attention heads" $Q^{(j)}$, $K^{(j)}$, $V^{(j)}$ denotes j-th attention head



- Apply self-attention in each attention head

$$\text{SelfAttention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{d^{1/2}}\right)V$$

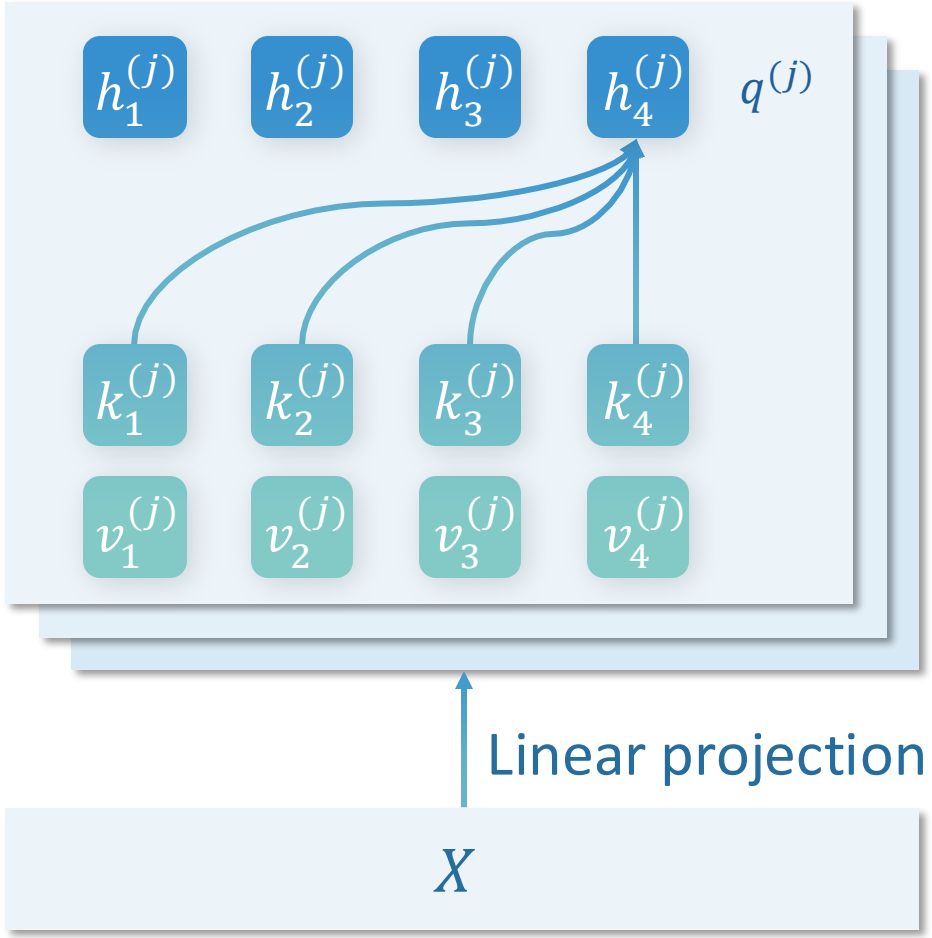
- Concatenate all output heads together as output

- Each head can correspond to different kind of information.
- Sometimes we can share the heads: GQA(group query attention) all heads share K, V but have different Q

How to get Q K V?

上海创智学院
Shanghai Innovation Institute

- Obtain Q, K, V from previous layer's hidden state X by linear projection



$$Q = XW_Q$$
$$K = XW_K$$
$$V = XW_V$$

- Can compute all heads and Q, K, V together then
- split/reshape out into individual Q, K, V with multiple heads

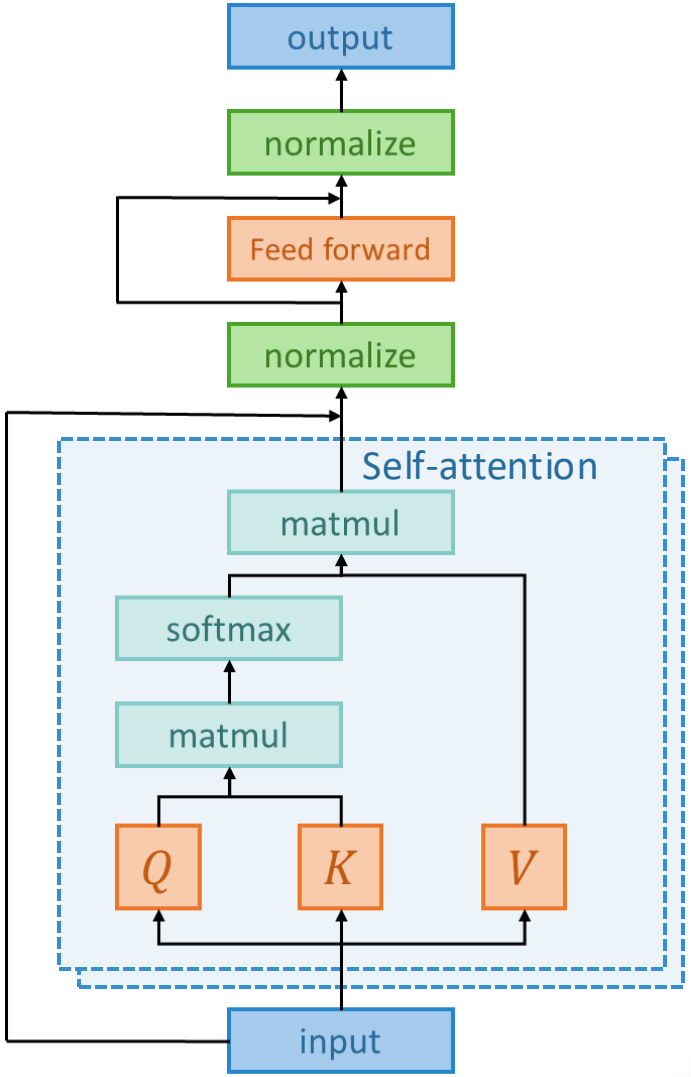
Transformer Block

上海创智学院
Shanghai Innovation Institute

- A typical transformer block

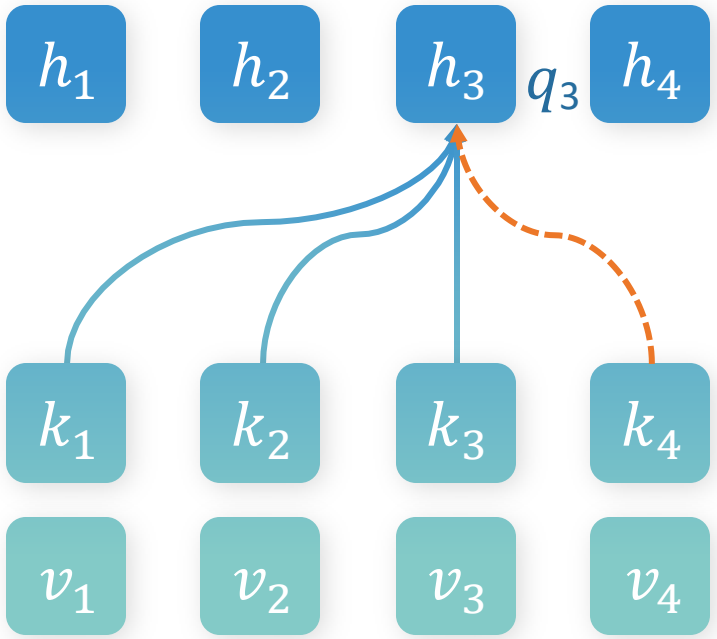
$$Z = \text{SelfAttention}(XW_K, XW_Q, XW_V)$$
$$Z = \text{LayerNorm}(X + Z)$$
$$H = \text{LayerNorm}(\text{ReLU}(ZW_1)W_2 + Z)$$

- (multi-head) self-attention, followed by a linear layer and ReLU and some additional residual connections and normalization



Masked Self-Attention

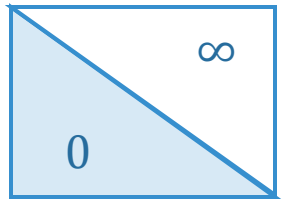
- In the matrix form, we are computing weighted average over all inputs



- In auto regressive models, usually it is good to maintain casual relation, and only attend to some of the inputs (e.g. skip the red dashed edge on the left). We can add "attention mask"

$$\text{MaskedSelfAttention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{d^{1/2}} - M\right)V$$

$$M_{ij} = \begin{cases} \infty, j > i \\ 0, j \le i \end{cases}$$



- Only attend to previous inputs. Depending on input structure and model, attention mask can change.
- We can also simply skip the computation that are masked out if there is a special implementation to do so

Discussions

上海创智学院
Shanghai Innovation Institute

- What are the advantages of transformers versus RNNs
- What are the disadvantages
- What are other possible ways to apply attention mask

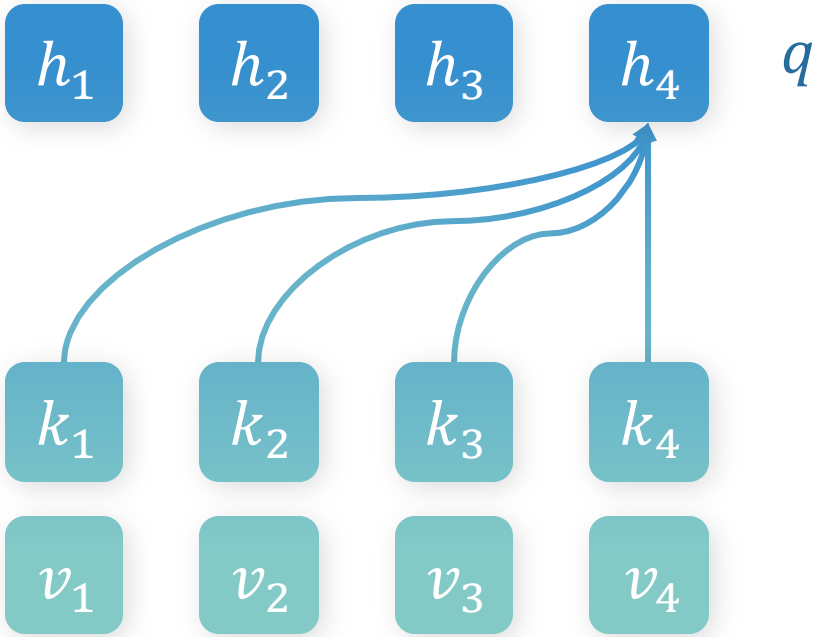
03

Recursive Attention

Self-Attention Recap

上海创智学院
Shanghai Innovation Institute

- Use q_t, k_t, v_t to refers to row t of the K matrix



- Conceptually, we compute the output in the following two steps:
- Pre-softmax "attention score"

$$s_i = \frac{1}{\sqrt{d}} q k_i^T$$

- Weighed average via softmax

$$h = \sum_i \text{softmax}(s)_i v_i = \frac{\sum_i \exp(s_i)\, v_i}{\sum_j \exp(sj)}$$

Generalizing Attention Score and Value Vector

上海创智学院
Shanghai Innovation Institute

- Pre-softmax "attention score" $s_i = \frac{1}{\sqrt{d}} q k_i^T$
- Define the following "Attention weight" for an index set I

$$s(I) = \log(\sum_{i \in I} \exp(s_i))$$

- Let us generalize the value vector v for index set I

$$\mathrm{v}(I) = \sum_{i \in I} softmax(s)_i v_i = \frac{\sum_{i \in I} \exp(s_i)\, v_i}{\exp(s(I))}$$

- Reference: flashinfer.ai

Generalizing Attention Score and Value Vector

上海创智学院
Shanghai Innovation Institute

- Pre-softmax "attention score" $s_i = \frac{1}{\sqrt{d}} q k_i^T$

$$s(I) = \log\left(\sum_{i \in I} \exp(s_i)\right), \mathrm{v}(I) = \sum_{i \in I} \mathrm{softmax}(s)_i v_i = \frac{\sum_{i \in I} \exp(s_i) v_i}{\exp(s(I))}$$

- When index set $I = \{i\}$, $\mathrm{s}(\{i\}) = s_i$, $\mathrm{v}(\{i\}) = v_i$
- When index set $I = \{1,2 \ldots t\}$, $\mathrm{v}(I)$ is the final output of the attention

Recursive Attention

上海创智学院
Shanghai Innovation Institute

$$s(I) = \log(\sum_{i \in I} \exp(s_i)) \, , \mathrm{v}(I) = \sum_{i \in I} \mathrm{softmax}(s)_i v_i = \frac{\sum_{i \in I} \exp(s_i) v_i}{\exp(s(I))}$$

- For any partition $\{I_j\}$ of I such that $I = \mathrm{U}_{j=1}^{n} I_j$, the following relation holds
- $\mathrm{s}(\mathrm{U}_{j=1}^{n} I_j) = \log\left(\sum_j \exp\left(s(I_j)\right)\right), \mathrm{v}(\mathrm{U}_{j=1}^{n} I_j) = \sum_j \mathrm{softmax}([s(I_1), s(I_2) \ldots])_j \, v(I_j)$
- We can use the same rule to recursively combine the vector and "attention score" of any subsets of indices
- When we obtain the value vector of all indices, that becomes the attention output

Discussions: Recursive Attention

上海创智学院
Shanghai Innovation Institute

$$s(I) = \log(\sum_{i \in I} \exp(s_i)) \, , \mathrm{v}(I) = \sum_{i \in I} \mathrm{softmax}(s)_i v_i = \frac{\sum_{i \in I} \exp(s_i) v_i}{\exp(s(I))}$$

- For any partition $\{I_j\}$ of I such that $I = \mathrm{U}_{j=1}^{n} I_j$, the following relation holds
- $\mathrm{s}(\mathrm{U}_{j=1}^{n} I_j) = \log\left(\sum_j \exp\left(s(I_j)\right)\right), \mathrm{v}(\mathrm{U}_{j=1}^{n} I_j) = \sum_j \mathrm{softmax}([s(I_1), s(I_2) \ldots])_j \, v(I_j)$
- Attention computation is **communicative** and **associative** and can be done in a divide-and-conquer fashion. An important property for a lot of system optimizations
- Discussion: what can we do with this property?

Acknowledgement

The development of this course, including its structure, content, and accompanying presentation slides, has been significantly influenced and inspired by the excellent work of instructors and institutions who have shared their materials openly. We wish to extend our sincere acknowledgement and gratitude to the following courses, which served as invaluable references and a source of pedagogical inspiration:

- Machine Learning Systems[15-442/15-642], by **Tianqi Chen** and **Zhihao Jia** at **CMU**.
- Advanced Topics in Machine Learning (Systems)[CS6216], by **Yao Lu** at **NUS**

While these materials provided a foundational blueprint and a wealth of insightful examples, all content herein has been adapted, modified, and curated to meet the specific learning objectives of our curriculum. Any errors, omissions, or shortcomings found in these course materials are entirely our own responsibility. We are profoundly grateful for the contributions of the educators listed above, whose dedication to teaching and knowledge-sharing has made the creation of this course possible.

System for Artificial Intelligence

Thanks

Siyuan Feng
Shanghai Innovation Institute