A Mathematical View of Attention Models in Deep Learning

Shuiwang Ji, Yaochen Xie
Department of Computer Science & Engineering
Texas A&M University

Attention Model

1. Given a set of n query vectors $q_1, q_2, \cdots, q_n \in \mathbb{R}^d$, m key vectors $k_1, k_2, \cdots, k_m \in \mathbb{R}^d$, and m value vectors $v_1, v_2, \cdots, v_m \in \mathbb{R}^p$, the attention mechanism computes a set of output vectors $o_1, o_2, \cdots, o_n \in \mathbb{R}^q$ by linearly combining the g-transformed value vectors $g(v_i) \in \mathbb{R}^q$ using the relations between the corresponding query vector and each key vector as coefficients.
2. Formally,

$$o_j = \frac{1}{C} \sum_{i=1}^{m} f(q_j, k_i) g(v_i), \tag{1}$$

where $f(q_j, k_i)$ characterizes the relation (*e.g.*, similarity) between q_j and k_i, $g(\cdot)$ is commonly a linear transformation as $g(v_i) = \boldsymbol{W}_v v_i \in \mathbb{R}^q$, where $\boldsymbol{W}_v \in \mathbb{R}^{q \times p}$, and $C = \sum_{i=1}^{m} f(q_j, k_i)$ is a normalization factor.

Attention Model

1. A commonly used similarity function is the embedded Gaussian, defined as $f(q_j, k_i) = \exp\left(\theta(q_j)^T \phi(k_i)\right)$, where $\theta(\cdot)$ and $\phi(\cdot)$ are commonly linear transformations as $\theta(q_j) = \boldsymbol{W}_q q_j$ and $\phi(k_i) = \boldsymbol{W}_k k_i$.
2. Note that if we treat the value vectors as inputs, each output vector o_j is dependent on all input vectors. When the embedded Gaussian similarity and linear transformation are used, these computations can be expressed succinctly in matrix form as

$$\boldsymbol{O} = \boldsymbol{W}_v \boldsymbol{V} \times \operatorname{softmax}\left((\boldsymbol{W}_k \boldsymbol{K})^T \boldsymbol{W}_q \boldsymbol{Q}\right), \tag{2}$$

   where $\boldsymbol{Q} = [q_1, q_2, \cdots, q_n] \in \mathbb{R}^{d \times n}$, $\boldsymbol{K} = [k_1, k_2, \cdots, k_m] \in \mathbb{R}^{d \times m}$, $\boldsymbol{V} = [v_1, v_2, \cdots, v_m] \in \mathbb{R}^{p \times m}$, $\boldsymbol{O} = [o_1, o_2, \cdots, o_n] \in \mathbb{R}^{q \times n}$, and softmax$(\cdot)$ computes a normalized version of the input matrix, where each column is normalized using the softmax function to sum to one.
3. Note that the number of output vectors is equal to the number of query vectors. In self-attention, we have $\boldsymbol{Q} = \boldsymbol{K} = \boldsymbol{V}$.

Attention Model

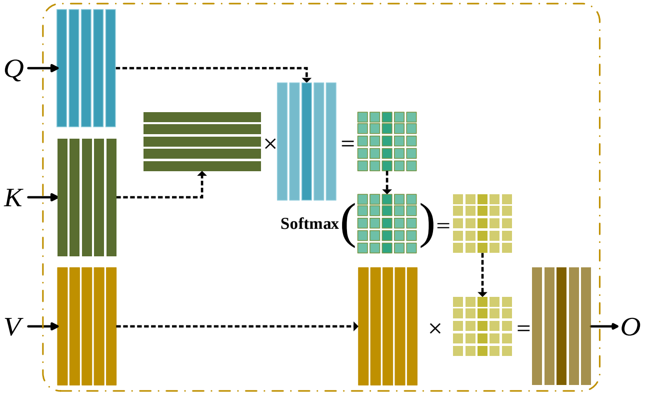

Figure: An illustration of the attention operator. Here, $\times$ denotes matrix multiplication, and Softmax$(\cdot)$ is the column-wise softmax operator. $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ are input matrices. A similarity score is computed between each query vector as a column of $\boldsymbol{Q}$ and each key vector as a column in $\boldsymbol{K}$. Softmax$(\cdot)$ normalizes these scores and makes them sum to 1. Multiplication between normalized scores and the matrix $\boldsymbol{V}$ yields the corresponding output vector.

Self-Attention

1. We introduce two specific types of the attention mechanism. The different types of attention mainly differ in how the $\boldsymbol{Q}, \boldsymbol{K}$ and $\boldsymbol{V}$ matrices are obtained, and their computations of the output given $\boldsymbol{Q}, \boldsymbol{K}$ and $\boldsymbol{V}$ are the same.
2. The self-attention captures the intra-correlation of a given input matrix $\boldsymbol{X} = [x_1, x_2, \cdots, x_n] \in \mathbb{R}^{d \times n}$. In the self-attention, we let $\boldsymbol{Q} = \boldsymbol{K} = \boldsymbol{V} = \boldsymbol{X}$. The attention operator then becomes

$$\boldsymbol{O} = \boldsymbol{W}_v \boldsymbol{X} \times \text{softmax}\left((\boldsymbol{W}_k \boldsymbol{X})^T \boldsymbol{W}_q \boldsymbol{X} \right), \tag{3}$$

3. In this case, the number of output vectors is determined by the the number of input vectors.

Attention with Learnable Query

1. The attention with learnable query is a common variation of the self-attention, where we still have $\boldsymbol{K} = \boldsymbol{V} = \boldsymbol{X}$. However, the query $\boldsymbol{Q} \in \mathbb{R}^{d \times n}$ is neither given as input nor dependent on the input.
2. Instead, we directly learn the $\boldsymbol{Q}$ matrix as trainable variables. Thus we have
$$\boldsymbol{O} = \boldsymbol{W}_v \boldsymbol{X} \times \text{softmax}\left((\boldsymbol{W}_k \boldsymbol{X})^T \boldsymbol{Q} \right). \tag{4}$$
3. Such type of attention mechanism is commonly used in NLP and graph neural networks (GNNs). It allows the networks to capture common features from all input instances during training since the query is independent of the input and is shared by all input instances.
4. Note that since the number of output vectors is determined by the number of query vectors, the output size of the attention mechanism with learned query is fixed and is no longer flexibly related to the input.

Multi-Head Attention

1. The multi-head attention consists of multiple attention operators with different groups of weight matrices.
2. Formally, for the i-th head in the M-head attention, we compute its output $\boldsymbol{H}_i$ by

$$\boldsymbol{H}_i = \boldsymbol{W}_v^{(i)} \boldsymbol{V} \times \mathrm{softmax}\left((\boldsymbol{W}_k^{(i)} \boldsymbol{K})^T \boldsymbol{W}_q^{(i)} \boldsymbol{Q} \right) \in \mathbb{R}^{q_i \times n}, \qquad (5)$$

   where $W_q^{(i)}, W_k^{(i)}$ and $W_v^{(i)}$ determine the similarity function f_i for the i-th head.
3. The final output of the multi-head attention is then computed as

$$\boldsymbol{O} = \boldsymbol{W}_o \begin{bmatrix} \boldsymbol{H}_1 \\ \vdots \\ \boldsymbol{H}_M \end{bmatrix} \in \mathbb{R}^{q \times n}, \qquad (6)$$

   where $\boldsymbol{W}_o \in \mathbb{R}^{q \times (\sum_i q_i)}$ is the learned weight matrix that projects the concatenated heads into the desired dimension.
4. The multi-head attention allows each head to attend different locations based on the similarity in different representation subspaces.

Attention for Higher Order Data

1. The attention mechanism was originally developed in natural language processing to process 1-D data.
2. It has been extended to deal with 2-D images and 3-D video data recently.
3. When deal with 2-D data, the inputs to the attention operator can be represented as 3-D tensors $\mathfrak{Q} \in \mathbb{R}^{h \times w \times c}$, $\mathfrak{K} \in \mathbb{R}^{h \times w \times c}$, and $\mathcal{V} \in \mathbb{R}^{h \times w \times c}$, where h, w, and c represent the height, width, and number of channels, respectively. Note that for notational simplicity, we have assumed the three tensors having the same size.

Attention for Higher Order Data

1. These tensors are first unfolded into matrices along mode-3, resulting in $\boldsymbol{Q}_{(3)}, \boldsymbol{K}_{(3)}, \boldsymbol{V}_{(3)} \in \mathbb{R}^{c \times hw}$.
2. Columns of these matrices are the mode-3 fibers of the corresponding tensors.
3. These matrices are used to compute output vectors as in regular attention described above. The output vectors are then folded back to a 3-D tensor $\mathfrak{O} \in \mathbb{R}^{h \times w \times q}$ by treating them as mode-3 fibers of $\mathfrak{O}$.
4. Note that the height and width of $\mathfrak{O}$ are equal to those of $\mathfrak{Q}$. That is, we can obtain an output with larger/smaller spatial size by providing an input $\mathfrak{Q}$ of correspondingly larger/smaller spatial size.
5. Again, we have $\mathfrak{Q} = \mathfrak{K} = \mathcal{V}$ in self-attention.

Attention for Higher Order Data

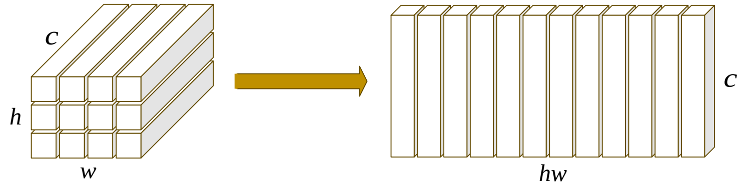

Figure: Conversion of a third-order tensor into a matrix by unfolding along mode-3. In this example, a $h \times w \times c$ tensor is unfolded into a $c \times hw$ matrix.

Invariance and Equivariance

Spatial permutation invariance and equivariance are two properties required by different tasks.

Definition

Consider an image or feature map $\boldsymbol{X} \in \mathbb{R}^{d \times n}$, where n denotes the spatial dimension and d denotes the number of features. Let π denotes a permutation of n elements. We call a transformation $\mathcal{T}_\pi : \mathbb{R}^{d \times n} \to \mathbb{R}^{d \times n}$ a spatial permutation if $\mathcal{T}_\pi(\boldsymbol{X}) = \boldsymbol{X} P_\pi$, where $P_\pi \in \mathbb{R}^{n \times n}$ denotes the permutation matrix associated with π, defined as $P_\pi = \left[\boldsymbol{e}_{\pi(1)}, \boldsymbol{e}_{\pi(2)}, \cdots, \boldsymbol{e}_{\pi(n)}\right]$, and $\boldsymbol{e}_i$ is a one-hot vector of length n with its i-th element being 1.

Definition

We call an operator $A : \mathbb{R}^{d \times n} \to \mathbb{R}^{d \times n}$ to be spatially permutation equivariant if $\mathcal{T}_\pi(A(\boldsymbol{X})) = A(\mathcal{T}_\pi(\boldsymbol{X}))$ for any X and any spatial permutation $\mathcal{T}_\pi$. In addition, an operator $A : \mathbb{R}^{d \times n} \to \mathbb{R}^{d \times n}$ is spatially permutation invariant if $A(\mathcal{T}_\pi(\boldsymbol{X})) = A(\boldsymbol{X})$ for any X and any spatial permutation $\mathcal{T}_\pi$.

Invariance and Equivariance

1. In the image domain, the (spatial) permutation invariance is essential when we perform the image-level prediction such as image classification, where we usually expect the prediction to remain the same as the input image is rotated or flipped.
2. On the other hand, the permutation equivariance is essential in the pixel-level prediction such as image segmentation or style translation where we expect the prediction to rotate or flip correspondingly to the rotation or flipping of the input image.
3. We now show the corresponding property of self-attention and attention with learned query. For simplicity, we only consider the single-head attention.

Invariance and Equivariance of Attention

Theorem

A self-attention operator A_s is permutation equivariant while an attention operator with learned query $A_{\boldsymbol{Q}}$ is permutation invariant. In particular, letting $\boldsymbol{X}$ denote the input matrix and $\mathcal{T}$ denotes any spatial permutation, we have

$$A_s(\mathcal{T}_\pi(\boldsymbol{X})) = \mathcal{T}_\pi(A_s(\boldsymbol{X})),$$

and

$$A_{\boldsymbol{Q}}(\mathcal{T}_\pi(\boldsymbol{X})) = A_{\boldsymbol{Q}}(\boldsymbol{X}).$$

Proof

Proof.

When applying a spatial permutation $\mathcal{T}_\pi$ to the input $\boldsymbol{X}$ of a self-attention operator A_s, we have

$$\begin{aligned} A_s(\mathcal{T}_\pi(\boldsymbol{X})) &= \boldsymbol{W}_v\mathcal{T}_\pi(\boldsymbol{X}) \cdot \mathrm{softmax}\left((\boldsymbol{W}_k\mathcal{T}_\pi(\boldsymbol{X}))^T \cdot \boldsymbol{W}_v\mathcal{T}_\pi(\boldsymbol{X})\right) \\ &= \boldsymbol{W}_v\boldsymbol{X}P_\pi \cdot \mathrm{softmax}\left((\boldsymbol{W}_k\boldsymbol{X}P_\pi)^T \cdot \boldsymbol{W}_q\boldsymbol{X}P_\pi\right) \\ &= \boldsymbol{W}_v\boldsymbol{X}P_\pi \cdot \mathrm{softmax}\left(P_\pi^T(\boldsymbol{W}_k\boldsymbol{X})^T \cdot \boldsymbol{W}_q\boldsymbol{X}P_\pi\right) \\ &= \boldsymbol{W}_v\boldsymbol{X}(P_\pi P_\pi^T) \cdot \mathrm{softmax}\left((\boldsymbol{W}_k\boldsymbol{X})^T \cdot \boldsymbol{W}_q\boldsymbol{X}\right)P_\pi \\ &= \boldsymbol{W}_v\boldsymbol{X} \cdot \mathrm{softmax}\left((\boldsymbol{W}_k\boldsymbol{X})^T \cdot \boldsymbol{W}_q\boldsymbol{X}\right)P_\pi \\ &= \mathcal{T}_\pi(A_s(\boldsymbol{X})). \end{aligned} \tag{7}$$

□

Proof

Proof.

Note that $P_\pi^T P_\pi = I$ since P_π is an orthogonal matrix. And it is easy to verify that

$$\mathrm{softmax}(P_\pi^T \boldsymbol{M} P_\pi) = P_\pi^T \mathrm{softmax}(\boldsymbol{M}) P_\pi$$

for any matrix $\boldsymbol{M}$. By showing $A_s(\mathcal{T}_\pi(\boldsymbol{X})) = \mathcal{T}_\pi(A_s(\boldsymbol{X}))$ we have shown that A_s is spatial permutation equivariant according to Definition 2. □

Proof

Proof.

Similarly, when applying $\mathcal{T}_\pi$ to the input of an attention operator A_Q with a learned query $\boldsymbol{Q}$, which is independent of the input $\boldsymbol{X}$, we have

$$\begin{aligned} A_Q(\mathcal{T}_\pi(\boldsymbol{X})) &= \boldsymbol{W}_v \mathcal{T}_\pi(\boldsymbol{X}) \cdot \operatorname{softmax}\left((\boldsymbol{W}_k \mathcal{T}_\pi(\boldsymbol{X}))^T \cdot \boldsymbol{Q} \right) \\ &= \boldsymbol{W}_v \boldsymbol{X}(P_\pi P_\pi^T) \cdot \operatorname{softmax}\left((\boldsymbol{W}_k \boldsymbol{X})^T \cdot \boldsymbol{Q} \right) \\ &= \boldsymbol{W}_v \boldsymbol{X} \cdot \operatorname{softmax}\left((\boldsymbol{W}_k \boldsymbol{X})^T \cdot \boldsymbol{Q} \right) \\ &= A_Q(\boldsymbol{X}). \end{aligned} \tag{8}$$

Since $A_{\boldsymbol{Q}}(\mathcal{T}_\pi(\boldsymbol{X})) = A_{\boldsymbol{Q}}(\boldsymbol{X})$, we have shown that $A_{\boldsymbol{Q}}$ is spatial permutation invariant according to Definition 2. □

Property of Convolutions

1. It is easy to verify that a convolution with a kernel size of 1 is equivariant to spatial permutations since the output values of a pixel only depends on the pixel itself.
2. However, convolutions with kernel sizes larger than 1 is neither spatially permutation invariant nor equivariant, because the output values of a pixel depends on the pixel and its neighbors with fixed order.
3. When the neighbors or the order of neighbors are changed during a permutation, the output value is consequently changed. As equivariance or invariance are desired in different tasks, certain approaches are used to help the convolutions learn to be equivariance or invariance. A common approach is to perform the data augmentation during training.
4. An exception exists for the translation operation. In particular, convolutions with kernel sizes larger than 1 are equivariant to translations.

THANKS!