FusedAttention: fusing attention matrix into attention vector

> *If some formula are not displayed clearly, chose the pdf format* **README.pdf** *in this repository.*

Self-Attention

Let $X \in \mathrm{R}^{h \times L}$ denotes a sequence of L feature vectors of dimensions h. Formally, X is projected by three matrices $W_Q \in \mathrm{R}^{u \times h}$, $W_K \in \mathrm{R}^{u \times h}$ and $W_V \in \mathrm{R}^{d \times h}$ to corresponding representations Q, K and V. The output for all positions is computed as follows

$$
\begin{aligned}
Q &= W_Q * X, \\
K &= W_K * X, \\
V &= W_V * X, \\
\alpha &= \mathrm{softmax}(K^T Q/\sqrt{d}), \\
Y &= V * \alpha.
\end{aligned}
$$

Note that the softmax funtion is appied column-wise. The Q, K, V and α are referred to as the queries, keys, values, and attention matrix respectively, following the common terminology.

Fused-Attention

In self-attention, the output Y at position t is computed as a weighted average of the feature representations of all positions with a weight proportional to a similarity score between the representations.

$$
\begin{aligned}
Q_t &= W_Q X_t, \\
K &= W_K X, \\
V &= W_V X, \\
\alpha_t &= \text{softmax}(K^T Q_t / \sqrt{d}), \\
Y_t &= V * \alpha_t.
\end{aligned}
$$

In short, self-attention maps L inputs to L outputs. In fused-attention, we rewrite the formula as

$$
\begin{aligned}
Q &= f_Q(X), \\
K &= f_K(X), \\
V &= f_V(X), \\
\alpha &= fuse\,(norm(K^T Q / \sqrt{d}))\,, \\
Y &= g_V(V * \alpha),
\end{aligned}
$$

where f_Q, f_K and f_V are any functions that are legal to input X, $norm$ is a normalization function default to softmax appied column-wise, $fuse$ is an aggregation function default to mean appied row-wise, g_V is a transform function related to V, thus the attention variable $\alpha \in \mathrm{R}^{L\times 1}$ transforms values V from a matrix $\in \mathrm{R}^{d\times L}$ to a vector $\in \mathrm{R}^{d\times 1}$. In brief, fused-attention maps L inputs to 1 output. There are two forms of X：

- The first case is that X is a tensor of any size. Ignoring the batch dimension, suppose the size of X is $(C, W_1, W_2, ..., W_n)$, where C is the number of channels, W_n is the spatial width at the n-th spatial dimention. In this case, take f_V for example, it first transforms X into V' with size $(C', W_1', W_2', ..., W_n')$, then transforms V' into V with size (d, L), where $d = C' \prod_{i=1}^{n} \Delta W_i'$, of which $\Delta W_i'$ is the window size at the i-th spatial dimention, L is the total number of patches. f_Q and f_K are similar to f_V. If the number of dimentions of Y is required to be the same with X, then g_V transforms $V * \alpha$ into Y with size $(C', \Delta W_1', \Delta W_2', ..., \Delta W_n')$, otherwise g_V does nothing.
- The second case is that X is a collection of L tensors of the same size. Also ignoring the batch dimension, X_j is the j-th element of X, suppose the size of X_j is $(C, W_1, W_2, ..., W_n)$, where C is the number of channels, W_n is the spatial width at the n-th spatial dimention. In this case, take f_V for example, it first transforms X_j into V_j' with size $(C', W_1', W_2', ..., W_n')$, then reshape V_j' to V_j of size $(C' \prod_{i=1}^{n} W_i', 1)$, then concatenate $V_j, j = 1, 2, ..., L$ into V with size

$(C' \prod_{i=1}^{n} W_i^{'}, L)$. f_Q and f_K are similar to f_V. As to g_V, it transforms $V * \alpha$ into Y with size $(C', W_1^{'}, W_2^{'}, ..., W_n^{'})$.