```python
class MultiHeadAttention(nn.Module):  #@save
    def __init__(self, num_hiddens, num_heads, dropout, bias=False, **kwargs):
        super().__init__()
        self.num_heads = num_heads
        self.attention = DotProductAttention(dropout)
        self.W_q = nn.LazyLinear(num_hiddens, bias=bias)
        self.W_k = nn.LazyLinear(num_hiddens, bias=bias)
        self.W_v = nn.LazyLinear(num_hiddens, bias=bias)
        self.W_o = nn.LazyLinear(num_hiddens, bias=bias)
```

```python
    def transpose_qkv(self, X):
        X = X.reshape(X.shape[0], X.shape[1], self.num_heads, -1)
        X = X.permute(0, 2, 1, 3)
        return X.reshape(-1, X.shape[2], X.shape[3])
```

transpose_qkv

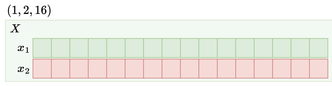
$B =$ batch size

$d_{model} =$ embedding size or num of hiddens

$\text{length} =$ no. of queries or key-value pairs

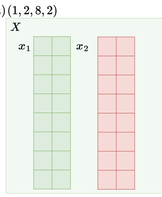
$h =$ num of heads

$(B,\ \text{length}, d_{model})$ $(1, 2, 16)$



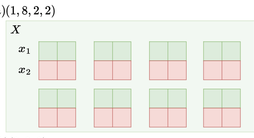
reshape

$(B,\ \text{length}, h, d_{model}/h)$ $(1, 2, 8, 2)$



permute

$(B, h,\ \text{length}, d_{model}/h)$ $(1, 8, 2, 2)$



reshape

$(B * h,\ \text{length}, d_{model}/h)$ $(8, 2, 2)$


```python
    def forward(self, queries, keys, values, valid_lens):
        queries = self.transpose_qkv(self.W_q(queries))
        keys = self.transpose_qkv(self.W_k(keys))
        values = self.transpose_qkv(self.W_v(values))
        if valid_lens is not None:
            valid_lens = torch.repeat_interleave(
                valid_lens, repeats=self.num_heads, dim=0
            )
        output = self.attention(
            queries,
            keys,
            values,
            valid_lens
        )
        output_concat = self.transpose_output(output)
        return self.W_o(output_concat)
```

queries=keys=values=X

after transpose_qkv

$(B * h,\ \text{length}, d_{model}/h)$ $(8, 2, 2)$

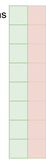
(B, length) $(1, 2)$

repeat dim=0 heads times

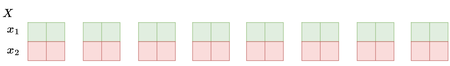
valid_lens

$(B * h, \text{length})$ $(8, 2)$

valid_lens

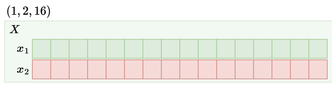


$(B * h,\ \text{length}, d_{model}/h)$ $(8, 2, 2)$



note that attention keep the dim

$(B,\ \text{length}, d_{model})$ $(1, 2, 16)$



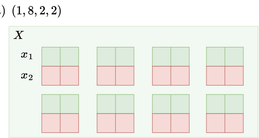
$(B,\ \text{length}, d_{model})$

```python
    def transpose_output(self, X):
        X = X.reshape(-1, self.num_heads, X.shape[1], X.shape[2])
        X = X.permute(0, 2, 1, 3)
        return X.reshape(X.shape[0], X.shape[1], -1)
```

$(B * h,\ \text{length}, d_{model}/h)$ $(8, 2, 2)$

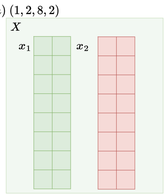
reshape

$(B, h,\ \text{length}, d_{model}/h)$ $(1, 8, 2, 2)$



permute

$(B,\ \text{length}, h, d_{model}/h)$ $(1, 2, 8, 2)$



reshape

$(B,\ \text{length}, d_{model})$ $(1, 2, 16)$