More Details about the Transformer-Based Model

July 2021

1 Image patches and linear projection

The details of this module is shown in Figure 2. Suppose the input image is $I \in R^{H\times W\times 3}$. I first split it into image patches, the size of each path is $Ph \times Pw$, the image patches IP can be represented as $IP \in R^{(\frac{H}{Ph}\times\frac{W}{Pw})\times(Ph\times Pw\times 3)}$. Besides the patches from the original image, I also extract the patches of exemplar, which can be written as $EP \in R^{K\times(Ph\times Pw\times 3)}$, K is the number of scaling exemplar. Then I concat them to get the Patches $P \in R^{(\frac{H}{Ph}\times\frac{W}{Pw}+K)\times(Ph\times Pw\times 3)}$.

There are total $(\frac{H}{Ph} \times \frac{W}{Pw} + K)$ patches, and the feature dimension of each patch is $(Ph \times Pw \times 3)$, the linear projection is to mapping the old feature to a new feature. The details of linear projection are shown in Figure 1. After the linear projection we can get the input embedding $IE \in (\frac{H}{Ph}\times\frac{W}{Pw}+K)\times d$, where d is the embedding dimension. The implementation of this linear projection is a simple convolution layer, the kernel size is the same as the patch size and the output channel is d.

2 Decoder

Since the Self-attention and the MLP will not change the dimension, so the shape of output embedding is the same as the input embedding. Then I drop the embedding of exemplar, and reshape the $(\frac{H}{Ph} \times \frac{W}{Pw}) \times d$ to $\frac{H}{Ph} \times \frac{W}{Pw} \times d$,

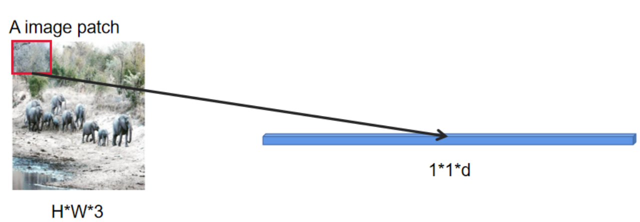

Figure 1: Details about linear projection

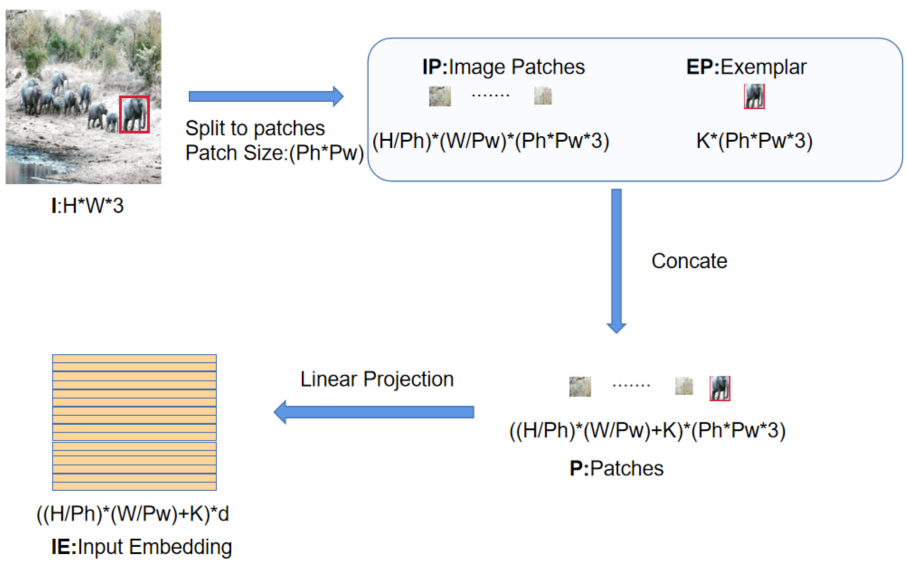

Figure 2: Patches and Linear Projection

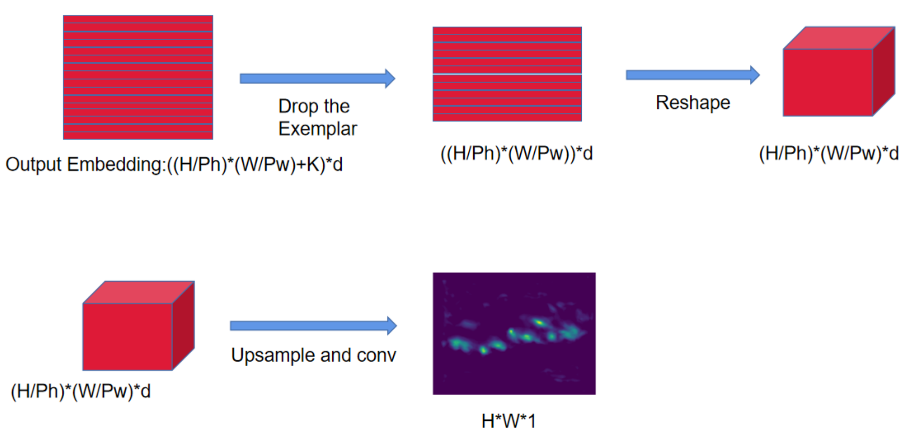

Figure 3: Decoder

which can be viewed as a feature map. Finally, I feed it into upsampling and conv layers and output the density map(same size as the input image).