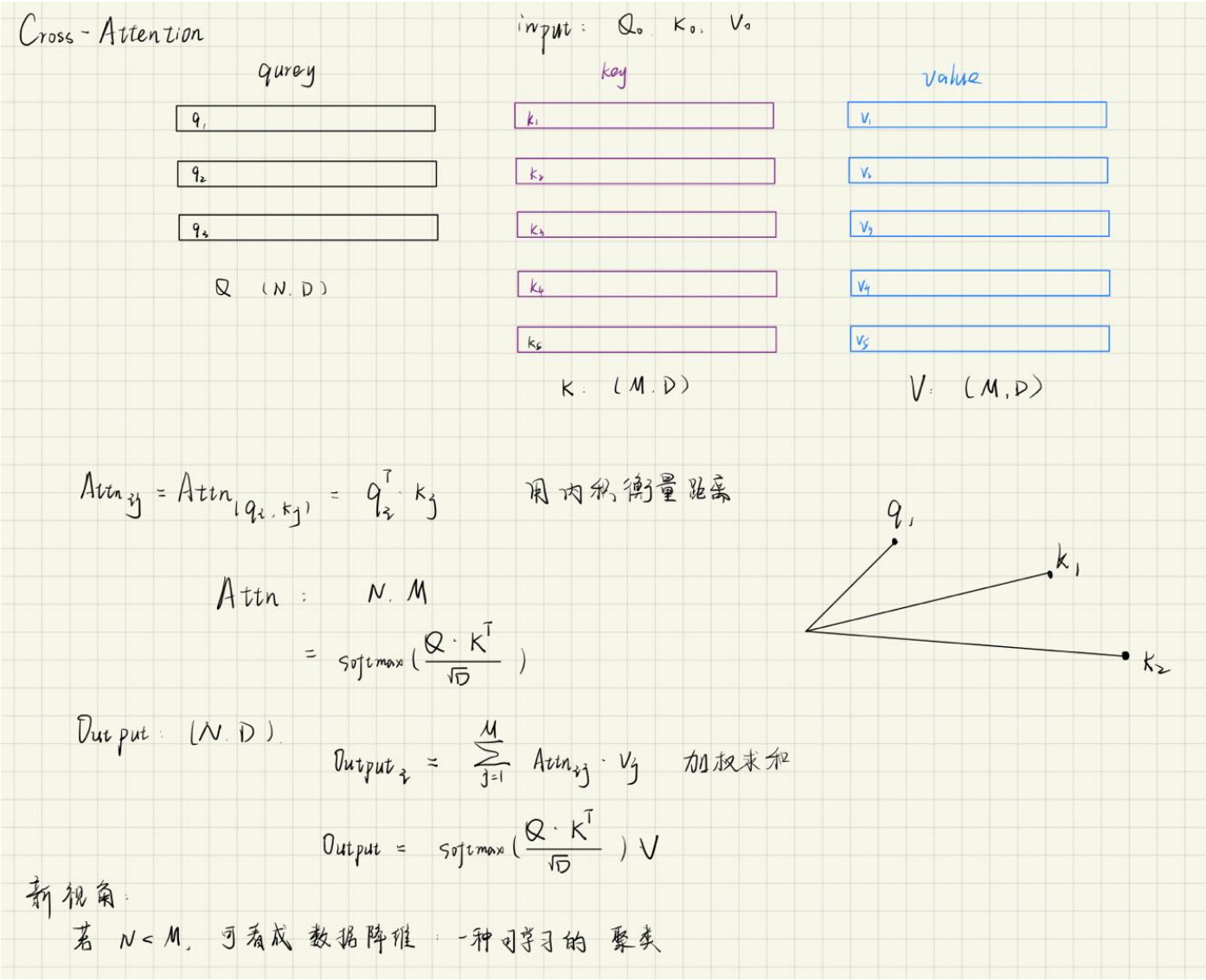
Author: Alexey Dosovitskiy
institute: Google
publication: ICLR 2021

PDF:
Dosovitskiy 等。 - 2021 - An Image is Worth 16x16 Words Transformers for Im.pdf

by YuLiu

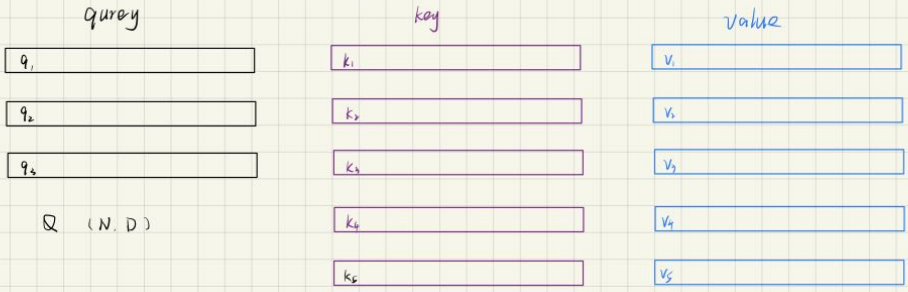
0. Attention



Self - Attention ? input : Q_0 Q_0 Q_0

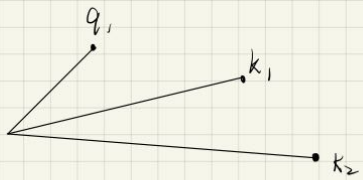
Multy - Head ?

Cross - Attention input : Q_0 K_0 V_0



Q (N, D) K (M, D) V (M, D)

$Attn_{ij} = Attn_{(q_i, k_j)} = q_i^T \cdot k_j$ 用内积衡量距离



$Attn$: N, M

$= Q \cdot K^T$

Output : (N, D)

$Output_i = \sum_{j=1}^{M} Attn_{ij} \cdot k_j$ 加权求和

$Output = Q \cdot K^T \cdot V$

新视角：

若 $N < M$，可看成 数据降维 一种可学习的 聚类

1. Motivation

- Attention结构在NLP已经成为标配，效果很好，但它在CV的应用很少，仅仅作为一些模块来使用
- 搞一个纯Transformer的模型，看看它的效果
- 尽可能少对Transformer做修改

2. Related work & Contribution

Transformer用于CV面临的问题

计算复杂度的问题

对于一个224×224 的图片，相当于一个长为50176的sequence，计算量过大
降低计算量方式

- 用特征图做输入，先卷积再用attention
- 专门搞一个针对图片的注意力结构，理论上有效，没有硬件加速。
- Stand-alone attention 孤立自注意力：使用局部注意力，只对一部分图片像素做attention，这种方法可以代替卷积
- Cordonnier： 把一张图片分成 2×2 的patch，再做attention，和我们的最像，效果还行，但只能在小分辨率图片上做

没有归纳偏置 inductive bias

- 需要更多的数据，更大的模型
- 泛化性能更强，扩展性更强

Contribution

- 一个纯Transformer模型，能在中等分辨率图片（128， 224，...）上做，效果可以达到甚至超过原来最好的模型
- 数据越多，模型越大，效果越好，Money is all you need

3. Insight

- 图像可以被分成patchs，作为一个sequence来做attention

4. Method

Overview

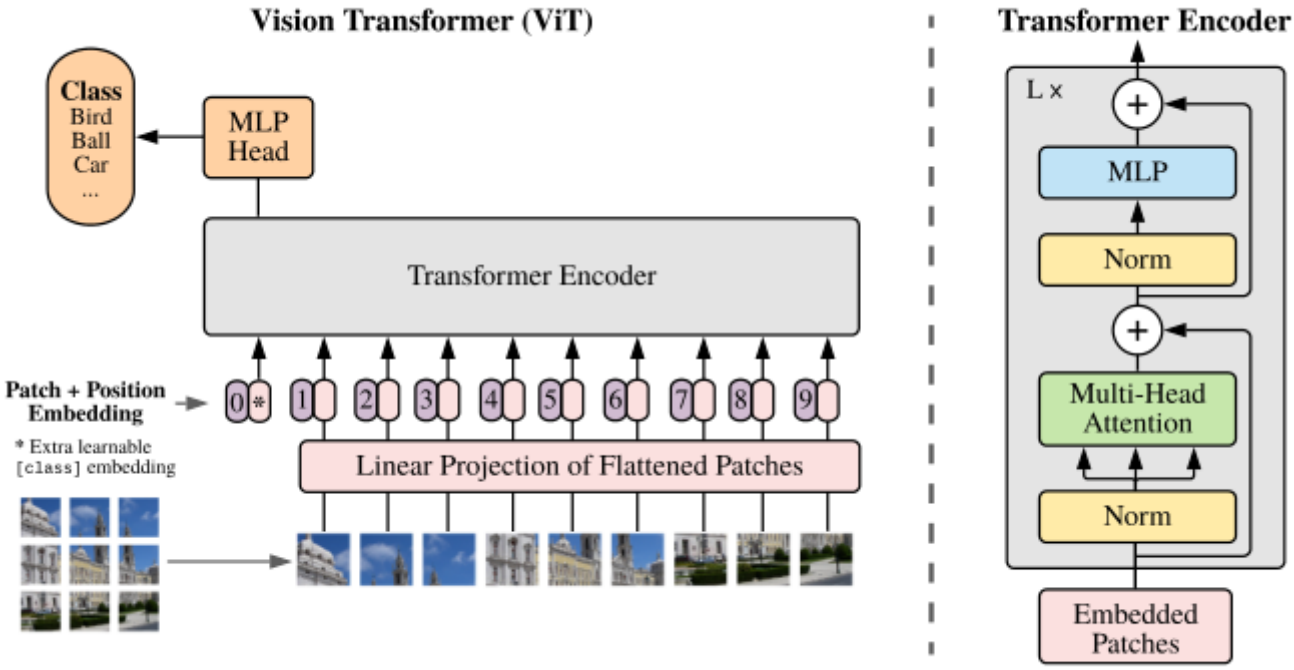


Image -> Sequence

1. 直接裁剪成196个16×16×3的patch，作为sequence：(B, 224, 224, 3) -> (B, 196, 768)
2. 对sequence做一个**线性投影**：(B, 196, 768) -> (B, 196, D)
3. 在最前面加一个用于分类的class token： (B, 196, D) -> (B, 197, D)
4. 加positional embedding: $x' = x + pos_emb$, (B, 197, D)
5. 过L层Transformer 的Encoder（Multi-self-attention), (B, 197, D)
6. 拿出原来的class token，过一个MLP**分类头**得到最后的分类结果

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}}; \mathbf{x}_p^1\mathbf{E}; \mathbf{x}_p^2\mathbf{E}; \cdots; \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}, \qquad \mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D}, \ \mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D} \tag{1}$$

$$\mathbf{z}'_\ell = \text{MSA}(\text{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1}, \qquad \ell = 1 \ldots L \tag{2}$$

$$\mathbf{z}_\ell = \text{MLP}(\text{LN}(\mathbf{z}'_\ell)) + \mathbf{z}'_\ell, \qquad \ell = 1 \ldots L \tag{3}$$

$$\mathbf{y} = \text{LN}(\mathbf{z}_L^0) \tag{4}$$

5. Experiments

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	**90.72** ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	**99.50** ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	–
CIFAR-100	**94.55** ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	–
Oxford-IIIT Pets	**97.56** ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	–
Oxford Flowers-102	99.68 ± 0.02	**99.74** ± 0.00	99.61 ± 0.02	99.63 ± 0.03	–
VTAB (19 tasks)	**77.63** ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

壕无人性

Pretaining

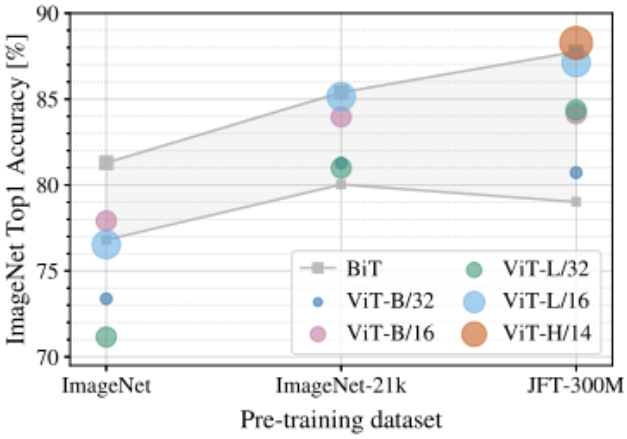


Figure 3: Transfer to ImageNet. While large ViT models perform worse than BiT ResNets (shaded area) when pre-trained on small datasets, they shine when pre-trained on larger datasets. Similarly, larger ViT variants overtake smaller ones as the dataset grows.

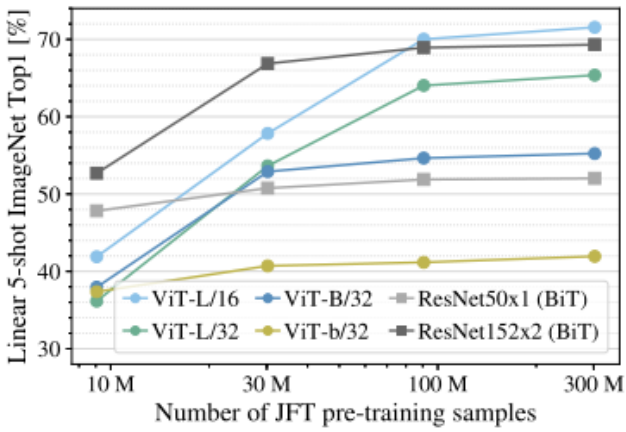
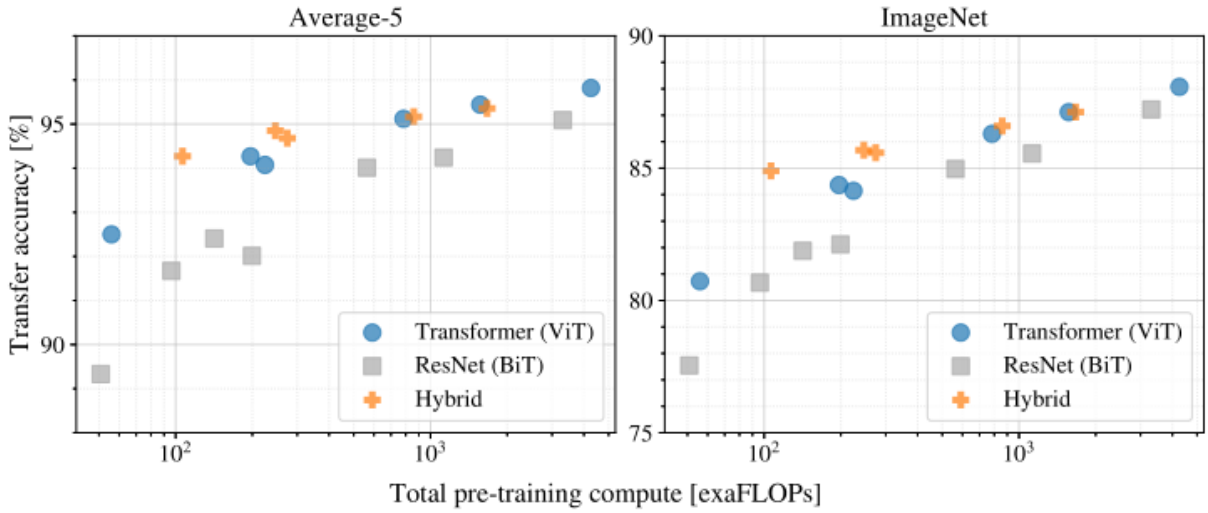


Figure 4: Linear few-shot evaluation on ImageNet versus pre-training size. ResNets perform better with smaller pre-training datasets but plateau sooner than ViT, which performs better with larger pre-training. ViT-b is ViT-B with all hidden dimensions halved.

结论：

- ViT小数据打不过CNN，中等数据集基本持平，大数据集效果更好
- 作为特征提取器，结论同上



结论：

- 相同计算成本，ViT效果更好
- 小模型，计算成本较低，混合模型的表现略优于ViT，但在大的模型上，这种差异就消失了
- 模型变大未出现明显饱和现象

分析

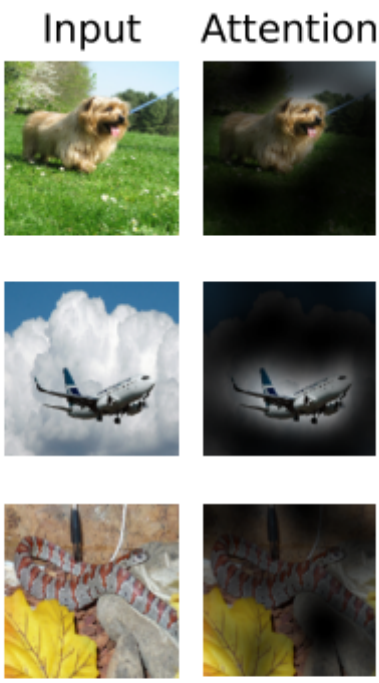
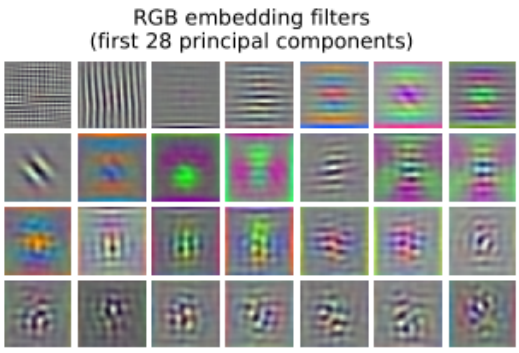
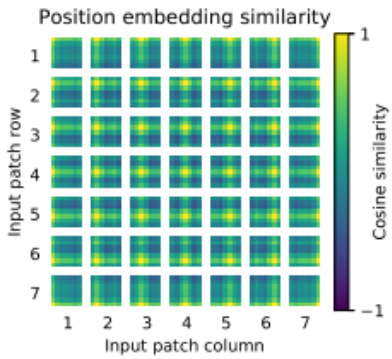
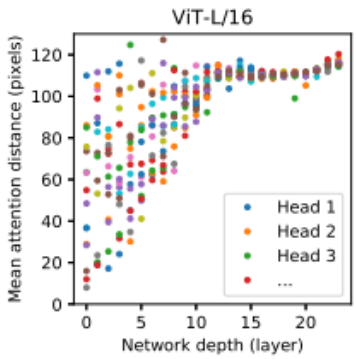


Figure 6: Representative examples of attention from the output token to the input space. See Appendix D.7 for details.







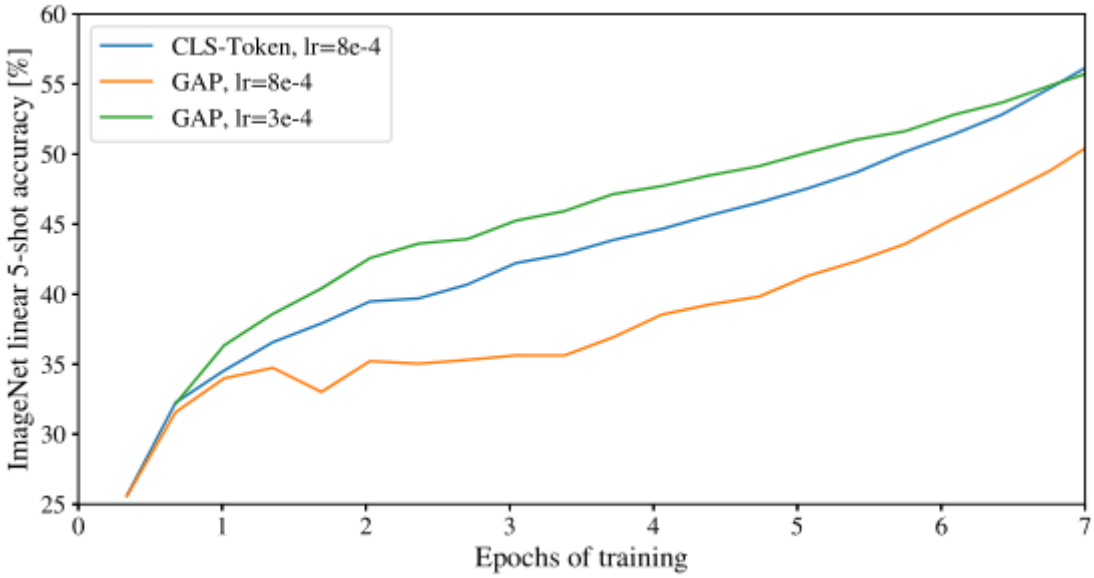
1. attention确实集中在物体上
2. 最初的embedding层学到了一些低维特征
3. positional embedding 学到了二维的位置信息
4. 层数越深，attention的视野越广 $d = d_{pixel} \times attn_weight$

自监督

像BERT一样，给图片做完形填空，先mask掉再reconstruc回来： MAE

6. Details

Class token



结论：不用Class token，对所有的向量做一个 GAP （global average pooling，全局池化），拿到的token也可以做，但是需要更低的学习率

Positional embedding

Pos. Emb.	Default/Stem	Every Layer	Every Layer-Shared
No Pos. Emb.	0.61382	N/A	N/A
1-D Pos. Emb.	0.64206	0.63964	0.64292
2-D Pos. Emb.	0.64001	0.64046	0.64022
Rel. Pos. Emb.	0.64032	N/A	N/A

Table 8: Results of the ablation study on positional embeddings with ViT-B/16 model evaluated on ImageNet 5-shot linear.

结论： 位置编码很重要，但不同位置编码没什么区别，可能是因为ViT处理的数据是patch level的，而不是pixel level的

简单实现

```python
def build_grid(resolution):
    ranges = [torch.linspace(0.0, 1.0, steps=res) for res in resolution]
    grid = torch.meshgrid(*ranges)
    grid = torch.stack(grid, dim=-1)
    grid = torch.reshape(grid, [resolution[0], resolution[1], -1])
    grid = grid.unsqueeze(0)
    return grid
```

7s

```python
class SoftPositionEmbed(nn.Module):
    def __init__(self, hidden_size: int, resolution):
        super().__init__()
        self.dense = nn.Linear(2, out_features=hidden_size)
        self.register_buffer("grid", build_grid(resolution))

    def forward(self, inputs):
        emb_proj = self.dense(self.grid).permute(0, 3, 1, 2)
        return inputs + emb_proj
```

```python
grid = build_grid([4, 4])
print(grid.shape)
print(grid)
```

✓ 0.1s

```
torch.Size([1, 4, 4, 2])
tensor([[[[0.0000, 0.0000],
          [0.0000, 0.3333],
          [0.0000, 0.6667],
          [0.0000, 1.0000]],

         [[0.3333, 0.0000],
          [0.3333, 0.3333],
          [0.3333, 0.6667],
          [0.3333, 1.0000]],

         [[0.6667, 0.0000],
          [0.6667, 0.3333],
          [0.6667, 0.6667],
          [0.6667, 1.0000]],

         [[1.0000, 0.0000],
          [1.0000, 0.3333],
          [1.0000, 0.6667],
          [1.0000, 1.0000]]]])
```

7. Thoughts

- 讲清楚Idea是怎么来的，讲清motivation，以及前人已经做了哪些工作
- 讲清自己的工作和前人之间的区别，哪些地方别人不行我行，突出自己的贡献
- 多思考自己的工作可以有哪些拓展的地方，给别人挖坑来研究