

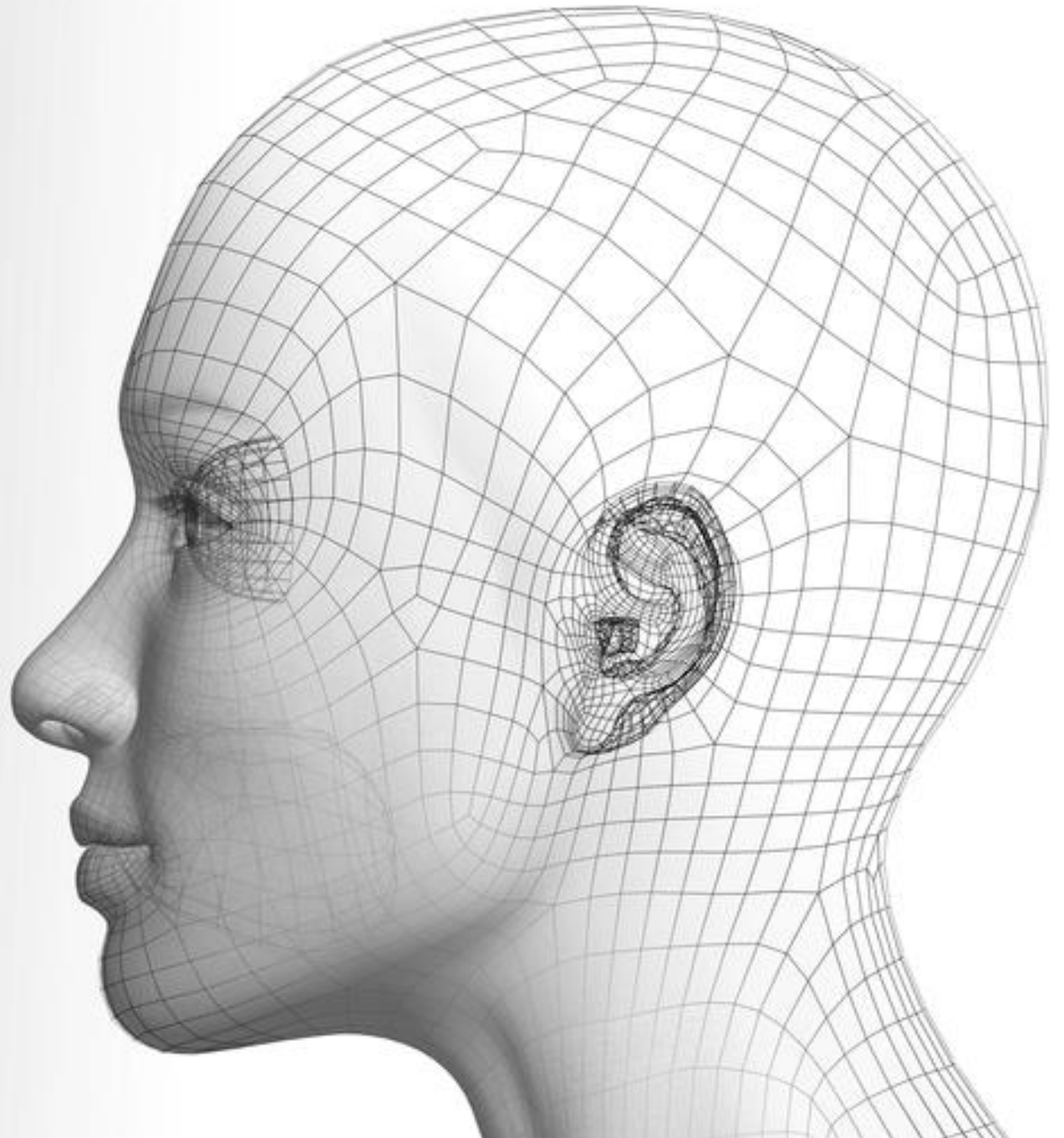
Tutorial 9

Transformers

Chen Change Loy (Cavan)

<https://www.mmlab-ntu.com/person/ccloy/>

<https://twitter.com/ccloy>



Question 1

The tutorial provides a simple walkthrough of the Vision Transformer. We hope you will be able to understand how it works by looking at the actual data flow during inference.

t9q1.ipynb

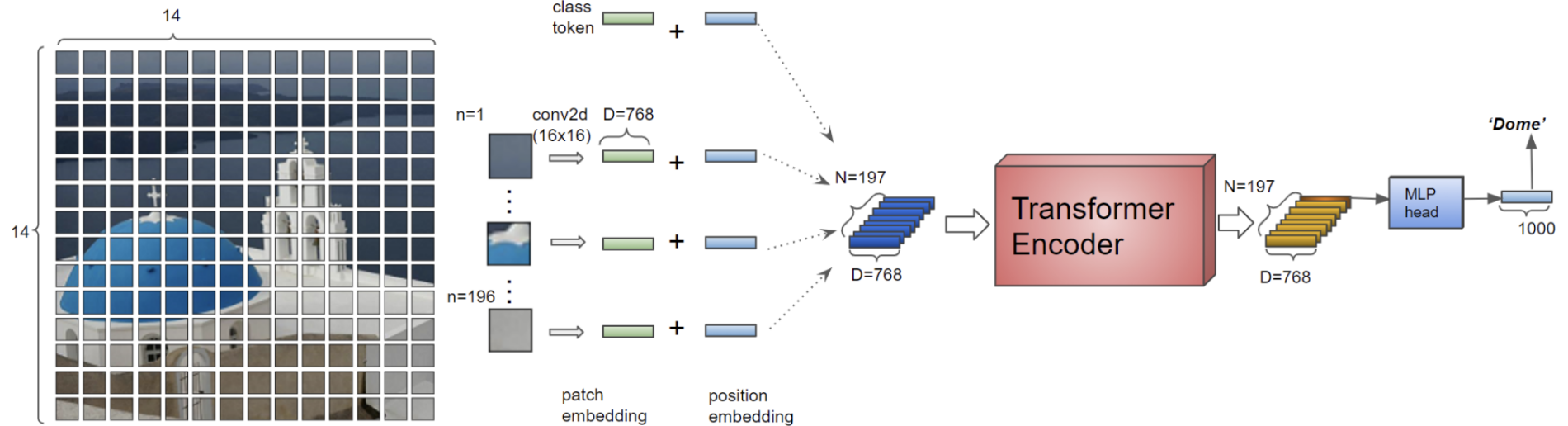


Figure 1. Vision Transformer inference pipeline.

1. Split Image into Patches

The input image is split into 14 x 14 vectors with dimension of 768 by Conv2d ($k=16 \times 16$) with stride=(16, 16).

2. Add Position Embeddings

Learnable position embedding vectors are added to the patch embedding vectors and fed to the transformer encoder.

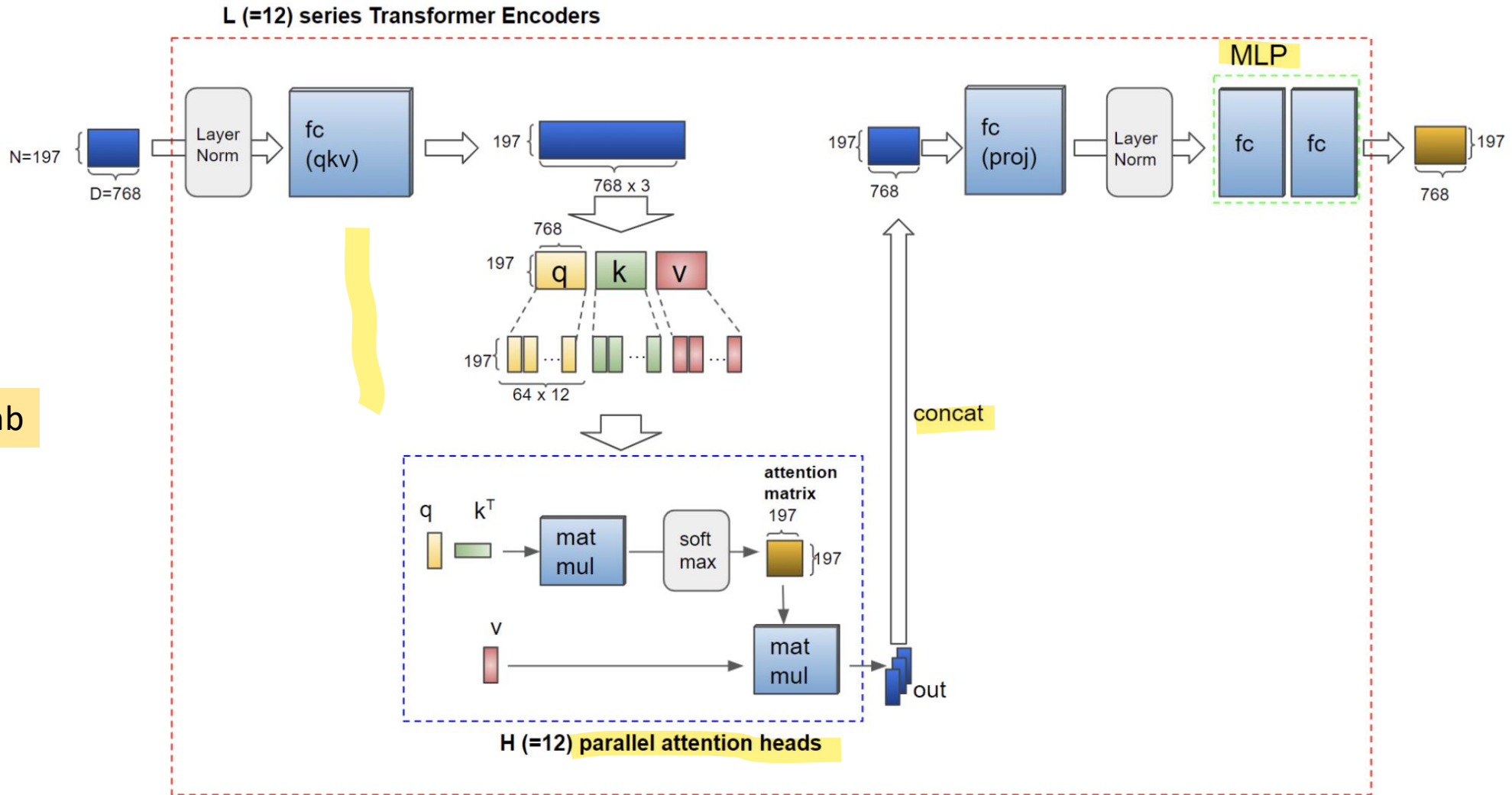
3. Transformer Encoder

The embedding vectors are encoded by the transformer encoder. The dimension of input and output vectors are the same. Details of the encoder are depicted in Fig. 2.

4. MLP (Classification) Head

The 0th output from the encoder is fed to the MLP head for classification to output the final classification results.

Question 1



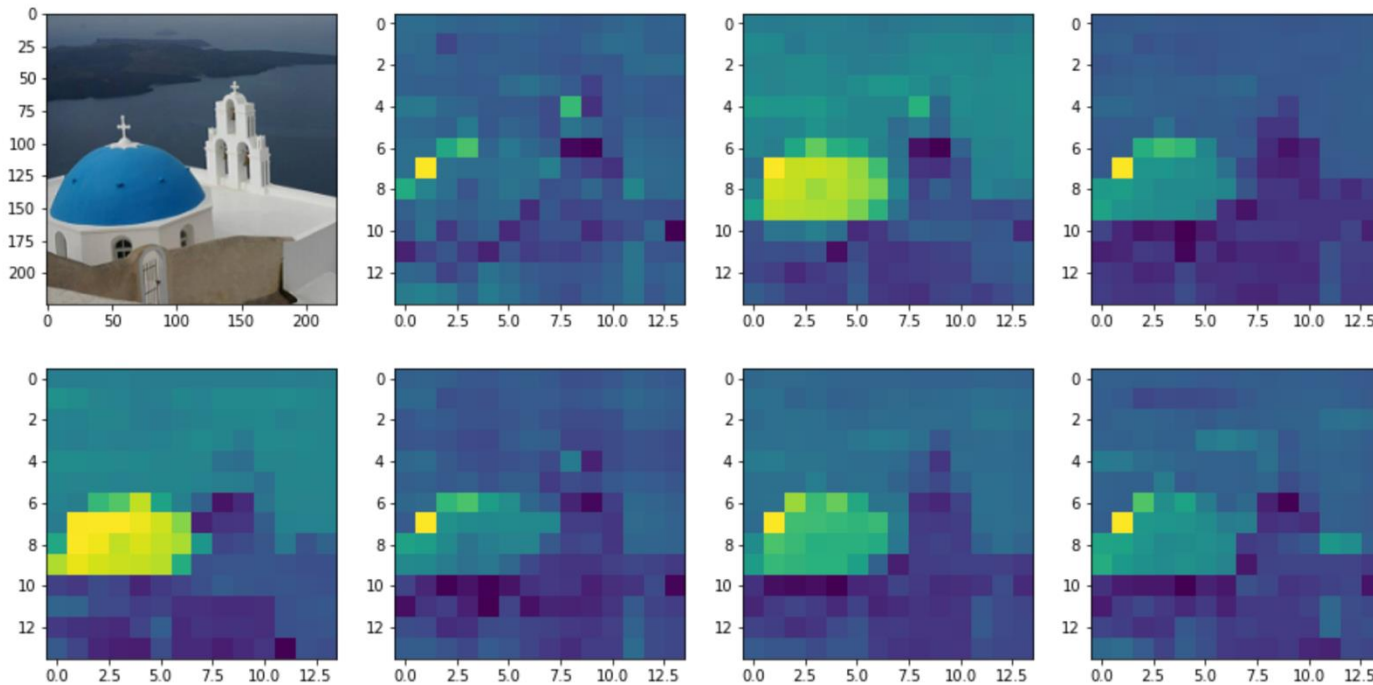
t9q1.ipynb

Question 1

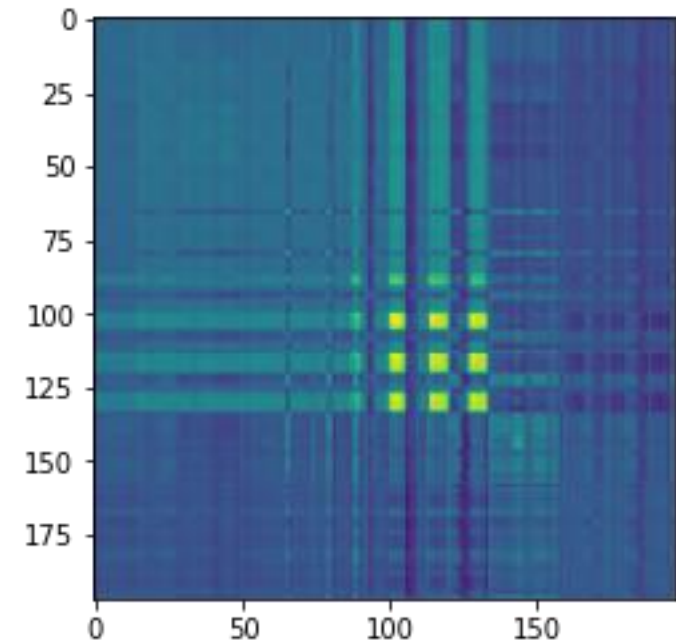
```
# Visualize attention matrix
fig = plt.figure(figsize=(16, 8))
fig.suptitle("Visualization of Attention", fontsize=24)
fig.add_axes()
img = np.asarray(img)
ax = fig.add_subplot(2, 4, 1)
ax.imshow(img)
for i in range(7): # visualize the 100th rows of attention matrices in the 0-7th heads
    attn_heatmap = attention_matrix[i, 100, 1:].reshape((14, 14)).detach().cpu().numpy()
    ax = fig.add_subplot(2, 4, i+2)
    ax.imshow(attn_heatmap)
```

visualize the 100th rows of attention matrices in the 0-7th heads

Visualization of Attention



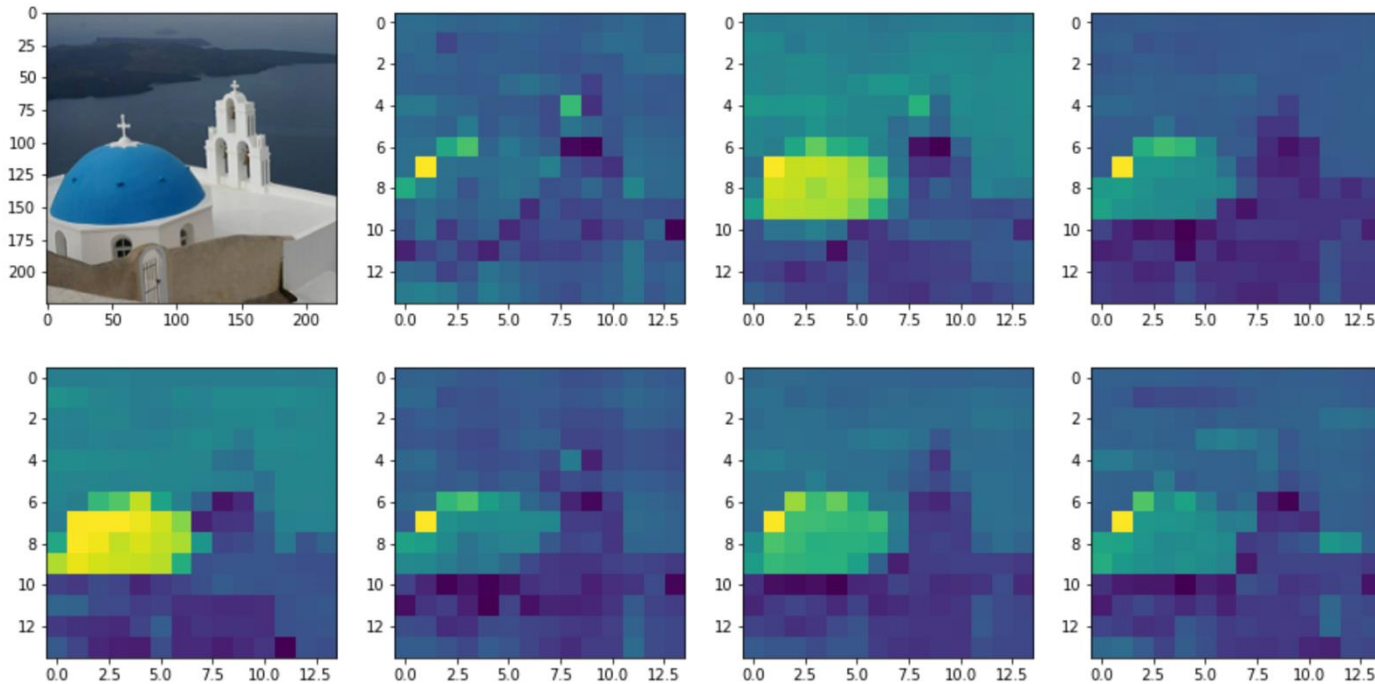
Showing the fourth attention matrix <197 x 197>



Question 1

```
# Visualize attention matrix
fig = plt.figure(figsize=(16, 8))
fig.suptitle("Visualization of Attention", fontsize=24)
fig.add_axes()
img = np.asarray(img)
ax = fig.add_subplot(2, 4, 1)
ax.imshow(img)
for i in range(7): # visualize the 100th rows of attention matrices in the 0-7th heads
    attn_heatmap = attention_matrix[i, 100, 1:].reshape((14, 14)).detach().cpu().numpy()
    ax = fig.add_subplot(2, 4, i+2)
    ax.imshow(attn_heatmap)
```

Visualization of Attention



visualize the 100th rows of attention matrices in the 0-7th heads

To check how the 100th patch attend to other patches

