SimAM: A Simple, Parameter-Free Attention Module for Convolutional Neural Networks (Supplementary Materials)

In this supplementary material, we first compare more recently similar re-calibration modules by using ResNet-50 (He et al., 2016) as backbone networks. These modules include selective kernel (SK) (Li et al., 2019b), attentive normalization (AttNorm) (Li et al., 2019a), dropout (Srivastava et al., 2014) as well as Swish function (Ramachandran et al., 2017). We also insert SimAM after all convolutional layers (SimAM-all). Moreover, we compare SE (Hu et al., 2018) with the proposed SimAM on another two variants of ResNet, ResNext-50 (32x4d) and Res2Net (Gao et al., 2019). We follow the standard training pipeline (100 epochs) to train all models from **scratch**. Results are shown in Table 1. As one can see, our SimAM-all achieves the best results and our SimAM also performs comparably against other modules based on ResNet-50 network. Moreover, the SimAM can also enhance the representation power of ResNeXt-50 and Res2Net-50.

Table 1. Top-1 accuracies (%) on ImageNet dataset with different options .

Model	Baseline	+SimAM	+SimAM-all	+SK	+AttNorm	+Dropout0.1	+Swish
ResNet-50	76.34%	77.46%	77.62%	77.50%	77.12%	75.51%	76.71%
Model	Baseline	+SimAM	+SE	Model	Baseline	+SimAM	+SE
ResNeXt-50	77.47%	78.00%	77.96%	**Res2Net-50**	77.94%	78.54%	78.41%

Second, we visualize the estimated 3-D weights by the proposed SimAM based on ResNet-50 network to demonstrate our mechanism. For each stage (model.layer1 to model.layer4) in ResNet-50, we show the weights generated in the last block on randomly selected 64 images from the ImageNet validation set. To show different channels of our 3-D weights, for each image, we randomly select 6 channels and show their selected index on each weight image. As one can see, our SimAM can attend some important points, curve (first two pictures), as well as some discriminative components (last two pictures).

References

Gao, S., Cheng, M.-M., Zhao, K., Zhang, X.-Y., Yang, M.-H., and Torr, P. H. Res2net: A new multi-scale backbone architecture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Hu, J., Shen, L., and Sun, G. Squeeze-and-excitation networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, 2018.

Li, X., Sun, W., and Wu, T. Attentive normalization. *arXiv:1908.01259*, 2019a.

Li, X., Wang, W., Hu, X., and Yang, J. Selective kernel networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 510–519, 2019b.

Ramachandran, P., Zoph, B., and Le, Q. V. Searching for activation functions. *arXiv:1710.05941*, 2017.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

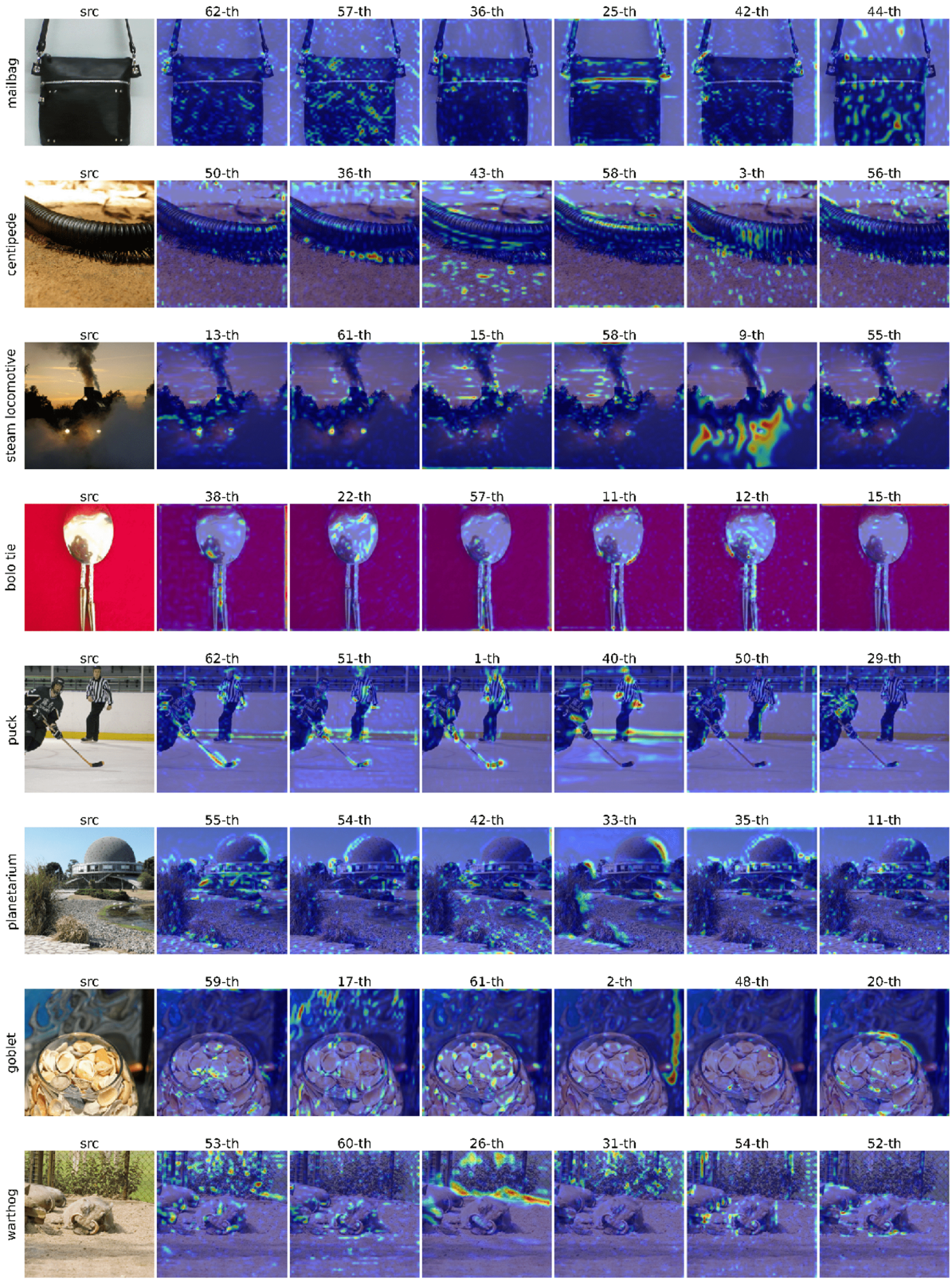

Figure 1. Visualization of attention weights generated by our method (from the last block in **"model.layer1"**). Each row represents one image with its correspondence attention weights. We randomly select different (shown as "X"-th) channels from our 3-D weights to show.

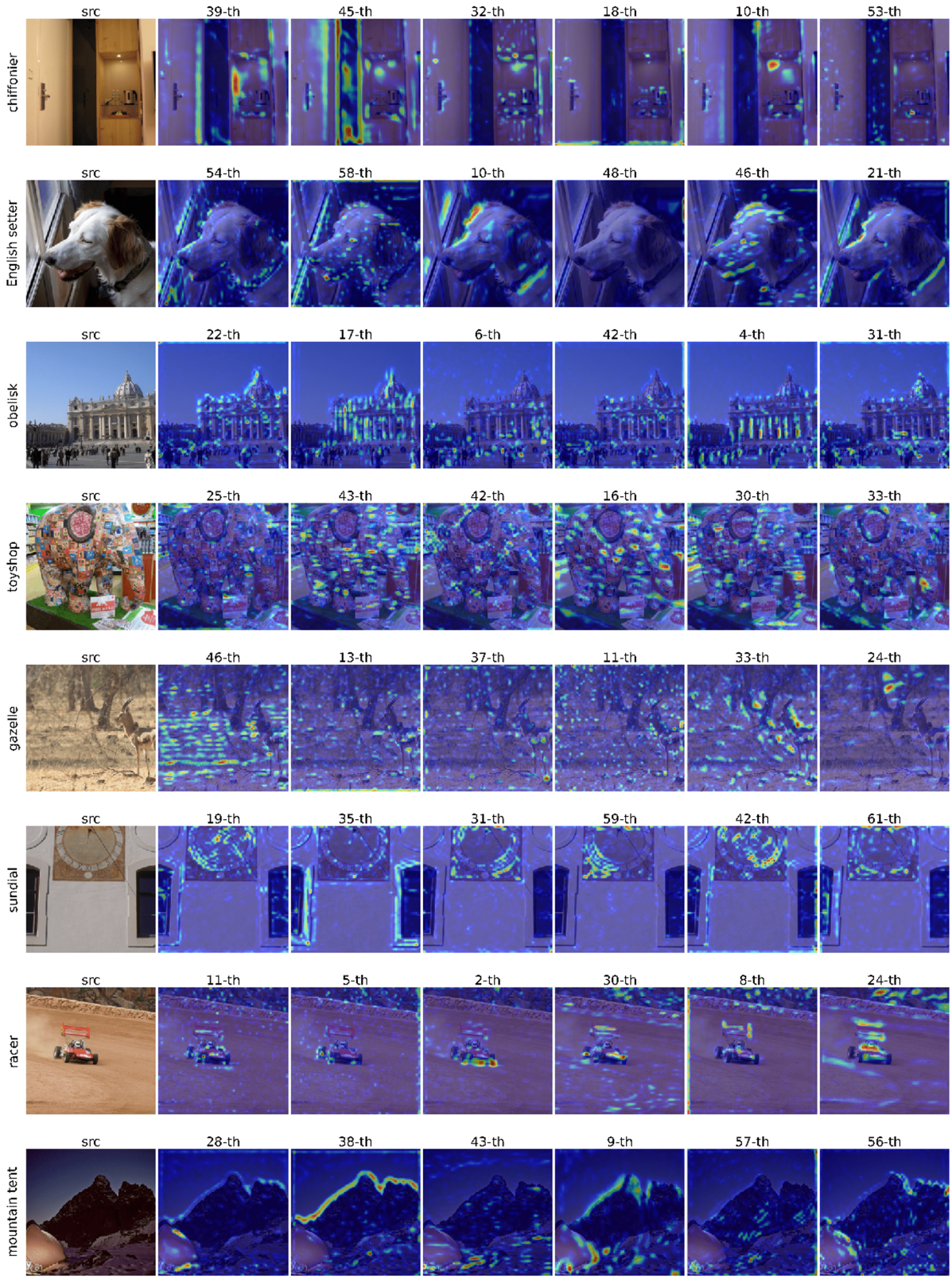

Figure 2. Visualization of attention weights generated by our method (from the last block in **"model.layer1"**). Each row represents one image with its correspondence attention weights. We randomly select different (shown as "X"-th) channels from our 3-D weights to show.

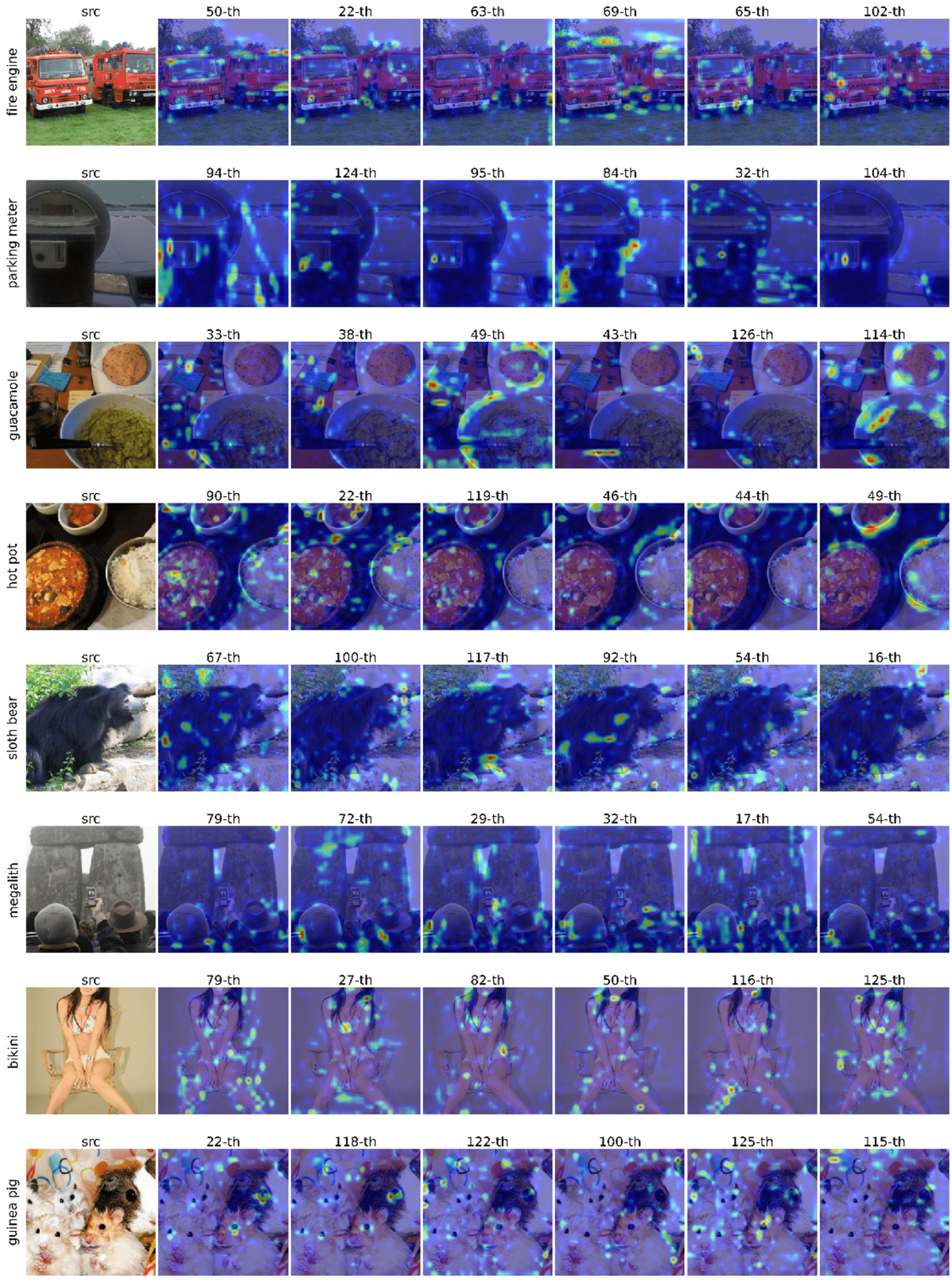

Figure 3. Visualization of attention weights generated by our method (from the last block in **"model.layer2"**). Each row represents one image with its correspondence attention weights. We randomly select different (shown as "X"-th) channels from our 3-D weights to show.

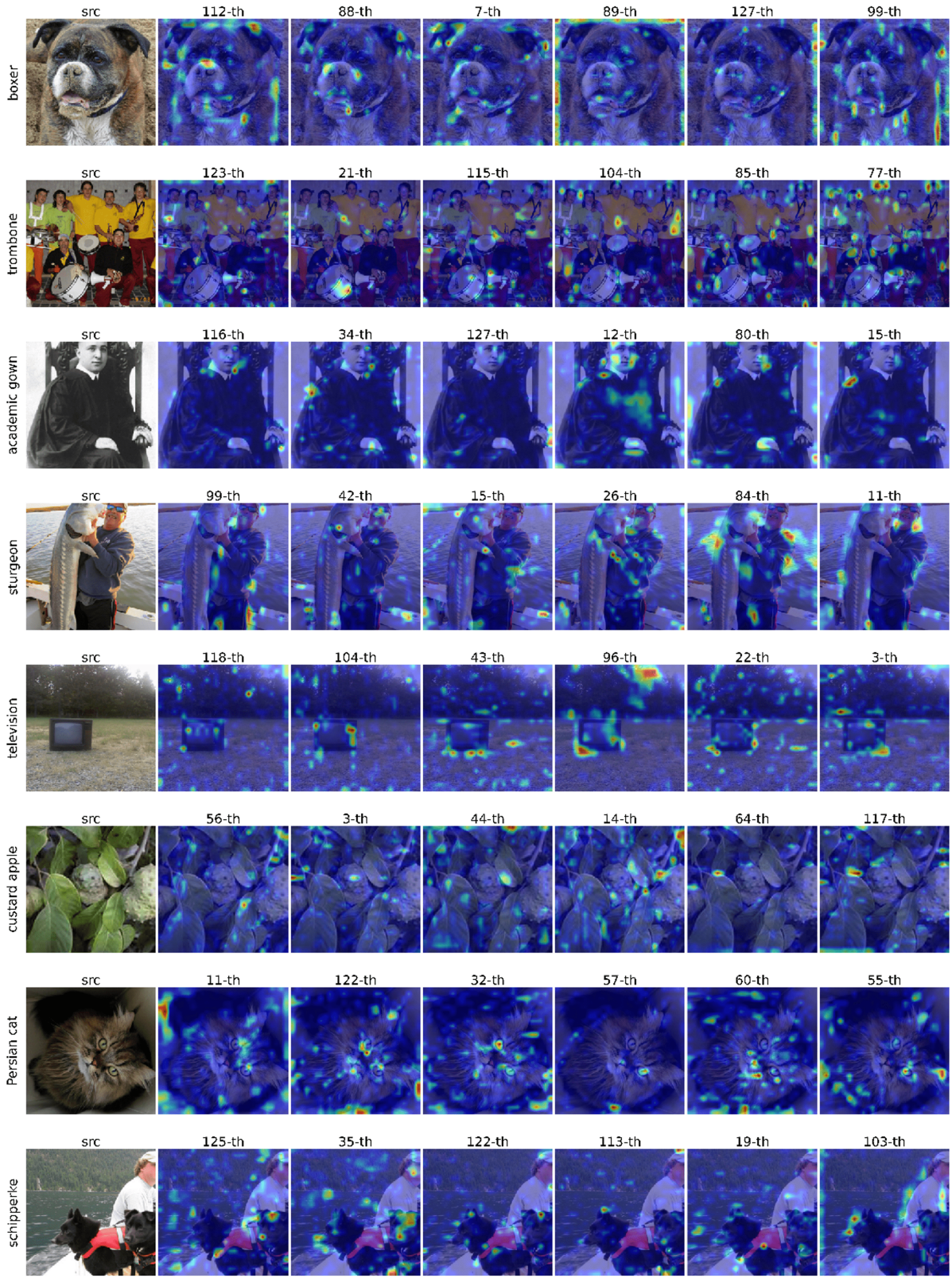

Figure 4. Visualization of attention weights generated by our method (from the last block in **"model.layer2"**). Each row represents one image with its correspondence attention weights. We randomly select different (shown as "X"-th) channels from our 3-D weights to show.

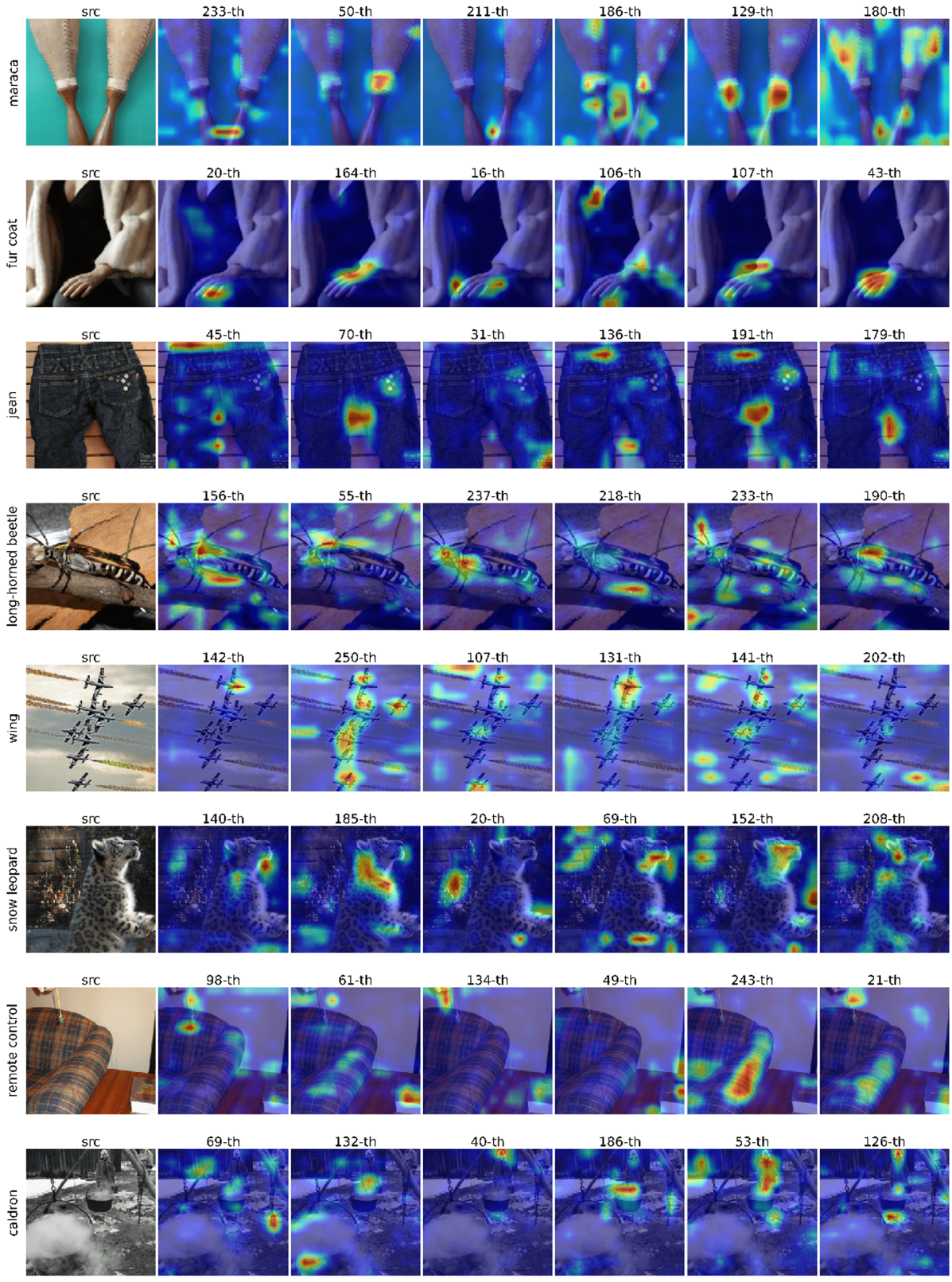

Figure 5. Visualization of attention weights generated by our method (from the last block in **"model.layer3"**). Each row represents one image with its correspondence attention weights. We randomly select different (shown as "X"-th) channels from our 3-D weights to show.

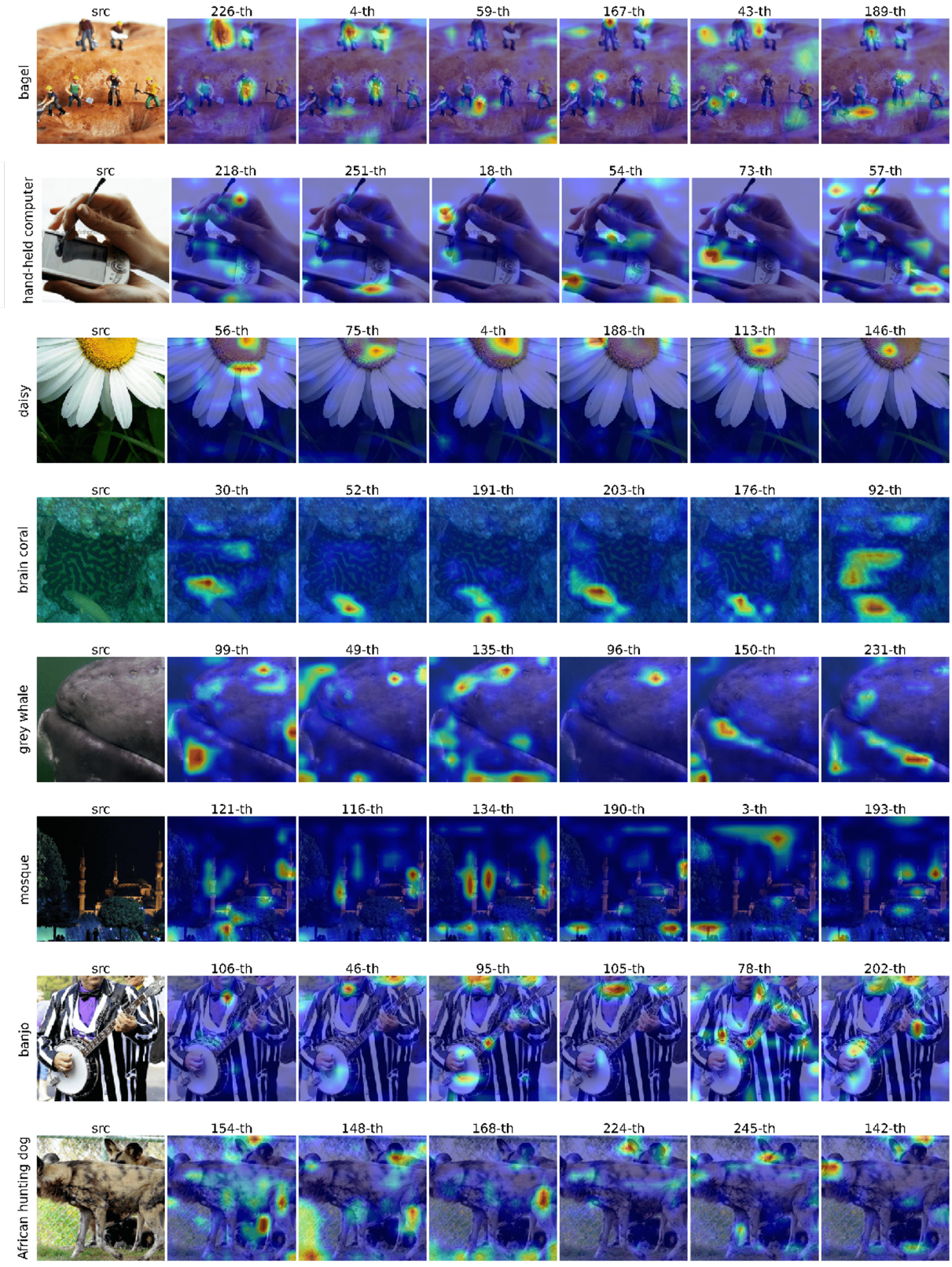

Figure 6. Visualization of attention weights generated by our method (from the last block in **"model.layer3"**). Each row represents one image with its correspondence attention weights. We randomly select different (shown as "X"-th) channels from our 3-D weights to show.

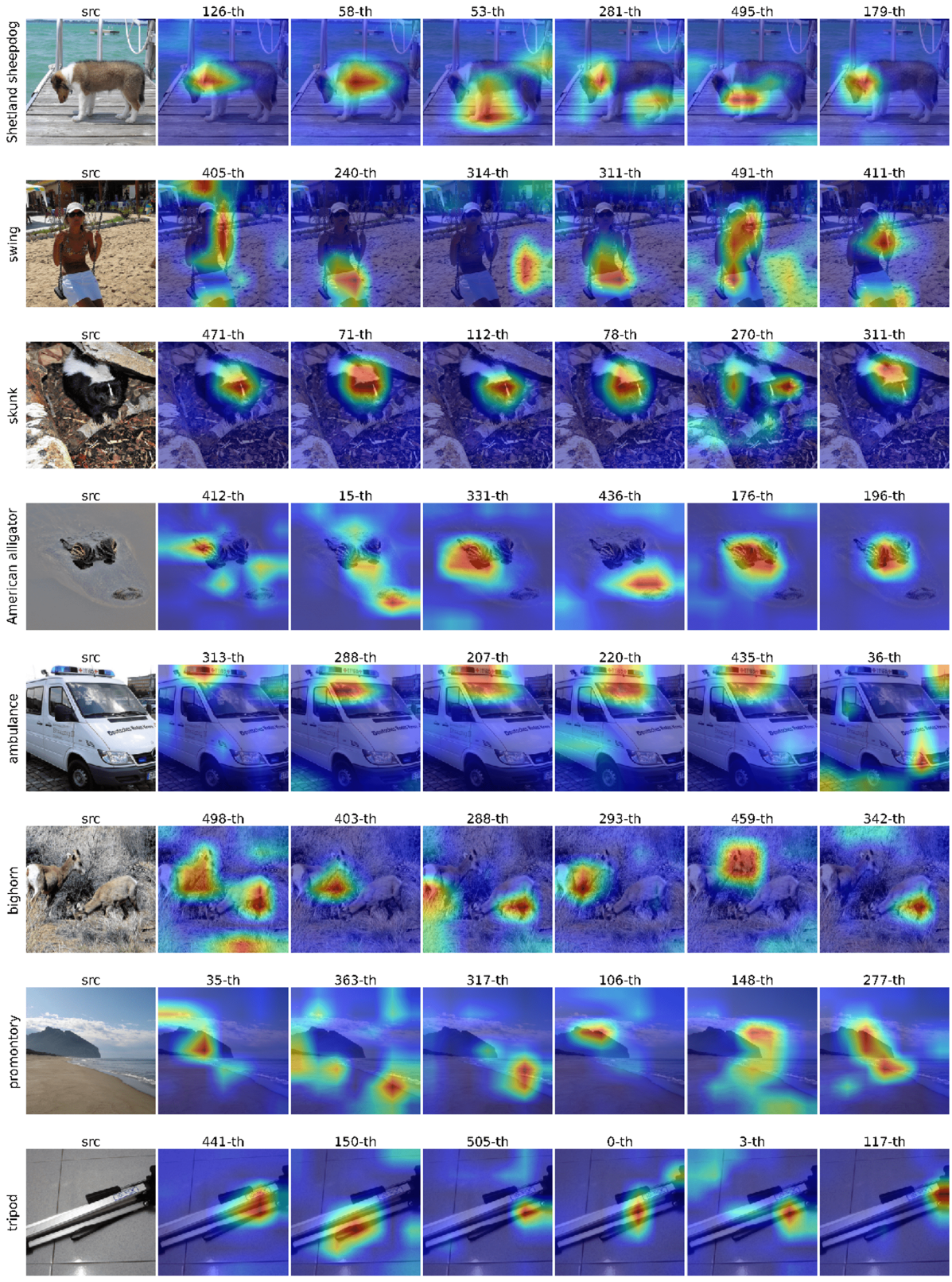

Figure 7. Visualization of attention weights generated by our method (from the last block in **"model.layer4"**). Each row represents one image with its correspondence attention weights. We randomly select different (shown as "X"-th) channels from our 3-D weights to show.

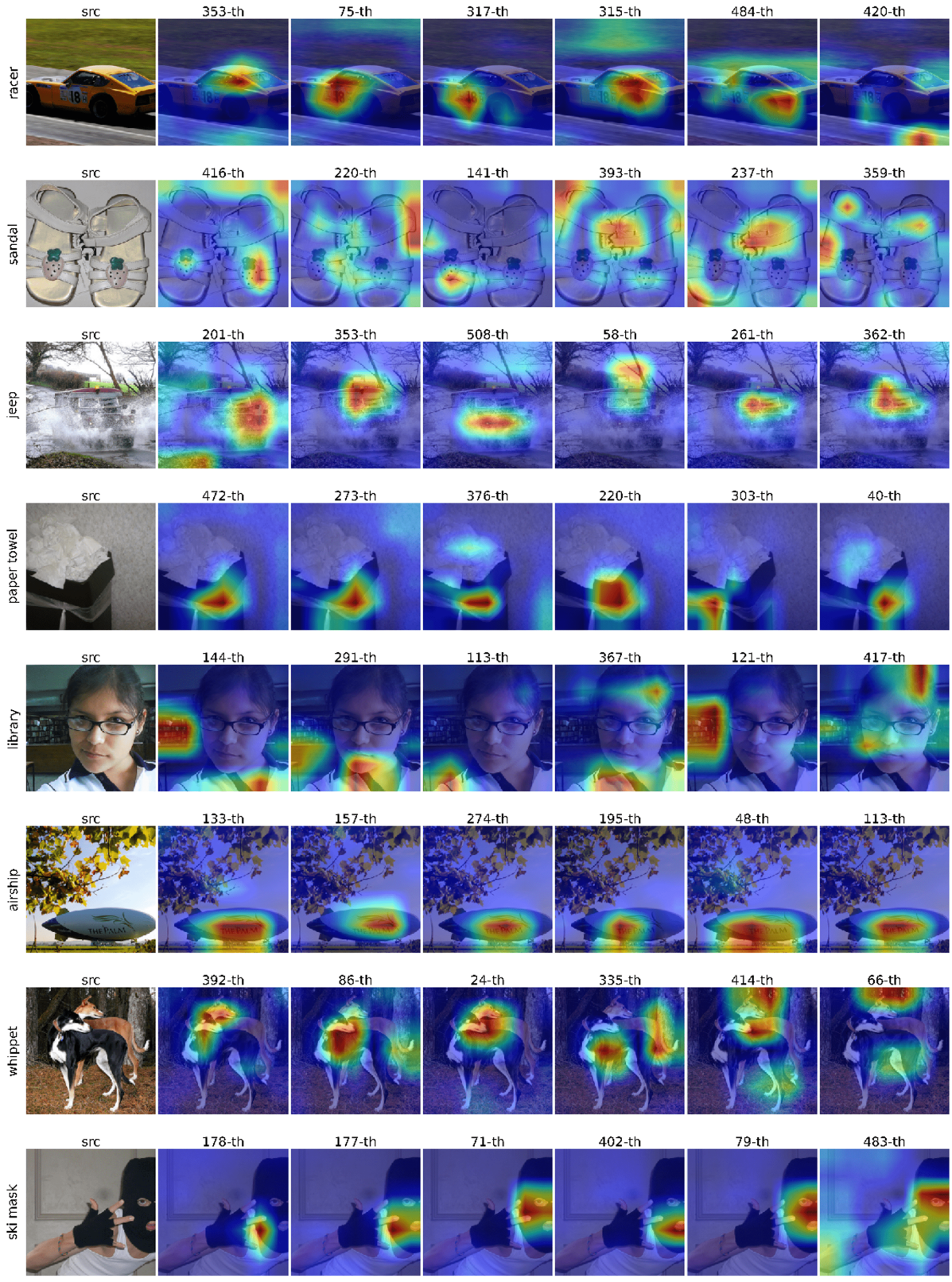

Figure 8. Visualization of attention weights generated by our method (from the last block in **"model.layer4"**). Each row represents one image with its correspondence attention weights. We randomly select different (shown as "X"-th) channels from our 3-D weights to show.