## Appendix A. The Misalignment Problem

As shown in Figure 5, up-sampling after the strided convolution with odd convolutional filter, *e.g.* $3 \times 3$, will cause the entire feature map to move to the lower right, which is problematic when we add the up-sampled shifted map with the unshifted map.

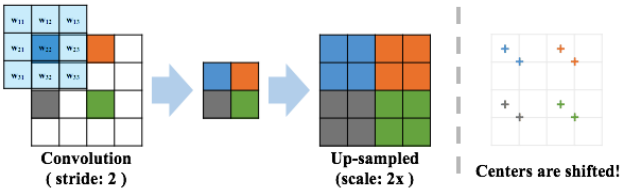



Figure 5: Strided convolution may cause misaligned feature maps after up-sampling.

## Appendix B. Relative Theoretical Gains of OctConv

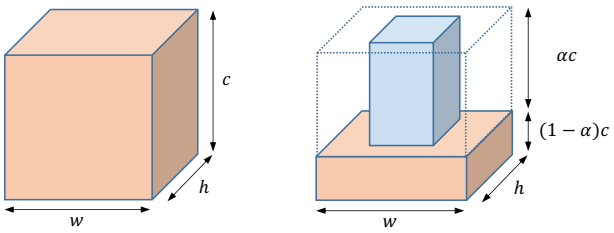
In Table 1 of the main paper, we reported the relative theoretical gains of the proposed multi-frequency feature representation over regular feature representation with respect to memory footprint and computational cost, as measured in FLOPS (*i.e.* multiplications and additions). In this section, we show how the gains are estimated in theory.

**Memory cost.** The proposed OctConv stores the feature representation in a multi-frequency feature representation as shown in Figure 6, where the low frequency tensor is stored in $2\times$ lower spatial resolution and thus cost $75\%$ less space to store the low frequency maps compared with the conventional feature representation. The relative memory cost is conditional on the ratio ($\alpha$) and is calculated by

$$1 - \frac{3}{4}\alpha. \tag{5}$$

**Computational cost.** The computational cost of OctConv is proportional to the number of locations and channels that are needed to be convolved on. Following the design shown in Figure 2 in the main paper, we need to compute four paths, namely $H \rightarrow H$, $H \rightarrow L$, $L \rightarrow H$, and $L \rightarrow L$.

We assume the convolution kernel size is $k \times k$, the spatial resolution of the high-frequency feature is $h \times w$, and there are $(1-\alpha)c$ channels in the high-frequency part and $\alpha c$ channels in the low-frequency part. Then the FLOPS for




(a) (b)

Figure 6: (a) The conventional feature representation used by vanilla convolution. (c) The proposed multi-frequency feature representation stores the smoothly changing, low-frequency maps in a low-resolution tensor to reduce spatial redundancy, used by Octave Convolution. The figure is rotated compared to the one in the main paper for clarity.

computing each paths are calculated as below.

$$\begin{aligned}
FLOPS(Y^{H\rightarrow H}) &= h \times w \times k^2 \times (1-\alpha)^2 \times c^2 \\
FLOPS(Y^{H\rightarrow L}) &= \frac{h}{2} \times \frac{w}{2} \times k^2 \times \alpha \times (1-\alpha) \times c^2 \\
FLOPS(Y^{L\rightarrow H}) &= \frac{h}{2} \times \frac{w}{2} \times k^2 \times (1-\alpha) \times \alpha \times c^2 \\
FLOPS(Y^{L\rightarrow L}) &= \frac{h}{2} \times \frac{w}{2} \times k^2 \times \alpha^2 \times c^2
\end{aligned} \tag{6}$$

We omit FLOPS for adding $Y^{H\rightarrow H}$ and $Y^{L\rightarrow H}$ together, as well as that of adding $Y^{L\rightarrow L}$ and $Y^{H\rightarrow H}$ together, since the FLOPS of such addition is less than $h \times w \times c$, and is negligible compared with other computational costs. The computational cost of the pooling operation is also ignorable compared with other computational cost. The nearest neighborhood up-sampling is basically duplicating values which does not involves any computational cost. Therefore, by adding up all FLOPS in Eqn 6, we can estimate the overall FLOPS for compute $Y^H$ and $Y^L$ in Eqn 7.

$$FLOPS([Y^H, Y^L]) = (1 - \frac{3}{4}\alpha(2-\alpha)) \times h \times w \times k^2 \times c^2 \tag{7}$$

For vanilla convolution, the FLOPS for computing output feature map $Y$ of size $c \times h \times w$ with the kernel size $k \times k$, and input feature map of size $c \times h \times w$, can be estimated as below.

$$FLOPS(Y) = h \times w \times k^2 \times c^2 \tag{8}$$

three out of four internal convolution operations are conducted on the lower resolution tensors except the first convolution, *i.e.* $f(X^H, W^{H\rightarrow H})$. Thus, the relative computational cost compared with vanilla convolution using the same kernel size and number of input/out channels is:

Therefore, the computational cost ratio between the OctConv and vanilla convolution is $(1 - \frac{3}{4}\alpha(2-\alpha))$.

$$\begin{aligned} &\frac{(1-\alpha)^2c^2 + \frac{1}{2}\alpha(1-\alpha)c^2 + \frac{1}{4}\alpha^2c^2}{c^2} \\ &= 1 - \frac{3}{4}\alpha(2-\alpha). \end{aligned} \tag{9}$$

Note that the computational cost of the pooling operation is ignorable and thus is not considered. The nearest neighborhood up-sampling is basically duplicating values which does not involves any computational cost.

## Appendix C. ImageNet Ablation Study Results

Table 7 shows that the gain of OctConv over baseline models increases as the test image resolution grows. Such ability of better detecting large objects can be explained as the larger receptive field of each OctConv.

Table 8 shows an ablation study to examine down-sampling and inter-octave connectivity on ImageNet. The results confirm the importance of having both inter-frequency communication paths. It also shows that pooling methods are better than strided convolution and the average pooling works the best.

Table 9 reports the values that are plotted in Figure 4 of the main text for clarity of presentation and to allow future work to compare to the precise numbers.

| Model | ratio ($\alpha$) | Testing Scale (*small* → *large*) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 256 | 320 | 384 | 448 | 512 | 576 | 640 | 740 |
| ResNet-50 | N/A | 77.2 | 78.6 | 78.7 | 78.7 | 78.3 | 77.6 | 76.7 | 75.8 |
| Oct-ResNet-50 | .5 | **+0.7** | **+0.7** | **+0.9** | **+0.9** | **+0.8** | **+1.0** | **+1.1** | **+1.2** |

Table 7: ImageNet classification accuracy. The short length of input images are resized to the target crop size while keeping the aspect ratio unchanged. A centre crop is adopted if the input image size is not square. ResNet-50 backbone trained with crops size of $256 \times 256$ pixels.

| Method | Down-sampling | Low → High | High → Low | Top-1 (%) |
|---|---|---|---|---|
| Oct-ResNet-50 ratio: 0.5 | avg. pooling | | | 76.0 |
| | avg. pooling | ✓ | | 76.4 |
| | avg. pooling | | ✓ | 76.4 |
| | strided conv. | ✓ | ✓ | 76.3 |
| | max. pooling | ✓ | ✓ | 77.0 |
| | avg. pooling | ✓ | ✓ | 77.4 |

Table 8: Ablation on down-sampling and inter-octave connectivity on ImageNet. Note that MG-Conv [25] uses max pooling for down-sampling.

| Backbone | | baseline | $\alpha = 0.125$ | $\alpha = 0.25$ | $\alpha = 0.5$ | $\alpha = 0.75$ |
|---|---|---|---|---|---|---|
| ResNet-26 | GFLOPs | 2.353 | 2.102 | 1.871 | 1.491 | 1.216 |
| | Top-1 acc. | 73.2 | 75.8 | 76.1 | 75.5 | 74.6 |
| DenseNet-121 | GFLOPs | 2.852 | 2.428 | 2.044 | - | - |
| | Top-1 acc. | 75.4 | 76.1 | 75.9 | - | - |
| ResNet-50 | GFLOPs | 4.105 | 3.587 | 3.123 | 2.383 | 1.891 |
| | Top-1 acc. | 77.0 | 78.2 | 78.0 | 77.4 | 76.7 |
| SE-ResNet-50 | GFLOPs | 4.113 | 3.594 | 3.130 | 2.389 | 1.896 |
| | Top-1 acc. | 77.6 | 78.7 | 78.4 | 77.9 | 77.4 |
| ResNeXt-50 | GFLOPs | 4.250 | - | 3.196 | 2.406 | 1.891 |
| | Top-1 acc. | 78.4 | - | 78.8 | 78.4 | 77.5 |
| ResNet-101 | GFLOPs | 7.822 | 6.656 | 5.625 | 4.012 | - |
| | Top-1 acc. | 78.5 | 79.2 | 79.2 | 78.7 | - |
| ResNeXt-101 | GFLOPs | 7.993 | - | 5.719 | 4.050 | - |
| | Top-1 acc. | 79.4 | - | 79.6 | 78.9 | - |
| ResNet-200 | GFLOPs | 15.044 | 12.623 | 10.497 | 7.183 | - |
| | Top-1 acc. | 79.6 | 80.0 | 79.8 | 79.5 | - |

Table 9: Ablation study on ImageNet in table form corresponding to the plots in Figure 4 in the main paper. Note: All networks are trained with naïve softmax loss without label smoothing [40] or mixup [48]