

Departamento de Física Médica - Centro atómico Bariloche - IB

Redes Convolucionales

Ariel Hernán Curiale
ariel.curiale@cab.cnea.gov.ar



Algunos slides fueron adaptados de Fei Fei Li, J. Johnson y S. Yeung, cs231n, Stanford 2017.

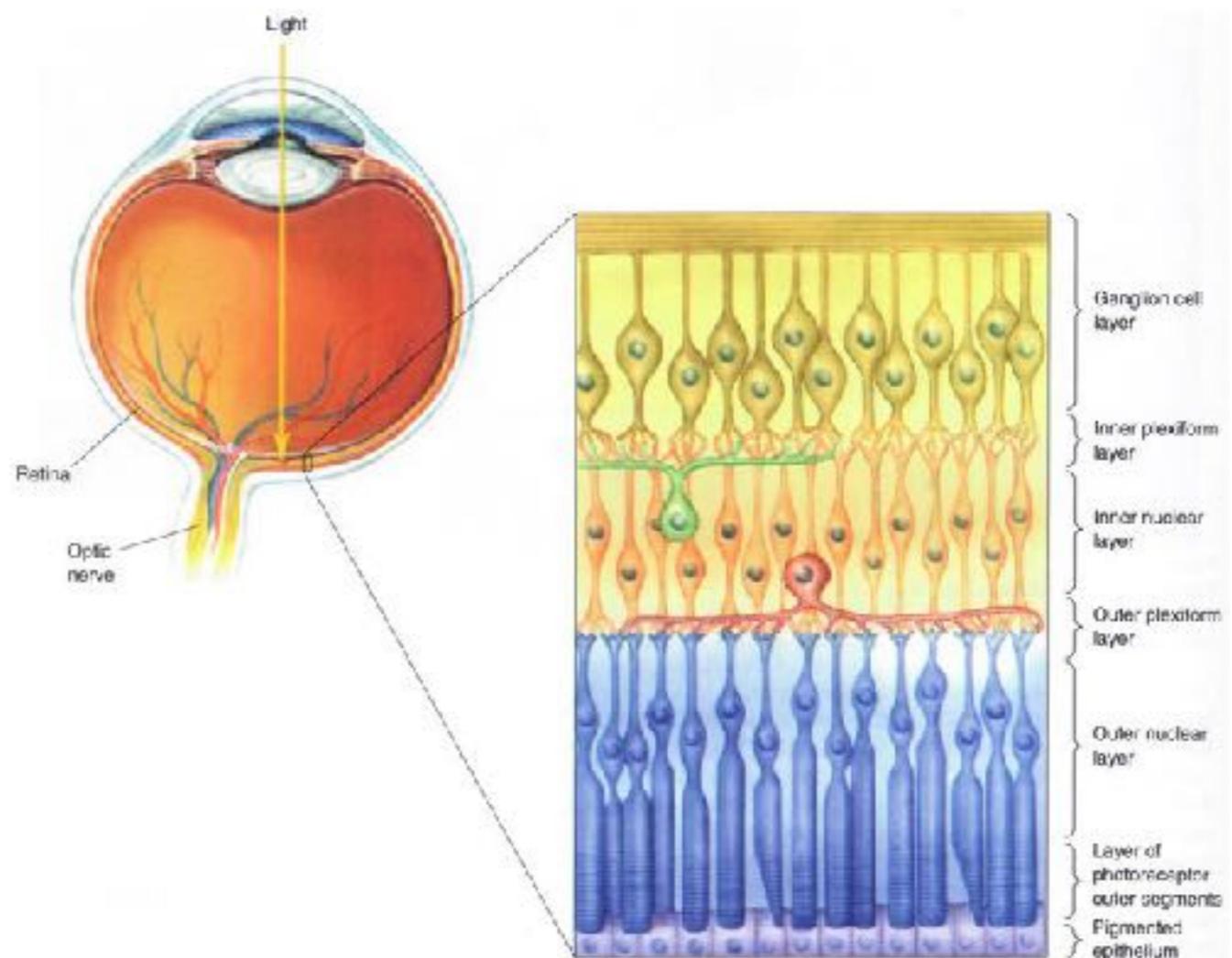
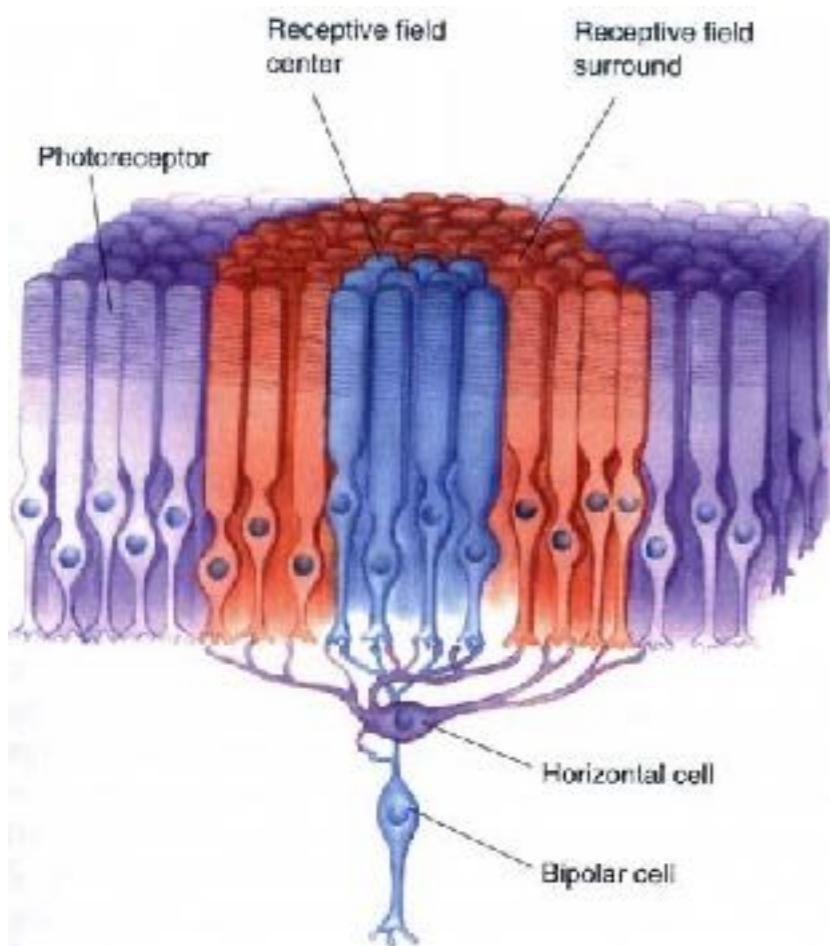


UNCUYO
UNIVERSIDAD
NACIONAL DE CUYO



Redes Convolucionales

- ❖ Convolutional Neural Networks (Fukushima 1980, LeCun, 1989): Este tipo de redes neuronales esta inspirada en la idea de campo receptivo de los sistemas sensoriales.
 - ❖ Por ejemplo, las células ganglionares de la retina pueden ser excitadas o inhibidas dependiendo del lugar del campo receptivo donde pasa el estimulo visual.



Introducción

- ❖ Hubel & Wiesel, 1959 (Premio Nobel 1981): Descubren distintos tipos de células en la corteza visual primaria
 - ❖ Simples: responden a orientación y posición.
 - ❖ Complejas: responden orientación, movimiento y dirección
 - ❖ Hipercomplejas (1965): responden a orientación, movimiento, dirección y longitud (end-points).

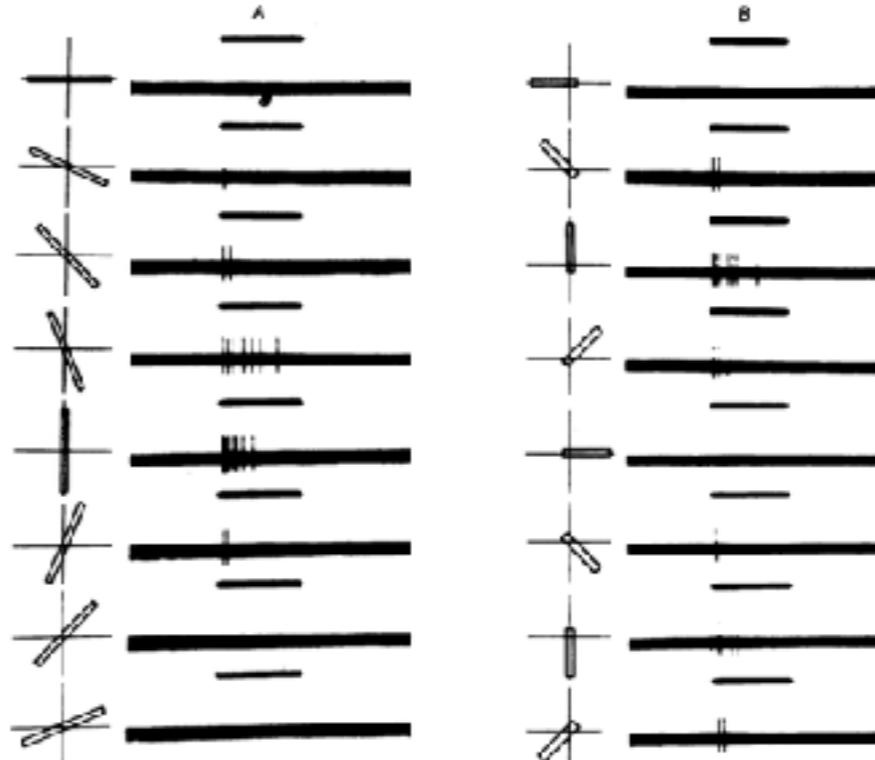
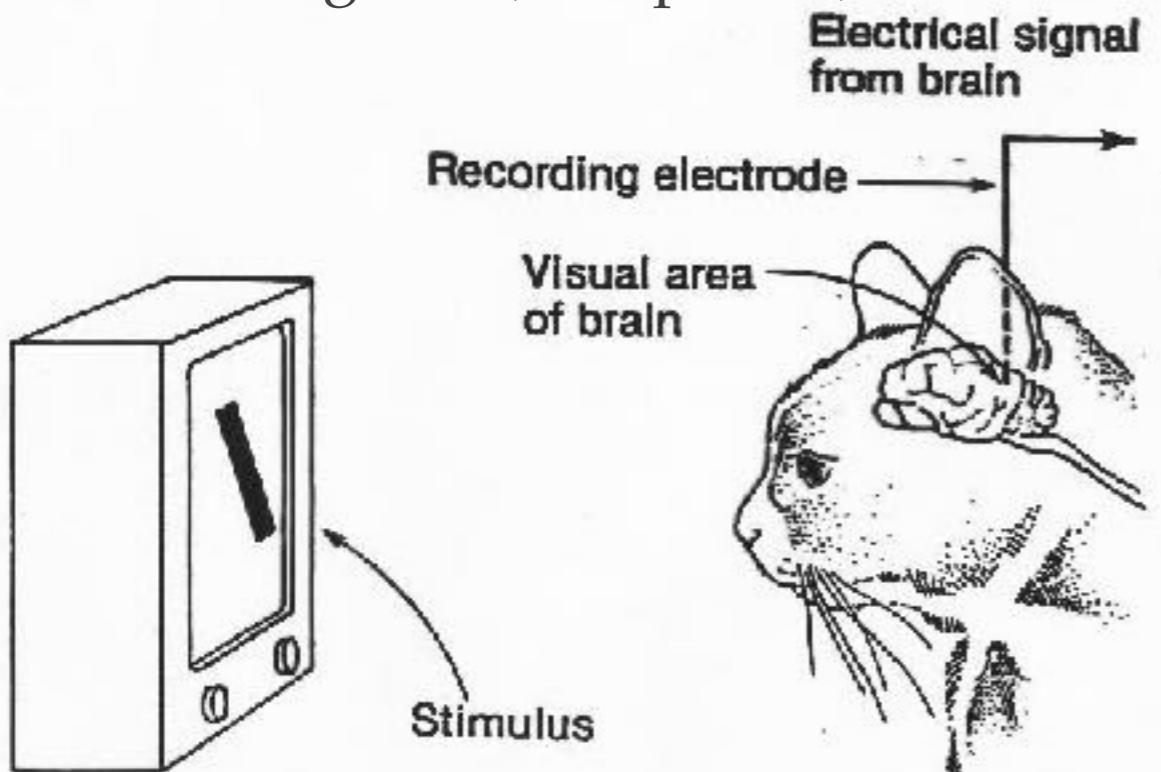
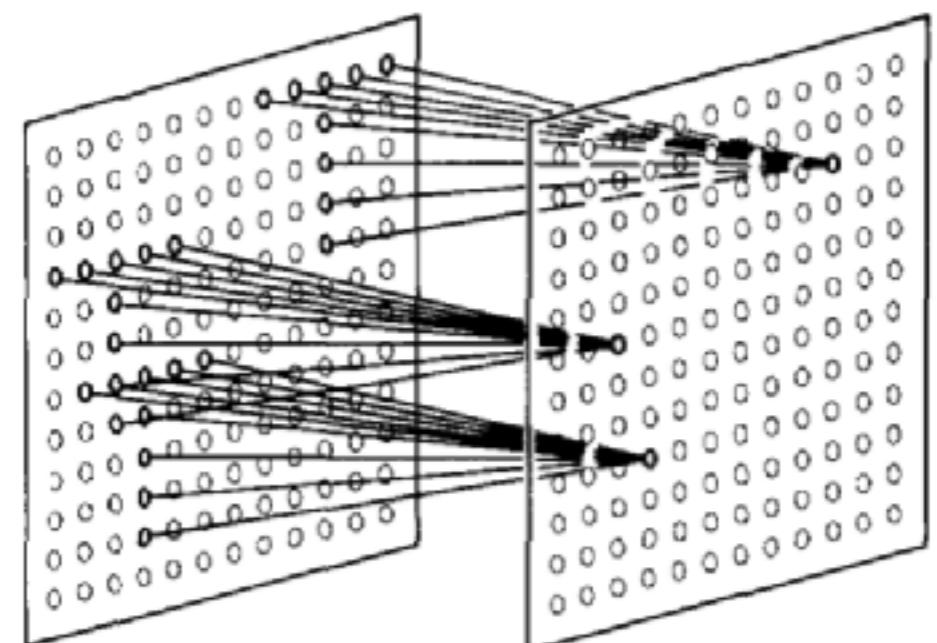
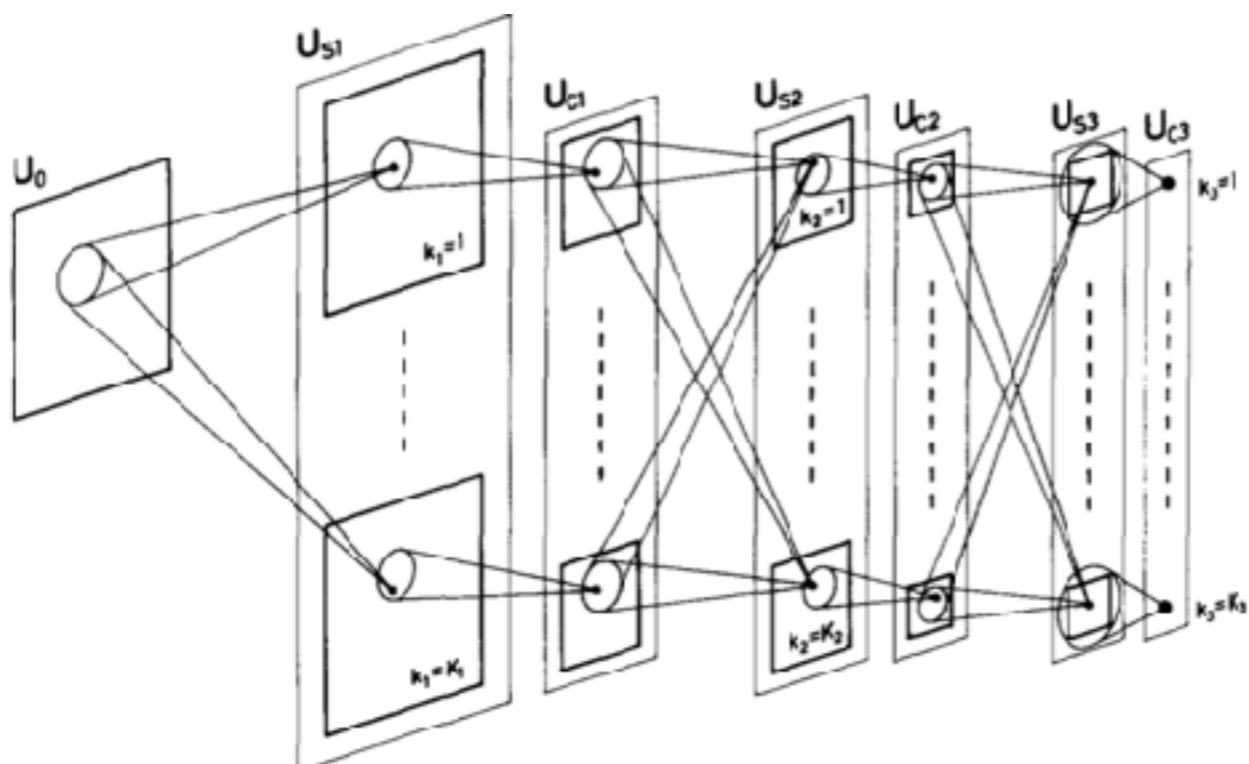


Fig. 3. Same unit as in Fig. 2. A, responses to shining a rectangular light spot, $1'' \times 8''$; centre of slit superimposed on centre of receptive field; successive stimuli rotated clockwise, as shown to left of figure. B, responses to a $1'' \times 8''$ slit oriented in various directions, with one end always covering the centre of the receptive field: note that this central region evoked responses when stimulated alone (Fig. 2a). Stimulus and background intensities as in Fig. 1; stimulus duration 1 sec.

Introducción

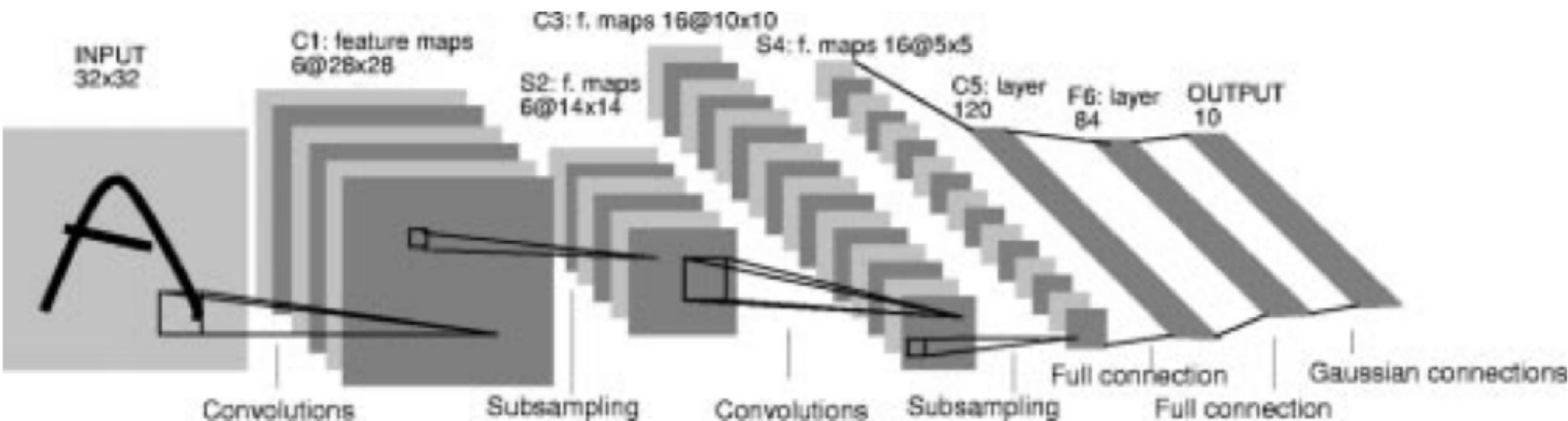
- ❖ Neocognitron [Fukushima 1980] fue la primer rede neuronal en implementar los descubrimientos de Hubel & Wiesel (campo receptivo, neuronas simple y complejas).



K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," Biological cybernetics, vol. 36, no. 4, pp. 193–202, 1980.

Introducción

- ❖ Gradient-based learning applied to document recognition [LeCun, Bottou, Bengio, Haffner 1998] muestran el primer ejemplo de como aplicar backpropagation a las redes convolucionales.



Redes Convolucionales

- ❖ ¿Que ganamos?

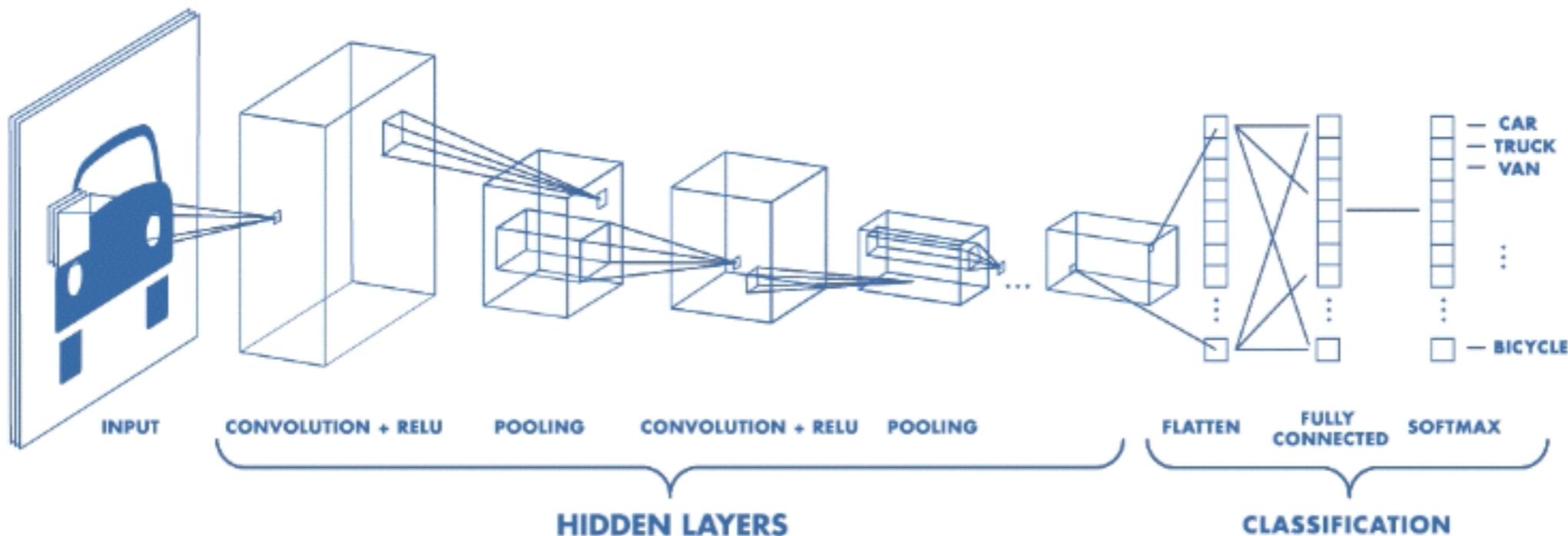
Redes Convolucionales

- ❖ ¿Que ganamos?
 - ❖ Se logra reducir notablemente la cantidad de parámetros a entrenar, pero aún no se cuenta con el poder de cómputo adecuado para ser aplicarlo a problemas interesantes.
- ❖ *Deep learning* [2006s - hoy].

Introducción

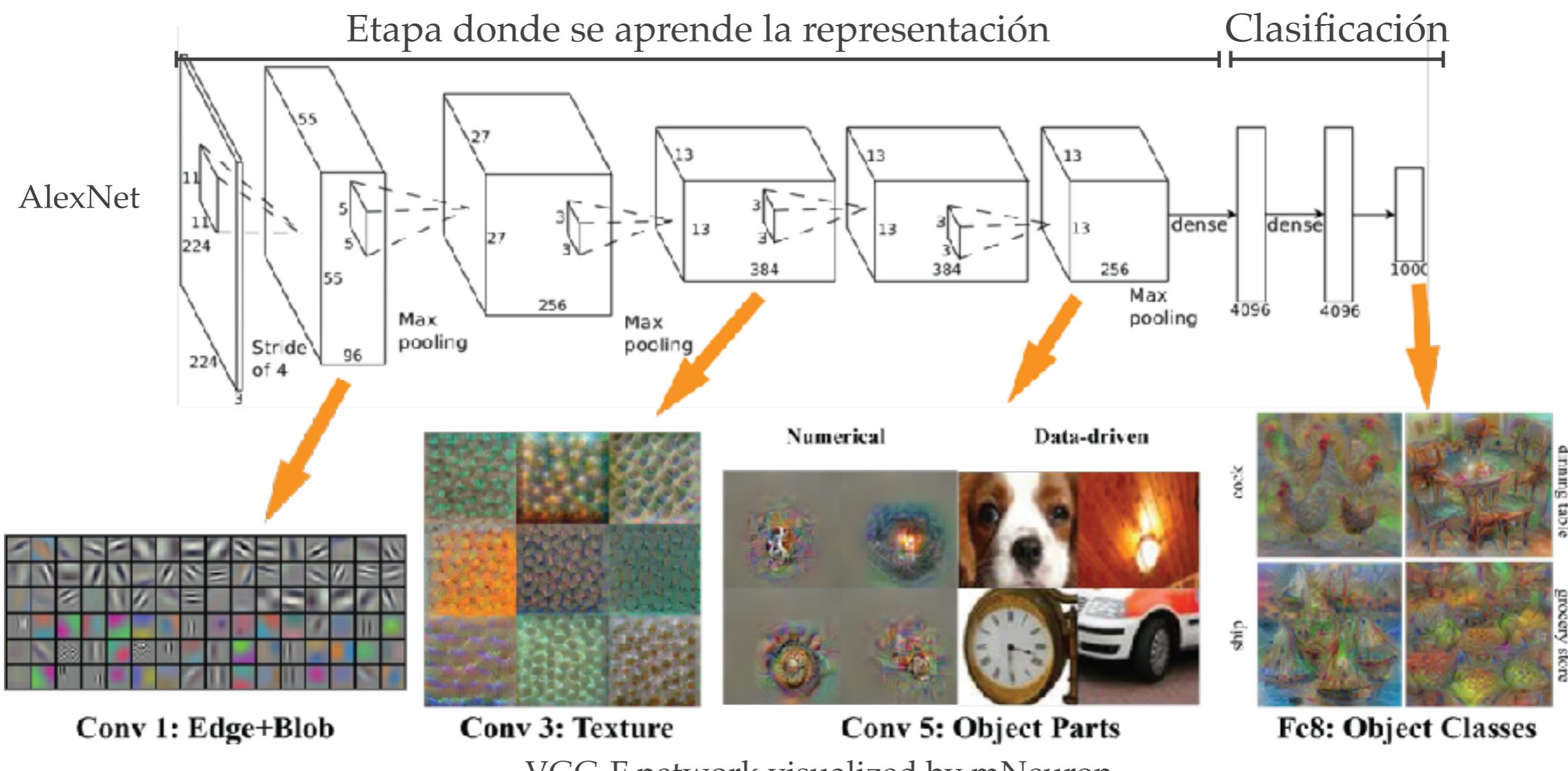
- ❖ Convolutional Neural Networks

- ❖ Cada neurona busca identificar en la capa anterior rasgos específicos.
- ❖ Todas las neuronas de una capa buscan el mismo rasgo pero en una posición diferente.
- ❖ Representation learning: intentamos aprender la representación de los datos y además la relación entre esta representación y la salida (primeras clases).
- ❖ Estas dos etapas se ven bien marcadas en CNN's



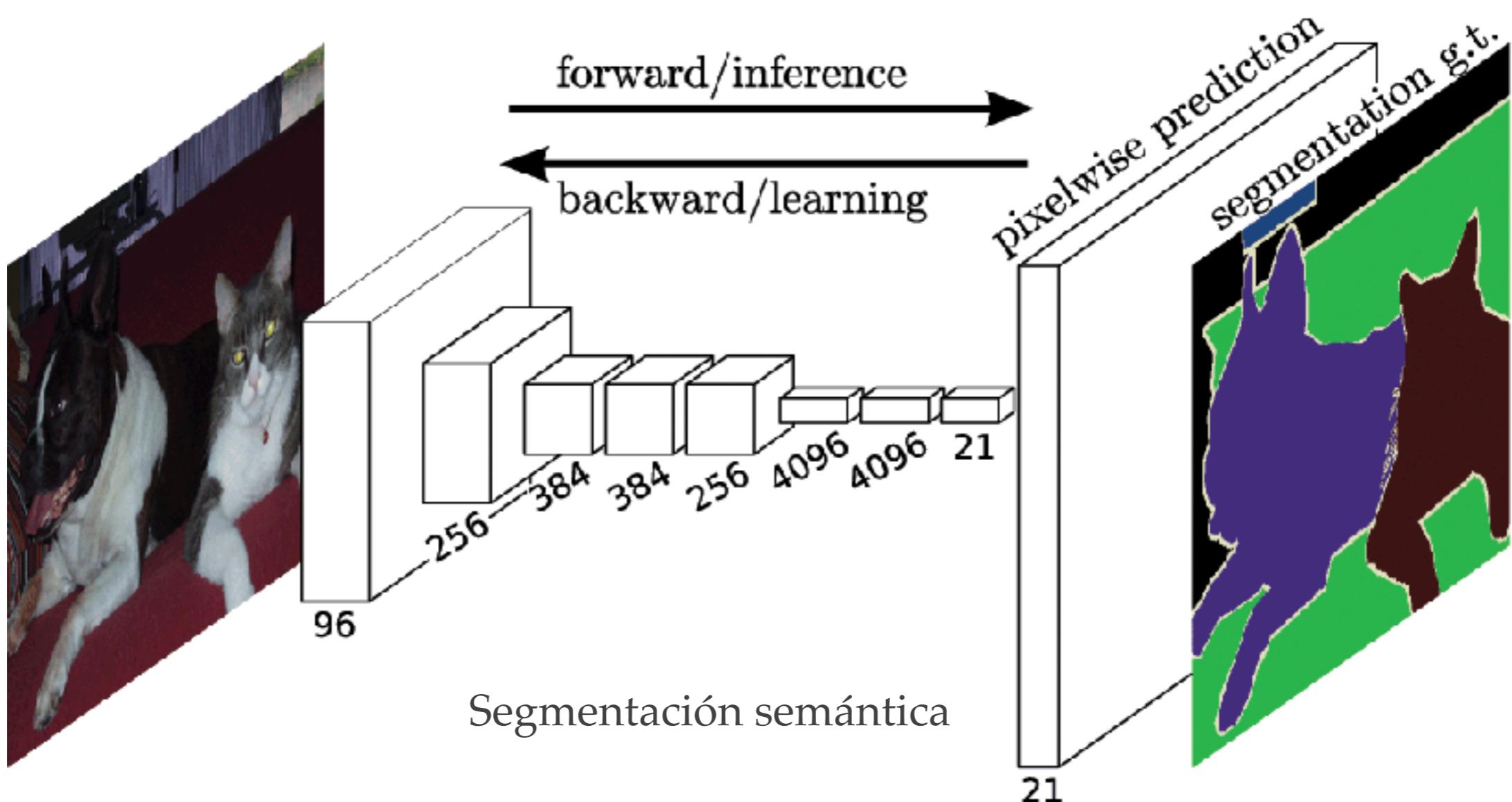
Introducción

- ❖ Imagenet classification with deep convolutional neural networks [Krizhevsky, Sutskever, Hinton, 2012]



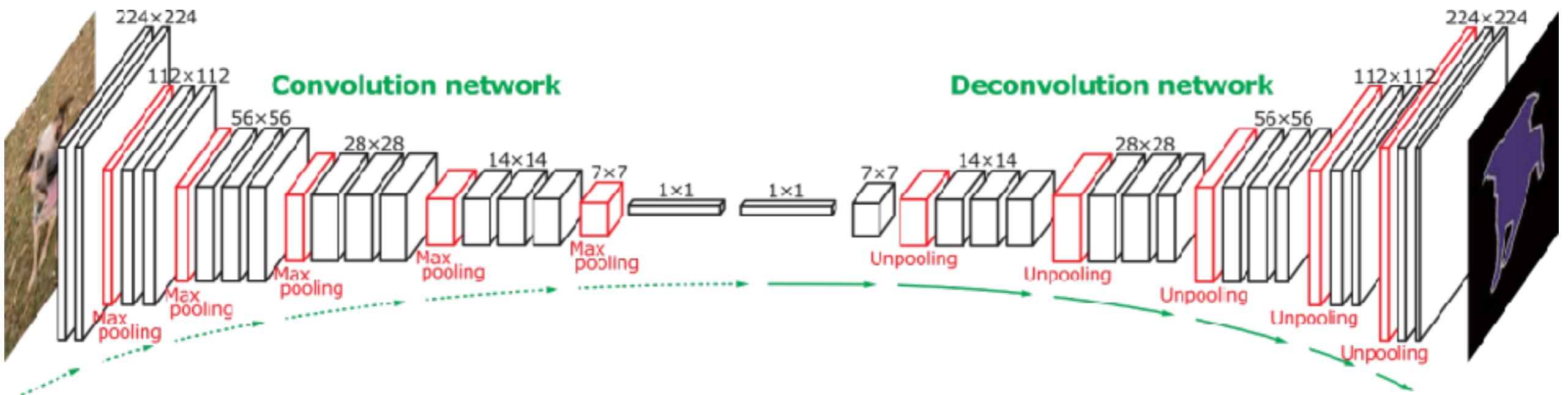
Introducción

- ❖ Fully convolutional neural networks (predicciones densas).



Introducción

- ❖ Fully convolutional neural networks (autoencoder).

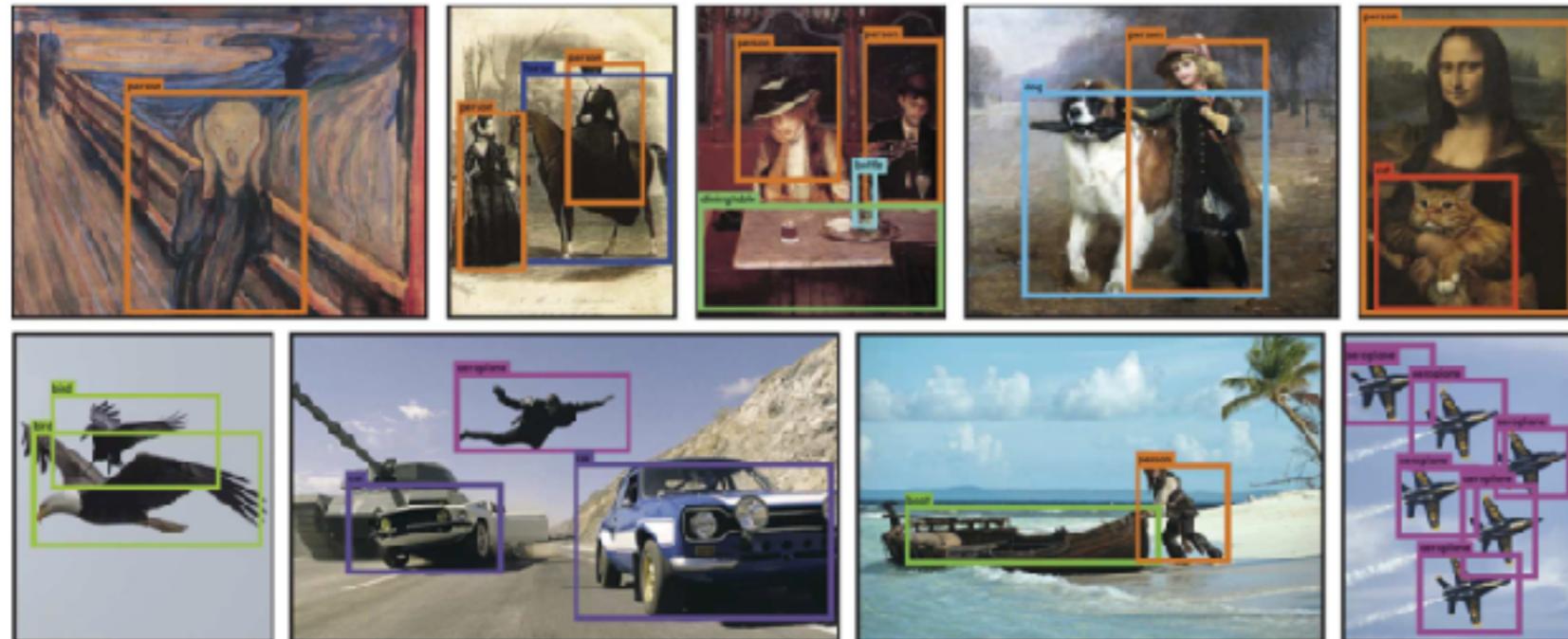


Clasificación

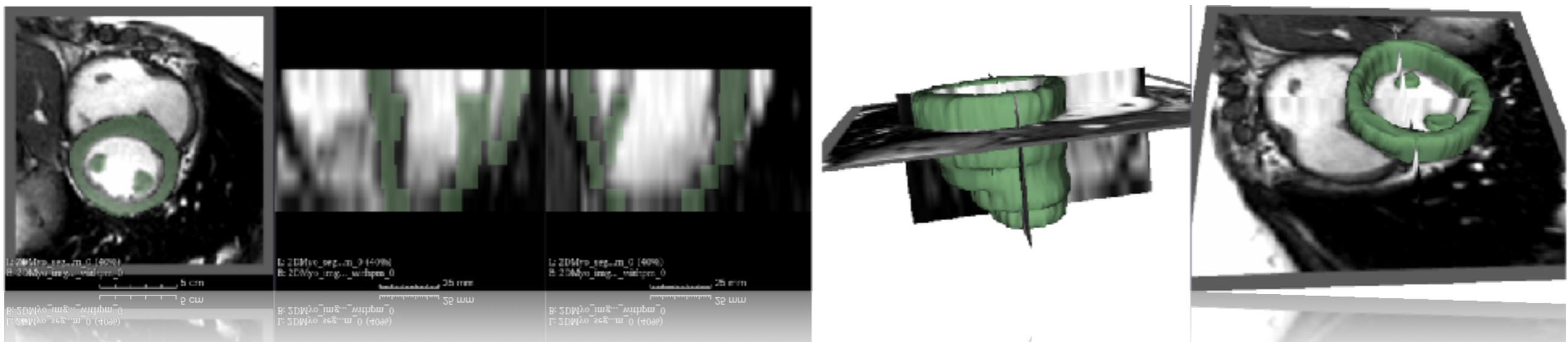


A. Krizhevsky, I. Sutskever, and G. E. Hinton,
"Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, pp. 1097–1105, 2012.

Detección y Segmentación

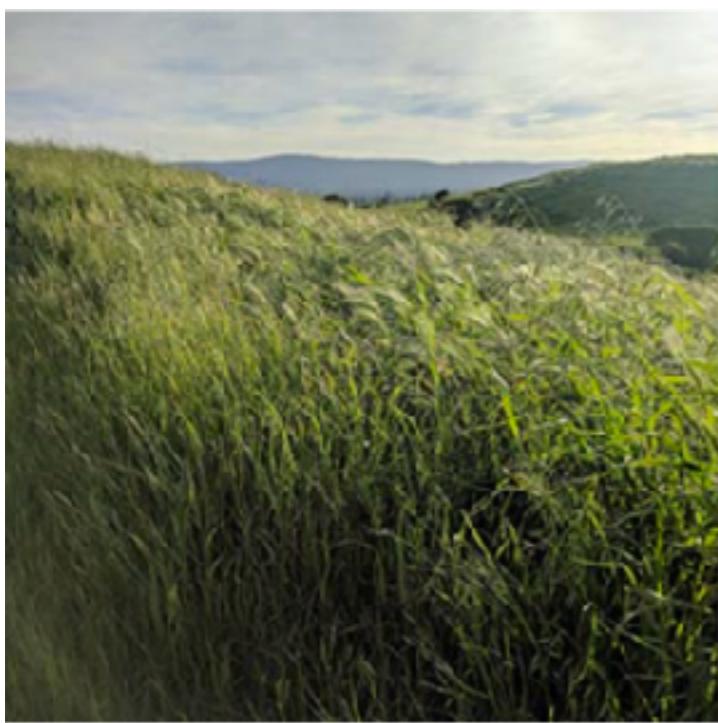
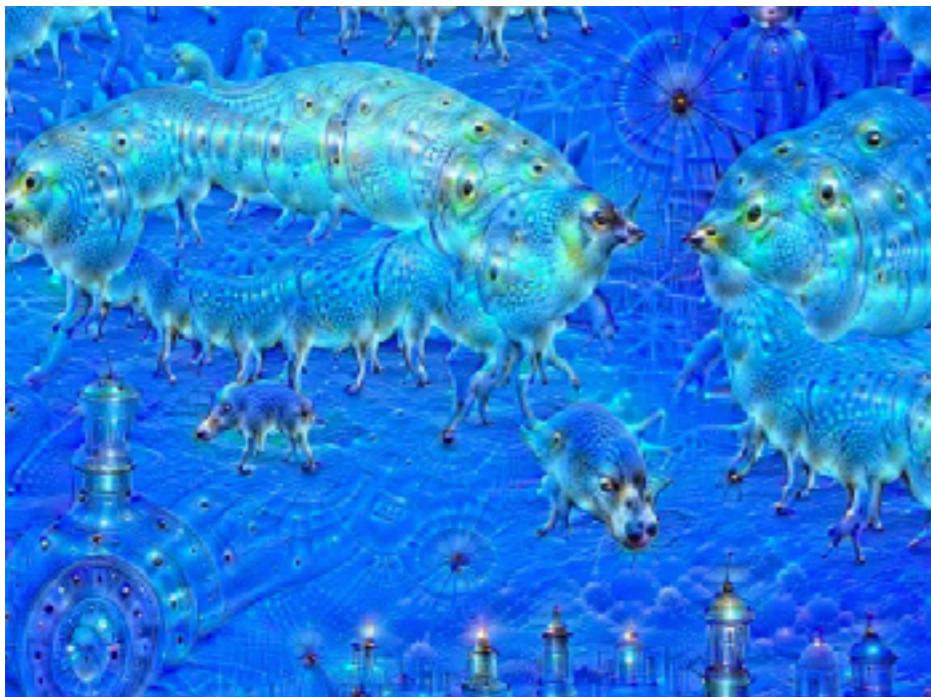


J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779– 788, 2016.



A. H. Curiale, F. D. Colavecchia, P. Kaluza, R. A. Isoardi, and G. Mato, "Automatic myocardial segmentation by using a deep learning network in cardiac mri," in 2017 XLIII Latin American Computer Conference (CLEI), pp. 1–6, Sept. 2017.

DeepDream



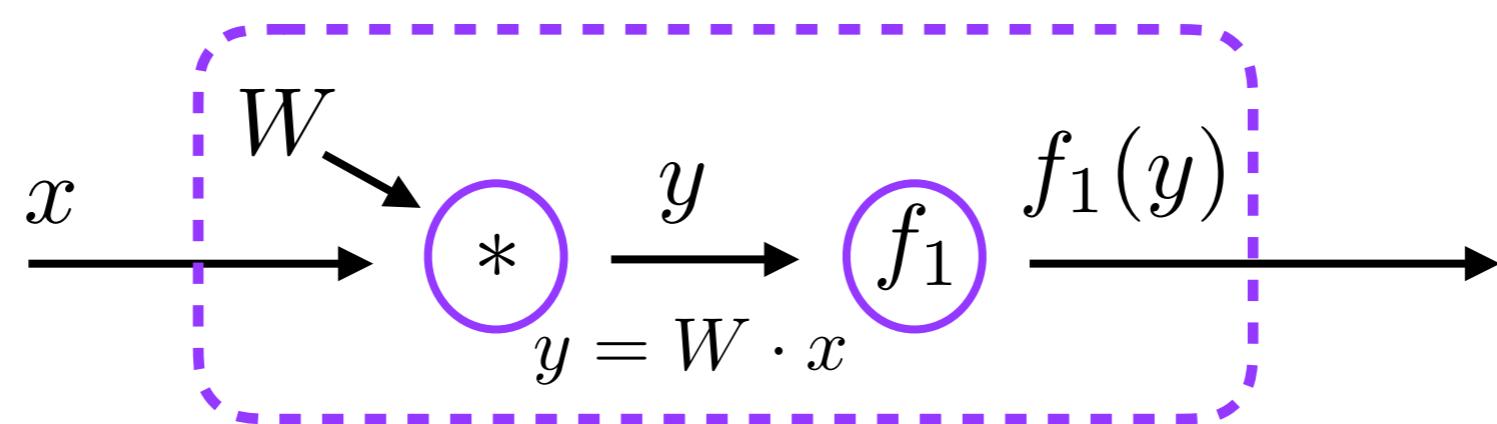
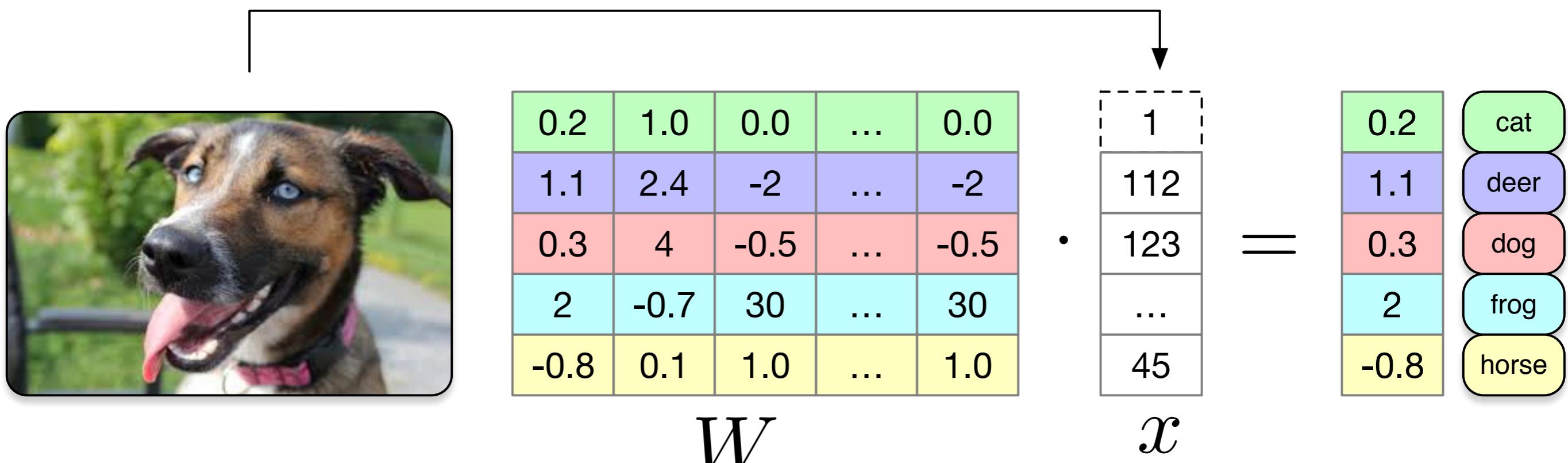
Source: Chollet F., Deep Learning with Python

Style Transfer

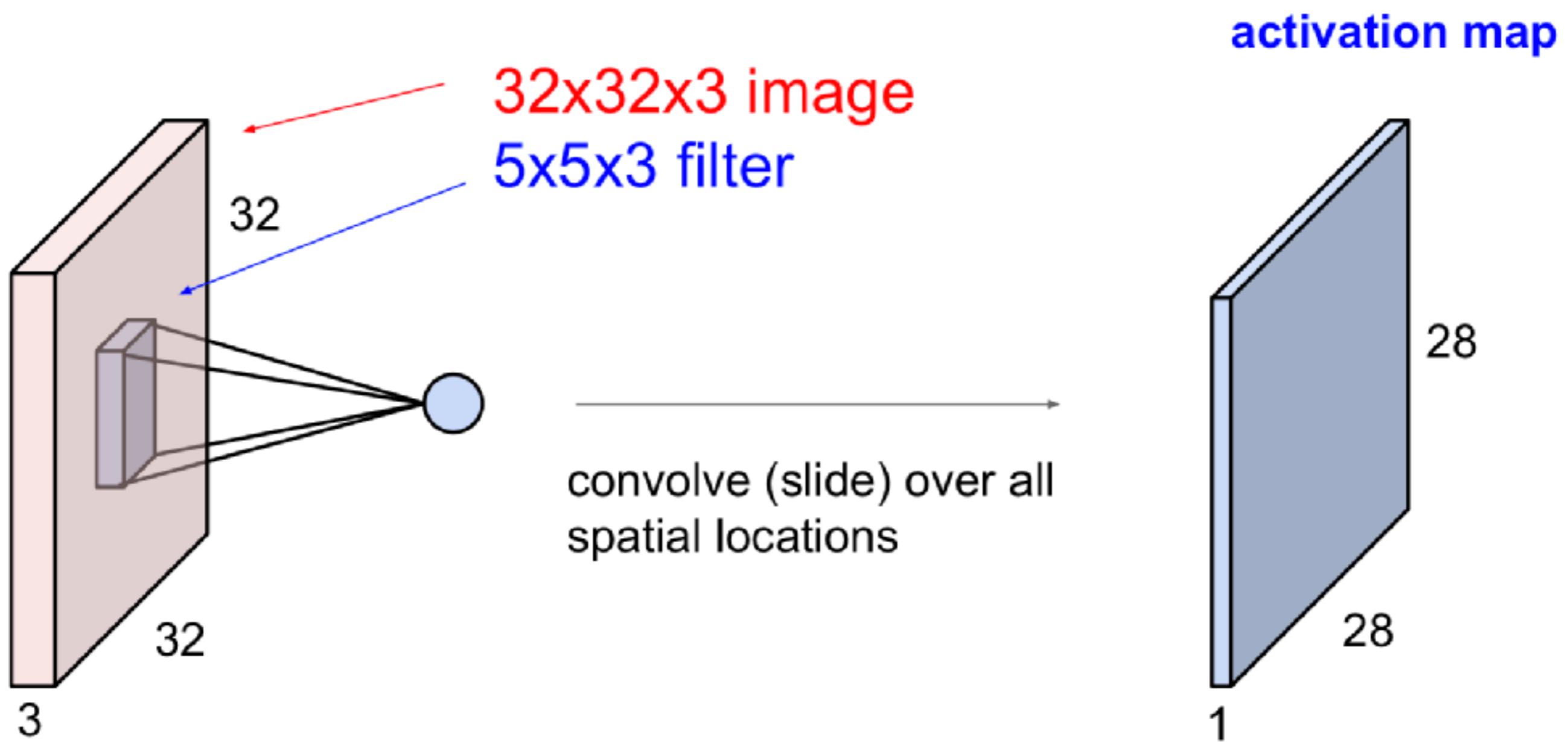


Source: Chollet F., Deep Learning with Python

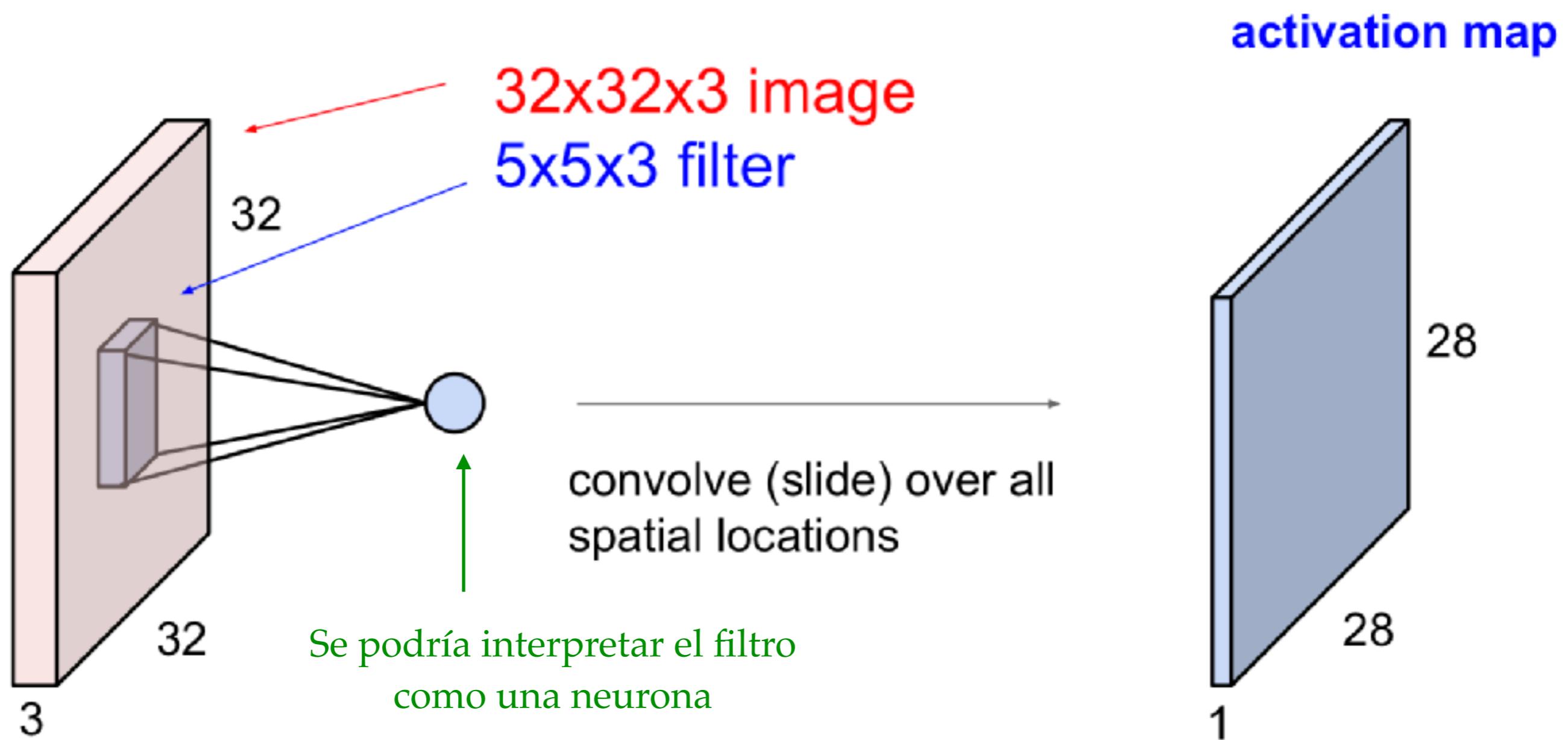
Capa densa o totalmente conectadas



Capa convolucional

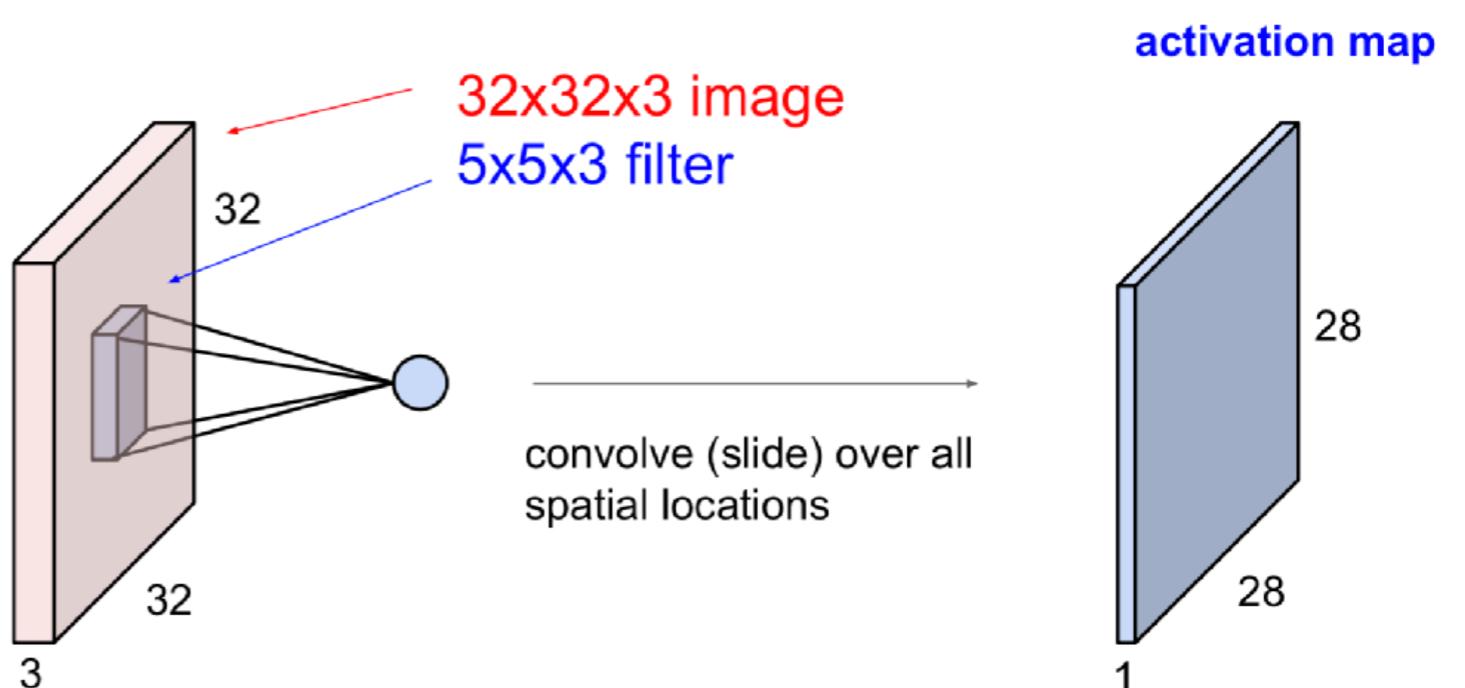


Capa convolucional

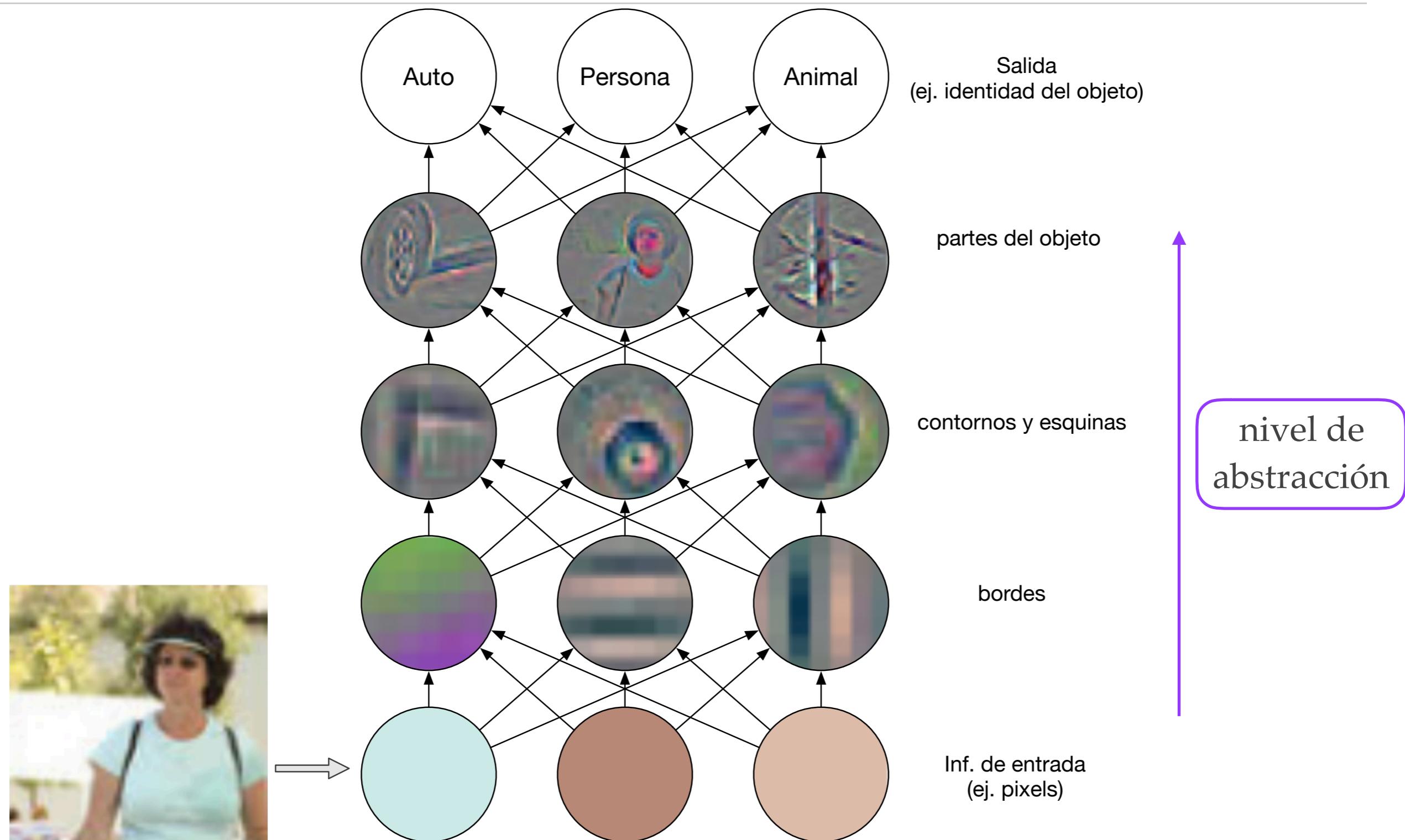


Mapa de activación de una capa convolucional

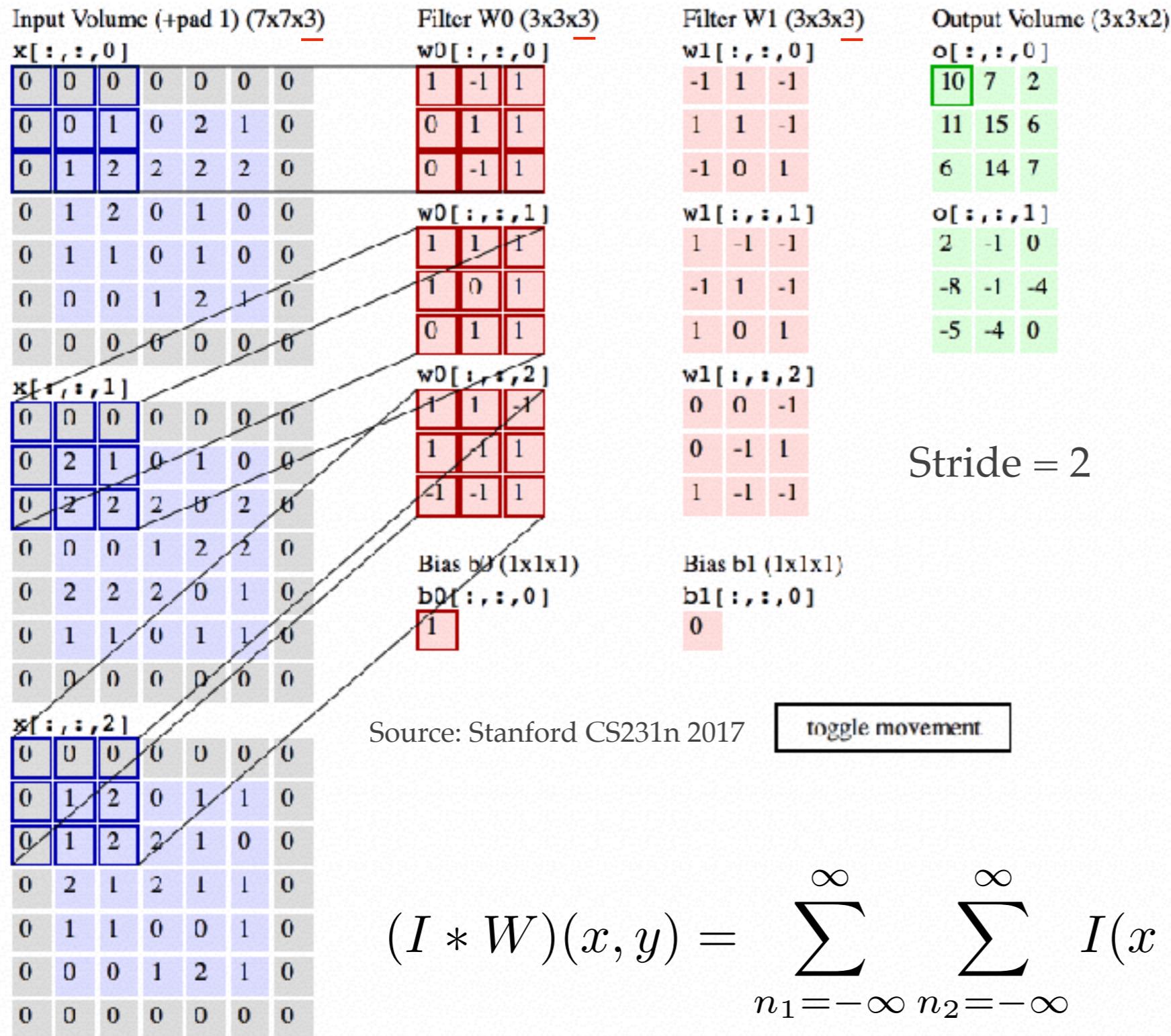
- ❖ Mapa de activación: representa la activación de las neuronas donde cada una:
 - ❖ Las neuronas comparte los pesos y el bias
 - ❖ Comparten la función de activación y el tamaño de campo receptivo.



Características de las capas convolucionales



Capa convolucional



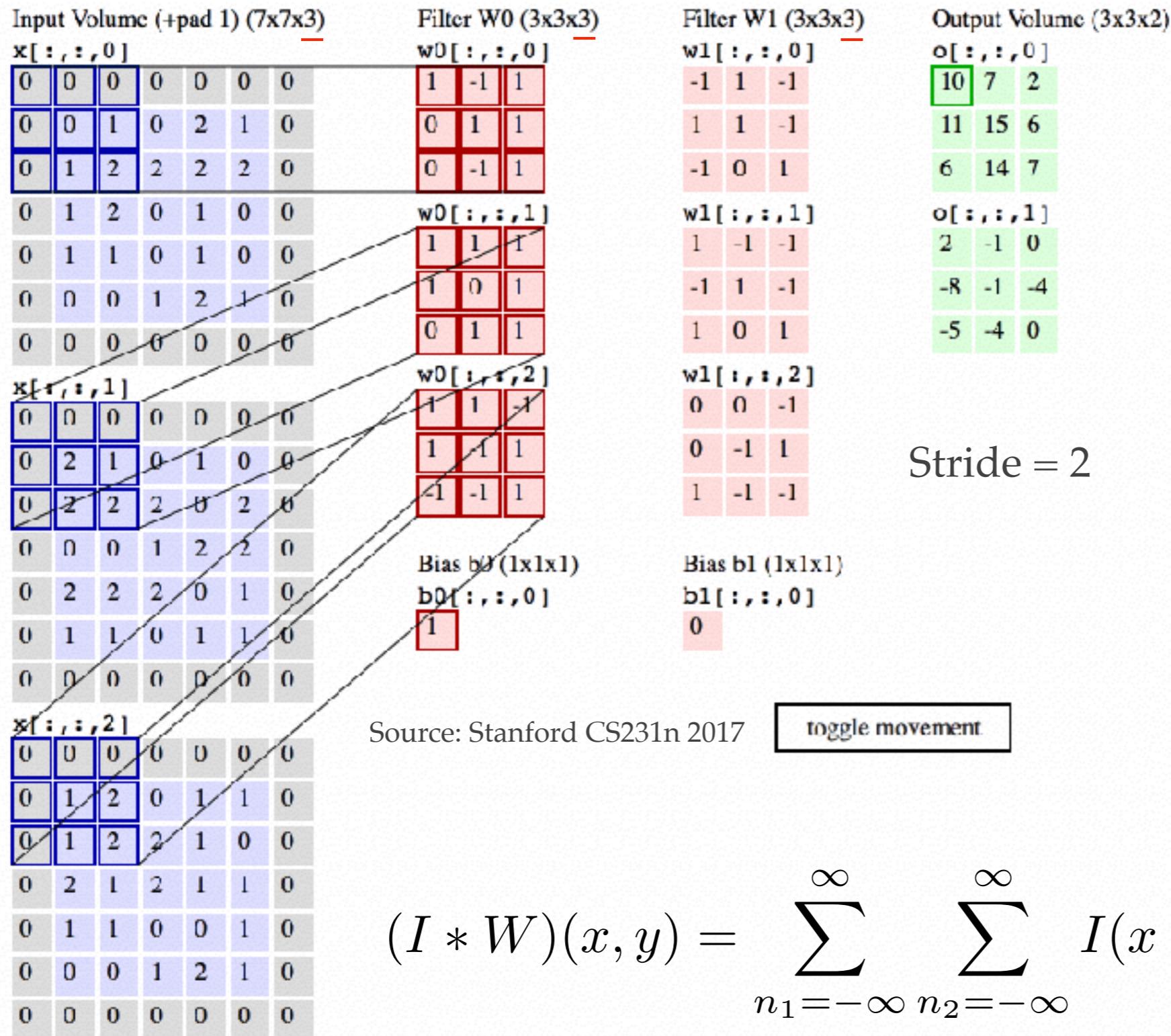
Tamaño de Salida:

$$\frac{N - K}{\text{stride}} + 1$$

Stride = 2

$$(I * W)(x, y) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} I(x - n_1, y - n_2)W(n_1, n_2)$$

Capa convolucional



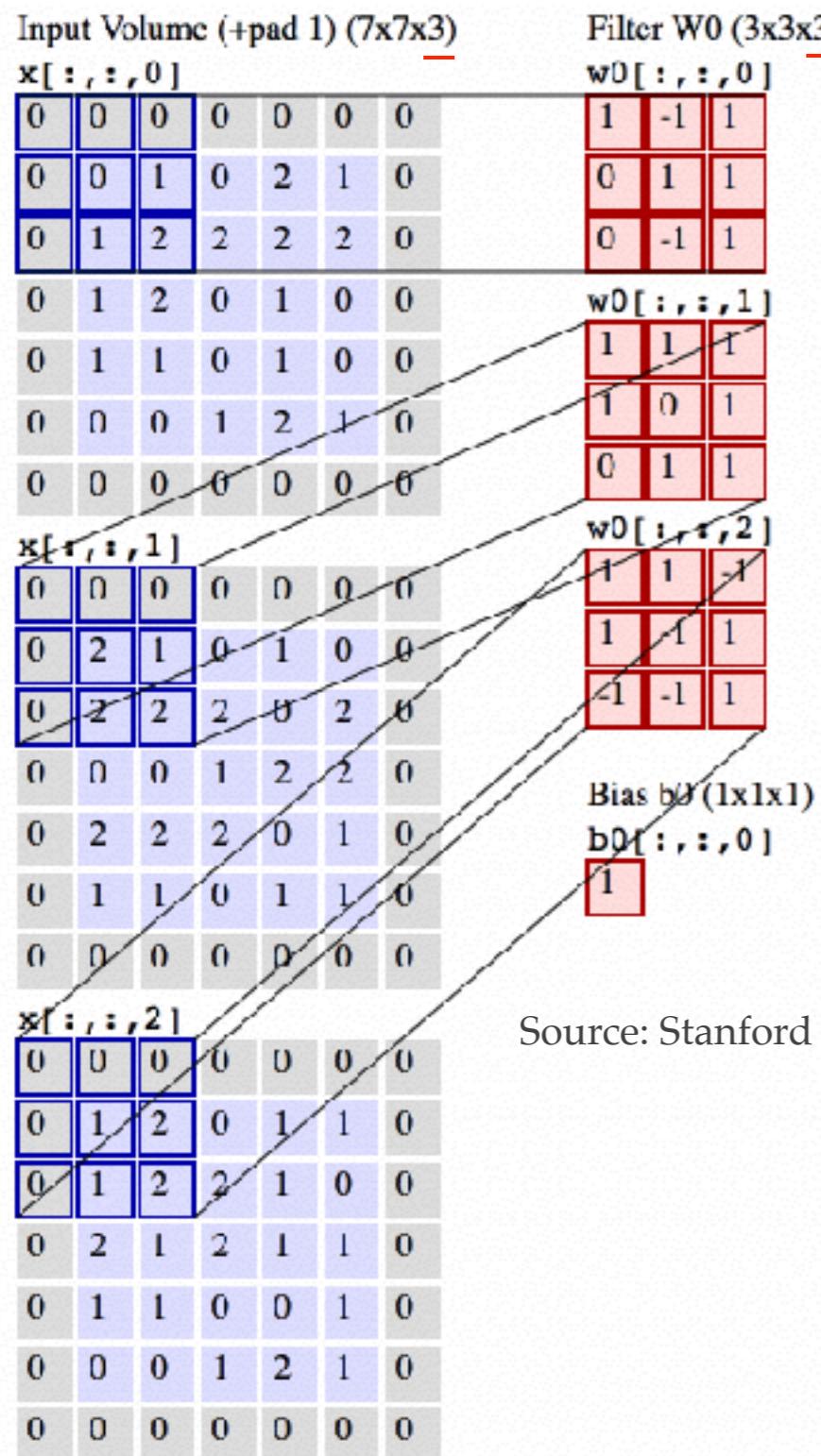
Tamaño de Salida:

$$\frac{N - K}{\text{stride}} + 1$$

Stride = 2

$$(I * W)(x, y) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} I(x - n_1, y - n_2)W(n_1, n_2)$$

Capa convolucional



Source: Stanford CS231n 2017

toggle movement

Tamaño de Salida:

$$\frac{N - K}{\text{stride}} + 1$$

Eje.:

Input: 50x50x3

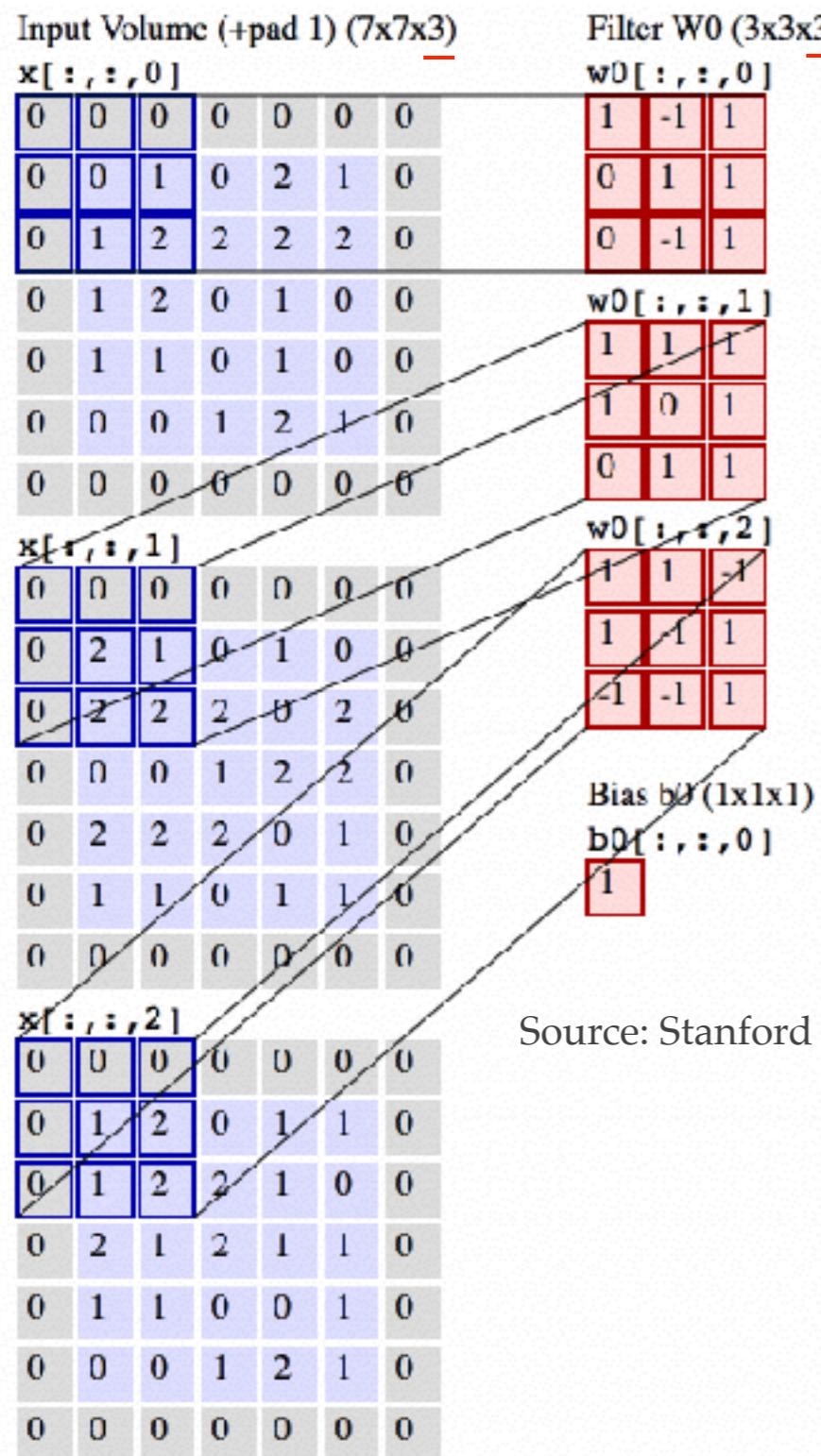
K: 5x5x3

Stride: 3

Zero padding

Output:

Capa convolucional



Source: Stanford CS231n 2017

toggle movement

Tamaño de Salida:

$$\frac{N - K}{\text{stride}} + 1$$

Eje.:

Input: 50x50x3

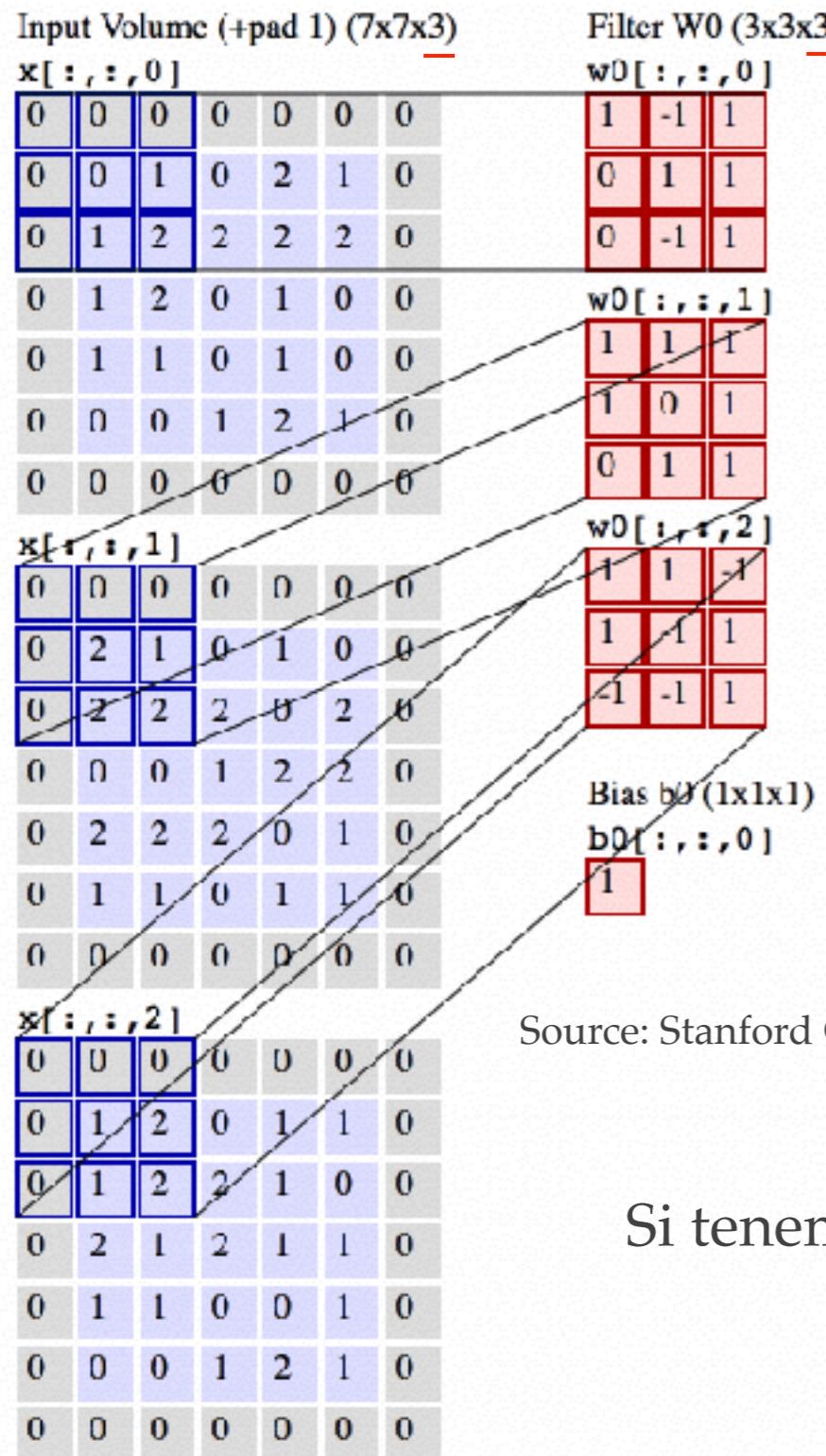
K: 5x5x3

Stride: 3

Zero padding

Output: 16x16x1

Capa convolucional



Source: Stanford CS231n 2017

toggle movement

Tamaño de Salida:

$$\frac{N - K}{\text{stride}} + 1$$

Eje.:

Input: 50x50x3

K: 5x5x3

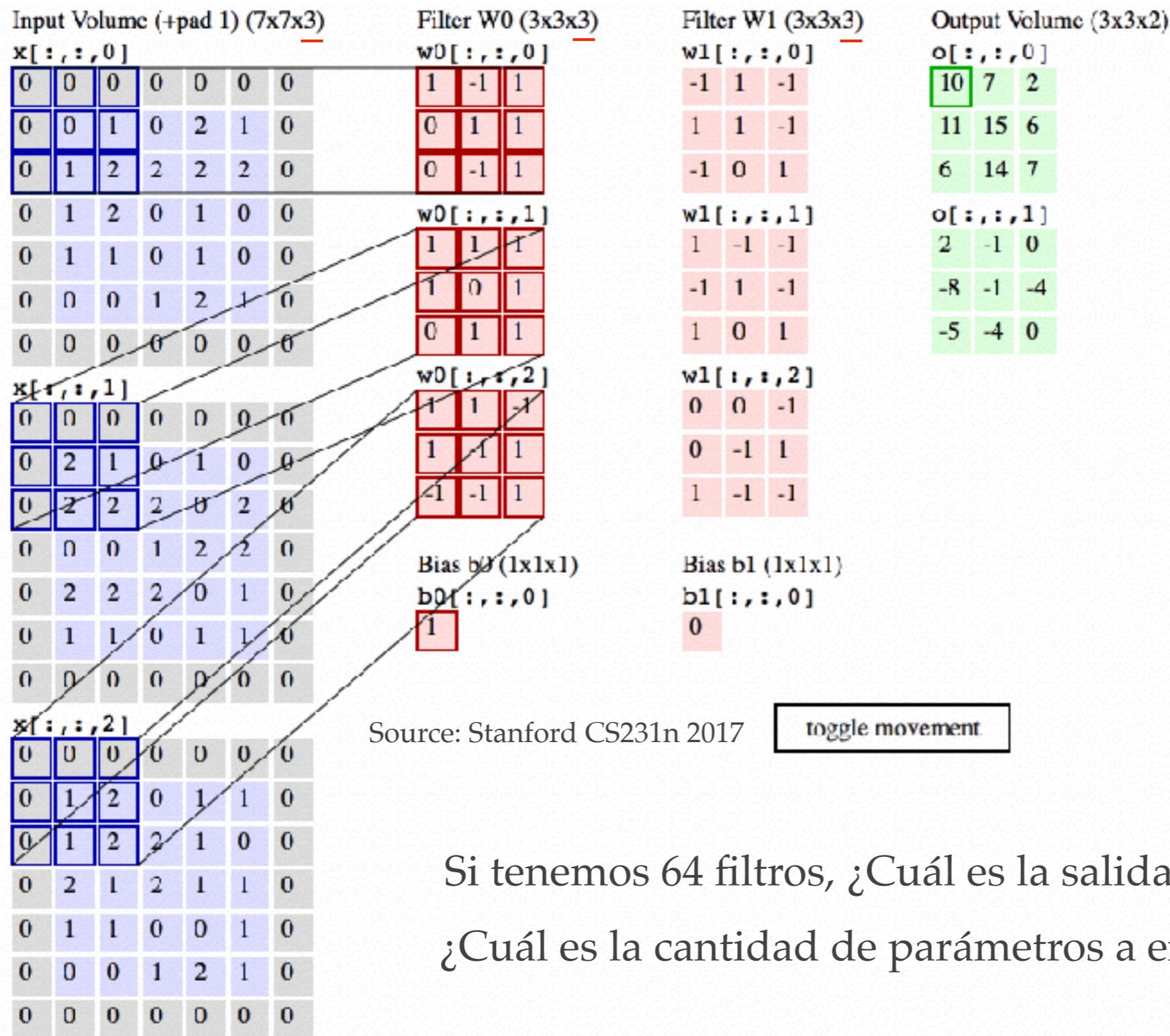
Stride: 3

Zero padding

Output: 16x16x1

Si tenemos 64 filtros, ¿Cuál es la salida?

Capa convolucional



Tamaño de Salida:

$$\frac{N - K}{\text{stride}} + 1$$

Eje.:

Input: 50x50x3

K: 5x5x3

Stride: 3

Zero padding

Output: 16x16x1

Si tenemos 64 filtros, ¿Cuál es la salida?

¿Cuál es la cantidad de parámetros a entrenar?

Capa convolucional

- ❖ Hiperparámetros: Tamaño del kernel, cantidad de filtros, si usa bias, la función de activación, el stride, si usa padding y cuanto, ...
- ❖ Valores típicos:
 - ❖ La cantidad de filtros normalmente es potencia de 2: 32, 64, 128, 512, 1024, etc.
 - ❖ Tamaño del kernel: 1, 3, 5, 7 (normalmente impar)
 - ❖ Stride: 1, 2
 - ❖ Zero Padding: 0, 1, 2 (acorde con el tamaño del kernel)

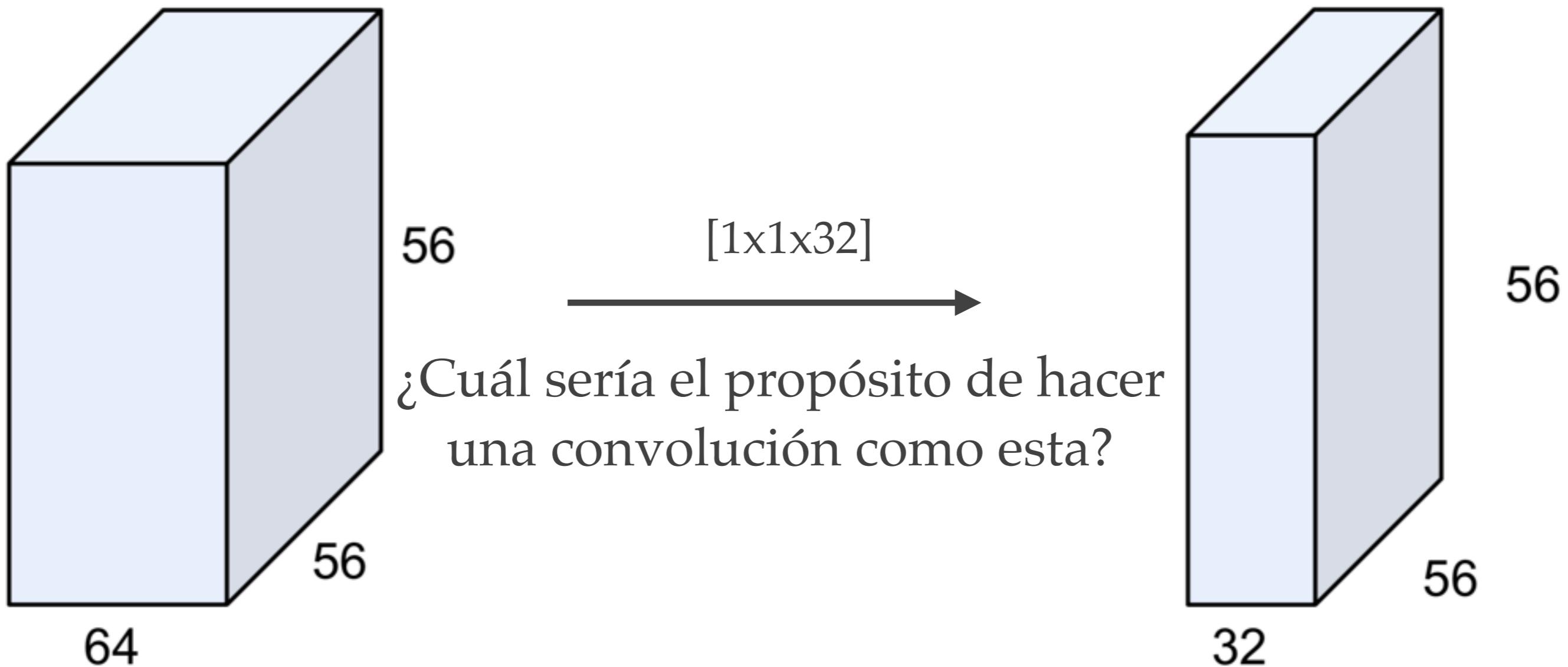
Capa convolucional

- ❖ Cantidad de parámetros para una entrada $N \times M \times D$:
 - ❖ kernel size: $K \times K \times D$ con $K = 1, 3, 5, \dots$
 - ❖ Cant. de filtros: W
 - ❖ Cant. de pesos total: $(K * K * D) * W$
 - ❖ Cant. de bias total: W
- ❖ Salida de H filtros para una entrada de $N \times M \times D$ con un kernel $K \times K \times D$ con stride S (N y M ya contemplan el padding $2P$):
 - ❖ $[(N - K)/S + 1] \times [(M - K)/S + 1] \times H$

Capa convolucional

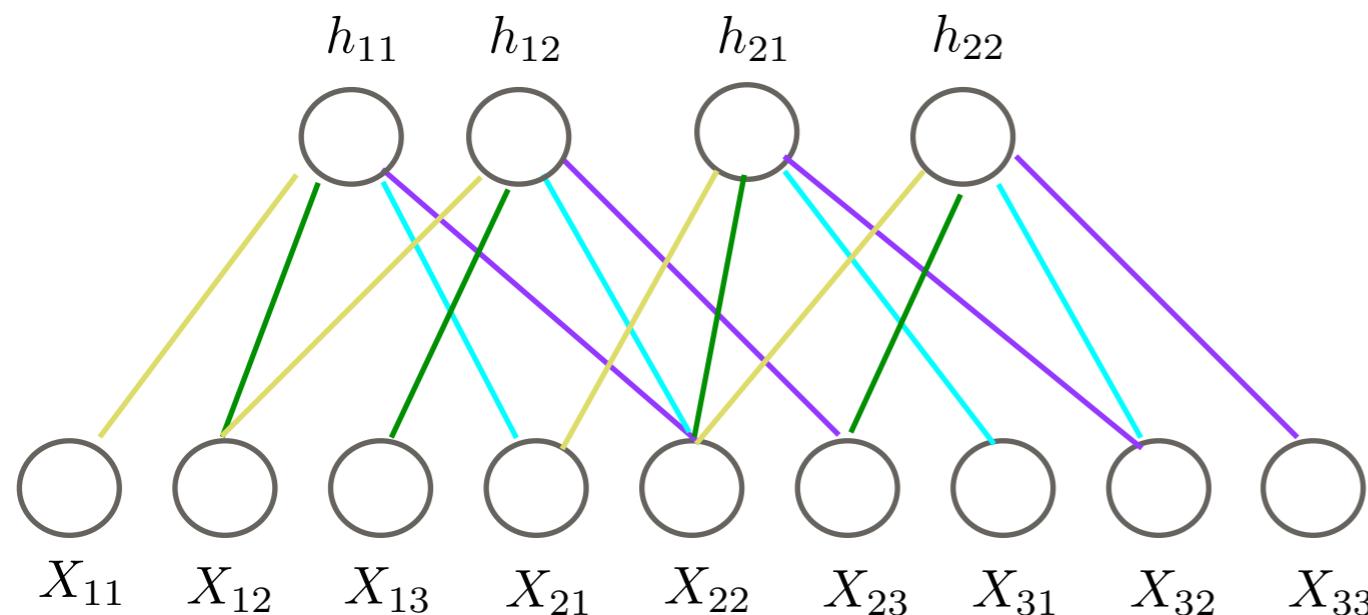
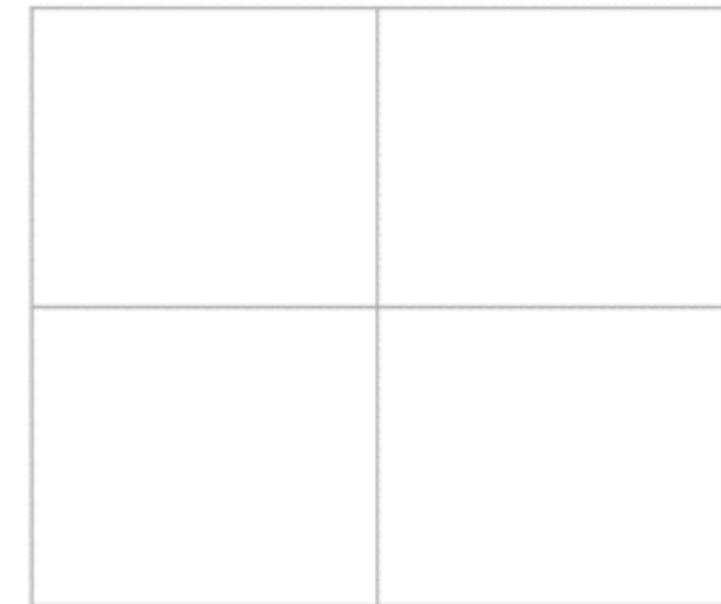
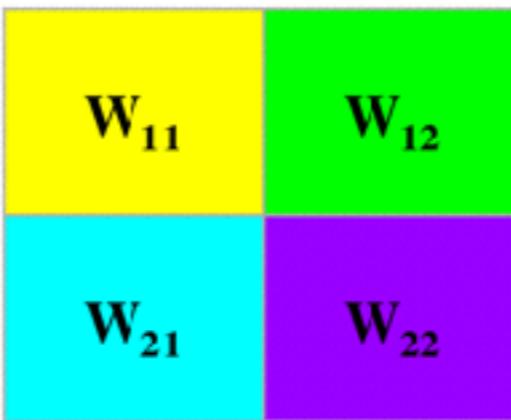
- ❖ Cantidad de parámetros para una entrada $N \times M \times D$:
 - ❖ kernel size: $K \times K \times D$ con $K = 1, 3, 5, \dots$
 - ❖ Cant. de filtros: W
 - ❖ Cant. de pesos total: $(K * K * D) * W$
 - ❖ Cant. de bias total: W El bias es único por filtro
- ❖ Salida de H filtros para una entrada de $N \times M \times D$ con un kernel $K \times K \times D$ con stride S (N y M ya contemplan el padding $2P$):
 - ❖ $[(N - K)/S + 1] \times [(M - K)/S + 1] \times H$

Capa convolucional



Capa convolucional: Backpropagation

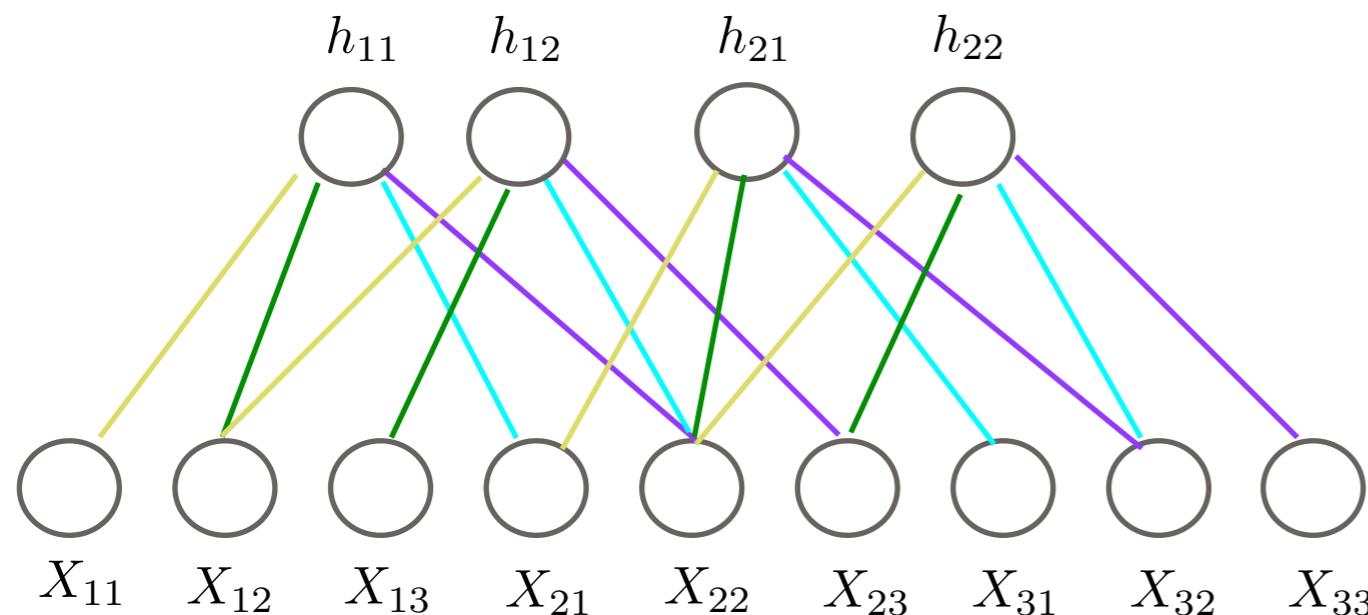
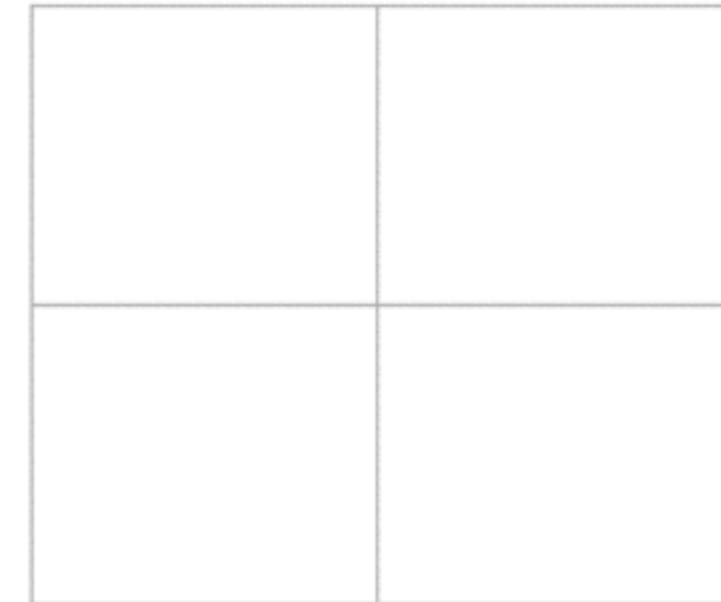
X_{11}	X_{12}	X_{13}
X_{21}	X_{22}	X_{23}
X_{31}	X_{32}	X_{33}



$$h_{11} = W_{11}X_{11} + W_{12}X_{12} + W_{21}X_{21} + W_{22}X_{22}$$
$$h_{12} = W_{11}X_{12} + W_{12}X_{13} + W_{21}X_{22} + W_{22}X_{23}$$
$$h_{21} = W_{11}X_{21} + W_{12}X_{22} + W_{21}X_{31} + W_{22}X_{32}$$
$$h_{22} = W_{11}X_{22} + W_{12}X_{23} + W_{21}X_{32} + W_{22}X_{33}$$

Capa convolucional: Backpropagation

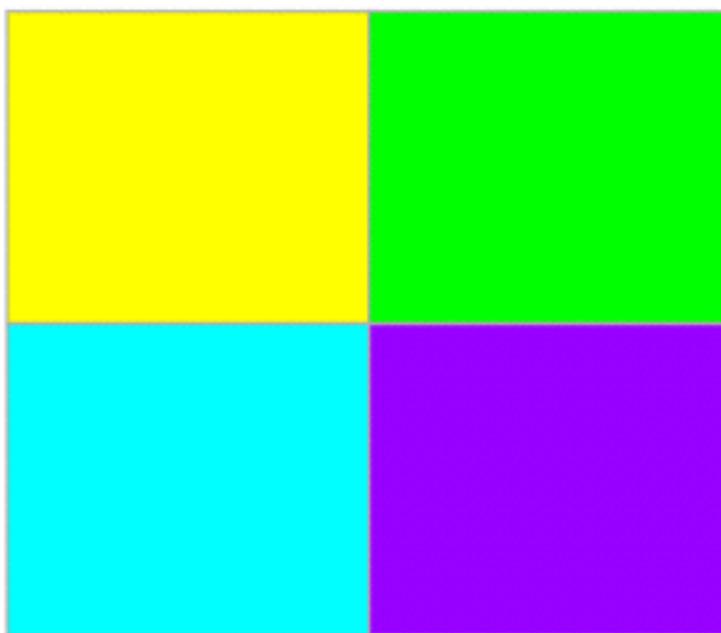
X_{11}	X_{12}	X_{13}
X_{21}	X_{22}	X_{23}
X_{31}	X_{32}	X_{33}



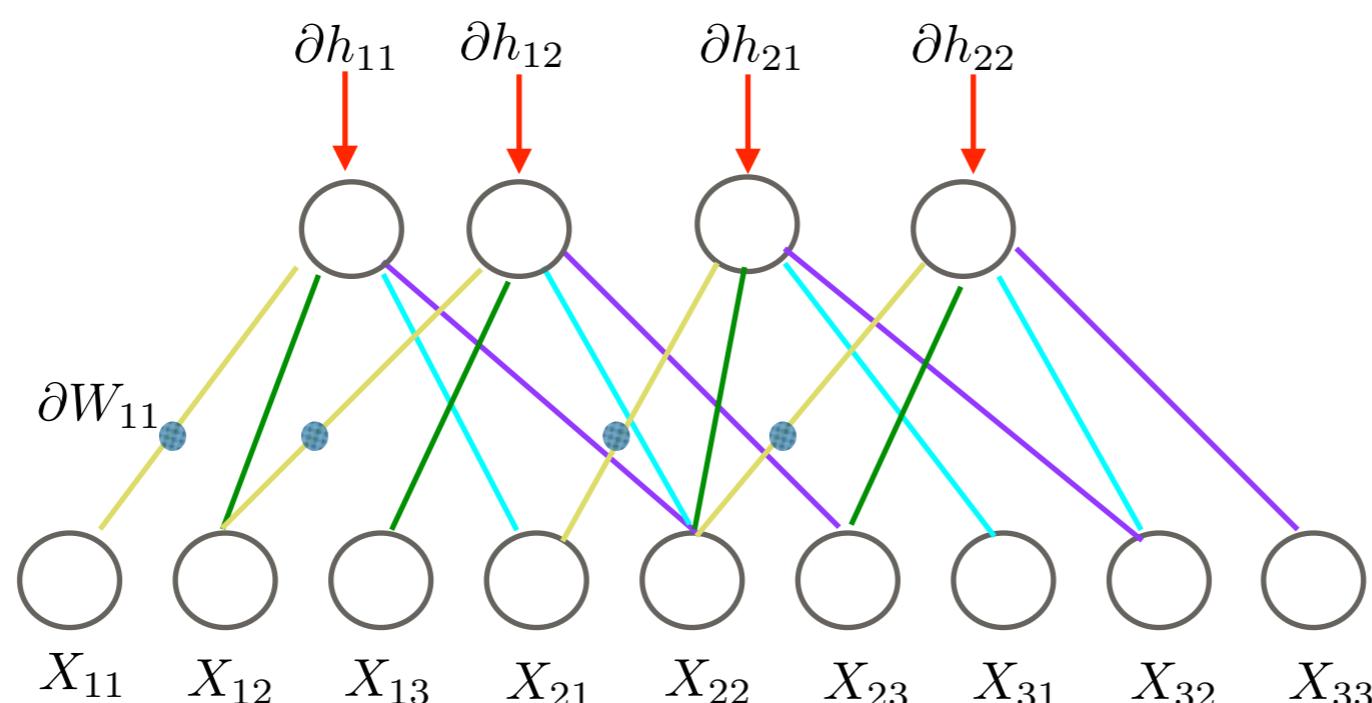
$$h_{11} = W_{11}X_{11} + W_{12}X_{12} + W_{21}X_{21} + W_{22}X_{22}$$
$$h_{12} = W_{11}X_{12} + W_{12}X_{13} + W_{21}X_{22} + W_{22}X_{23}$$
$$h_{21} = W_{11}X_{21} + W_{12}X_{22} + W_{21}X_{31} + W_{22}X_{32}$$
$$h_{22} = W_{11}X_{22} + W_{12}X_{23} + W_{21}X_{32} + W_{22}X_{33}$$

Capa convolucional: Backpropagation

X_{11}	X_{12}	X_{13}
X_{21}	X_{22}	X_{23}
X_{31}	X_{32}	X_{33}



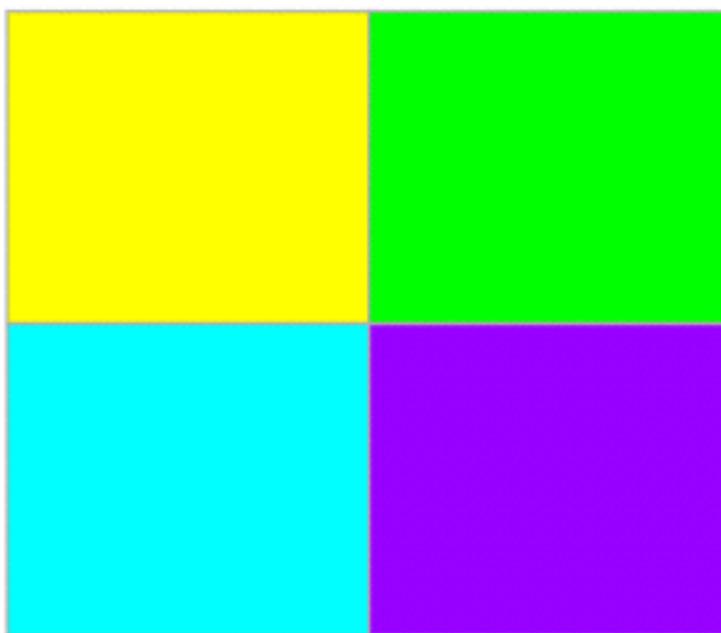
∂h_{11}	∂h_{12}
∂h_{21}	∂h_{22}



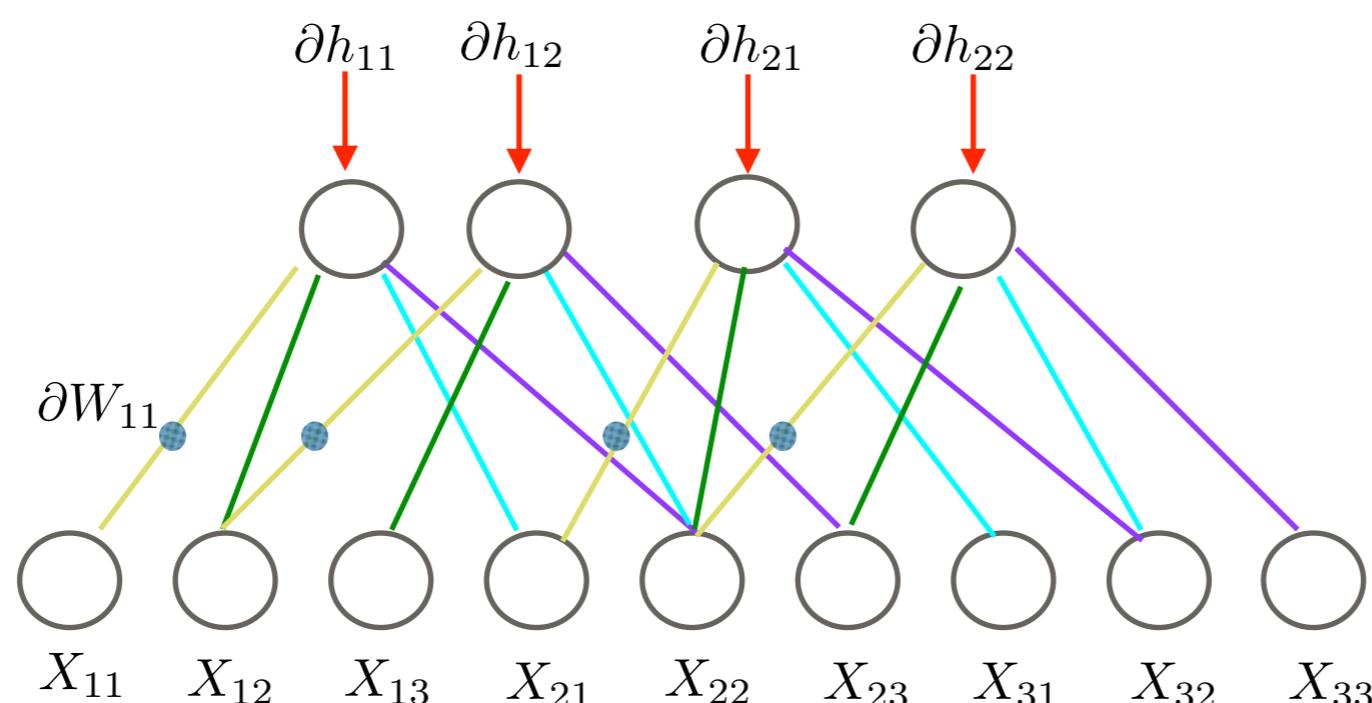
$$\partial h_{ij} = \frac{\partial L}{\partial h_{ij}}$$

Capa convolucional: Backpropagation

X_{11}	X_{12}	X_{13}
X_{21}	X_{22}	X_{23}
X_{31}	X_{32}	X_{33}



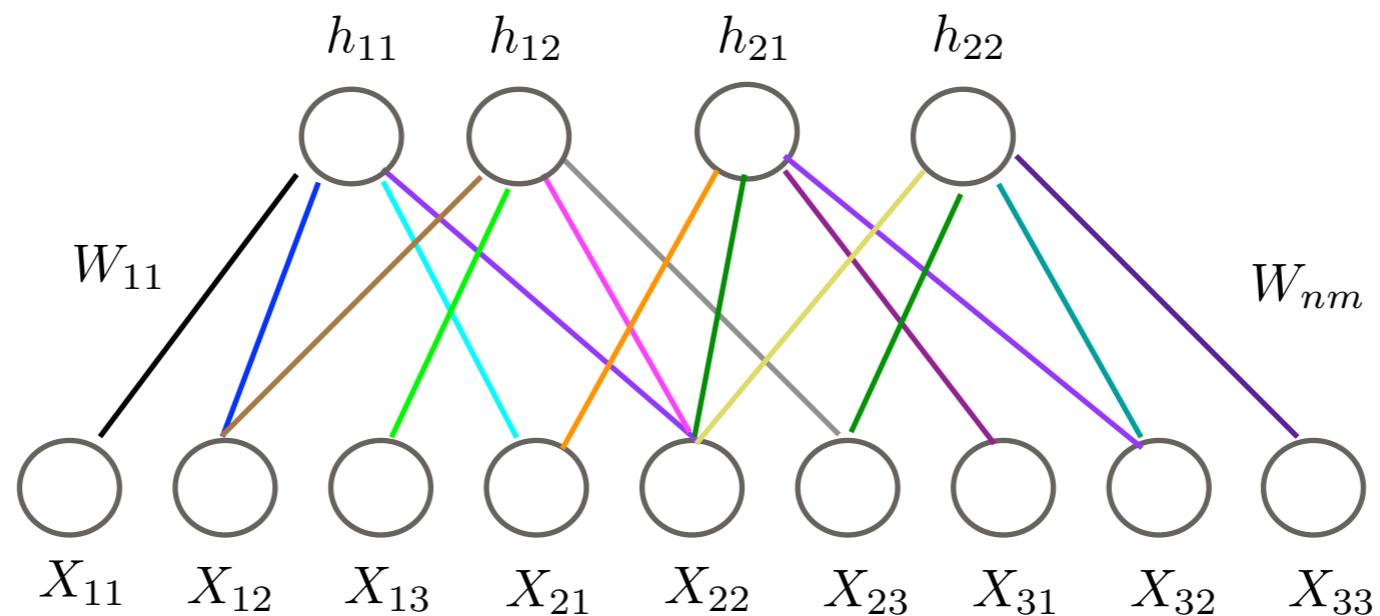
∂h_{11}	∂h_{12}
∂h_{21}	∂h_{22}



$$\partial h_{ij} = \frac{\partial L}{\partial h_{ij}}$$

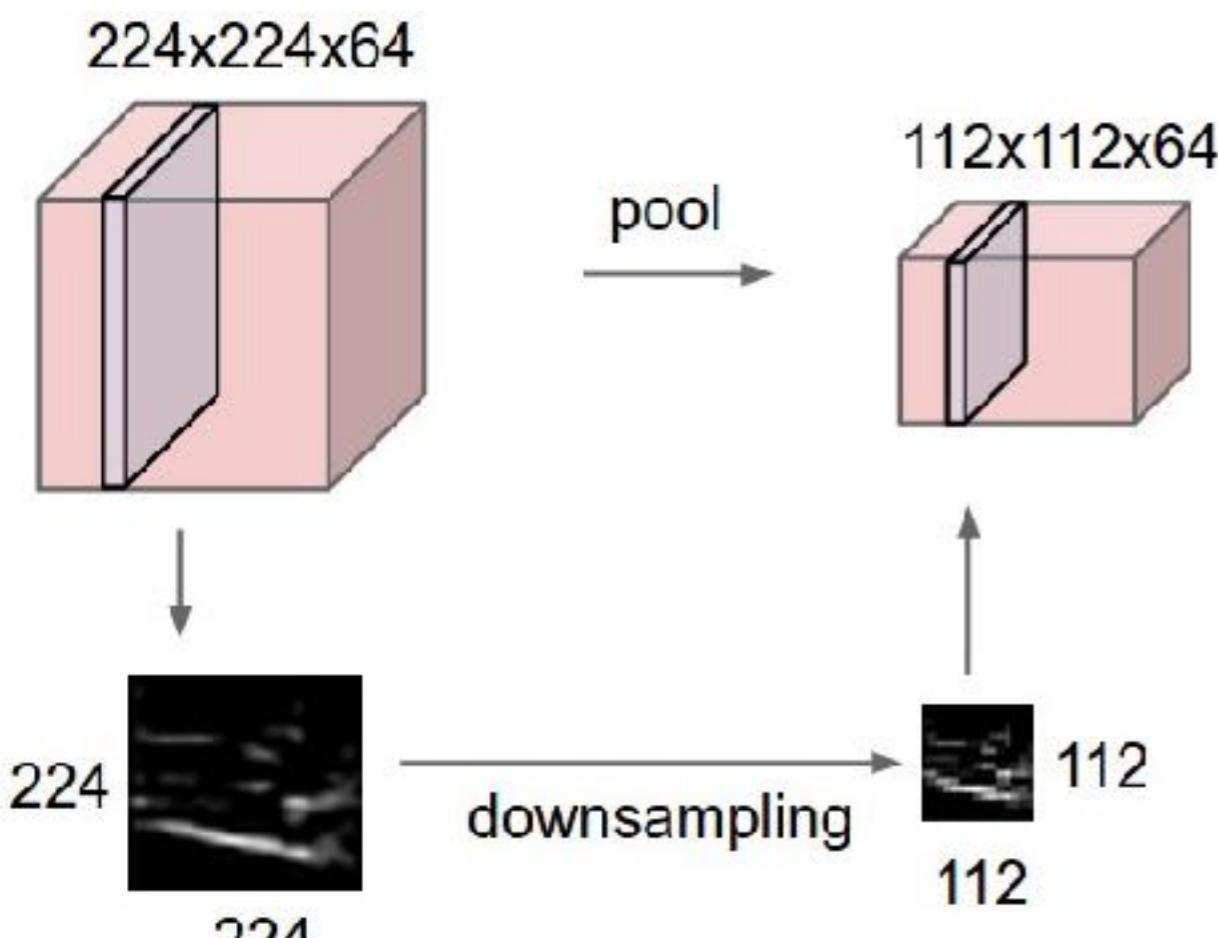
Capa Localmente conectada

No confundir esta capa con la capa convolucional. Las capas localmente conectadas **no comparten los pesos** como pasa en las convolucionales. Por lo tanto no se modifica el algoritmo de backpropagation

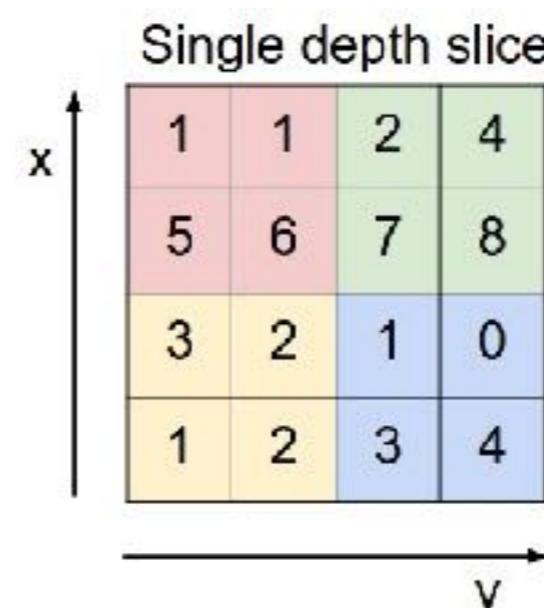


Submuestreo / Pooling

❖ Pool



❖ Max pooling



Hiperparámetros: Tamaño y Stride

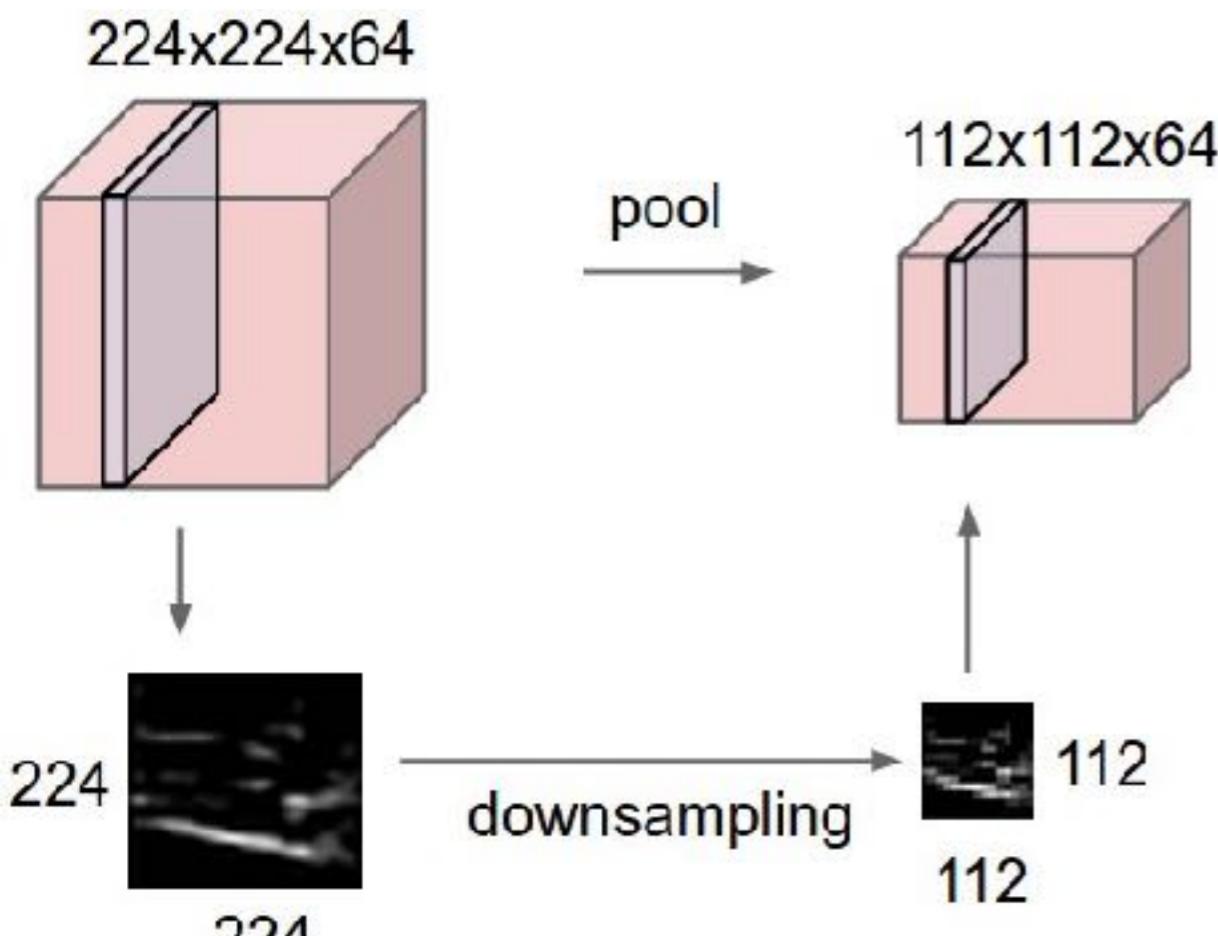
Tamaño de Salida:

$$\frac{N - K}{\text{stride}} + 1$$

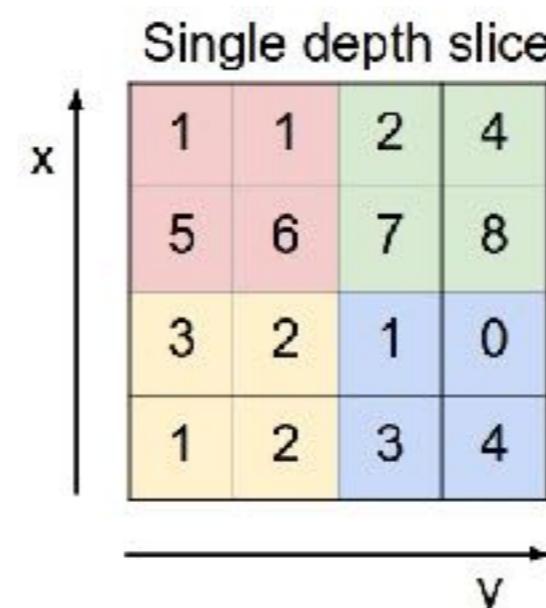
No es común usar Padding

Submuestreo / Pooling

❖ Pool



❖ Max pooling



Hiperparámetros: Tamaño y Stride

Tamaño de Salida:

$$\frac{N - K}{\text{stride}} + 1$$

No es común usar Padding

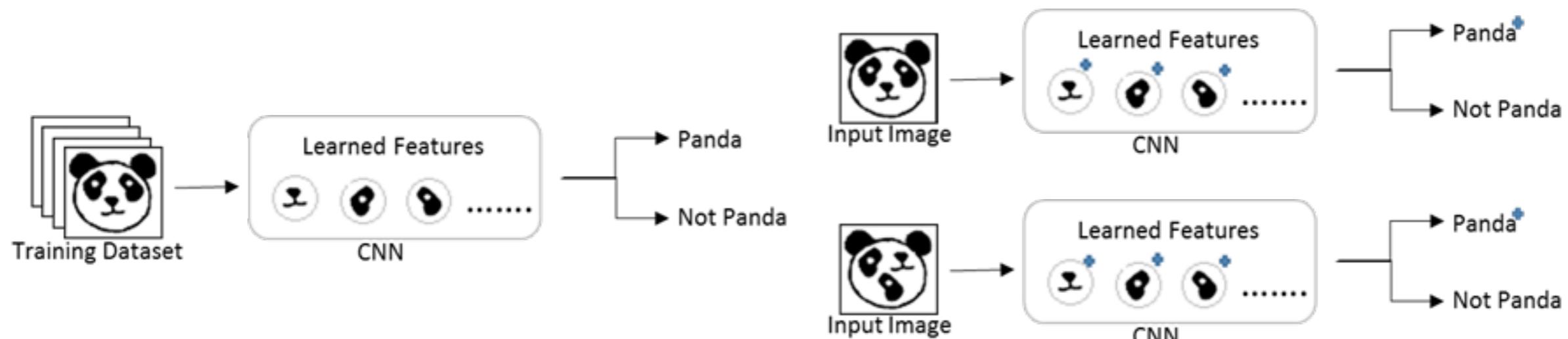
¿Cantidad de parámetros a entrenar?

Submuestreo / Pooling

- ❖ MaxPooling
- ❖ AveragePooling: ojo puede perder la activación al promediar
- ❖ GlobalMaxPooling: en secuencias temporales
- ❖ GlobalAveragePooling: en sec. temporales

Submuestreo / Pooling

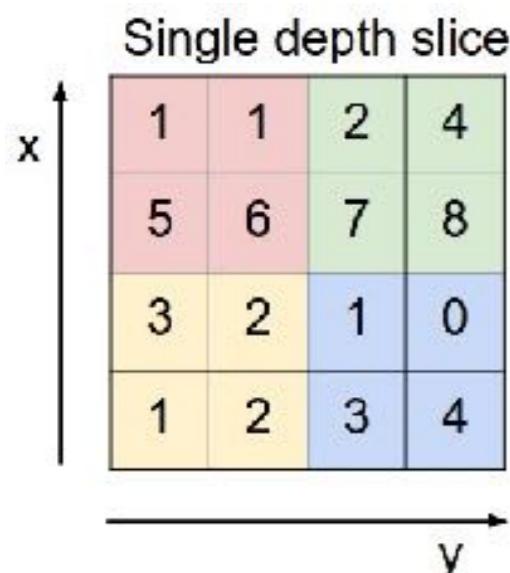
- ❖ Pros
 - ❖ Reduce la representación de la entrada o imagen
 - ❖ Da invariancia espacial a las características
- ❖ Contras
 - ❖ Se pierde la noción espacial de las características y su relación entre ellas



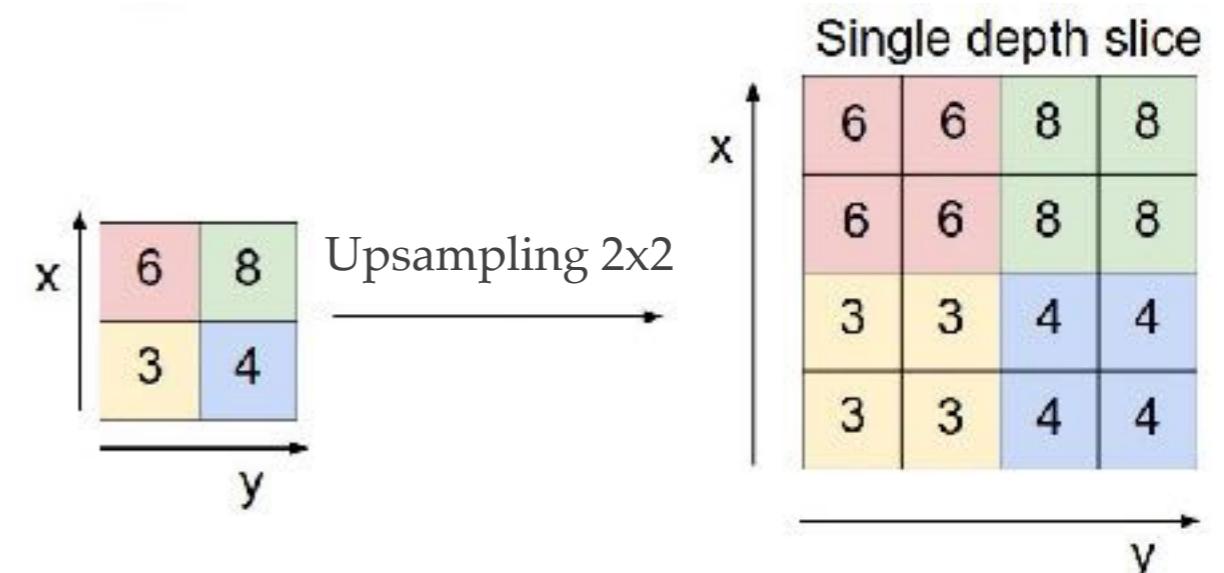
Submuestreo / Pooling

- ❖ UpSampling

- ❖ Max pool



- ❖ Up Sampling



Hiperparámetros: Tamaño

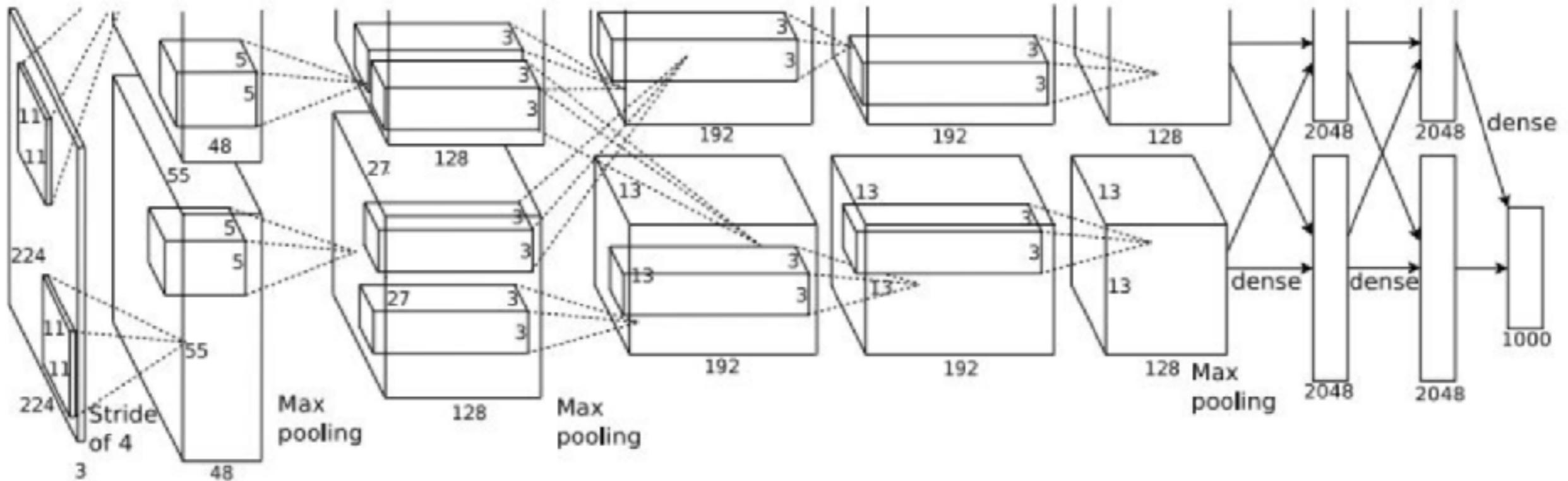
Tamaño de Salida: Tamaño x Input

Img. source: Stanford CS231n 2017

Casos de estudio

AlexNet

[Krizhevsky et al. 2012]

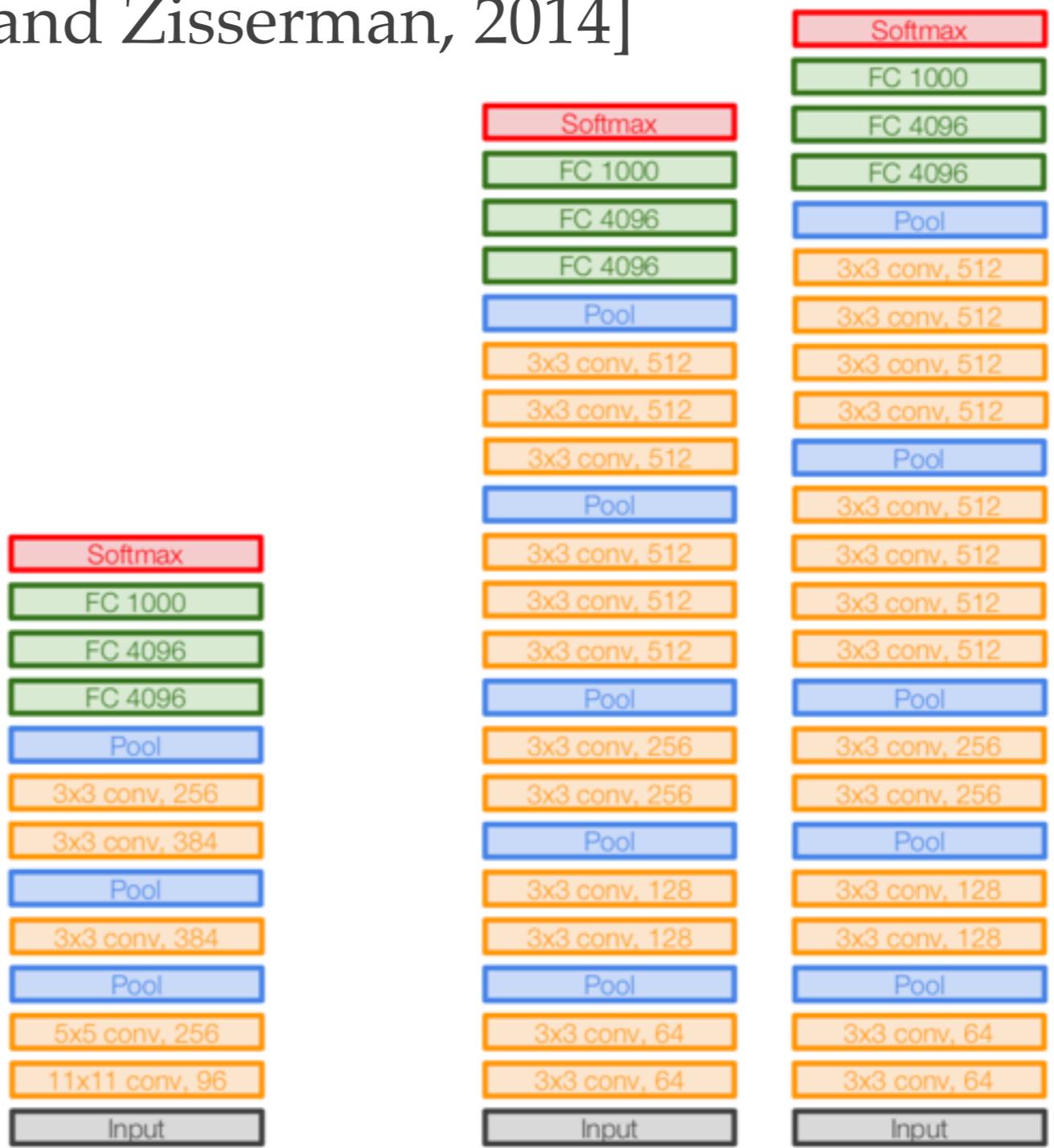


- ❖ Convolución con filtros de 11x11, 5x5 y 3x3
- ❖ Primer uso de ReLU
- ❖ Fuerte utilización de **aumentación de datos**
- ❖ Uso de L2 como regularización

VGGNet

[Simonyan and Zisserman, 2014]

- ❖ Filtros más pequeños
- ❖ Redes más profundas



AlexNet

VGG16

VGG19

VGGNet

[Simonyan and Zisserman, 2014]

- ❖ Filtros más pequeños
- ❖ Redes más profundas
- ❖ Juntar 3 filtros de 3x3 con un stride de 1 dan un campo receptivo efectivo similar al de una conv. de 7x7

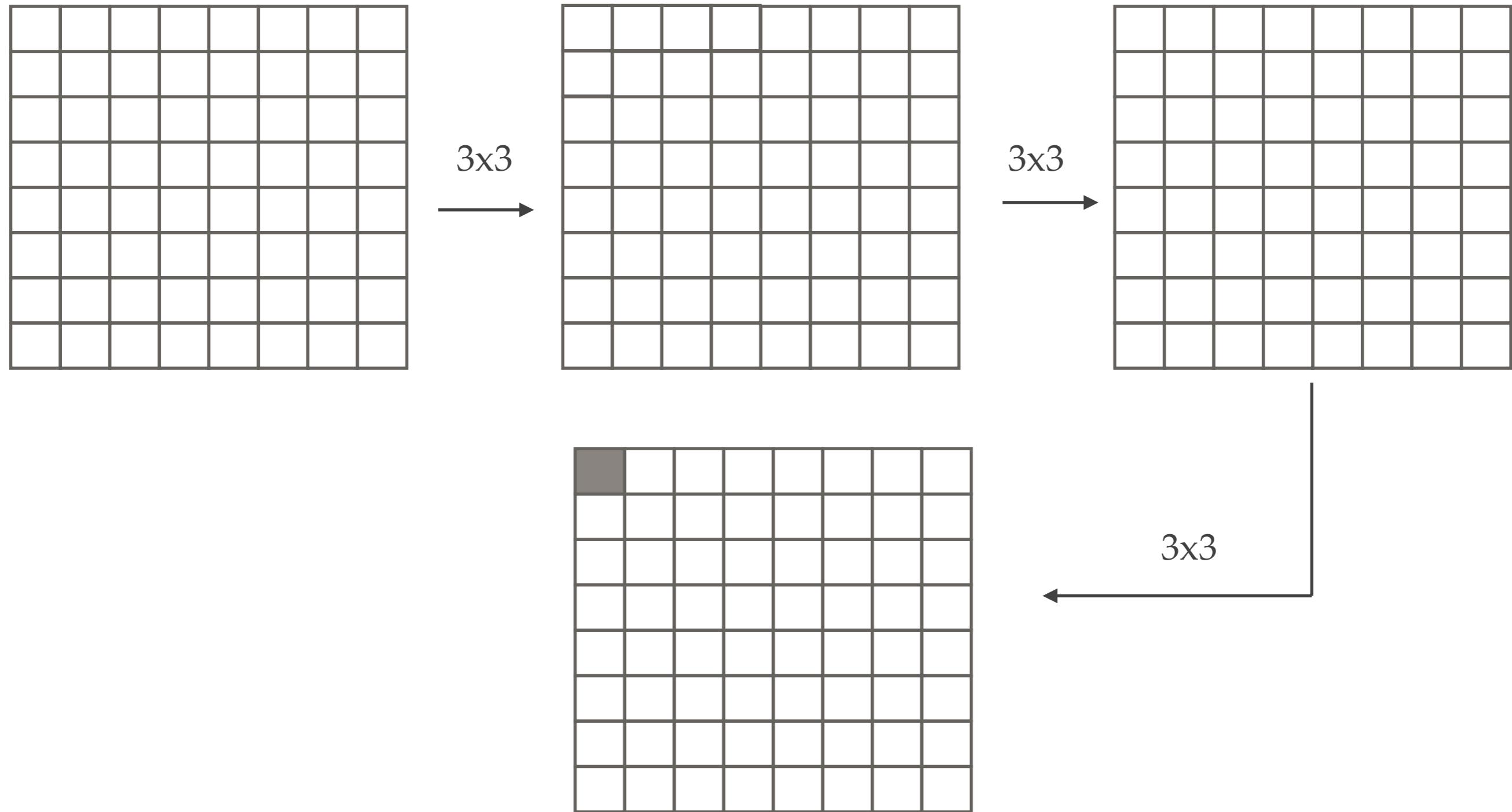


AlexNet

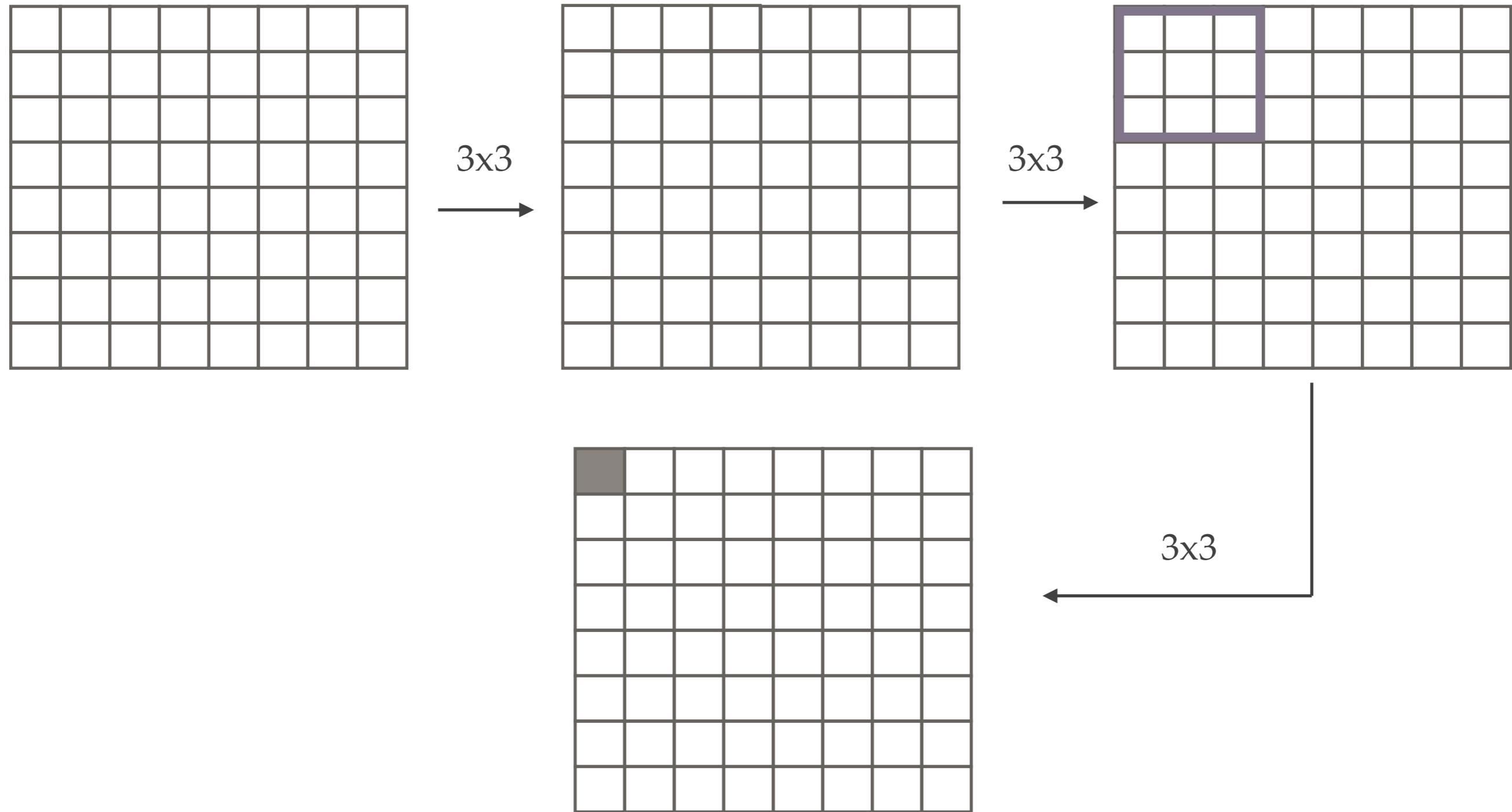
VGG16

VGG19

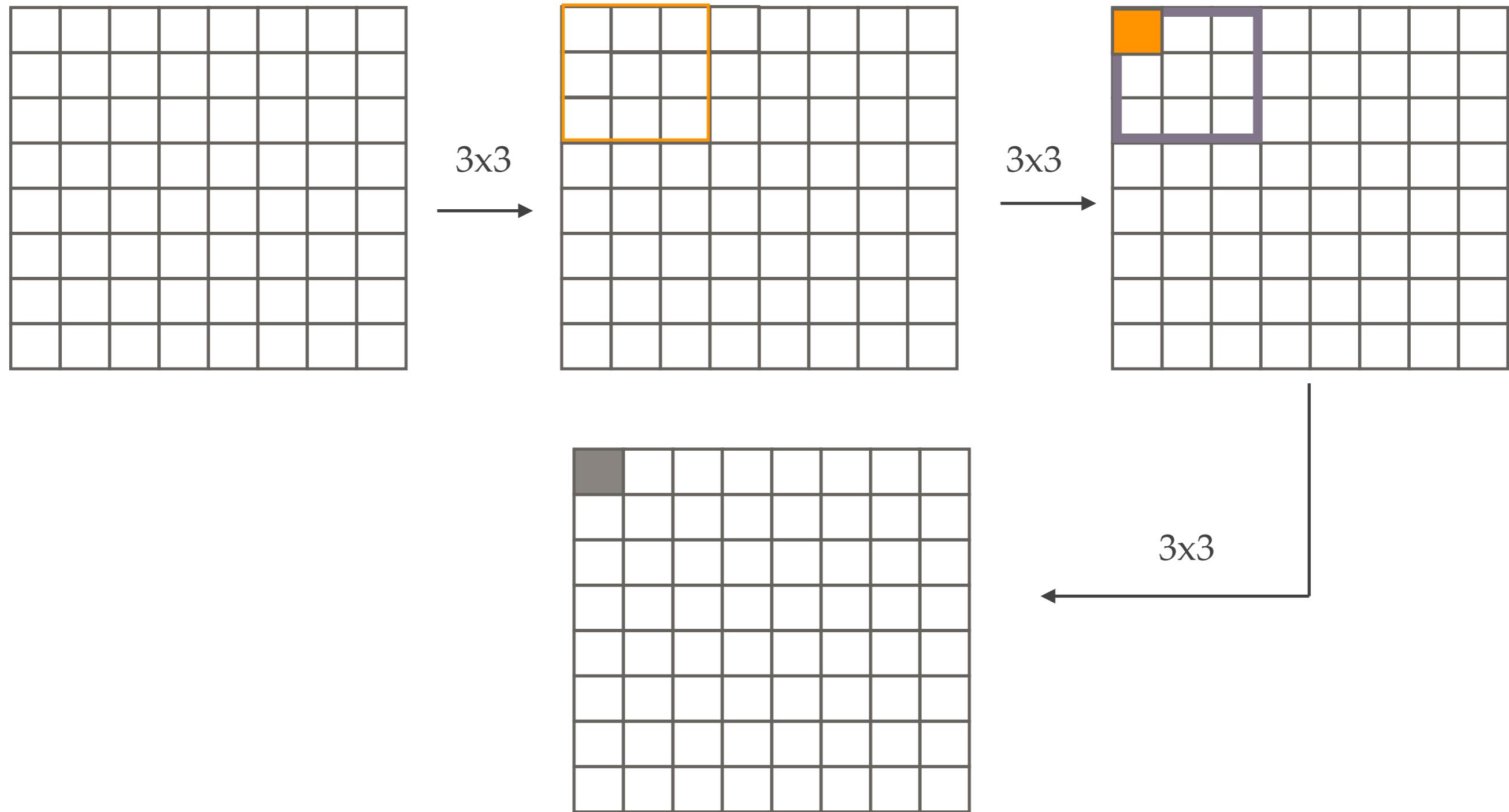
VGGNet



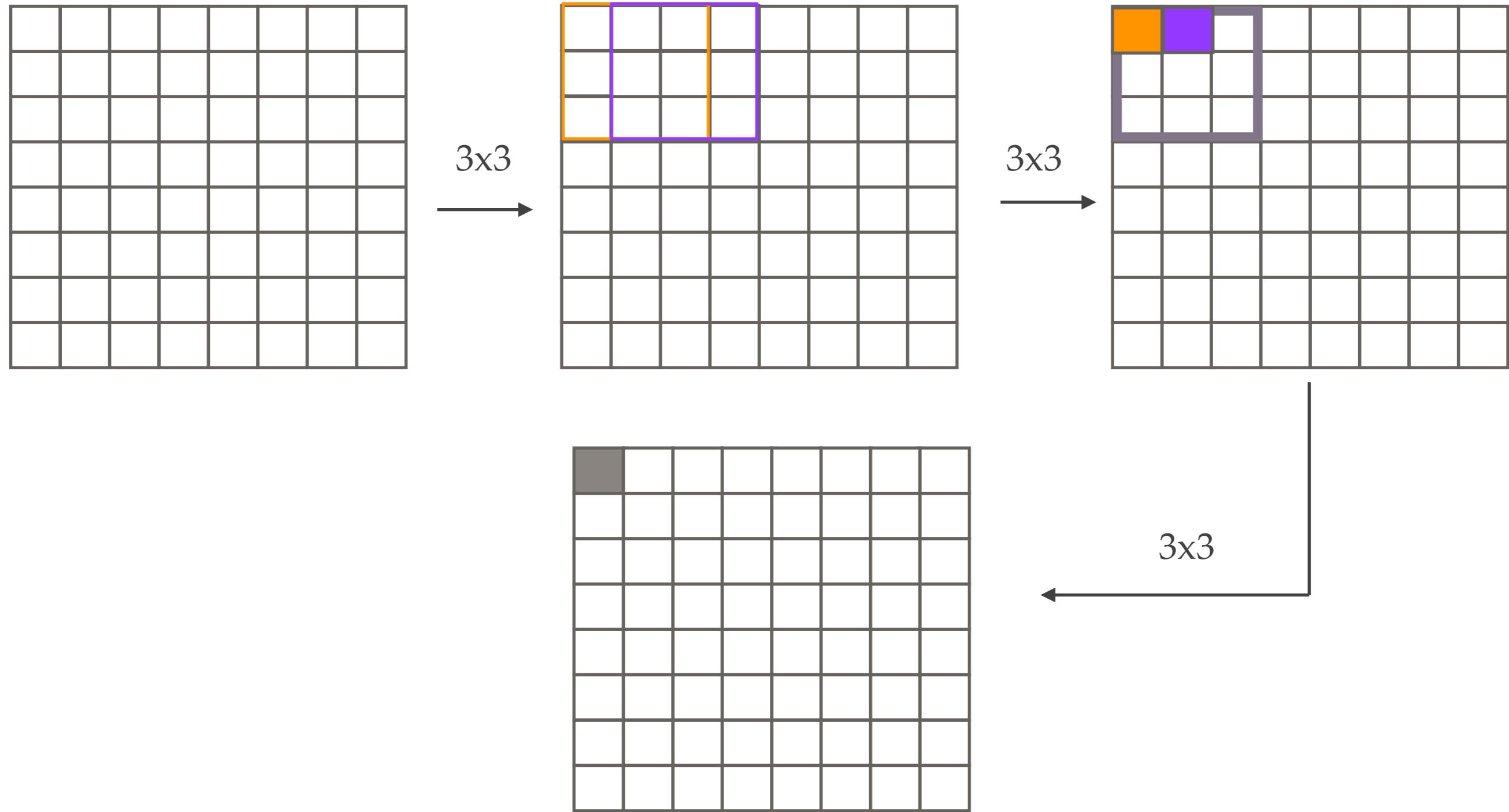
VGGNet



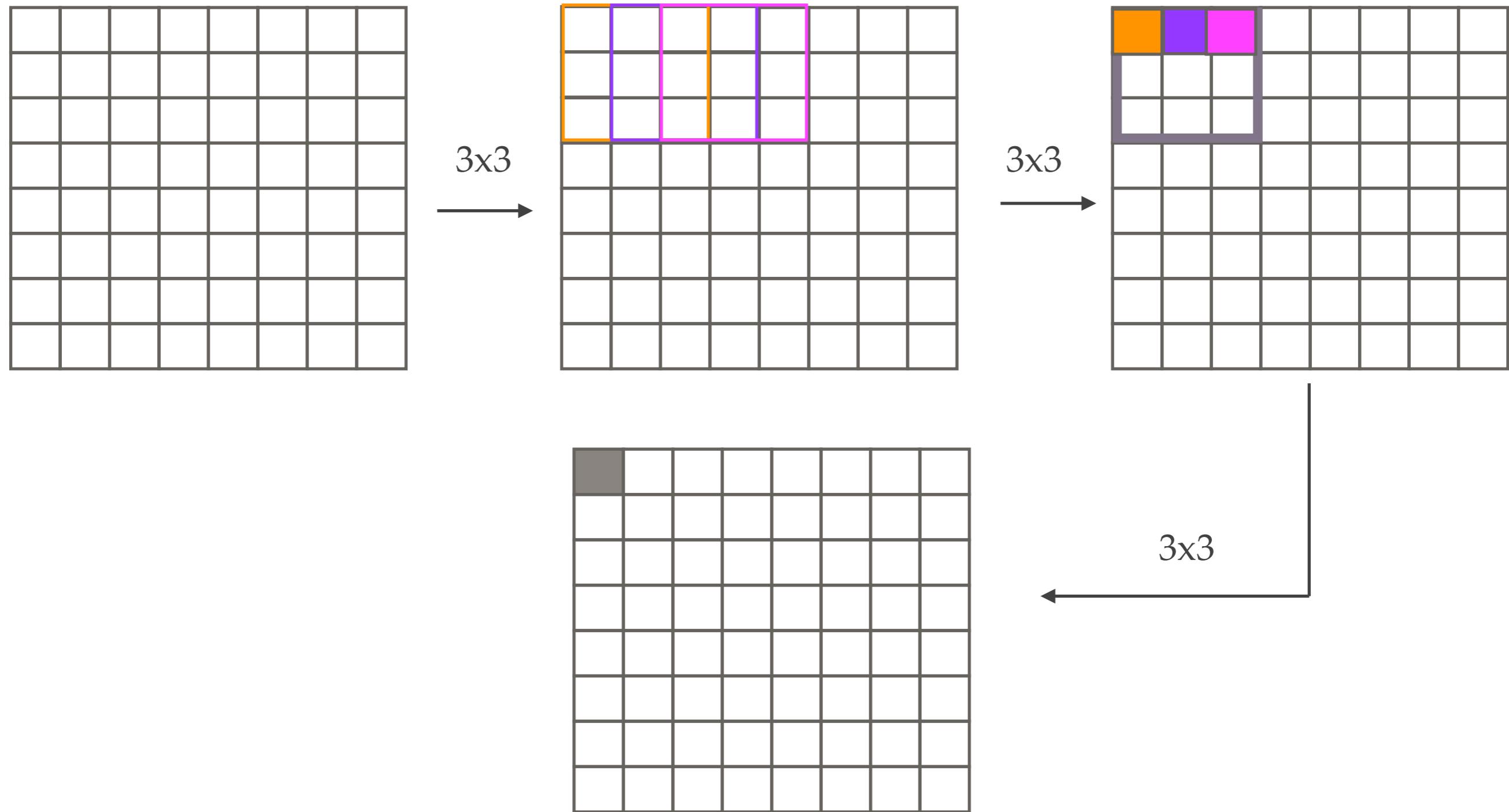
VGGNet



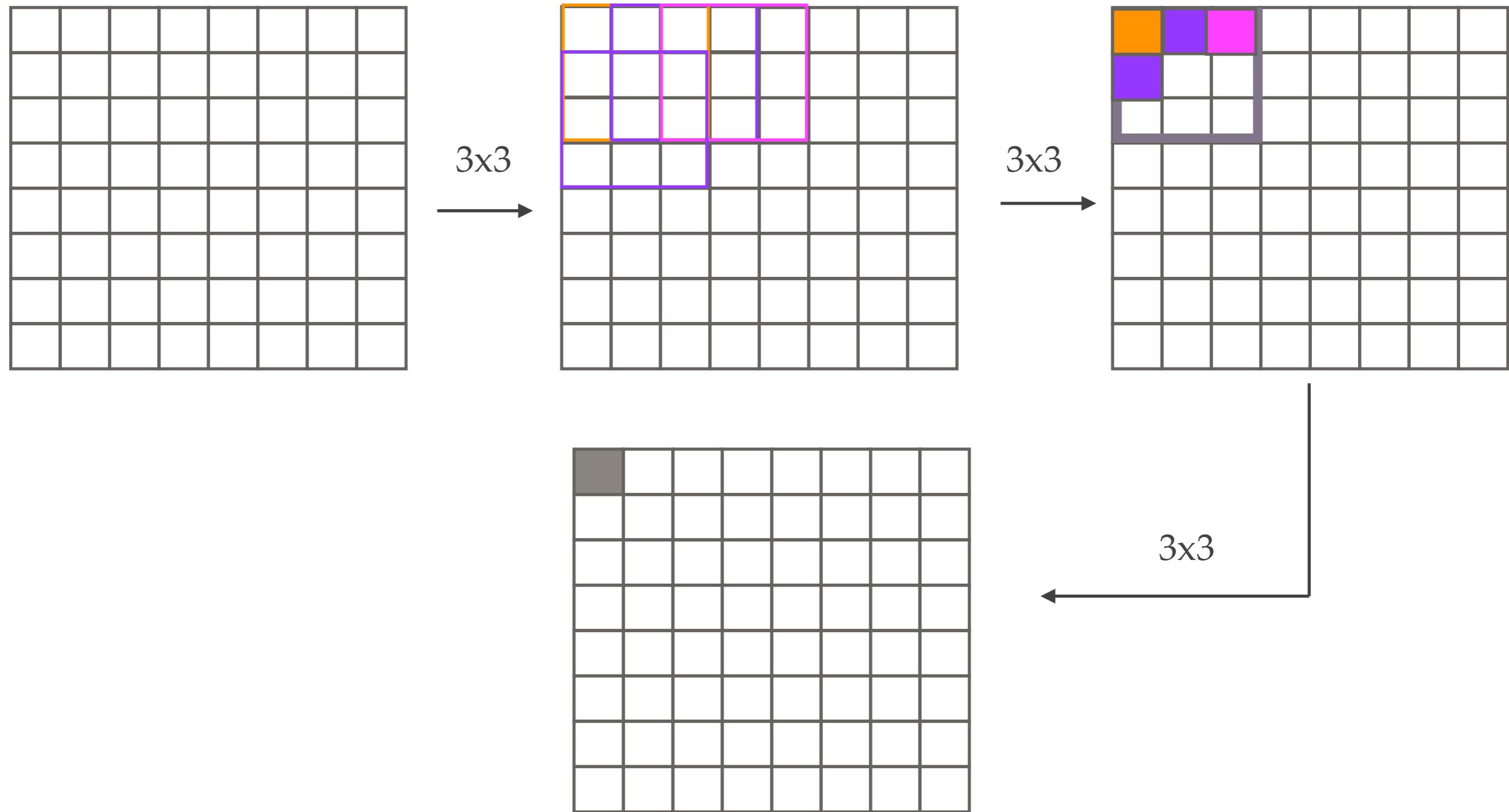
VGGNet



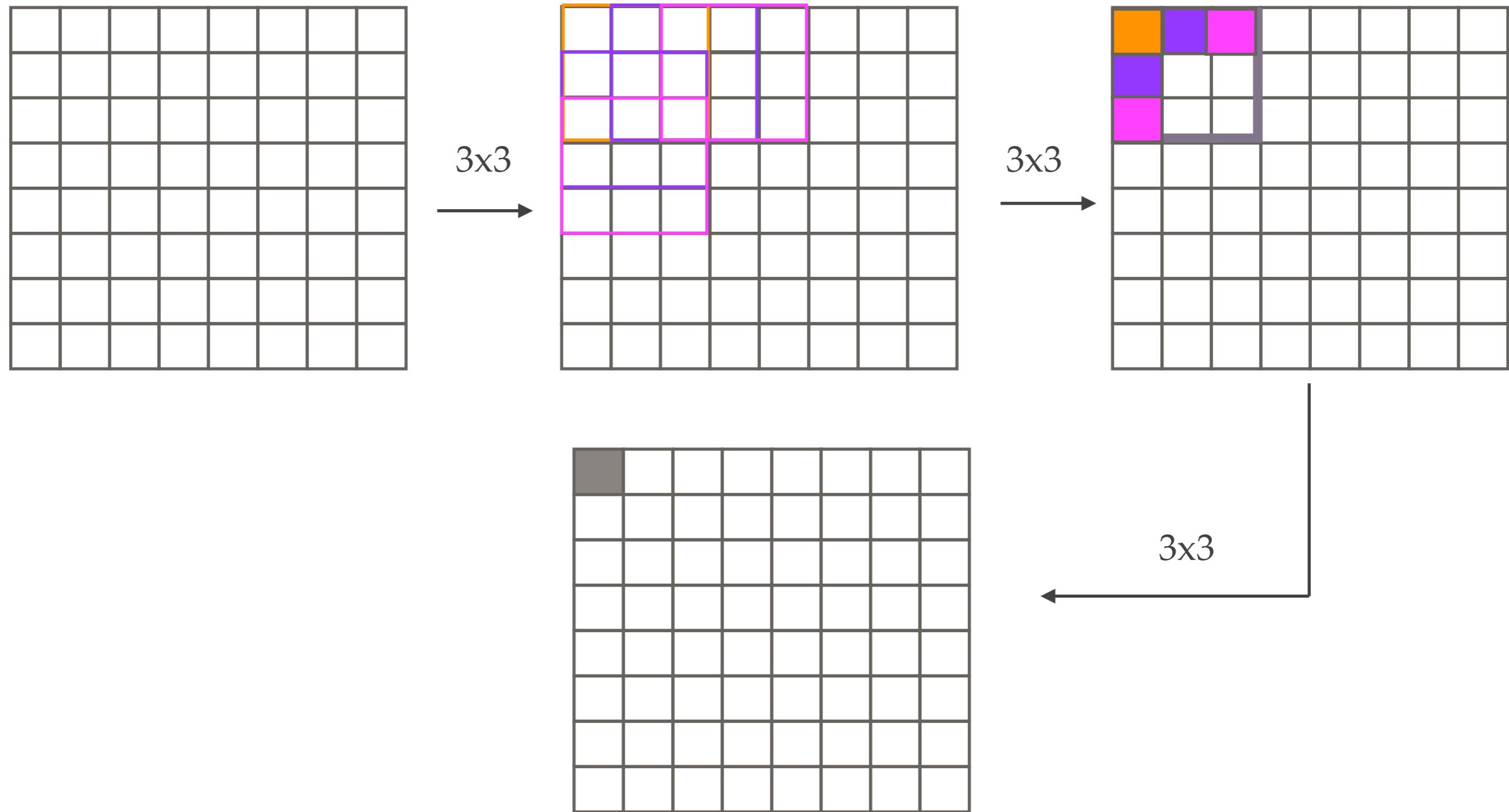
VGGNet



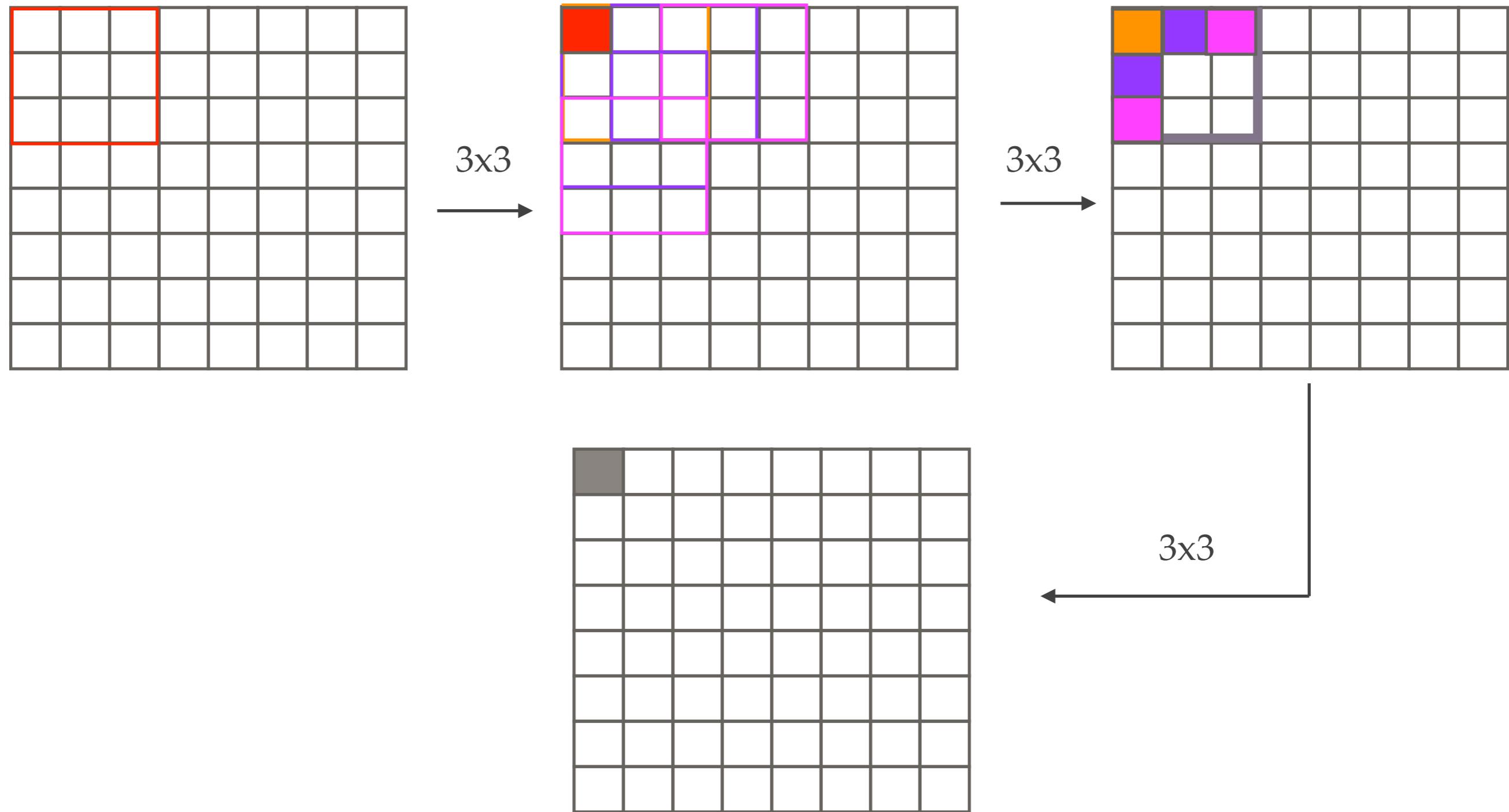
VGGNet



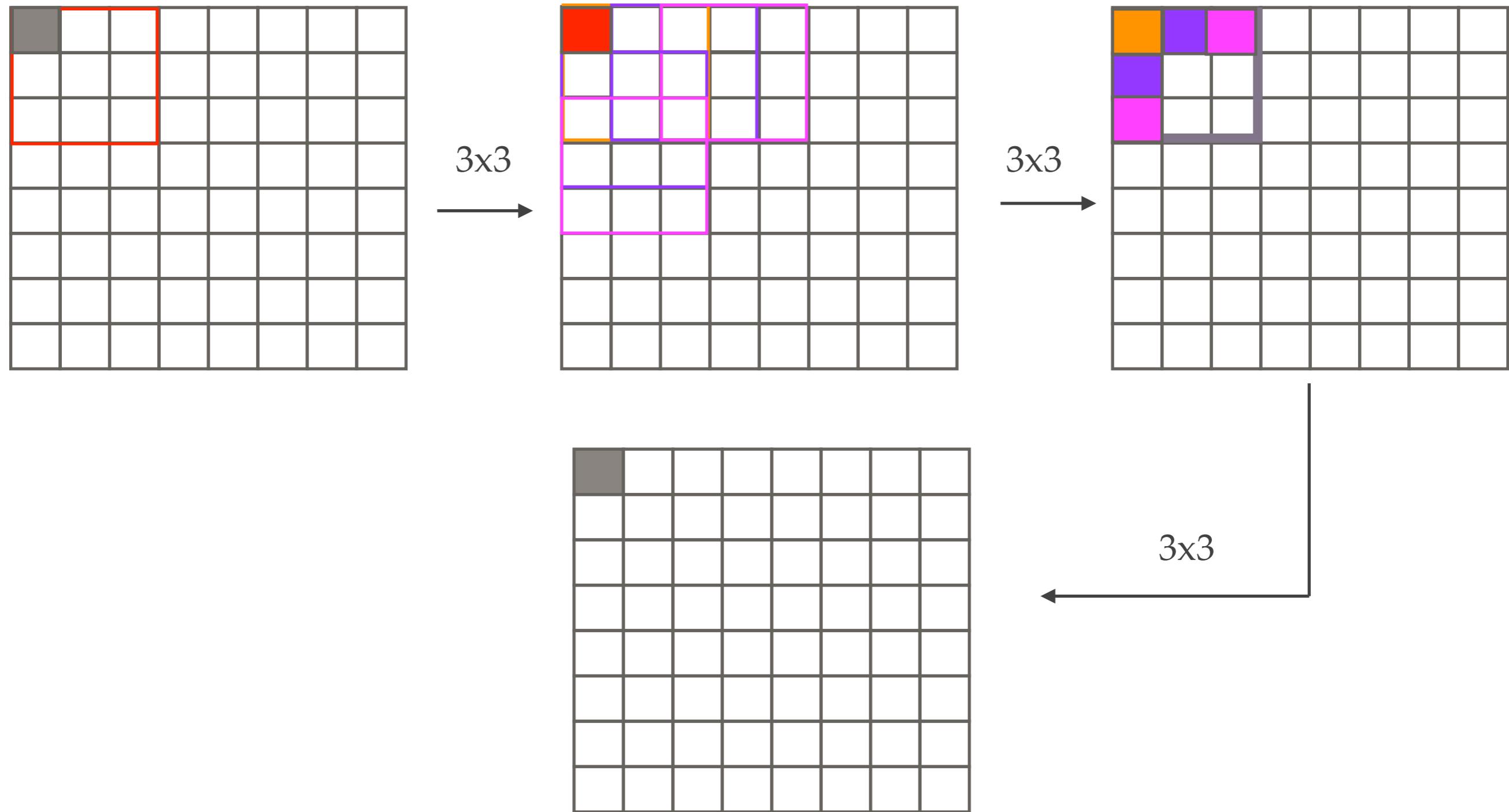
VGGNet



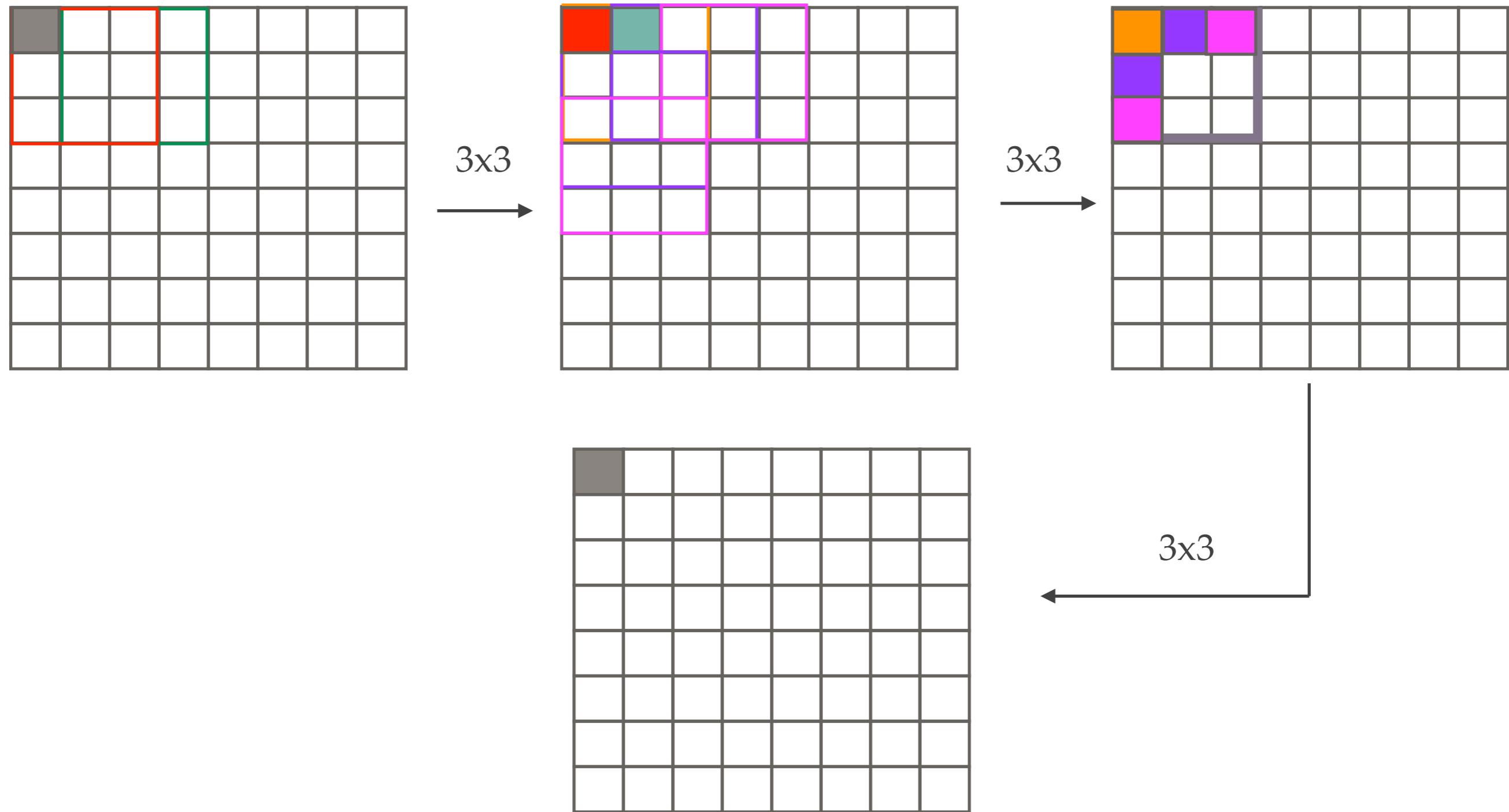
VGGNet



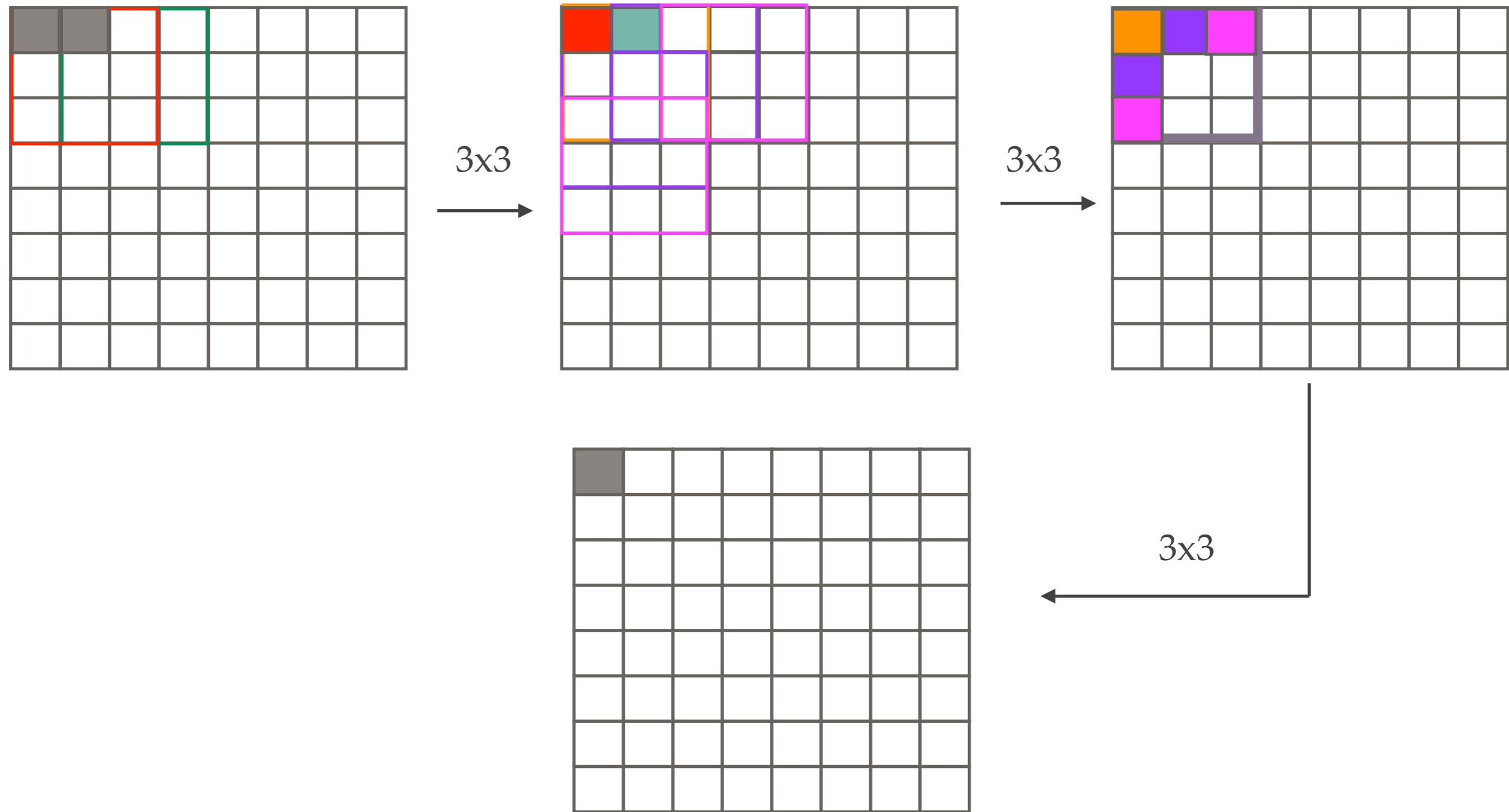
VGGNet



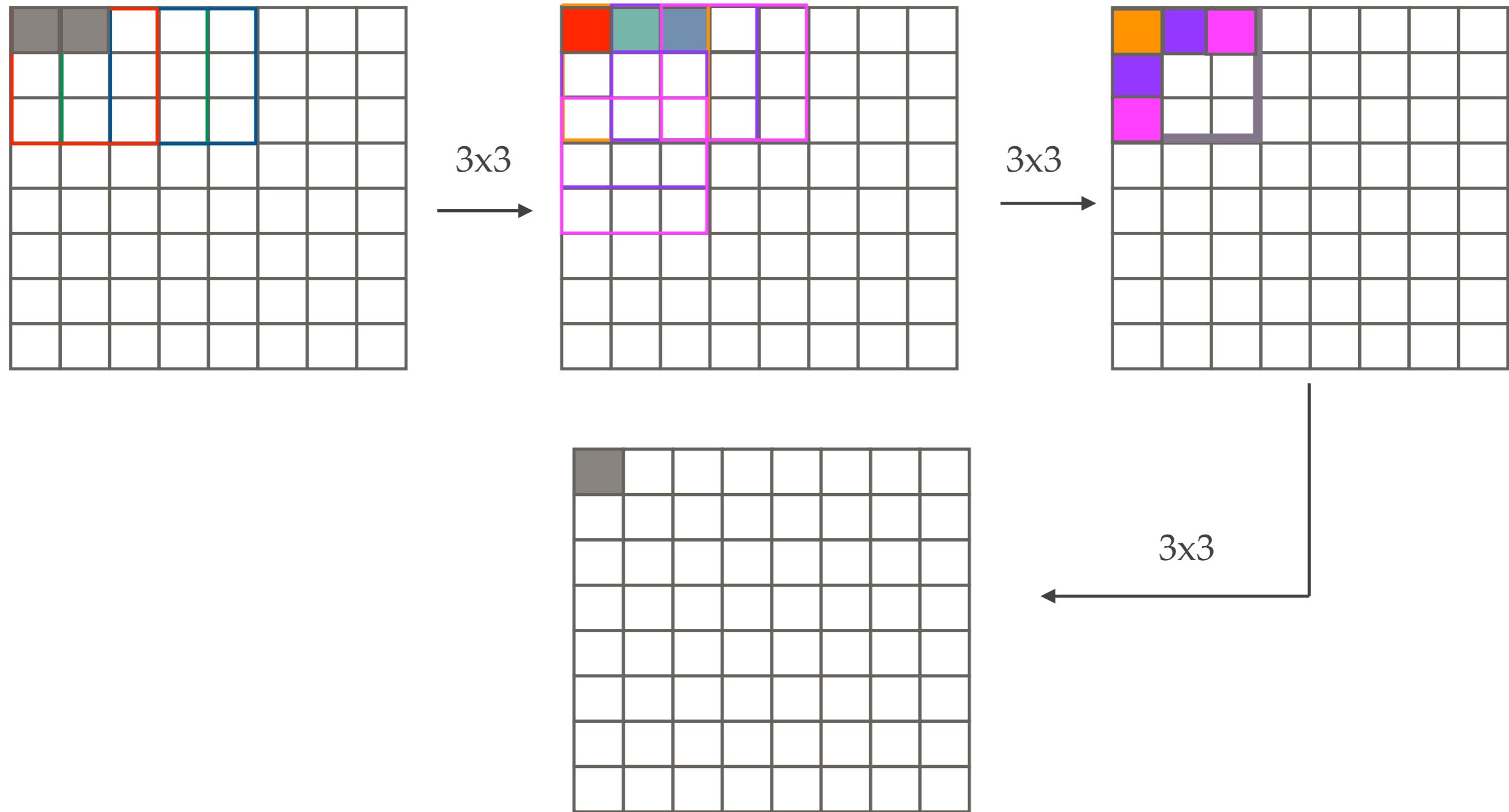
VGGNet



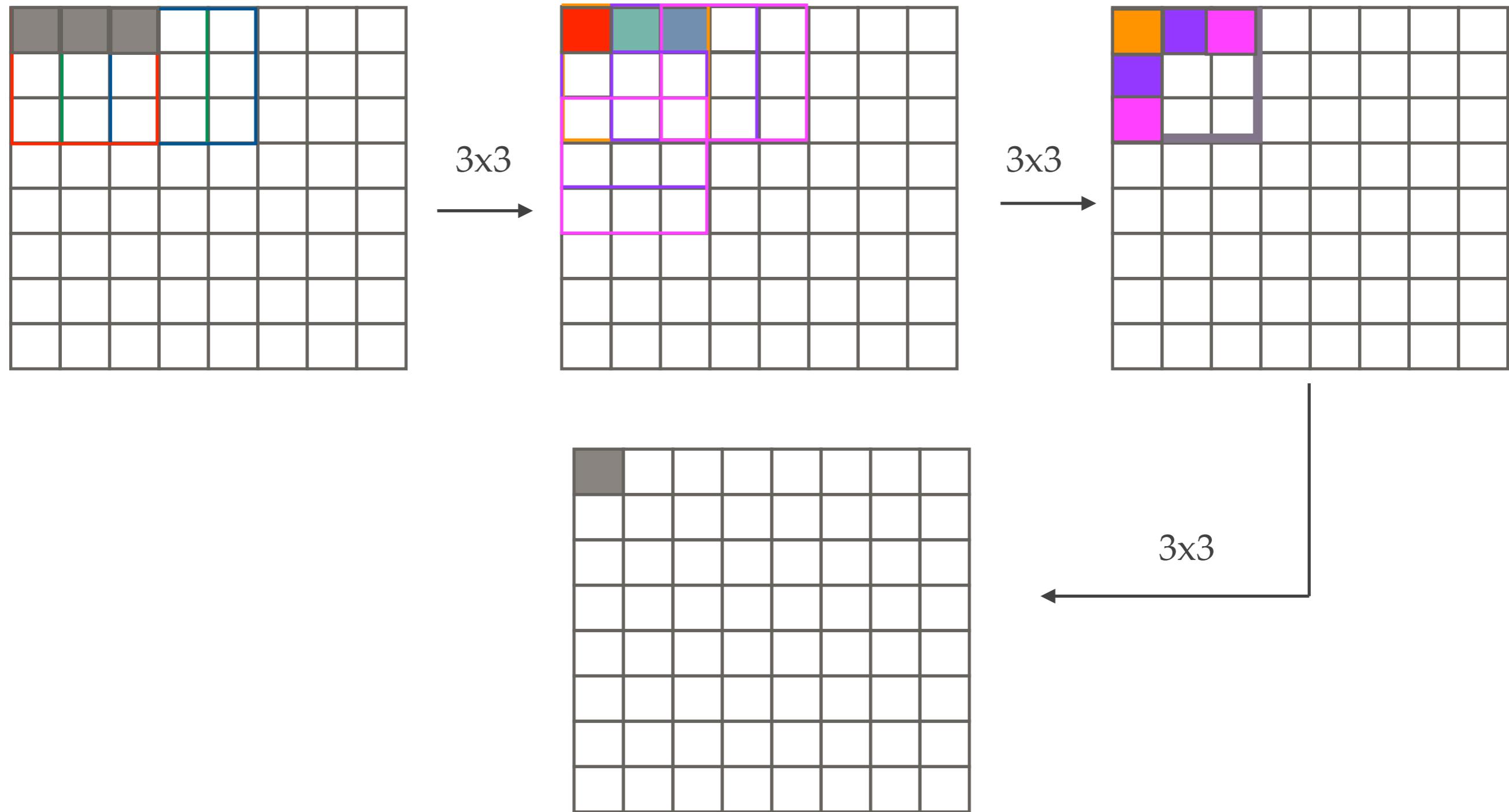
VGGNet



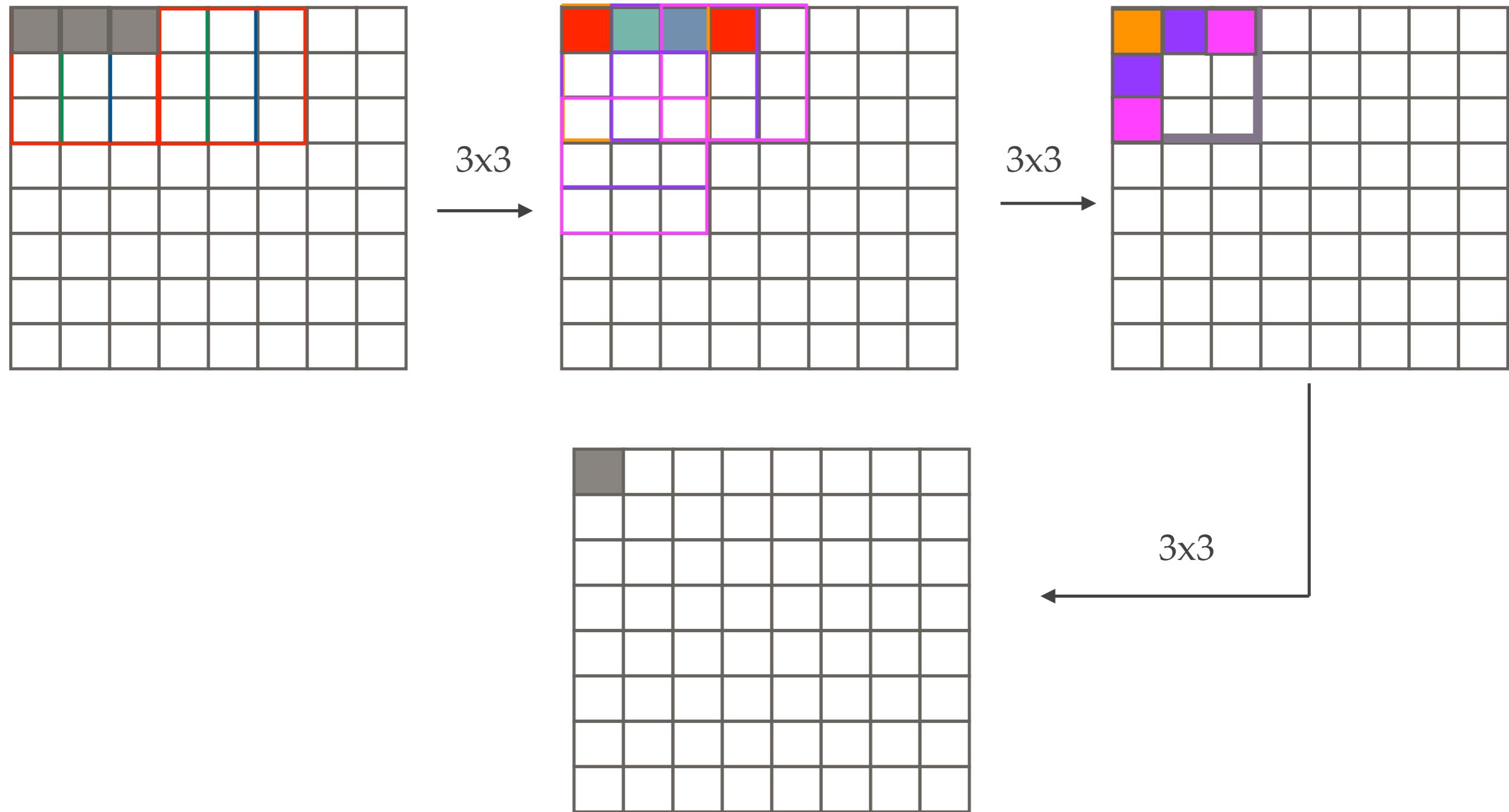
VGGNet



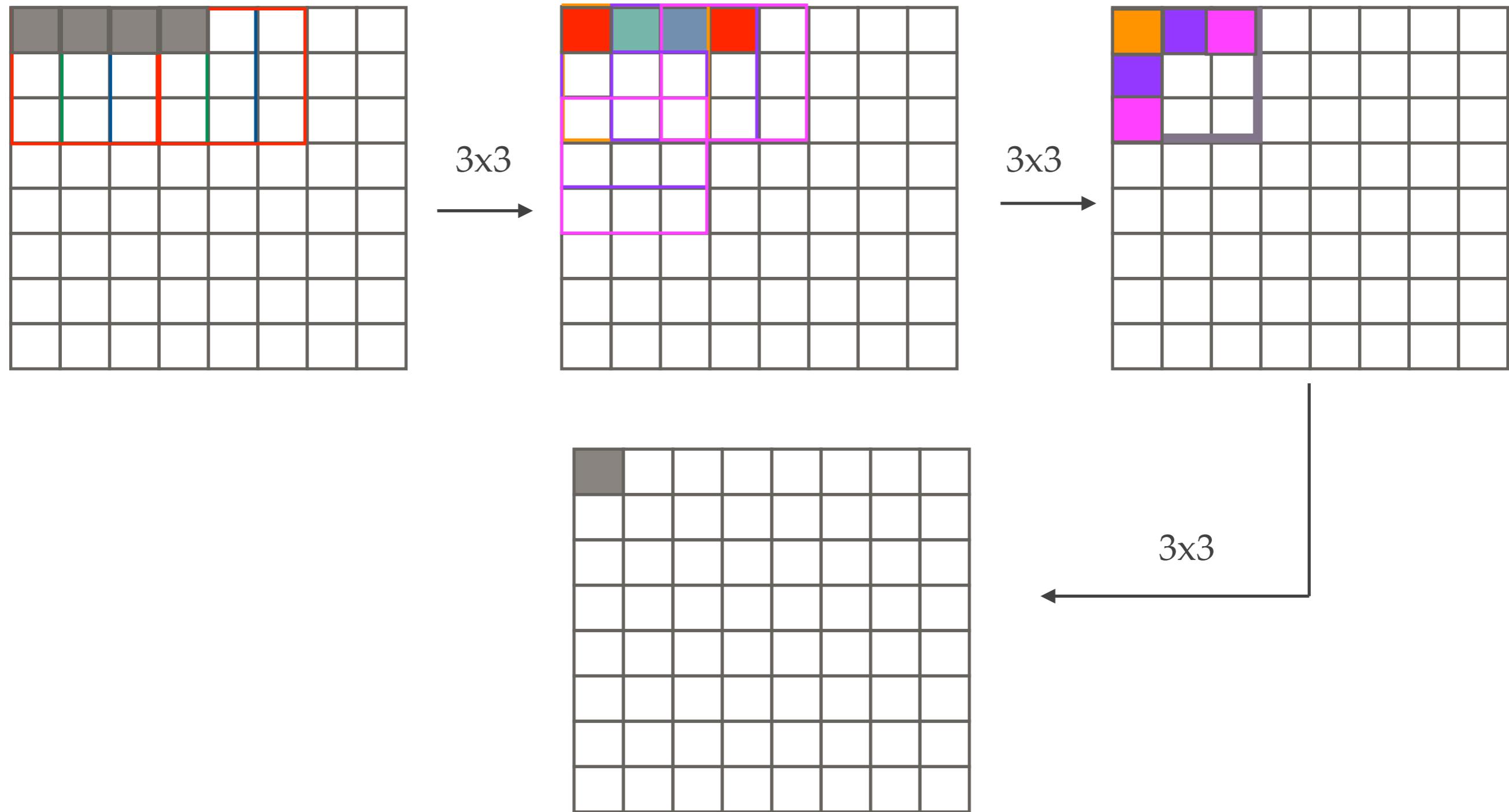
VGGNet



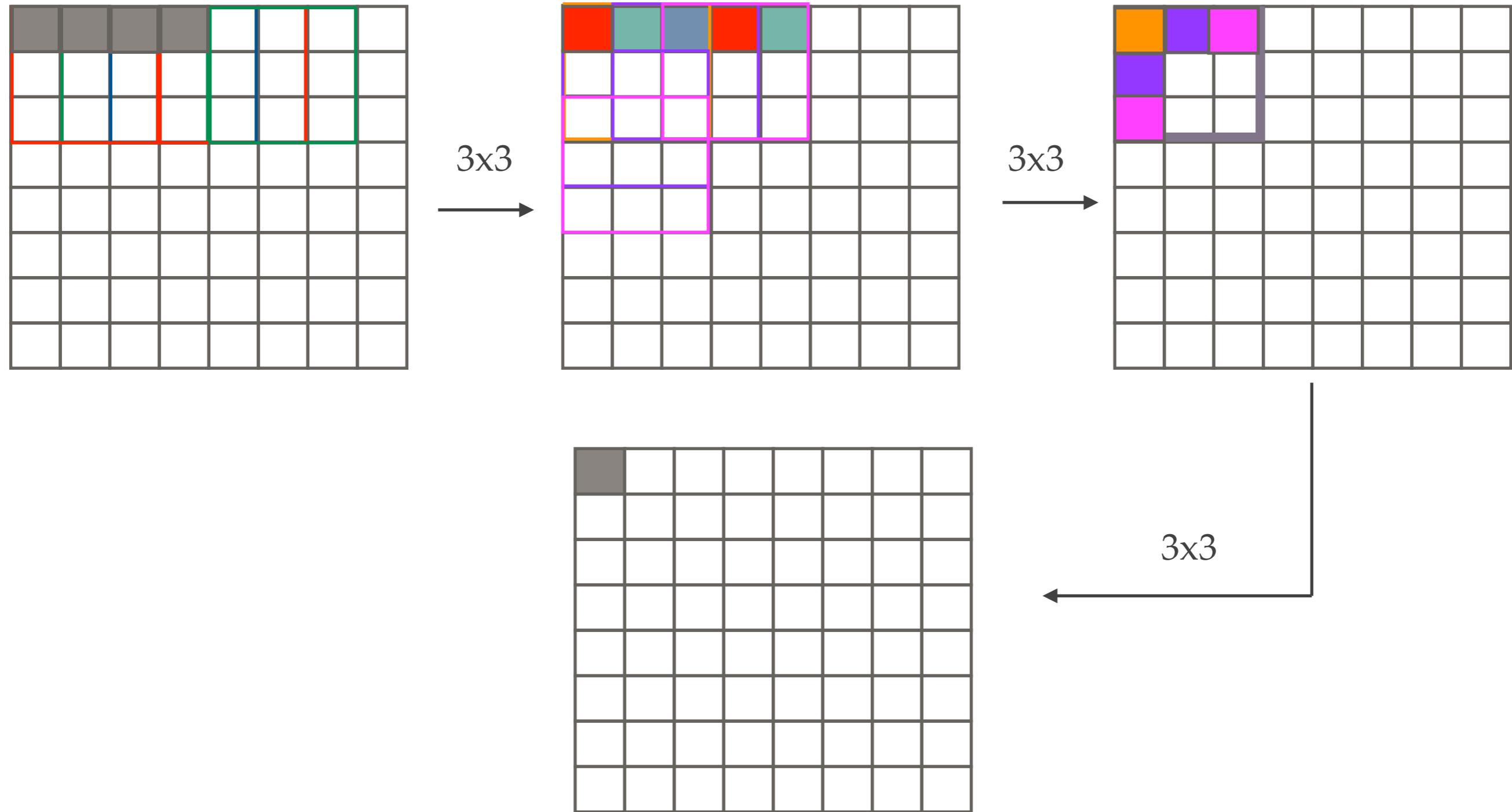
VGGNet



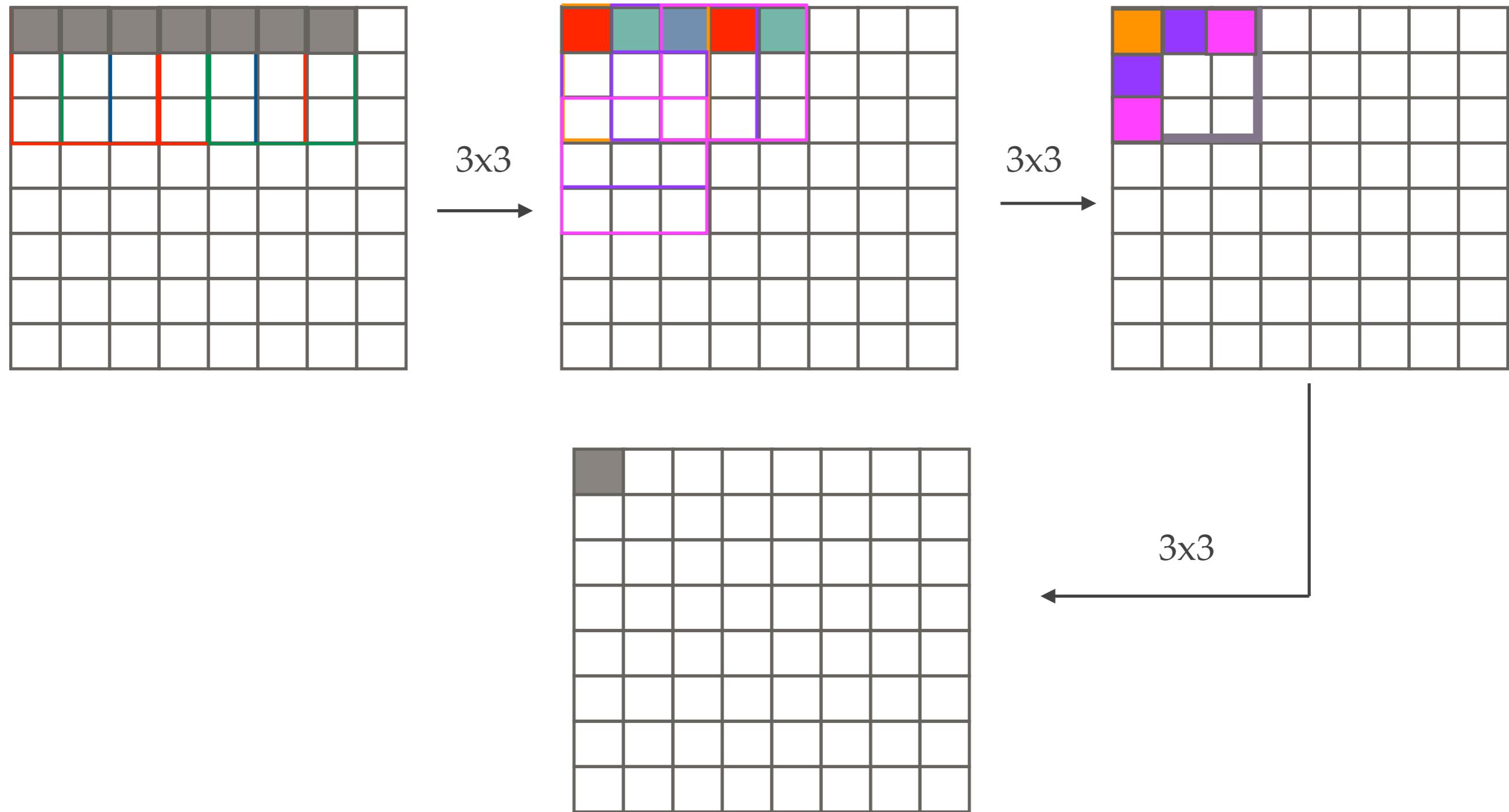
VGGNet



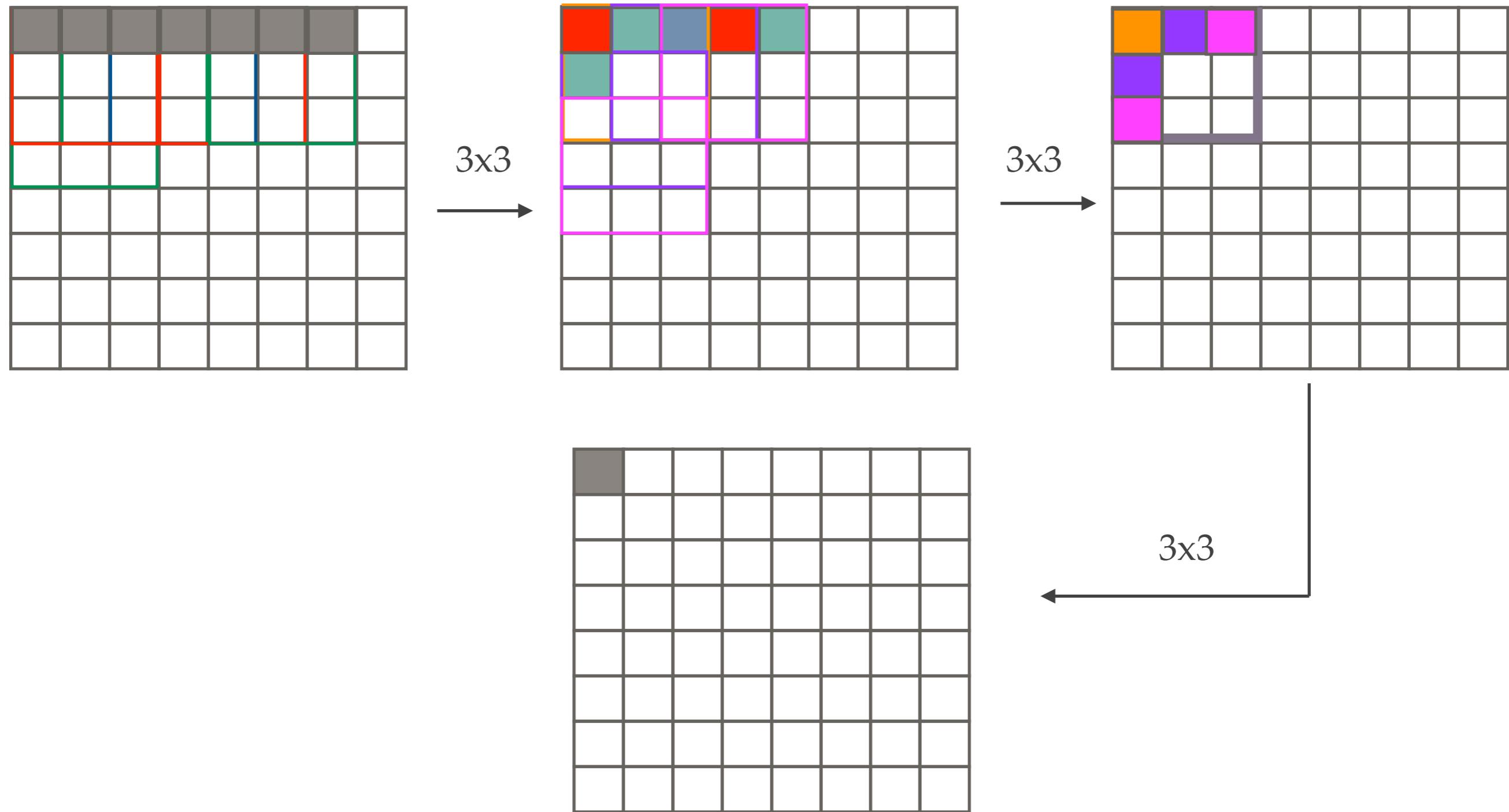
VGGNet



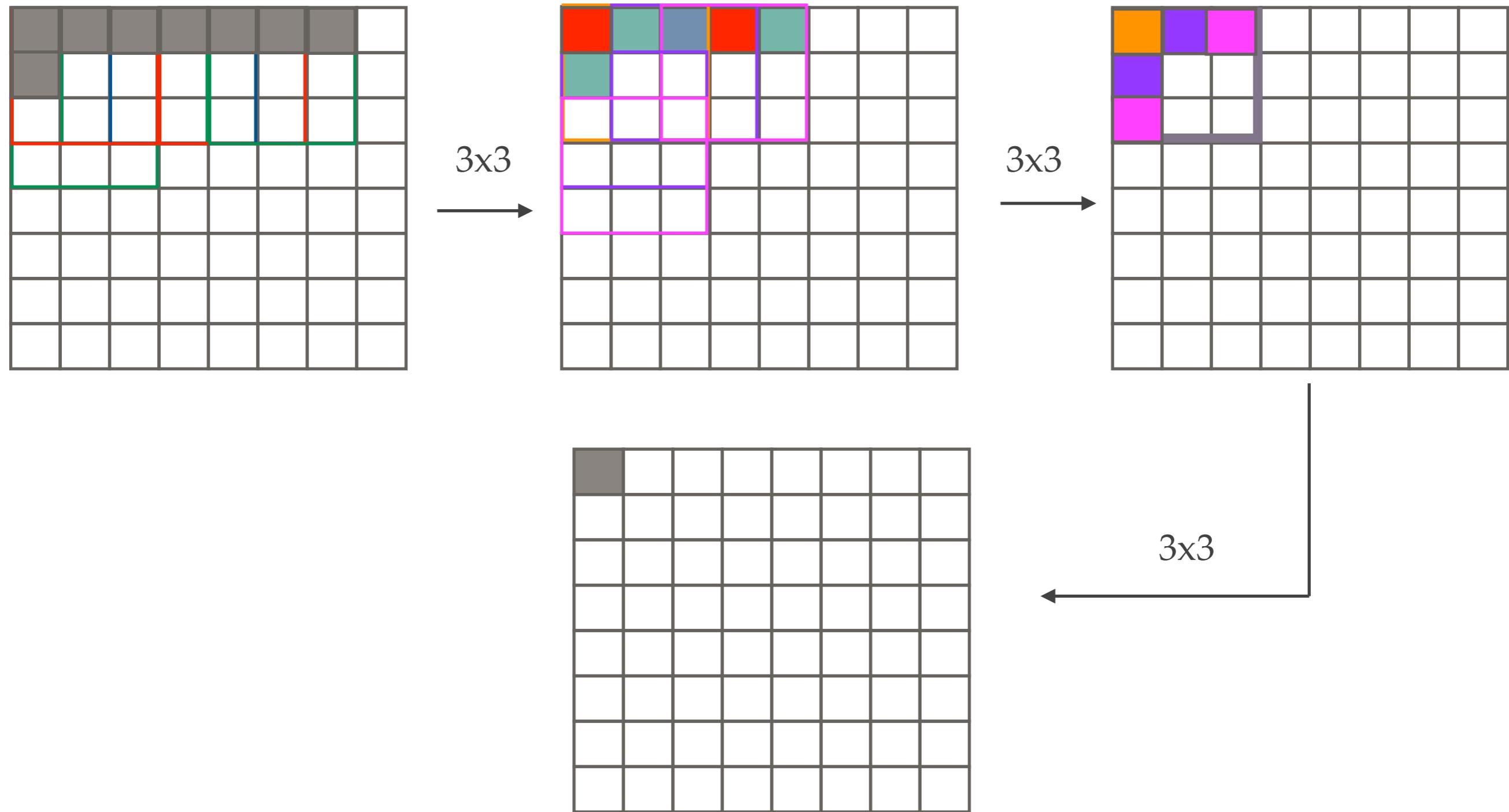
VGGNet



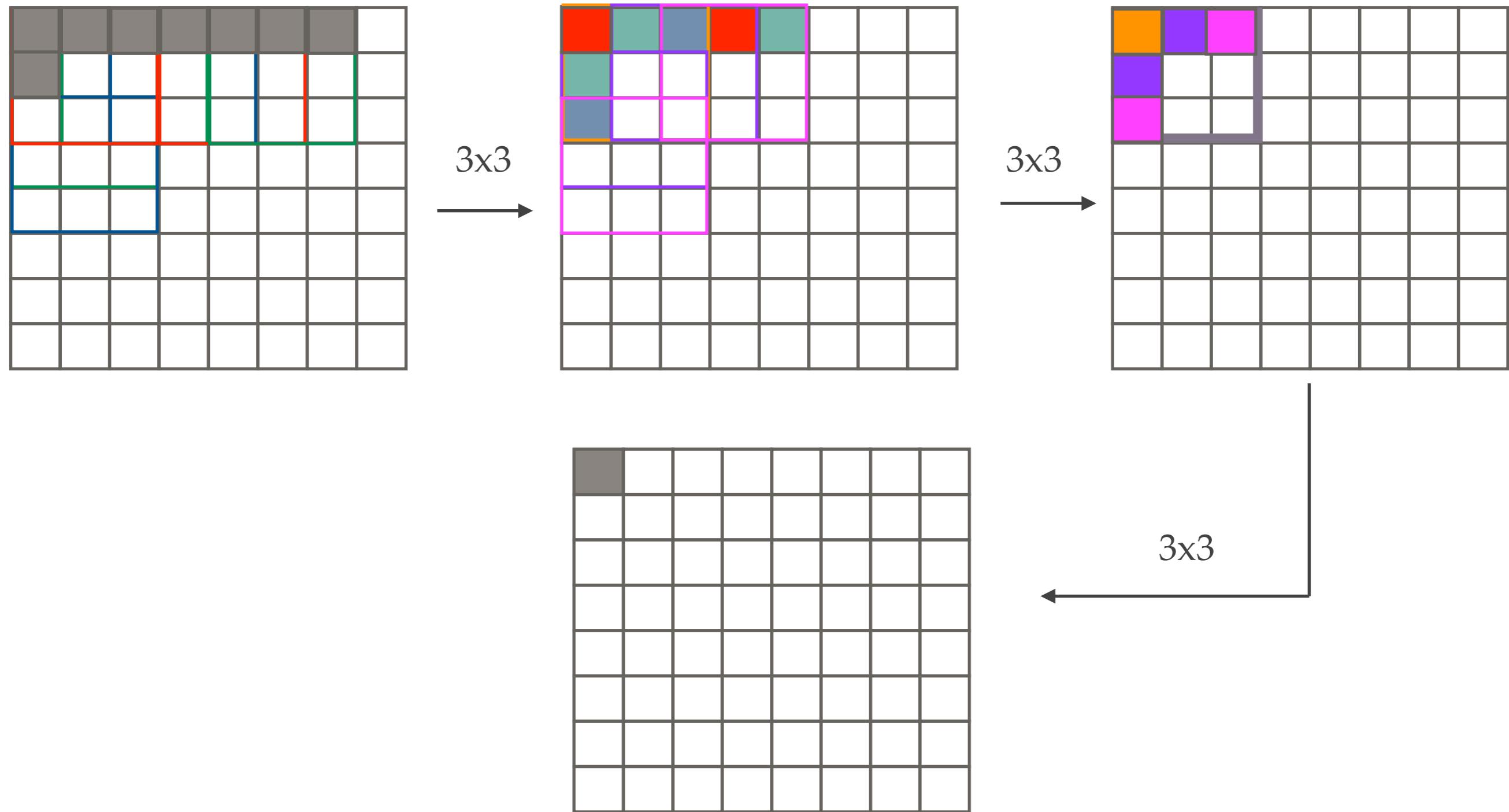
VGGNet



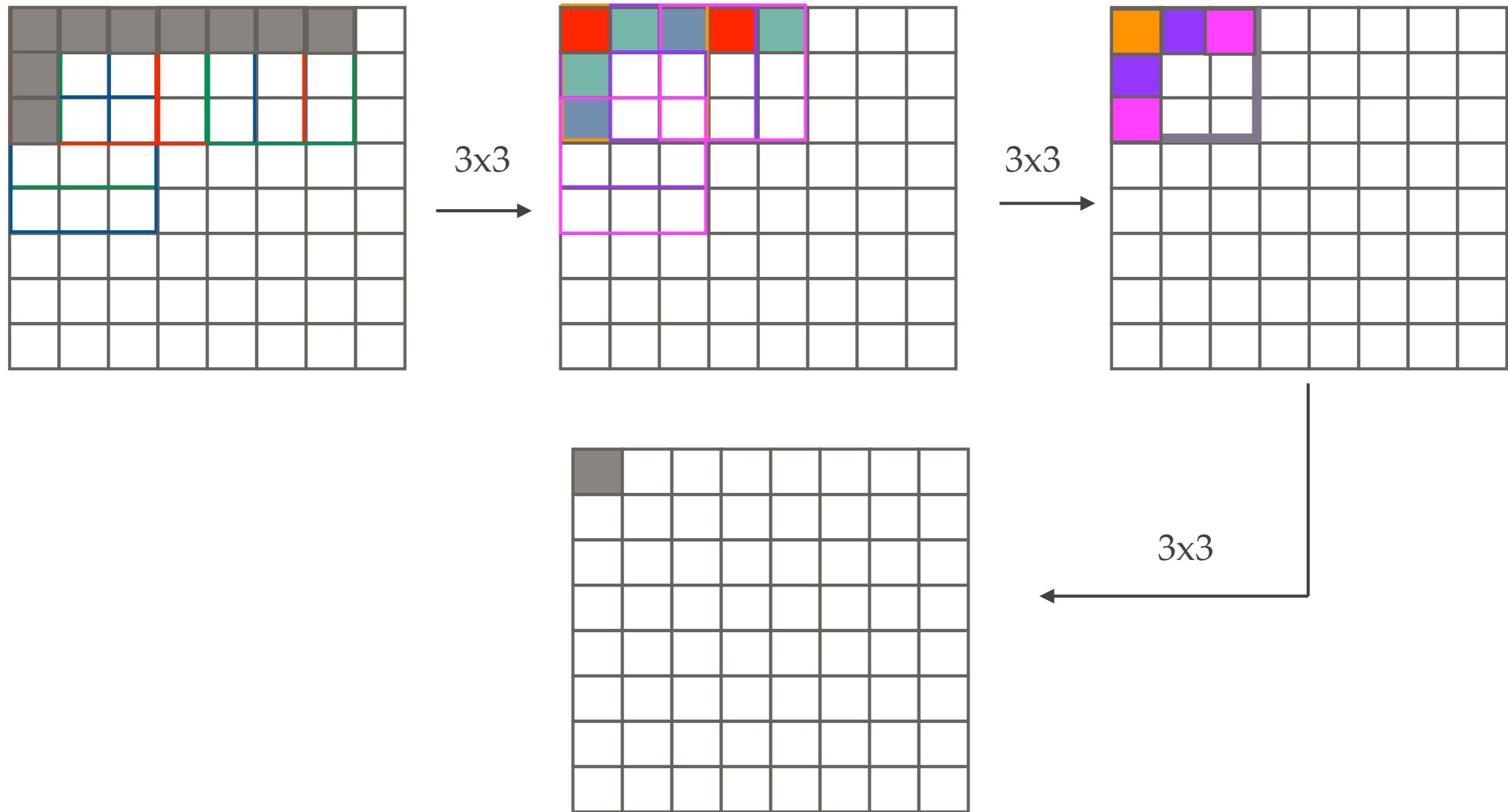
VGGNet



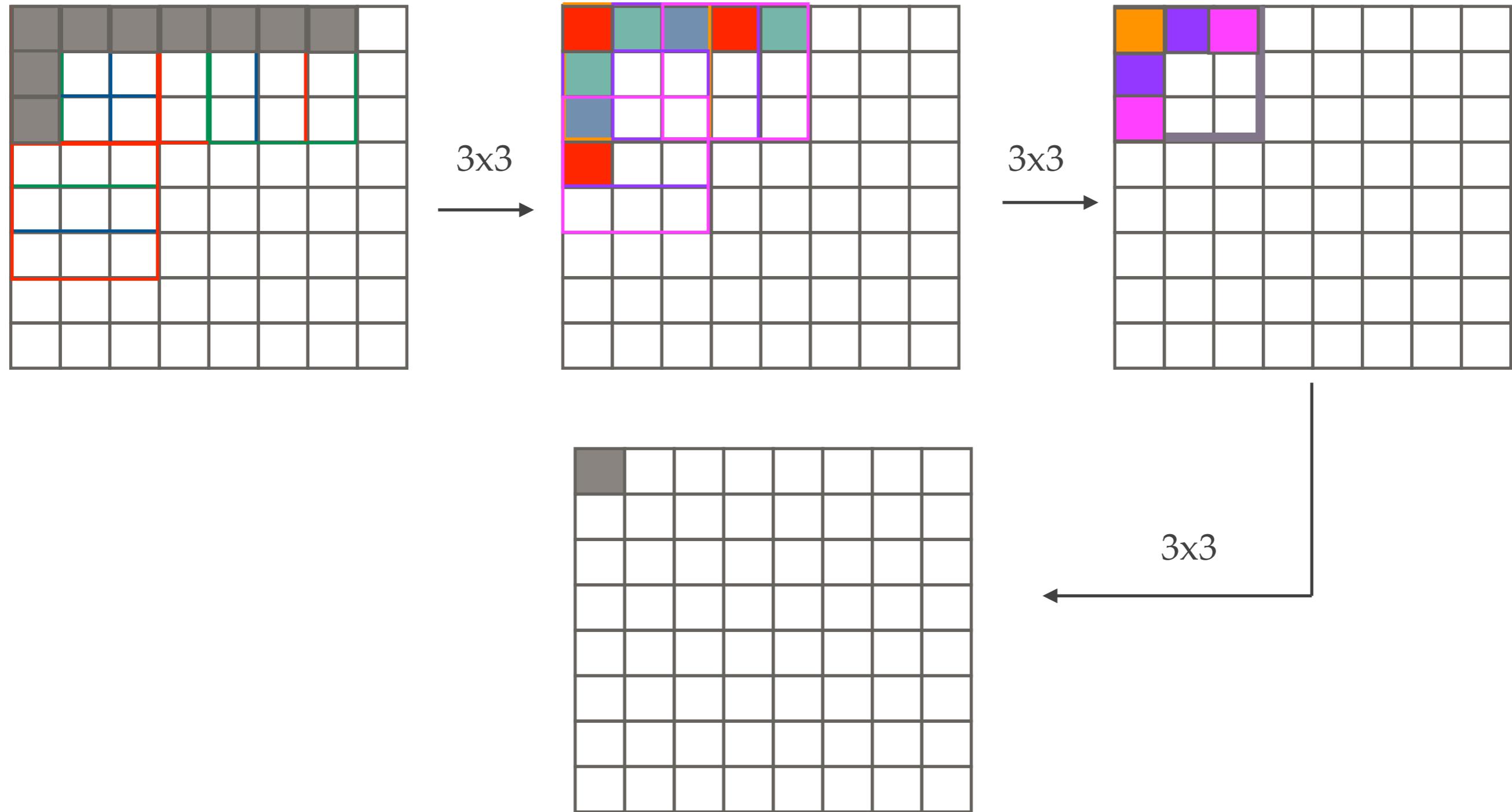
VGGNet



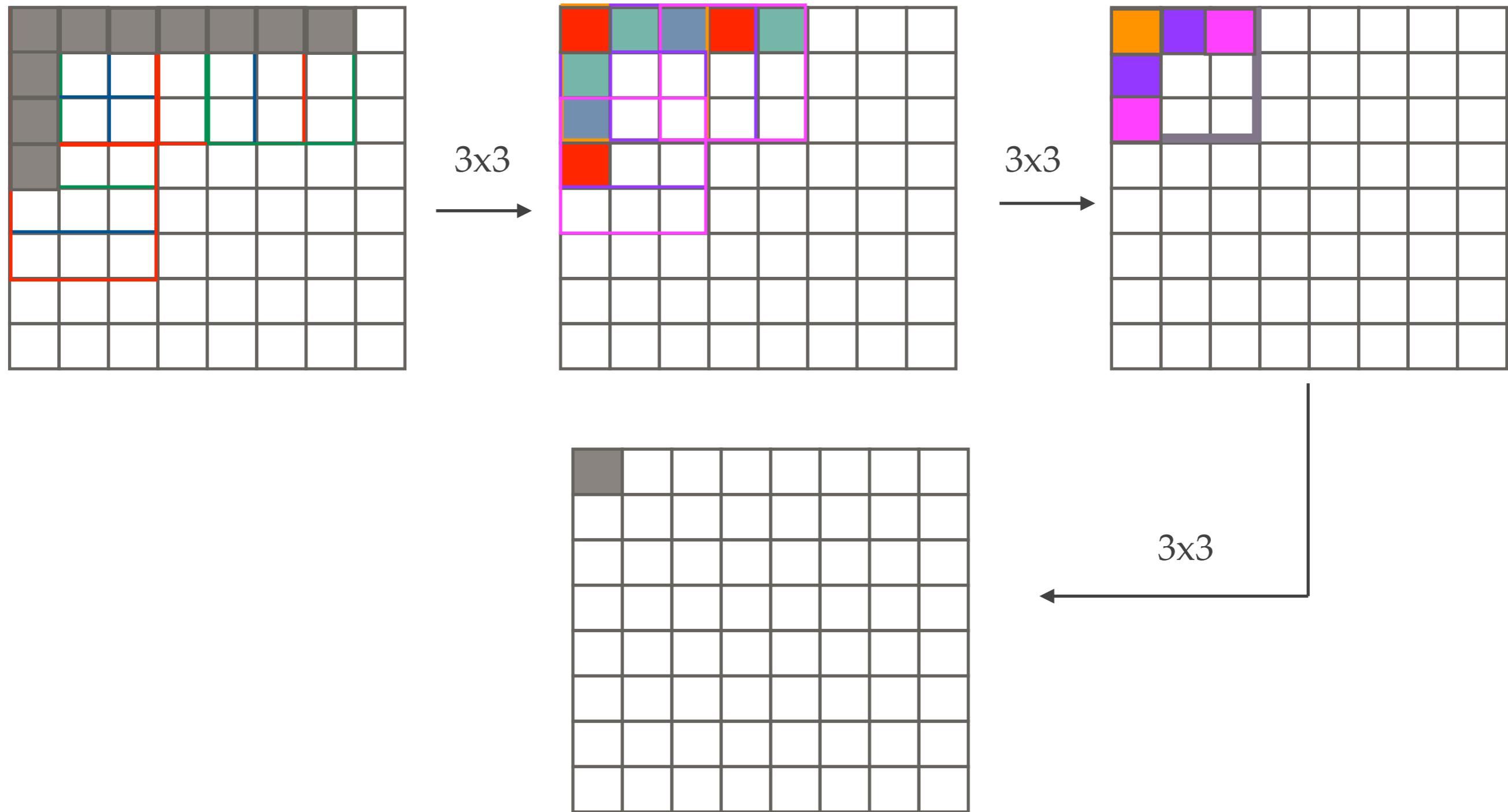
VGGNet



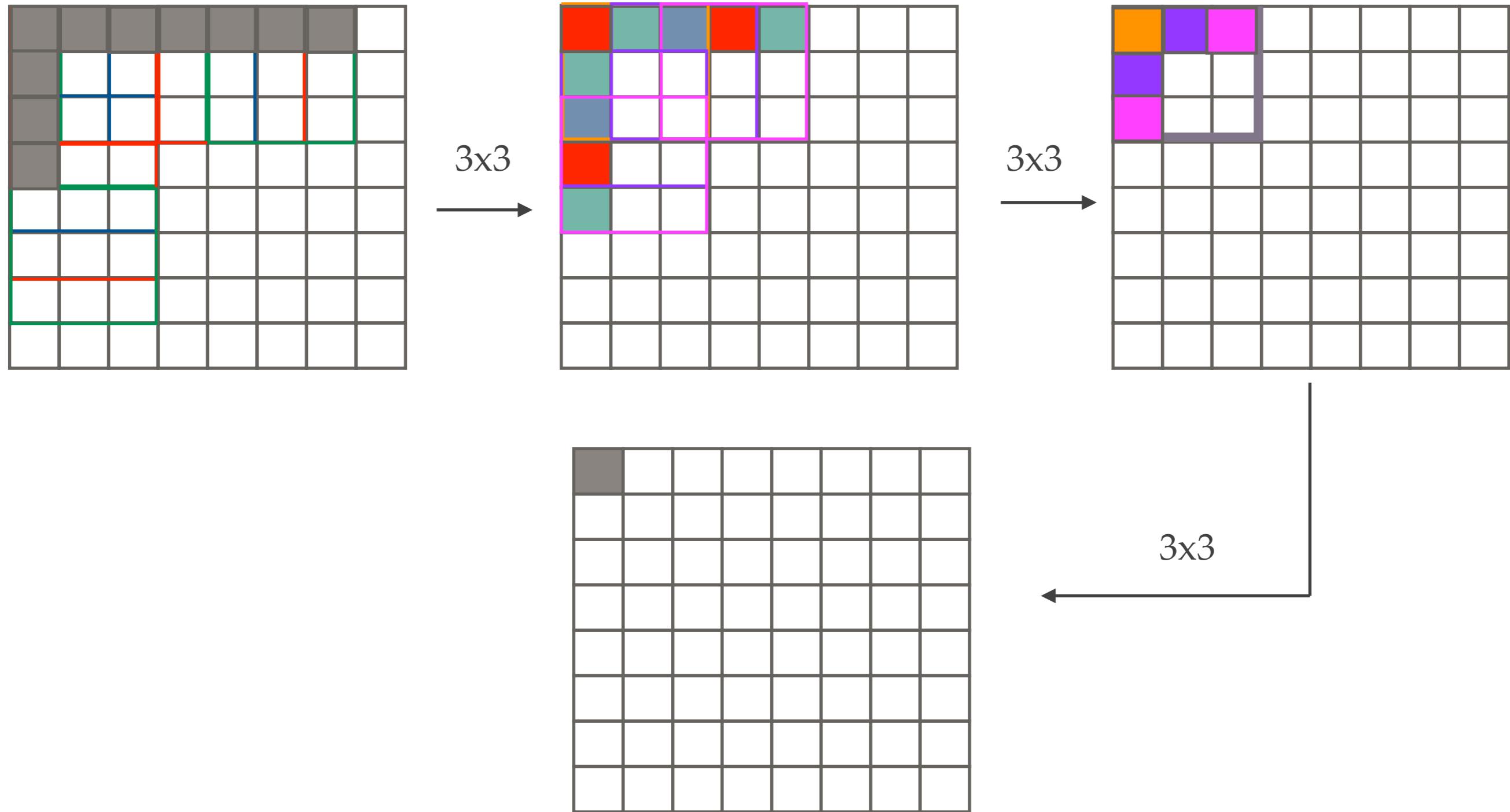
VGGNet



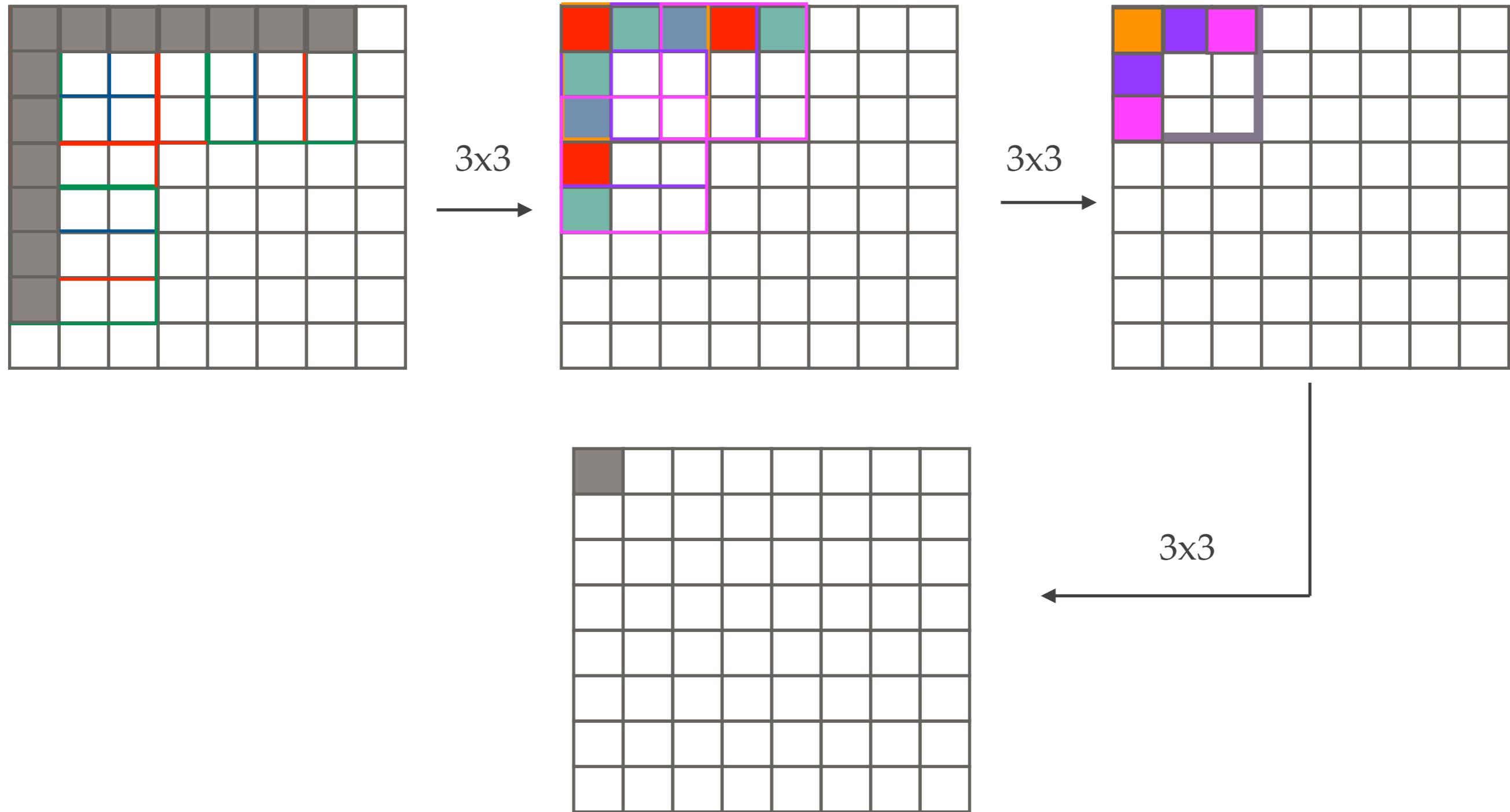
VGGNet



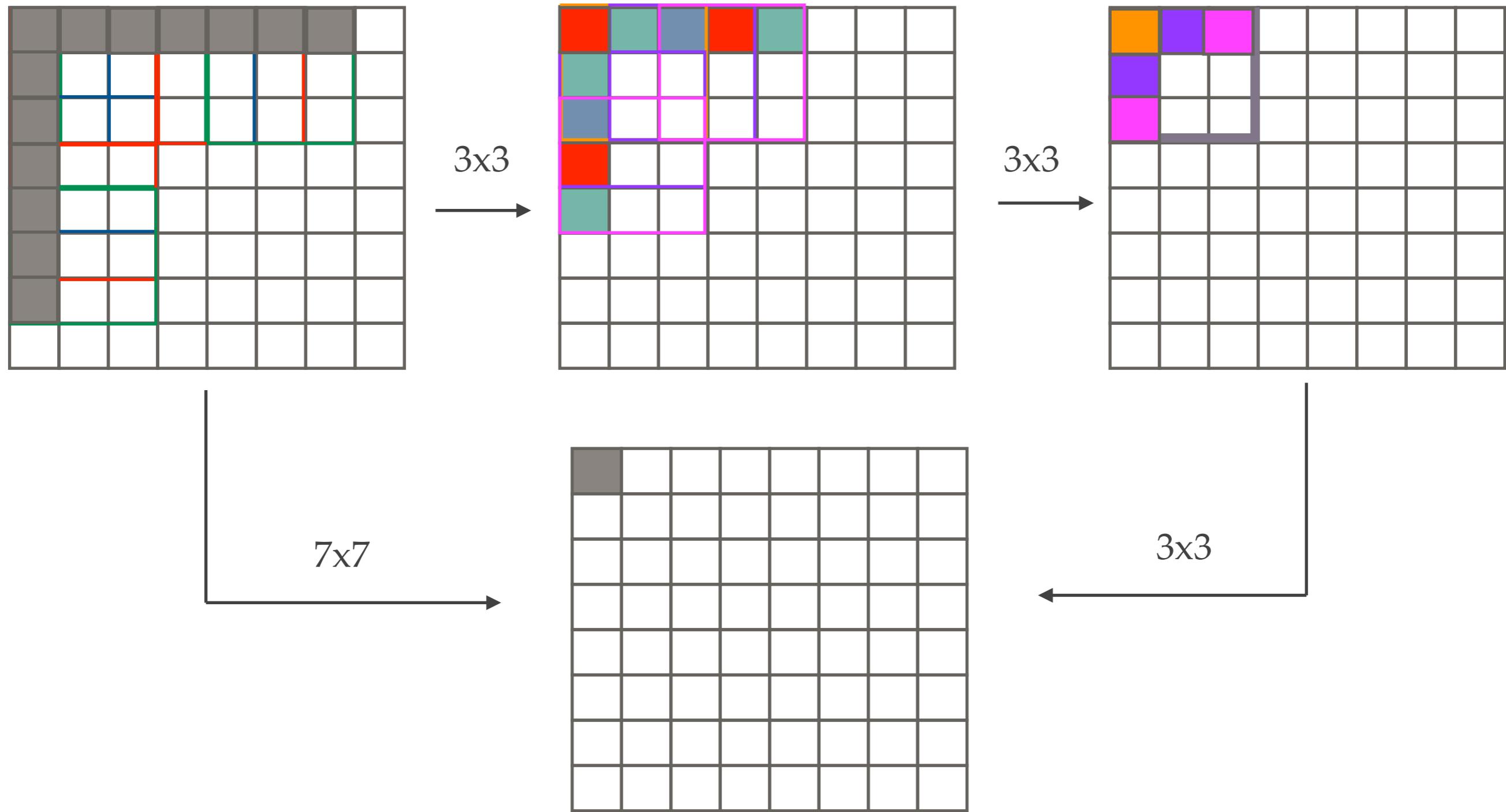
VGGNet



VGGNet

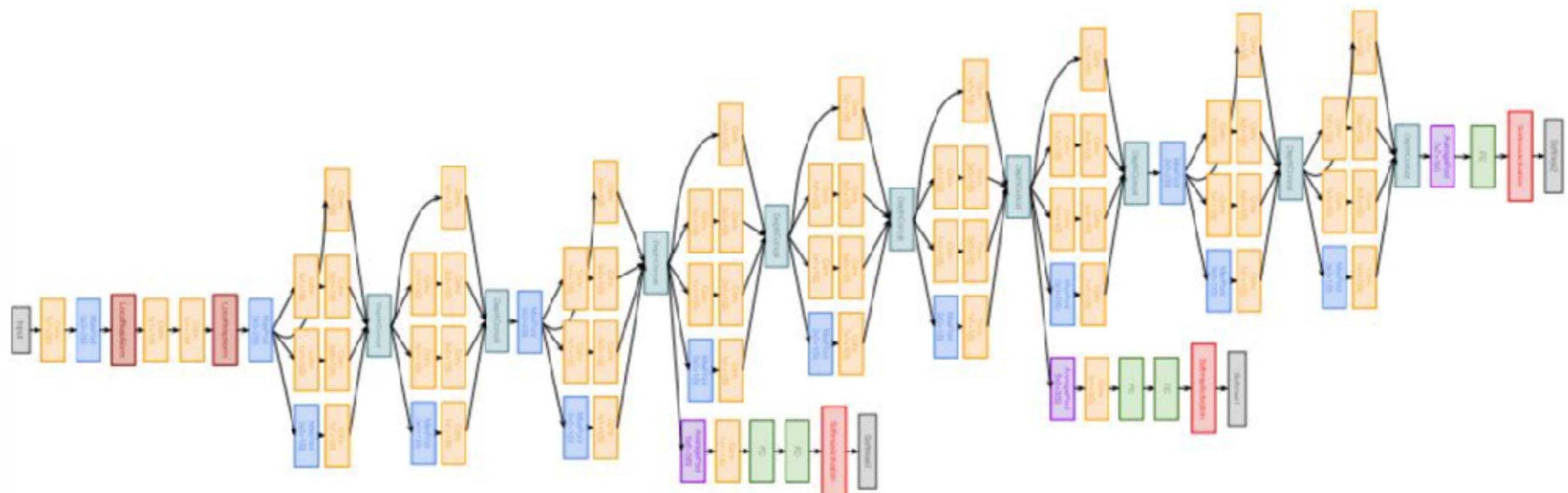


VGGNet



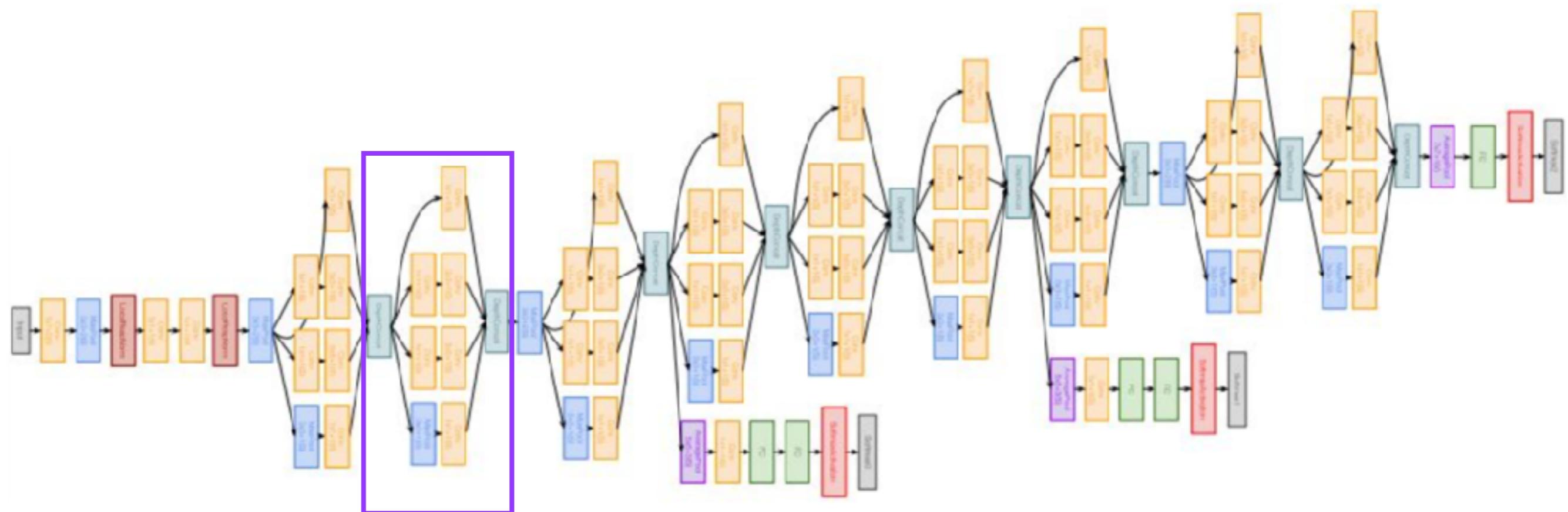
GoogLeNet (Inception)

[Szegedy et al., 2014]



GoogLeNet (Inception)

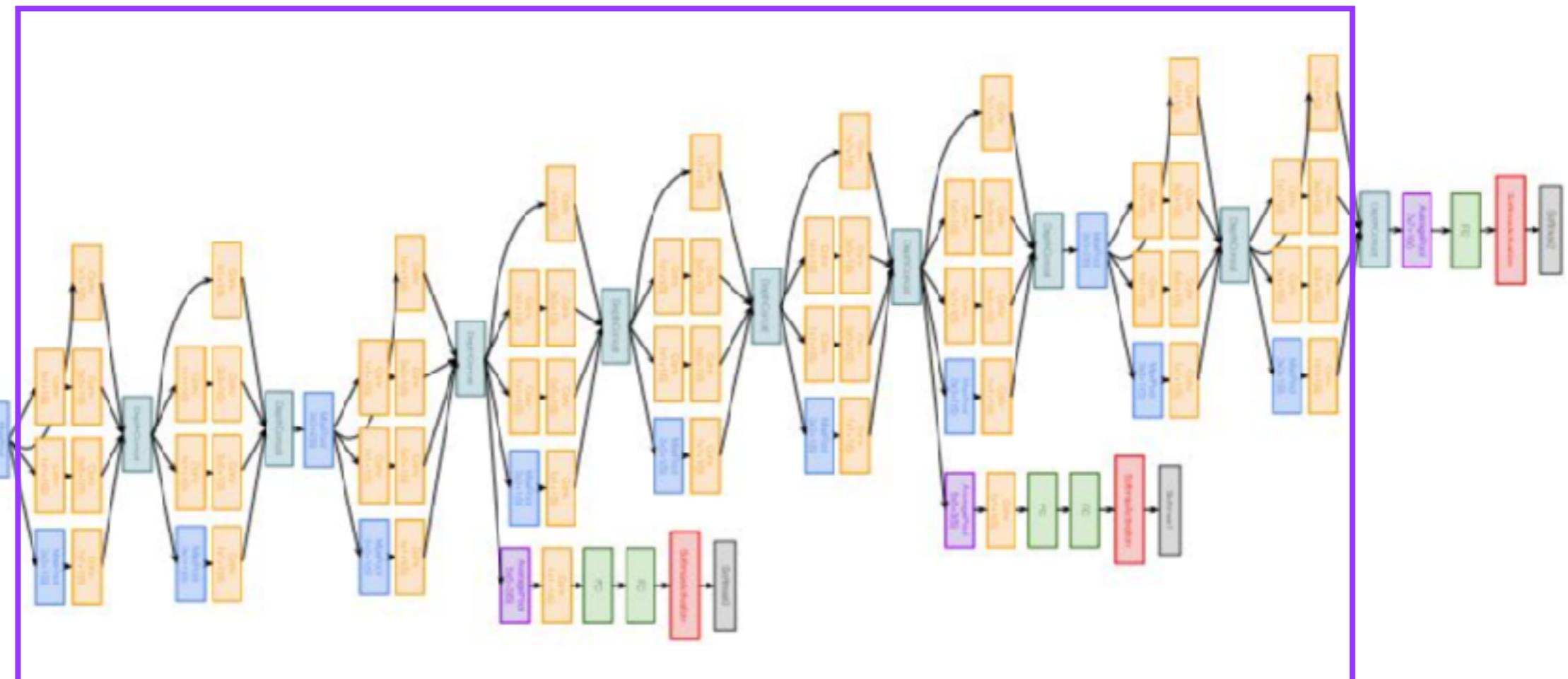
[Szegedy et al., 2014]



Módulo Inception

GoogLeNet (Inception)

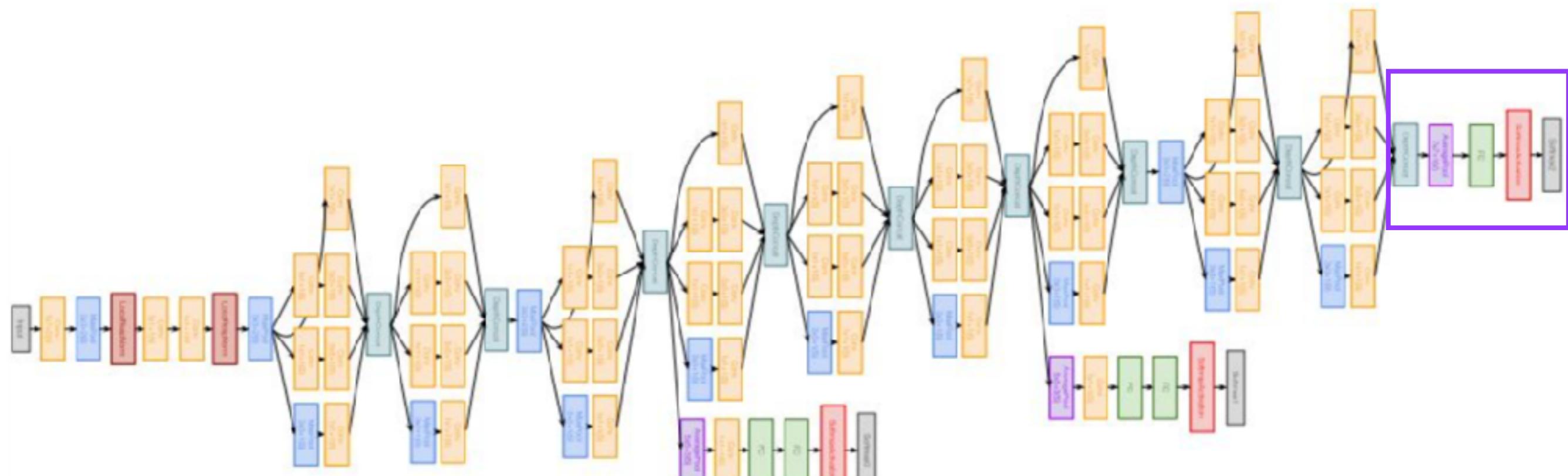
[Szegedy et al., 2014]



9 módulos Inception

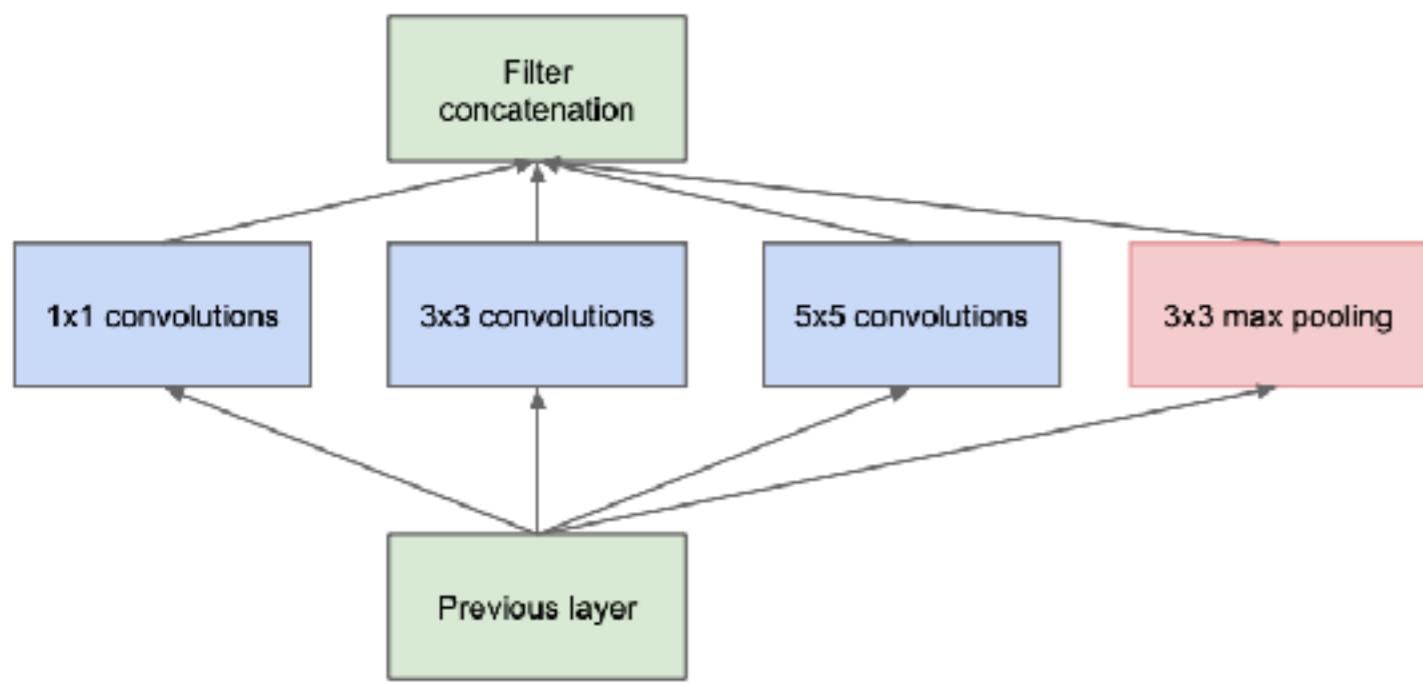
GoogLeNet (Inception)

[Szegedy et al., 2014]

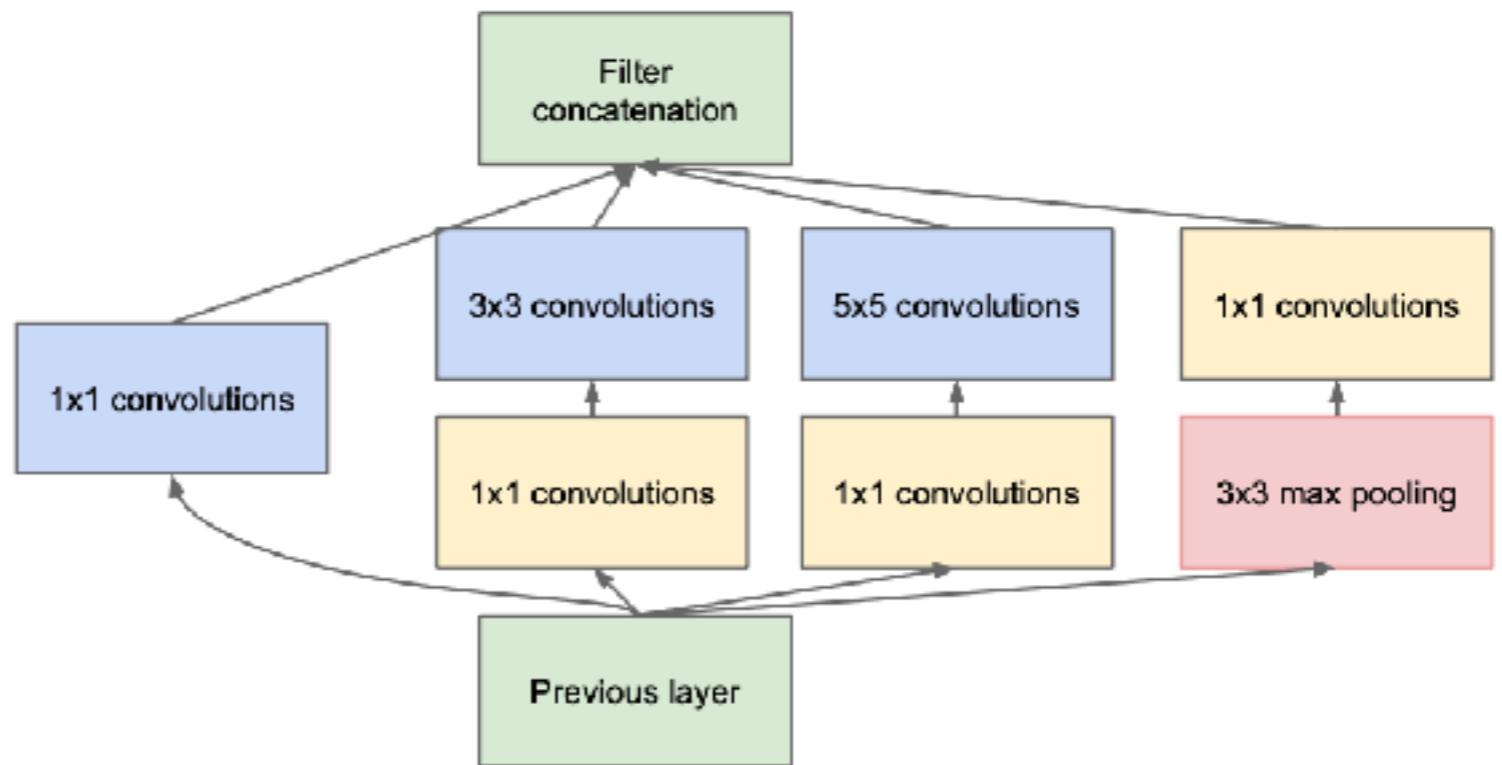


GoogLeNet (Inception)

[Szegedy et al., 2014]



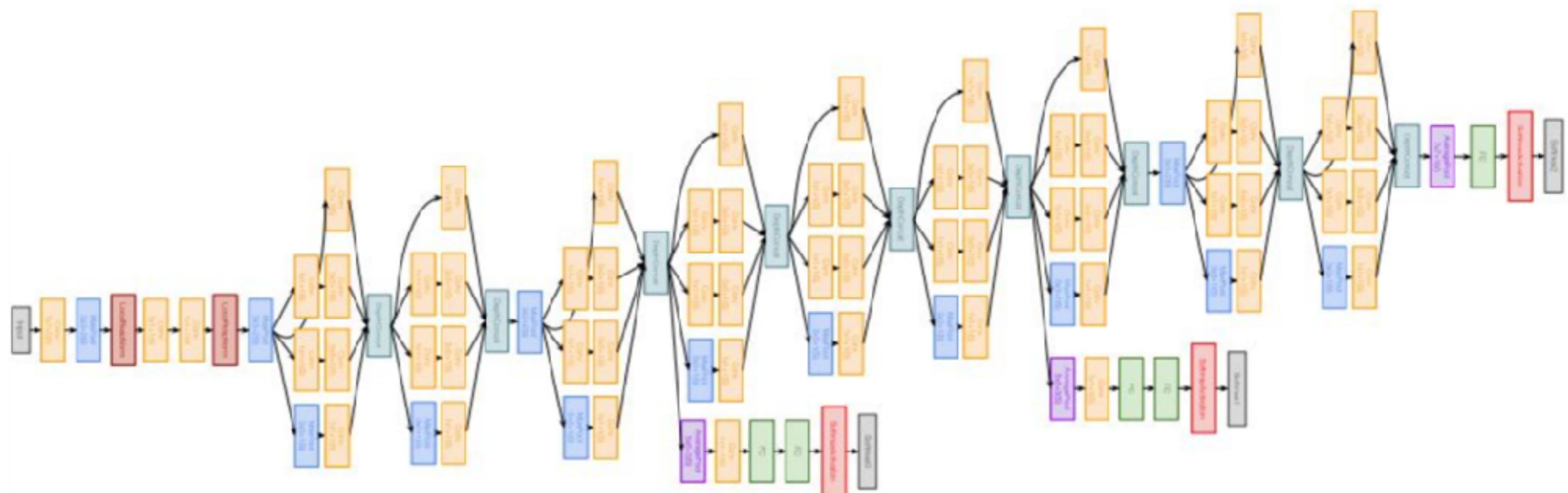
Módulo Inception (naive)



Módulo Inception con reducción de la dimensionalidad

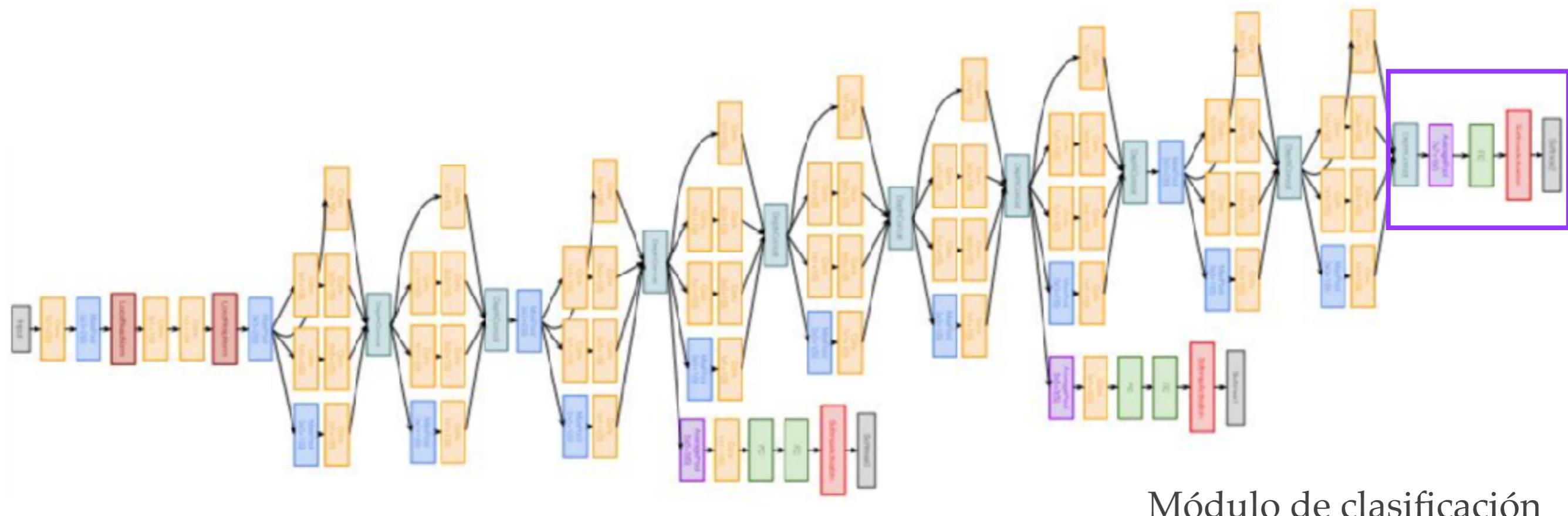
GoogLeNet (Inception)

[Szegedy et al., 2014]



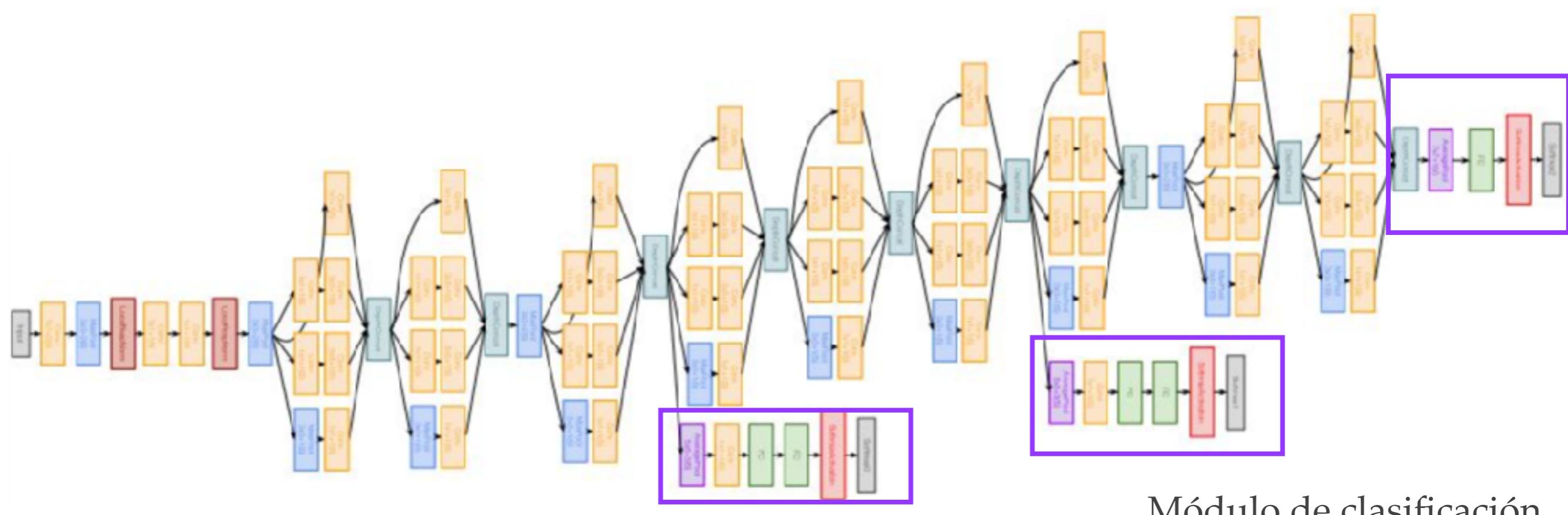
GoogLeNet (Inception)

[Szegedy et al., 2014]



GoogLeNet (Inception)

[Szegedy et al., 2014]



Estrategia de reinyección del gradiente mediante una clasificación adicional

ResNet

[He et al., 2016]

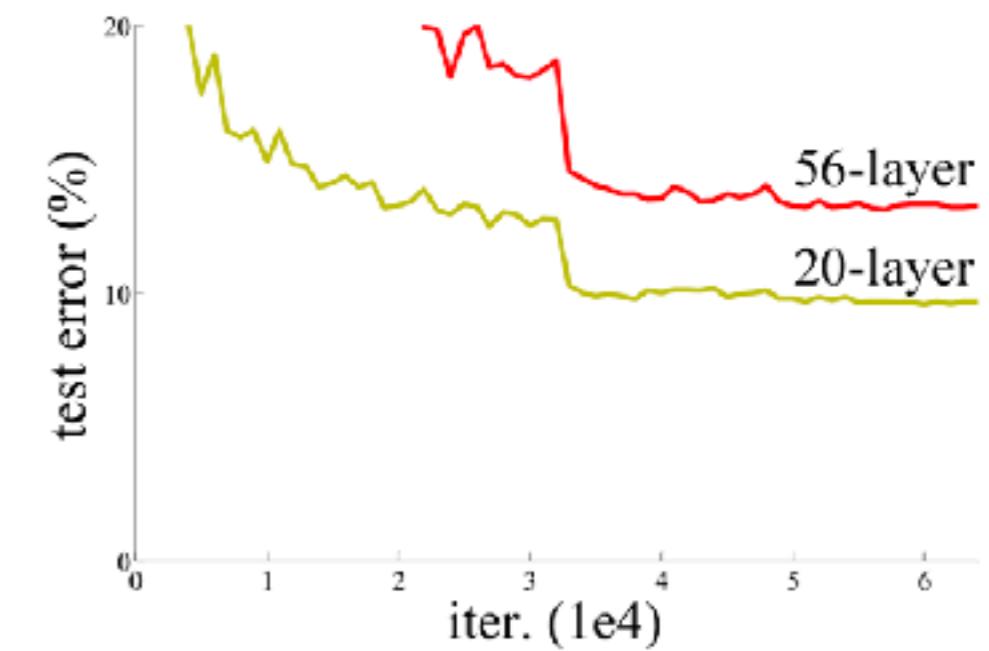
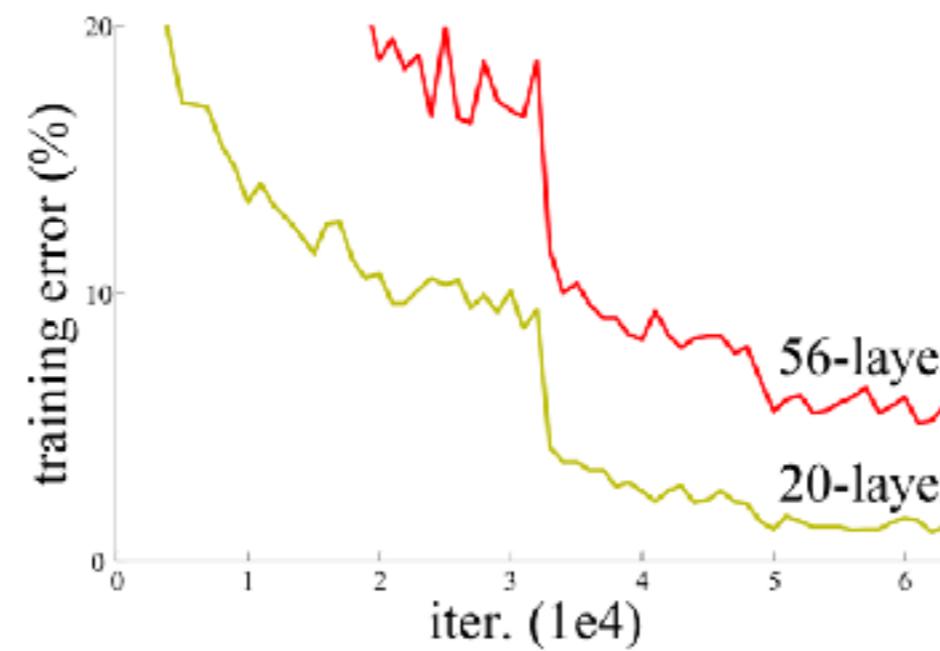
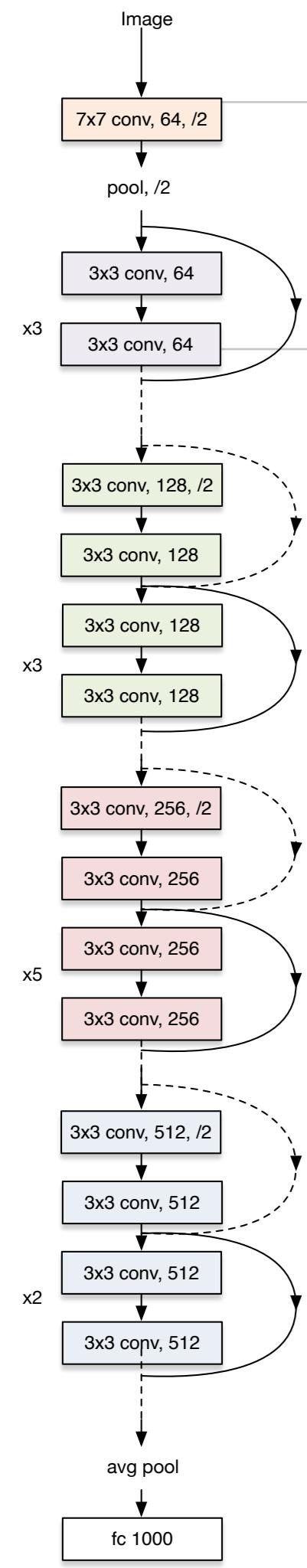
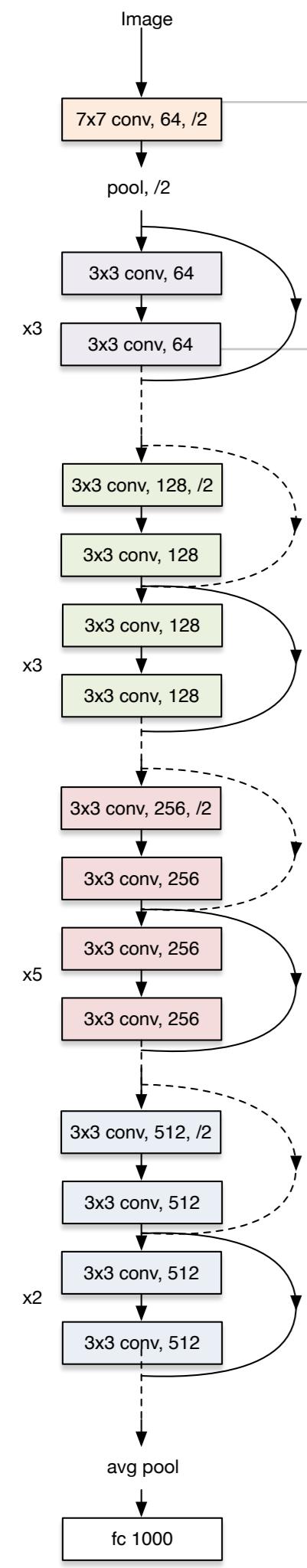


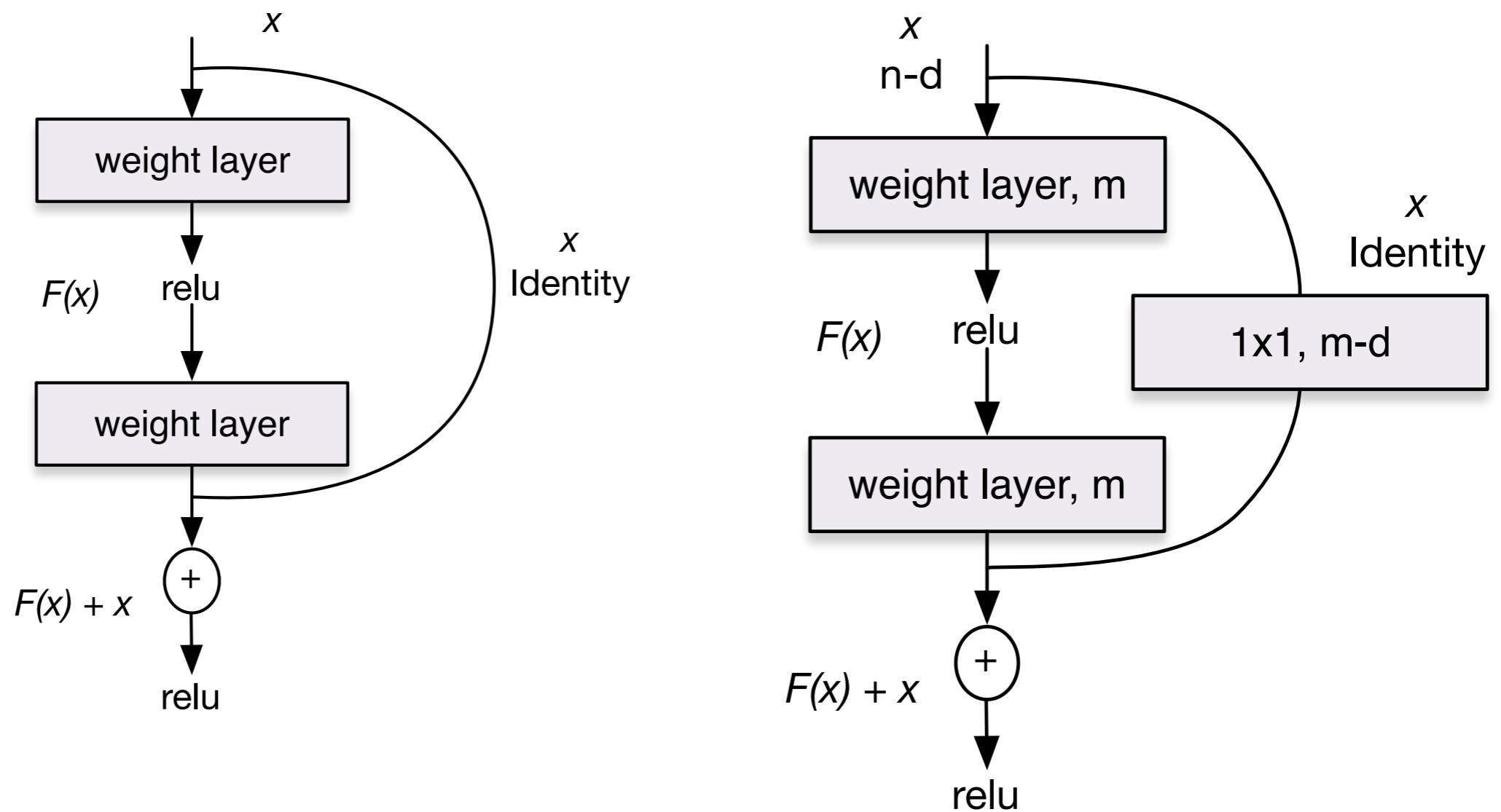
Image Source: He et al., 2016

Plantean como atacar el problema de la degradación

ResNet



Hipótesis [He et al., 2016]: es más fácil optimizar un mapeo residual que el mapeo original. Sea $H(x)$ el mapeo original, proponen optimizar $F(x) = H(x) - x$



ResNet

[He et al., 2016]

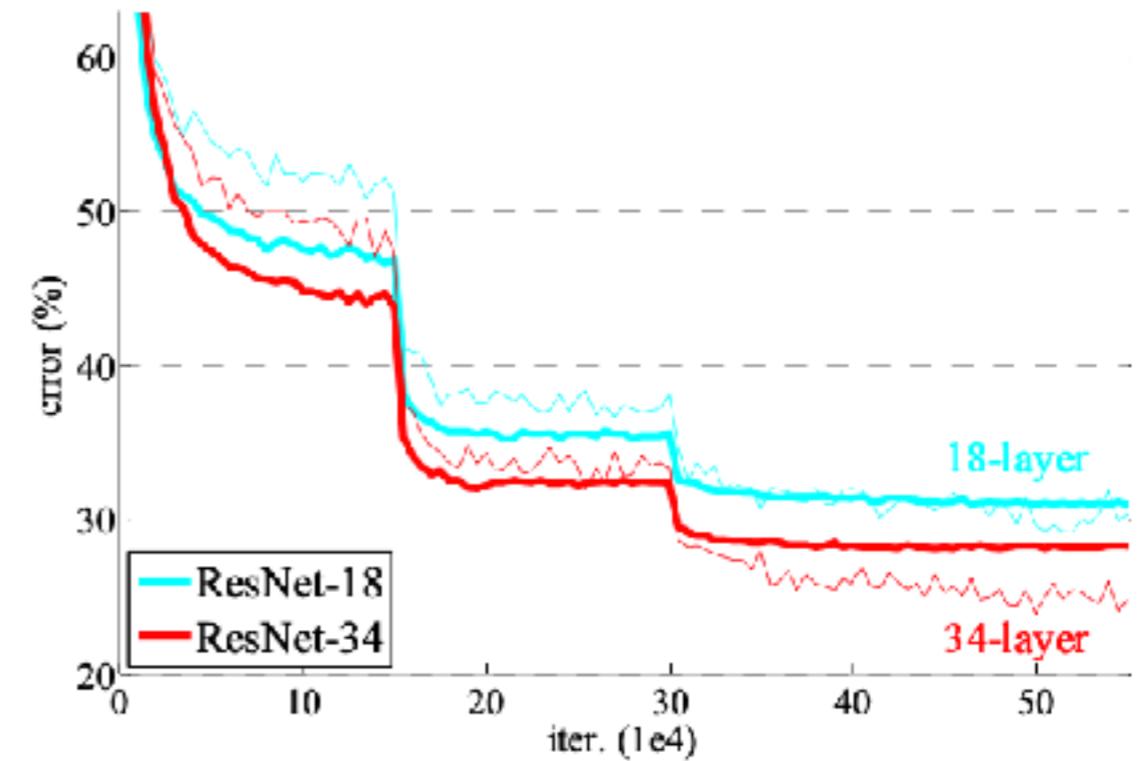
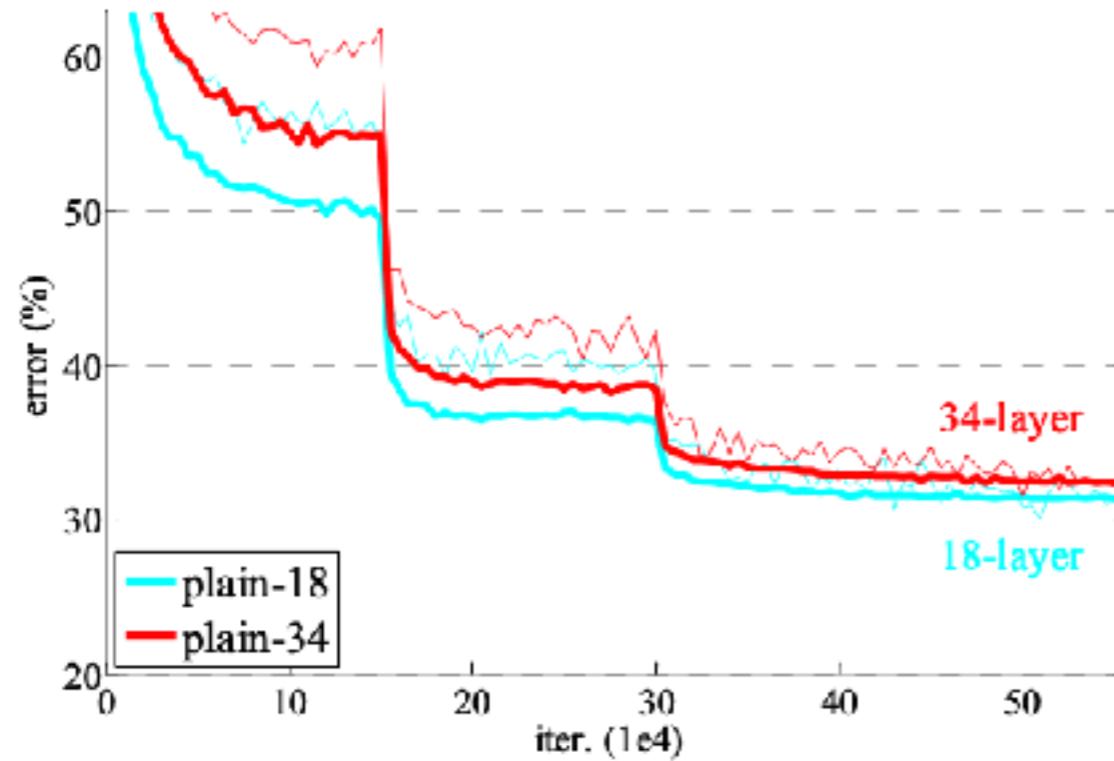


Figure 4. Training on ImageNet. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.