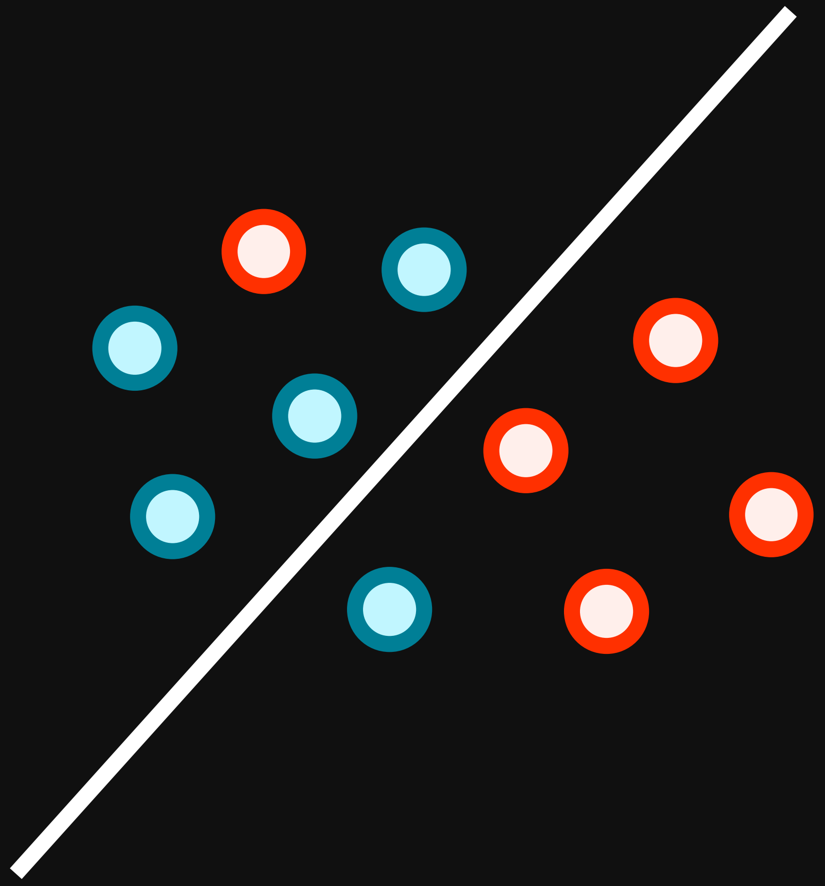
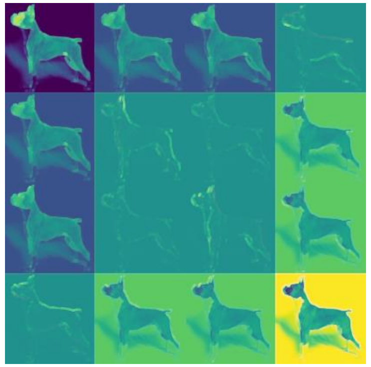
Basics of Machine Learning

Dmitry Ryabokon, github.com/dryabokon



Lesson 17
Intro to Deep Learning



Intro to Deep Learning

Summary

- CNNs out of box
- Some datasets available for research
- Basic Operations

CNNs out of box

The popular networks

Classification

- LeNet Model
- AlexNet Model
- VGG Model
- ResNet Paper
- YOLO9000 Paper
- DenseNet Paper

Segmentation

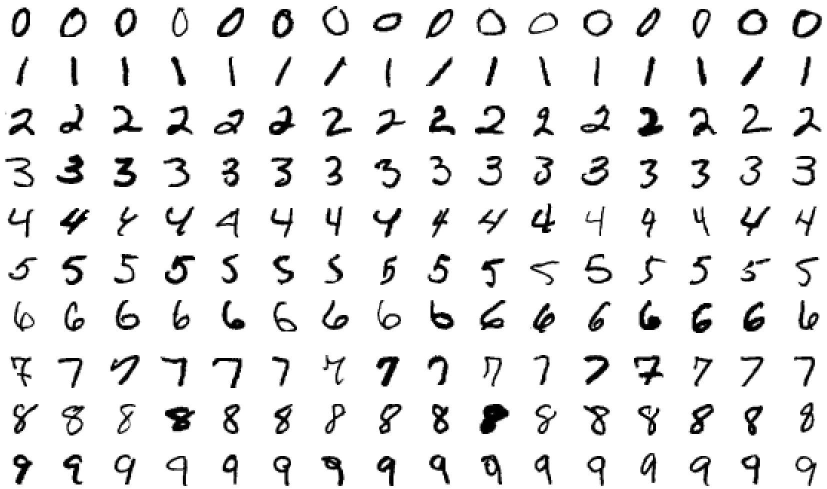
- FCN8 Paper
- SegNet Paper
- U-Net Paper
- E-Net Paper
- ResNetFCN Paper
- PSPNet Paper
- Mask RCNN Paper

Detection

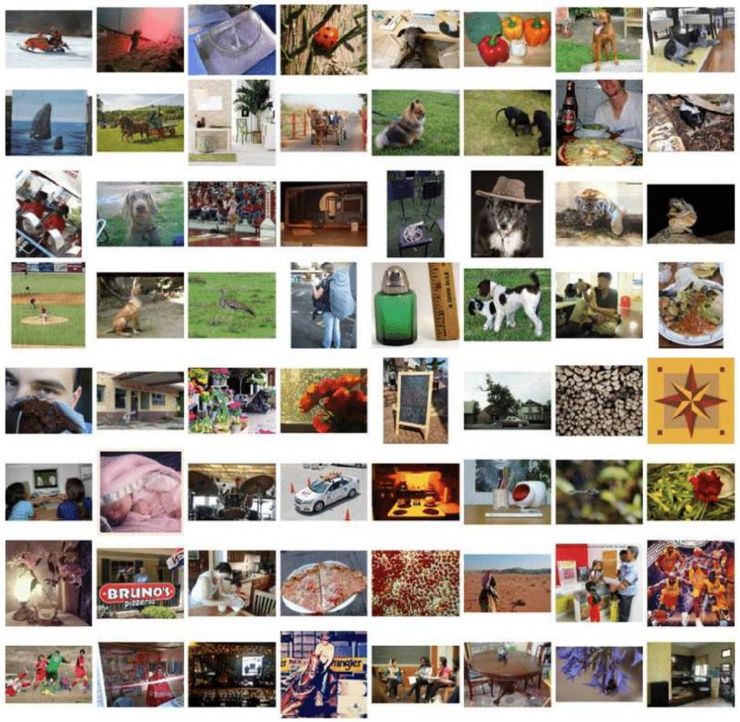
- Faster RCNN Paper
- SSD Paper
- YOLOv2 Paper
- R-FCN Paper

CNNs out of box

Some datasets available for research



MNIST: 10 classes, ~7000 ex. per class

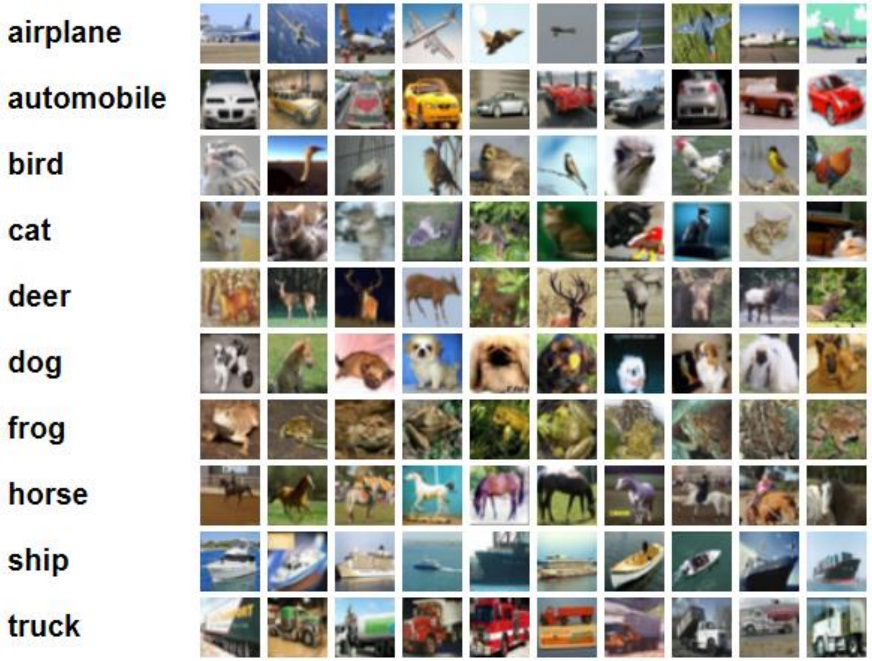


ImageNet: 1000 classes, ~100 ex per class

CNNs out of box

Some datasets available for research

The Street View House Numbers
10 classes, ~2000 ex. per class

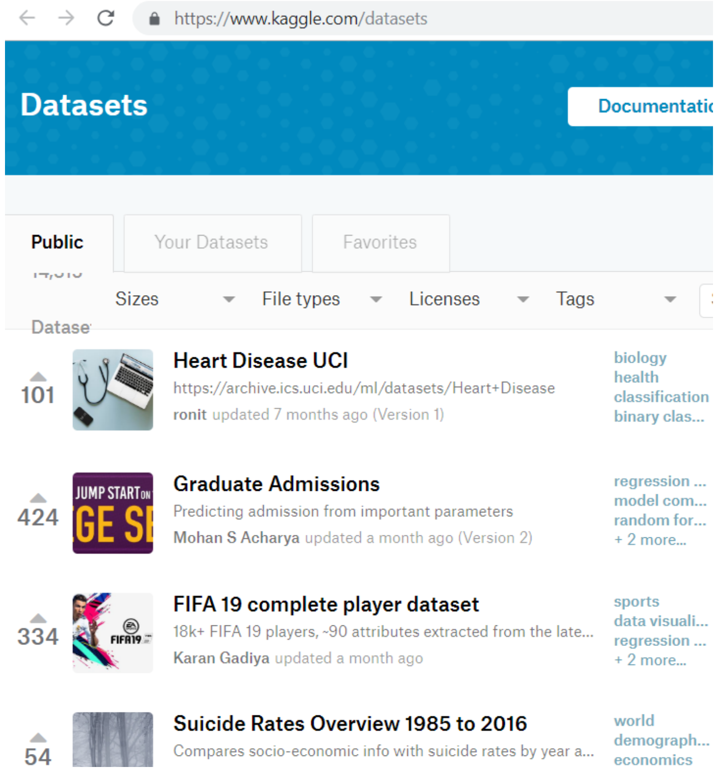


CIFAR: 10 classes, 6000 ex. per class
100 classes, 600 ex per class

CNNs out of box

Some datasets available for research

Olivetti database: 40 classes



101 **Heart Disease UCI**
https://archive.ics.uci.edu/ml/datasets/Heart+Disease
ronit updated 7 months ago (Version 1)
biology
health
classification
binary clas...

424 **Graduate Admissions**
Predicting admission from important parameters
Mohan S Acharya updated a month ago (Version 2)
regression ...
model com...
random for...
+ 2 more...

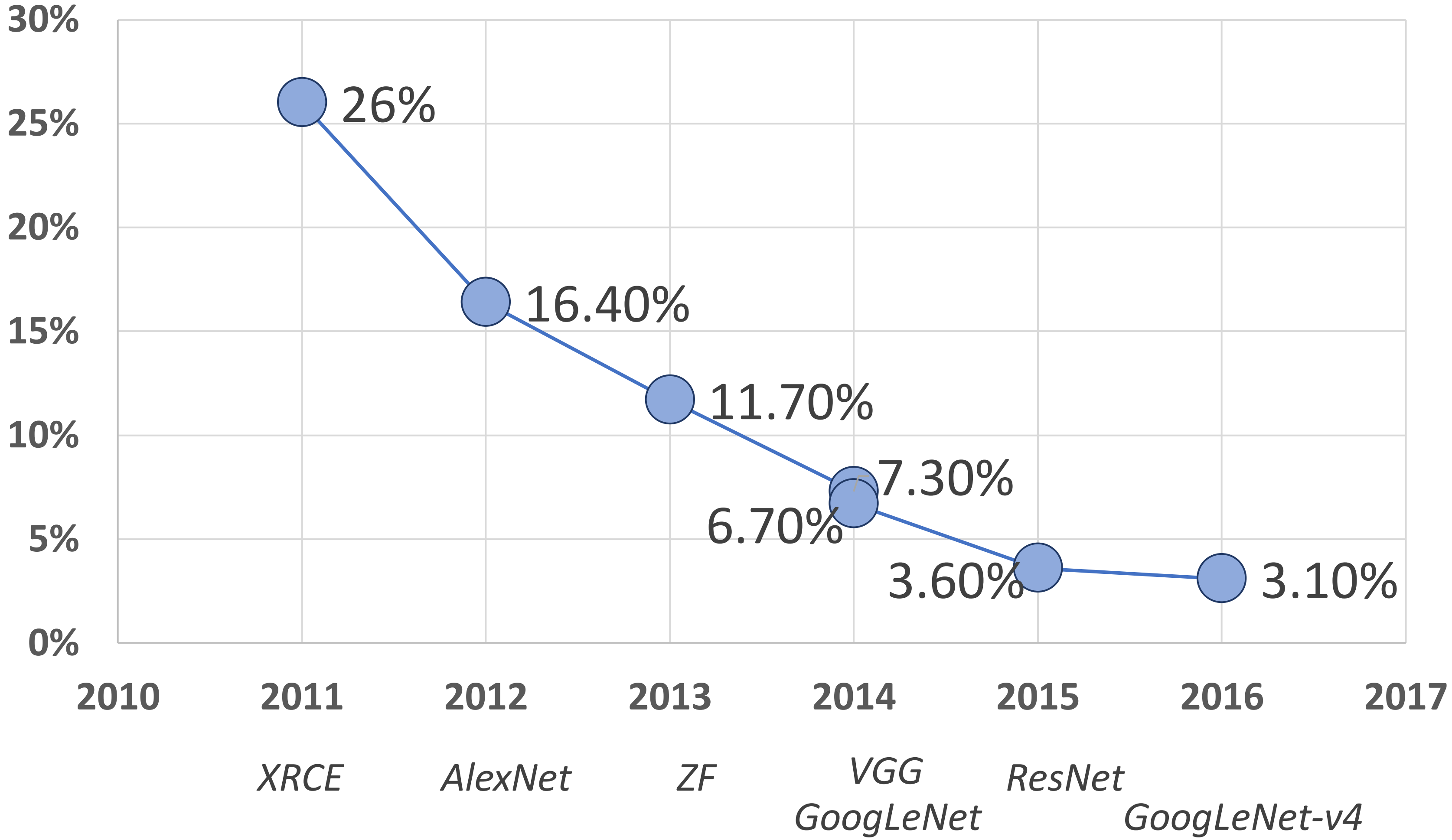
334 **FIFA 19 complete player dataset**
18k+ FIFA 19 players, ~90 attributes extracted from the late...
Karan Gadiya updated a month ago
sports
data visuali...
regression ...
+ 2 more...

54 **Suicide Rates Overview 1985 to 2016**
Compares socio-economic info with suicide rates by year a...
world
demograph...
economics

.. and much more @ kaggle

CNNs out of box

ImageNet Classification error



CNNs out of box

Usage of the neural networks

identify the name of a street (in France) from an image

compressing and decompressing images

semantic image segmentation

real-world image text extraction.

image classification

image-to-text

unsupervised learning

3D object reconstruction

image matching and retrieval

automatic speech recognition

localizing and identifying multiple objects in a single image

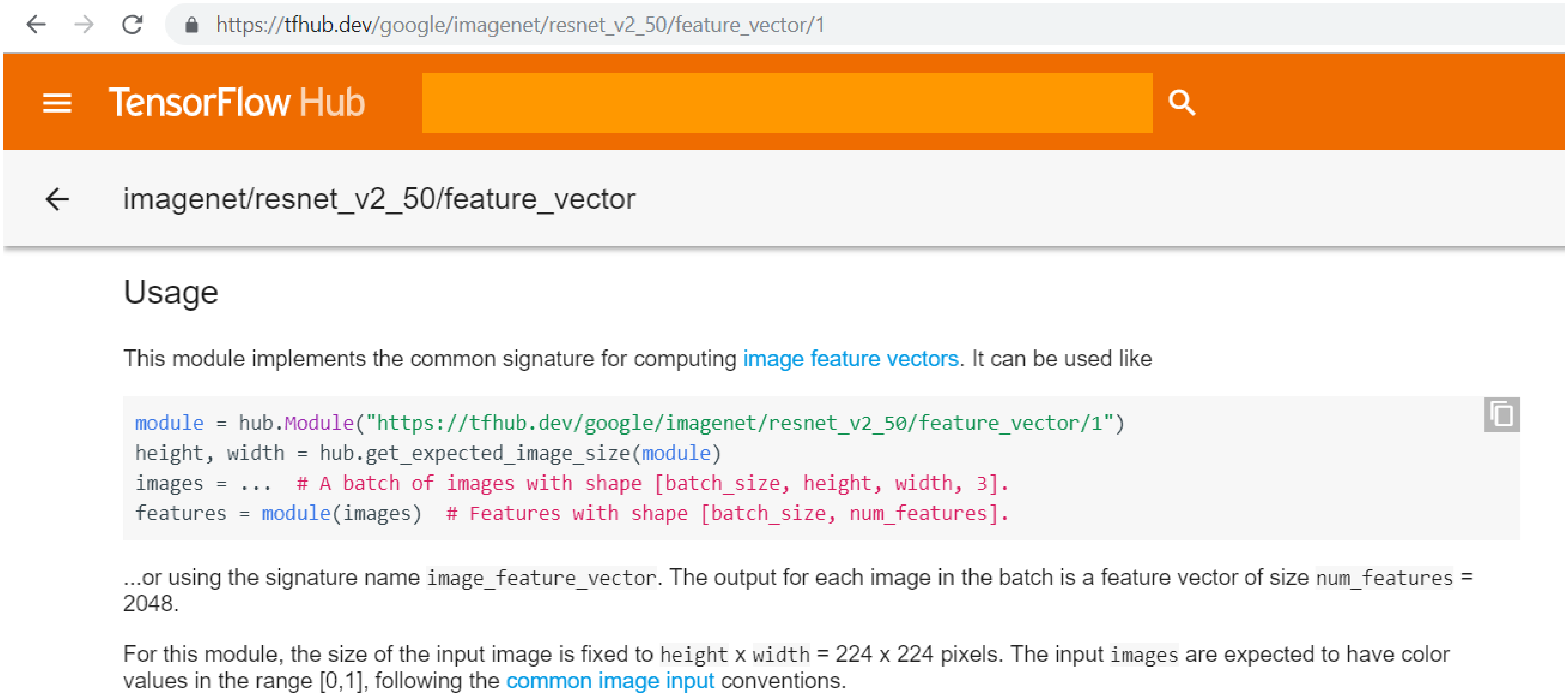
computer vision

discovery of latent 3D keypoints

predicting future video frames

https://github.com/keras-team/keras/tree/master/examples
https://github.com/tensorflow/models/tree/master/research

CNNs out of box



https://tfhub.dev/google/imagenet/resnet_v2_50/feature_vector/1
TensorFlow Hub

imagenet/resnet_v2_50/feature_vector

Usage

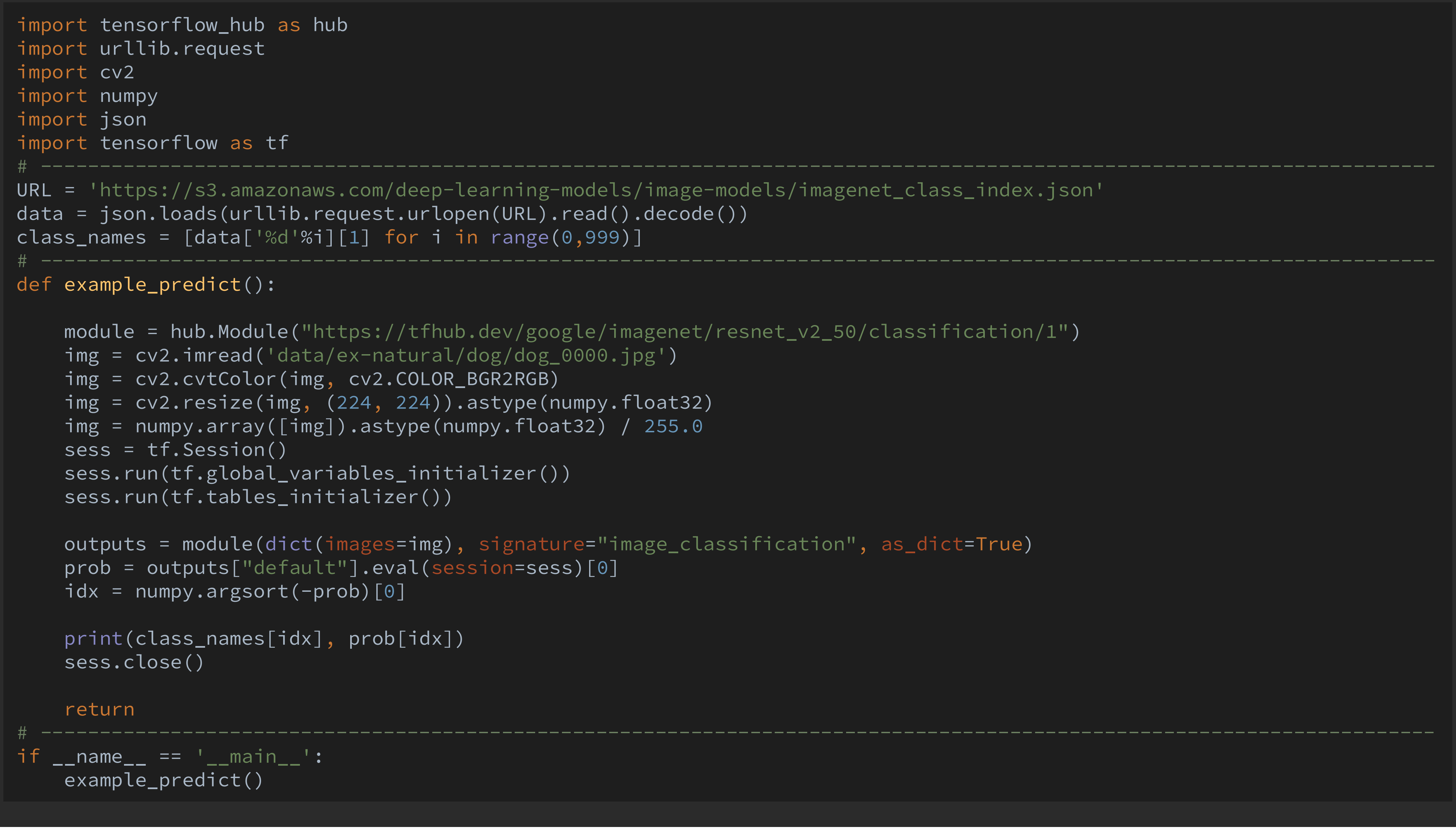
This module implements the common signature for computing image feature vectors. It can be used like

```
module = hub.Module("https://tfhub.dev/google/imagenet/resnet_v2_50/feature_vector/1")
height, width = hub.get_expected_image_size(module)
images = ...  # A batch of images with shape [batch_size, height, width, 3].
features = module(images)  # Features with shape [batch_size, num_features].
```

...or using the signature name `image_feature_vector`. The output for each image in the batch is a feature vector of size `num_features` = 2048.

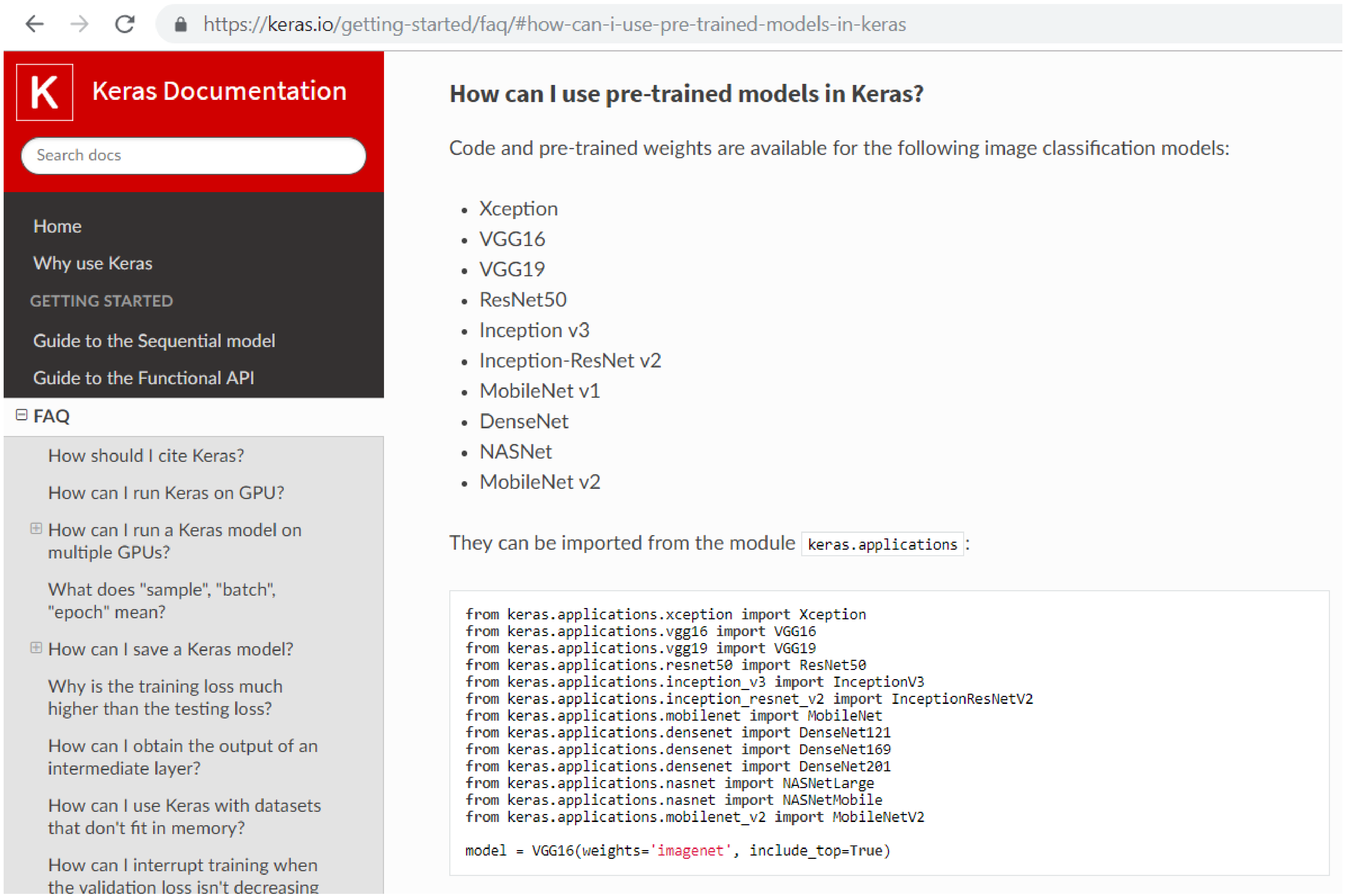
For this module, the size of the input image is fixed to `height` x `width` = 224 x 224 pixels. The input `images` are expected to have color values in the range [0,1], following the common image input conventions.

CNNs out of box



```python
import tensorflow_hub as hub
import urllib.request
import cv2
import numpy
import json
import tensorflow as tf
# ----------------------------------------------------------------------------------------------------------------------
URL = 'https://s3.amazonaws.com/deep-learning-models/image-models/imagenet_class_index.json'
data = json.loads(urllib.request.urlopen(URL).read().decode())
class_names = [data['%d'%i][1] for i in range(0,999)]
# ----------------------------------------------------------------------------------------------------------------------
def example_predict():

    module = hub.Module("https://tfhub.dev/google/imagenet/resnet_v2_50/classification/1")
    img = cv2.imread('data/ex-natural/dog/dog_0000.jpg')
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    img = cv2.resize(img, (224, 224)).astype(numpy.float32)
    img = numpy.array([img]).astype(numpy.float32) / 255.0
    sess = tf.Session()
    sess.run(tf.global_variables_initializer())
    sess.run(tf.tables_initializer())

    outputs = module(dict(images=img), signature="image_classification", as_dict=True)
    prob = outputs["default"].eval(session=sess)[0]
    idx = numpy.argsort(-prob)[0]

    print(class_names[idx], prob[idx])
    sess.close()

    return
# ----------------------------------------------------------------------------------------------------------------------
if __name__ == '__main__':
    example_predict()
```

CNNs out of box


https://keras.io/getting-started/faq/#how-can-i-use-pre-trained-models-in-keras
K
Keras Documentation
Search docs
Home
Why use Keras
GETTING STARTED
Guide to the Sequential model
Guide to the Functional API
⊟ FAQ
How should I cite Keras?
How can I run Keras on GPU?
⊞ How can I run a Keras model on multiple GPUs?
What does "sample", "batch", "epoch" mean?
⊞ How can I save a Keras model?
Why is the training loss much higher than the testing loss?
How can I obtain the output of an intermediate layer?
How can I use Keras with datasets that don't fit in memory?
How can I interrupt training when the validation loss isn't decreasing

How can I use pre-trained models in Keras?

Code and pre-trained weights are available for the following image classification models:

- Xception
- VGG16
- VGG19
- ResNet50
- Inception v3
- Inception-ResNet v2
- MobileNet v1
- DenseNet
- NASNet
- MobileNet v2

They can be imported from the module `keras.applications`:

```
from keras.applications.xception import Xception
from keras.applications.vgg16 import VGG16
from keras.applications.vgg19 import VGG19
from keras.applications.resnet50 import ResNet50
from keras.applications.inception_v3 import InceptionV3
from keras.applications.inception_resnet_v2 import InceptionResNetV2
from keras.applications.mobilenet import MobileNet
from keras.applications.densenet import DenseNet121
from keras.applications.densenet import DenseNet169
from keras.applications.densenet import DenseNet201
from keras.applications.nasnet import NASNetLarge
from keras.applications.nasnet import NASNetMobile
from keras.applications.mobilenet_v2 import MobileNetV2

model = VGG16(weights='imagenet', include_top=True)
```

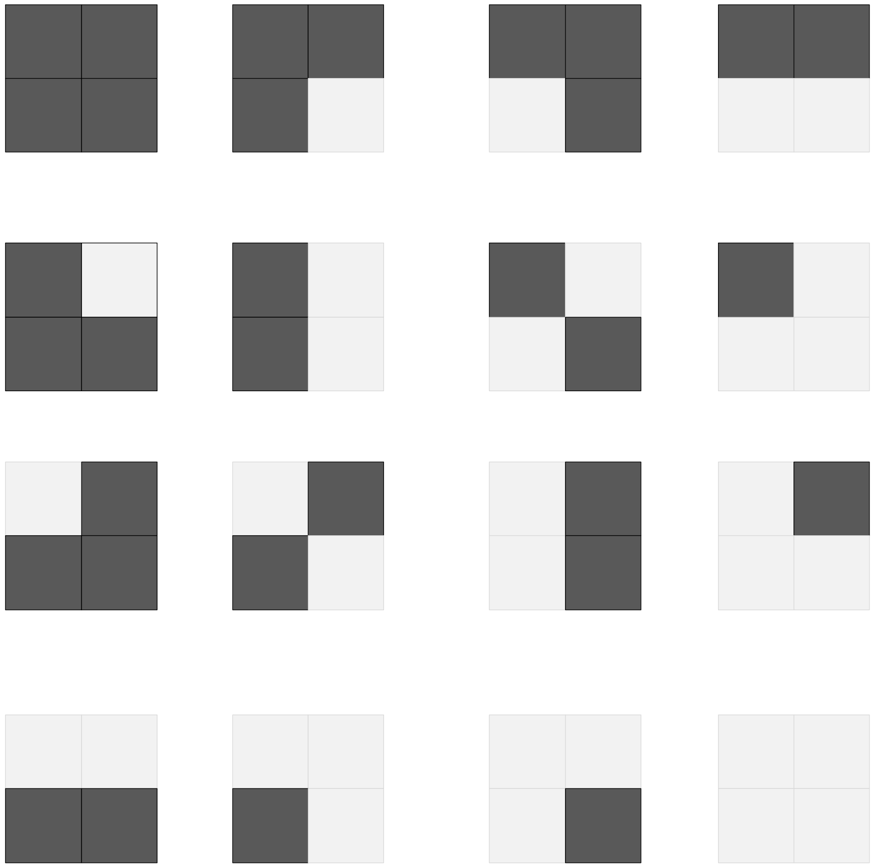
CNNs out of box

```
from keras.applications import MobileNet
from keras.applications.xception import Xception
from keras.applications.mobilenet import preprocess_input
import urllib.request
import cv2
import numpy
import json
from keras import backend as K
K.set_image_dim_ordering('tf')
# ---------------------------------------------------------------------------------------------------------------
-------
URL = 'https://s3.amazonaws.com/deep-learning-models/image-models/imagenet_class_index.json'
data = json.loads(urllib.request.urlopen(URL).read().decode())
class_names = [data['%d'%i][1] for i in range(0,999)]
# ---------------------------------------------------------------------------------------------------------------
-------
def example_predict():

    CNN = MobileNet()
    #CNN = Xception()

    img = cv2.imread('data/ex-natural/dog/dog_0000.jpg')
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    img = cv2.resize(img, (224, 224)).astype(numpy.float32)

    prob = CNN.predict(preprocess_input(numpy.array([img])))
    idx = numpy.argsort(-prob[0])[0]
    print(class_names[idx], prob[0, idx])

    return
# ---------------------------------------------------------------------------------------------------------------
-------
if __name__ == '__main__':
    example_predict()
```

Basic operations

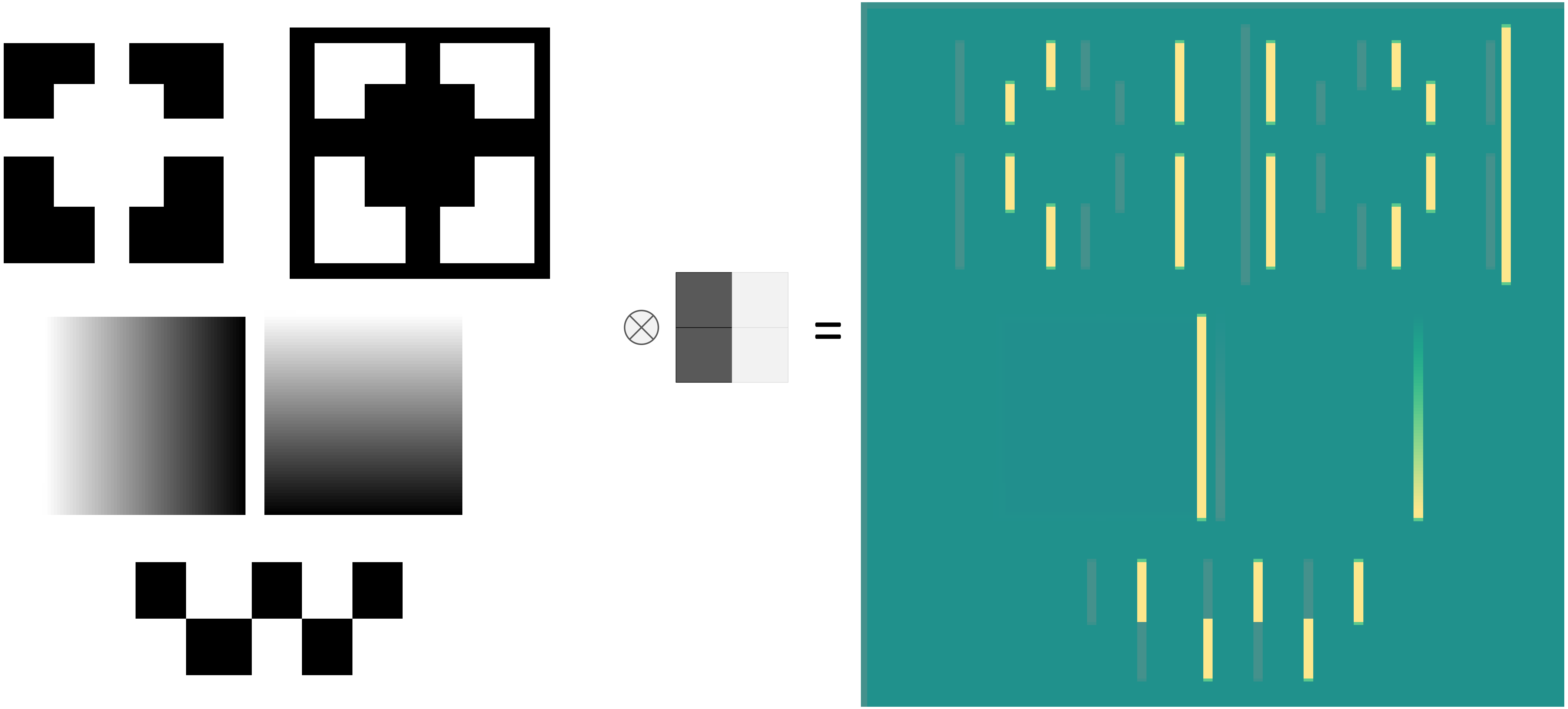
Basic operations

Convolution



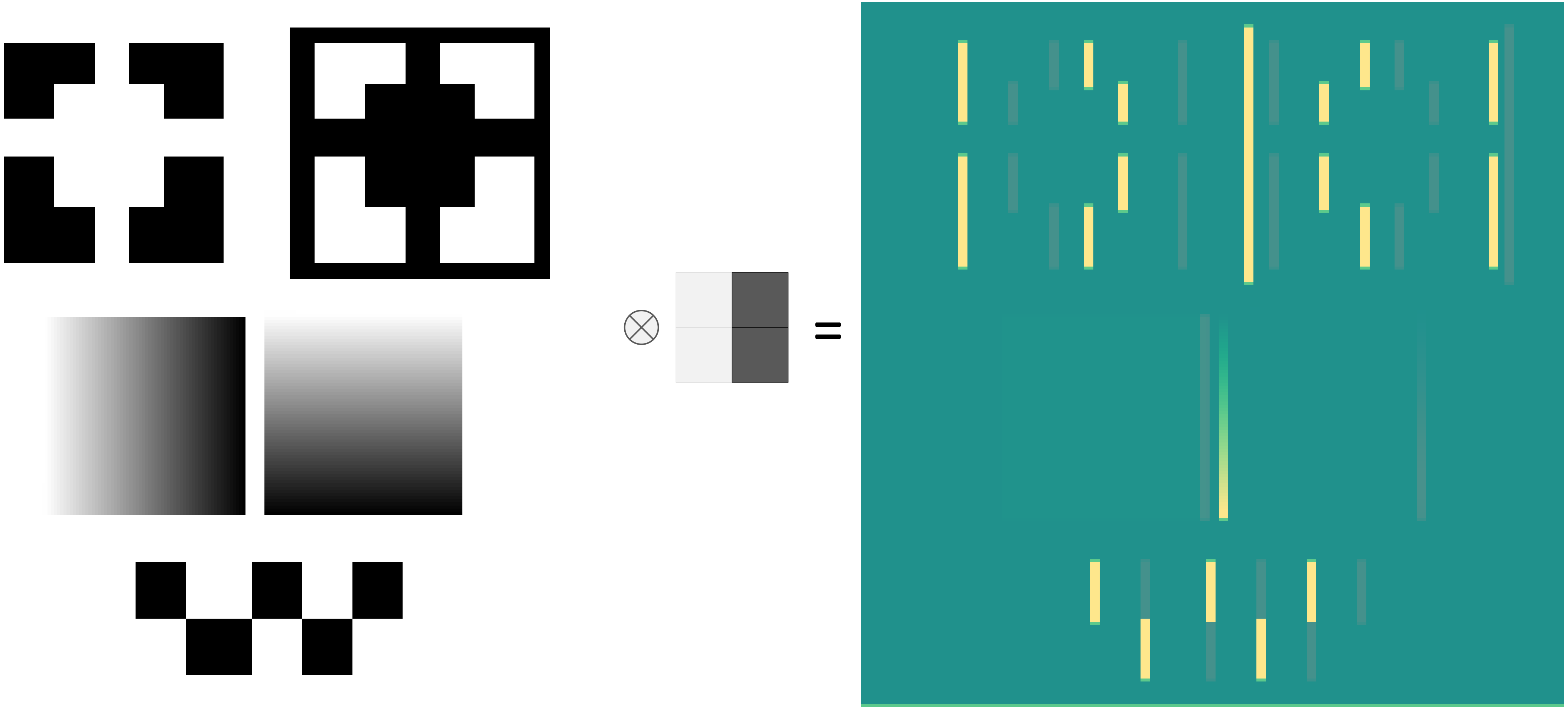
Basic operations

Convolution



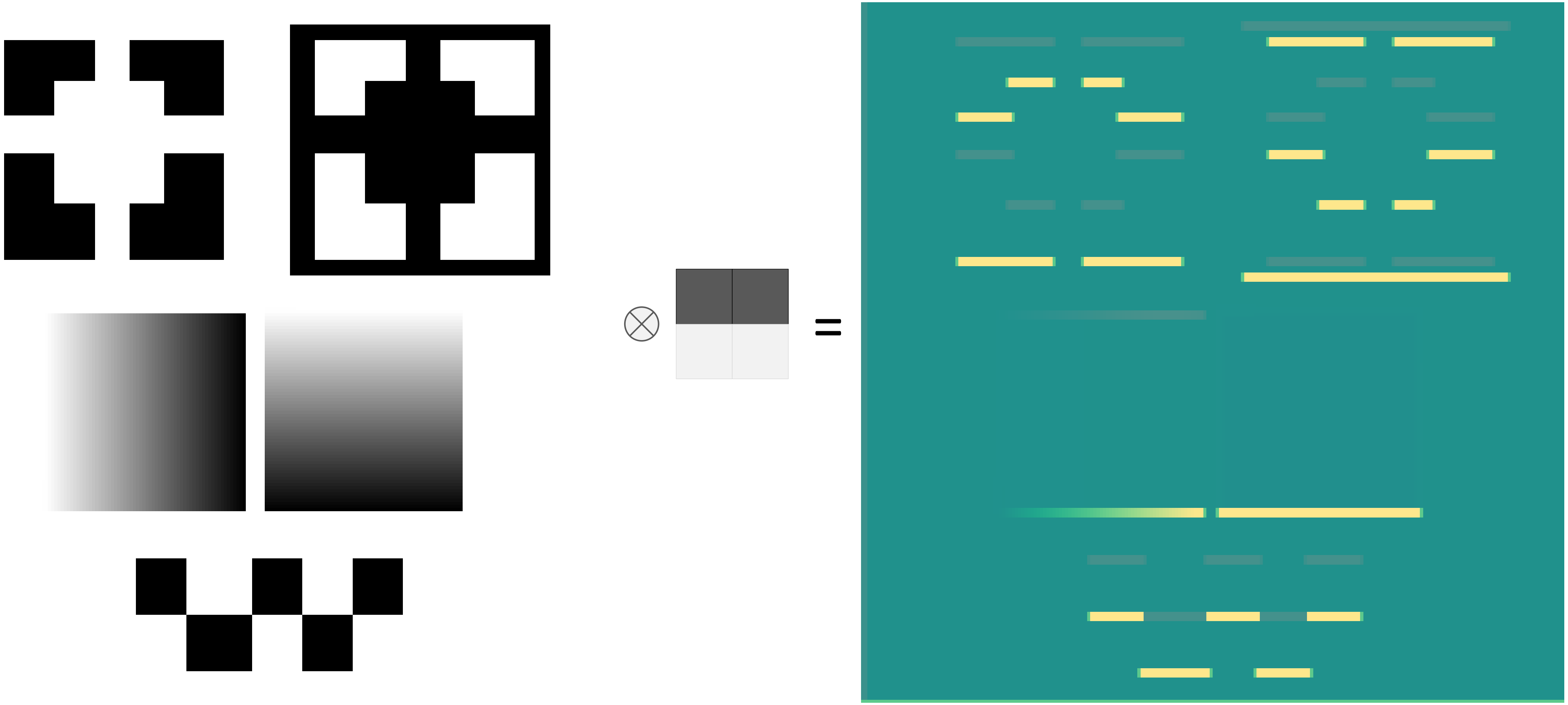
Basic operations

Convolution



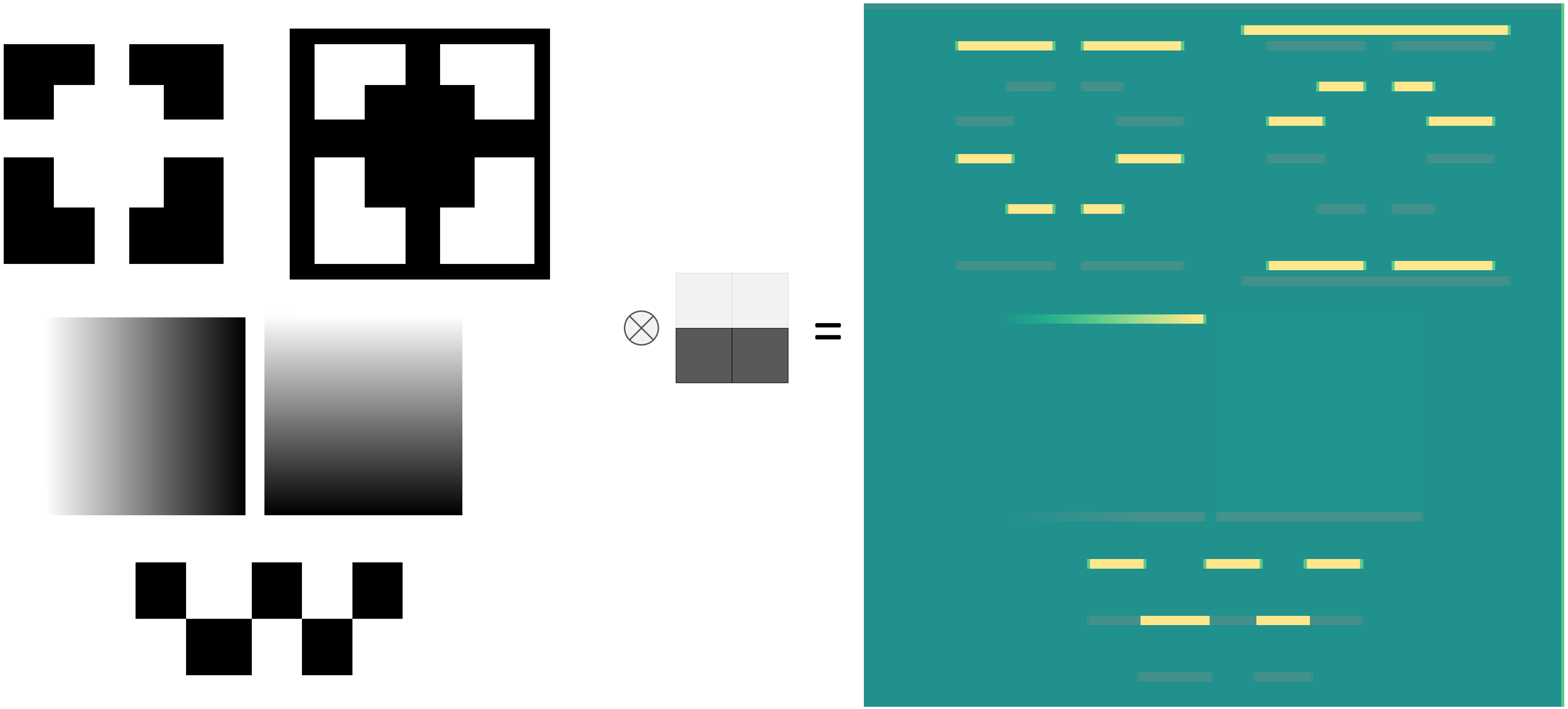
Basic operations

Convolution



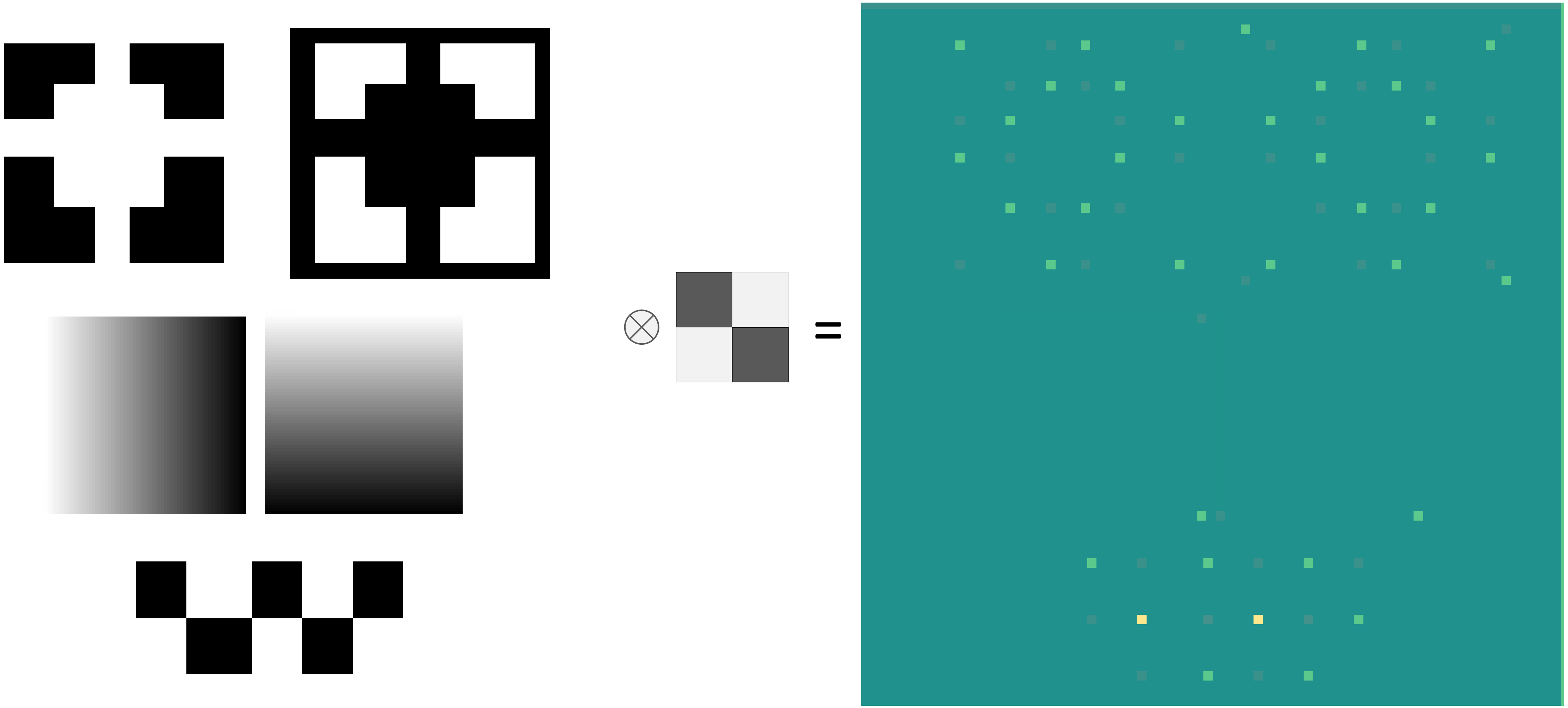
Basic operations

Convolution



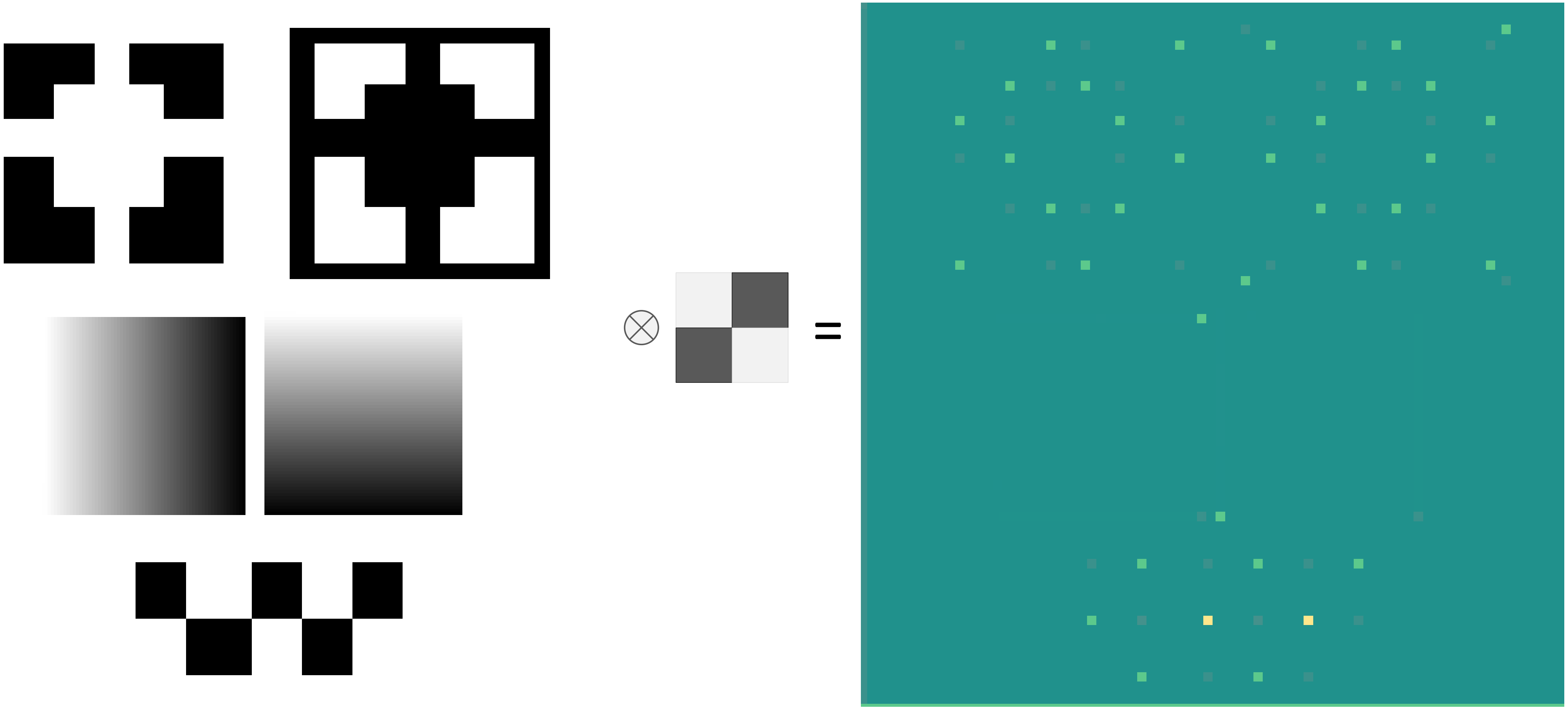
Basic operations

Convolution



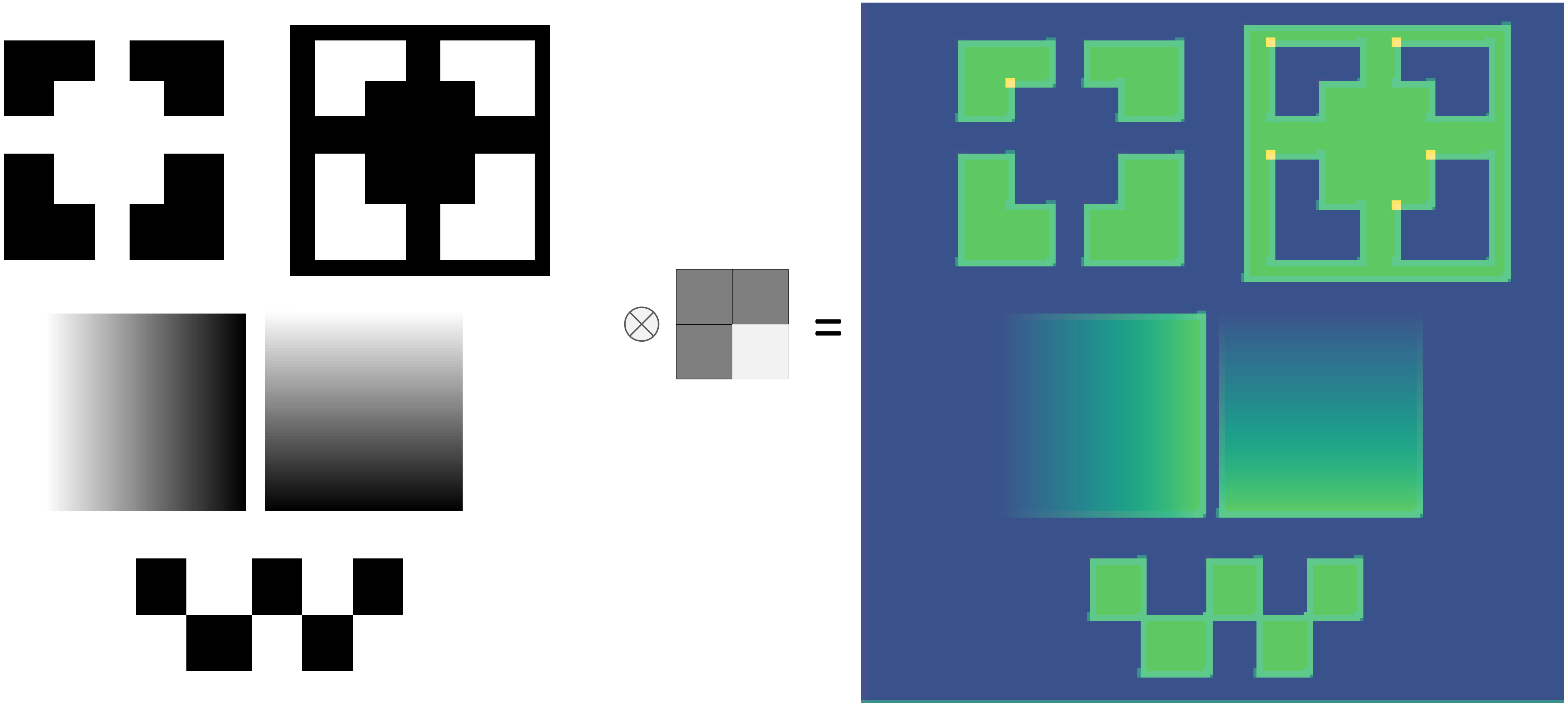
Basic operations

Convolution



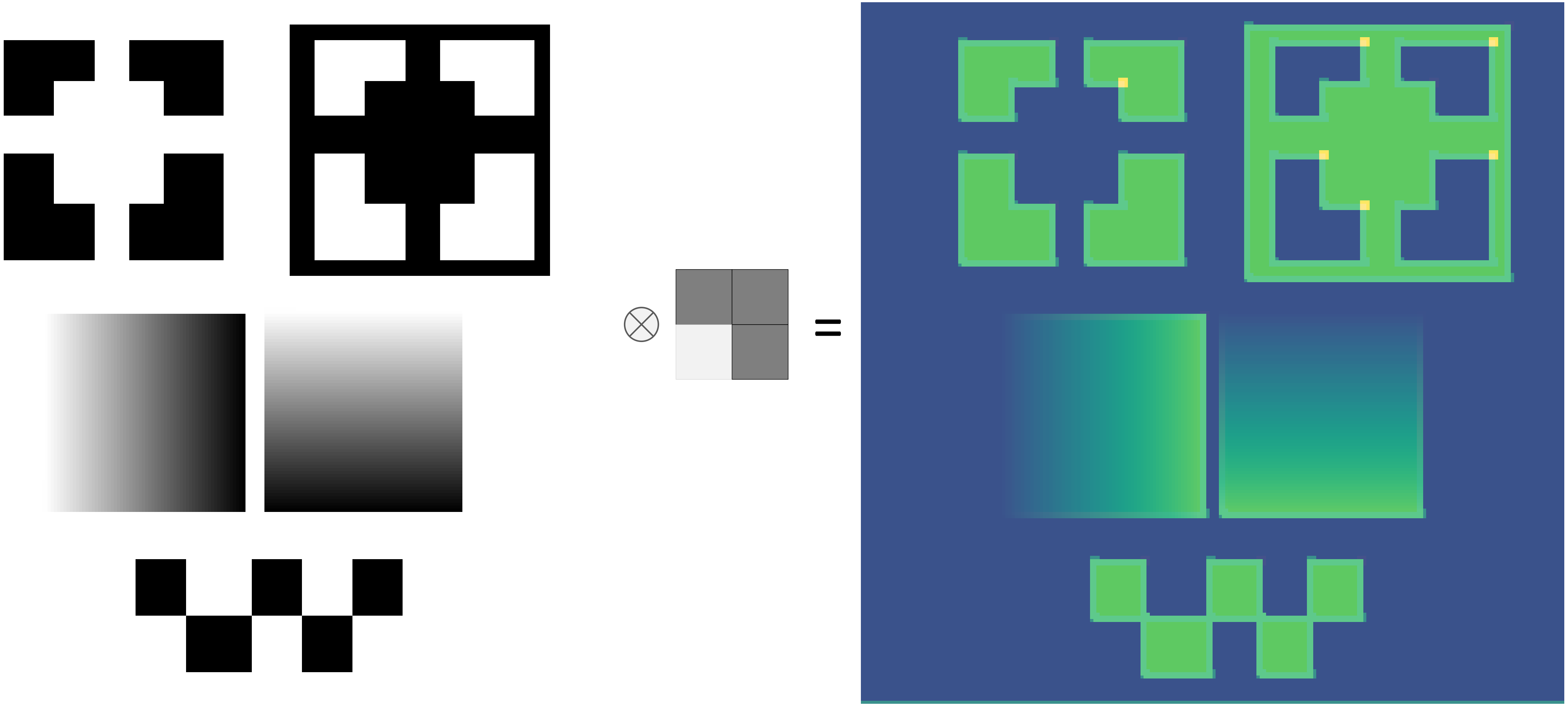
Basic operations

Convolution



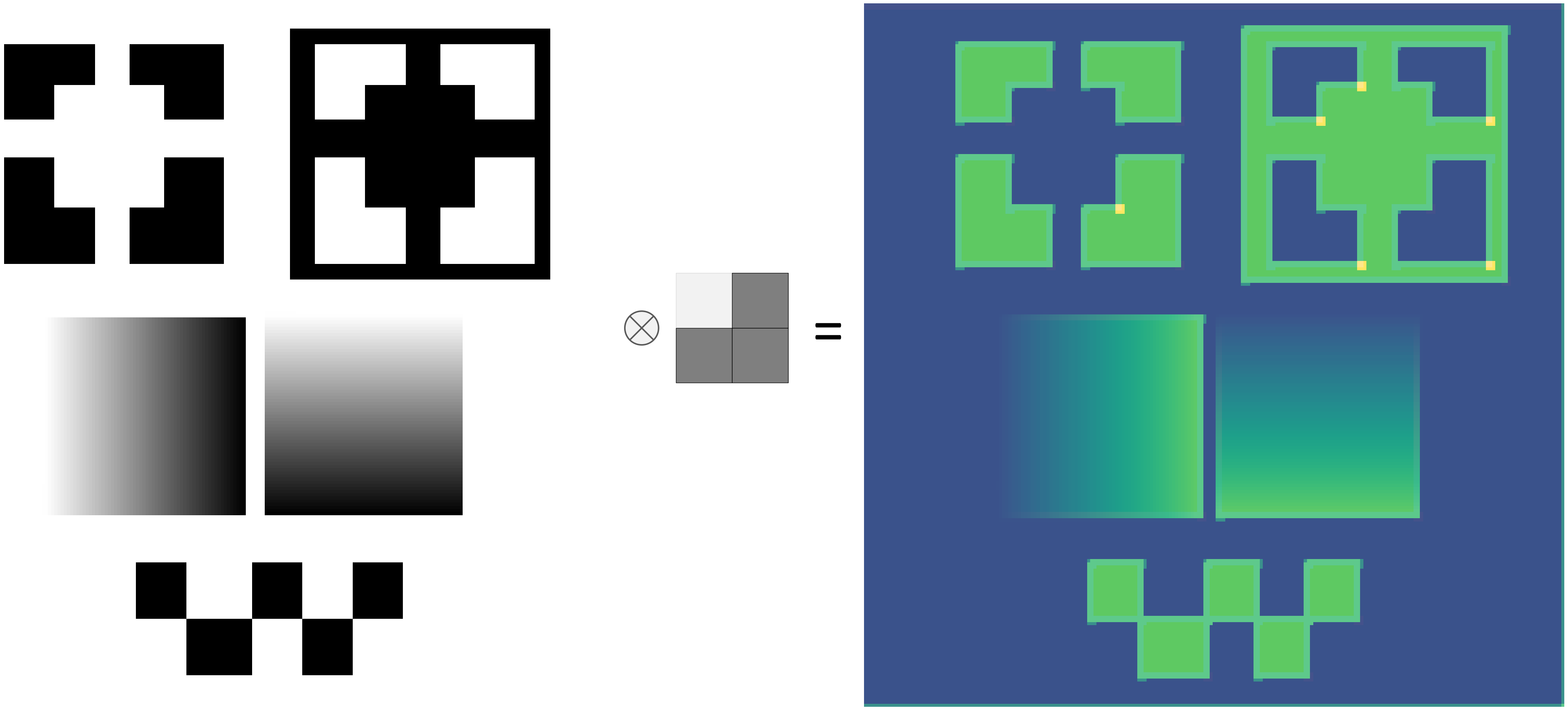
Basic operations

Convolution



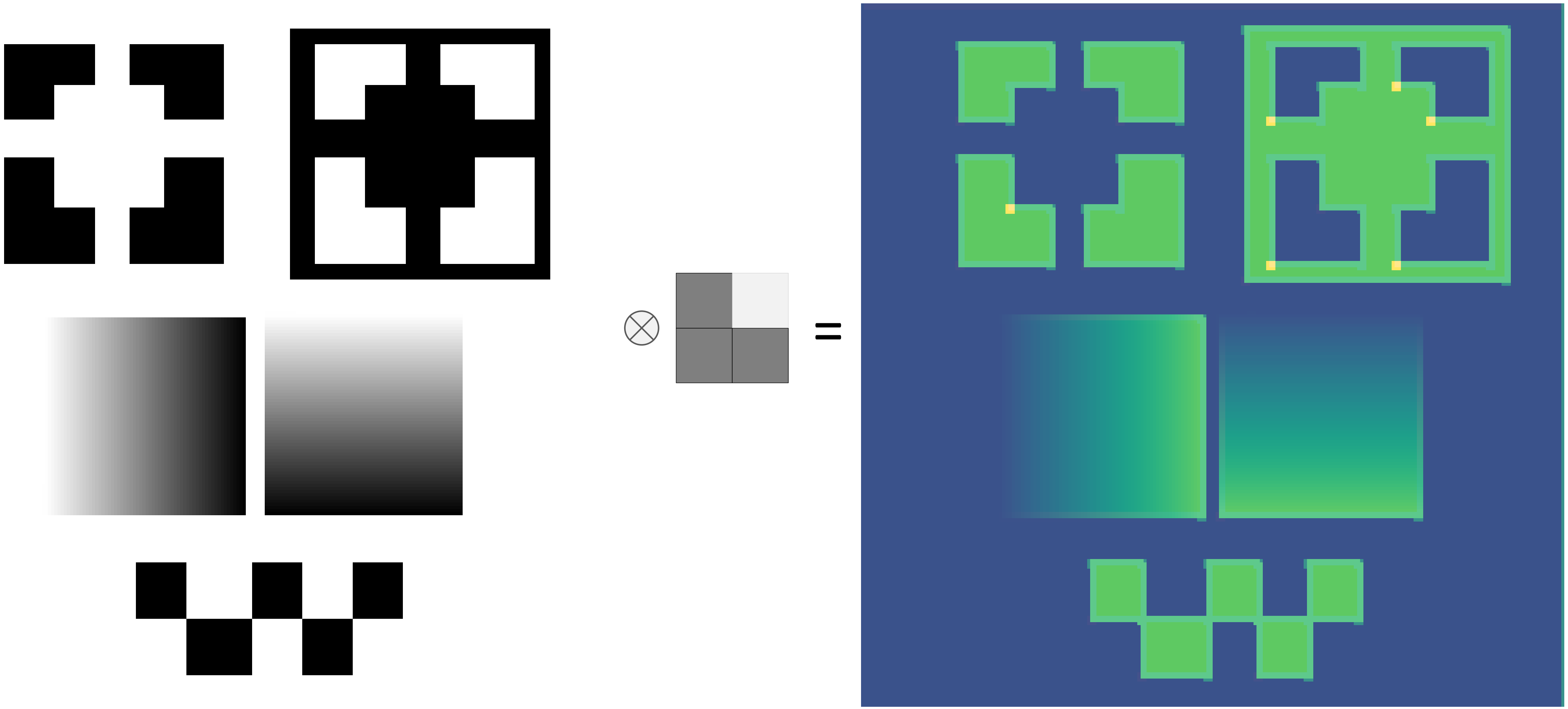
Basic operations

Convolution



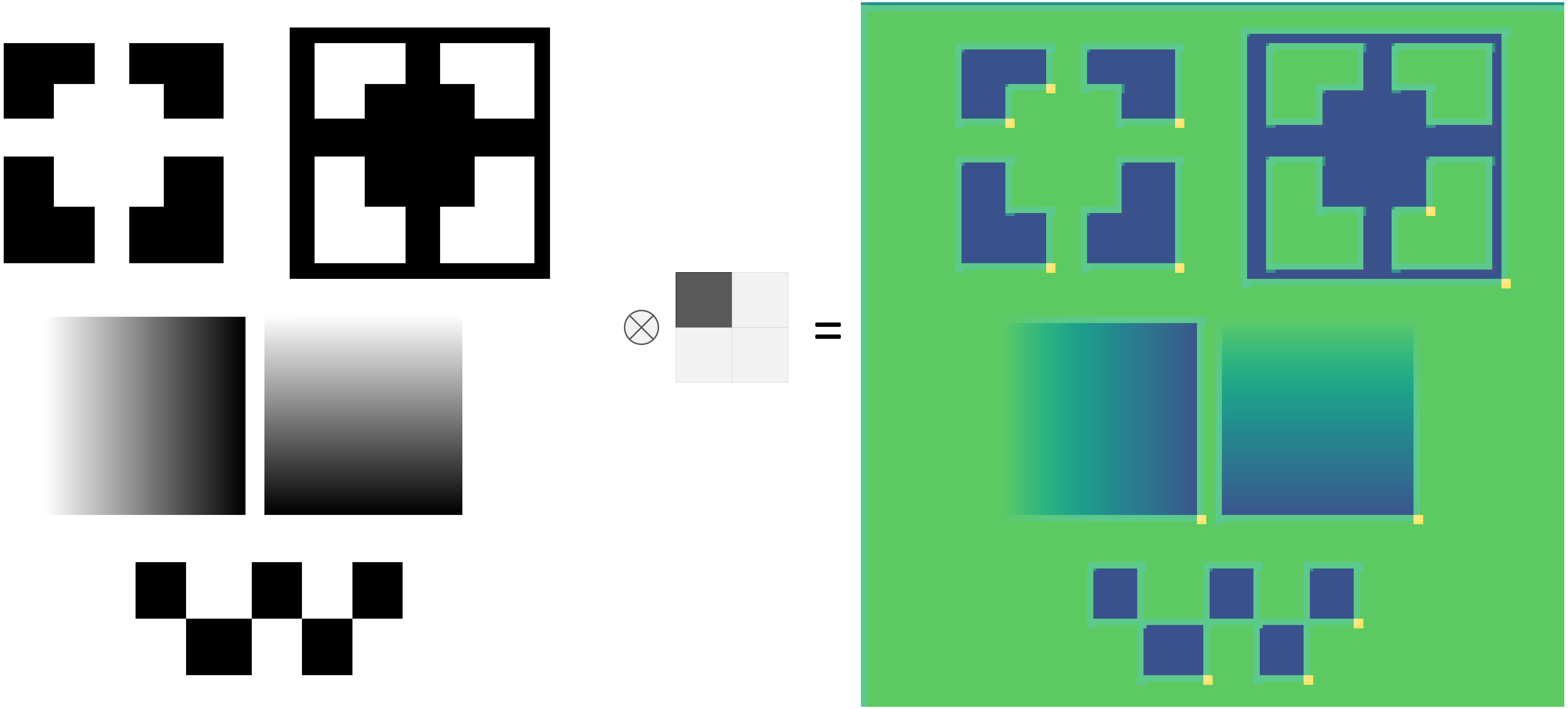
Basic operations

Convolution



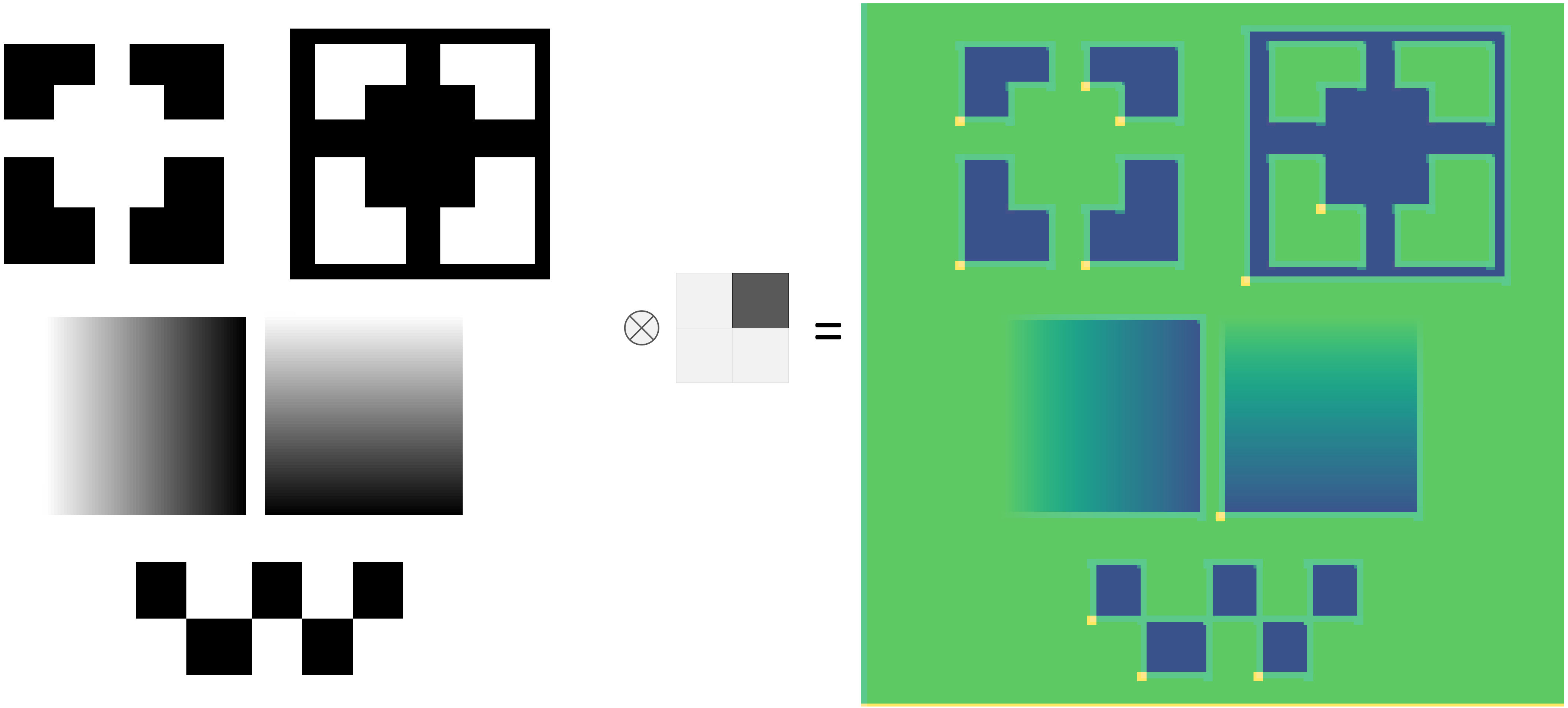
Basic operations

Convolution



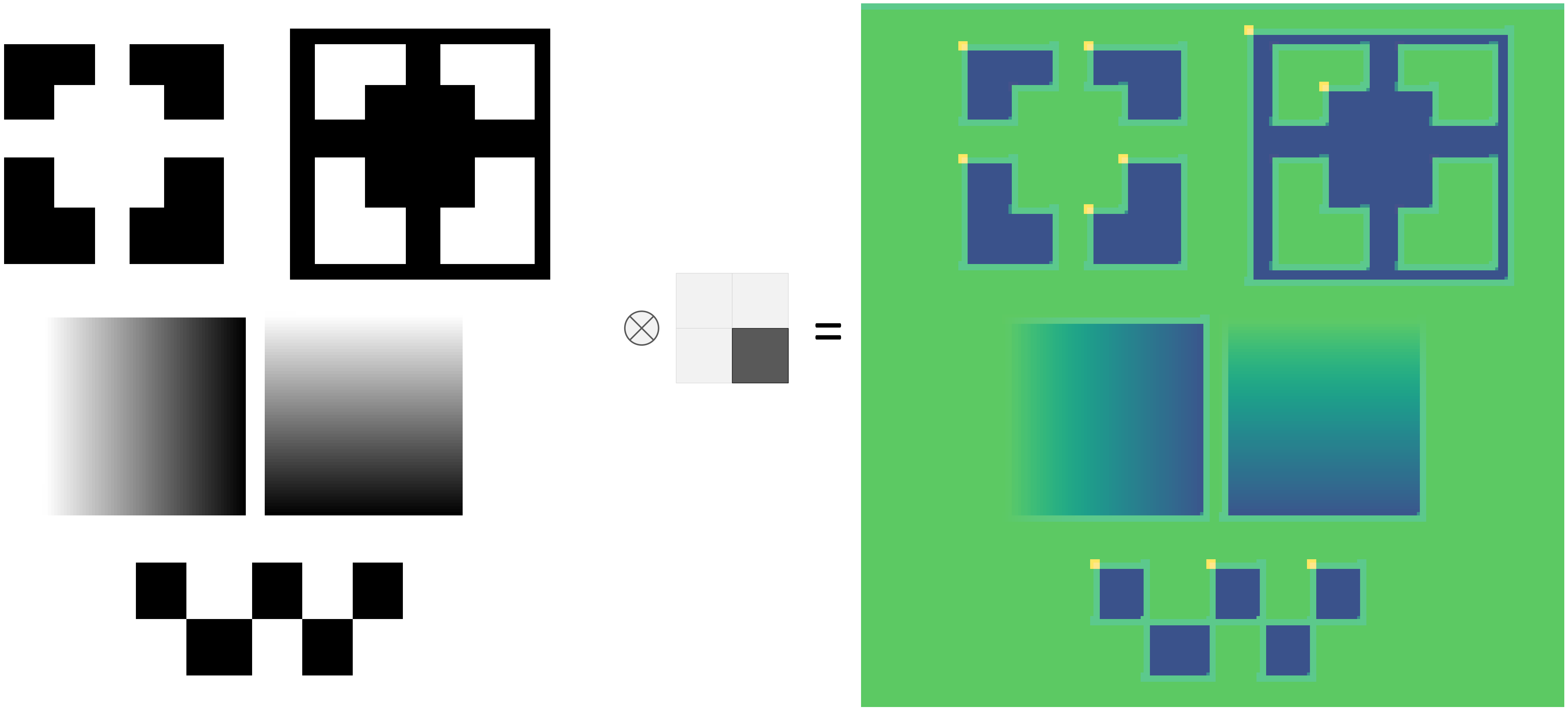
Basic operations

Convolution



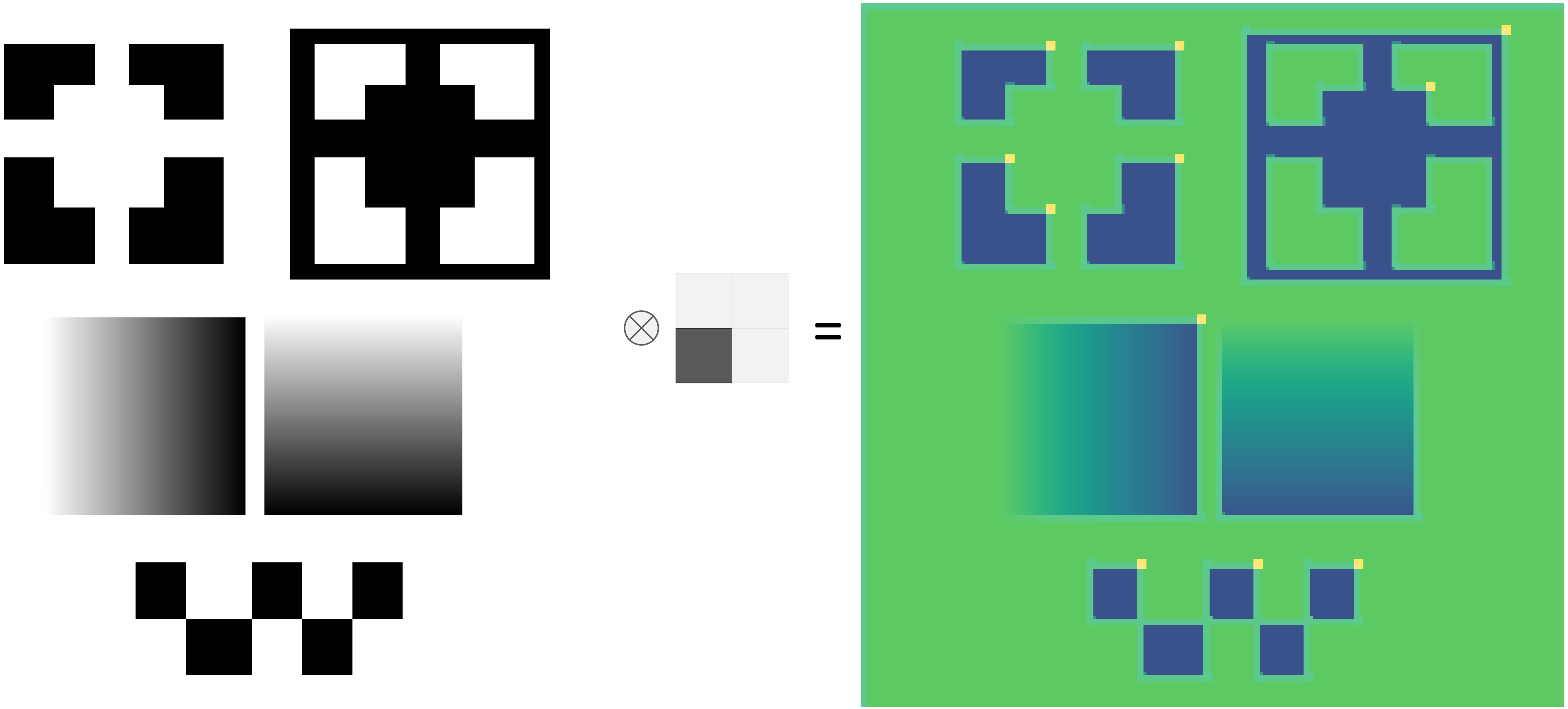
Basic operations

Convolution



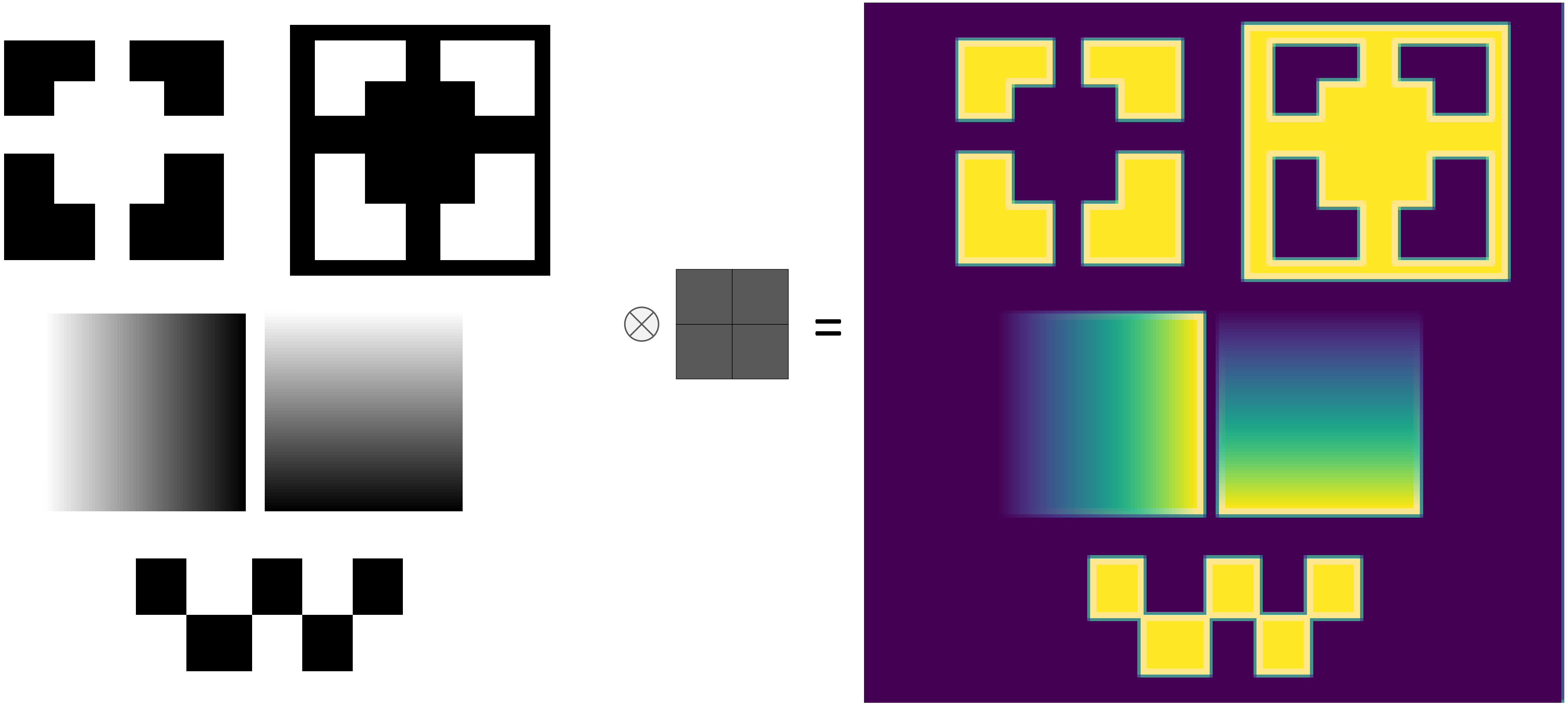
Basic operations

Convolution



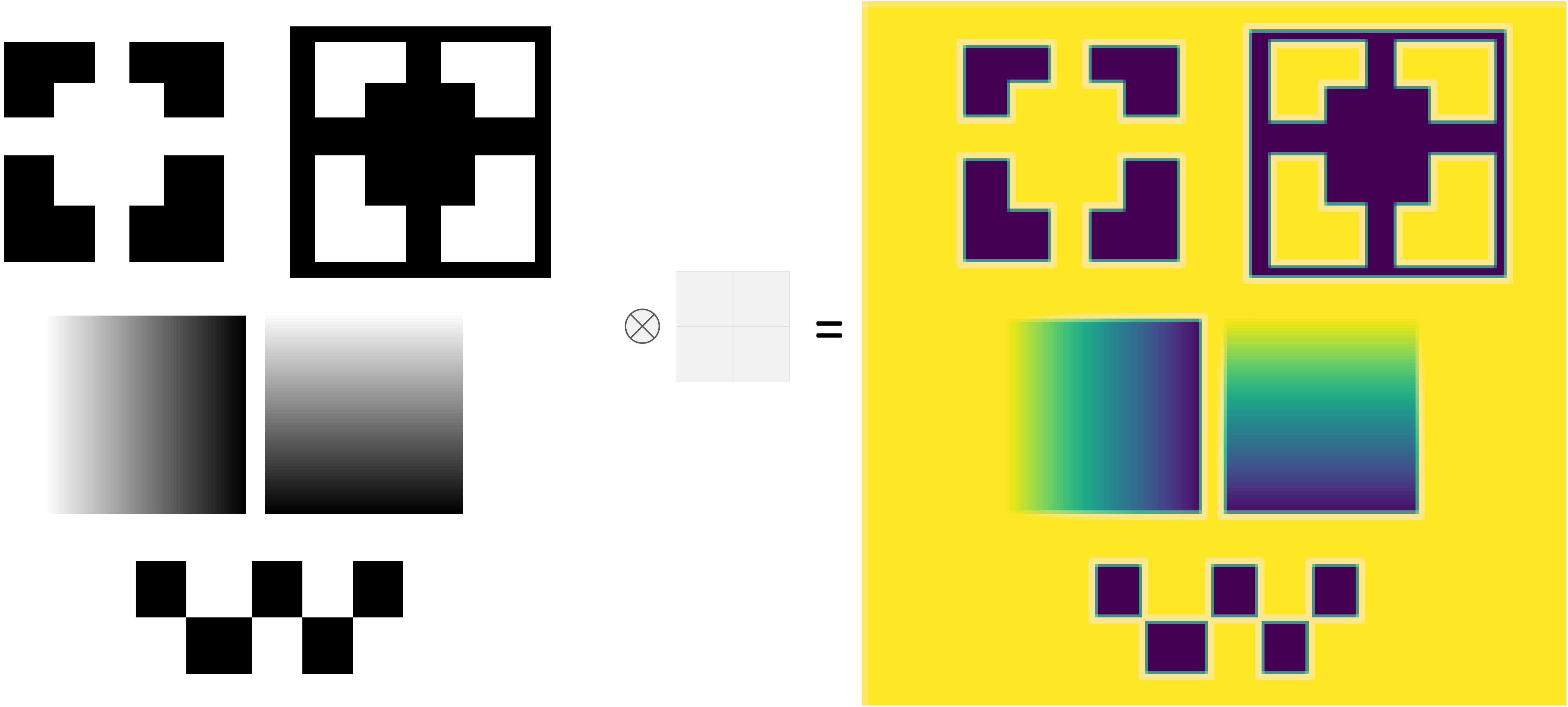
Basic operations

Convolution



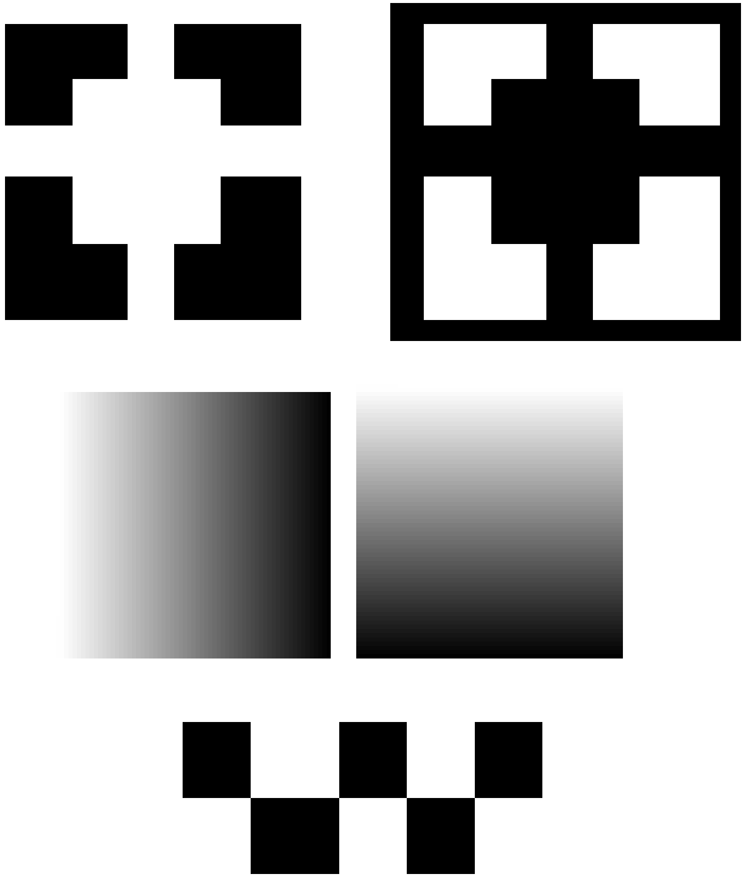
Basic operations

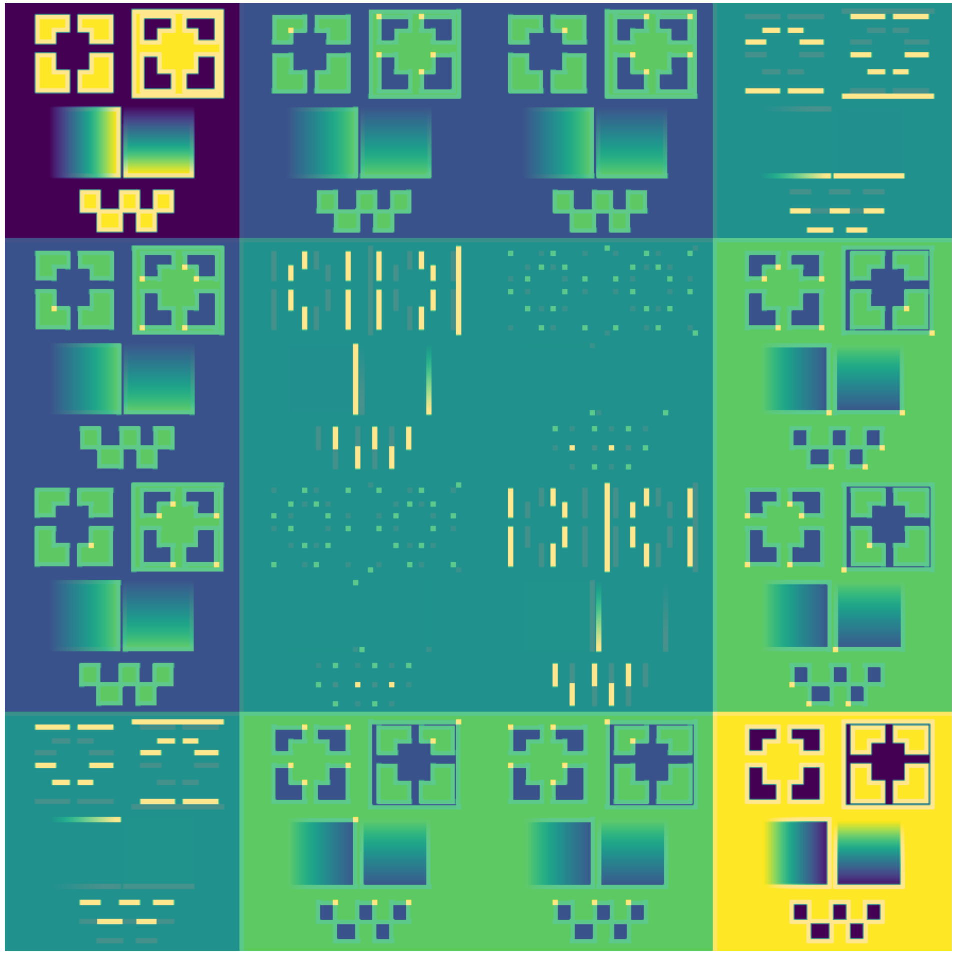
Convolution



Basic operations

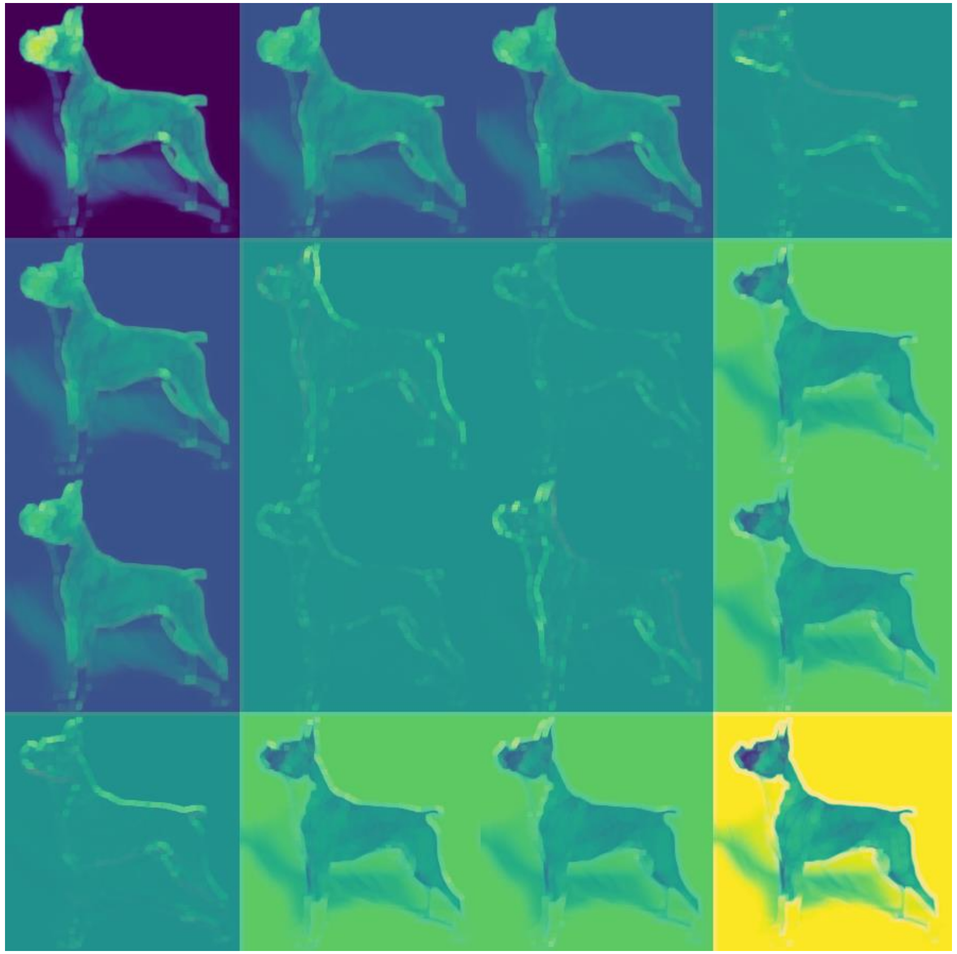
Convolution





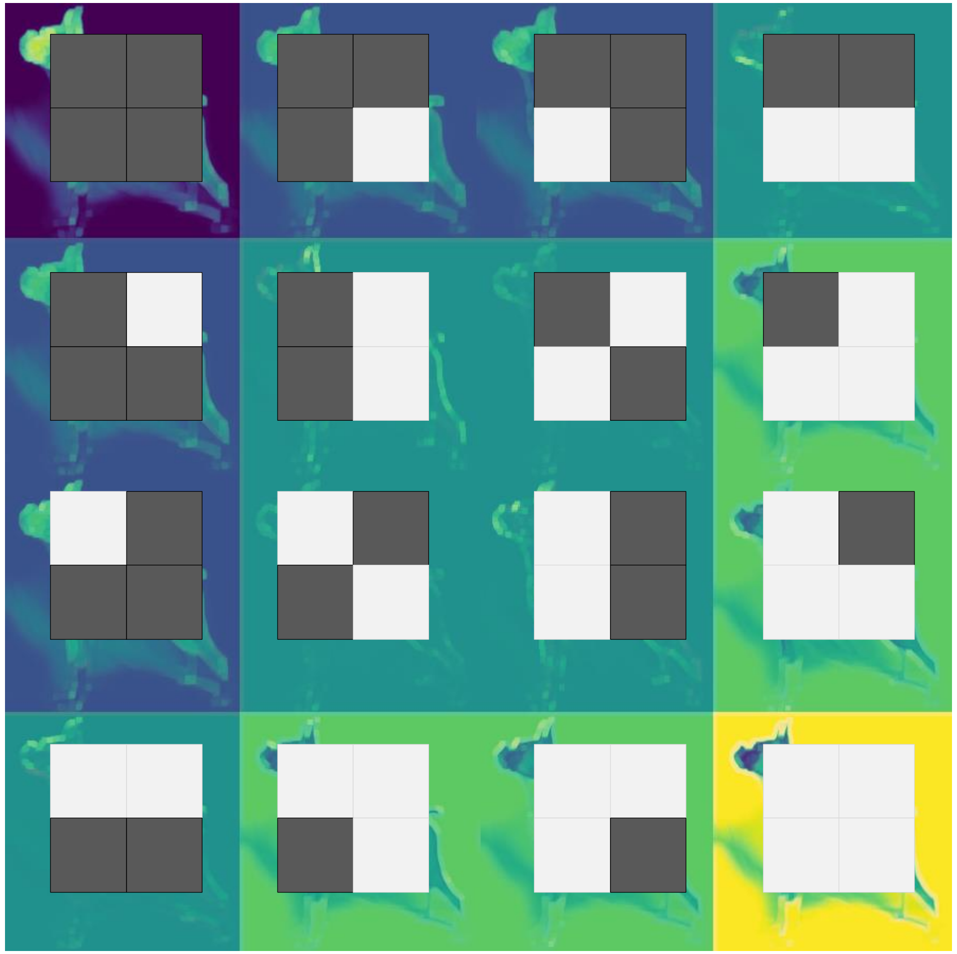
Basic operations

Convolution



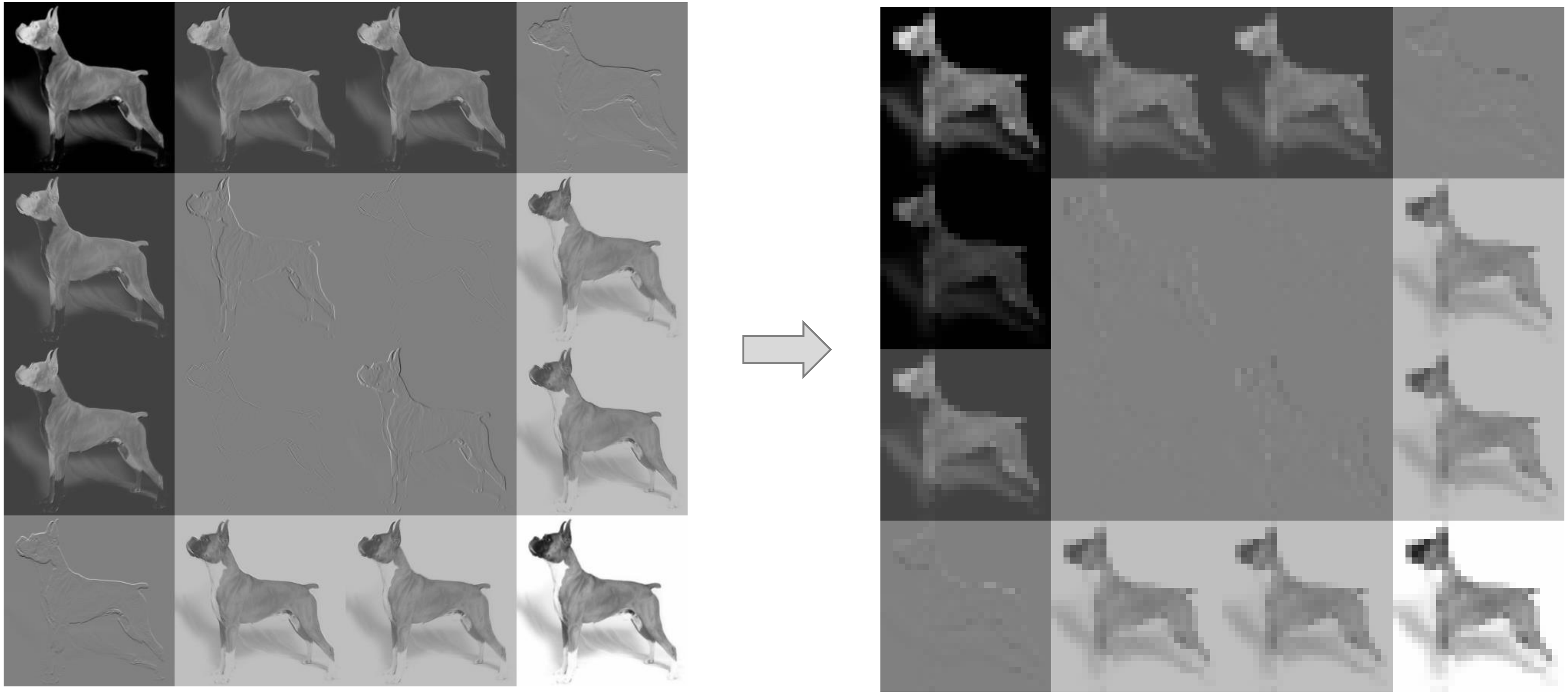
Basic operations

De-Convolution



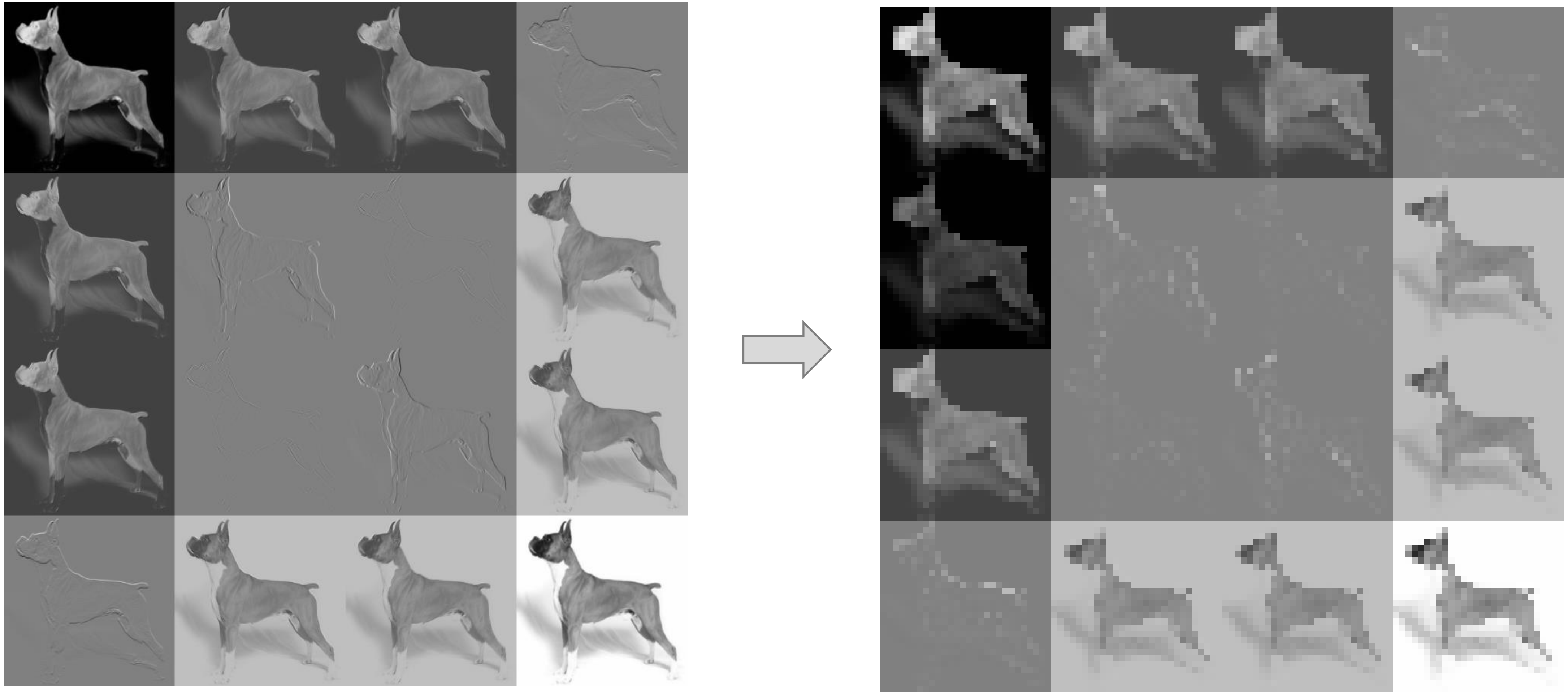
Basic operations

Pool: avg



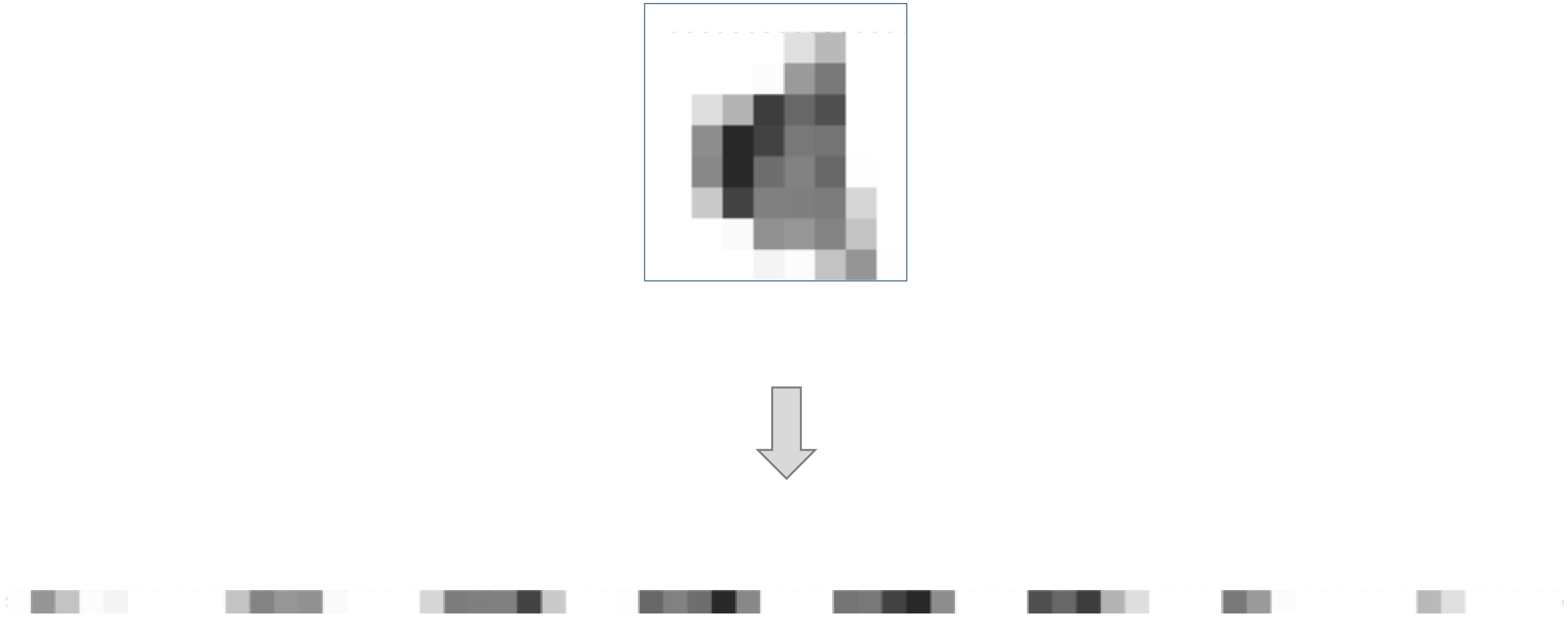
Basic operations

Pool: max



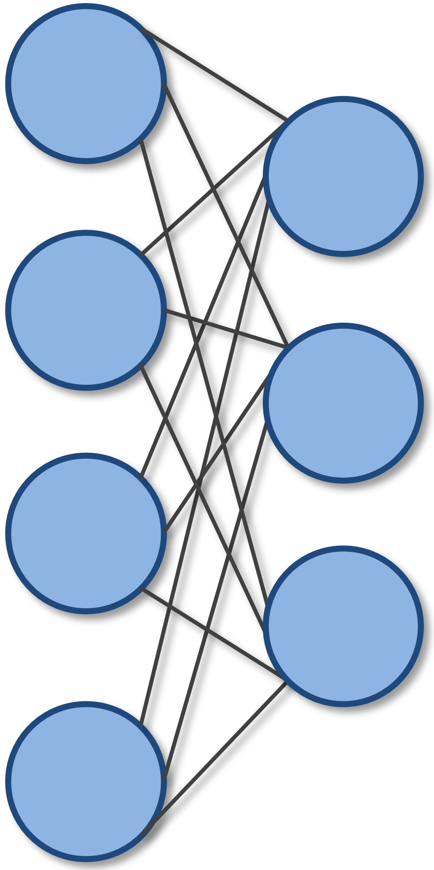
Basic operations

Flatten



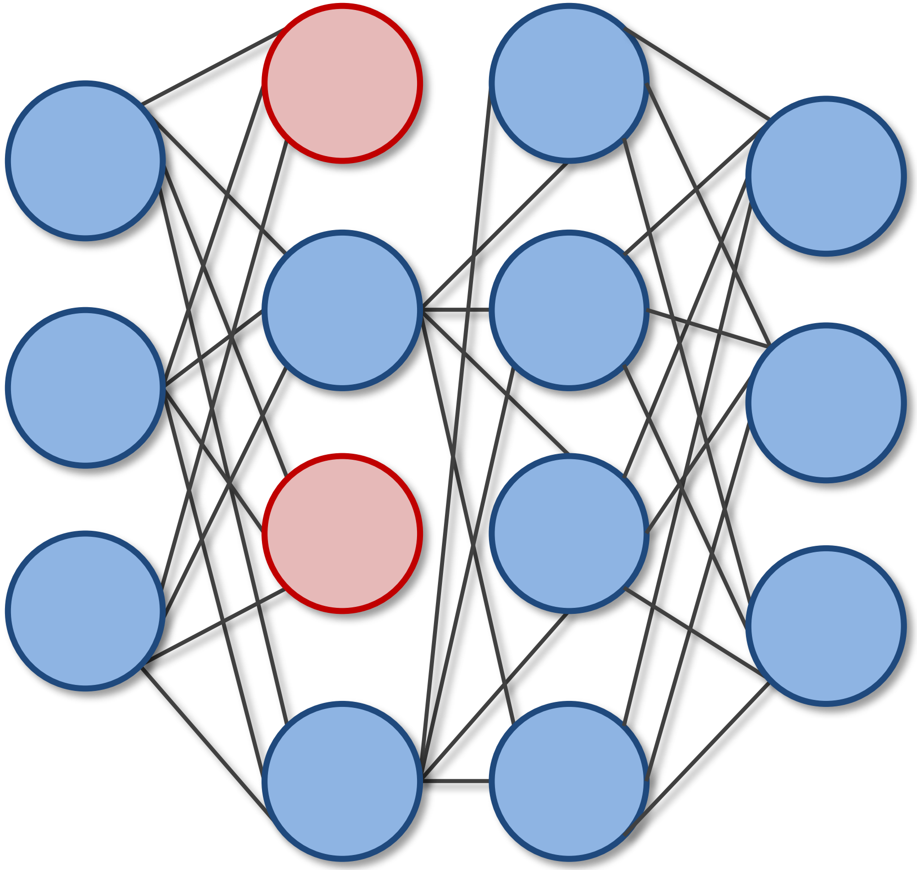
Basic operations

Fully Connected



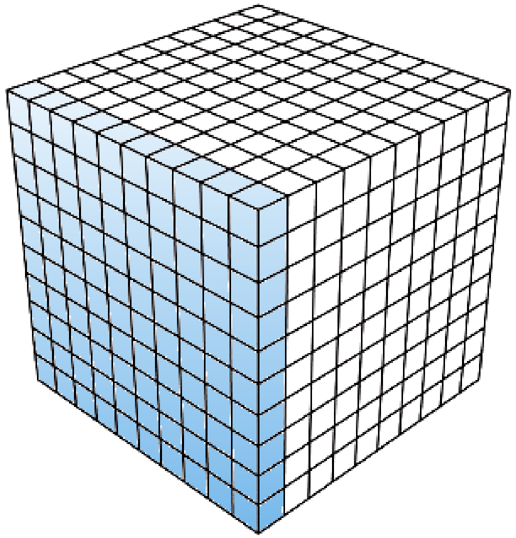
Basic operations

Dropout

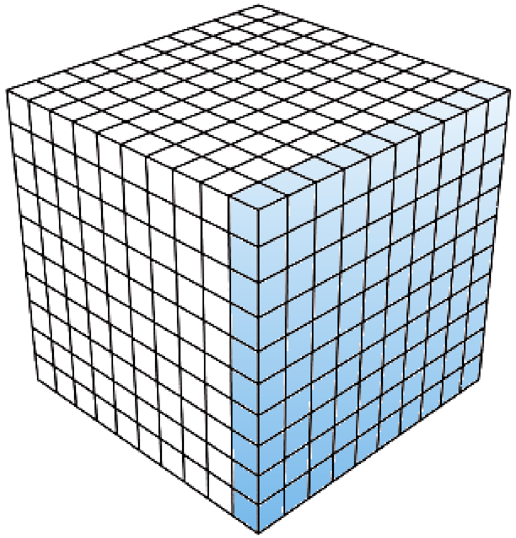


Basic operations

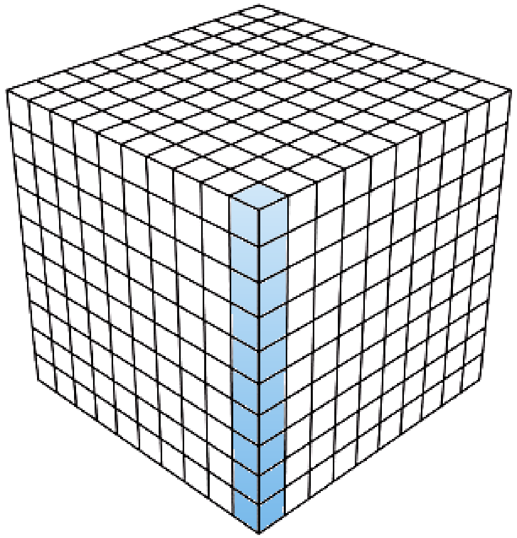
Normalization



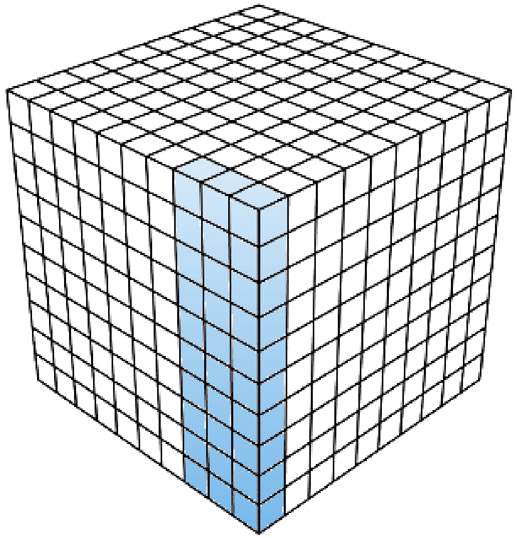
layer



batch



instance



group