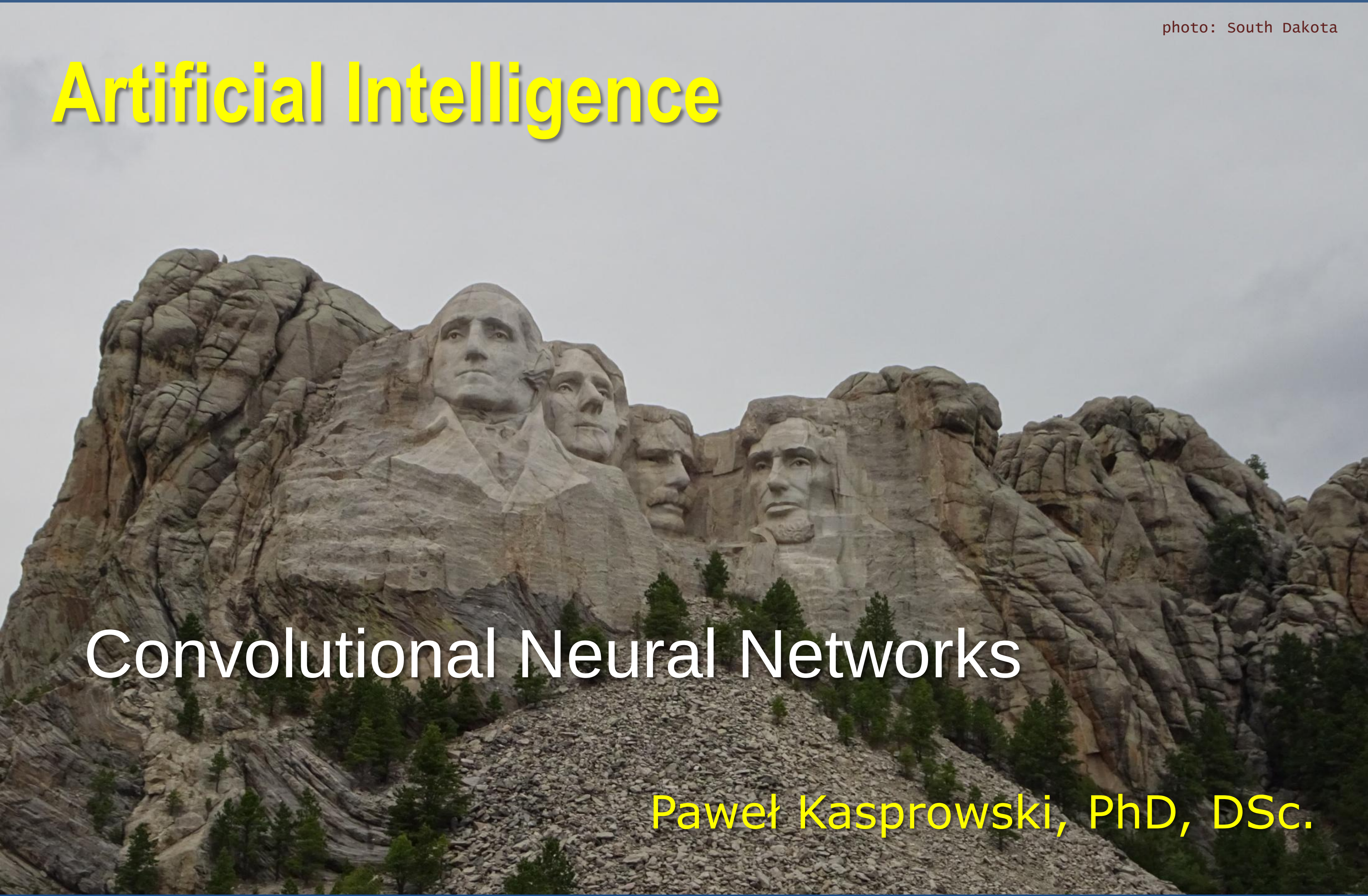


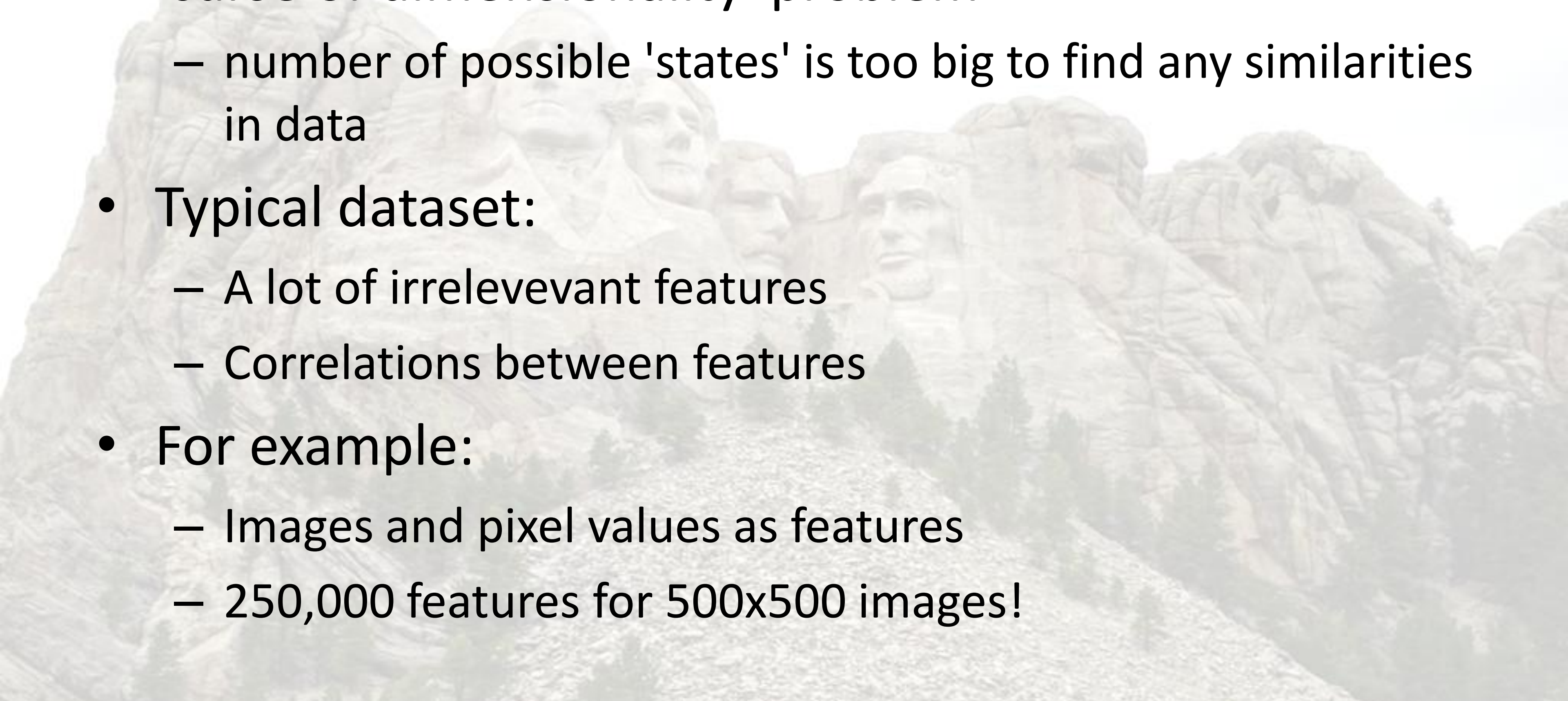
photo: South Dakota

Artificial Intelligence

Convolutional Neural Networks

Paweł Kasprowski, PhD, DSc.

Curse of dimensionality



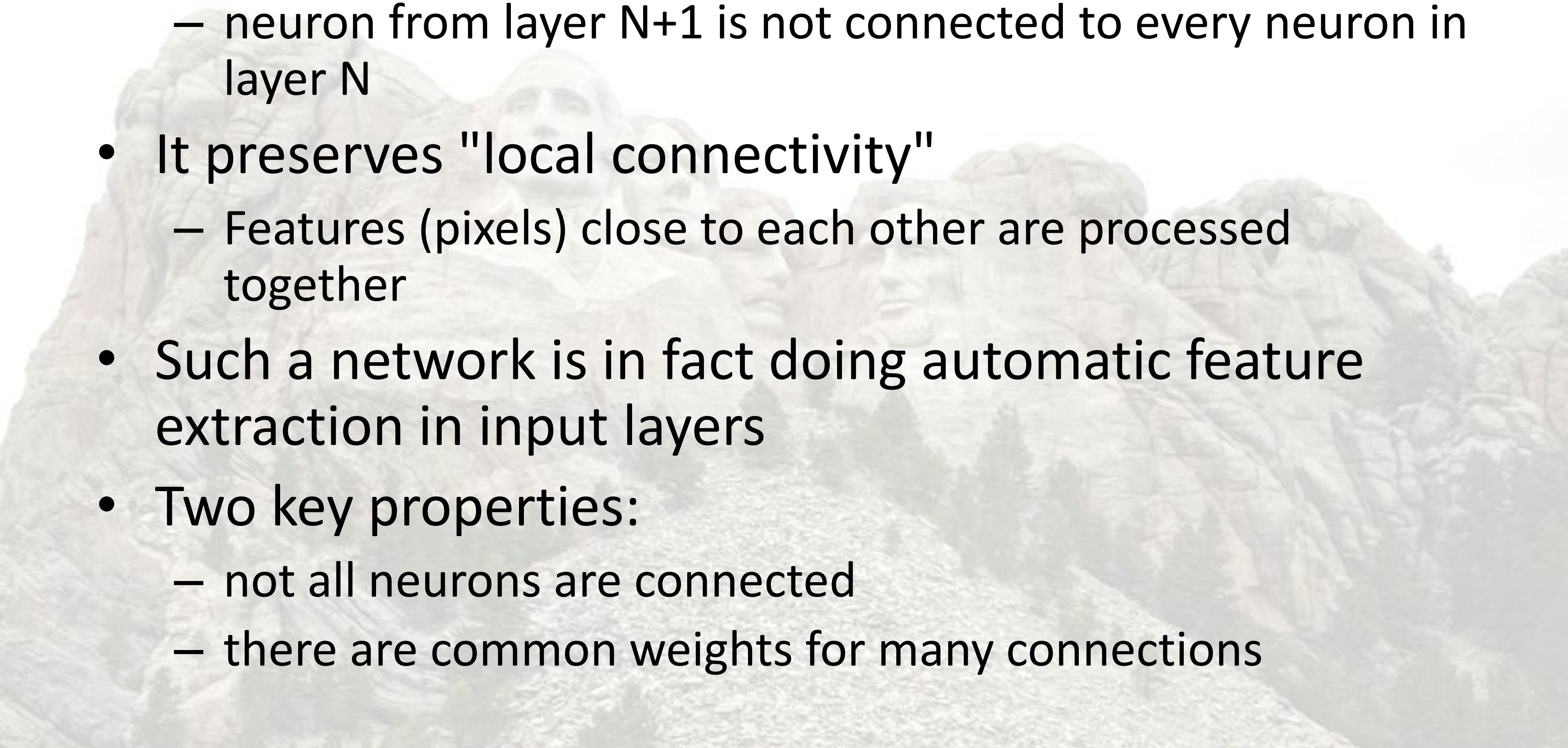
- If the number of features is too big we have the 'curse of dimensionality' problem
  - number of possible 'states' is too big to find any similarities in data
- Typical dataset:
  - A lot of irrelevevant features
  - Correlations between features
- For example:
  - Images and pixel values as features
  - 250,000 features for 500x500 images!

Feature extraction

- All previously mentioned algrorithms treat input as a set of independent features
  - preferably not correlated
- So the first and most important part of classification is **feature extraction** from real data
- For instance in image classification it could be [1]:
  - **Color** - Color Channel Statistics (Mean, Standard Deviation) and Color Histogram
  - **Shape** - Hu Moments, Zernike Moments
  - **Texture** - Haralick Texture, Local Binary Patterns (LBP)
  - **Others** - Histogram of Oriented Gradients (HOG), Threshold Adjancency Statistics (TAS)

[1] https://gogul09.github.io/software/image-classification-python

Convolutional Neural Networks



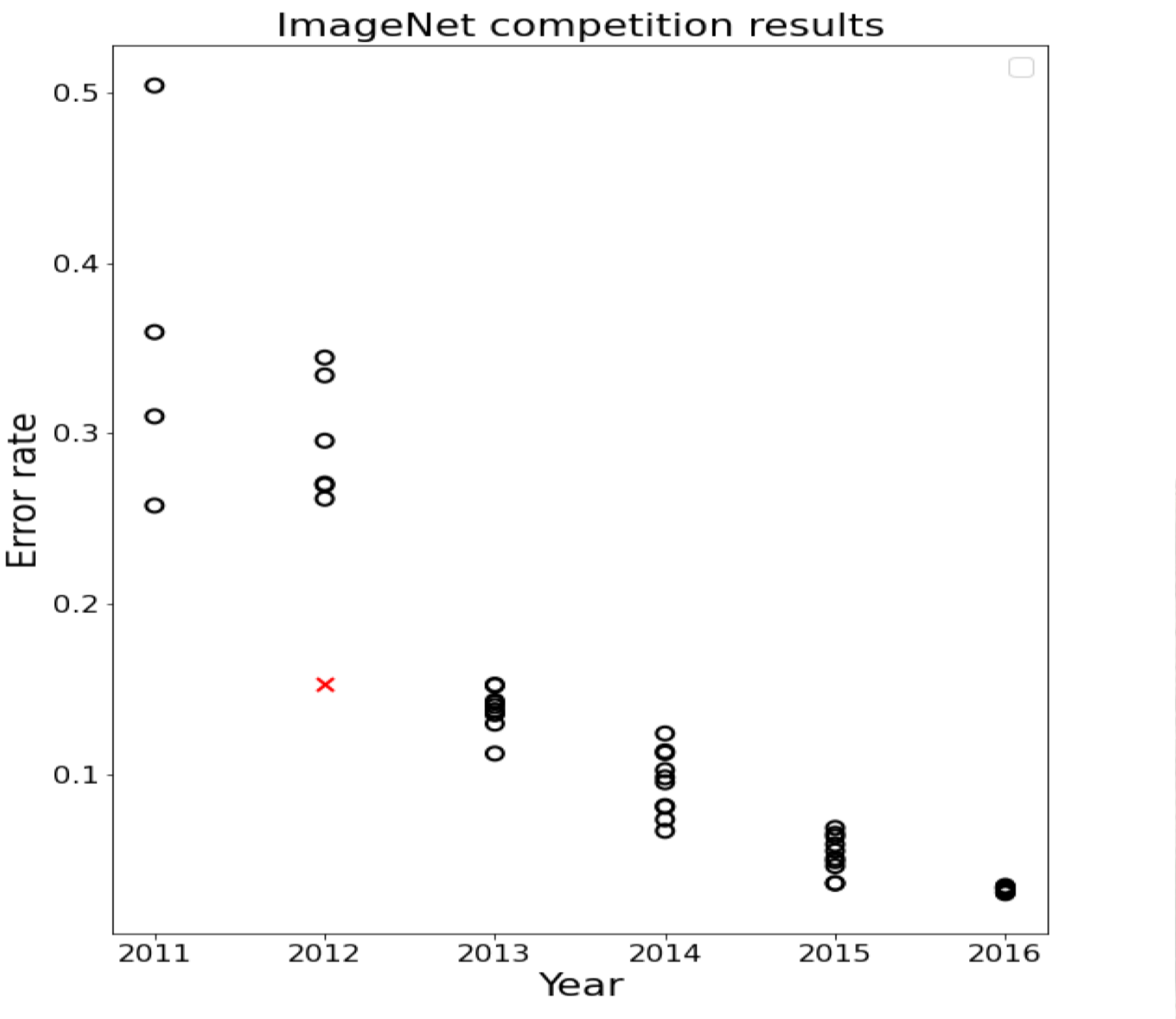
- Network that is not "dense"
  - neuron from layer N+1 is not connected to every neuron in layer N
- It preserves "local connectivity"
  - Features (pixels) close to each other are processed together
- Such a network is in fact doing automatic feature extraction in input layers
- Two key properties:
  - not all neurons are connected
  - there are common weights for many connections

Brief history

- 1982 – Kunihiko Fukushima, Neocognitron
  - pattern recognition
- 1989 – Yann LeCunn, LeNet-5
- 2010 – ImageNet Competition
  - 1000 classes, over million of images
- 2012 – Alex Krirzevsky, George Hinton: AlexNet
  - 15% error rate for ImageNet (runner-up: 26%)
  - 8 layers
- 2015 – Deep Residual Nets wins ImageNet
  - over 150 layers!

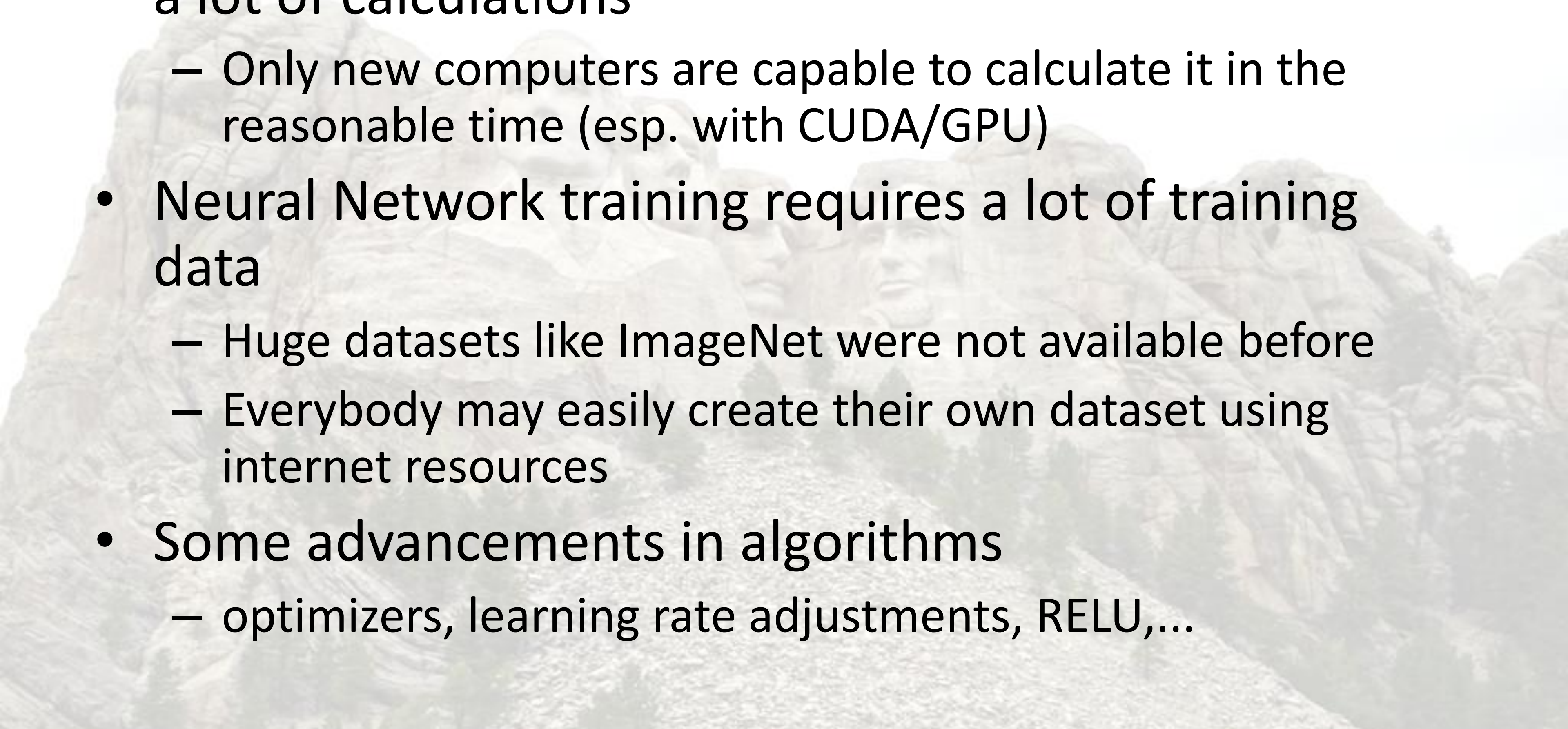
AlexNet

- Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Communications of the ACM* 60.6 (2017): 84-90.
  - **Cited by 73,253**



https://m.marefa.org/

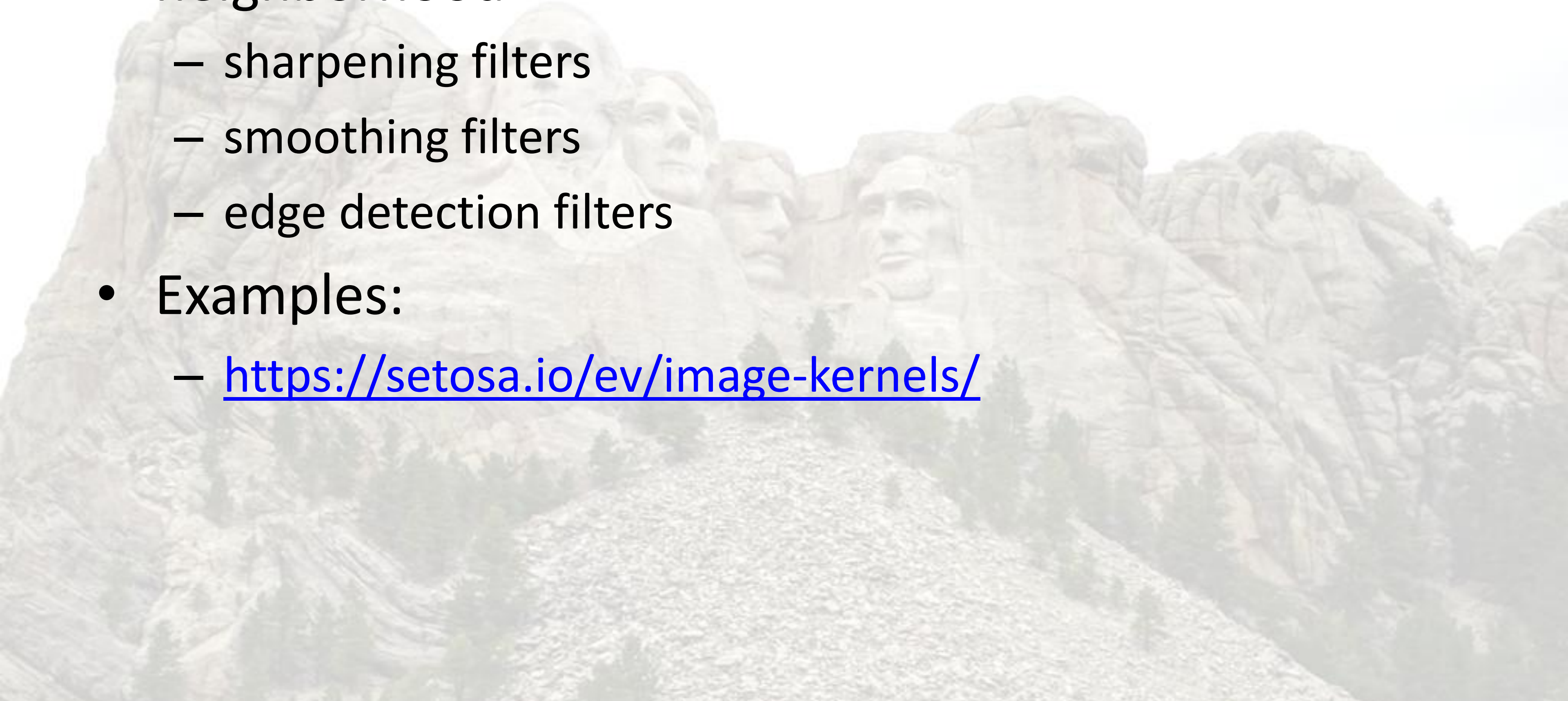
Why now?



- Neural Network training and predicting involves a lot of calculations
  - Only new computers are capable to calculate it in the reasonable time (esp. with CUDA/GPU)
- Neural Network training requires a lot of training data
  - Huge datasets like ImageNet were not available before
  - Everybody may easily create their own dataset using internet resources
- Some advancements in algorithms
  - optimizers, learning rate adjustments, RELU,...

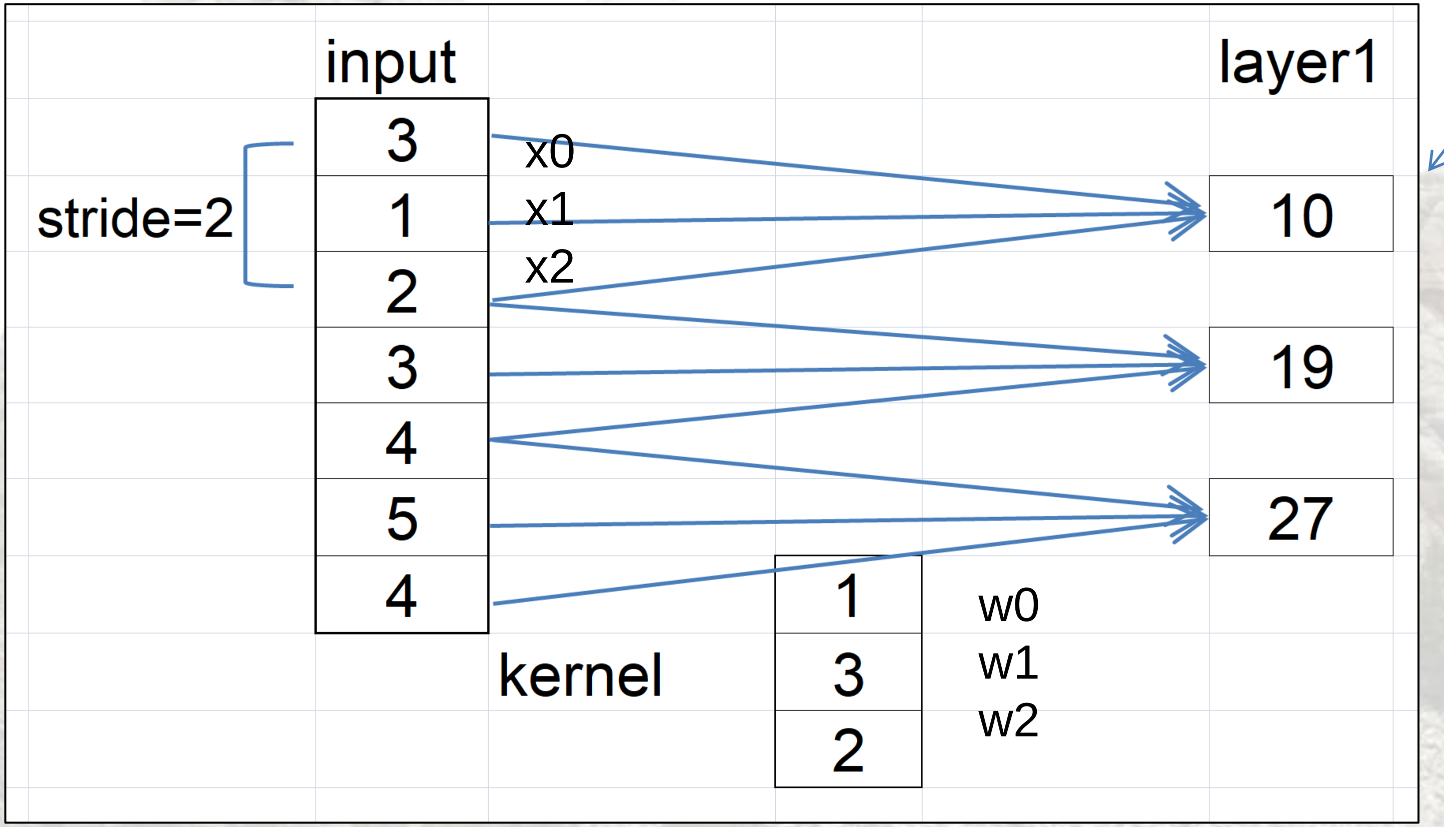
Convolutional Filter

- The filter (matrix NxM) applied to each pixel and its neighborhood
  - sharpening filters
  - smoothing filters
  - edge detection filters
- Examples:
  - https://setosa.io/ev/image-kernels/

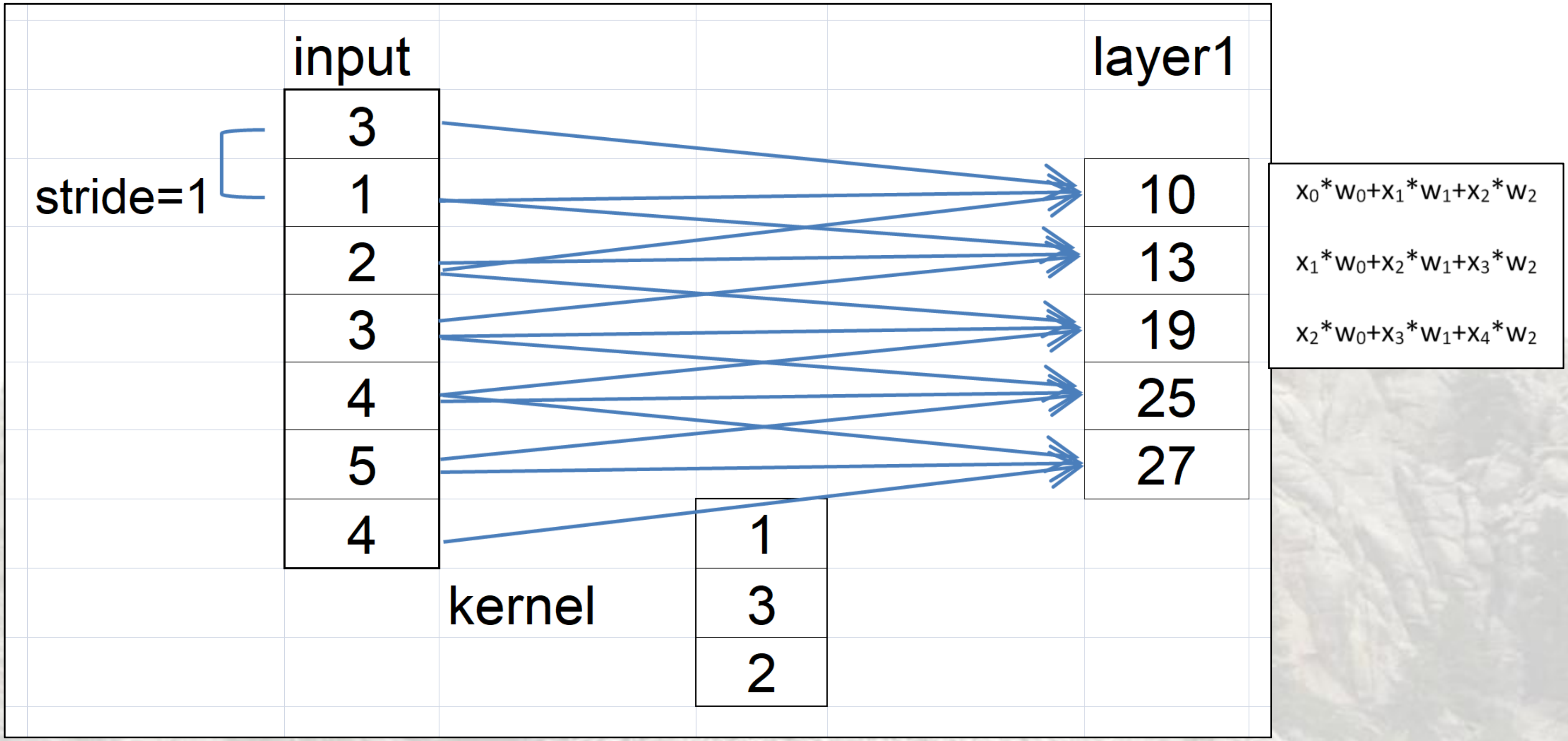


CNN - 1D example

$x_0 * w_0 + x_1 * w_1 + x_2 * w_2$



CNN - stride=1

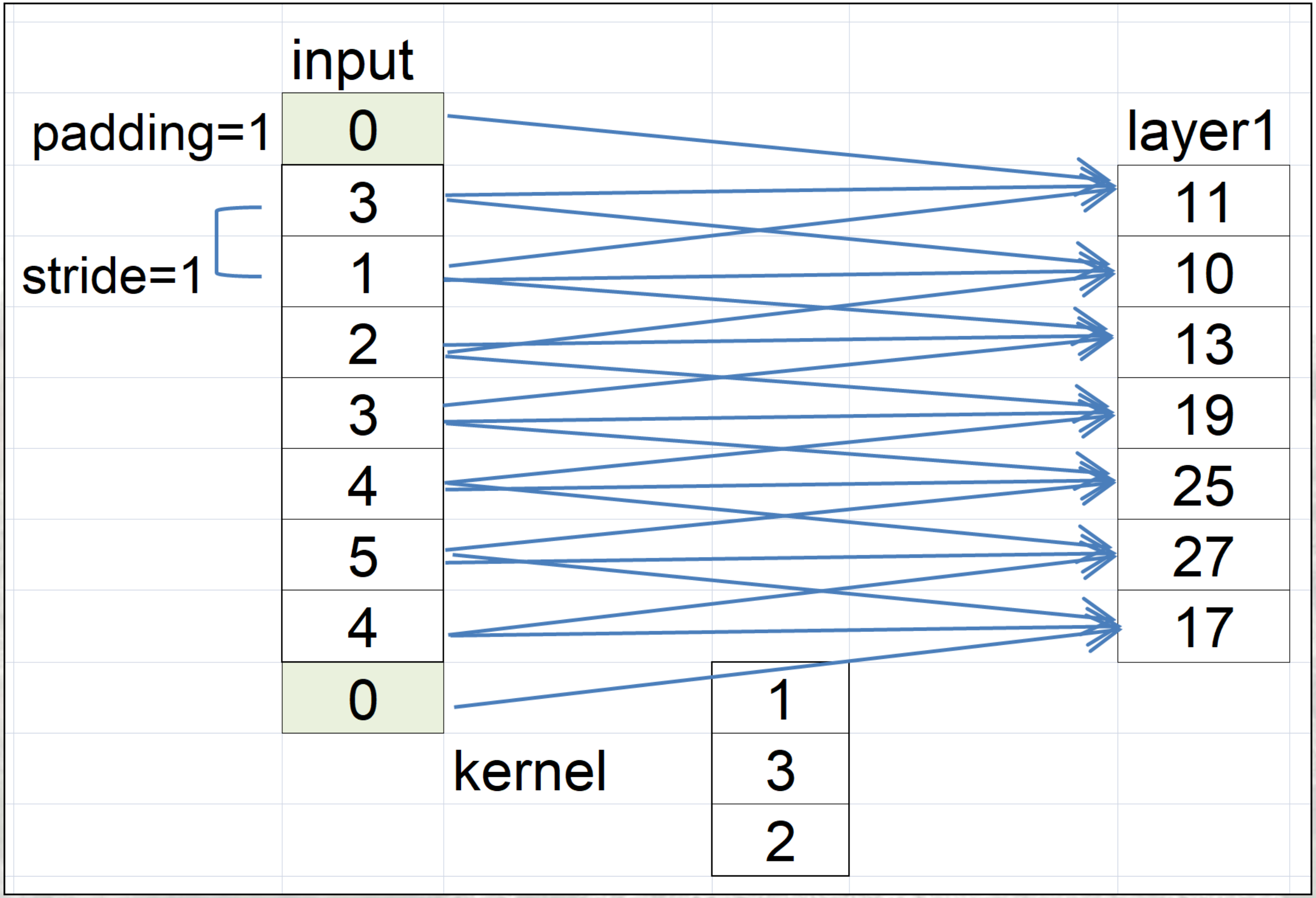


$x_0*w_0+x_1*w_1+x_2*w_2$

$x_1*w_0+x_2*w_1+x_3*w_2$

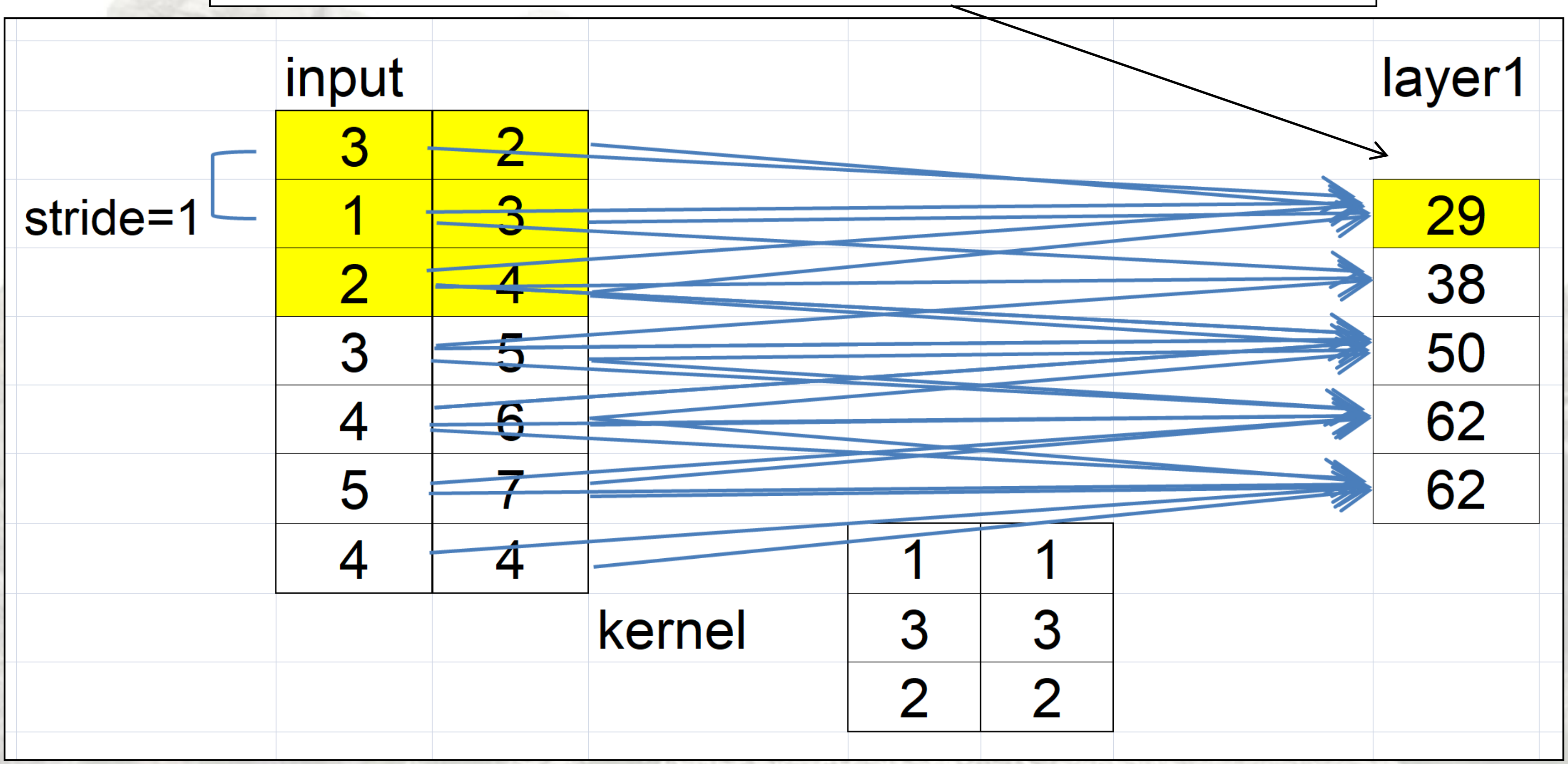
$x_2*w_0+x_3*w_1+x_4*w_2$

CNN - stride=1, padding=1



CNN - 2D

$$x_{00}{}^{*}w_{00}+x_{01}{}^{*}w_{01}+x_{02}{}^{*}w_{02}+x_{10}{}^{*}w_{10}+x_{11}{}^{*}w_{11}+x_{12}{}^{*}w_{12}$$



input

3	2
1	3
2	4
3	5
4	6
5	7
4	4

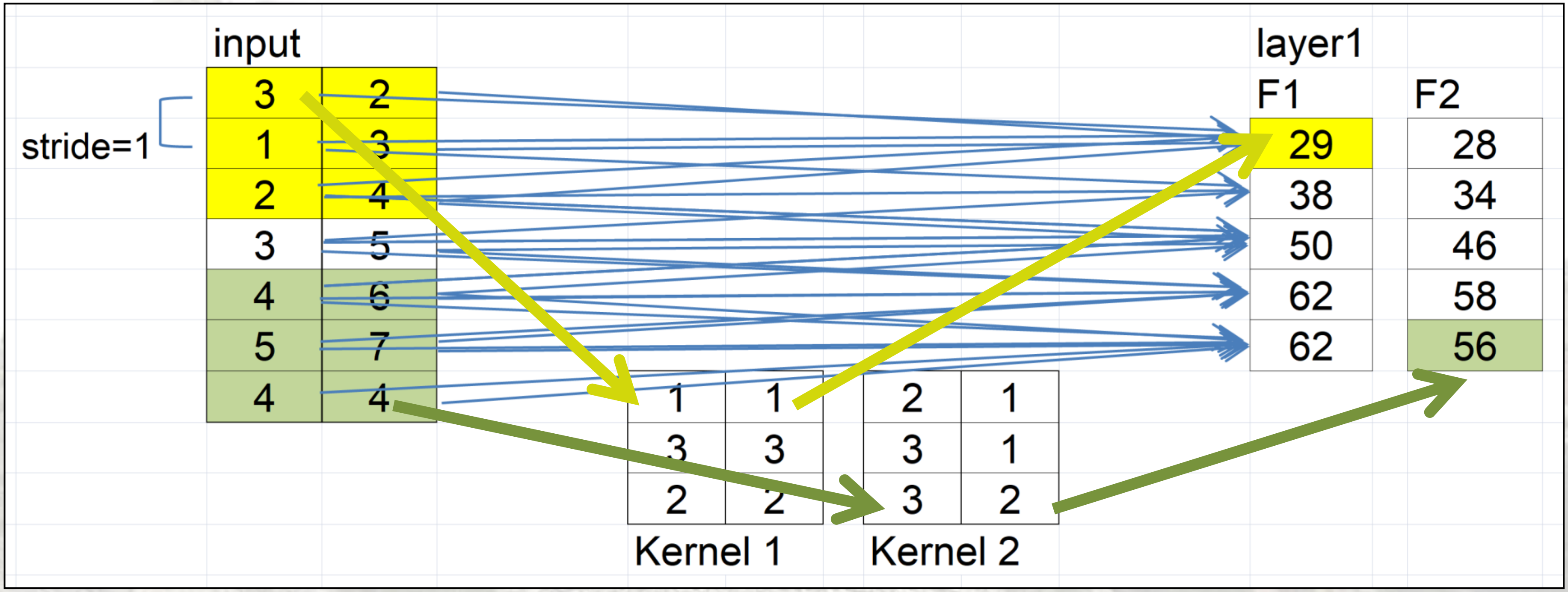
kernel

1	1
3	3
2	2

layer1

29
38
50
62
62

CNN – 2D, many filters



stride=1

input

3	2
1	3
2	4
3	5
4	6
5	7
4	4

Kernel 1

1	1
3	3
2	2

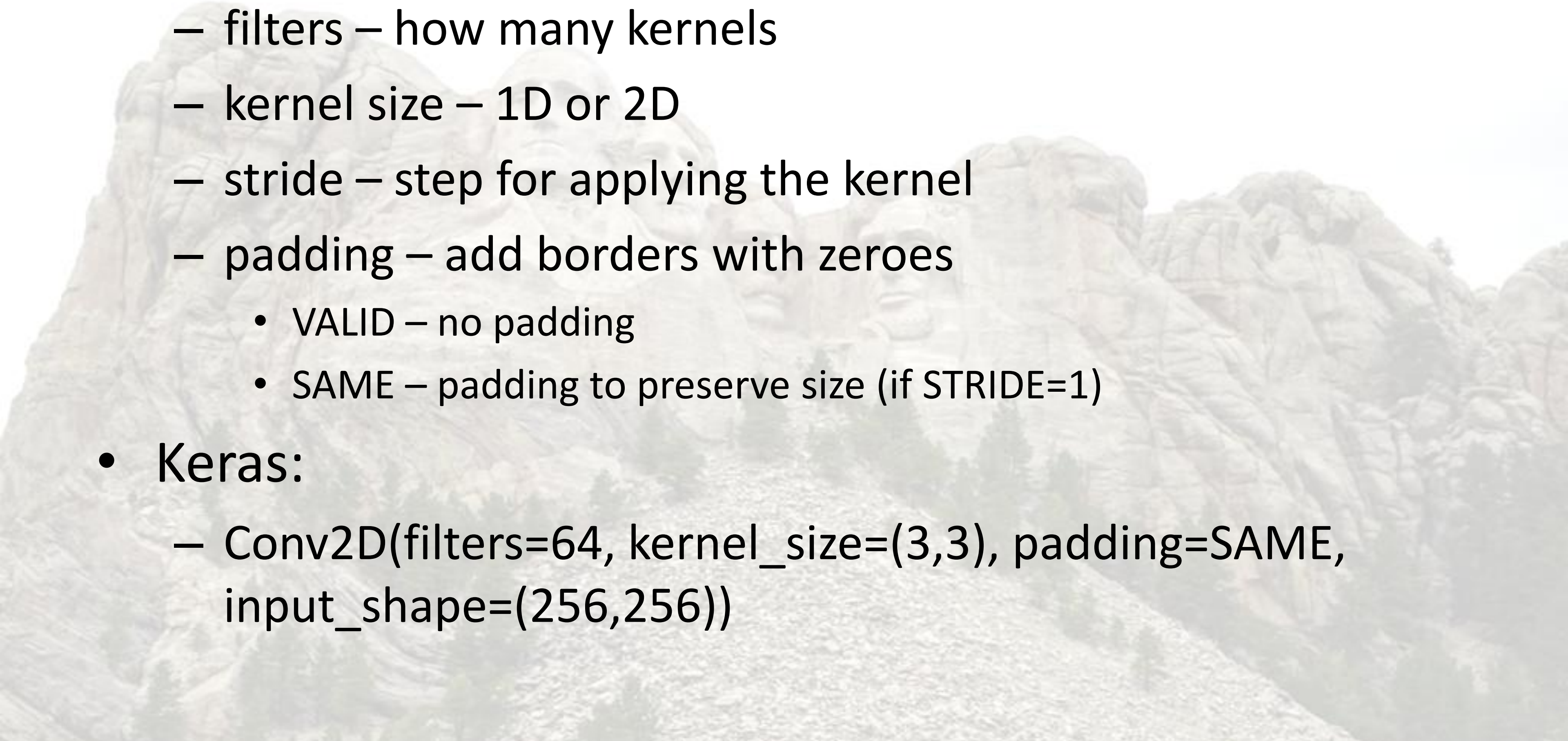
Kernel 2

2	1
3	1
3	2

layer1

F1	F2
29	28
38	34
50	46
62	58
62	56

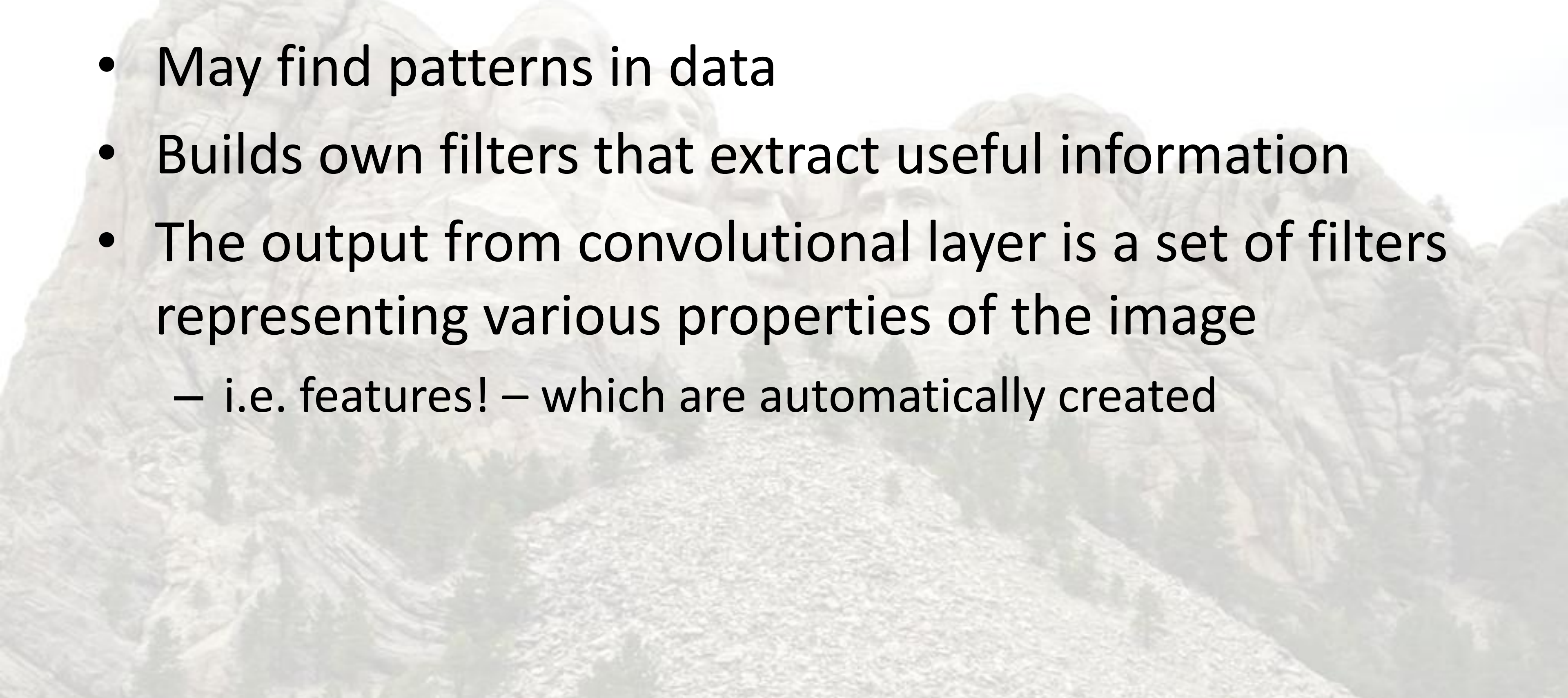
Convolutional Layer



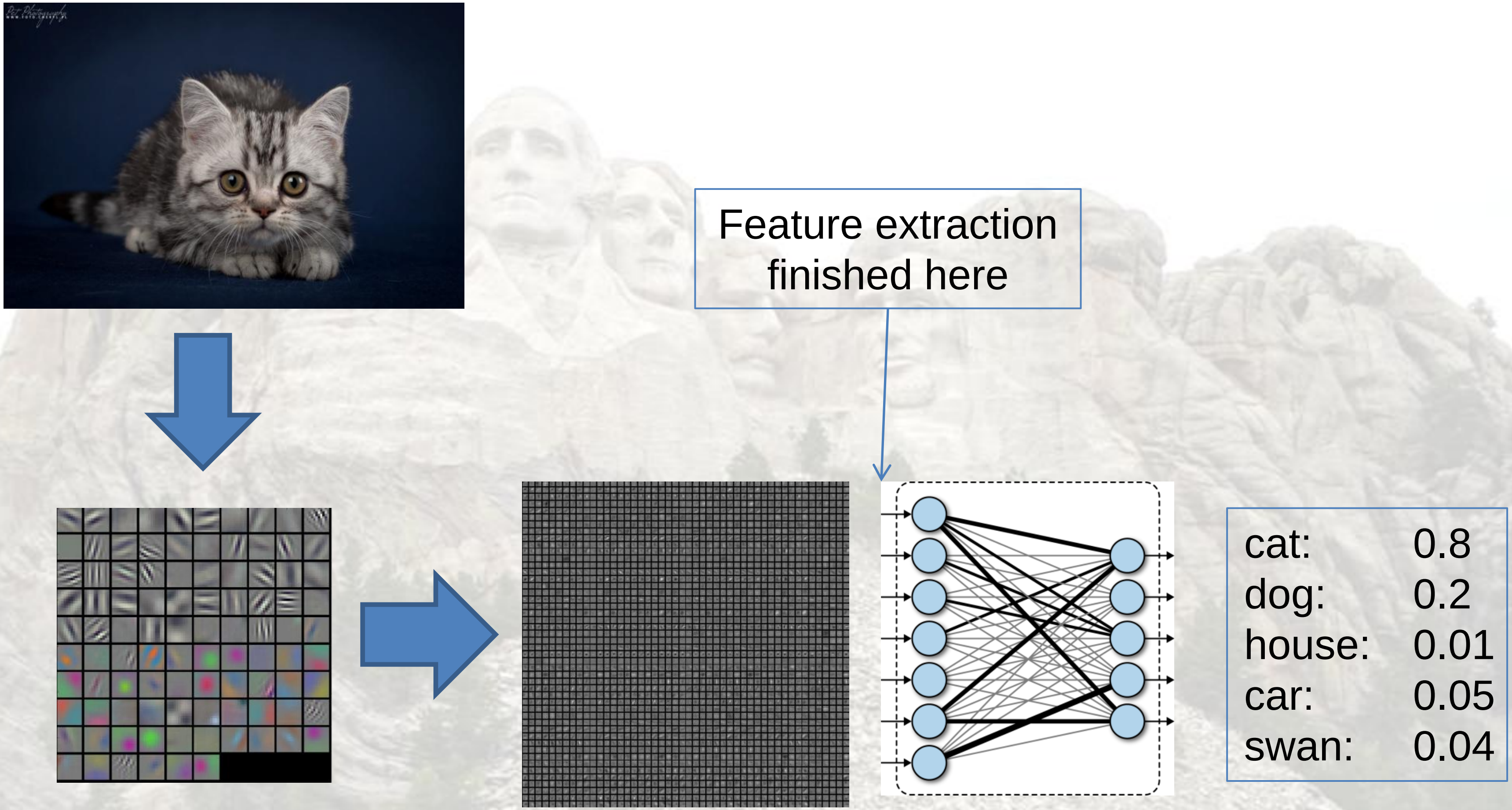
- Parameters:
  - filters – how many kernels
  - kernel size – 1D or 2D
  - stride – step for applying the kernel
  - padding – add borders with zeroes
    - VALID – no padding
    - SAME – padding to preserve size (if STRIDE=1)
- Keras:
  - Conv2D(filters=64, kernel_size=(3,3), padding=SAME, input_shape=(256,256))

Advantages of CNN

- Takes into account spatial/temporal relationships between features
- May find patterns in data
- Builds own filters that extract useful information
- The output from convolutional layer is a set of filters representing various properties of the image
  - i.e. features! – which are automatically created



Example of CNN



Feature extraction finished here

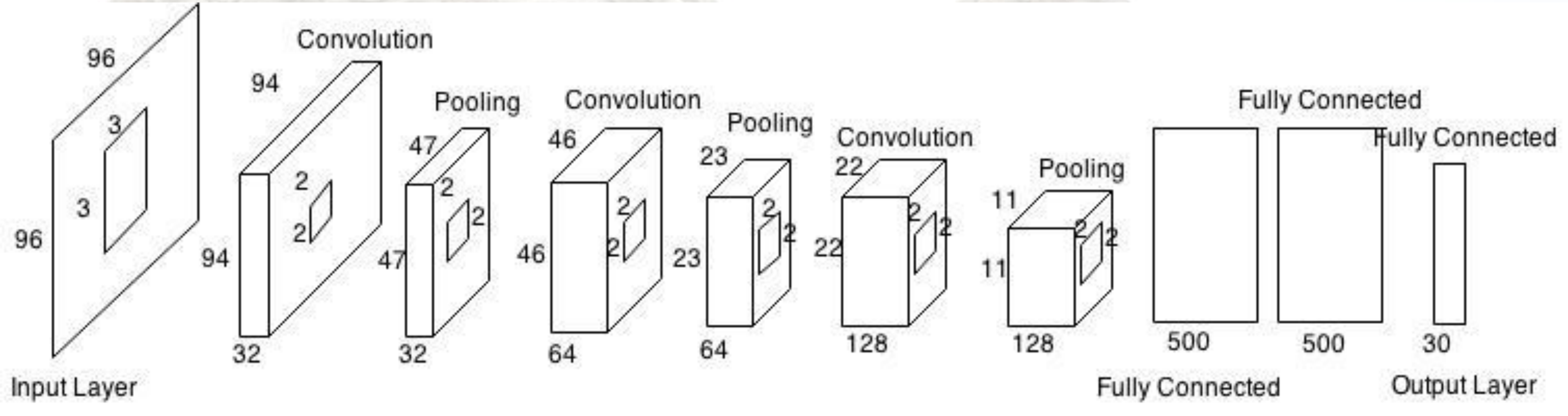
cat:	0.8
dog:	0.2
house:	0.01
car:	0.05
swan:	0.04

CNN Layers in Keras

- Conv2D(filters=16, kernel_size=(3, 3), padding="same",input_shape=(120,120))
  - classic layer
- MaxPooling2D(pool_size=(2, 2))
  - calculate max for given area
  - reduces size
- Dropout(rate=0.25)
  - randomly set rate percent of weights to zero
  - helps to prevent overfitting
- BatchNormalization(axes=-1)
  - normalizes output from the layer

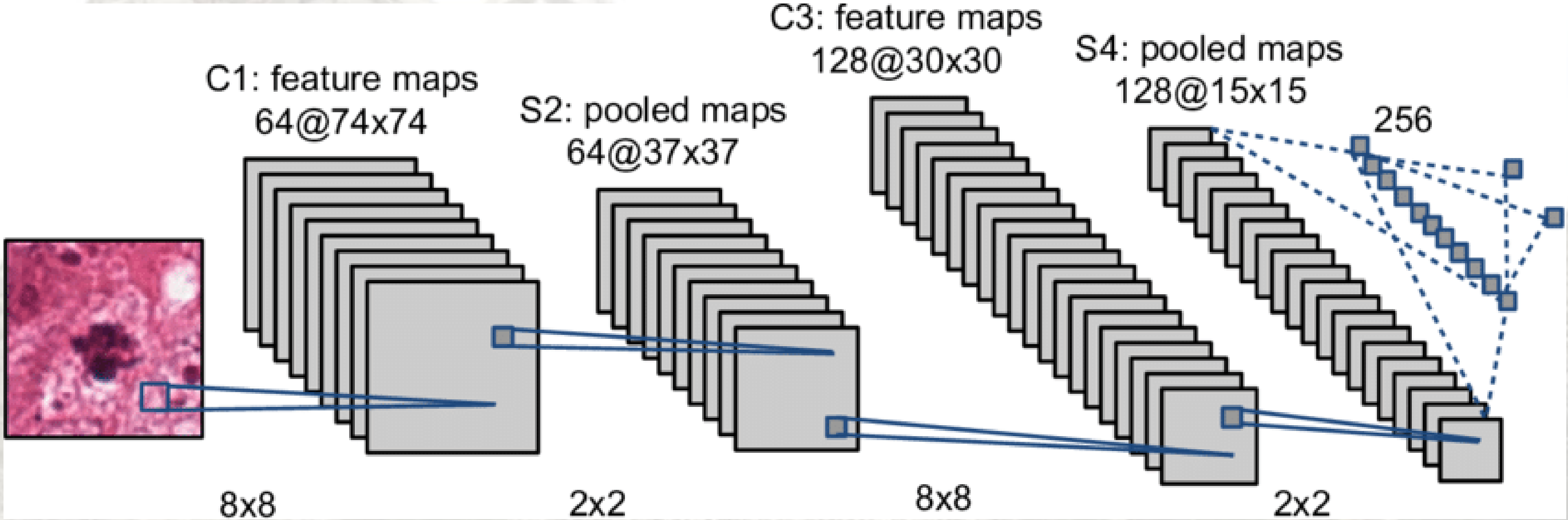
Cascade of layers

- Conv2D>MaxPooling>Conv2D>MaxPooling>Dropout
- Example:



https://www.hackerearth.com/practice/notes/

Another example



https://www.researchgate.net/publication/266734716

Flowers as images

flowers.ipynb

- Classification using:
  - Classic ANN
  - Decision Tree
  - Convolutional Neural Network