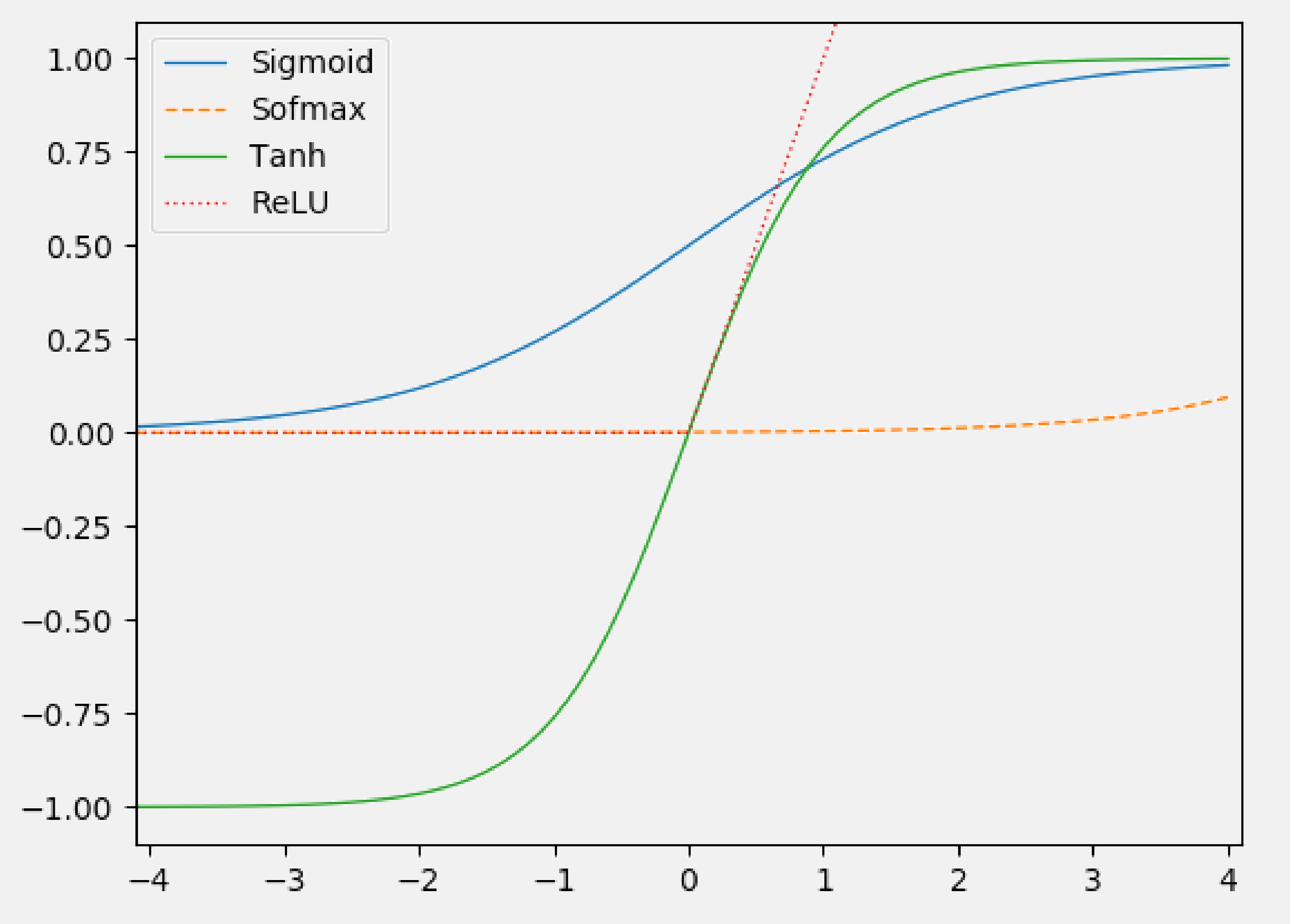
Seismic Attributes - from Interactive Interpretation to Machine Learning

Rafael Pires de Lima and Kurt J. Marfurt
(The University of Oklahoma)

Deep Learning and Convolutional Neural Networks

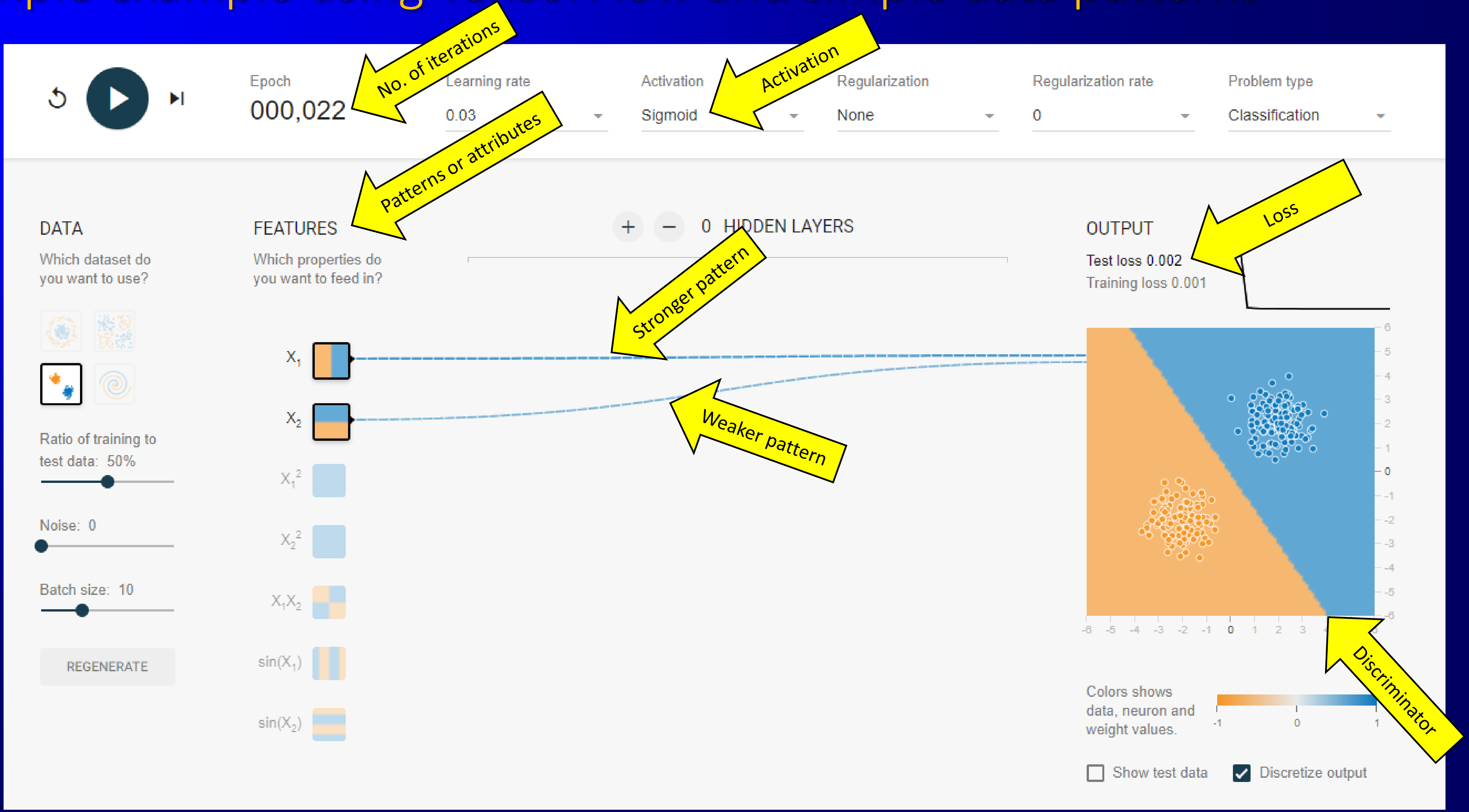
TensorFlow

Common activation functions (that answer a binary question)



No! Yes!

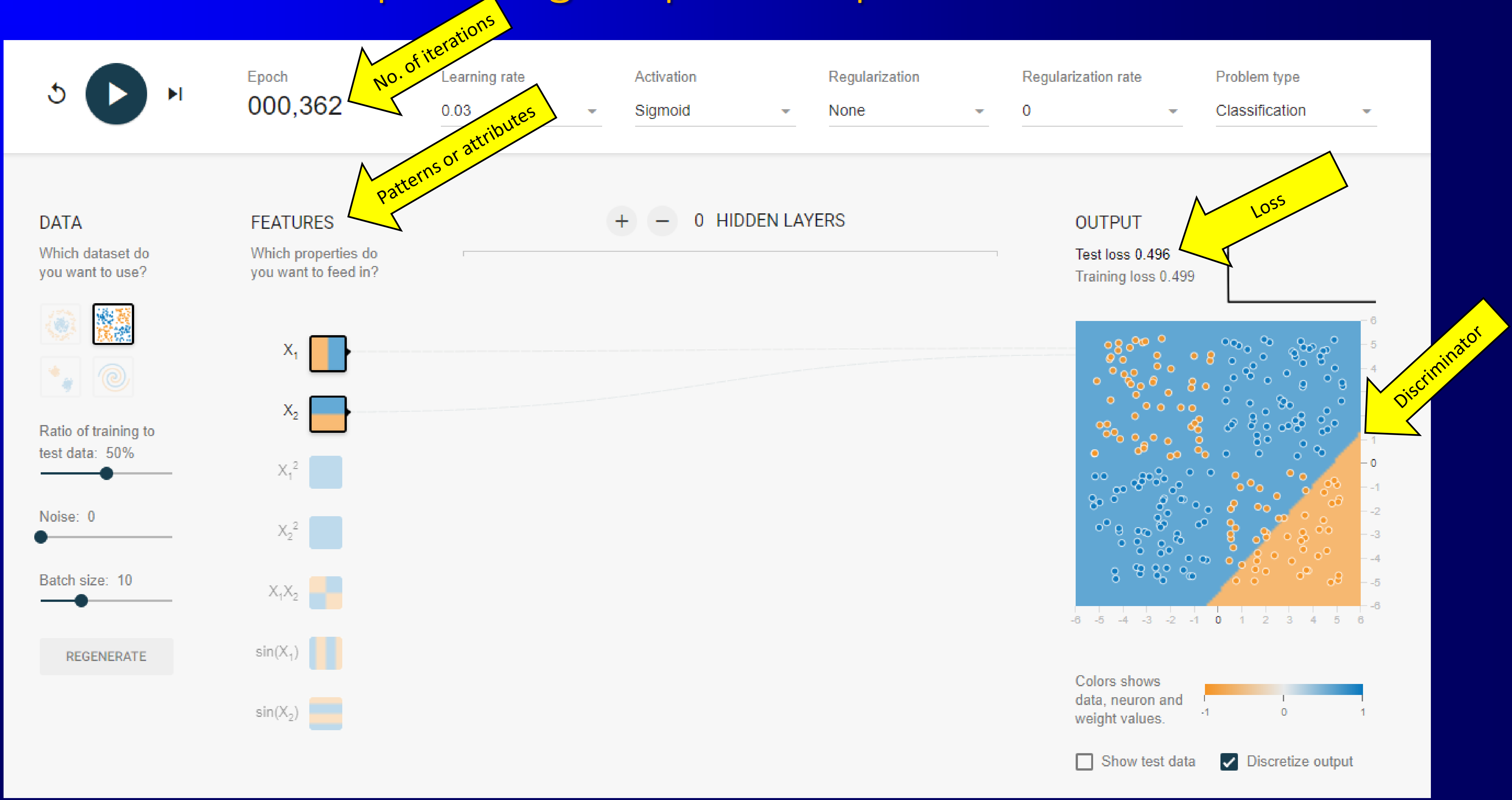
A simple example using TensorFlow and simple data patterns



18a-4

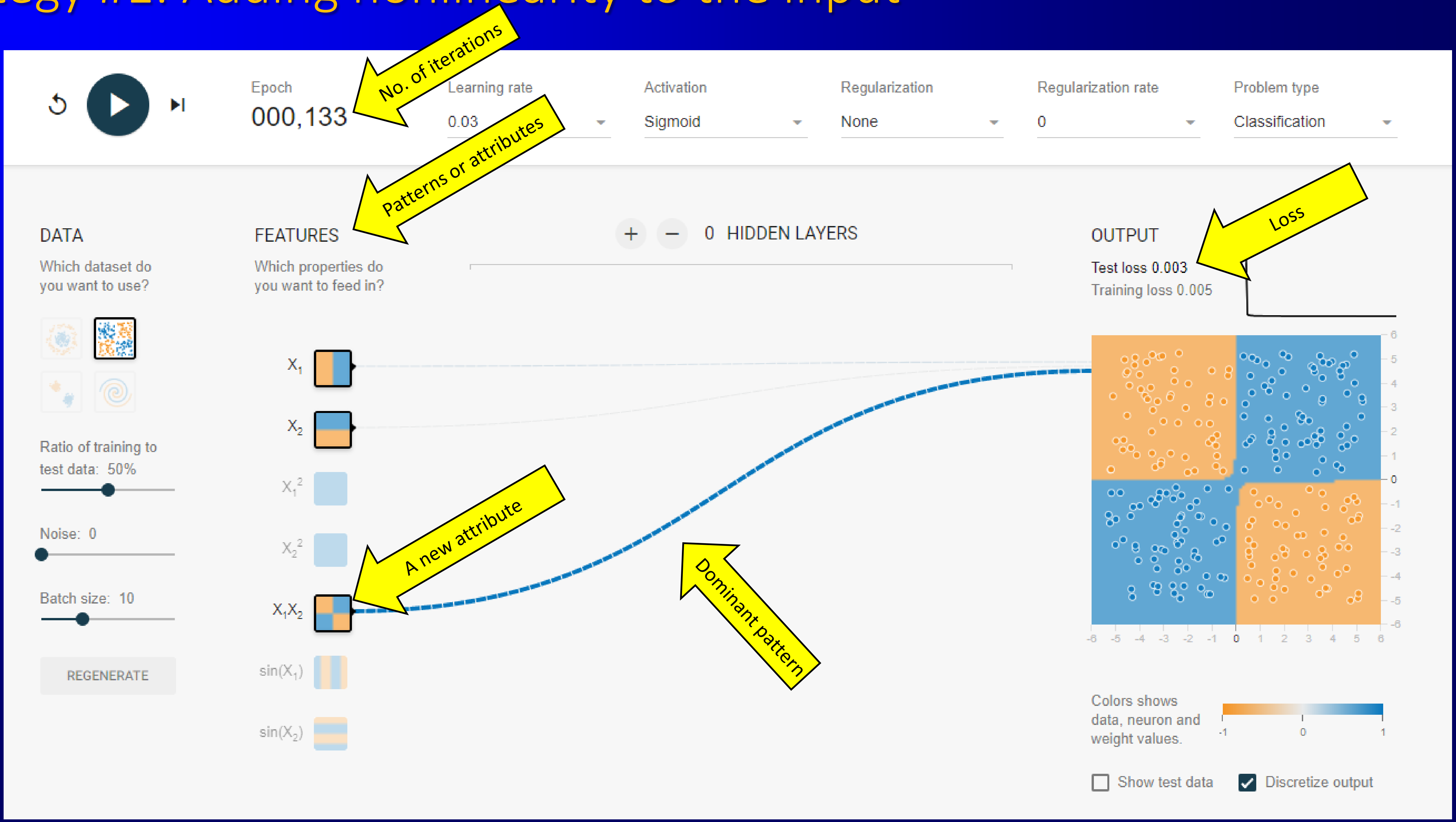
(https://playground.tensorflow.org/)

A more difficult example using simple data patterns



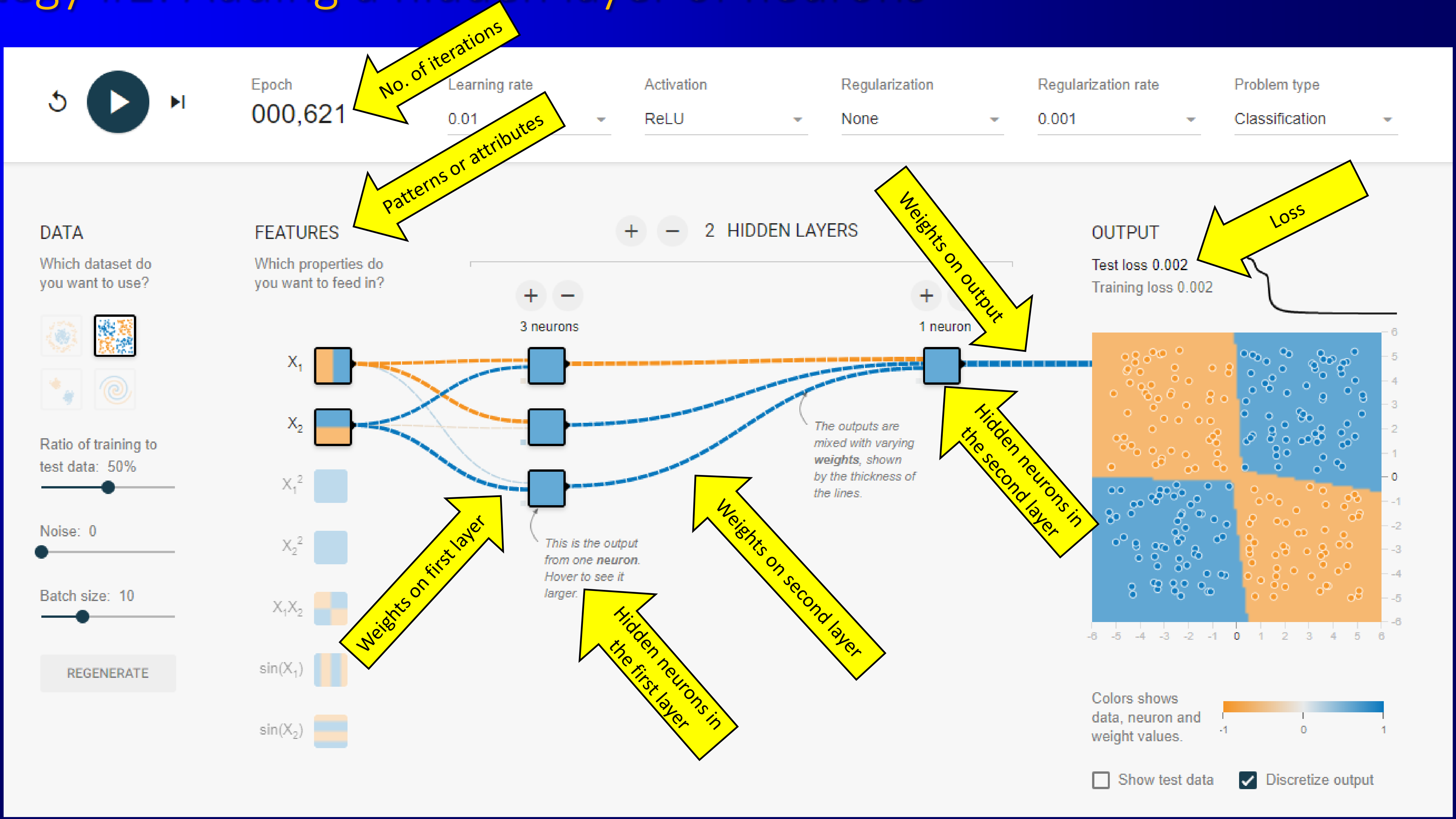
(https://playground.tensorflow.org/)

Strategy #1: Adding nonlinearity to the input



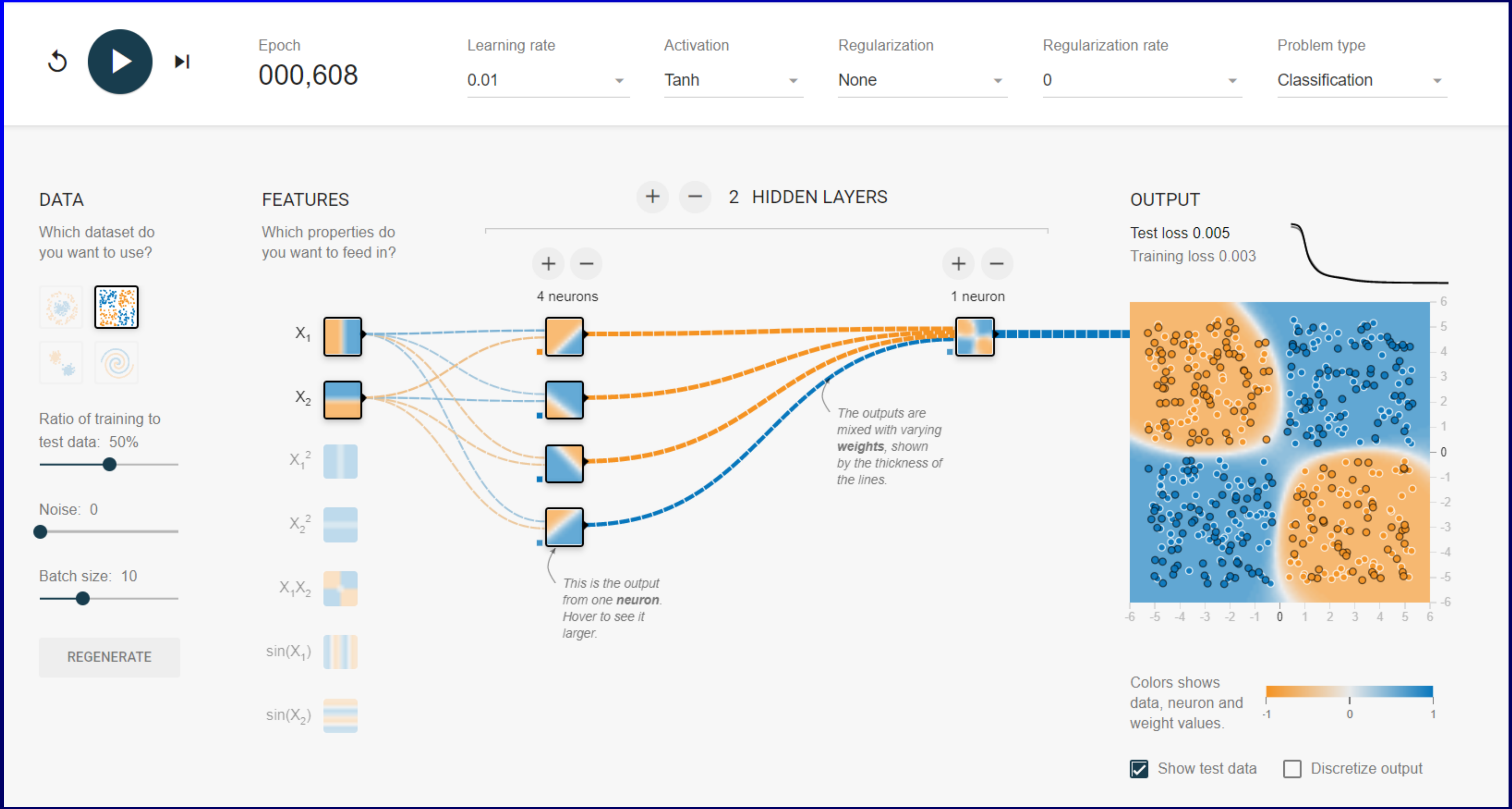
(https://playground.tensorflow.org/)

Strategy #2: Adding a hidden layer of neurons

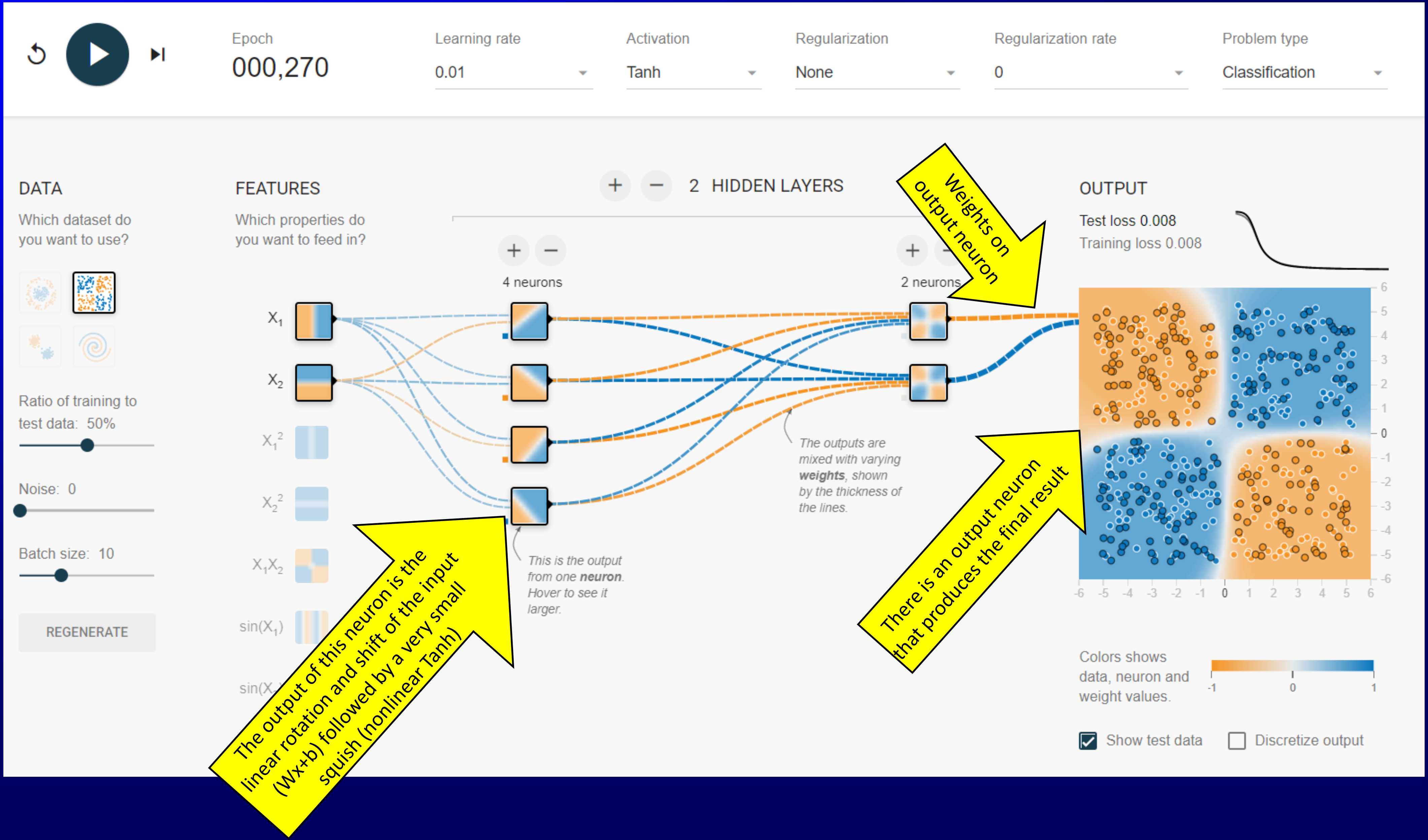


(https://playground.tensorflow.org/)

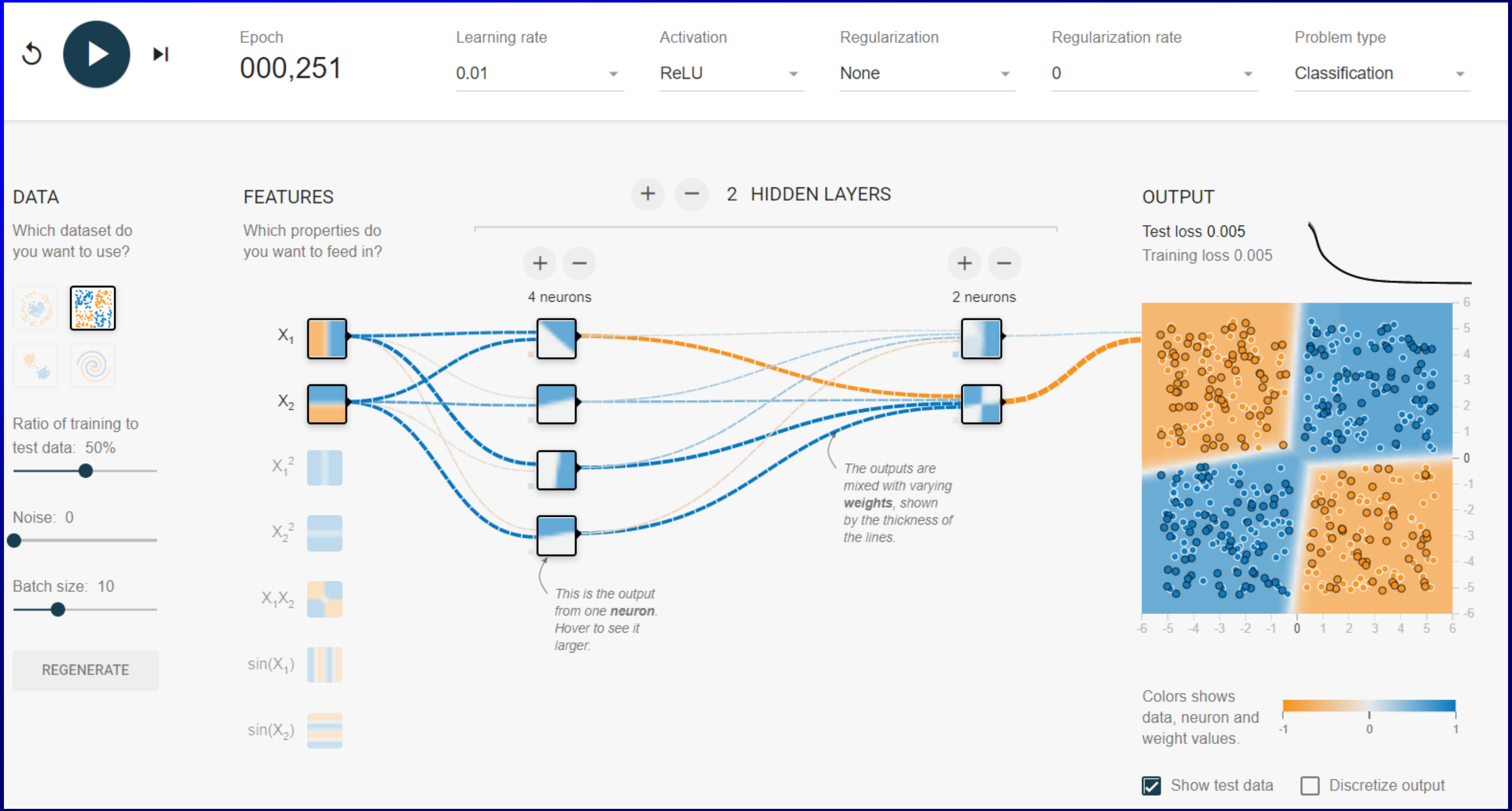
Examining the output of each neuron



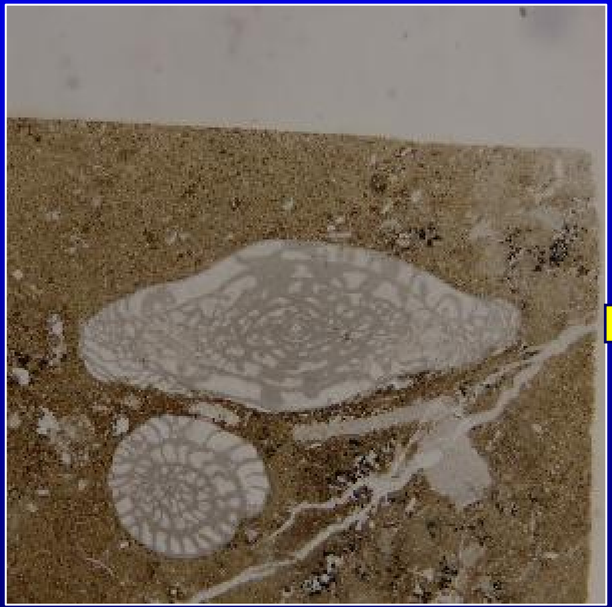
Adding a 2nd neuron in hidden layer 2 (tanh activator).



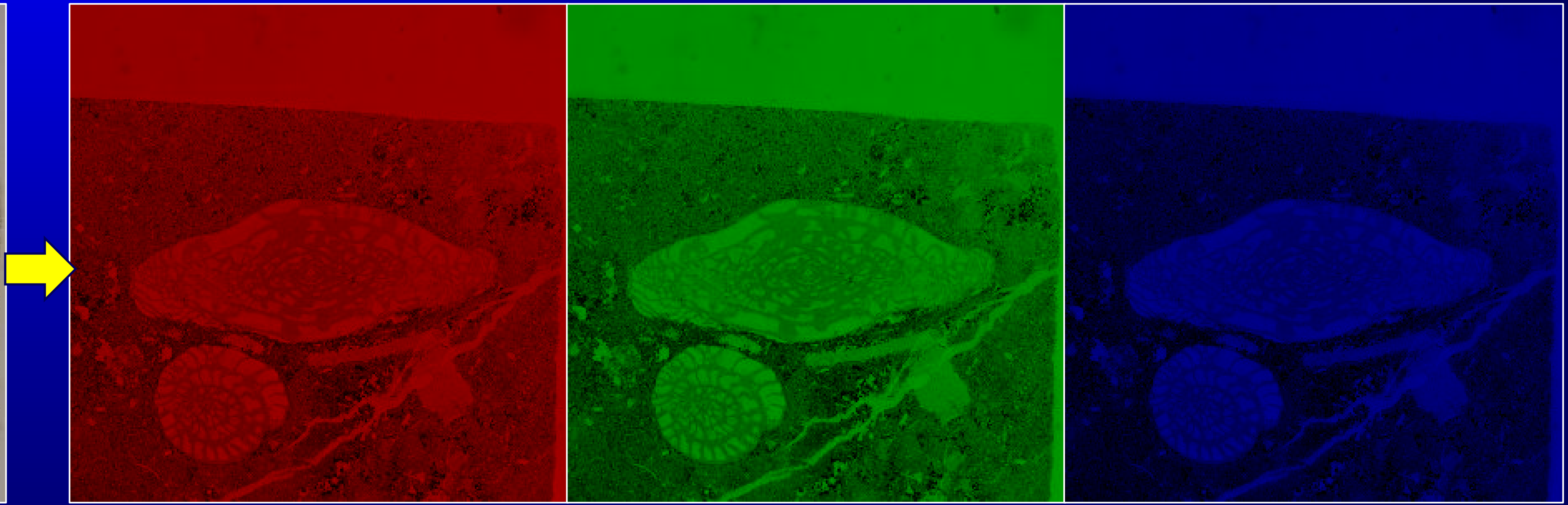
Adding a 2nd neuron in hidden layer 2 (ReLU activator).



Example 1: Classification of fusilinids



RGB composed image



Red channel | Green channel | Blue channel

(Pires de Lima et al., 2020)

Convolution of pixels in a photograph at two locations

Pad the edges with zeros

0	0	0	0	0	0
0	10	10	5	5	0
0	12	12	10	10	0
0	14	14	20	20	0
0	16	16	20	20	0
0	0	0	0	0	0

*

-1	0	1
-1	0	1
-1	0	1

=

22	-7	-7	-15
36	-1	-1	-35
42	8	8	-50
30	10	10	-40

Amplitude

Positive

Negative

(Pires de Lima et al., 2020)

Convolution of pixels in a photograph using a running window

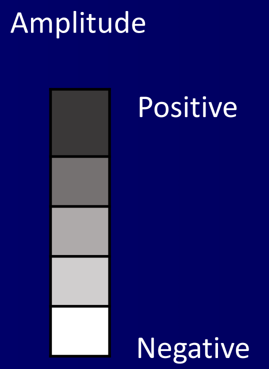
0	0	0	0	0	0
0	10	10	5	5	0
0	12	12	10	10	0
0	14	14	20	20	0
0	16	16	20	20	0
0	0	0	0	0	0

*

-1	0	1
-1	0	1
-1	0	1

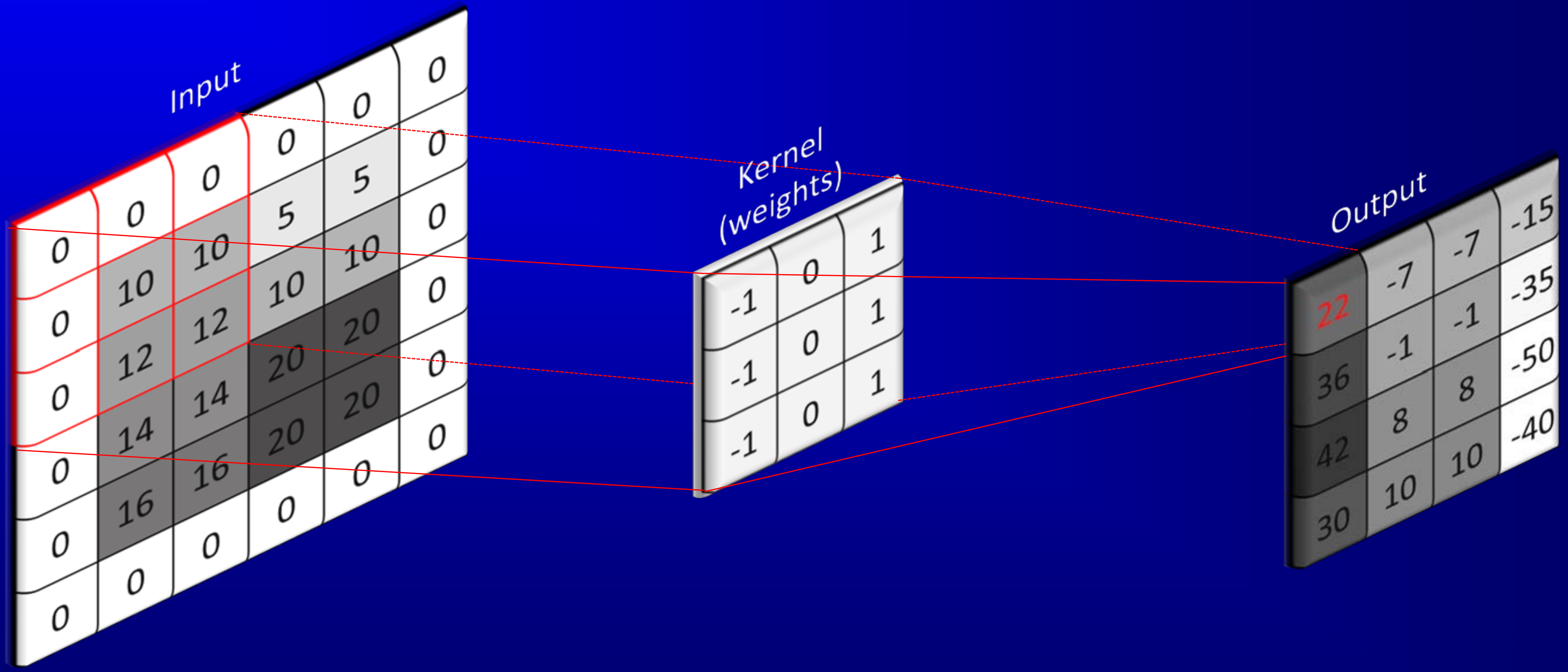
=

22	-7	-7	-15
36	-1	-1	-35
42	8	8	-50
30	10	10	-40



(Pires de Lima et al., 2020)

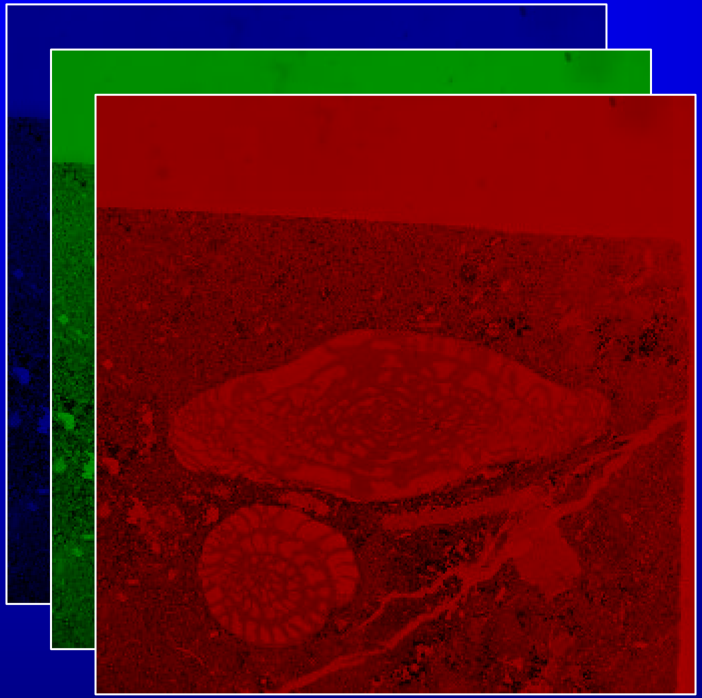
Convolution



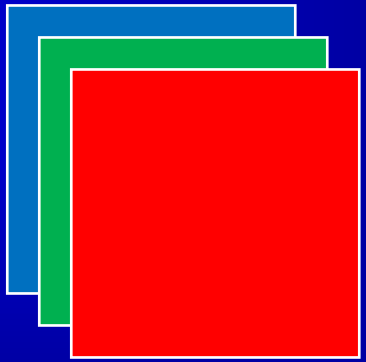
Output = 0*(-1) + 0*0 + 0*1 + 0*(-1) + 10*0 + 10*1 + 0*(-1) + 12*0 + 12*1 = 22

(Pires de Lima et al., 2020)

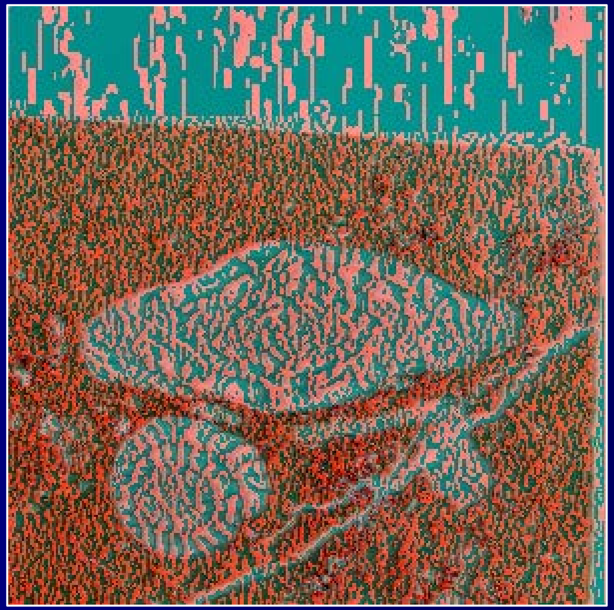
Convolution with 3D (3-component or RGB image) input



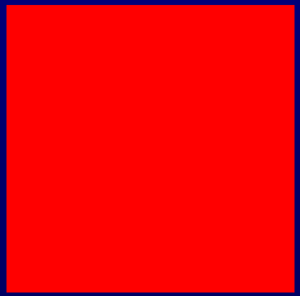
*



=

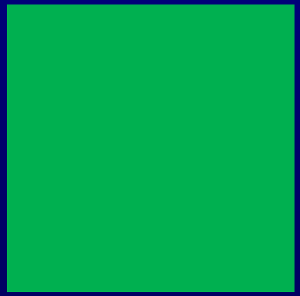


where:



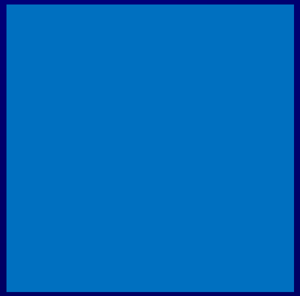
=

-1	0	1
-1	0	1
-1	0	1



=

0	0	0
0	1	0
0	0	0

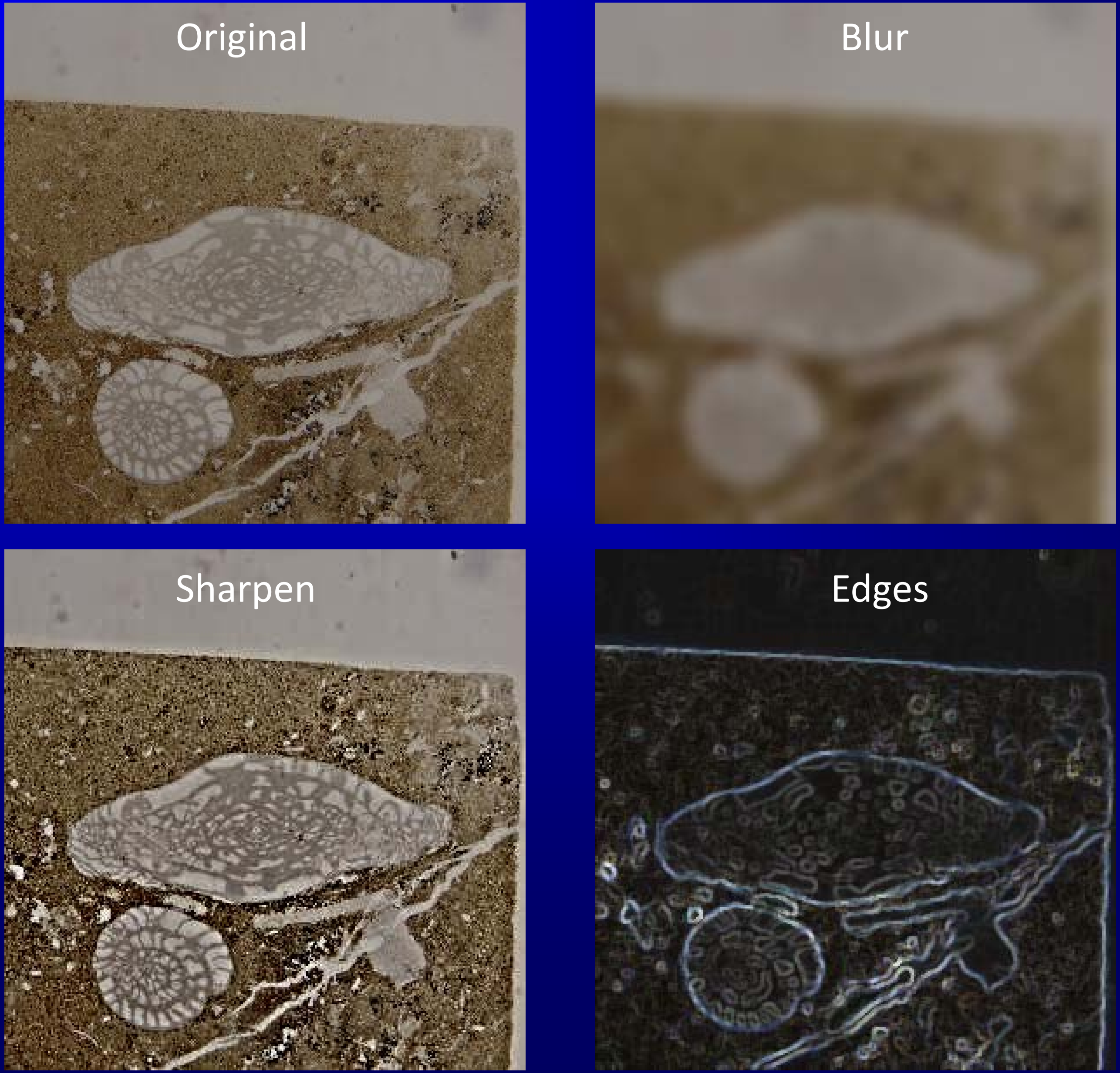


=

0	0	0
0	1	0
0	0	0

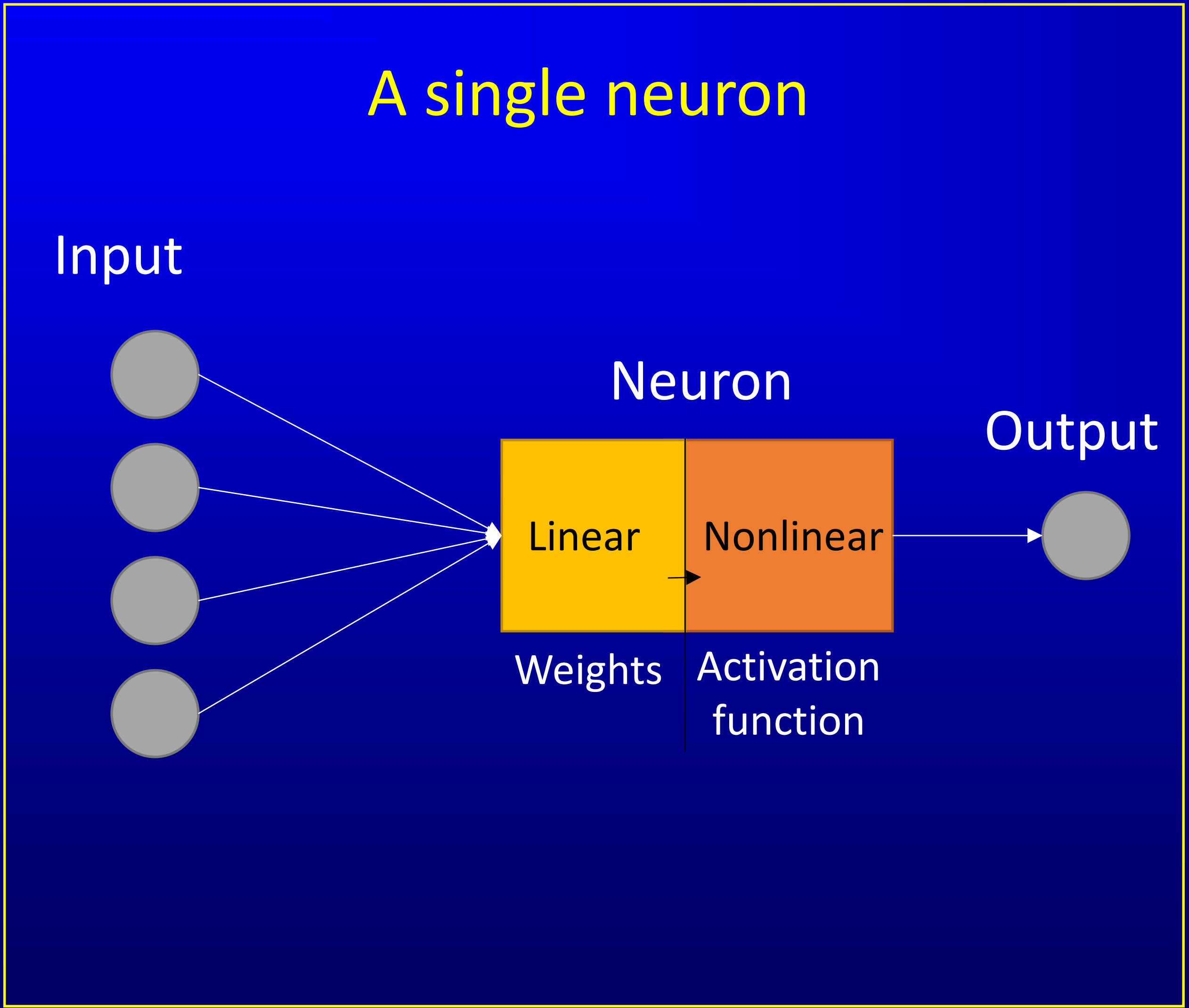
(Pires de Lima et al., 2020)

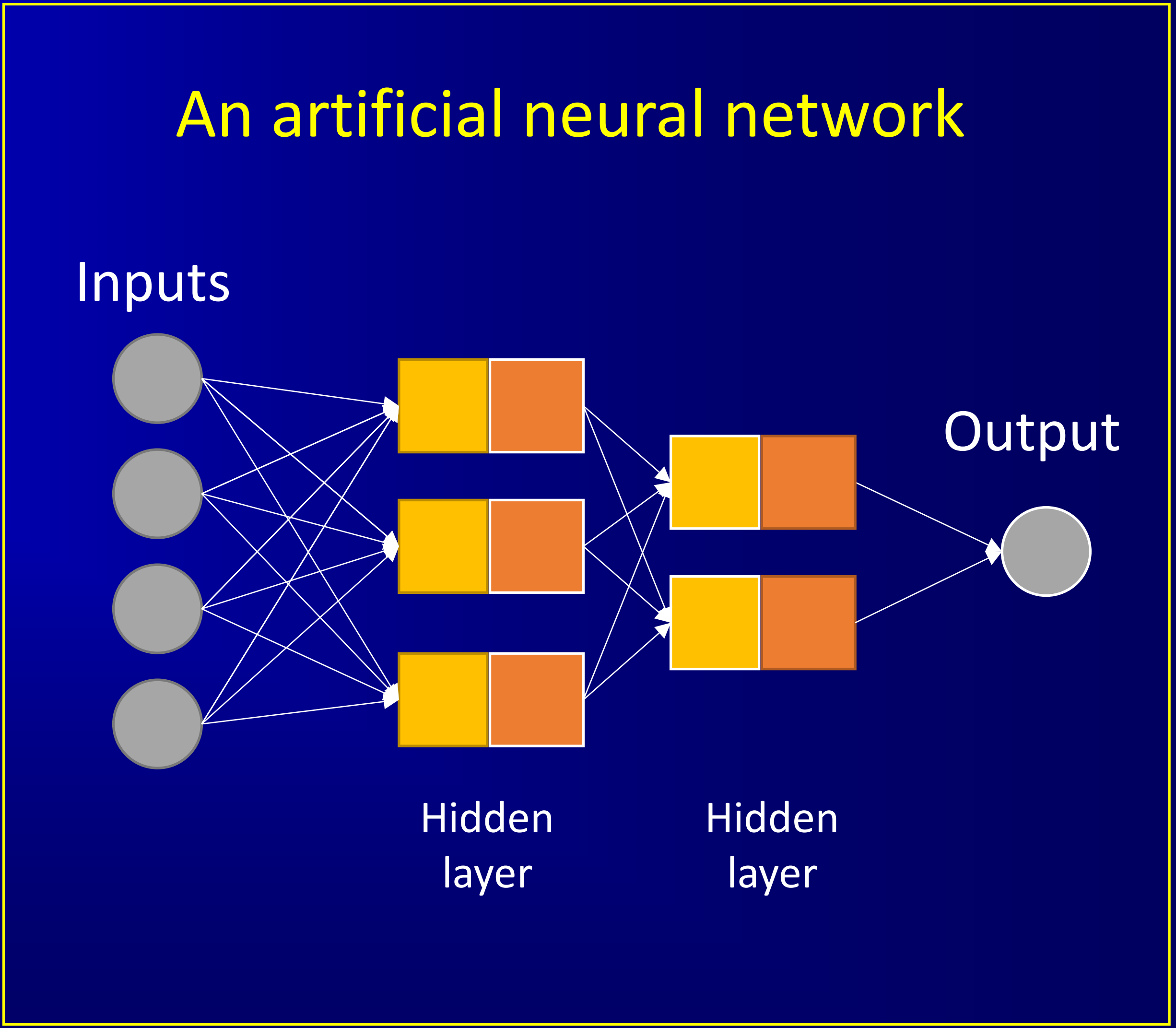
Common PhotoShop filters based on convolutions



(Pires de Lima et al., 2020)

Networks of neurons





Visualizing application of a single neuron

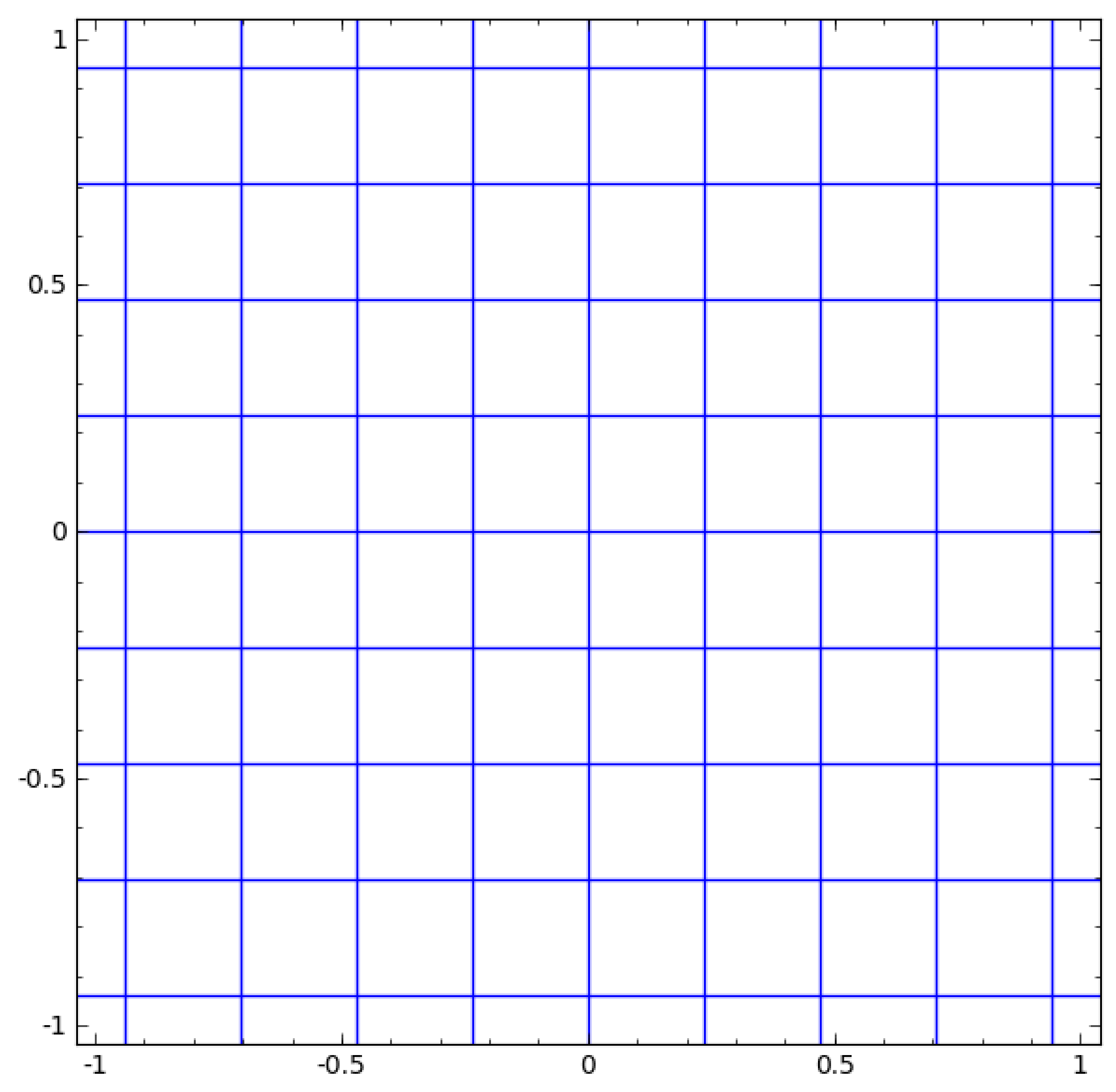
Step 1: Linear distortion of the input attribute data **x**:

$$\mathbf{z} = (\mathbf{W}\mathbf{x}+\mathbf{b})$$

where the "weight" matrix **W** is equivalent to convolution and where the rotated and scaled result is translated (biased) the vector **b**

Step 2: Point-wise application using a tanh activator function:

$$\mathbf{r}=\tanh(\mathbf{z})$$

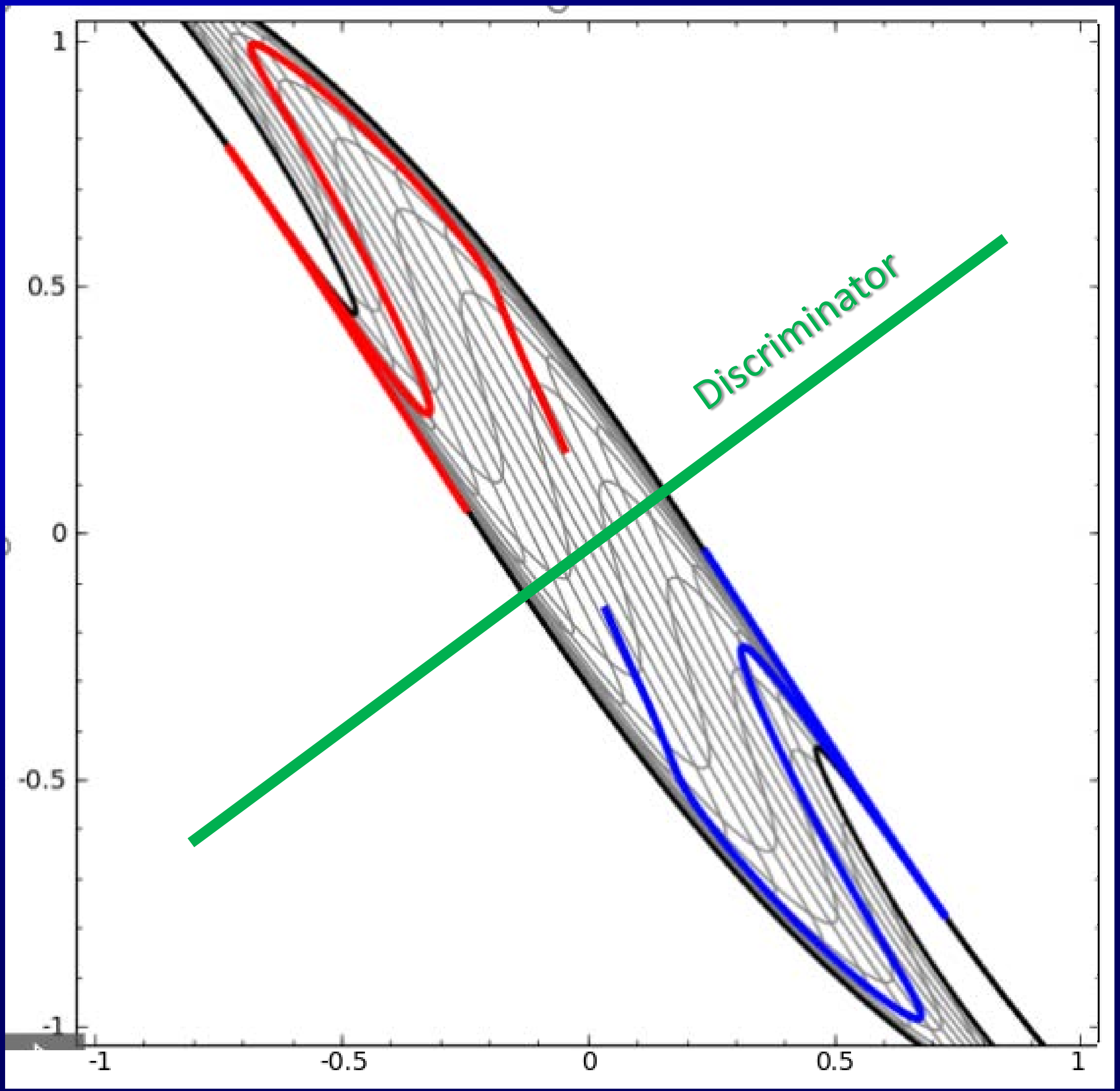


(http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/)

Visualizing the application of four neurons

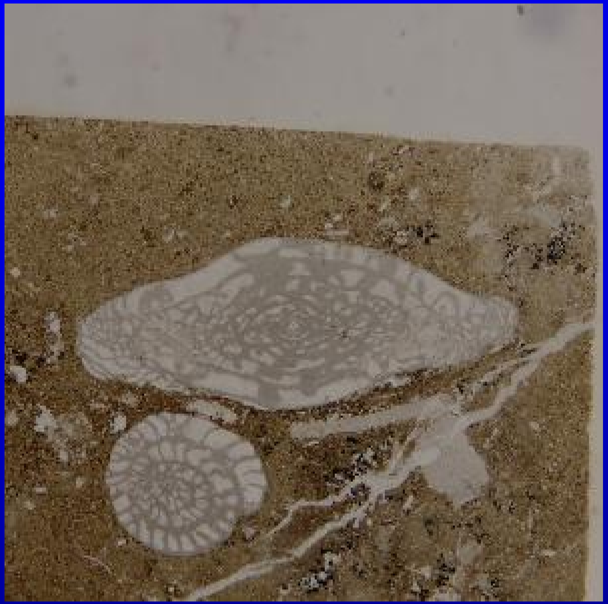
- Originally the two classes are "entangled" and cannot be separated by a linear discriminator

- After applying linear transformation and space distortion (four hidden layers) we can separate the two classes with a planar surface

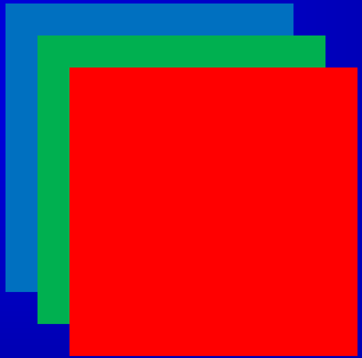


(http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/)

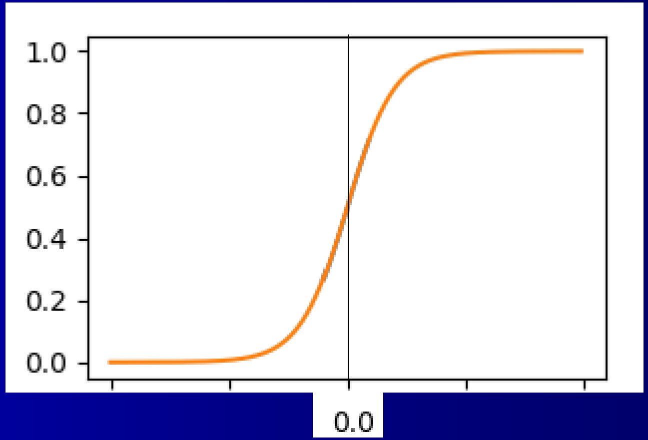
Activation function



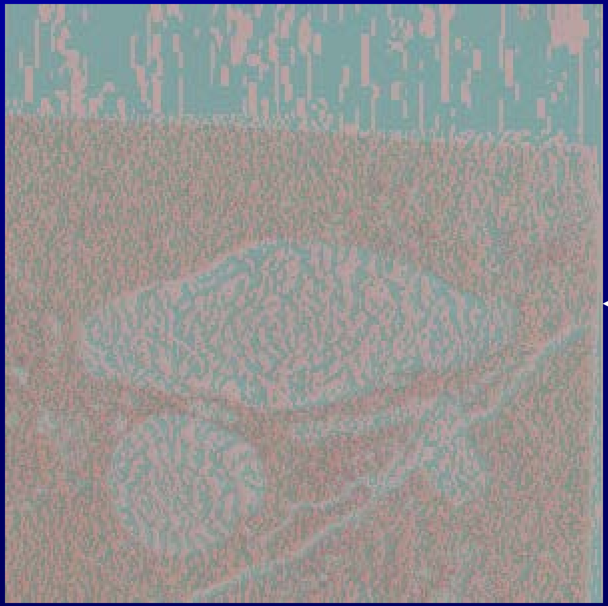
*



Activation function:
Elementwise multiplication



=



(Pires de Lima et al., 2020)

"Pooling" multiple input values into a single output

There are different ways to pool the data, including:

- Maximum
- Minimum
- Mean
- Median

0	0	0	0	0	0
0	10	10	5	5	0
0	12	12	10	10	0
0	14	14	20	20	0
0	16	16	20	20	0
0	0	0	0	0	0

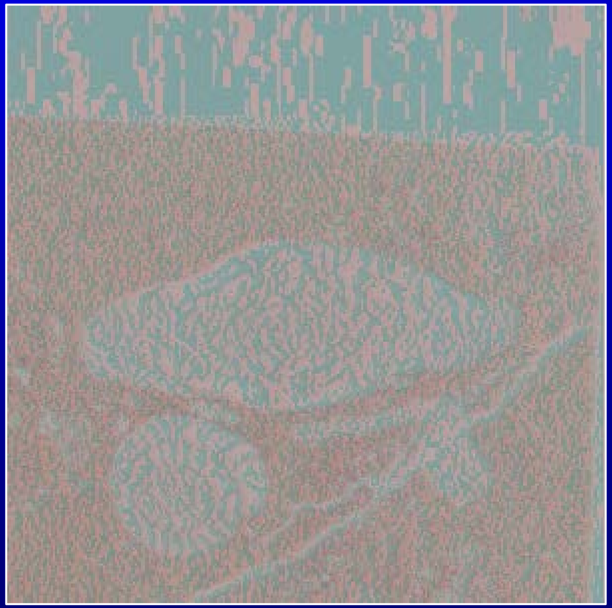
"Max" pooling

3x3 kernels applied using stride 3

12	10
16	20

An example of pooling

Max pooling (maximum value in a subset window) is a common choice



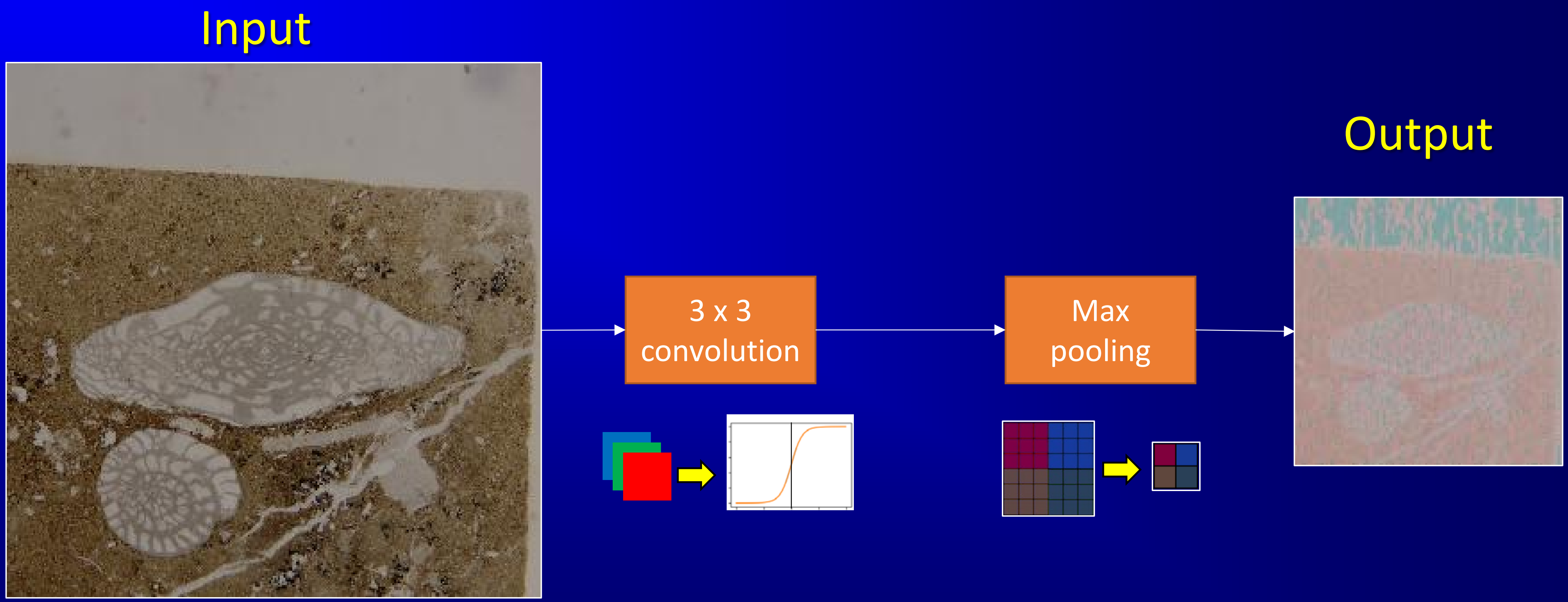
300x300 pixels

Max pooling

2x2 kernels stride 2

150x150 pixels

A very simple (“incomplete”) Convolutional Neural Network



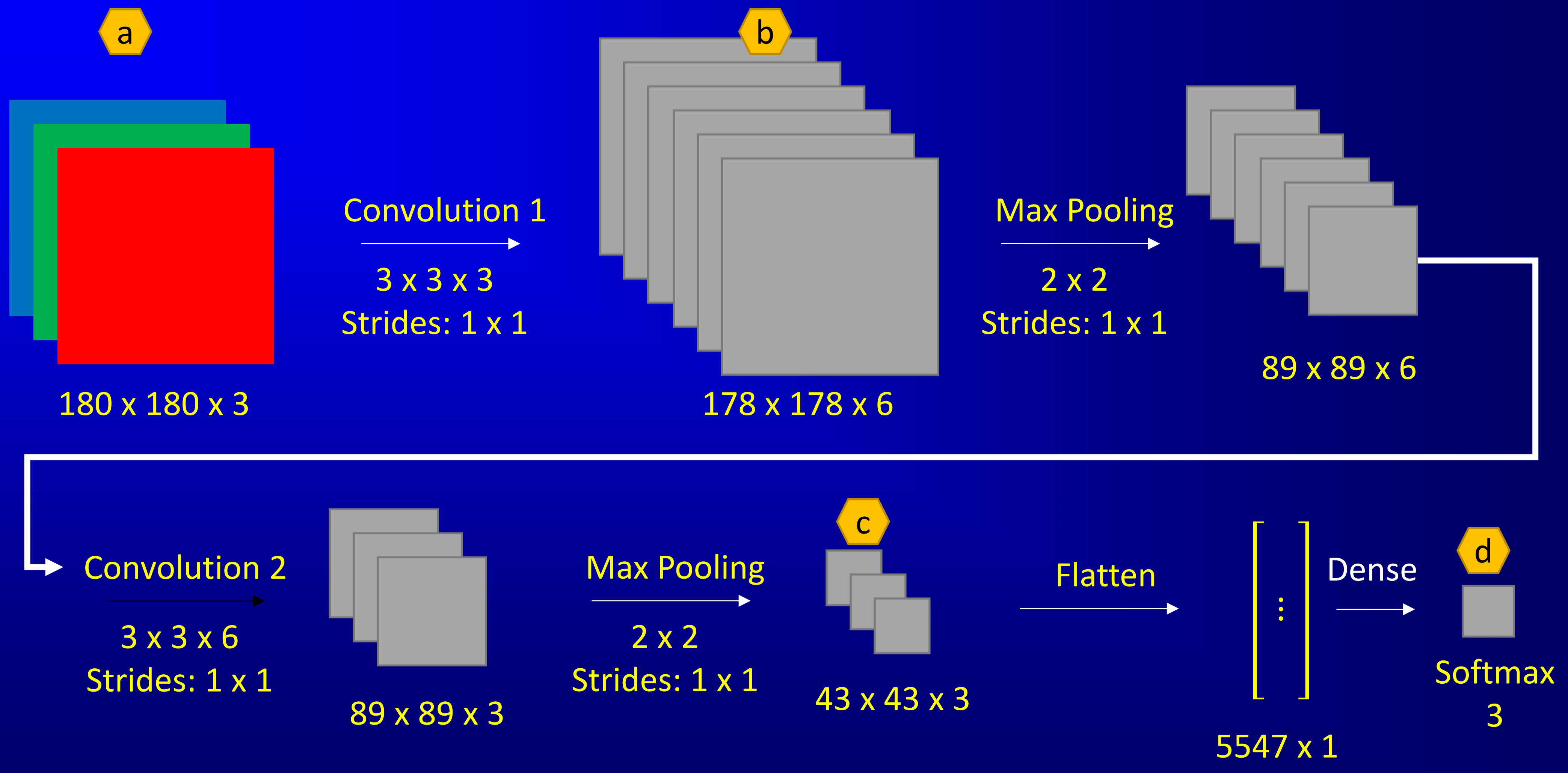
Three distinct classes

Class 1

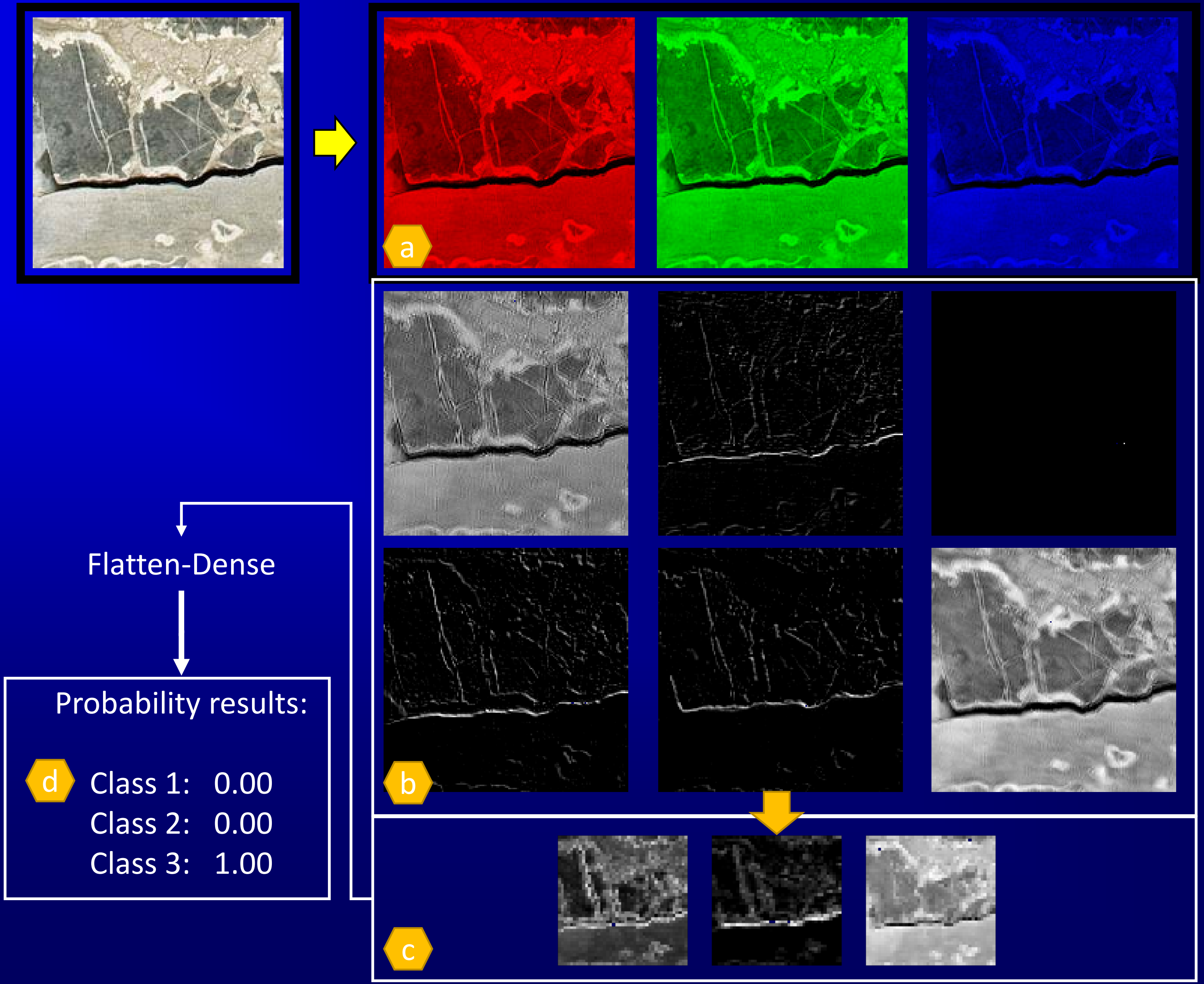
Class 2

Class 3

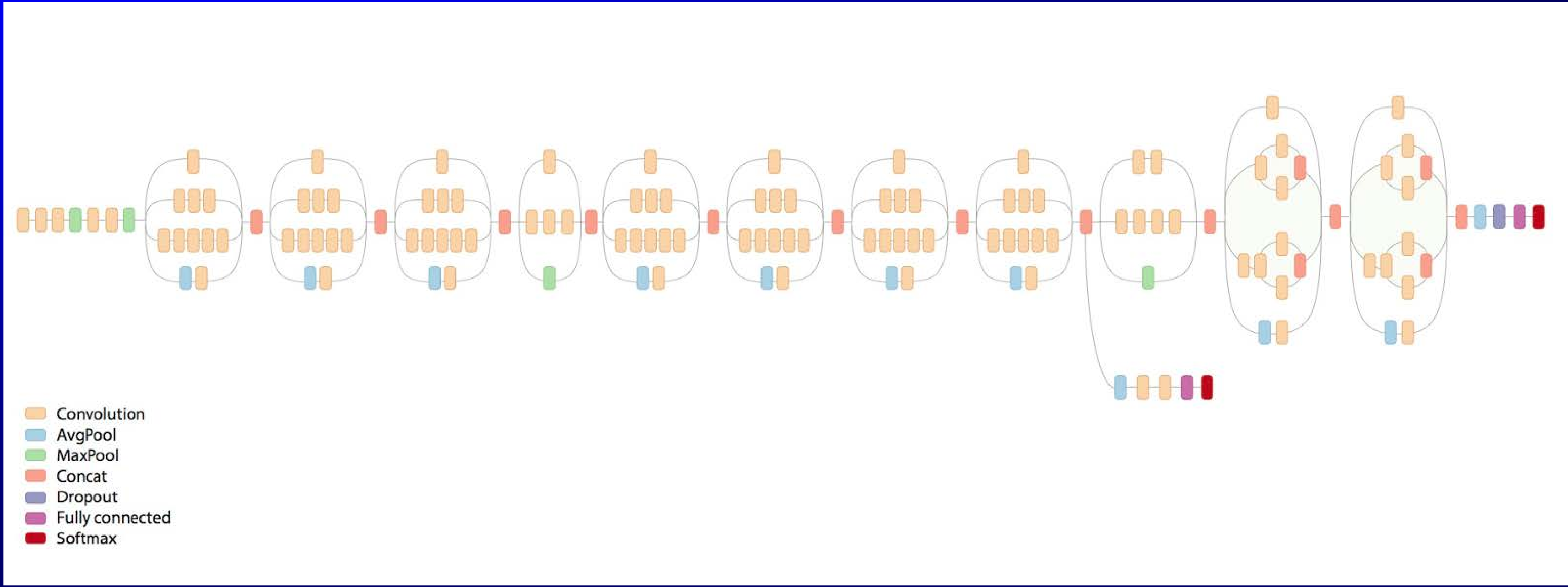
A not-so-deep convolutional neural network



Example


a
b
c
Flatten-Dense
Probability results:
d
Class 1: 0.00
Class 2: 0.00
Class 3: 1.00

Deep Convolutional Neural Networks



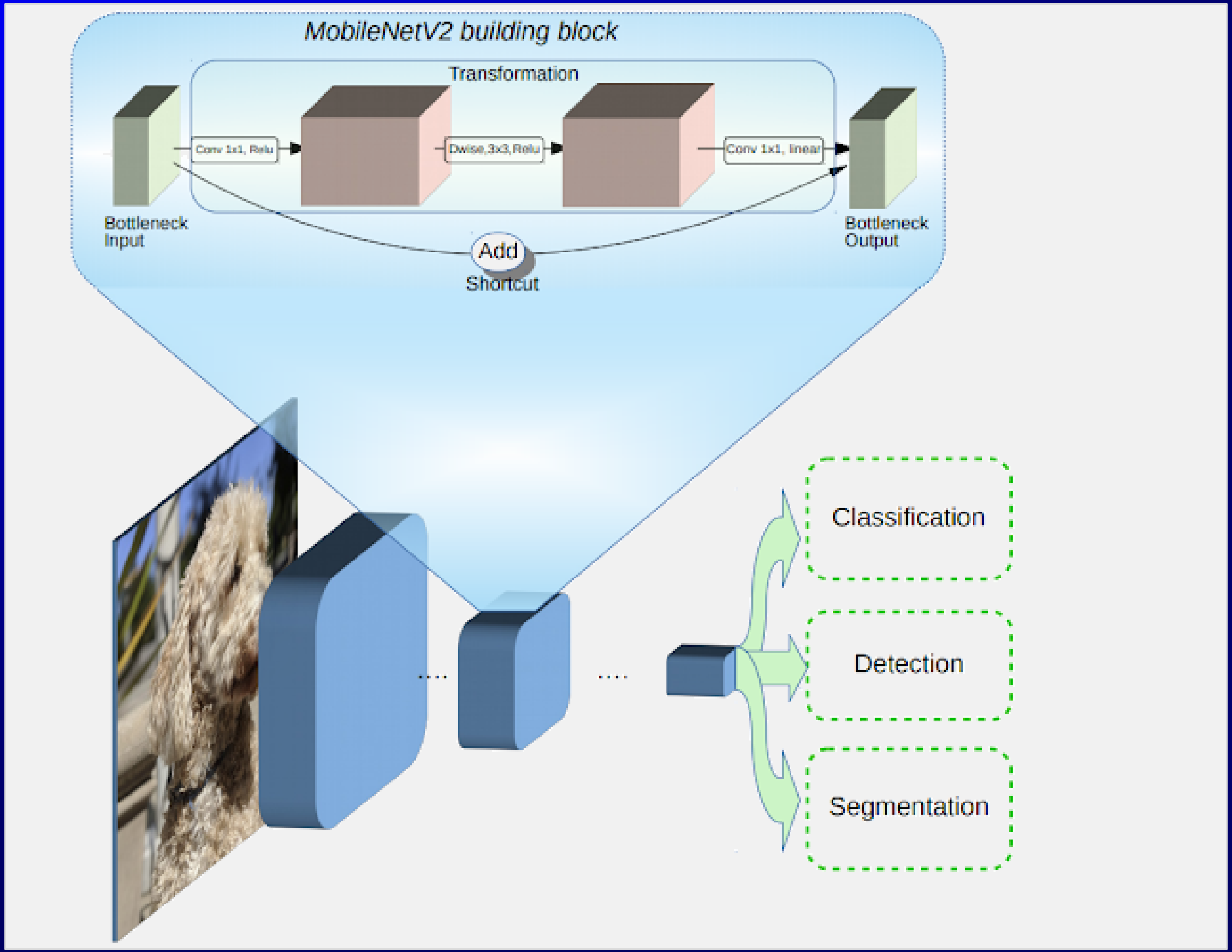
https://ai.googleblog.com/2016/08/improving-inception-and-image.html

(Szegedy et al. , 2015)

(Inception V3)

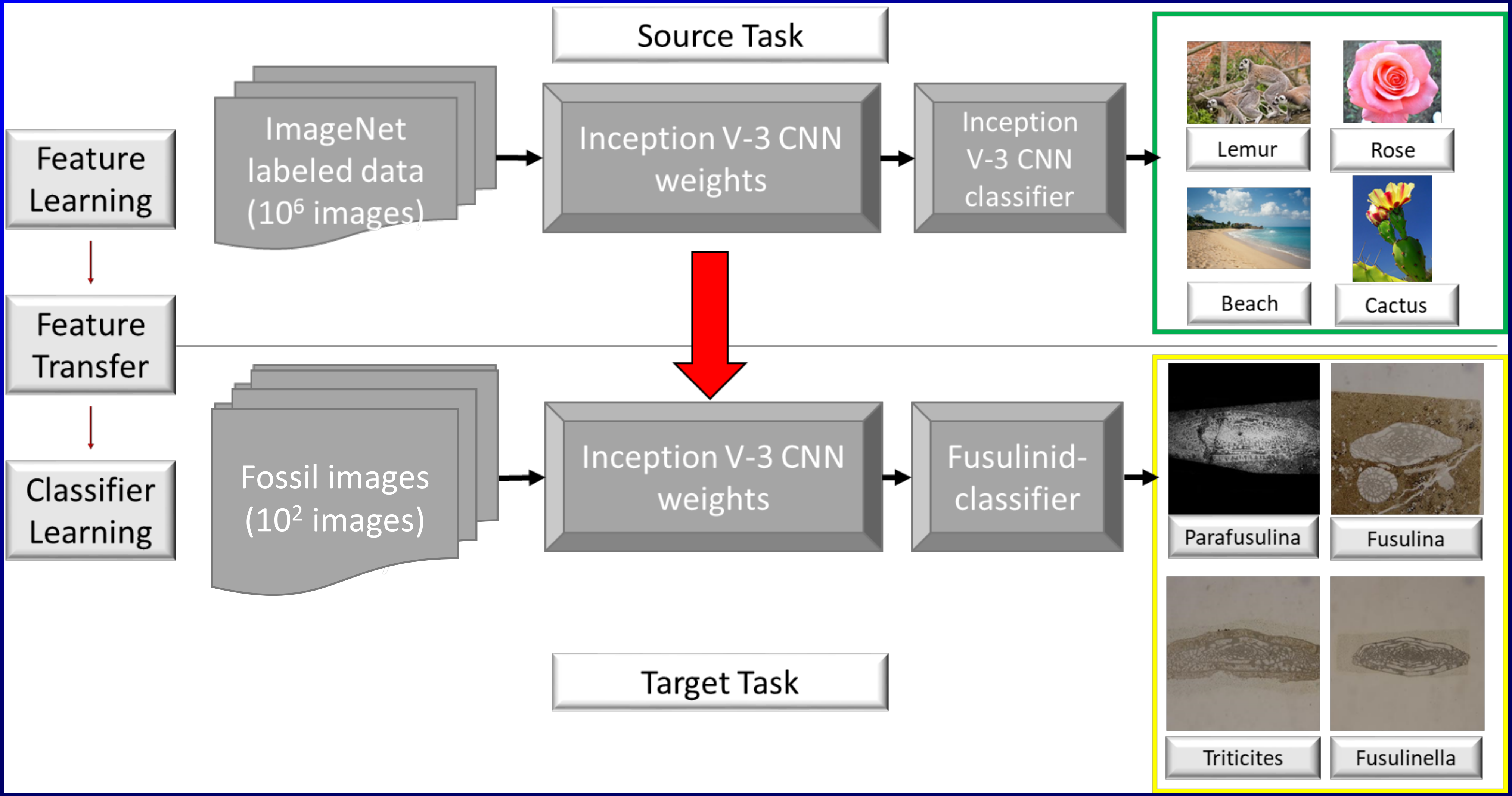
Deep Convolutional Neural Networks

ResNetV2-50



https://ai.googleblog.com/2018/04/mobilenetv2-next-generation-of-on.html

Transfer learning from the V-3 CNN classifier

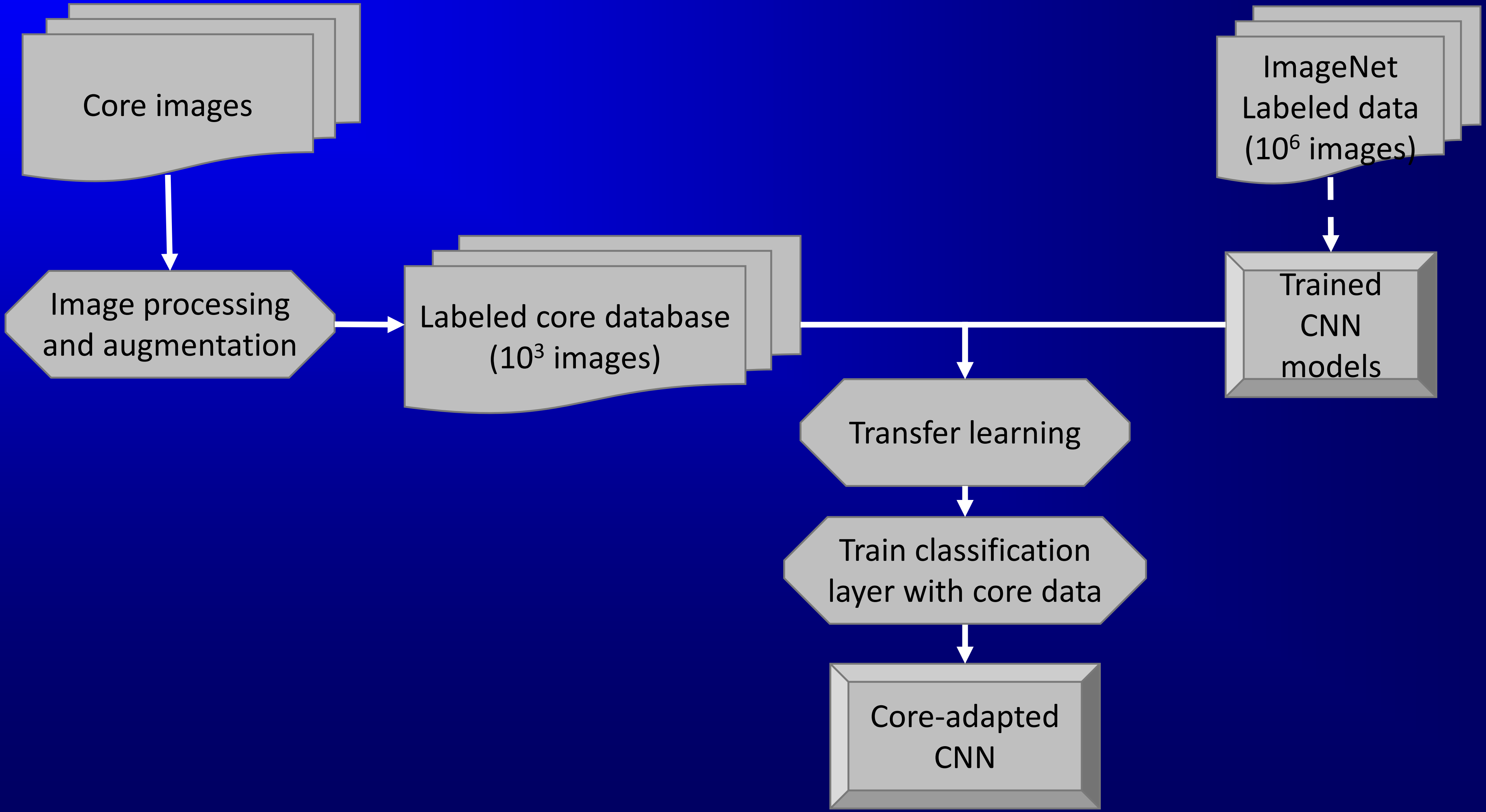


(Modified from Oquab et al., 2014)

Oklahoma Petroleum Information Center

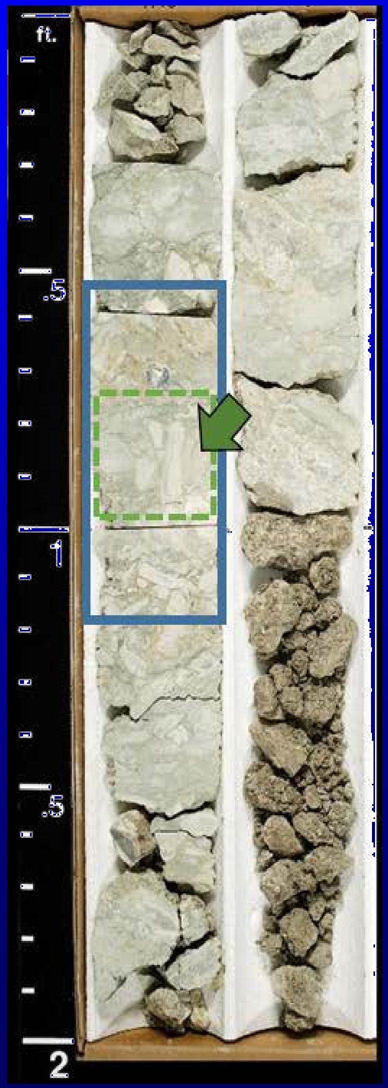
Let's apply our method to some Mississippi Lime core analyzed by OGS petroleum geologist Fnu Suriamin

Core analysis workflow



(Rafael Pires de Lima et al., 2018)

Core analysis (Mississippi Lime play, Oklahoma)



(Pires de Lima et al., 2018)

Augmenting the training data through rotation


Original image


Flipped horizontally


Flipped vertically

(Pires de Lima et al., 2018)

Augmenting the training data through rotation

Original image
Rotated -5°

Flipped horizontally
Rotated -5°

Flipped vertically
Rotated -5°

(Rafael Pires de Lima et al., 2018)

Augmenting the training data through rotation

Original image
Rotated +5°

Flipped horizontally
Rotated +5°

Flipped vertically
Rotated +5°

(Rafael Pires de Lima et al., 2018)

A Bedded skeletal peloidal packstone-grainstone sample image from the core not used in the CNN training.

Lithofacies	Probability
Bedded skeletal peloidal packstone-grainstone	**0.82**
Chert breccia in greenish shale matrix	0.15
Spiculitic mudstone-wackestone	0.02
Splotchy packstone-grainstone	0.01

(Rafael Pires de Lima et al., 2018)

A spiculitic mudstone-wackestone sample image from the core not used in the CNN training.

Lithofacies	Probability
Bedded skeletal peloidal packstone-grainstone	0.01
Chert breccia in greenish shale matrix	0.02
Spiculitic mudstone-wackestone	**0.89**
Splotchy packstone-grainstone	0.08

(Rafael Pires de Lima et al., 2018)

A splotchy packstone-grainstone sample image from the core not used in the CNN training.

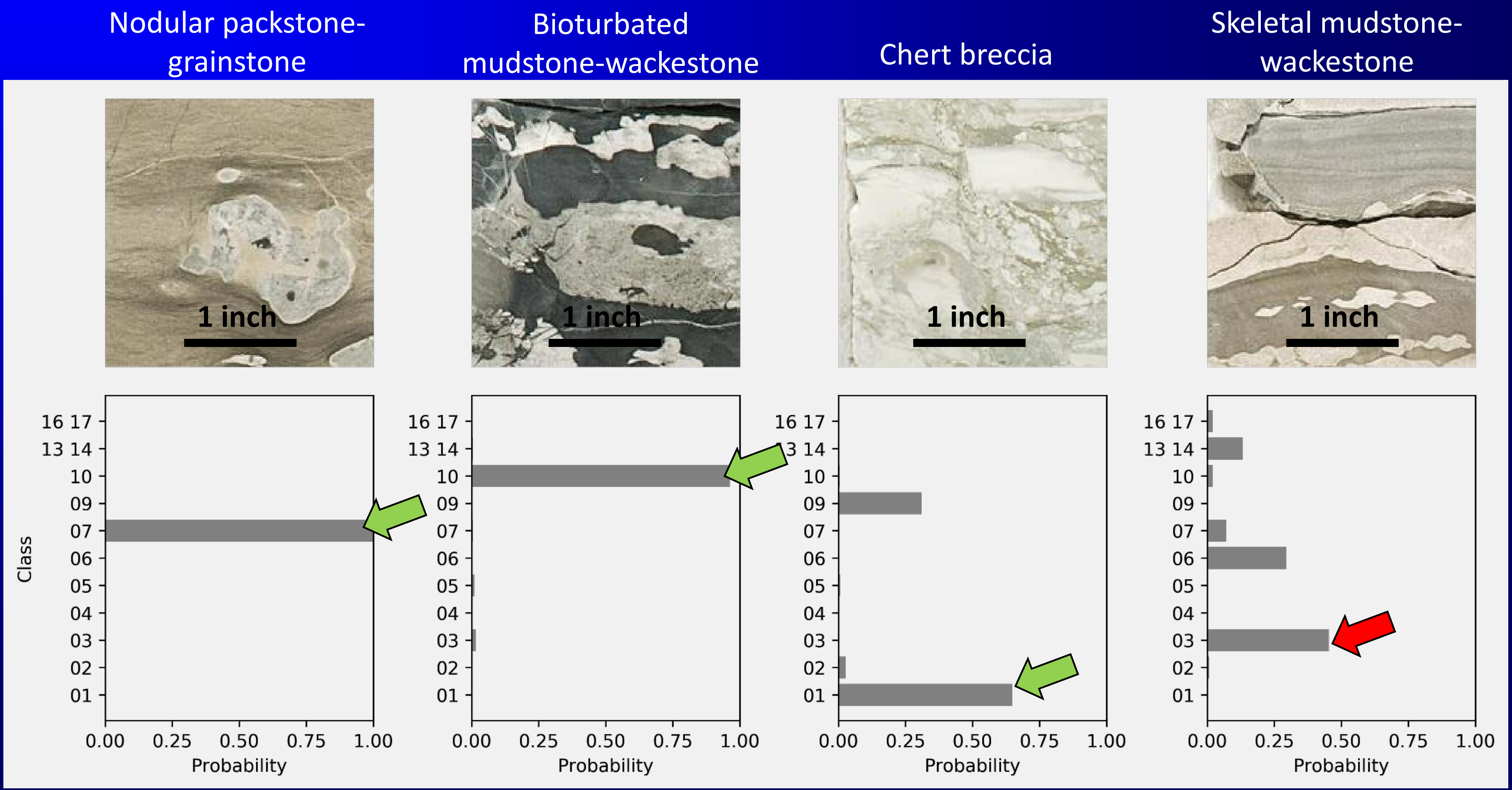
Lithofacies	Probability
Bedded skeletal peloidal packstone-grainstone	0.00
Chert breccia in greenish shale matrix	0.07
Spiculitic mudstone-wackestone	0.02
Splotchy packstone-grainstone	**0.91**

(Rafael Pires de Lima et al., 2018)

Training and test (validation) data

<table>
<tr><th>Class</th><th>Lithofacies</th><th>Training set</th><th>Test set</th></tr>
<tr><td>01</td><td>Chert breccia in greenish shale matrix</td><td>*218</td><td>3</td></tr>
<tr><td>02</td><td>Chert breccia</td><td>*236</td><td>3</td></tr>
<tr><td>03</td><td>Skeletal mudstone-wackestone</td><td>*258</td><td>4</td></tr>
<tr><td>04</td><td>Skeletal grainstone</td><td>*160</td><td>3</td></tr>
<tr><td>05</td><td>Splotchy packstone grainstone</td><td>*344</td><td>4</td></tr>
<tr><td>06</td><td>Bedded skeletal peloidal packstone-grainstone</td><td>*416</td><td>4</td></tr>
<tr><td>07</td><td>Nodular packstone-grainstone</td><td>445</td><td>11</td></tr>
<tr><td>08</td><td>Skeletal peloidal packstone-grainstone</td><td>not used</td><td>not used</td></tr>
<tr><td>09</td><td>Bioturbated skeletal peloidal packstone-grainstone</td><td>795</td><td>19</td></tr>
<tr><td>10</td><td>Bioturbated mudstone-wackestone</td><td>*150</td><td>4</td></tr>
<tr><td>11</td><td>Brecciated spiculitic mudstone</td><td>not used</td><td>not used</td></tr>
<tr><td>12</td><td>Intraclast spiculitic mudstone</td><td>not used</td><td>not used</td></tr>
<tr><td>13</td><td>Spiculitic mudstone-wackestone</td><td rowspan="2">3077</td><td rowspan="2">79</td></tr>
<tr><td>14</td><td>Argillaceous spiculitic mudstone-wackestone</td></tr>
<tr><td>15</td><td>Glauconitic sandstone</td><td>not used</td><td>not used</td></tr>
<tr><td>16</td><td>Shale</td><td rowspan="2">789</td><td rowspan="2">17</td></tr>
<tr><td>17</td><td>Shaly claystone</td></tr>
<tr><td colspan="2">Total number of samples in each set</td><td>6888</td><td>151</td></tr>
</table>

Examples of the classification performed by the retrained ResNetV2

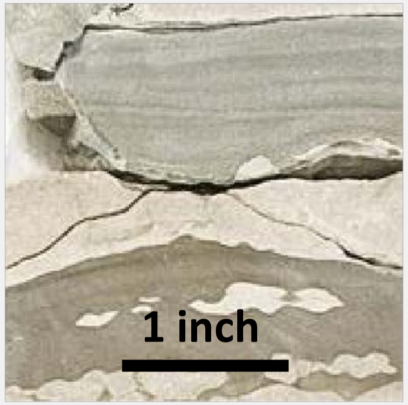


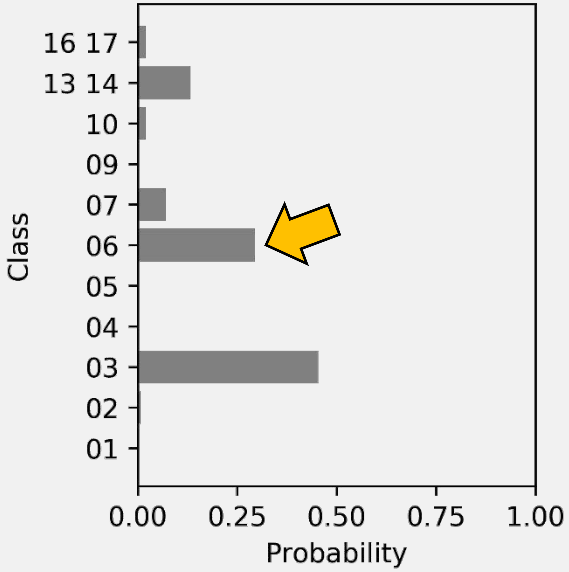
An example of misclassification

Example of Class 06
Bedded skeletal peloidal packstone-grainstone

Example of Class 03
Skeletal mudstone-wackestone
Training Example

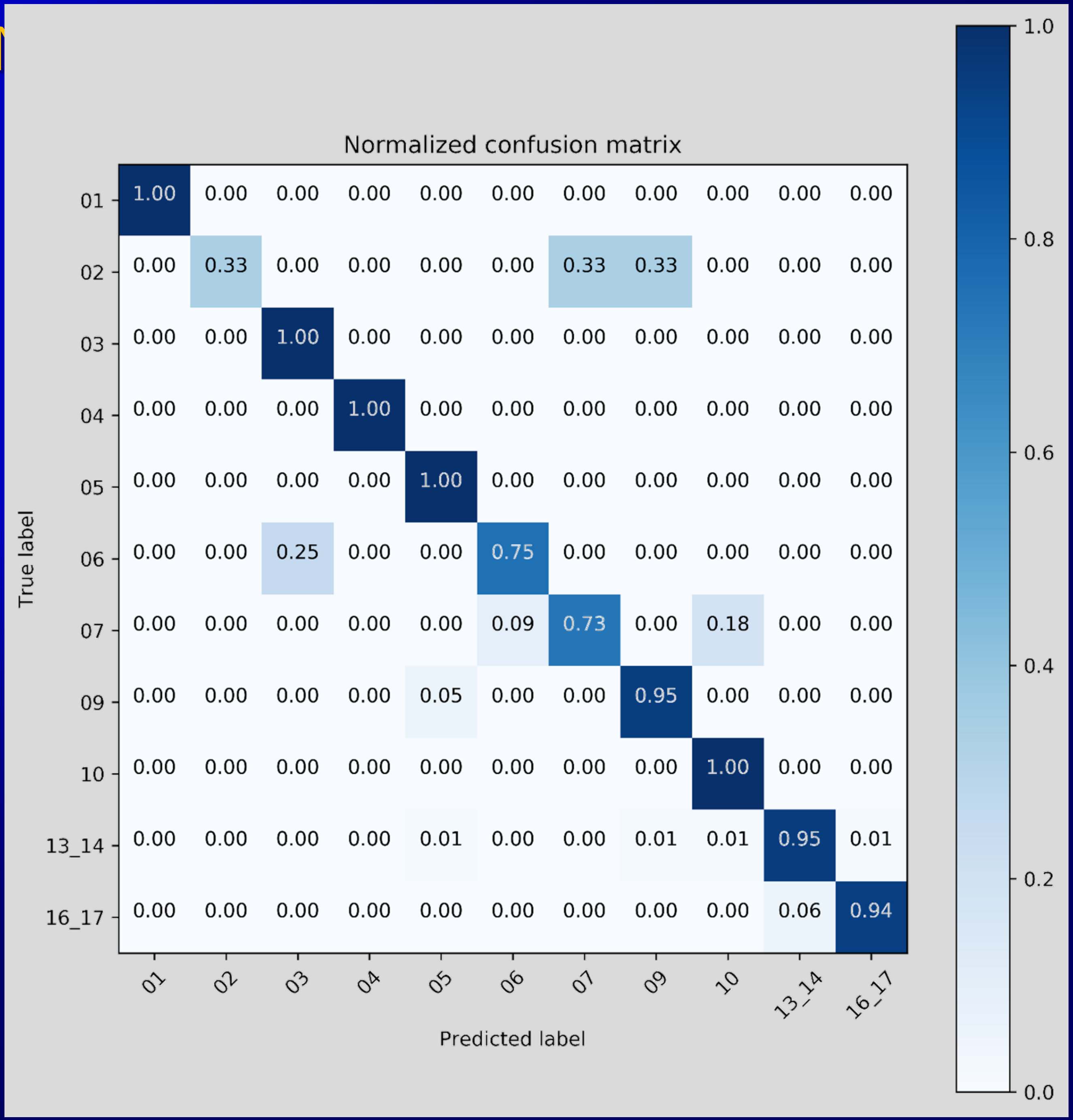
Skeletal mudstone-wackestone



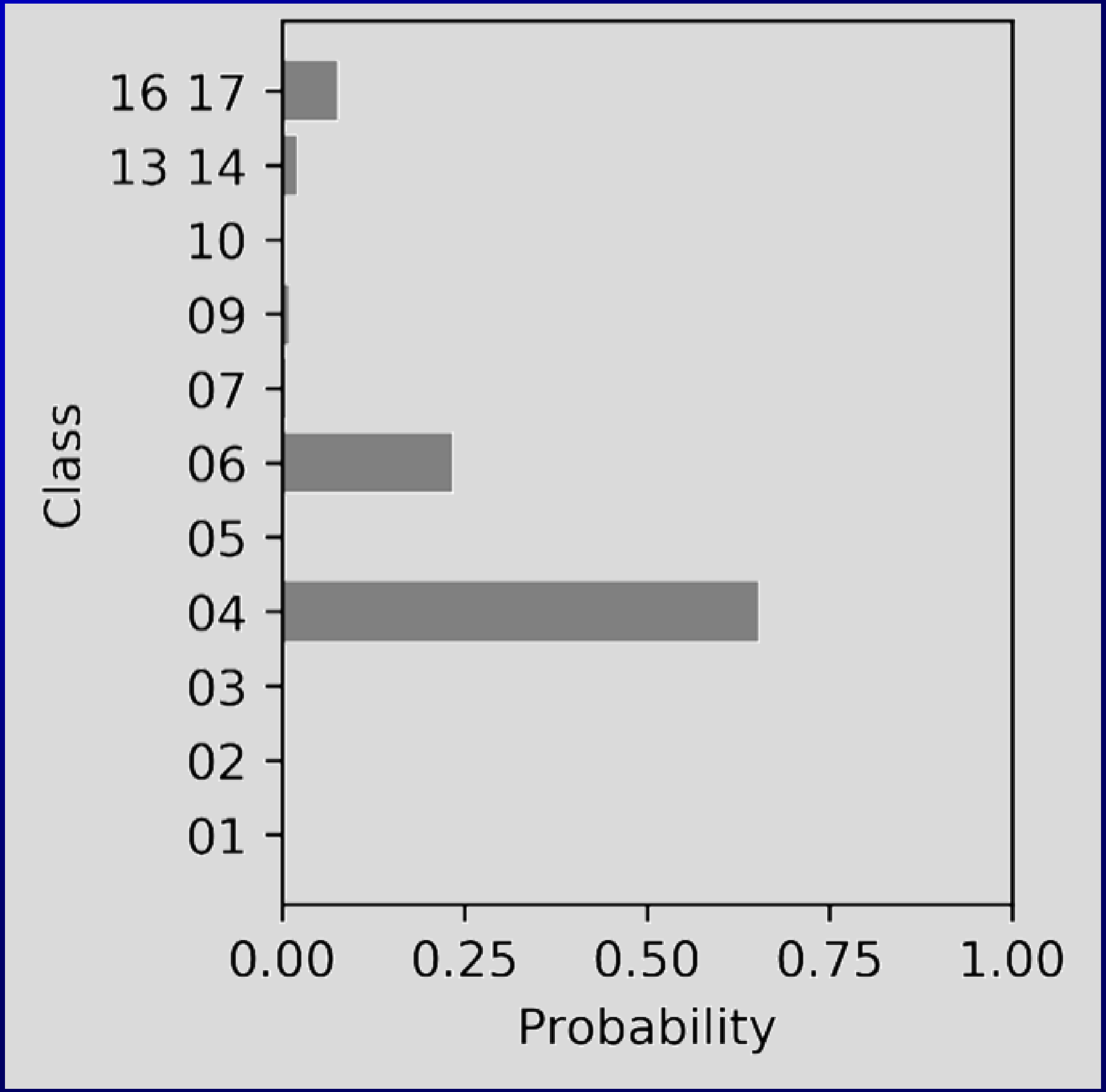


Confusion matrix of retrained CN

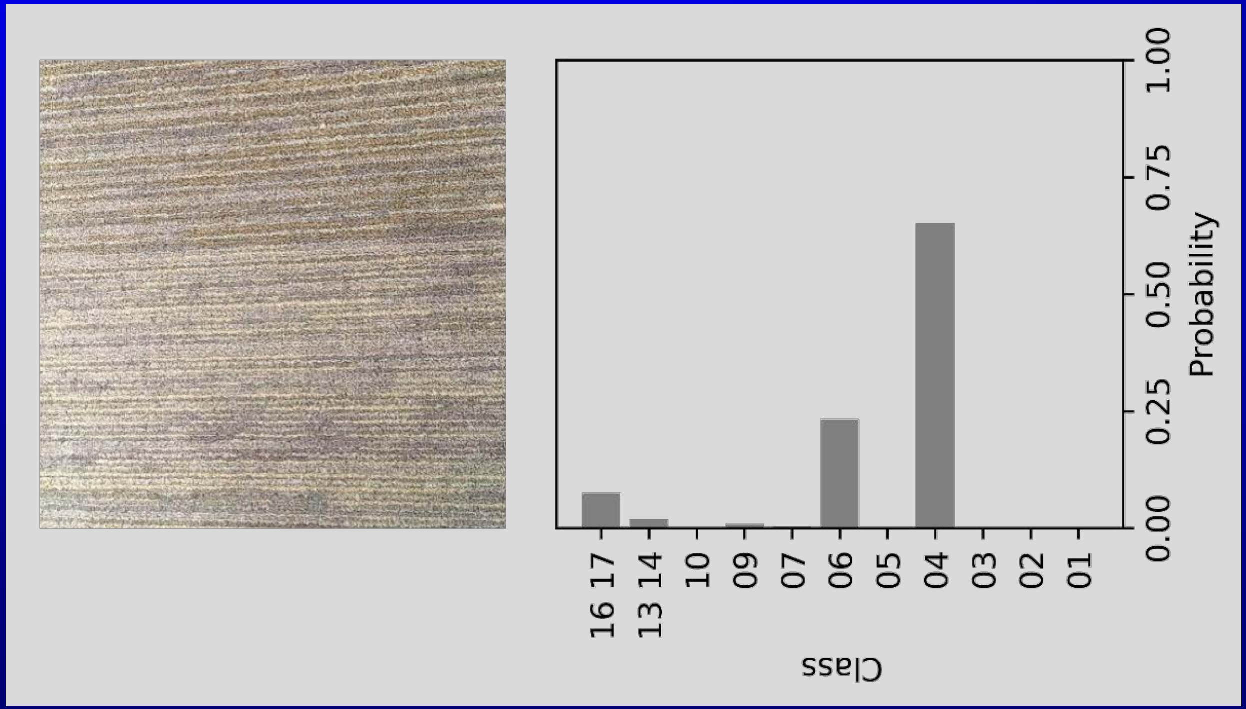
Very good concordance between labels provided by the expert and class assigned by the CNN



Pitfalls: Can you identify this rock?



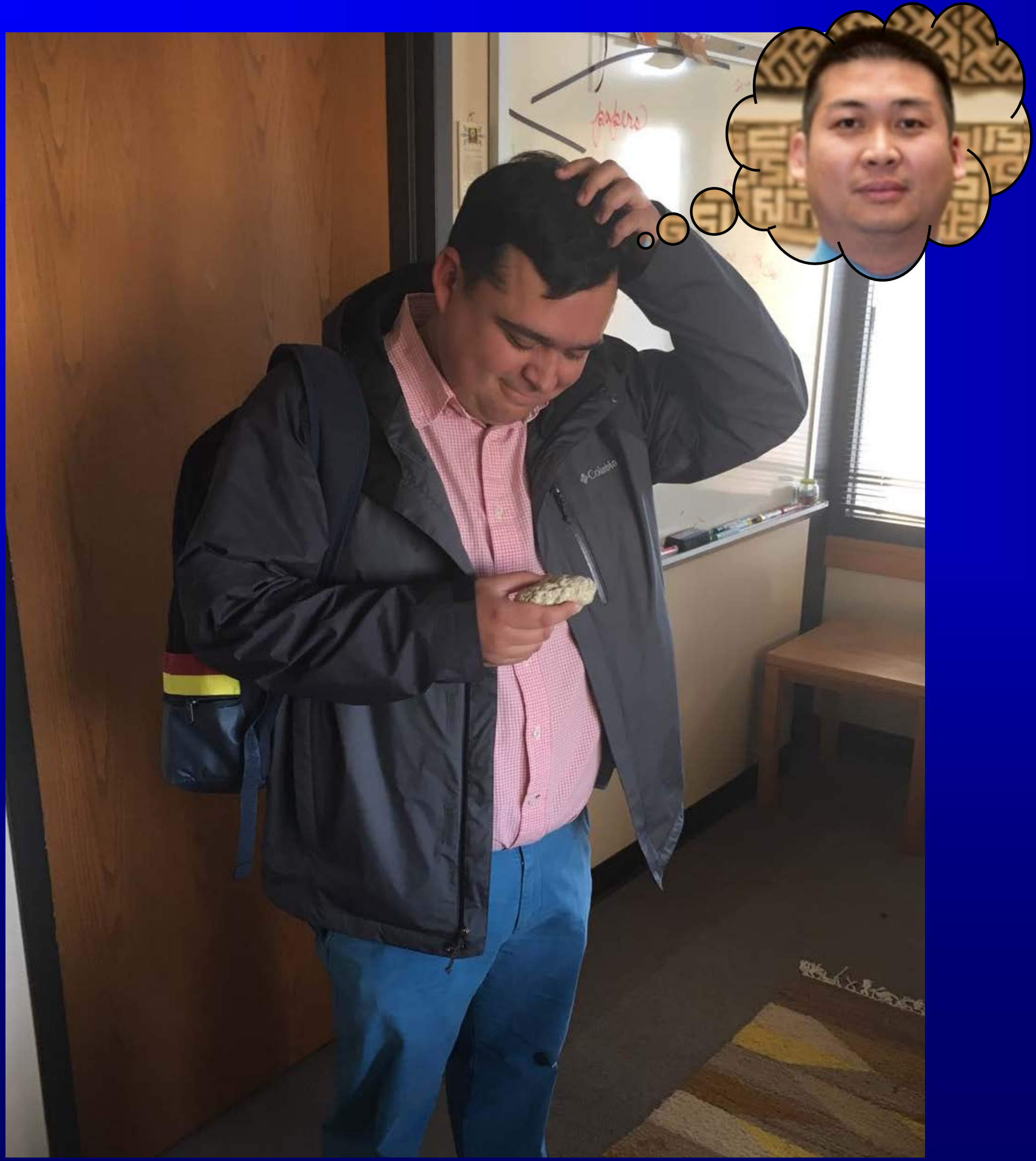
Carpet classification and examples of class 04



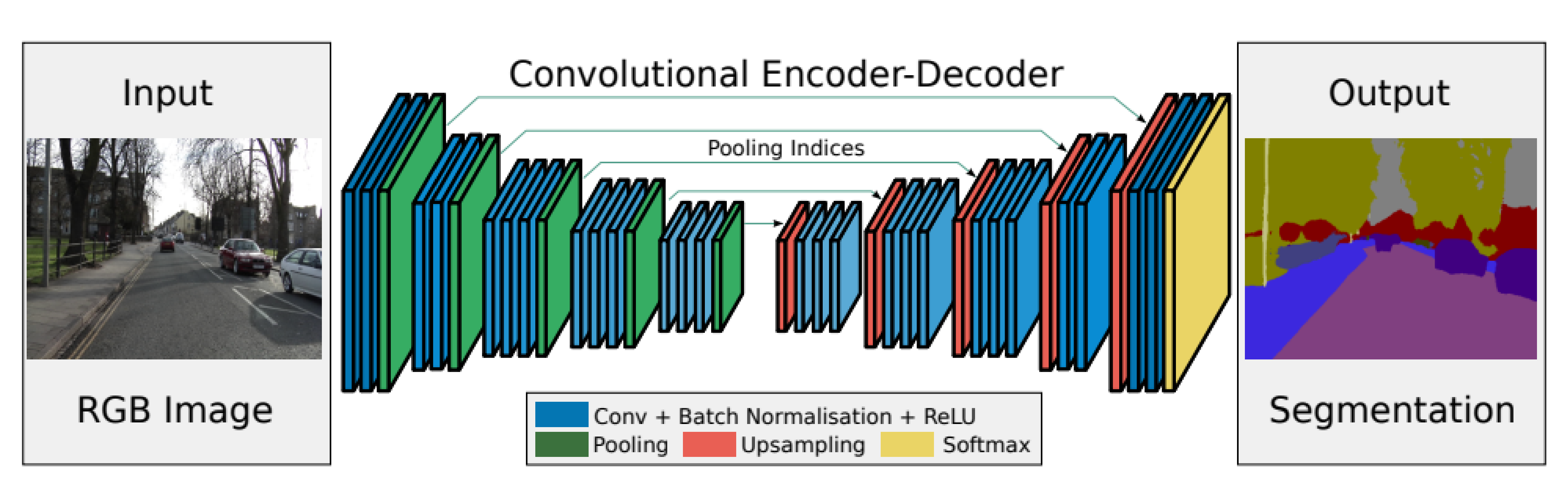
a

b

Next steps: Construct a geologist confusion matrix

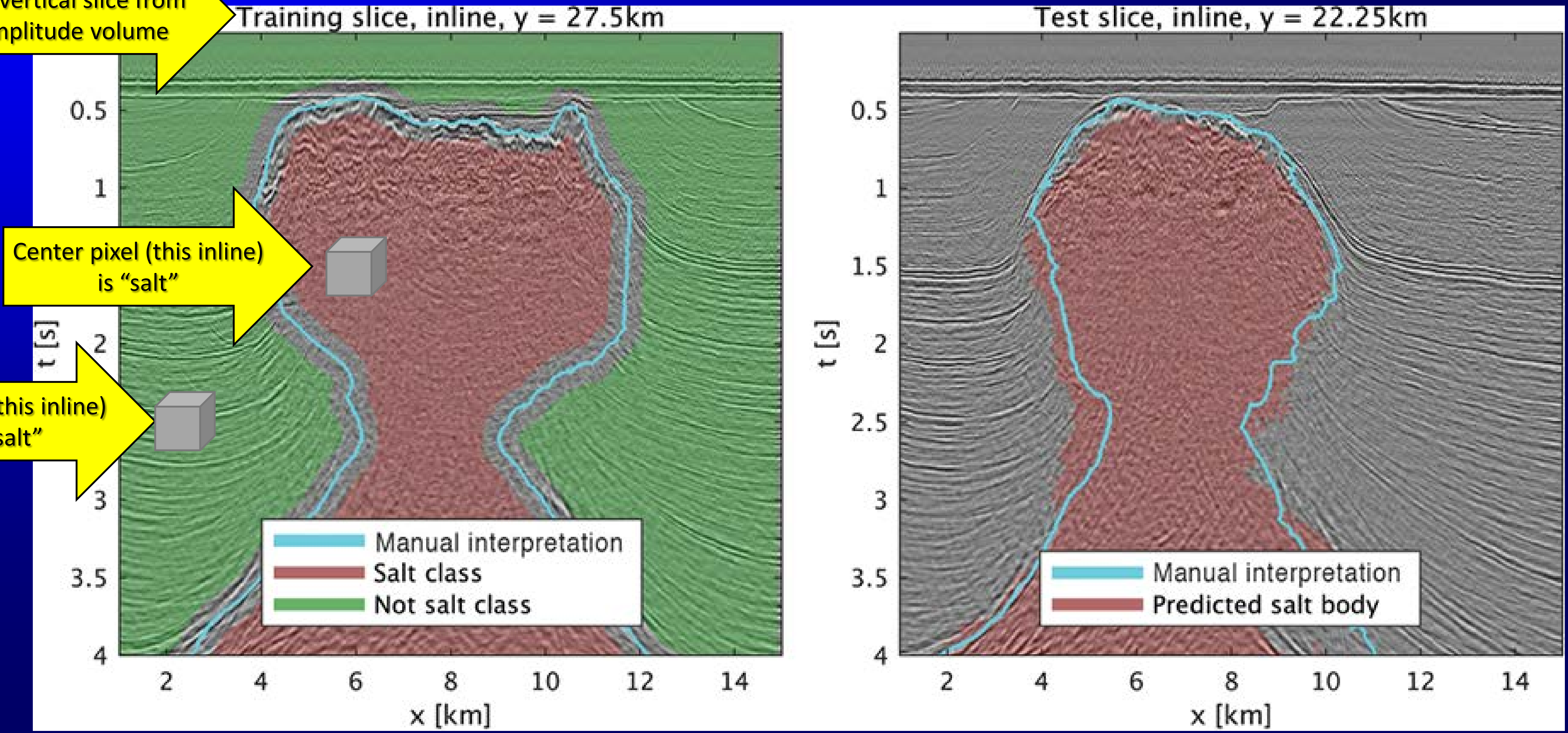


What about seismic amplitude patterns?



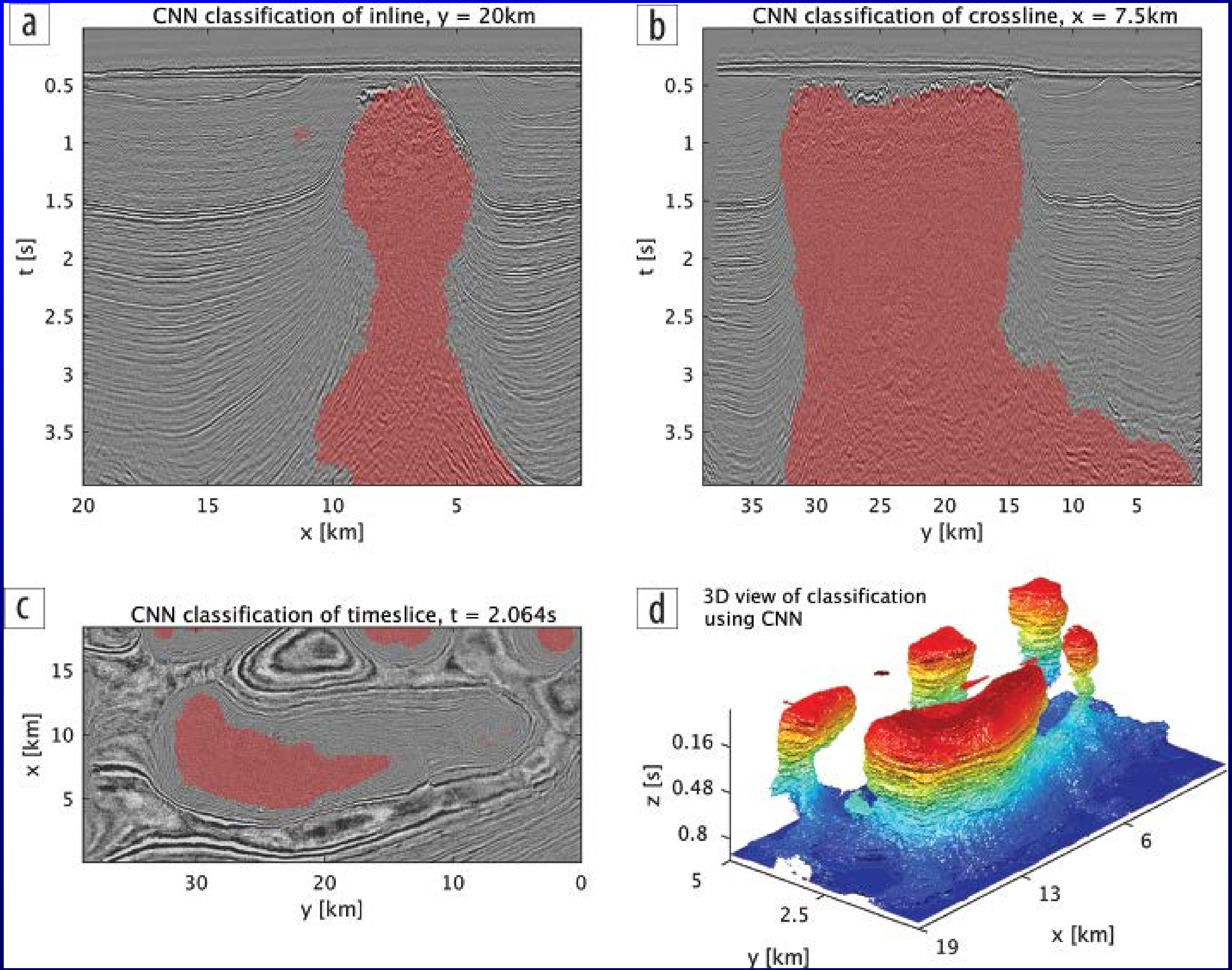
Salt segmentation – training and test slices

Training is one vertical slice from 3D seismic amplitude volume



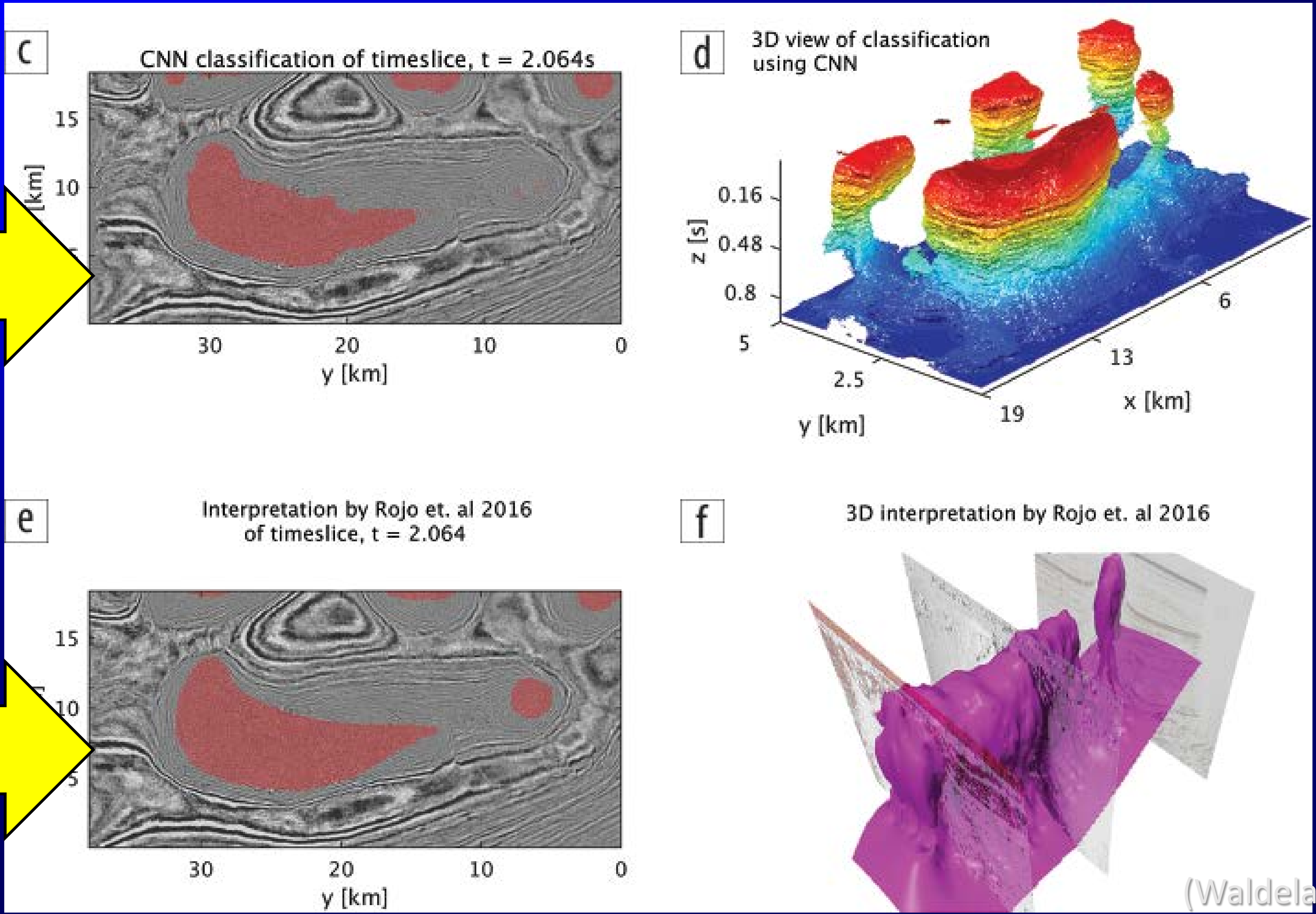
(Waldeland et al., 2018)

Salt segmentation – training and test slices



(Waldeland et al., 2018)

Salt is relatively insensitive to nonstationary source wavelet



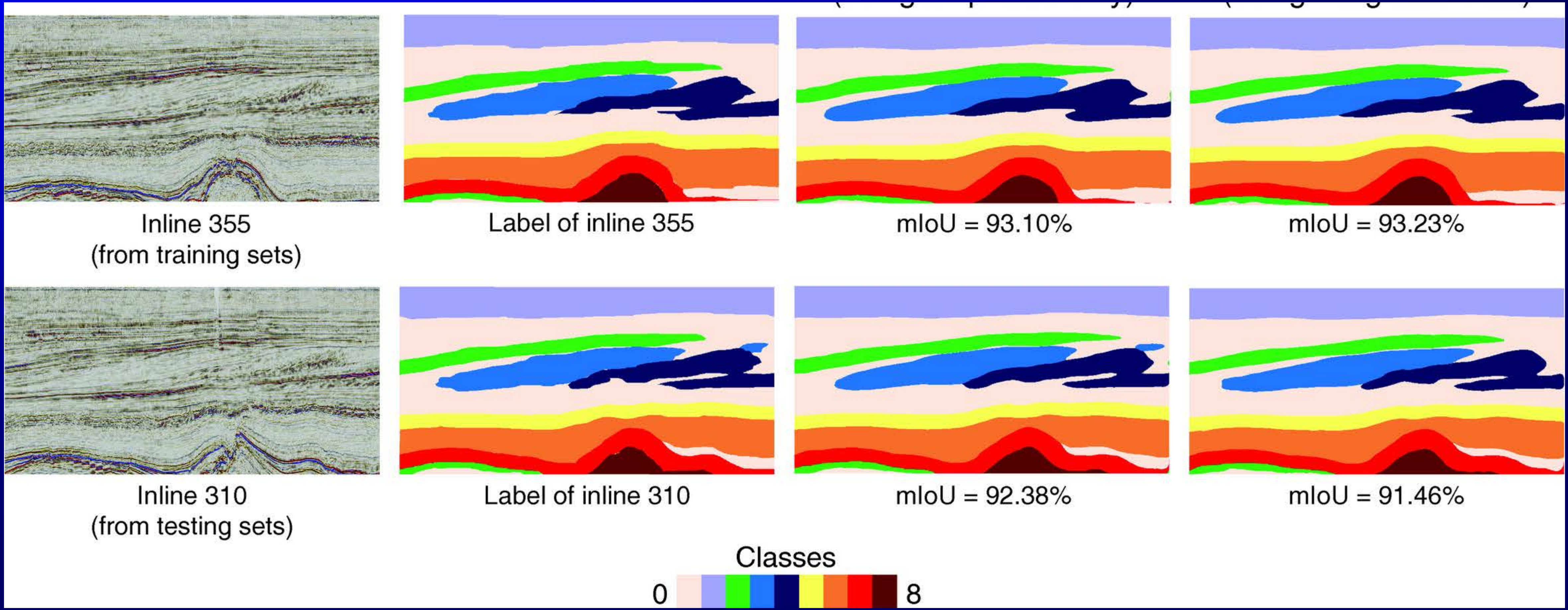
CNN results

Human interpretation

(Waldeland et al., 2018)

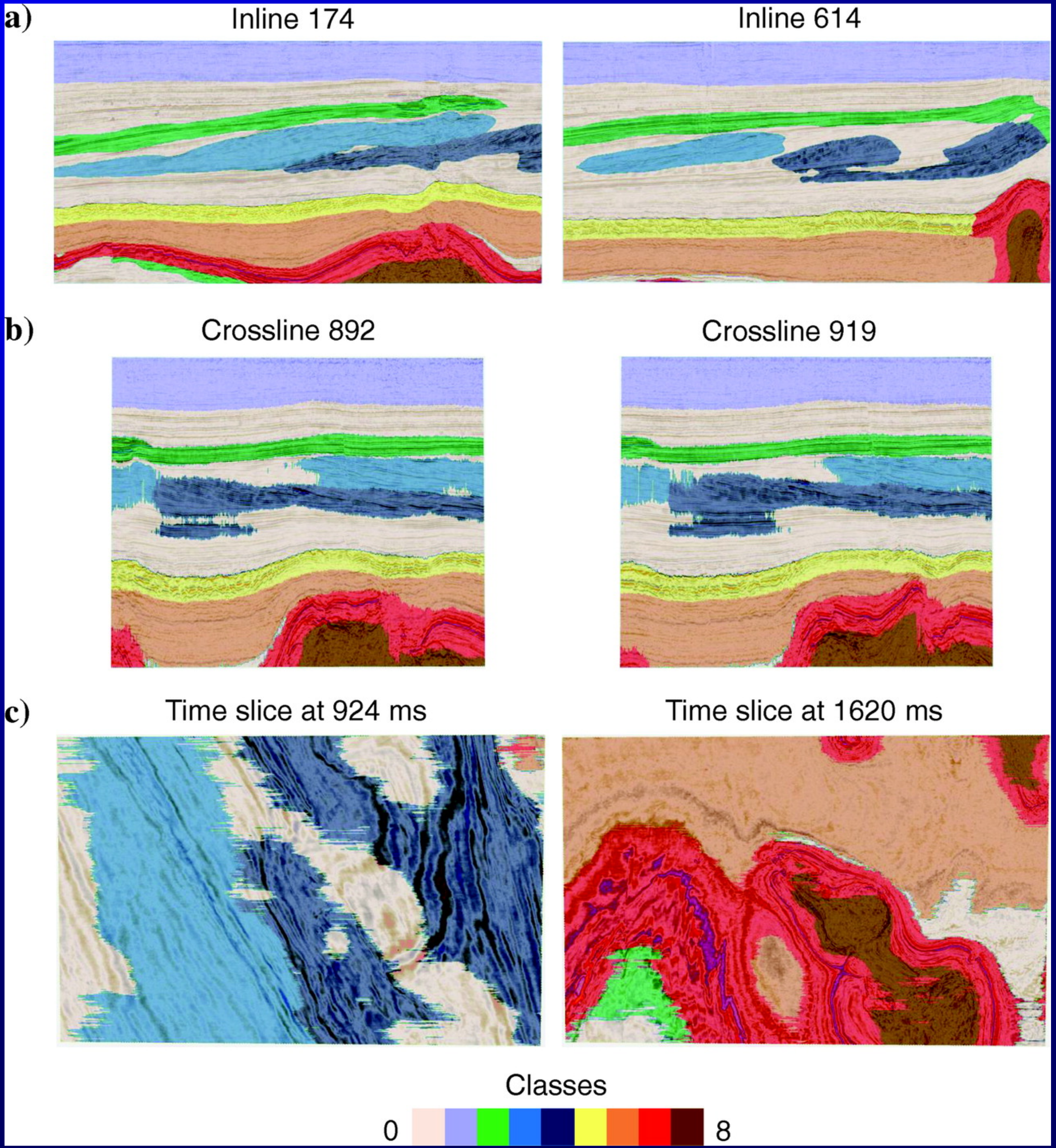
Progress in seismic facies classification (F3 data volume)

Amplitude | Labels | Prediction near training (Amplitude only) | Prediction near training (Amplitude, RMS, Chaos)



(Zhang et al., 2021)

Progress in seismic facies classification (F3 data volume)



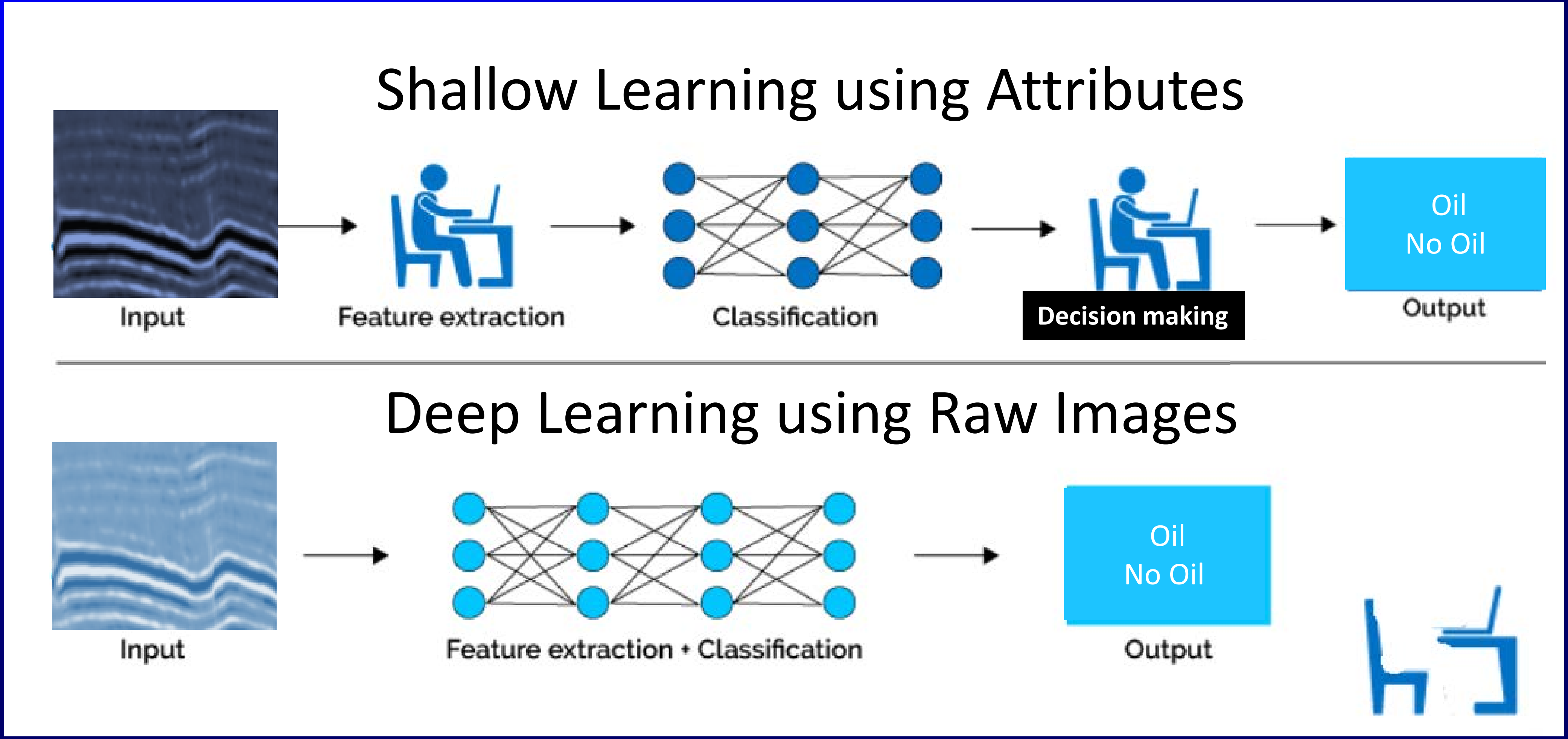
Prediction far from training

(Zhang et al., 2021)

Conclusions

- CNN is not magic
  - The convolutions used are similar to those used in PhotoShop and in seismic attribute analysis
  - “Pooling” is not unlike blocking and smoothing used in well log analysis
  - The activations are the same kind of “thresholds” interpreters use in predicting lithofacies from gamma ray logs (e.g. sandstone, mudstone, shale)
- CNN is currently the best option to perform 2D image classification
- CNN is not perfect; there will be misclassifications (OK for fossils, but not for cars!)
- Training data are tedious to generate; it is common to use a pretrained CNN and simply apply it to their data
- CNN will *not* replace the expert interpreter
  - Interpreters will be needed to train the network
  - Interpreters will be able to analyze vastly greater amounts of data
  - Interpreters will be needed to quality control the results

Traditional shallow learning vs. deep learning



We will be needed!

We will be replaced!

(modified from Seif, 2018)