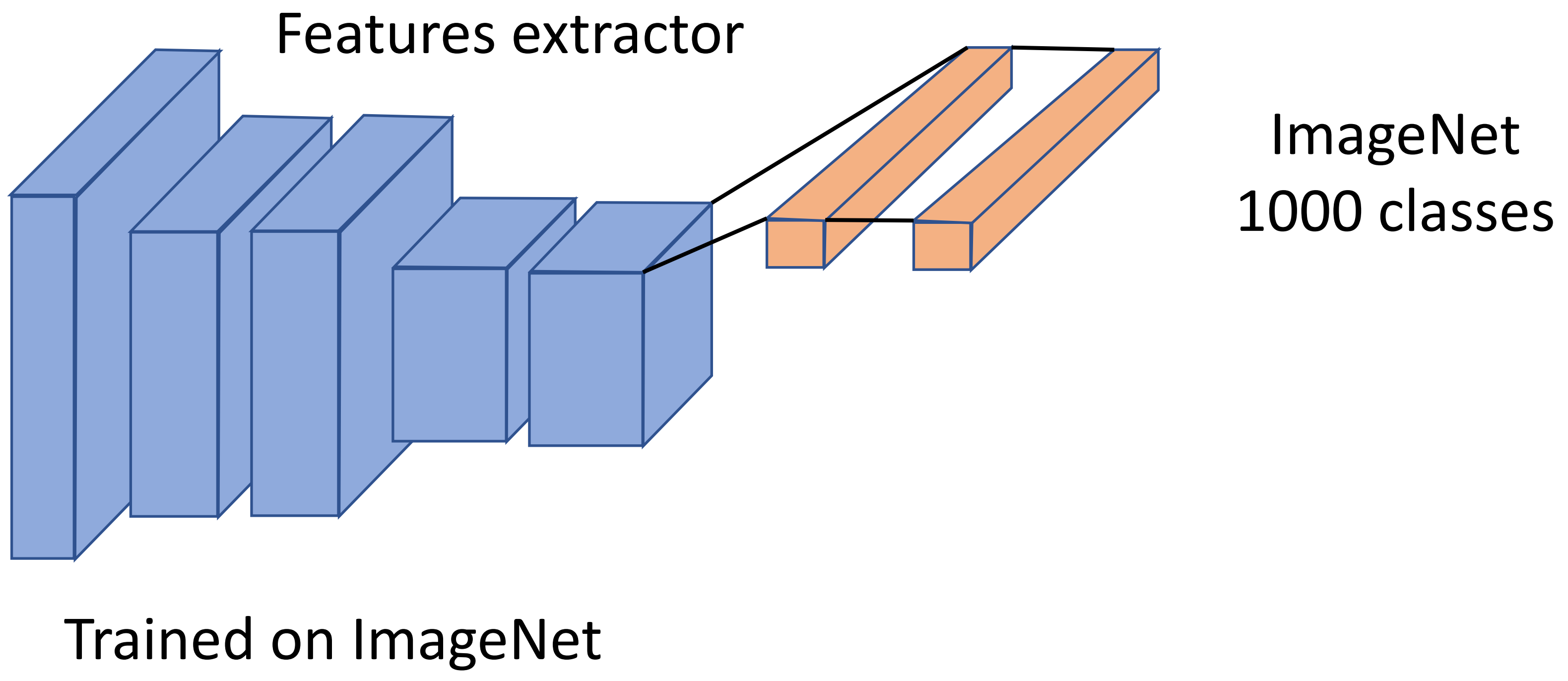
MML minor #7

Нейросети: transfer learning, другие CNN задачи и автокодировщики

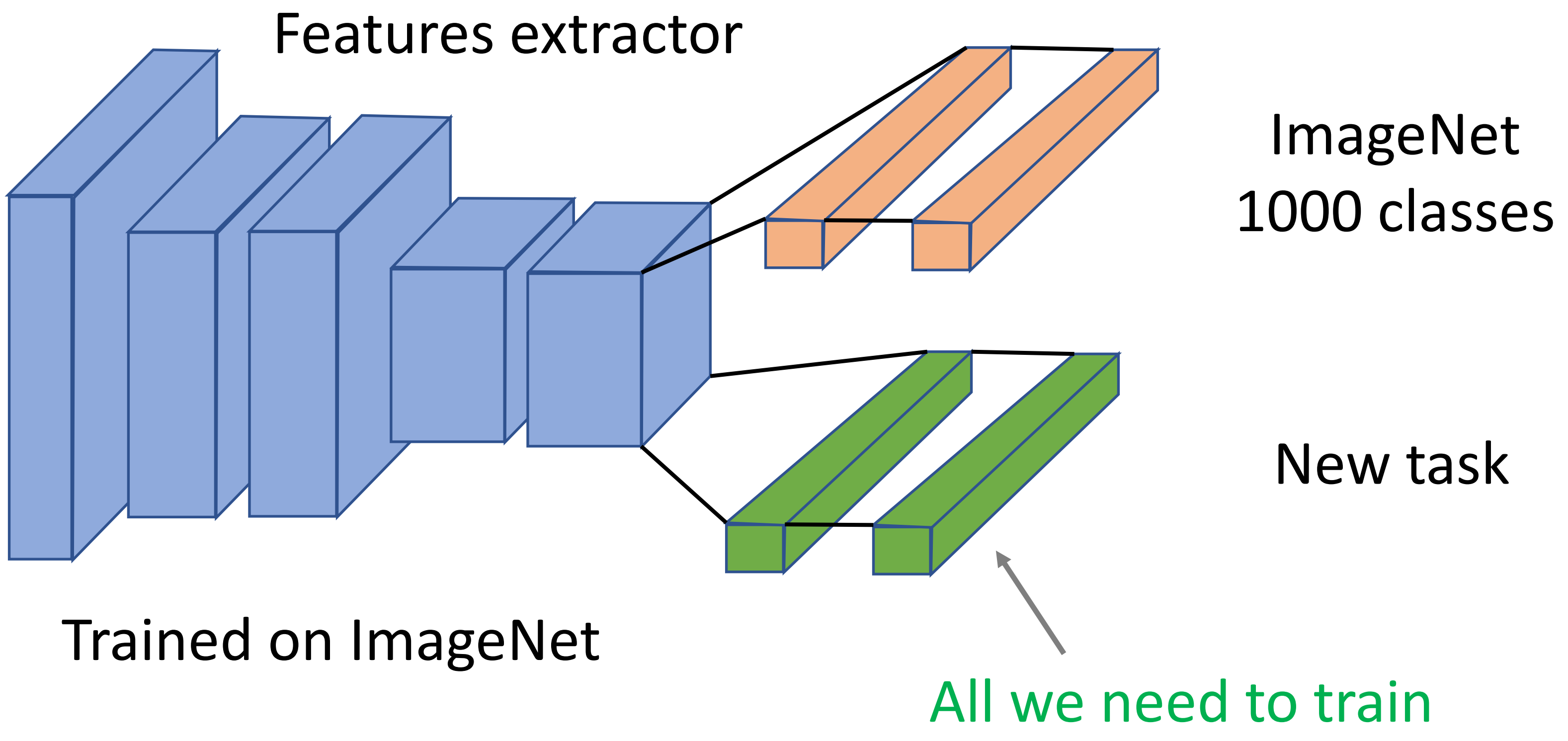
Transfer learning

- Deep networks learn complex features extractor, but we need lots of data to train it from scratch!
- What if we can reuse an existing features extractor for a new task?



Transfer learning

- Deep networks learn complex features extractor, but we need lots of data to train it from scratch!
- What if we can reuse an existing features extractor for a new task?



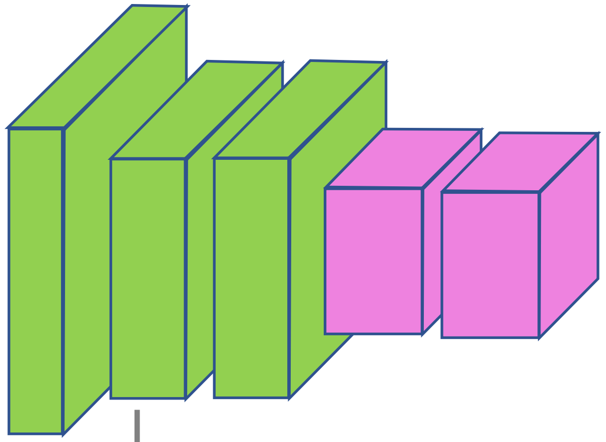
Transfer learning

- You need less data to train (for training only final MLP)
- It works if a domain of a new task is similar to ImageNet’s
- Won’t work for human emotions classification,
  ImageNet doesn’t have people faces in the dataset!

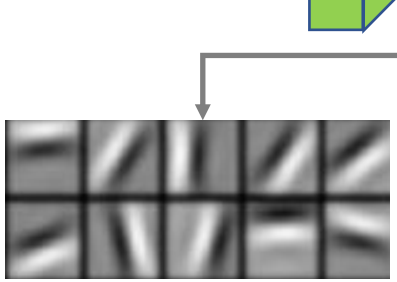
Transfer learning

- But what if we need to classify human emotions?
- Maybe we can partially reuse ImageNet features extractor?

Extractor
we want:



Activation
stimuli:



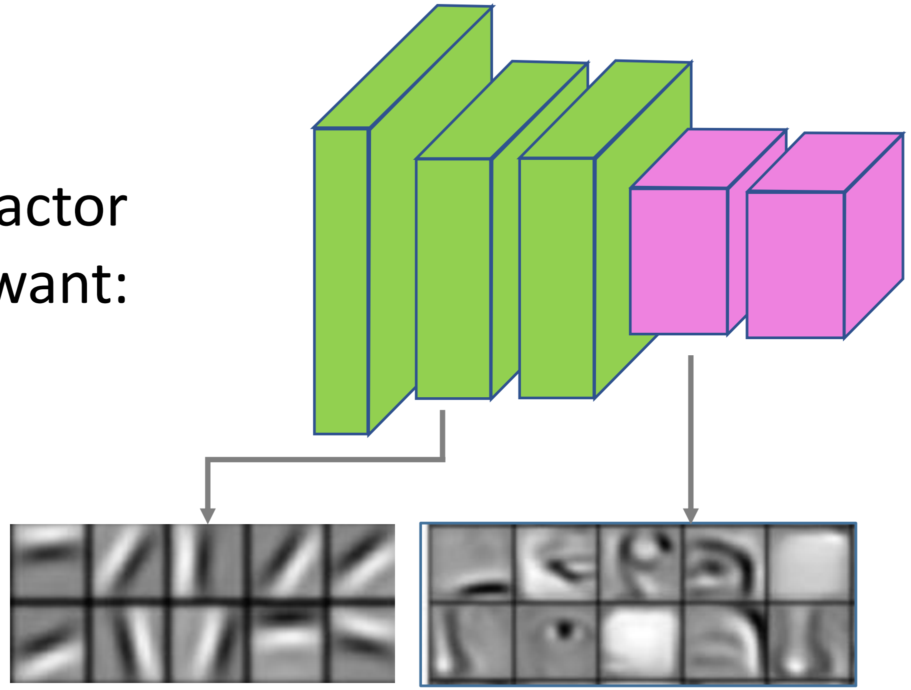
http://web.eecs.umich.edu/~honglak/icml09-ConvolutionalDeepBeliefNetworks.pdf

Transfer learning

- But what if we need to classify human emotions?
- Maybe we can partially reuse ImageNet features extractor?

Extractor we want:

Activation stimuli:

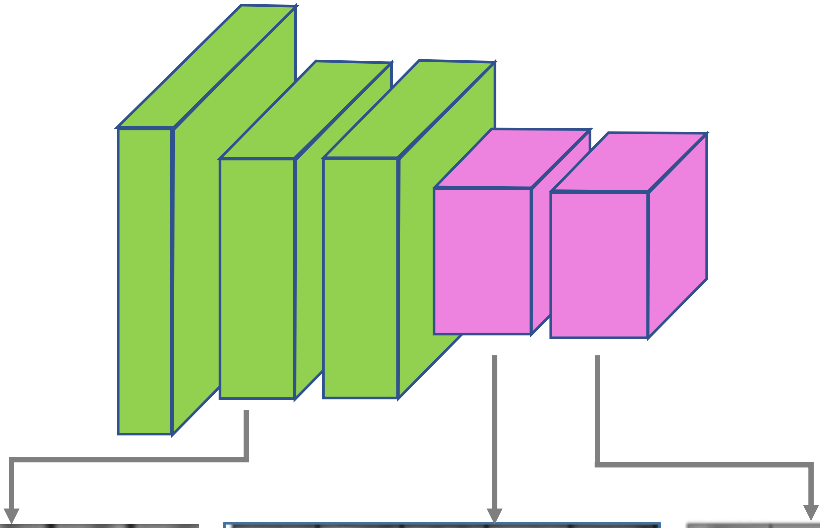


http://web.eecs.umich.edu/~honglak/icml09-ConvolutionalDeepBeliefNetworks.pdf

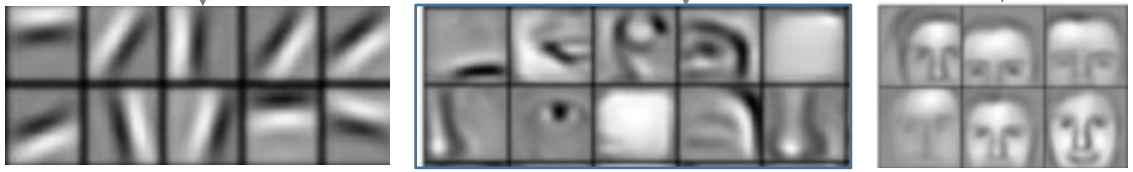
Transfer learning

- But what if we need to classify human emotions?
- Maybe we can partially reuse ImageNet features extractor?

Extractor
we want:



Activation
stimuli:



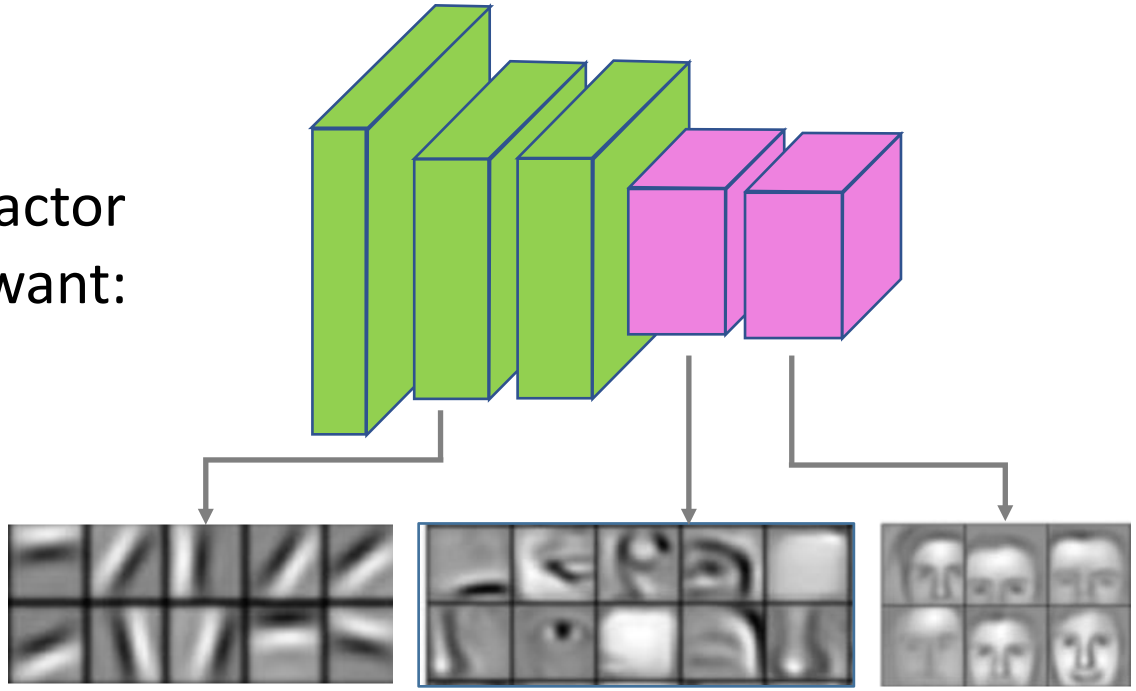
Honglak Lee, http://web.eecs.umich.edu/~honglak/icml09-ConvolutionalDeepBeliefNetworks.pdf

Transfer learning

- But what if we need to classify human emotions?
- Maybe we can partially reuse ImageNet features extractor?

Extractor
we want:

Activation
stimuli:

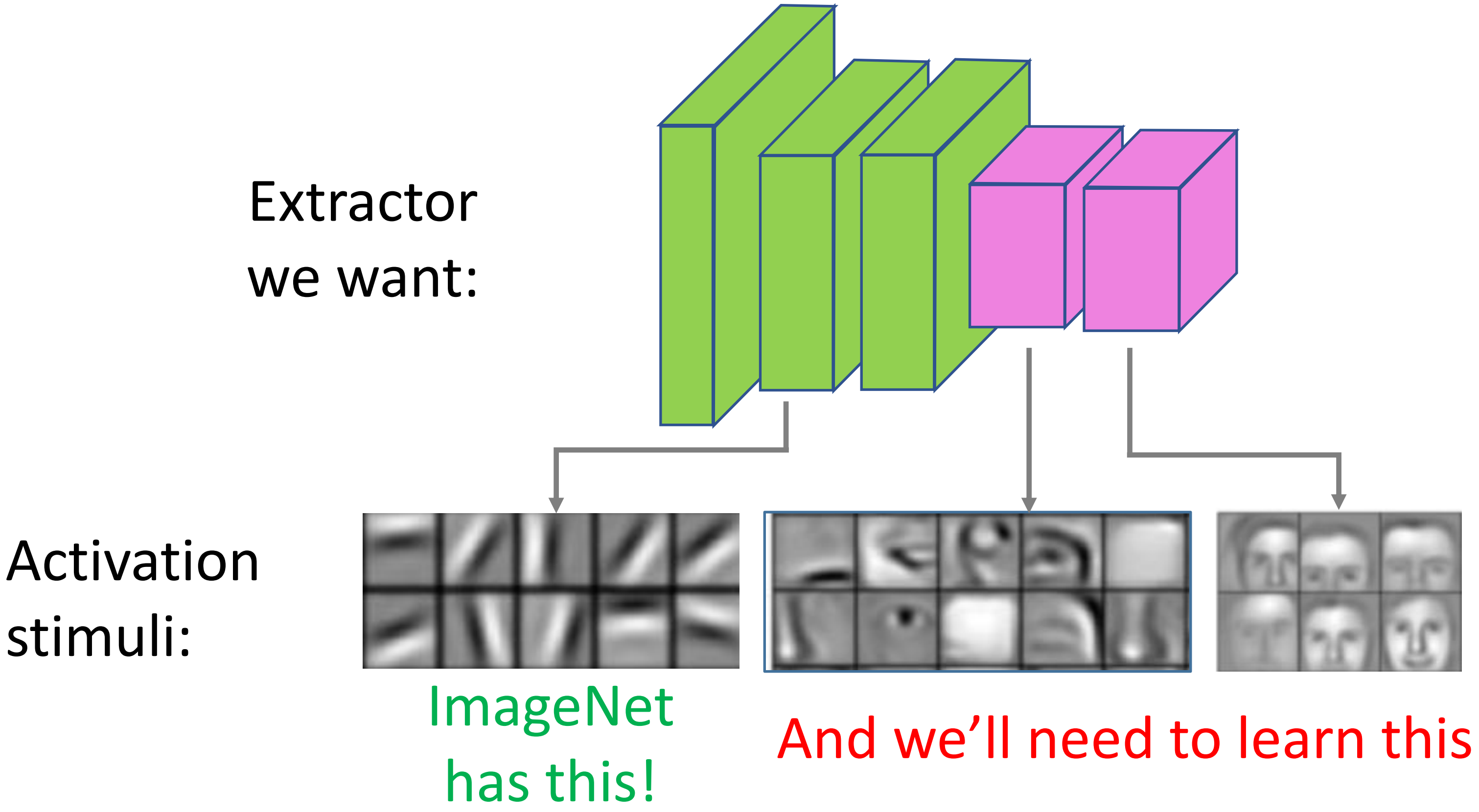


ImageNet
has this!

Honglak Lee, http://web.eecs.umich.edu/~honglak/icml09-ConvolutionalDeepBeliefNetworks.pdf

Transfer learning

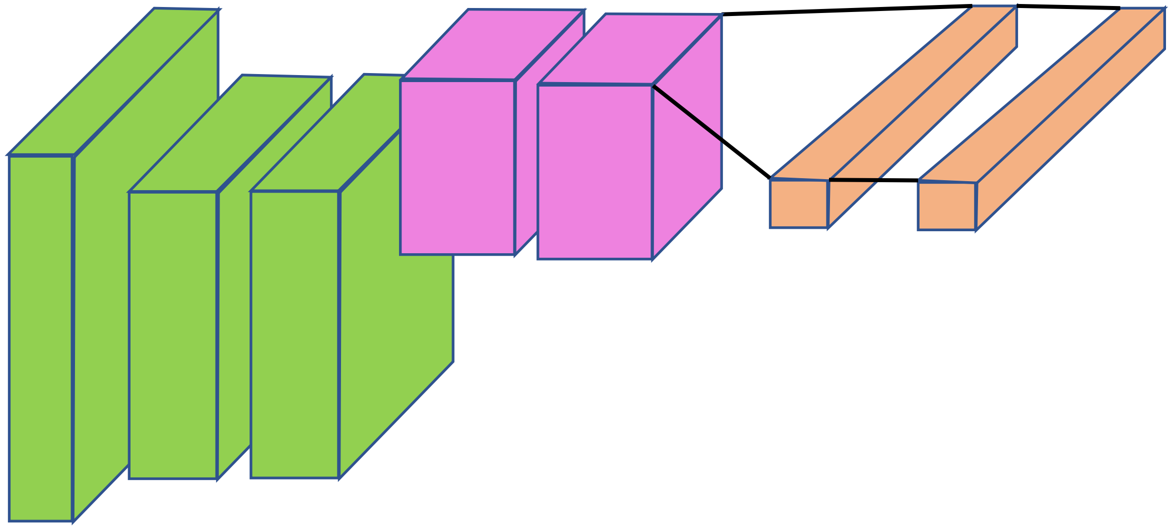
- But what if we need to classify human emotions?
- Maybe we can partially reuse ImageNet features extractor?



Honglak Lee, http://web.eecs.umich.edu/~honglak/icml09-ConvolutionalDeepBeliefNetworks.pdf

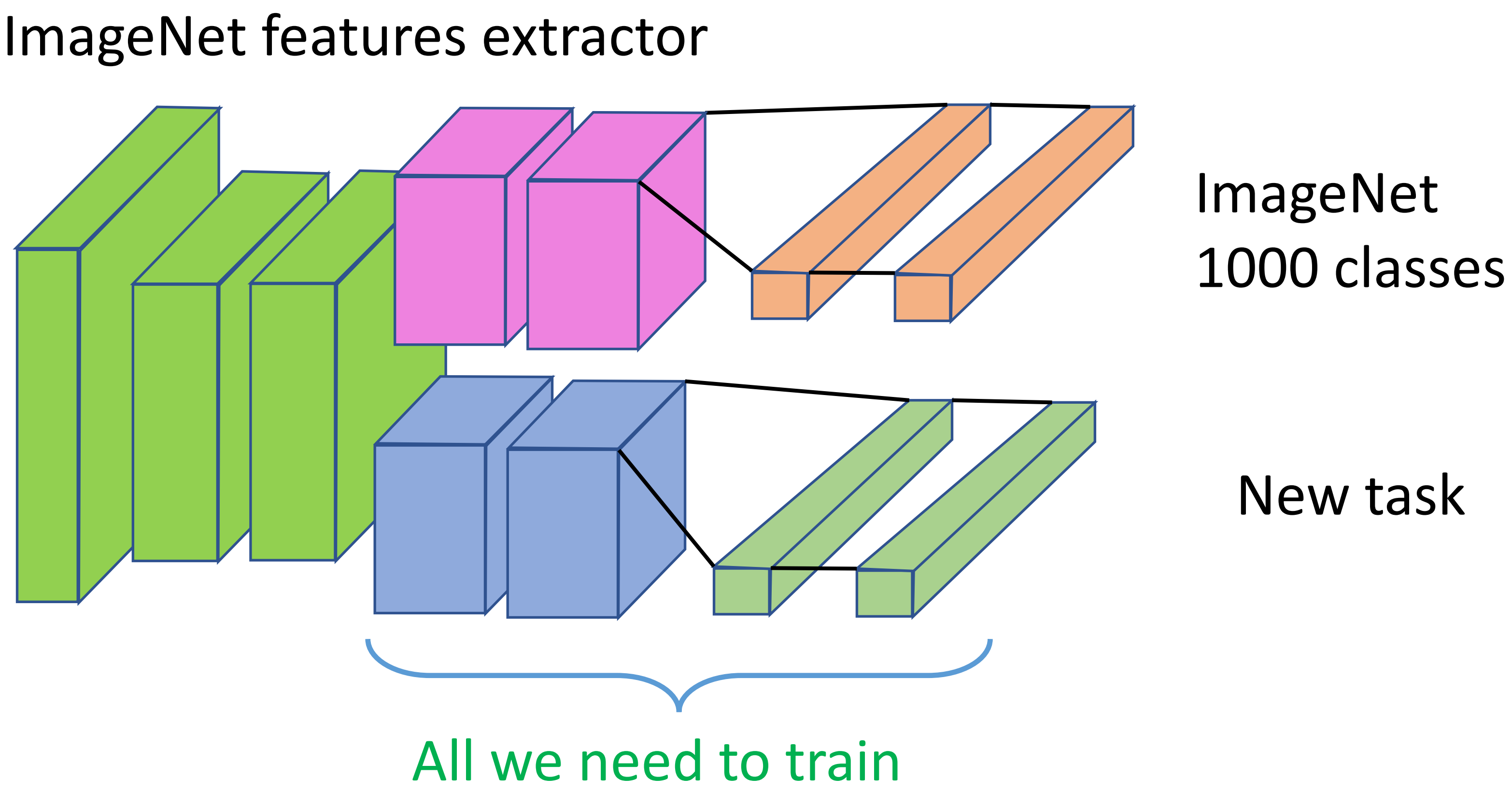
Transfer learning

ImageNet features extractor

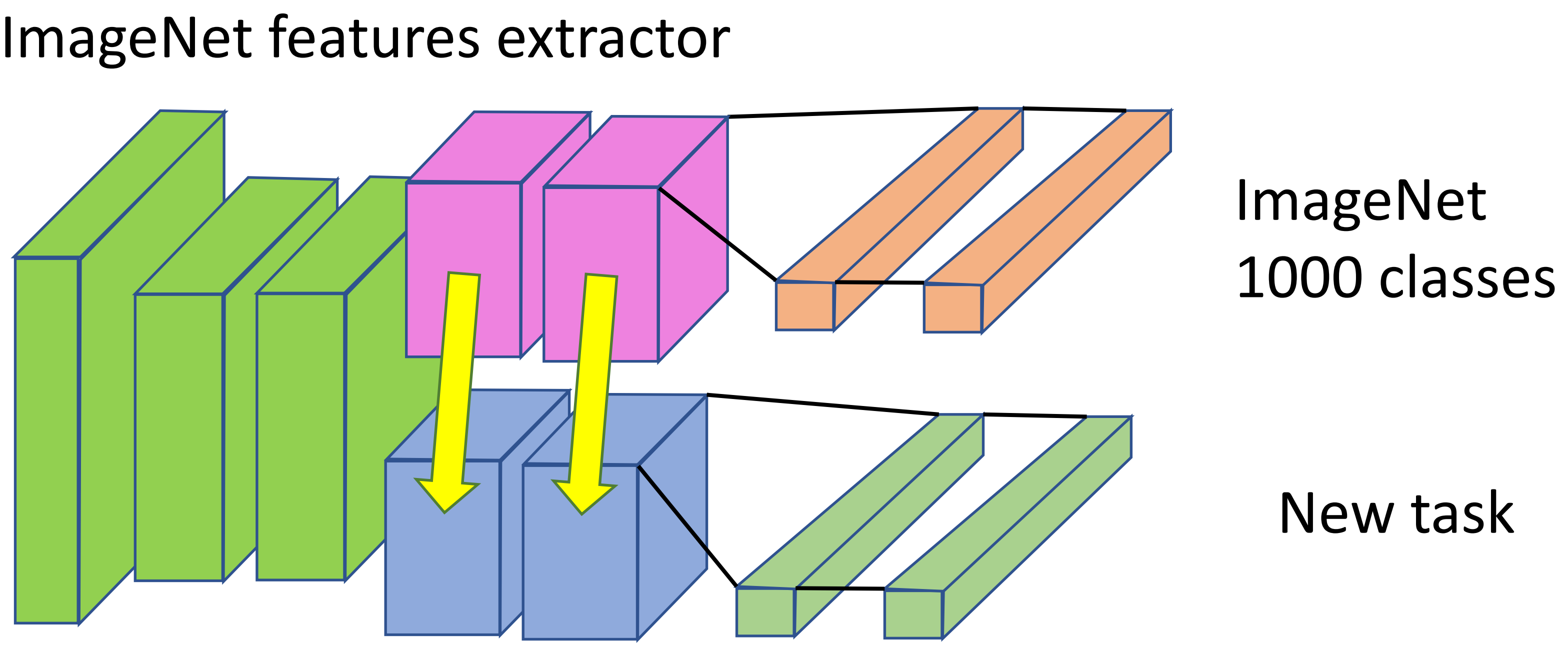


ImageNet
1000 classes

Transfer learning



Fine-tuning



- You can initialize deeper layers with values from ImageNet.
- This is called **fine-tuning**, because you don’t start with a random initialization.
- Propagate all gradients with smaller learning rate.

Fine-tuning

- Very frequently used thanks to wide spectrum of ImageNet classes
- Keras has the weights of pre-trained VGG, Inception, ResNet architectures
- You can fine-tune a bunch of different architectures and make an ensemble out of them!

Takeaways

	ImageNet domain	Not similar to ImageNet
Small dataset	Train last MLP layers	
Big dataset		

Takeaways

	ImageNet domain	Not similar to ImageNet
Small dataset	Train last MLP layers	
Big dataset	Fine-tuning of deeper layers	

Takeaways

	ImageNet domain	Not similar to ImageNet
Small dataset	Train last MLP layers	
Big dataset	Fine-tuning of deeper layers	Train from scratch

Takeaways

	ImageNet domain	Not similar to ImageNet
Small dataset	Train last MLP layers	Collect more data
Big dataset	Fine-tuning of deeper layers	Train from scratch

Other computer vision tasks

We’ve examined image classification task

Other computer vision tasks

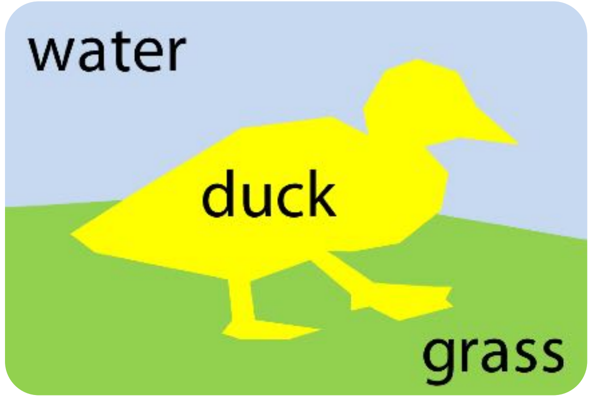
We've examined image classification task

Semantic segmentation:

Other computer vision tasks

We’ve examined image classification task

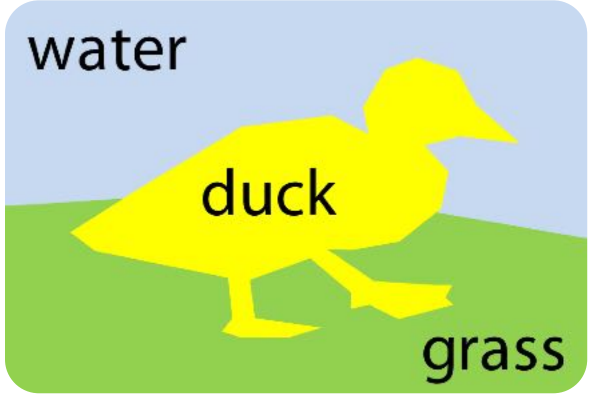
Semantic segmentation:



Other computer vision tasks

We've examined image classification task

Semantic segmentation:

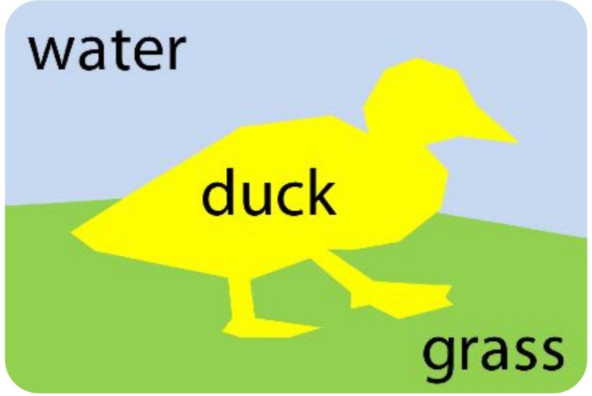


Object classification
+ localization:

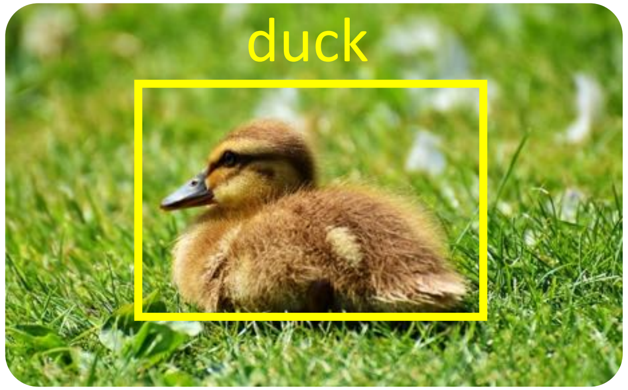
Other computer vision tasks

We've examined image classification task

Semantic segmentation:



Object classification
+ localization:

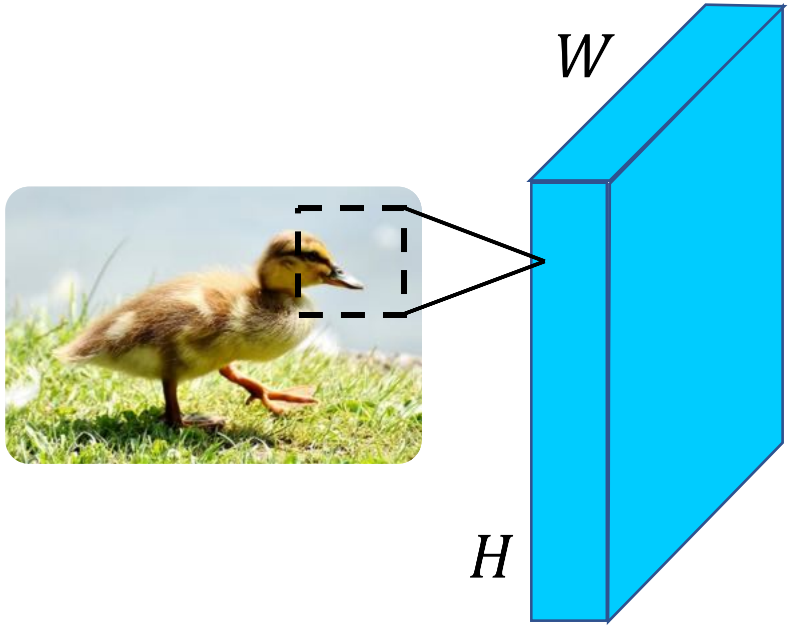


Semantic segmentation

We need to classify each pixel

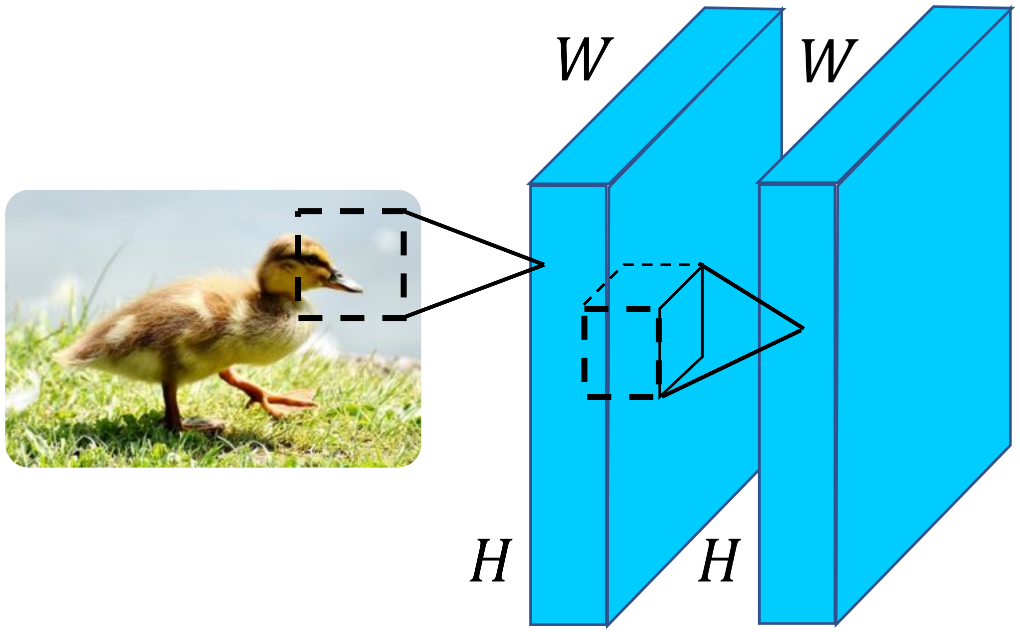
Semantic segmentation

We need to classify each pixel



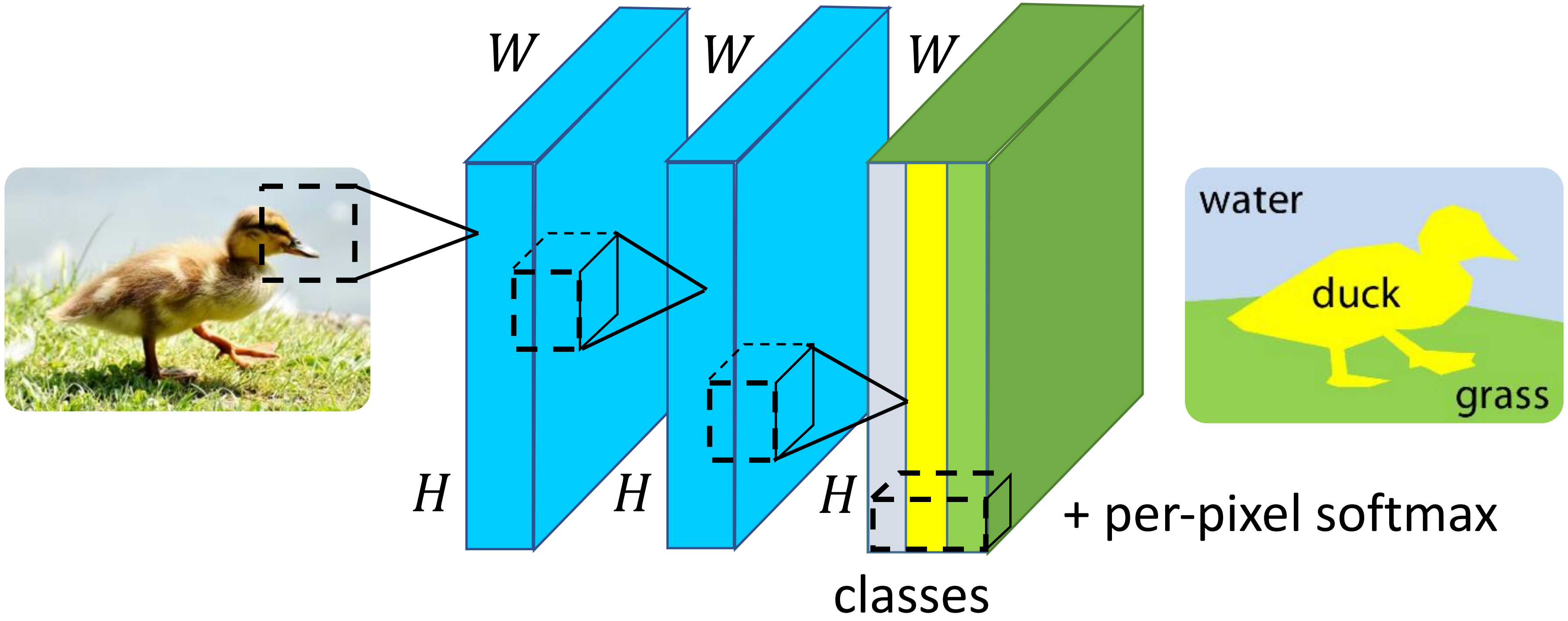
Semantic segmentation

We need to classify each pixel



Semantic segmentation

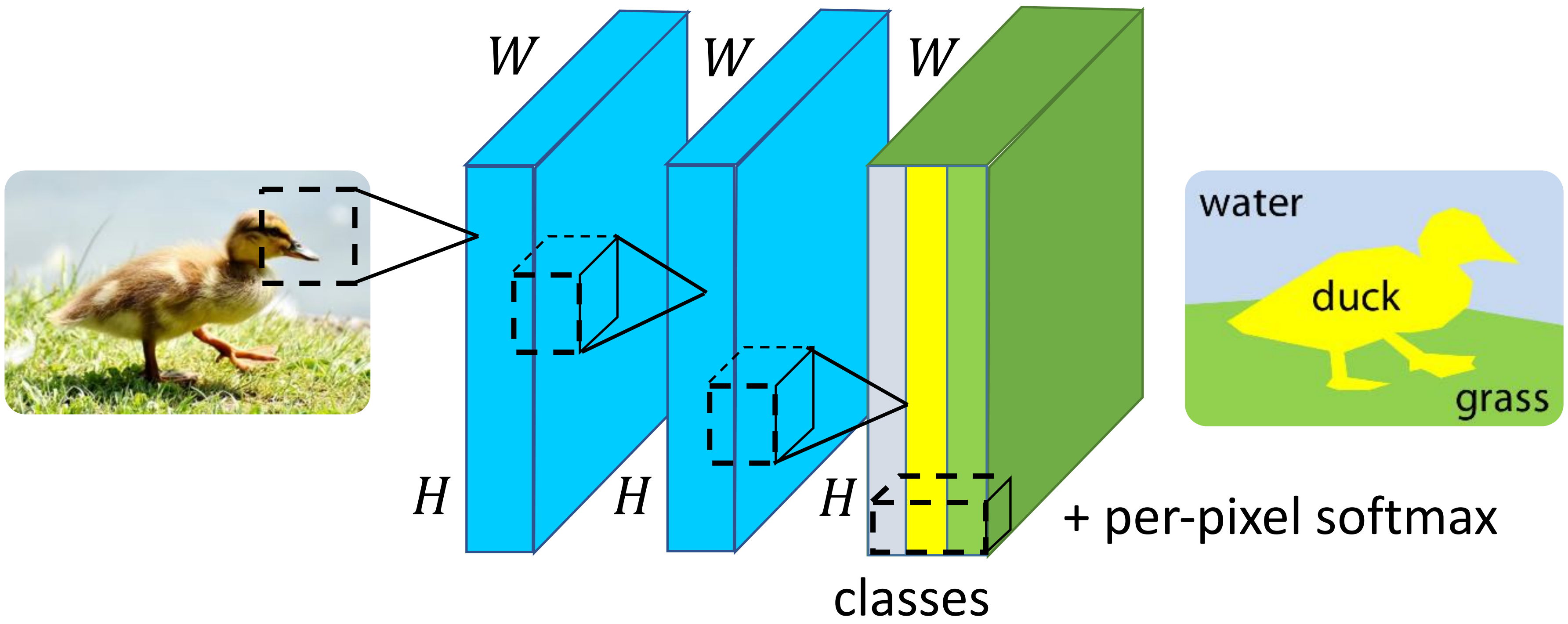
We need to classify each pixel



Naïve approach: stack convolutional layers
and add per-pixel softmax

Semantic segmentation

We need to classify each pixel

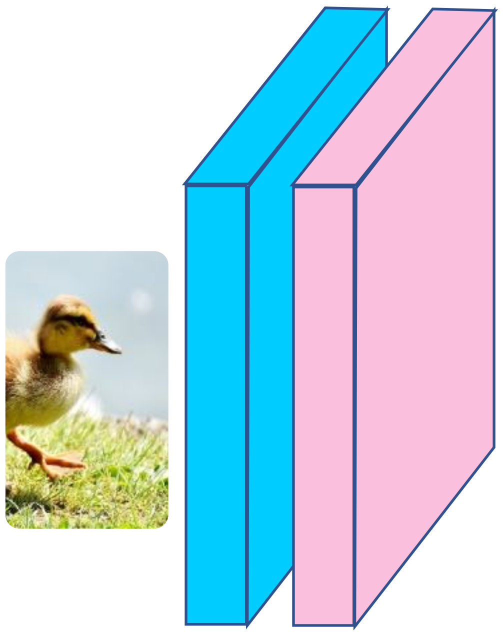


Naïve approach: stack convolutional layers
and add per-pixel softmax

We go deep but don’t add pooling, too expensive

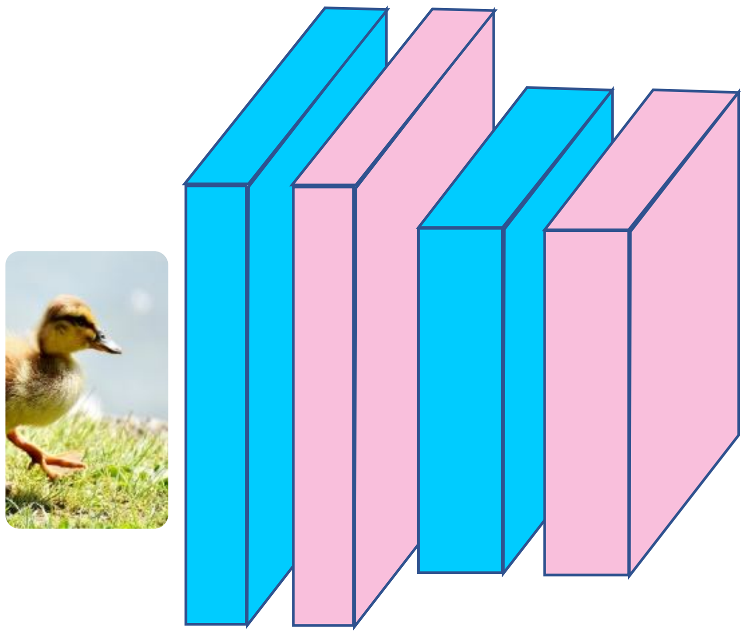
Semantic segmentation

Let's add pooling, which acts like down-sampling



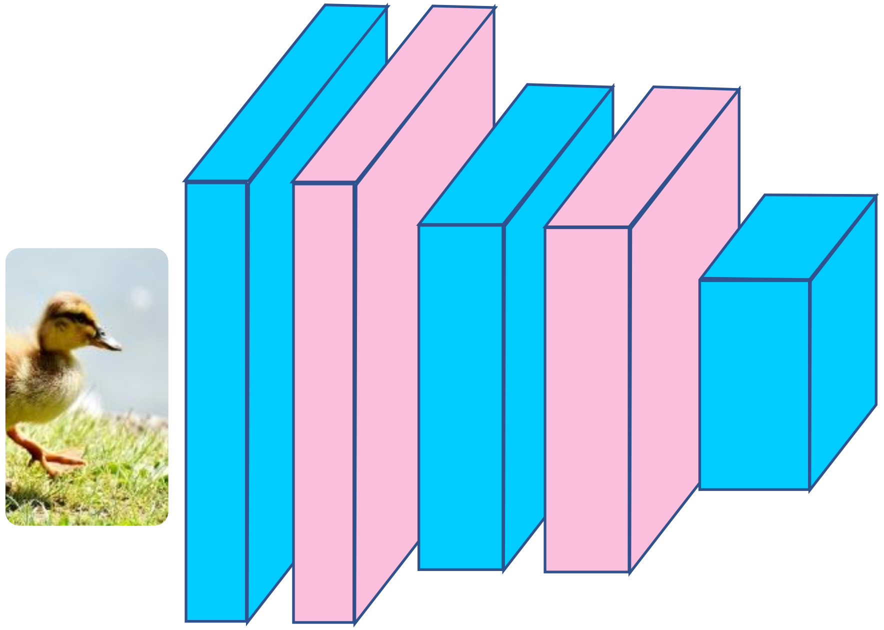
Semantic segmentation

Let’s add pooling, which acts like down-sampling



Semantic segmentation

Let’s add pooling, which acts like down-sampling

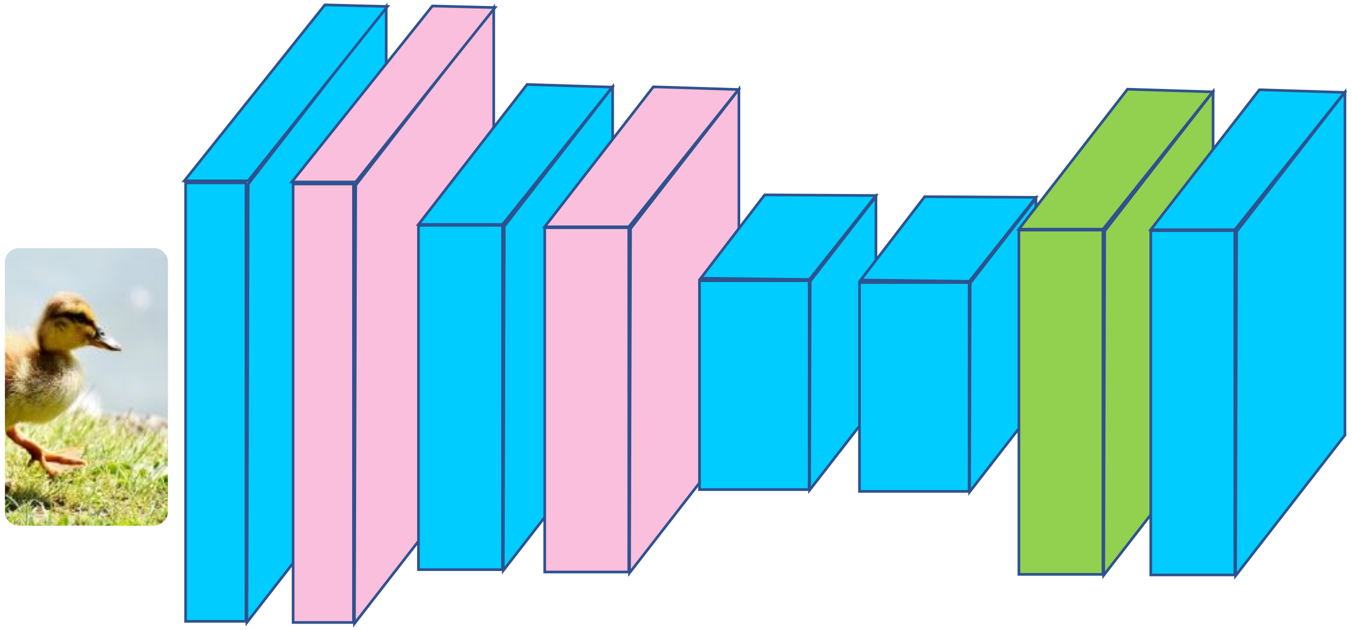


Wait a second!
We need to classify
each pixel!

Need to do **un**pooling!

Semantic segmentation

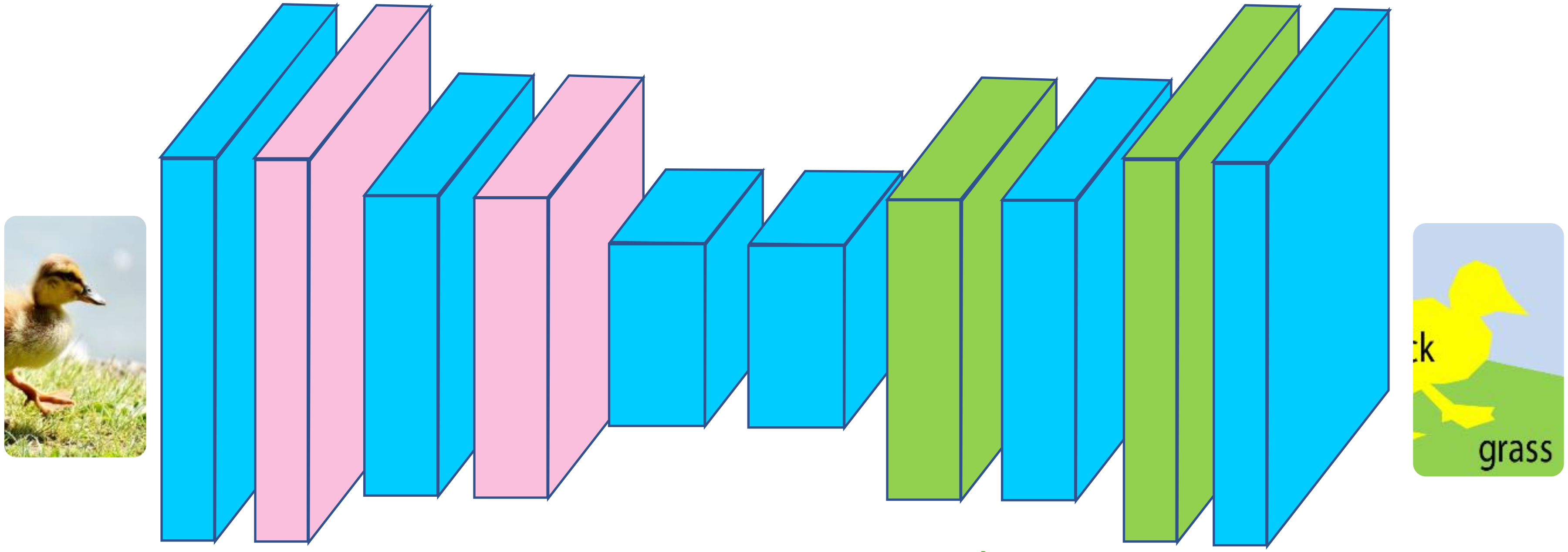
Let’s add pooling, which acts like down-sampling



up-sampling

Semantic segmentation

Let’s add pooling, which acts like down-sampling

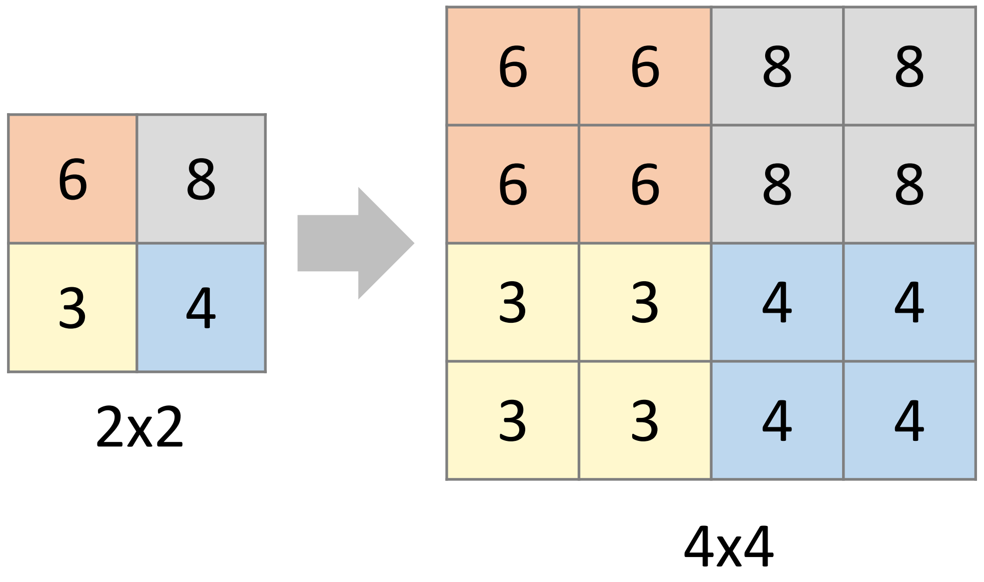


up-sampling

up-sampling

Nearest neighbor unpooling

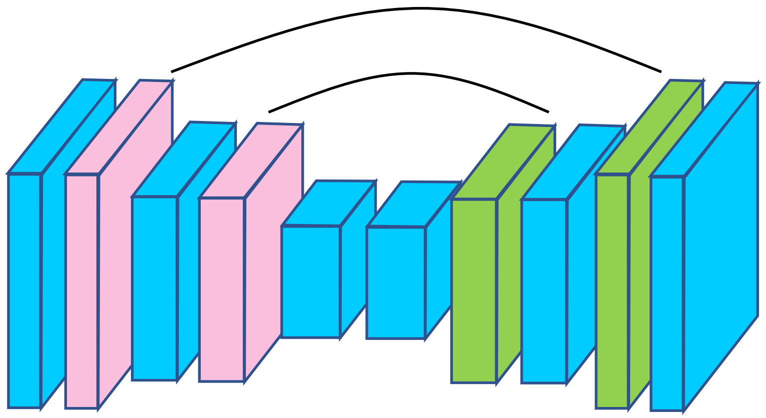
Fill with nearest neighbor values



Pixelated and not crisp!

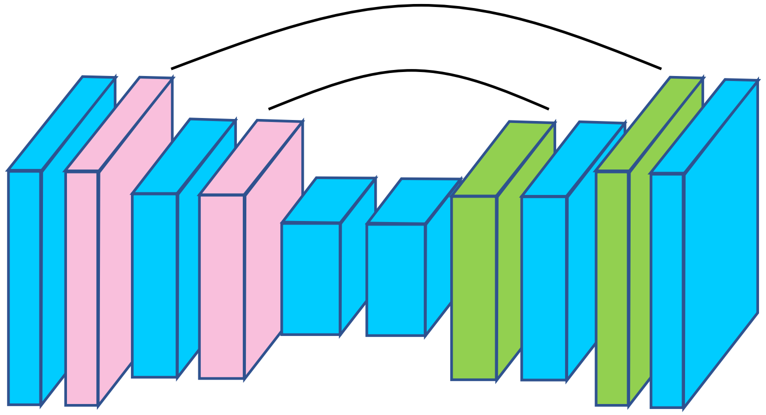
Max unpooling

Corresponding pairs of downsampling and upsampling layers



Max unpooling

Corresponding pairs of
downsampling and
upsampling layers



Remember which element was max during pooling, and fill that position during unpooling:

1	2	4	3
2	5	2	0
1	5	2	1
7	3	3	8

4x4

→

5	4
7	8

2x2

→ … →

1	2
3	4

2x2

→

0	0	2	0
0	1	0	0
0	0	0	0
3	0	0	4

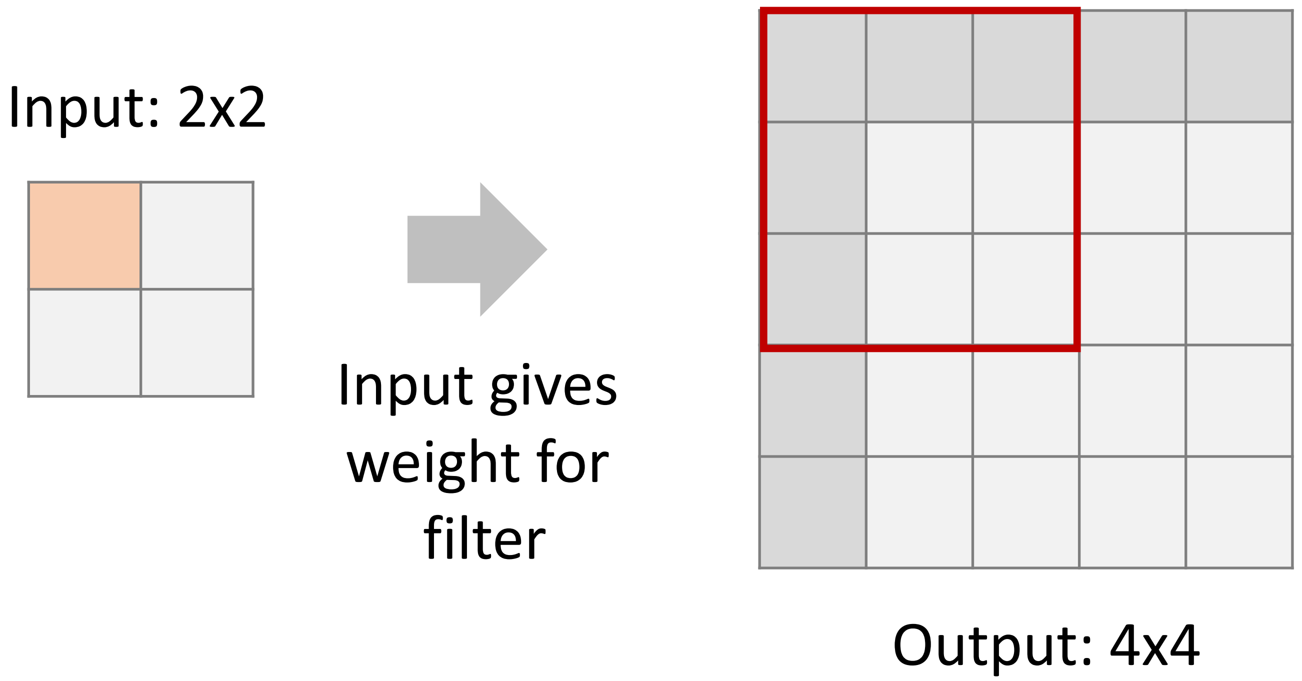
4x4

Learnable unpooling

- Previous approaches are not data-driven!
- We can replace max pooling layer with convolutional layer that has a bigger stride!
- What if we can apply convolutions to do unpooling?

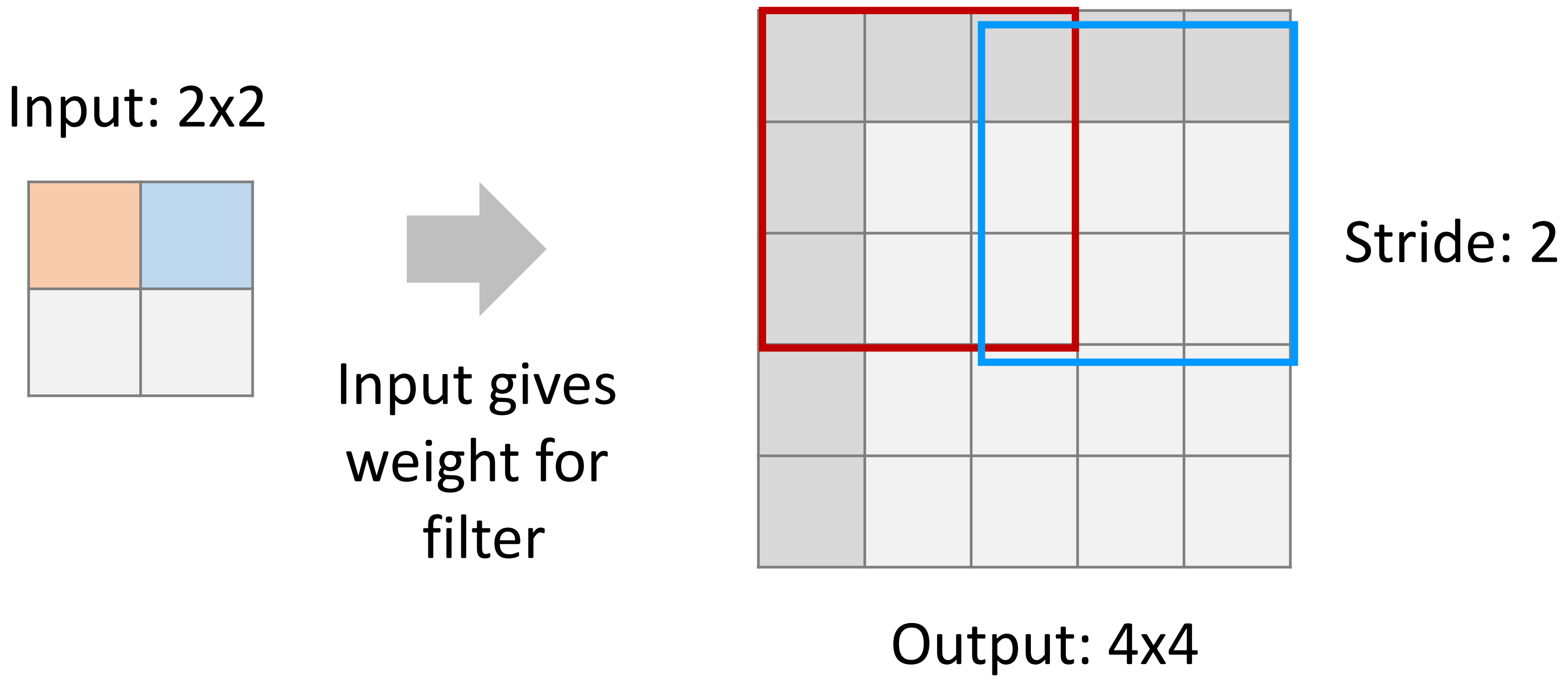
Learnable unpooling

- Previous approaches are not data-driven!
- We can replace max pooling layer with convolutional layer that has a bigger stride!
- What if we can apply convolutions to do unpooling?



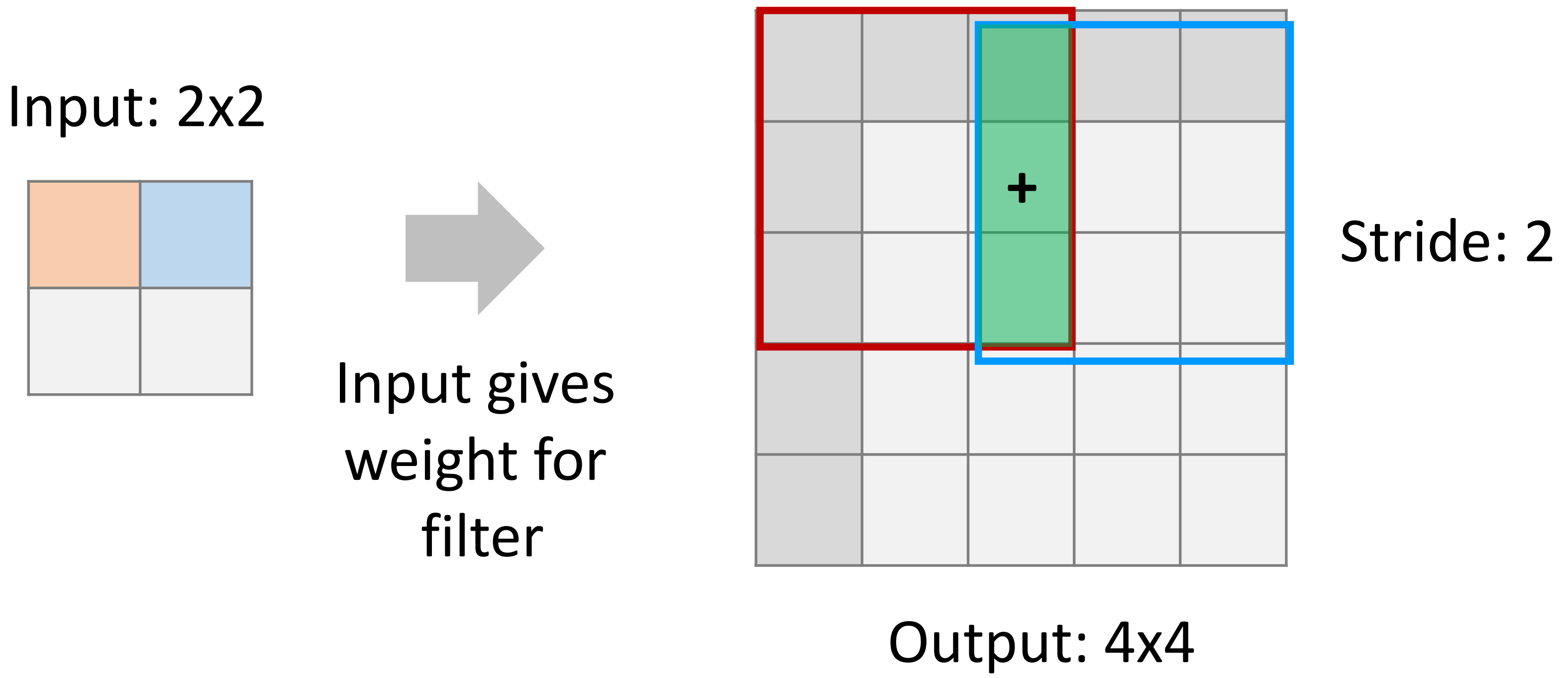
Learnable unpooling

- Previous approaches are not data-driven!
- We can replace max pooling layer with convolutional layer that has a bigger stride!
- What if we can apply convolutions to do unpooling?

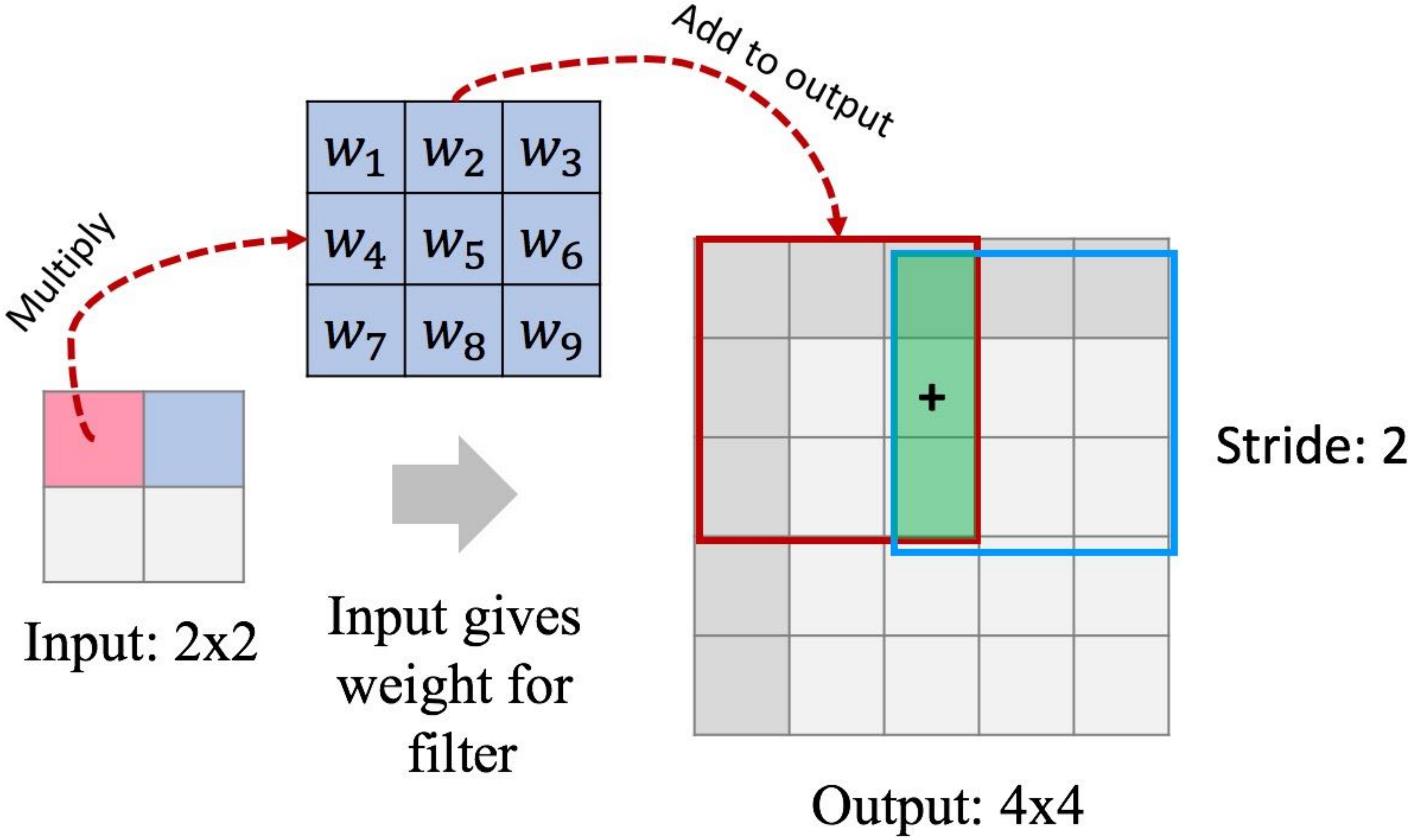


Learnable unpooling

- Previous approaches are not data-driven!
- We can replace max pooling layer with convolutional layer that has a bigger stride!
- What if we can apply convolutions to do unpooling?

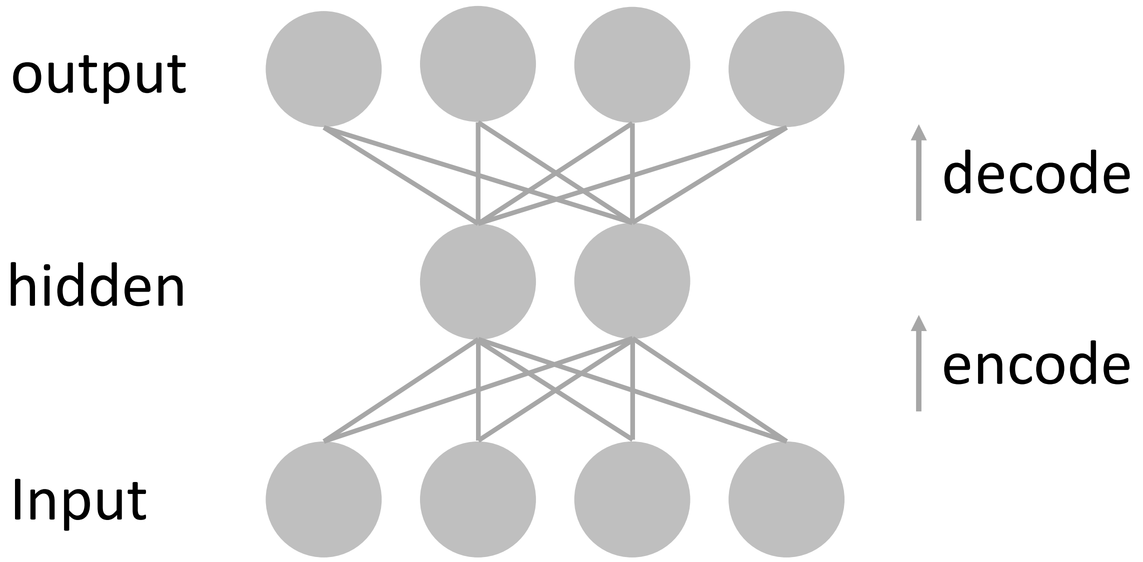


Transpose convolution



Autoencoders

- Encoder = data to hidden
- Decoder = hidden to data
- Decoder(Encoder(x)) ~ x



Linear case: “similar” to PCA or SVD

Example: matrix factorization

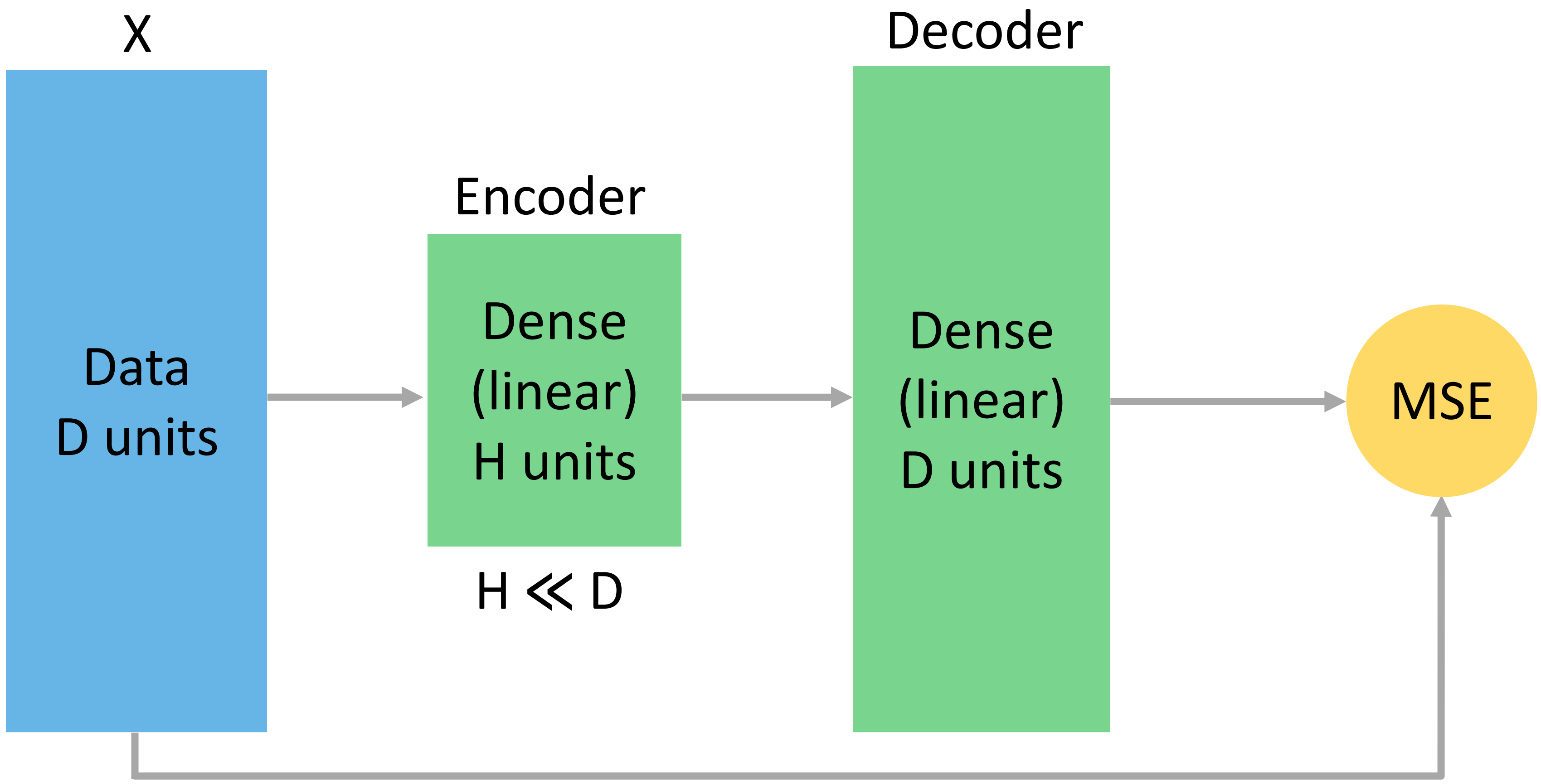
$$X = U \times V^T$$

Minimizing reconstruction error

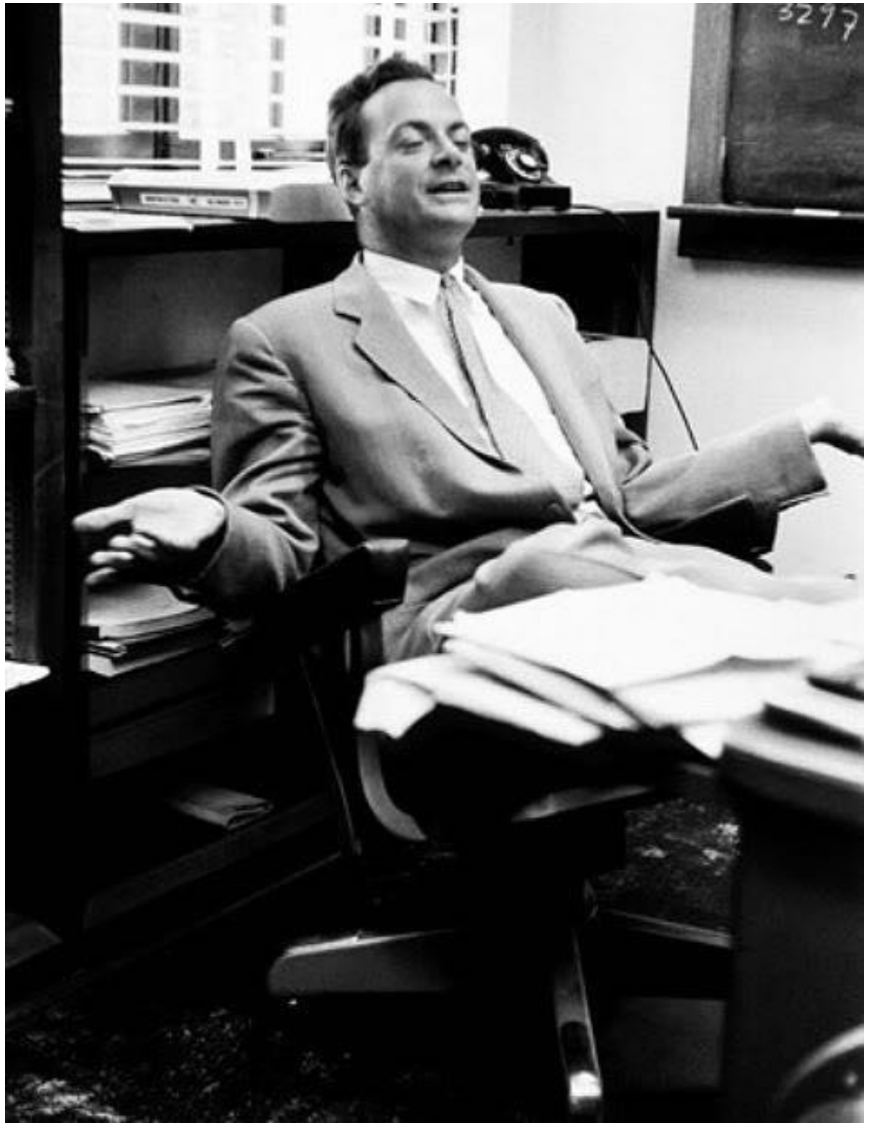
$$\|X - U \cdot V^T\| \to \min_{U,V}$$

Matrix decompositions

Как нейросеть

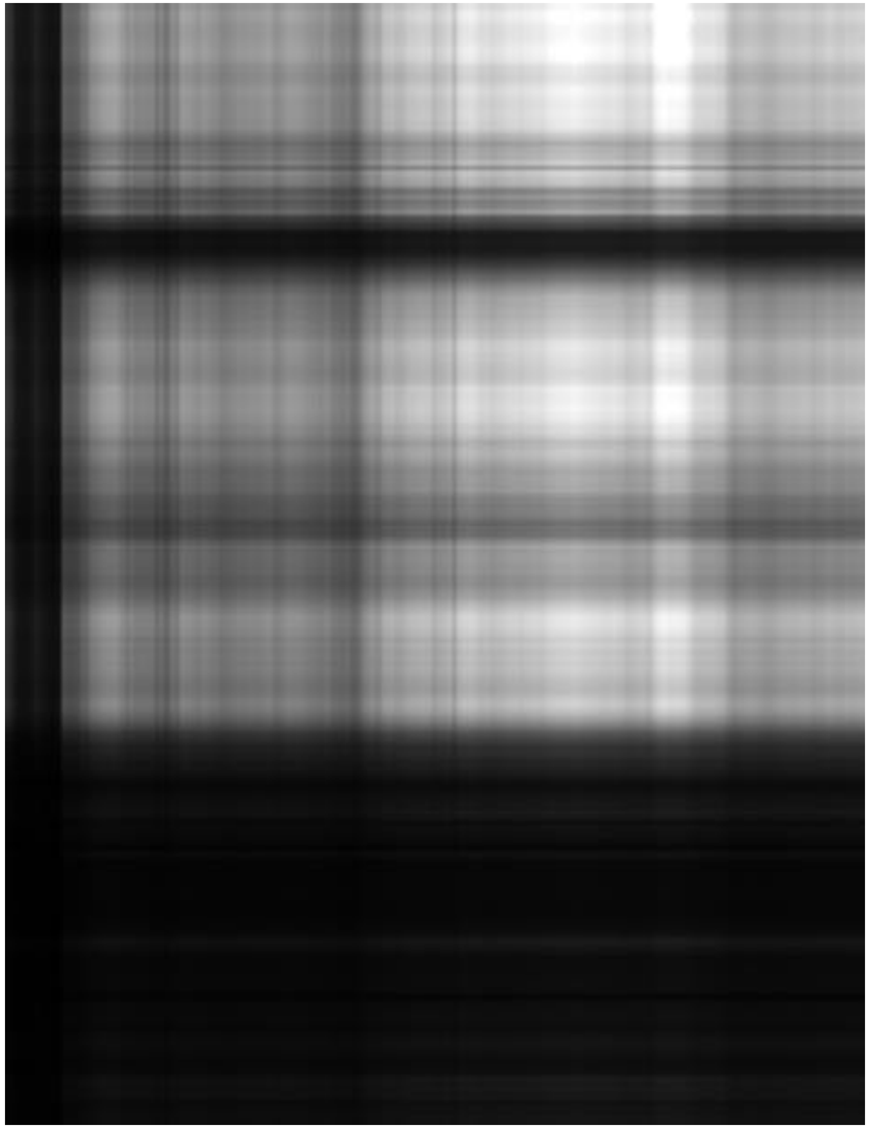


Интерпретация



Исходная картинка, попробуем применить SVD

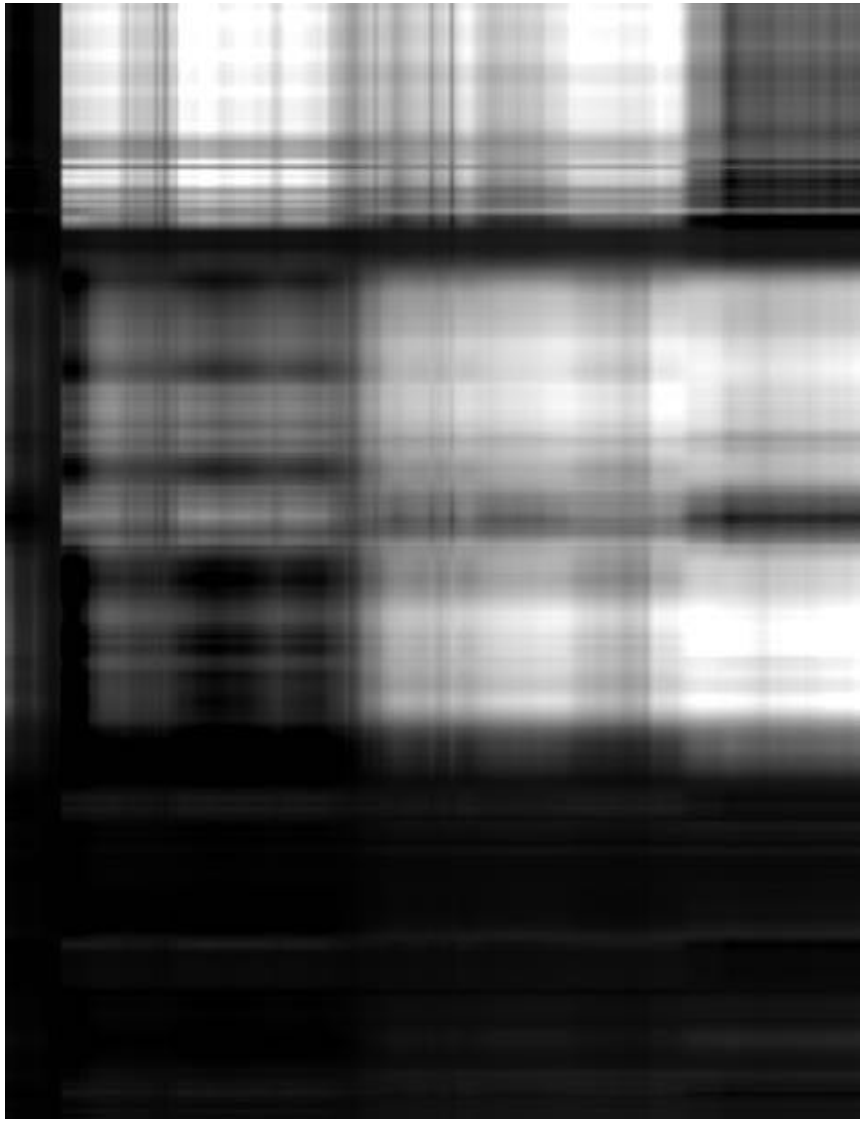
Интерпретация



Применили SVD и взяли только первый главный фактор.

k = 1

Интерпретация



Взяли 2 главных фактора.

$k = 2$

Интерпретация

k = 10

Интерпретация

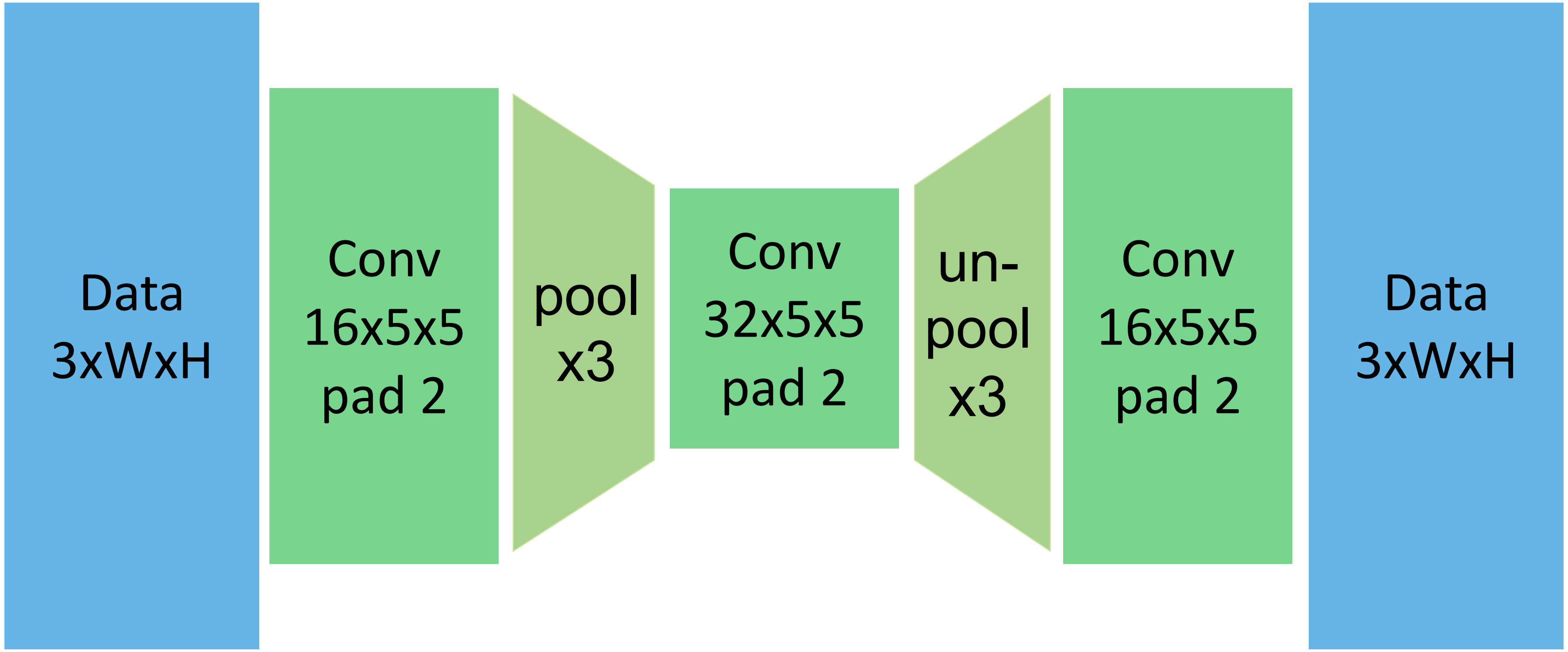
k = 50

Довольно неплохо!

Исходный размер был: 475x620.

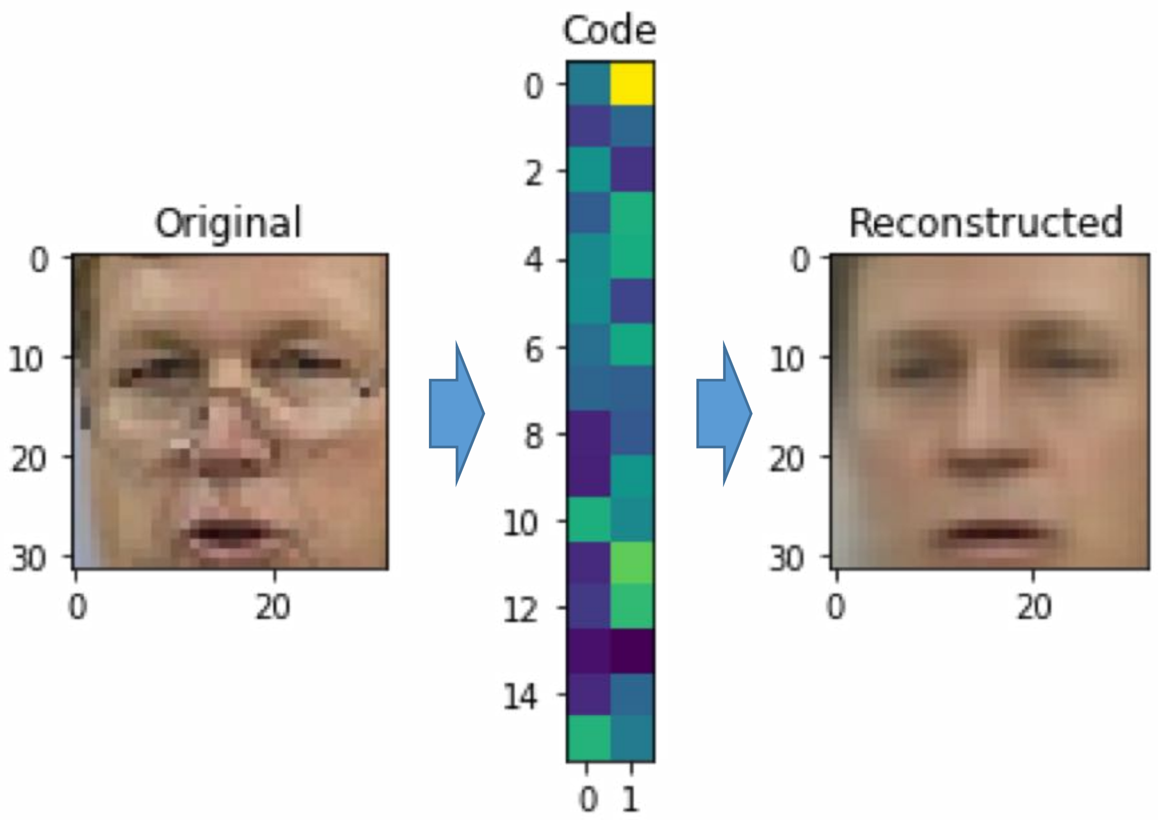
Что дальше?

Нужно учить глубокое преобразование!



Working with neural representations

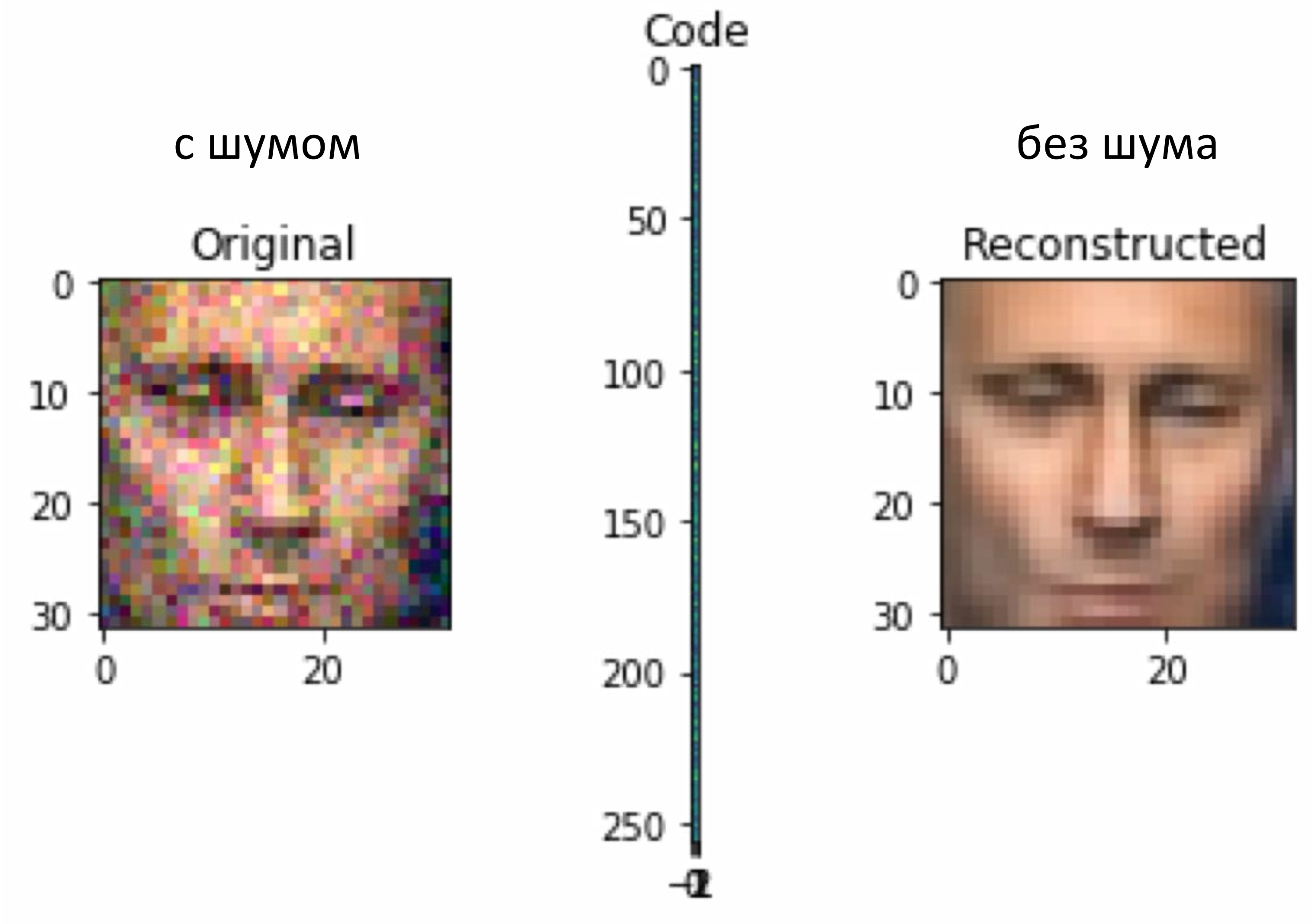
- We need to understand that a NN can convert an object to a small dense vector, which encodes the semantics:



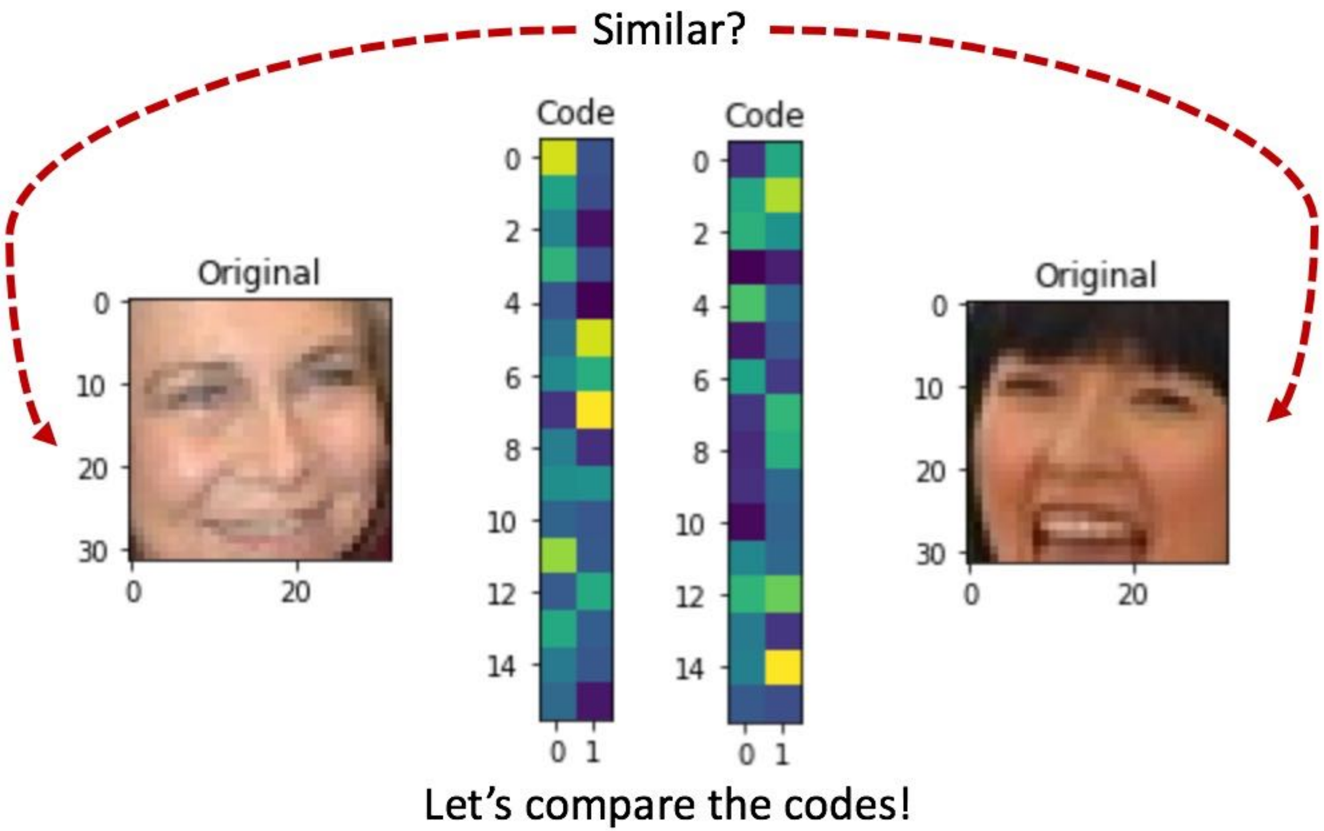
Denoising autoencoders

с шумом

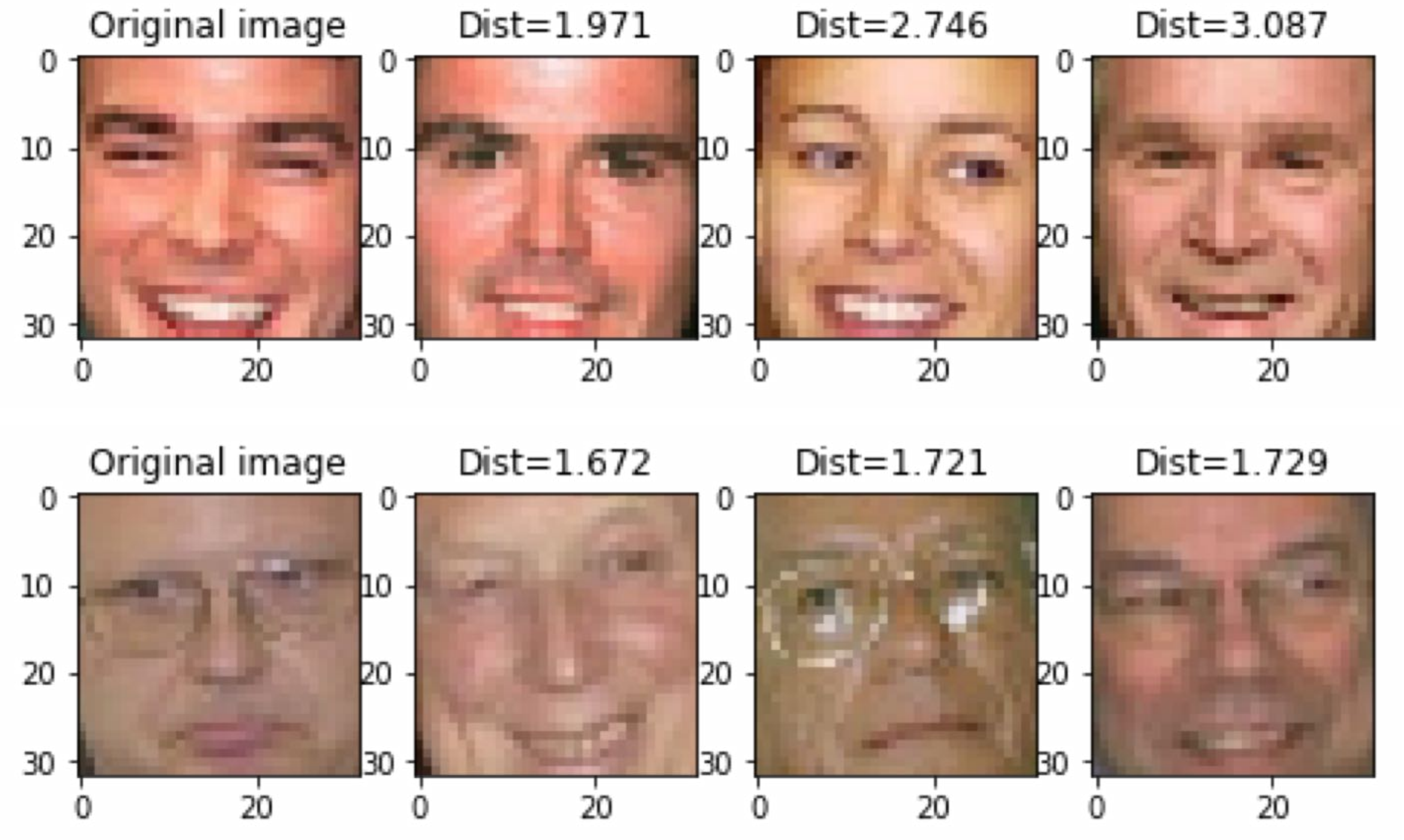
без шума



Поиск похожих картинок



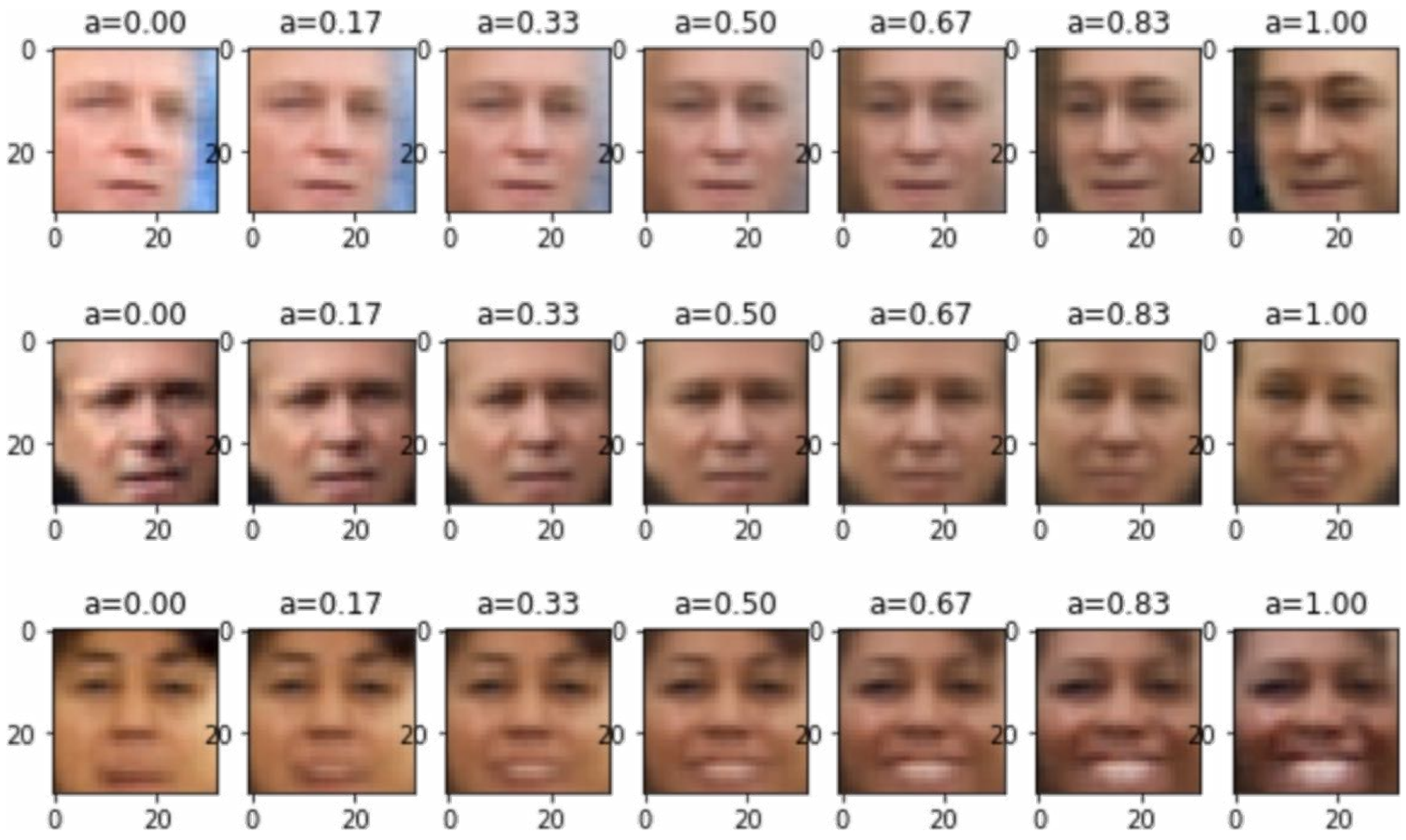
Поиск похожих картинок



Working with neural representations

- You can play around with those vectors to morph images:

А

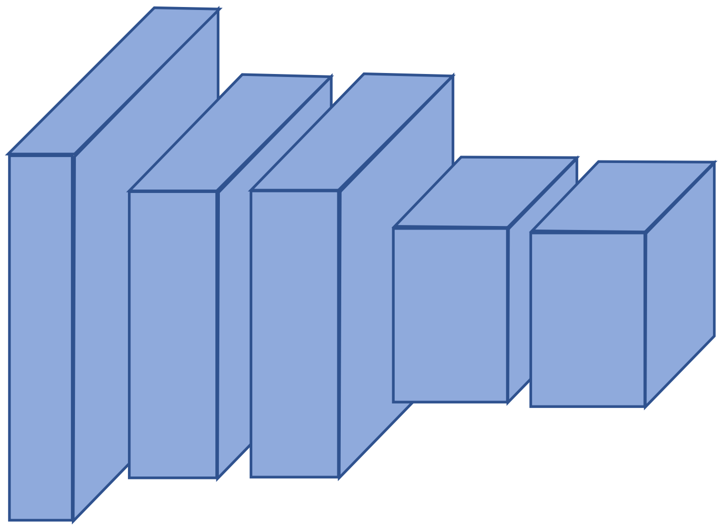


Б

Путешествуя в пространстве имбедингов плавно меняется картинка

И самое классное!

- Нам не нужны размеченные данные!
- Учим экстрактор фичей забесплатно!



Ссылки

- https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html
- https://github.com/hse-aml/intro-to-dl/blob/master/week4/Autoencoders-task.ipynb