

Визуализация нейронных сетей и генерация изображений

Александр Дьяконов

22 марта 2020 года

План

Зачем наблюдать? – За чем наблюдать?

Способы визуализации

Генерация изображений, текстур, пейзажей

Стилизация изображений, быстрая стилизация

Дистилляция данных

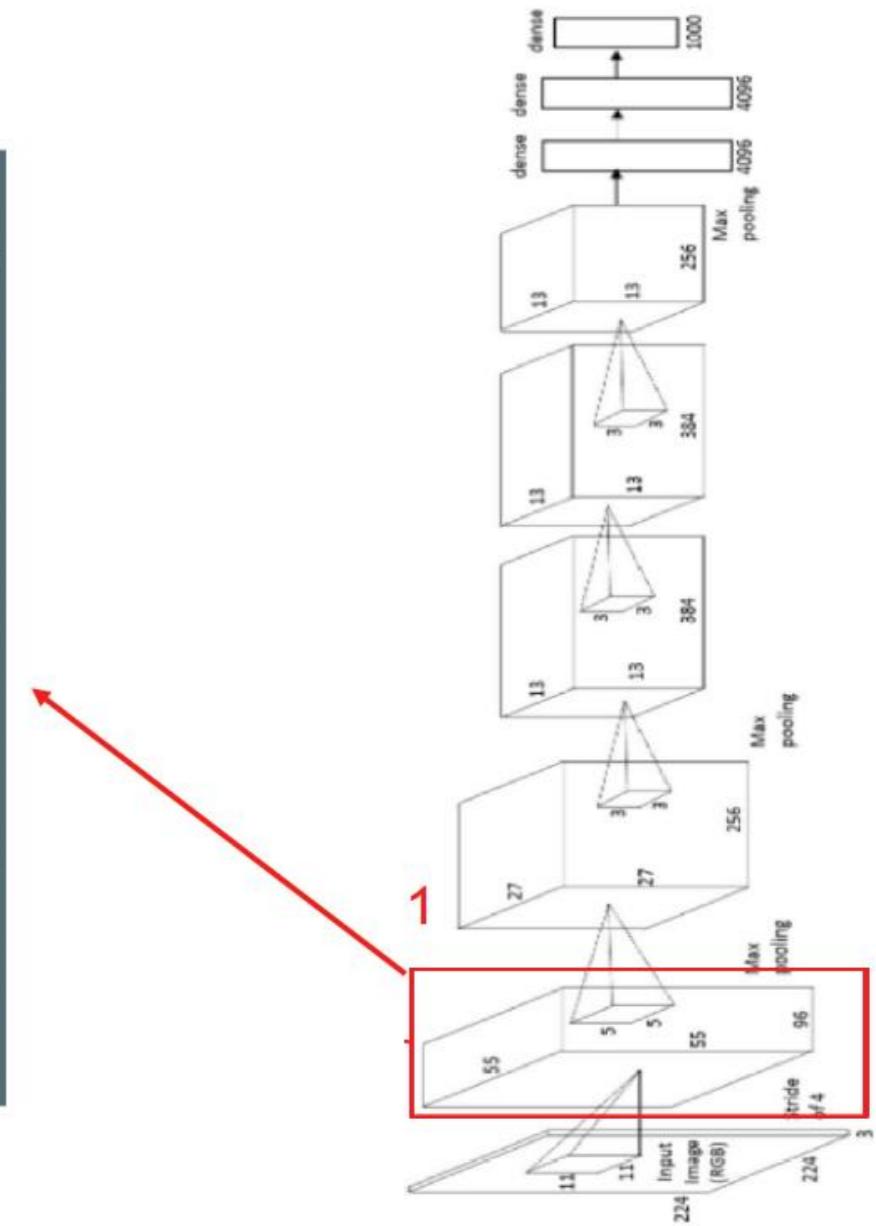
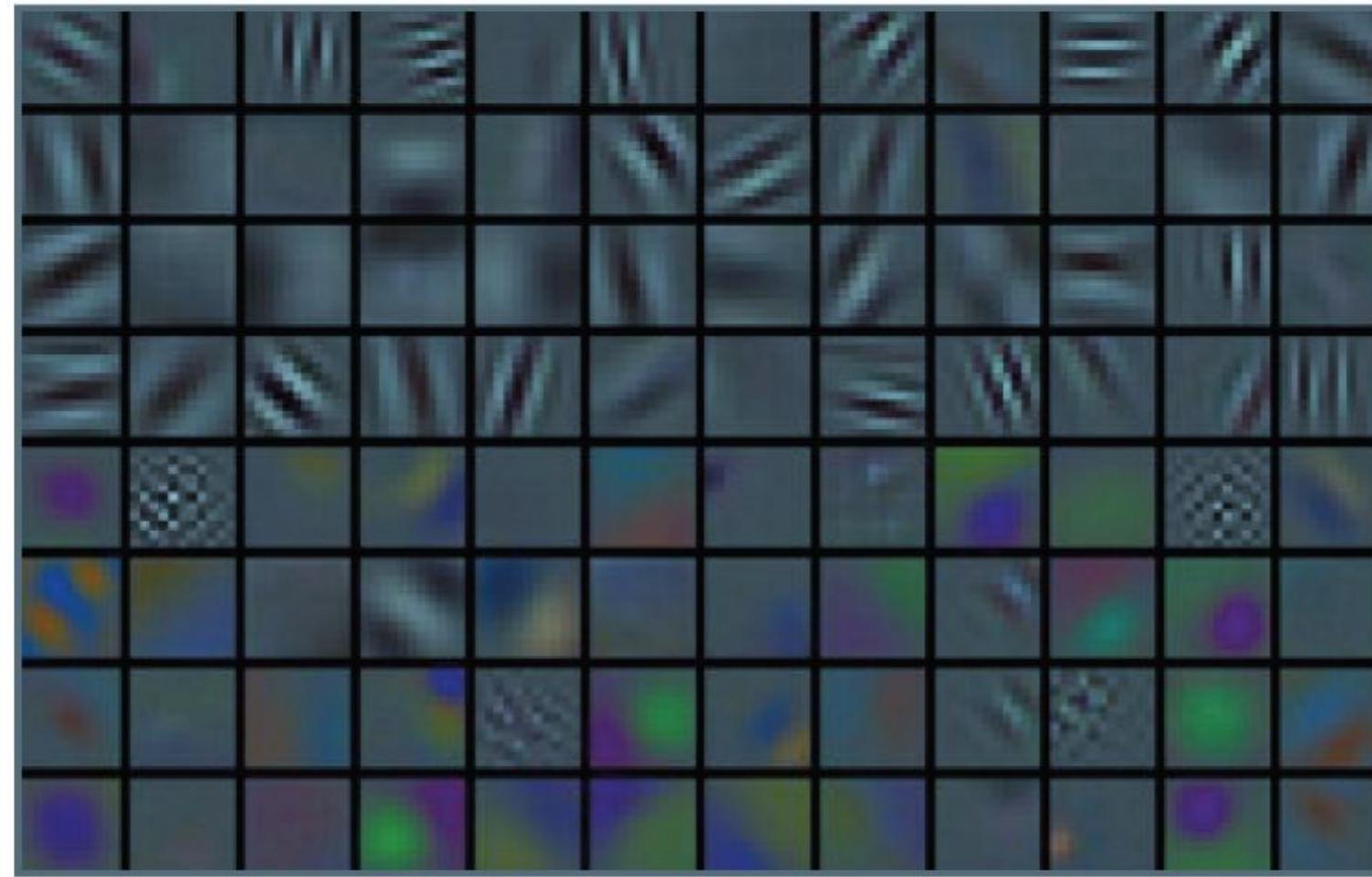
Зачем наблюдать?

- как и почему NN работает
- наблюдать преобразования признаковых пространств
- помогает изобретать новые подходы
- помогает видеть проблемы в данных / моделях

За чем можно наблюдать в NN?

- **параметры (ex: фильтры как картинки)**
- **внутренние активации – как картинки**
- **распределения активаций (на отдельных объектах)**
- **производные по входу**
- **входы, максимизирующие какой-то ответ**
- **визуализация в промежуточных признаковых пространствах
(любые средства стандартного ML)**
- **«adversarial samples» / обманные изображения (Fooling Images)**
будет отдельная тема

Визуализация весов



Для первых слоёв понятная интерпретация, для других – делаю по-другому

Визуализация весов

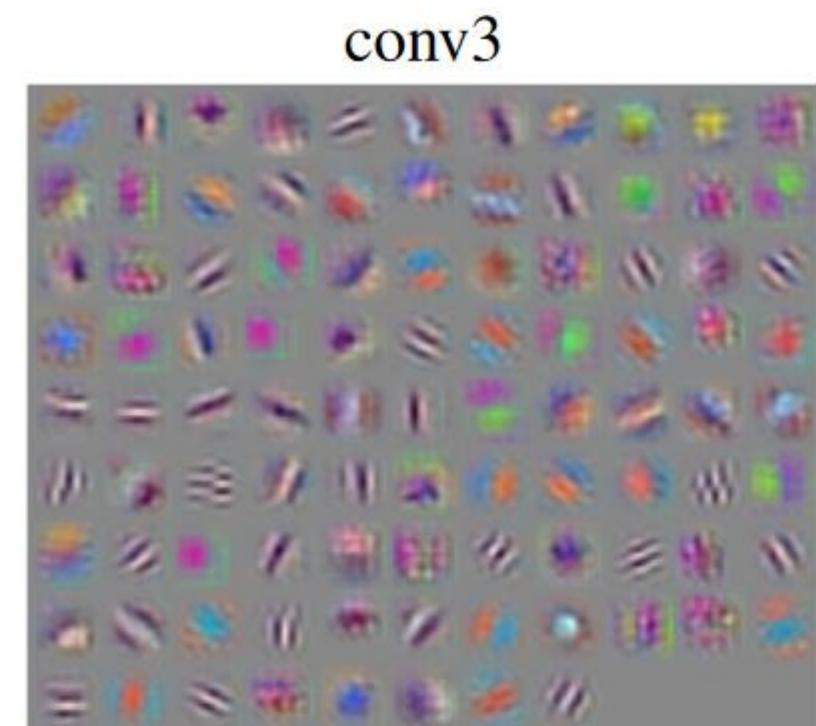


Figure 2: Visualizations of patterns learned by the lower layers (conv1-conv3) of the network trained on ImageNet. Each single patch corresponds to one filter. Interestingly, Gabor filters only appear in the third layer.

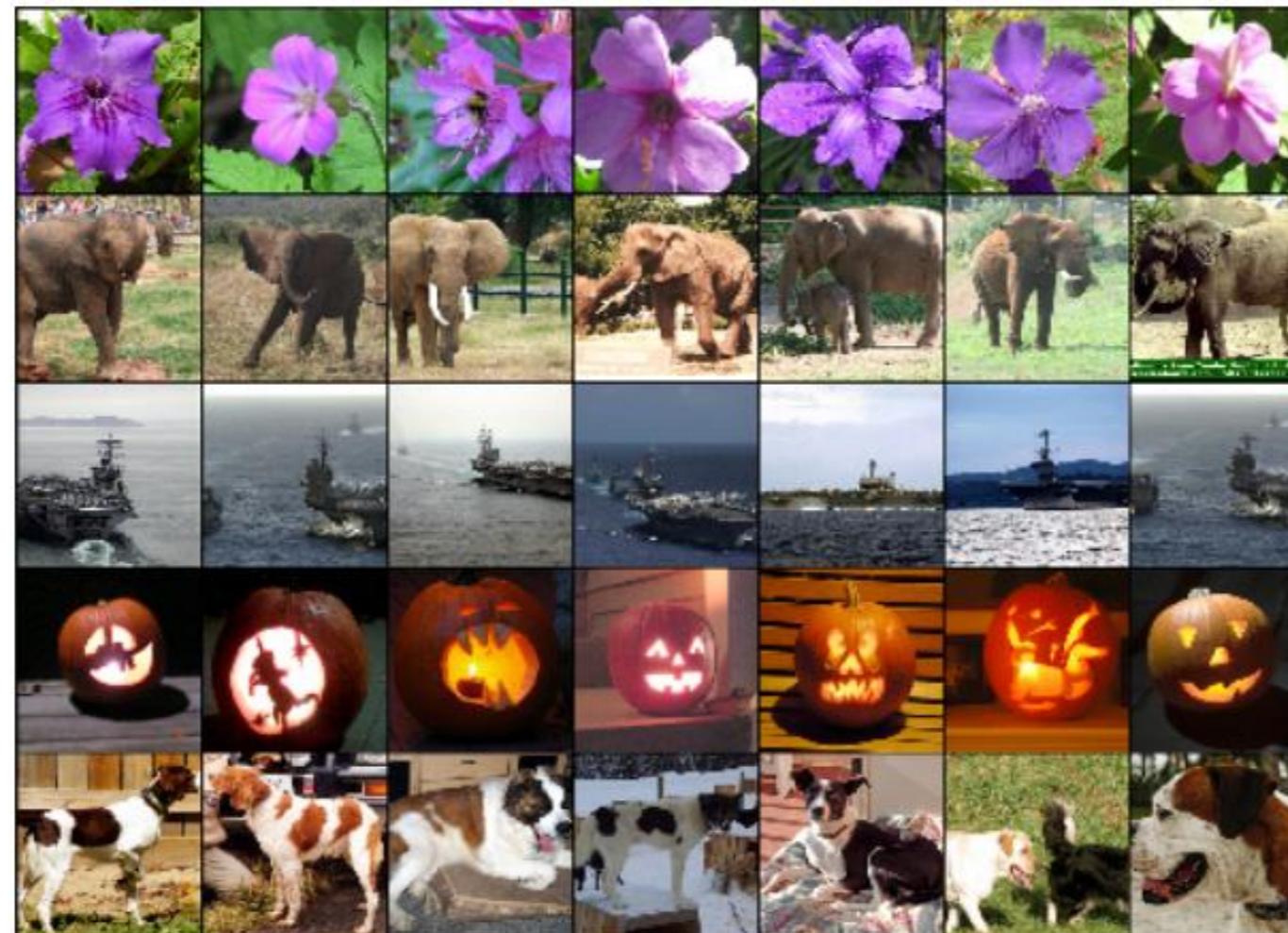
если недостаточно чёткие картинки – мало учили!

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, Martin Riedmiller «Striving for simplicity: the all convolutional net» <https://arxiv.org/pdf/1412.6806.pdf>

Стандартные средства в признаковых пространствах

Последний полносвязный слой

Можно смотреть соседей в этом признаковом пространстве



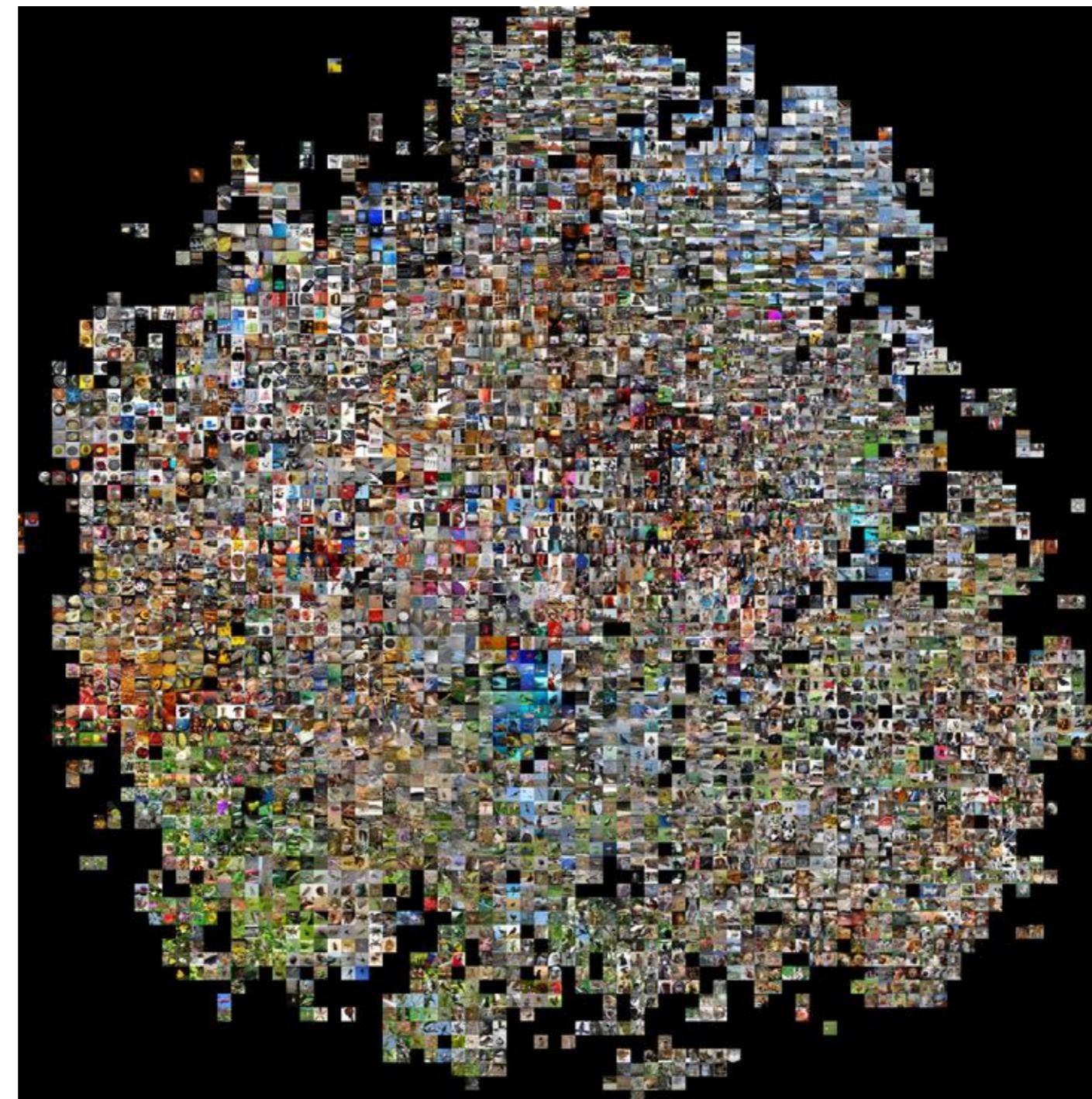
<https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>

Стандартные средства в признаковых пространствах

Последний полносвязный слой – применить уменьшение размерности... t-SNE в \mathbb{R}^2

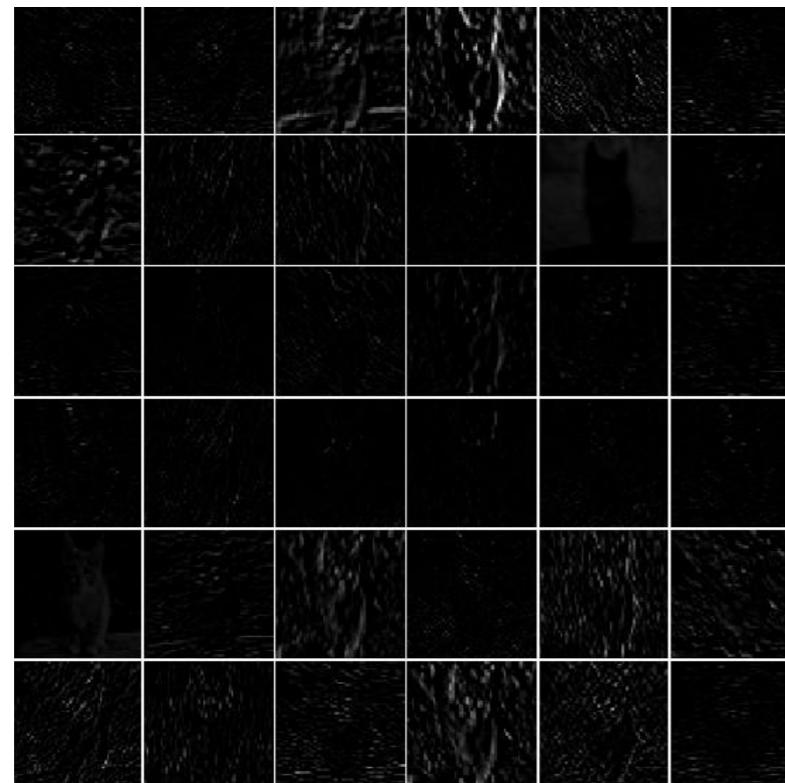


<https://cs.stanford.edu/people/karpathy/cnnembed/>

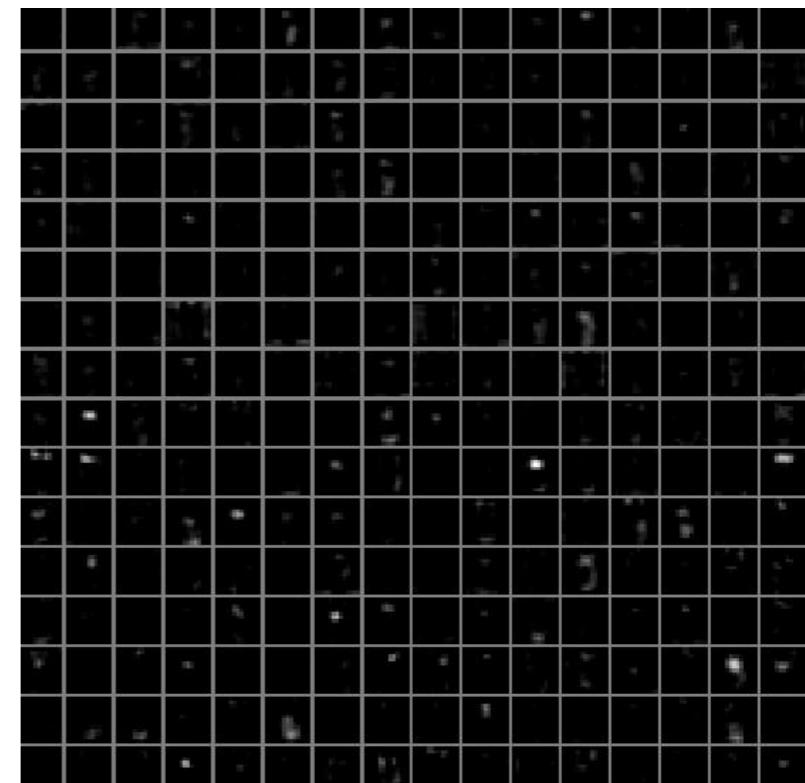


Анализ активации нейронов

первые слои



глубокие слои



чёрный цвет – ноль, на поздних слоях активации разреженные!

<https://stevenrush.github.io/understanding-cnn/>

Анализ активации нейронов

Средние слои: на каких изображениях максимальные значения активаций



<https://arxiv.org/pdf/1412.6806.pdf>

Анализ активации нейронов

Средние слои: на каких изображениях максимальные значения активаций

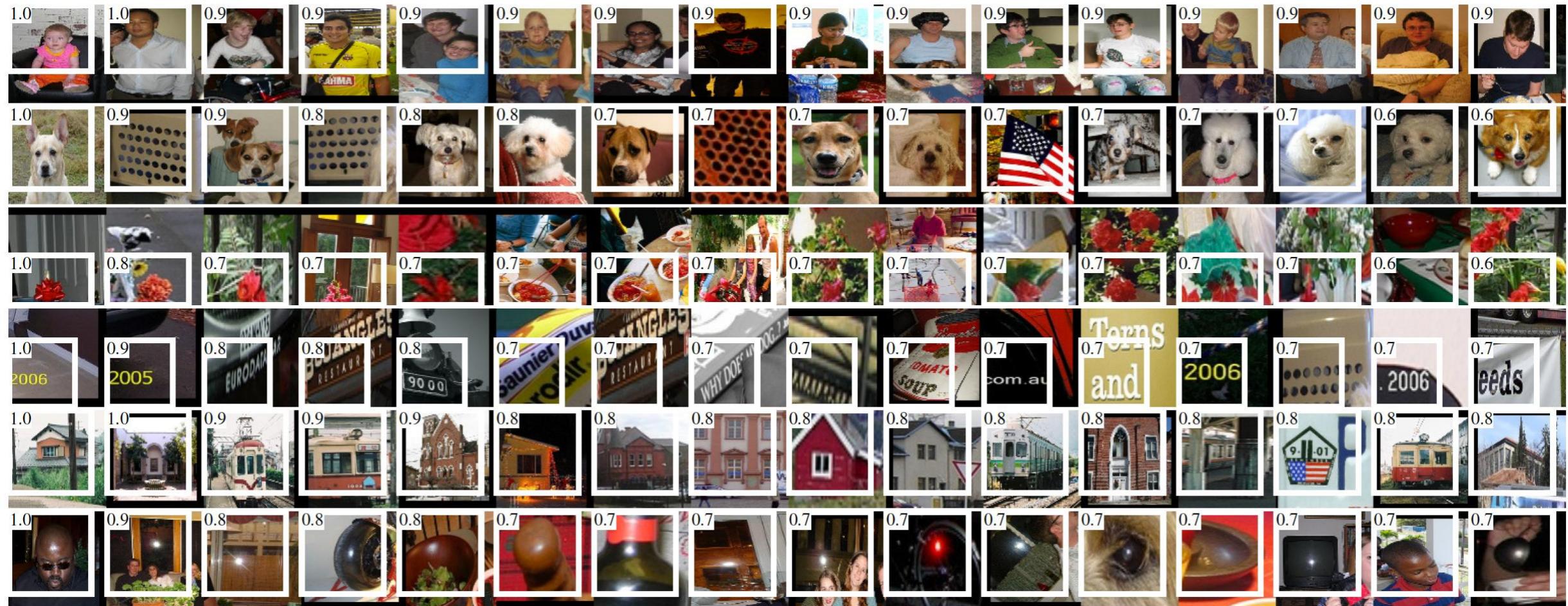
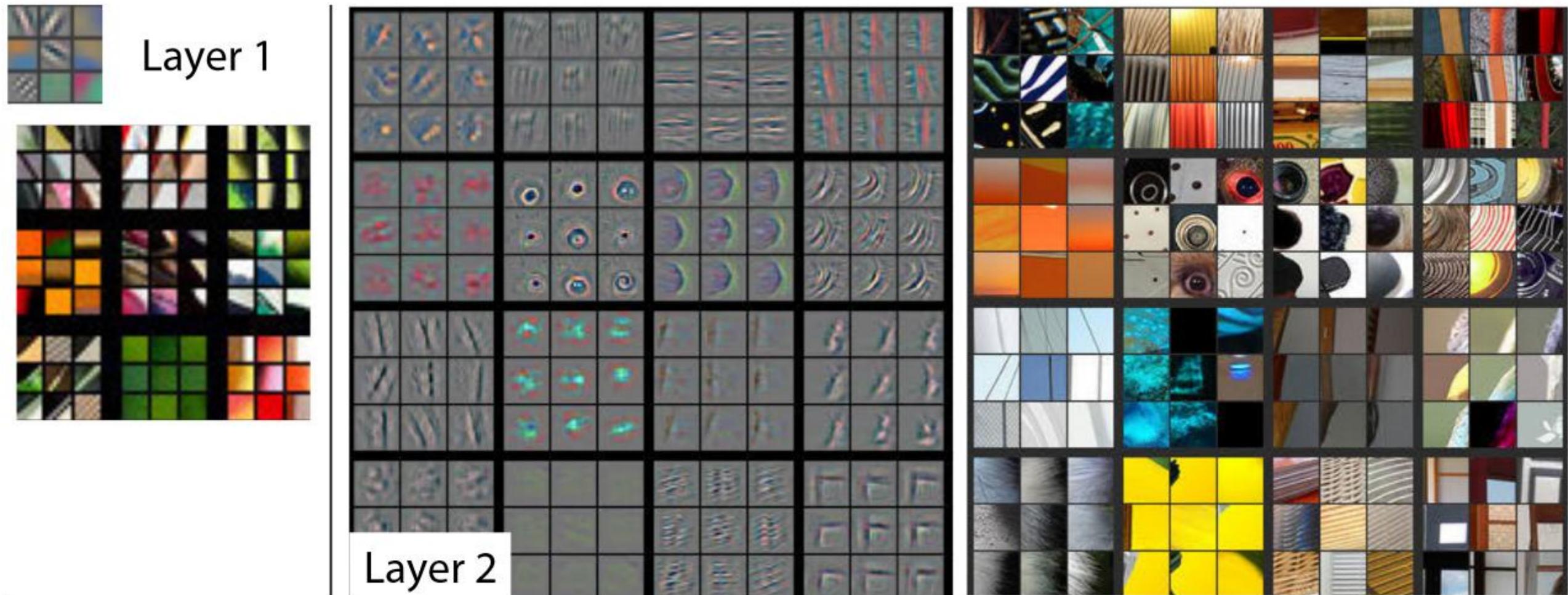


Figure 4. Top 16 images that triggered the maximum activation in the intermediate layers.

Rich feature hierarchies for accurate object detection and semantic segmentation – Girshick, et al - 2013

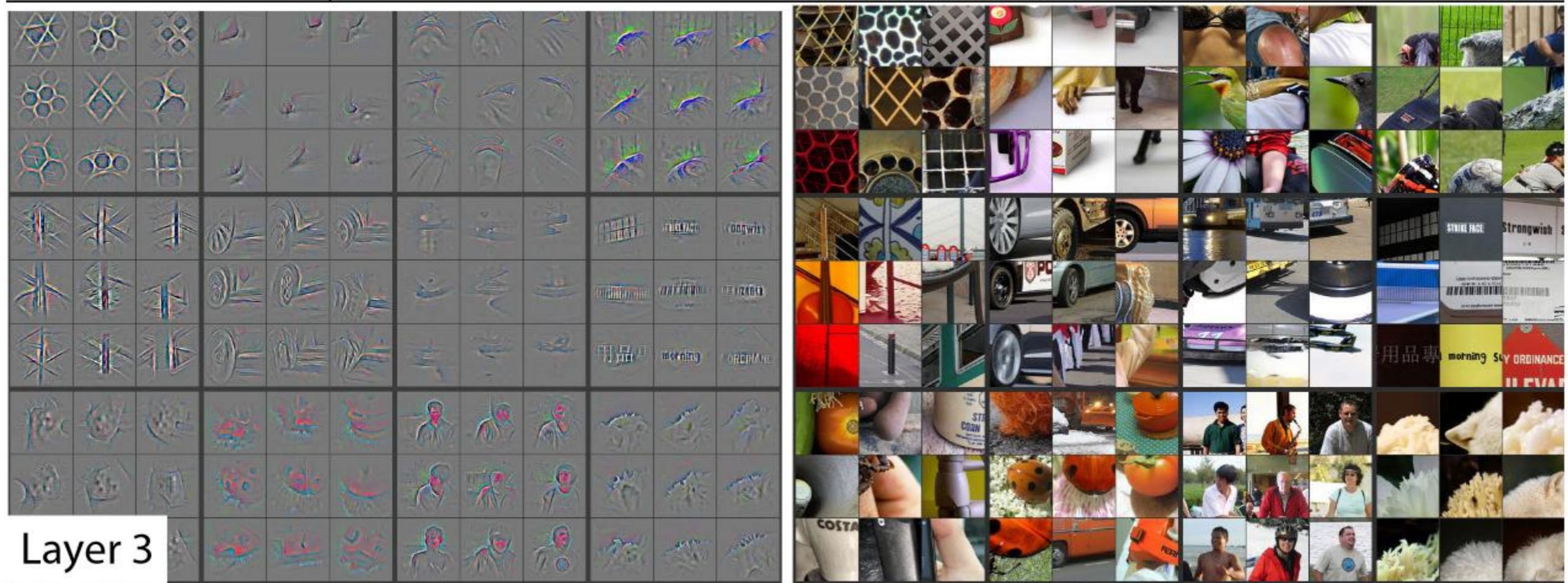
Visualizing the neurons along the way to the top

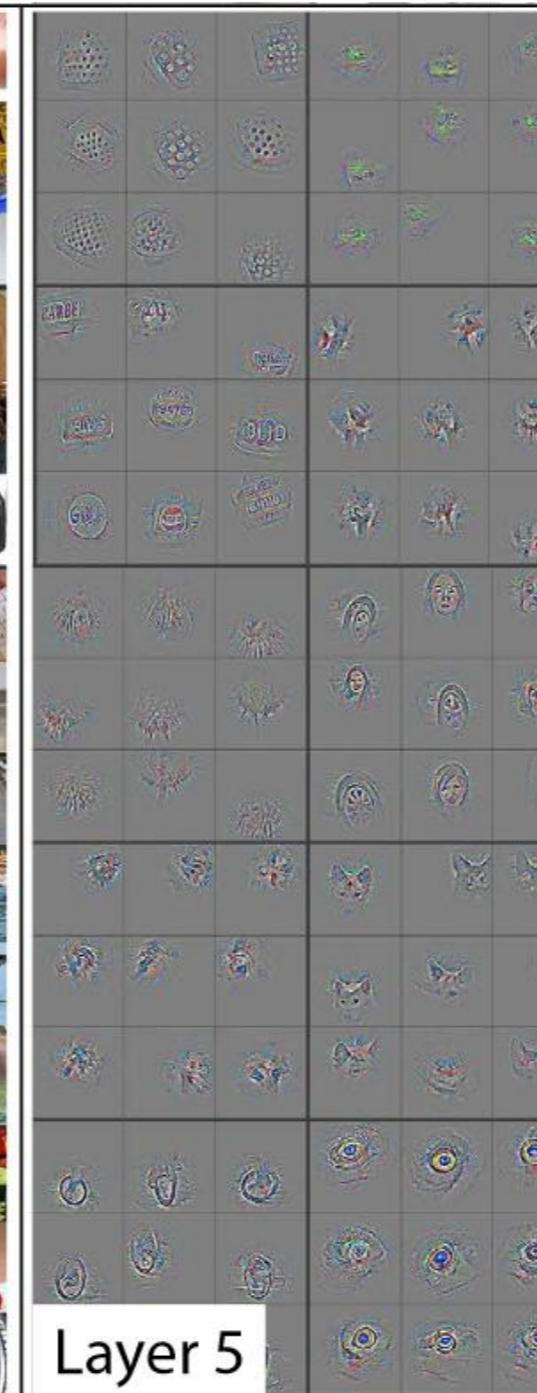
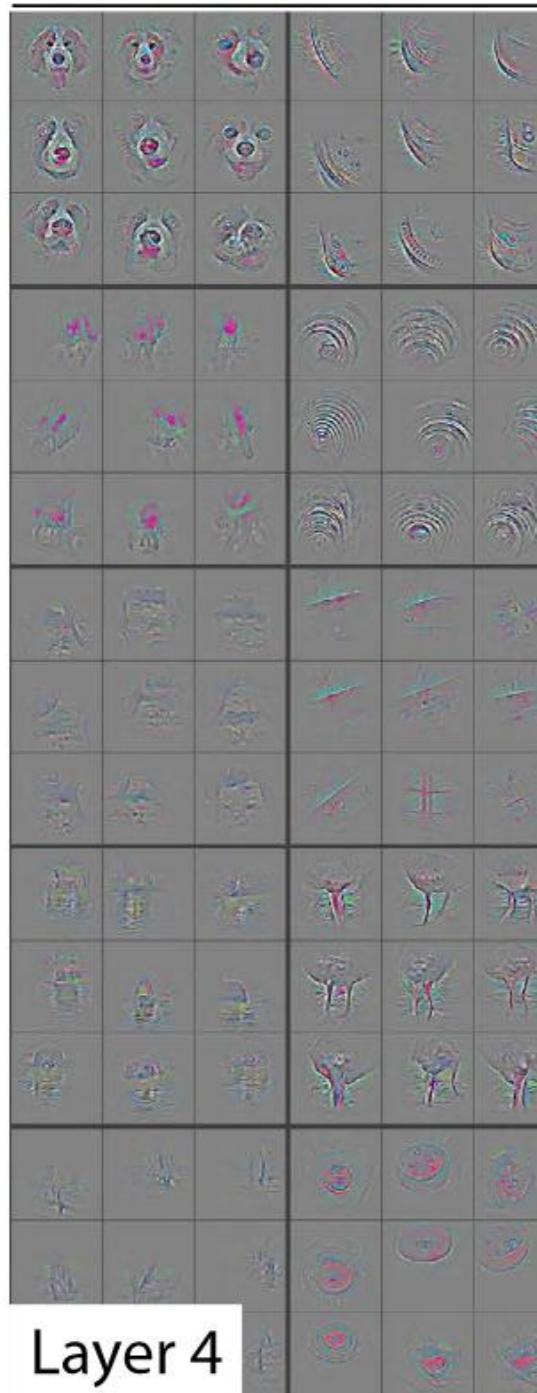
для слоёв 2-5 выбираем нейрон, выбираем 9 картинок, на которых его активация максимальна, «делаем deconv» дальше



Zeiler, Rob Fergus «Visualizing and Understanding Convolutional Networks» <https://arxiv.org/pdf/1311.2901.pdf>

Visualizing the neurons along the way to the top





Чувствительность к удалению (Occlusion sensitivity)

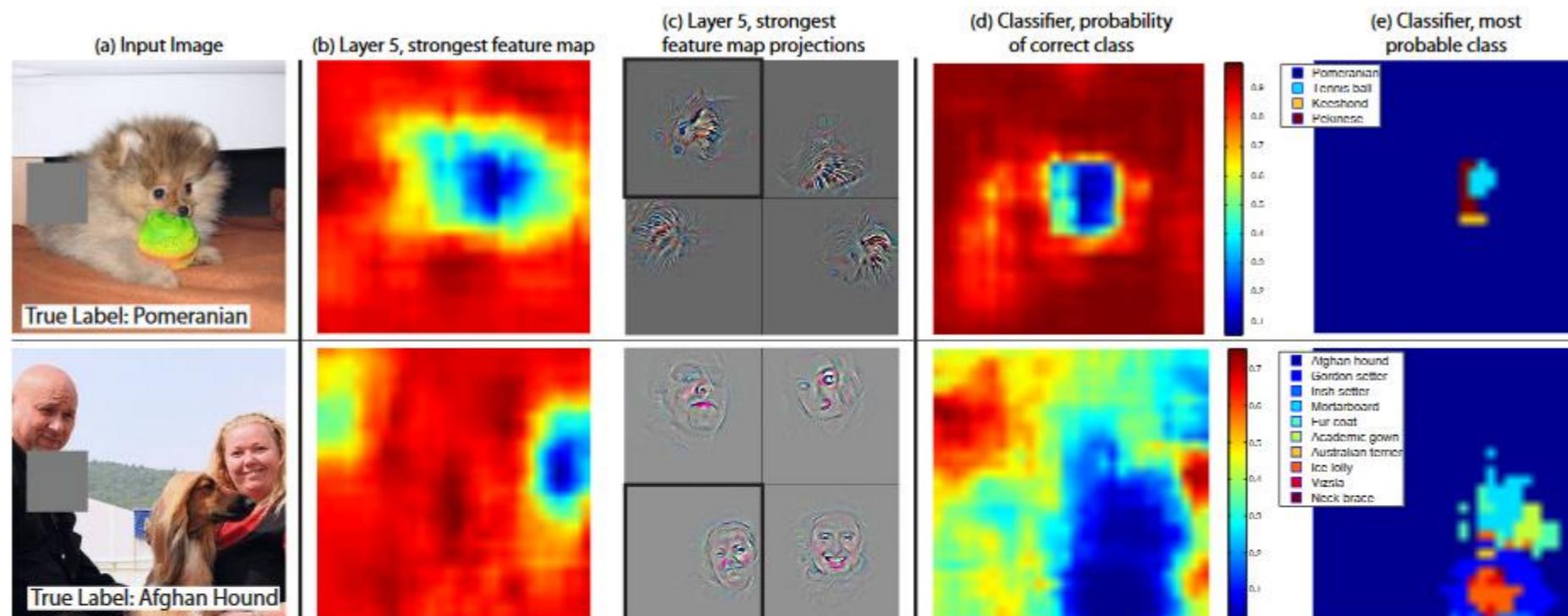


Figure 7. Three test examples where we systematically cover up different portions of the scene with a gray square (1st column) and see how the top (layer 5) feature maps ((b) & (c)) and classifier output ((d) & (e)) changes. (b): for each position of the gray scale, we record the total activation in one layer 5 feature map (the one with the strongest response in the unoccluded image). (c): a visualization of this feature map projected down into the input image (black square), along with visualizations of this map from other images. The first row example shows the strongest feature to be the dog's face. When this is covered-up the activity in the feature map decreases (blue area in (b)). (d): a map of correct class probability, as a function of the position of the gray square. E.g. when the dog's face is obscured, the probability for "pomeranian" drops significantly. (e): the most probable label as a function of occluder position. E.g. in the 1st row, for most locations it is "pomeranian", but if the dog's face is obscured but not the ball, then it predicts "tennis ball". In the 2nd example, text on the car is the strongest feature in layer 5, but the classifier is most sensitive to the wheel. The 3rd example contains multiple objects. The strongest feature in layer 5 picks out the faces, but the classifier is sensitive to the dog (blue region in (d)), since it uses multiple feature maps.

Чувствительность к удалению (Occlusion sensitivity)

Чтобы оценить, какие пиксели отвечают за отнесению к классу

**Закрывать последовательно часть изображения – 2D-гистограмма вероятности
принадлежности к заданному классу при закрытии с центром в ij -м пикселе**

Matthew D Zeiler, Rob Fergus «Visualizing and Understanding Convolutional Networks»

<https://arxiv.org/pdf/1311.2901.pdf>

Чувствительность к удалению (Occlusion sensitivity)

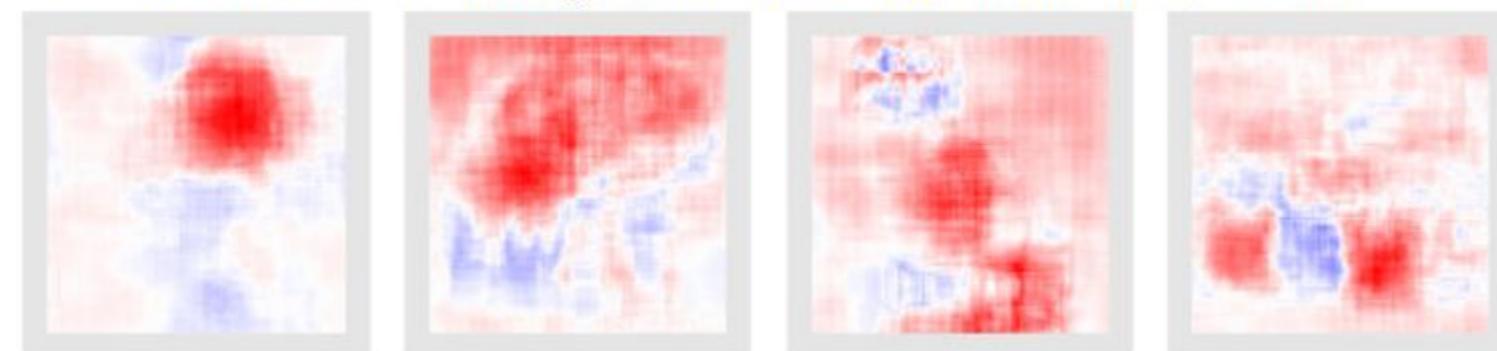
Original images



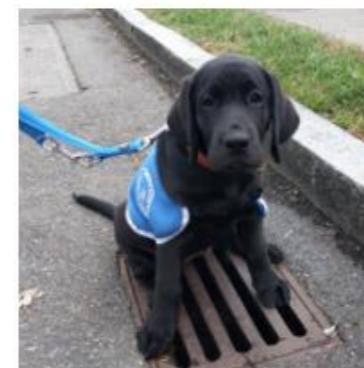
Occlusion mask 32×32



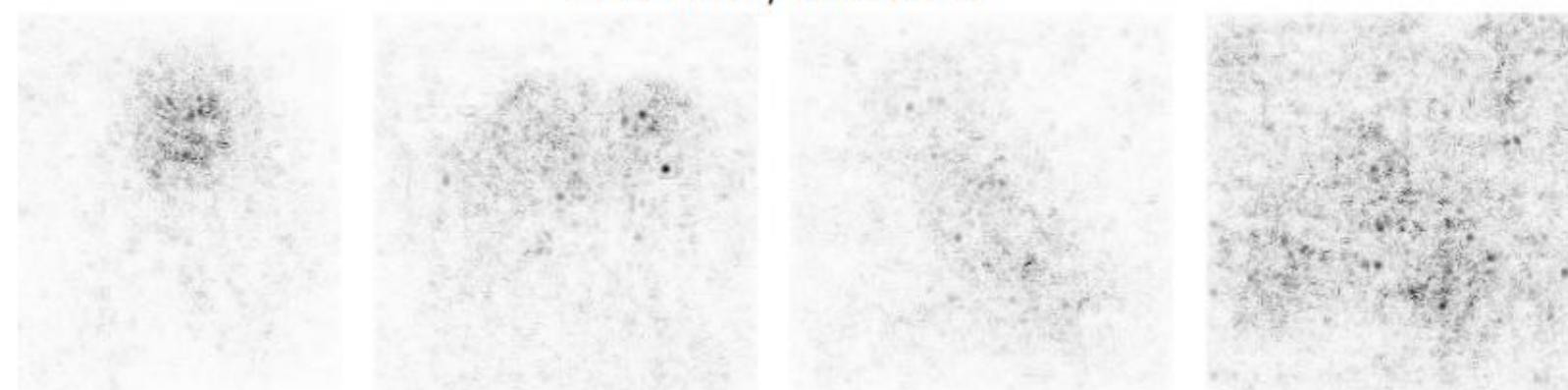
Occlusion sensitivity, mask 32×32 , stride of 2, AlexNet



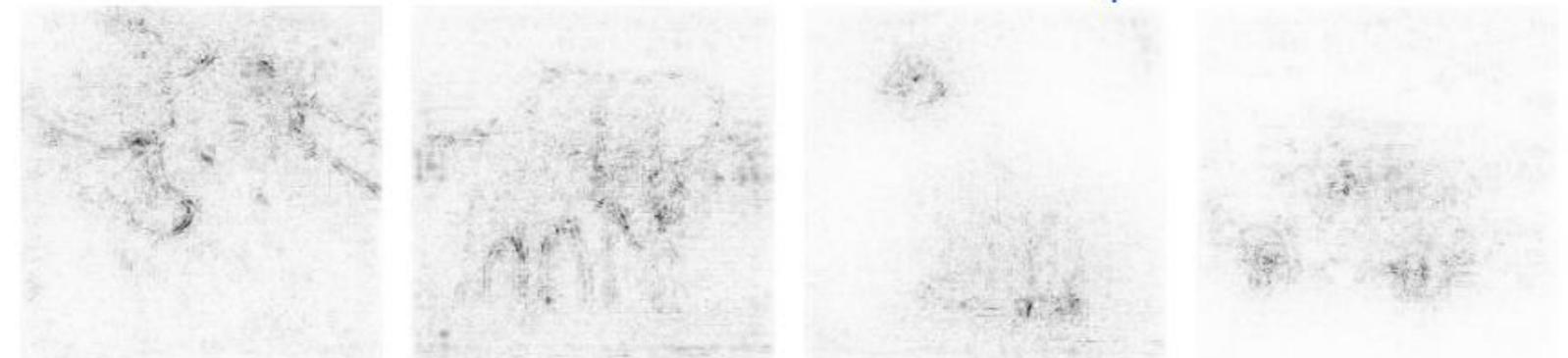
«Saliency maps» – градиенты (их модули) по входу



Gradient, AlexNet



SmoothGrad, AlexNet, $\sigma = \frac{\Delta}{4}$



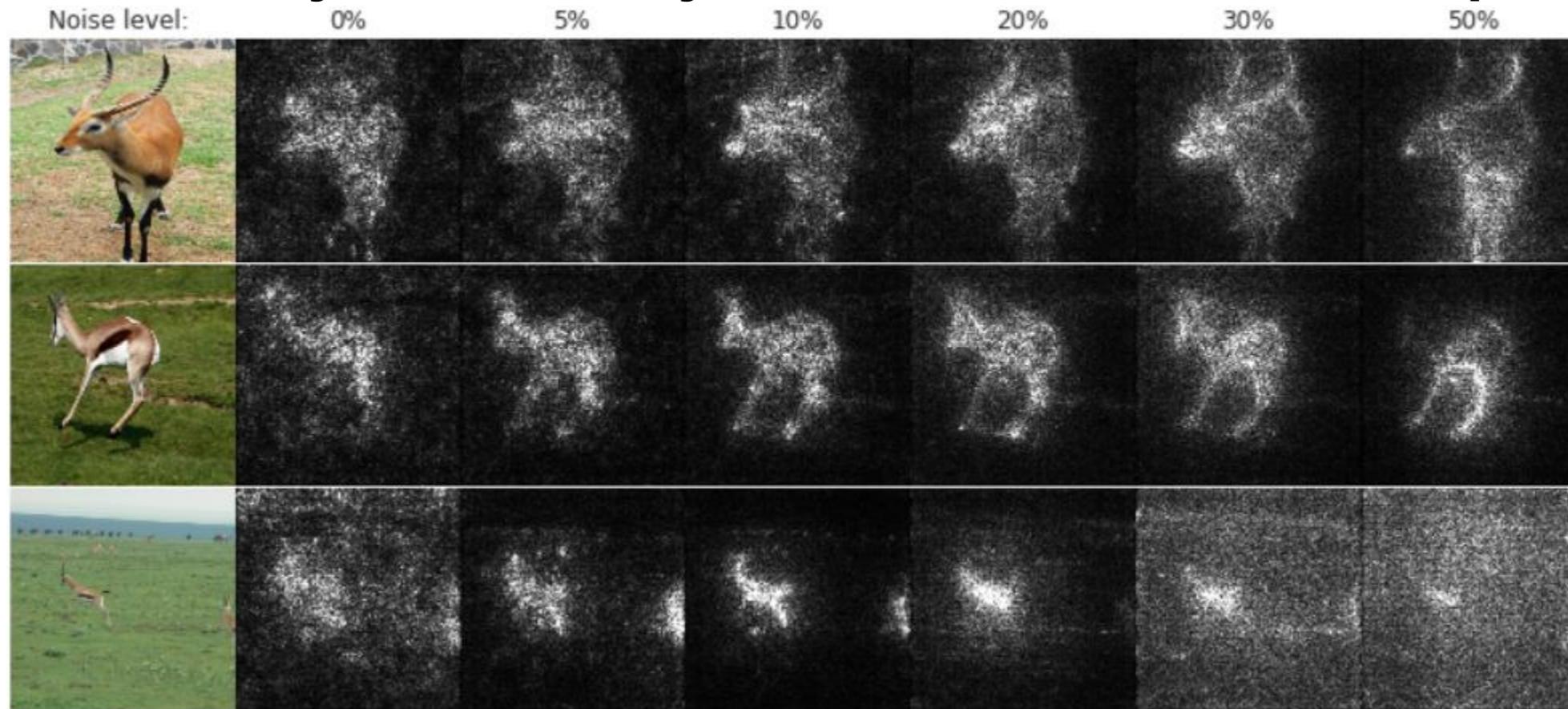
«Saliency maps» – градиенты (их модули) по входу



<https://arxiv.org/pdf/1312.6034.pdf>

«Saliency maps» – градиенты по входу

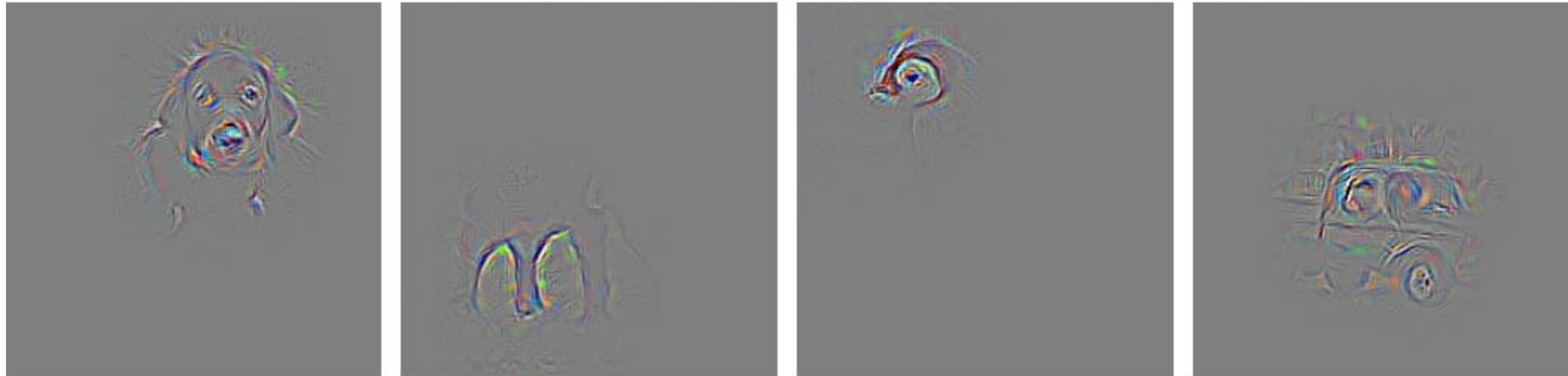
с помощью шума можно получать более адекватные иллюстрации



Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, Martin Wattenberg «SmoothGrad: removing noise by adding noise» //
<https://arxiv.org/abs/1706.03825>

«guided back-propagation» ~ градиенты по входу

AlexNet, max feature response, guided back-propagation



сейчас объясним как...

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, Martin Riedmiller «Striving for Simplicity: The All Convolutional Net» //
<https://arxiv.org/abs/1412.6806>

Как пропускается сигнал обратно по сети...

The deconvolutional network («deconvnet») approach
– для визуализации чему обучились дальние слои

для этого сигнал пускаем обратно по сети
часто конкретный нейрон = 1, остальные = 0
след. слайд – как

если используется pooling – запоминаем, где был максимум
но тогда есть зависимость от конкретного изображения!!!

смотрим на получившееся изображение

Другой подход – использовать обратное распространение – The backward pass
по большому счёту, разница в том, как осуществляется обратный проход через ReLU

«guided backpropagation» – комбинация этих методов, см **след. слайд**

Как пропускается сигнал обратно по сети...

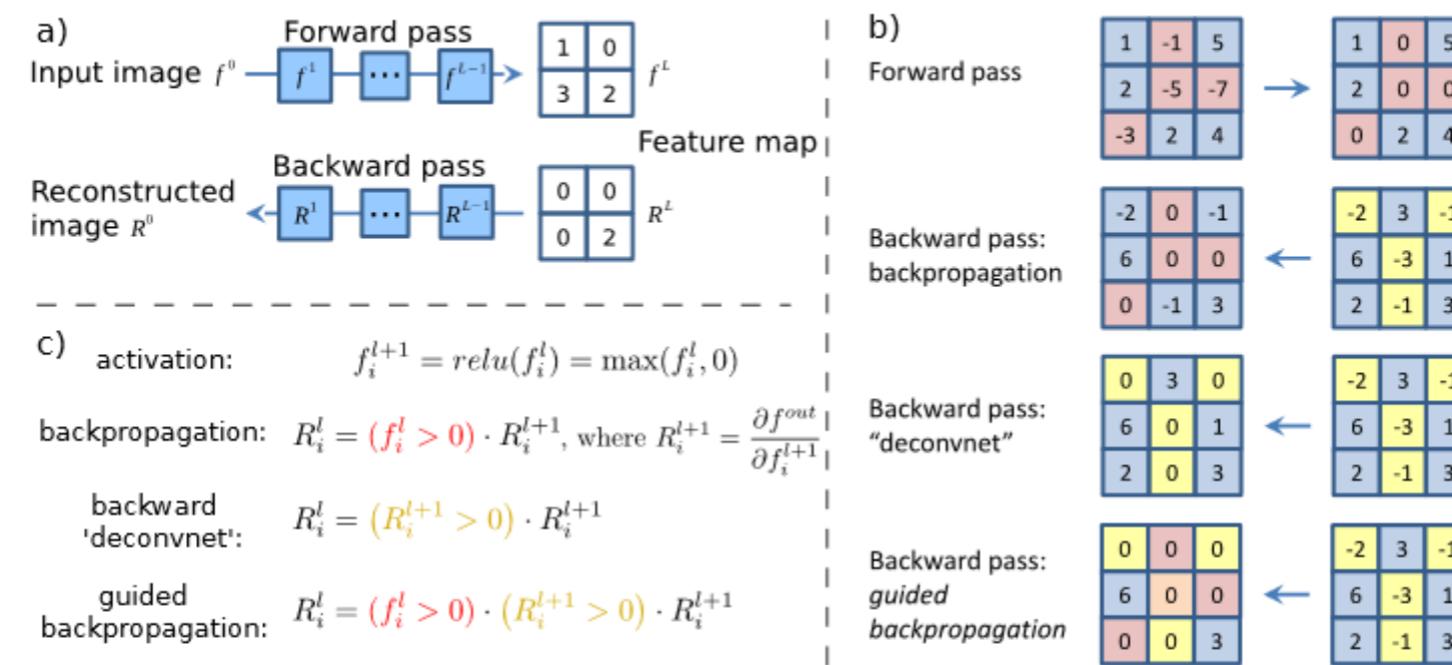


Figure 1: Schematic of visualizing the activations of high layer neurons. a) Given an input image, we perform the forward pass to the layer we are interested in, then set to zero all activations except one and propagate back to the image to get a reconstruction. b) Different methods of propagating back through a ReLU nonlinearity. c) Formal definition of different methods for propagating a output activation *out* back through a ReLU unit in layer l ; note that the 'deconvnet' approach and guided backpropagation do not compute a true gradient but rather an imputed version.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, Martin Riedmiller «Striving for simplicity:the all convolutional net» <https://arxiv.org/pdf/1412.6806.pdf>

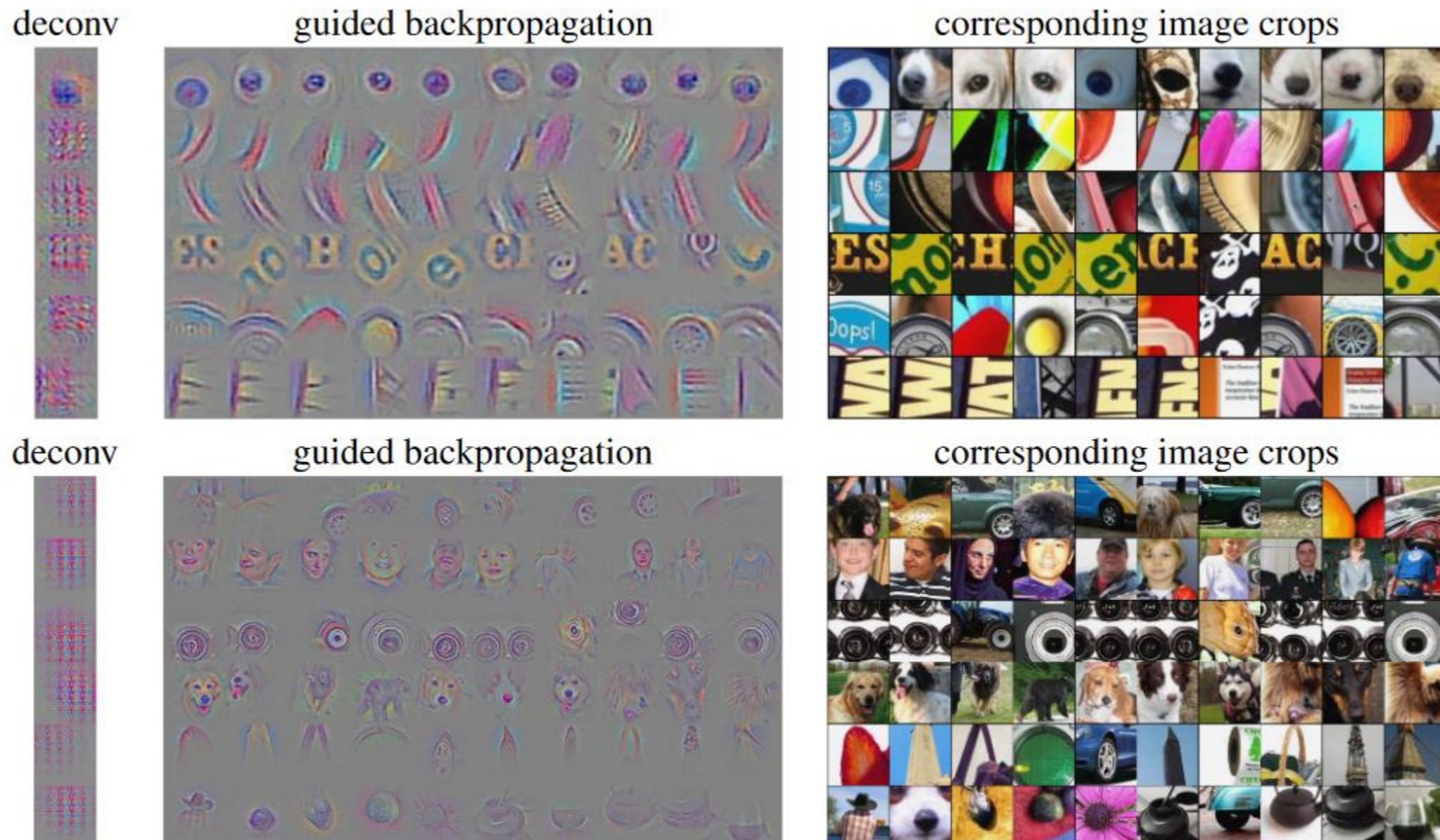


Figure 3: Visualization of patterns learned by the layer conv6 (top) and layer conv9 (bottom) of the network trained on ImageNet. Each row corresponds to one filter. The visualization using “guided backpropagation” is based on the top 10 image patches activating this filter taken from the ImageNet dataset. Note that image sizes are not preserved (in order to save space).

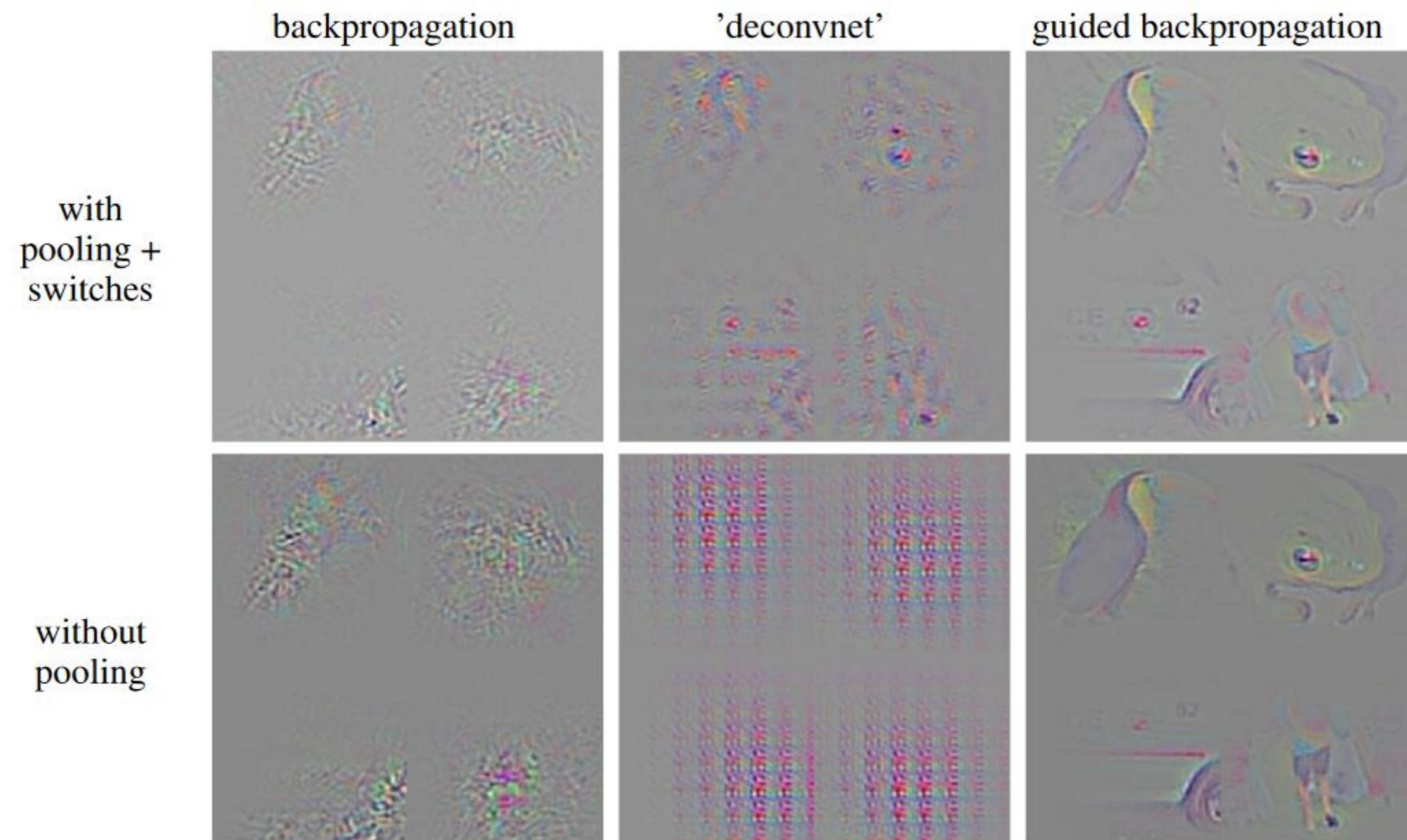


Figure 4: Visualization of descriptive image regions with different methods from the single largest activation in the last convolutional layer conv12 of the network trained on ImageNet. Reconstructions for 4 different images are shown.

Анализ отдельных нейронов

**Сгенерировать изображения,
которые максимизируют активацию выделенного нейрона
(методом обратного распространения ошибки)**

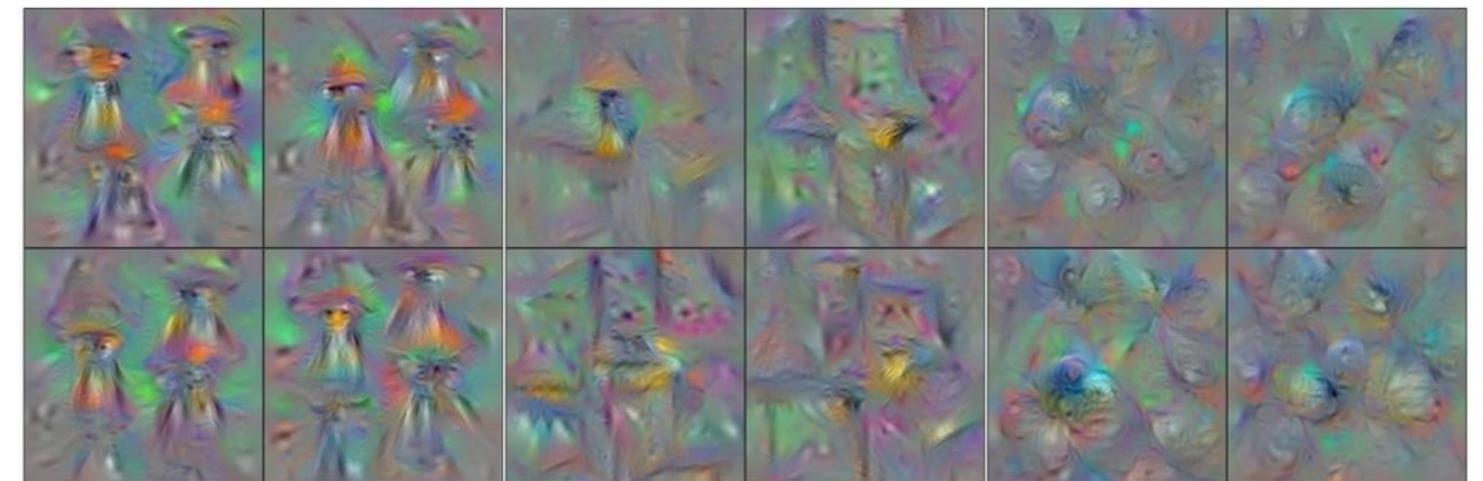
Анализ отдельных нейронов



Pirate Ship

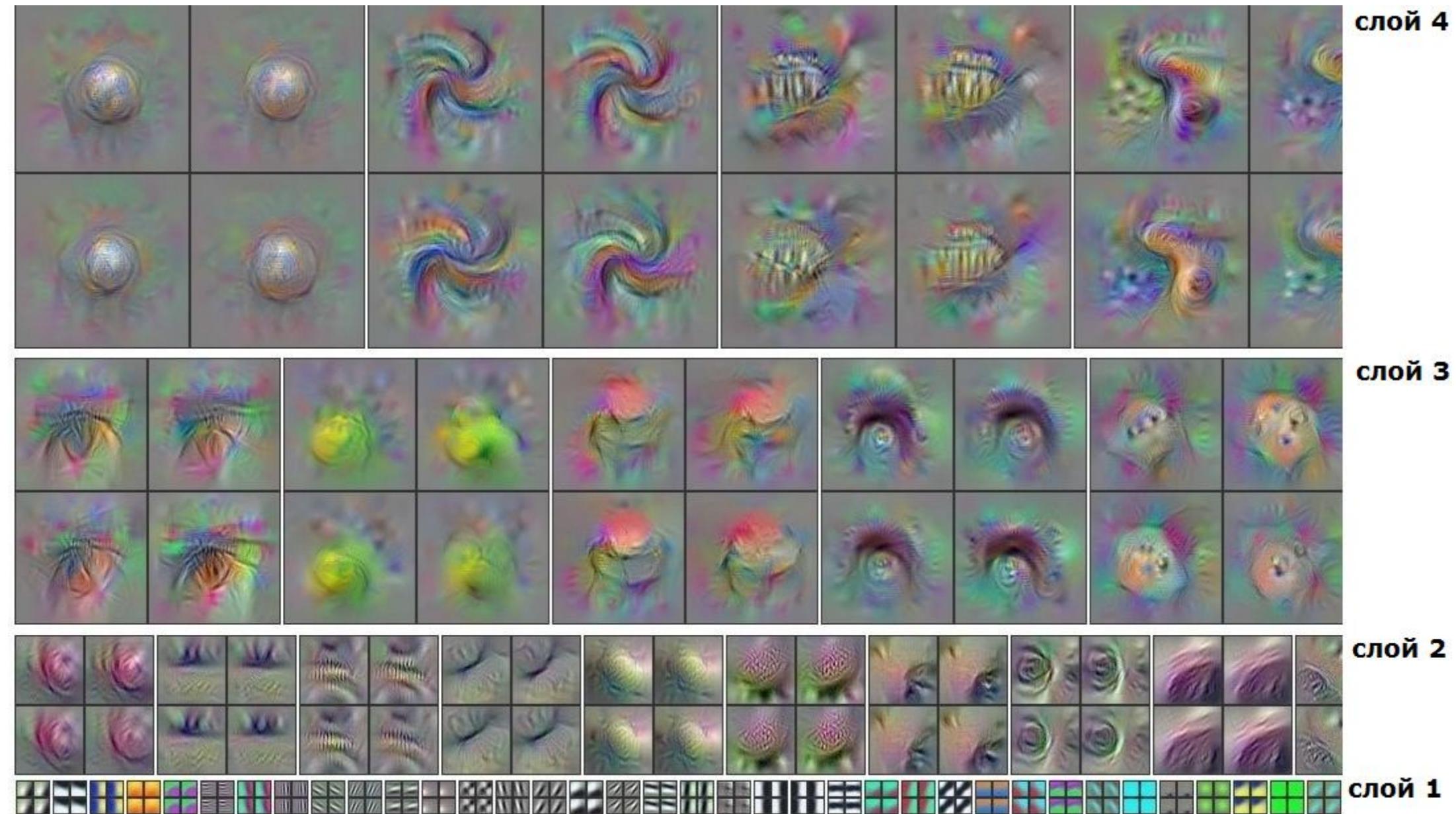
Rocking Chair

Teddy Bear



Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, Hod Lipson «Understanding Neural Networks Through Deep Visualization» // <https://arxiv.org/pdf/1506.06579.pdf>

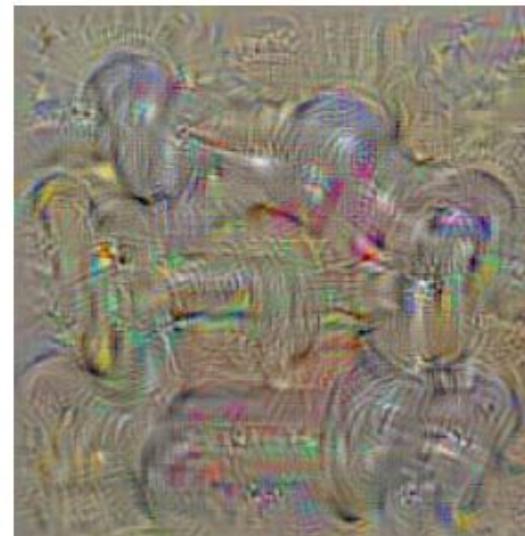
Анализ отдельных нейронов



Анализ отдельных нейронов: оптимизационный подход

**найти изображения, которые максимизируют оценку за определённый класс
+ использовать регуляризацию**

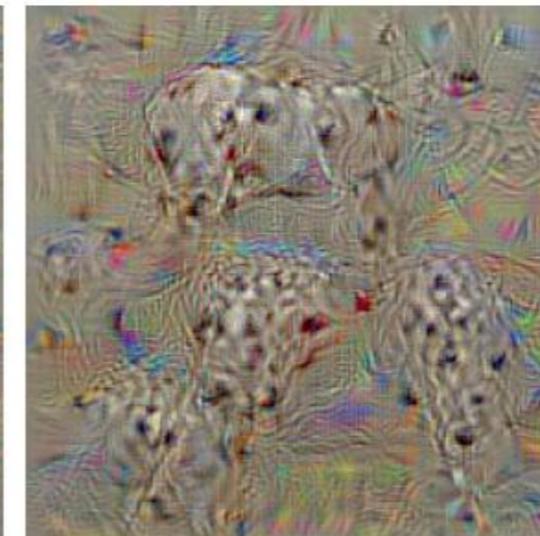
Оптимизационный подход



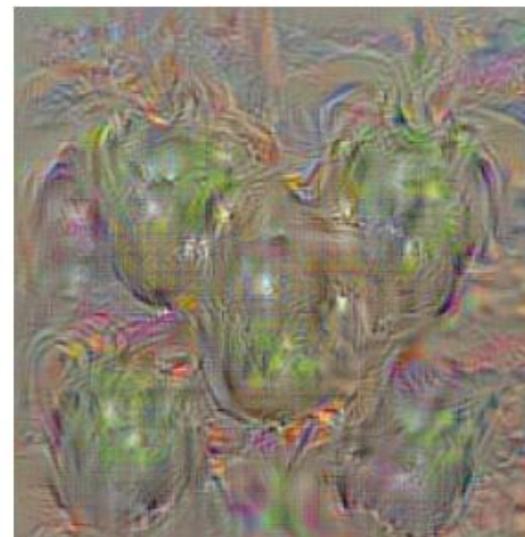
dumbbell



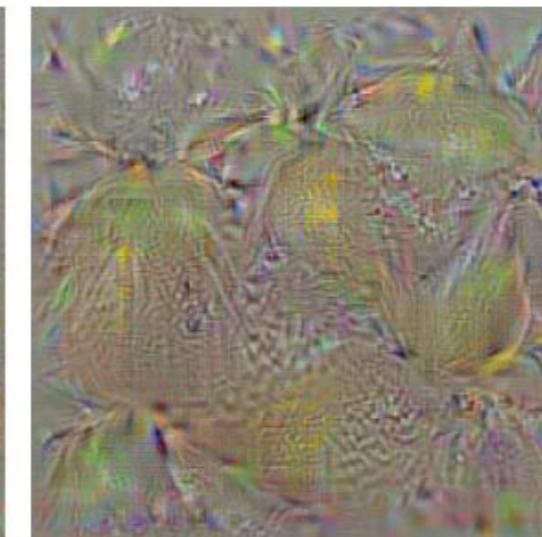
cup



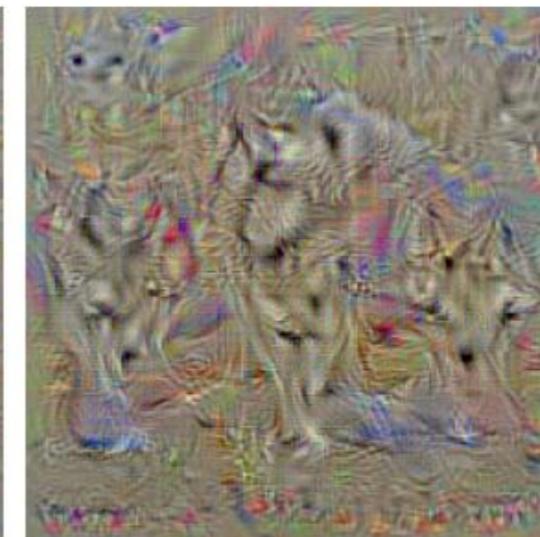
dalmatian



bell pepper



lemon

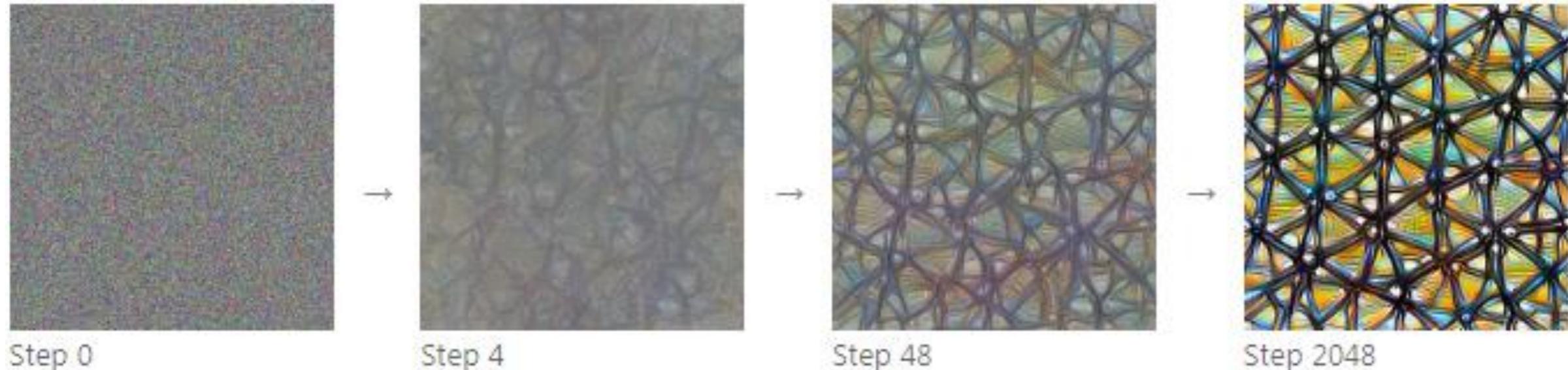


husky

Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps [Karen Simonyan et al 2014](#)

Визуализация нейронных сетей

когда с помощью обратного распространения смотрим на чём активируется нейрон..



Если попробовать сразу красиво не получится... грамотная регуляризация

- **наказывать разницу соседних пикселей**
- **размытие изображения после k шагов**
- **добавление устойчивости к некоторым преобразованиям**
- **априорное распределение на генерируемые изображения**
- **градиентный спуск в другом – декоррелируемом пространстве (в базисе Фурье)**

<https://distill.pub/2017/feature-visualization/>

За что отвечают отдельные нейроны, каналы, слои

Different **optimization objectives** show what different parts of a network are looking for.

n layer index

x,y spatial position

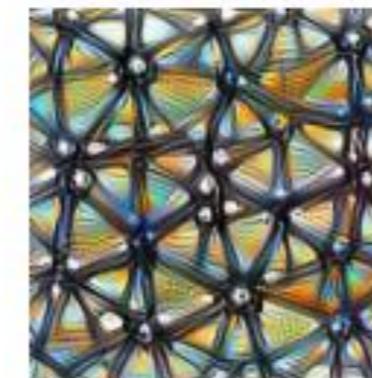
z channel index

k class index



Neuron

$\text{layer}_n[x, y, z]$



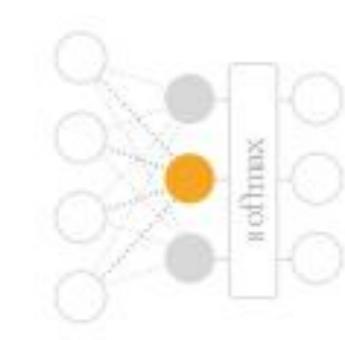
Channel

$\text{layer}_n[:, :, :, z]$



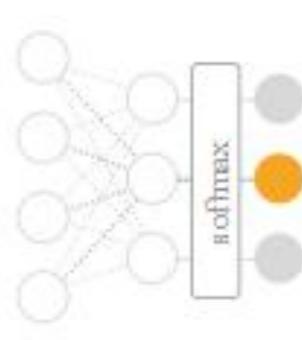
Layer/DeepDream

$\text{layer}_n[:, :, :, :]^2$



Class Logits

$\text{pre_softmax}[k]$



Class Probability

$\text{softmax}[k]$

Необходимость большого числа экспериментов с рандомизацией По одному эксперименту (прав. рис.) кажется, что «голова собаки»



Optimization with diversity. Layer mixed4a, Unit 143



Optimization with diversity reveals multiple types of balls. Layer mixed5a, Unit 9



Optimization with diversity show cats, foxes, but also cars. Layer mixed4e, Unit 55

Можно оптимизировать сразу несколько нейронов



Визуализация: семантические словари

{активация нейрона: его визуализация}



<https://distill.pub/2018/building-blocks/>

Современные методы

Suraj Srinivas, François Fleuret «Full-Gradient Representation for Neural Network Visualization» <https://arxiv.org/pdf/1905.00780.pdf>

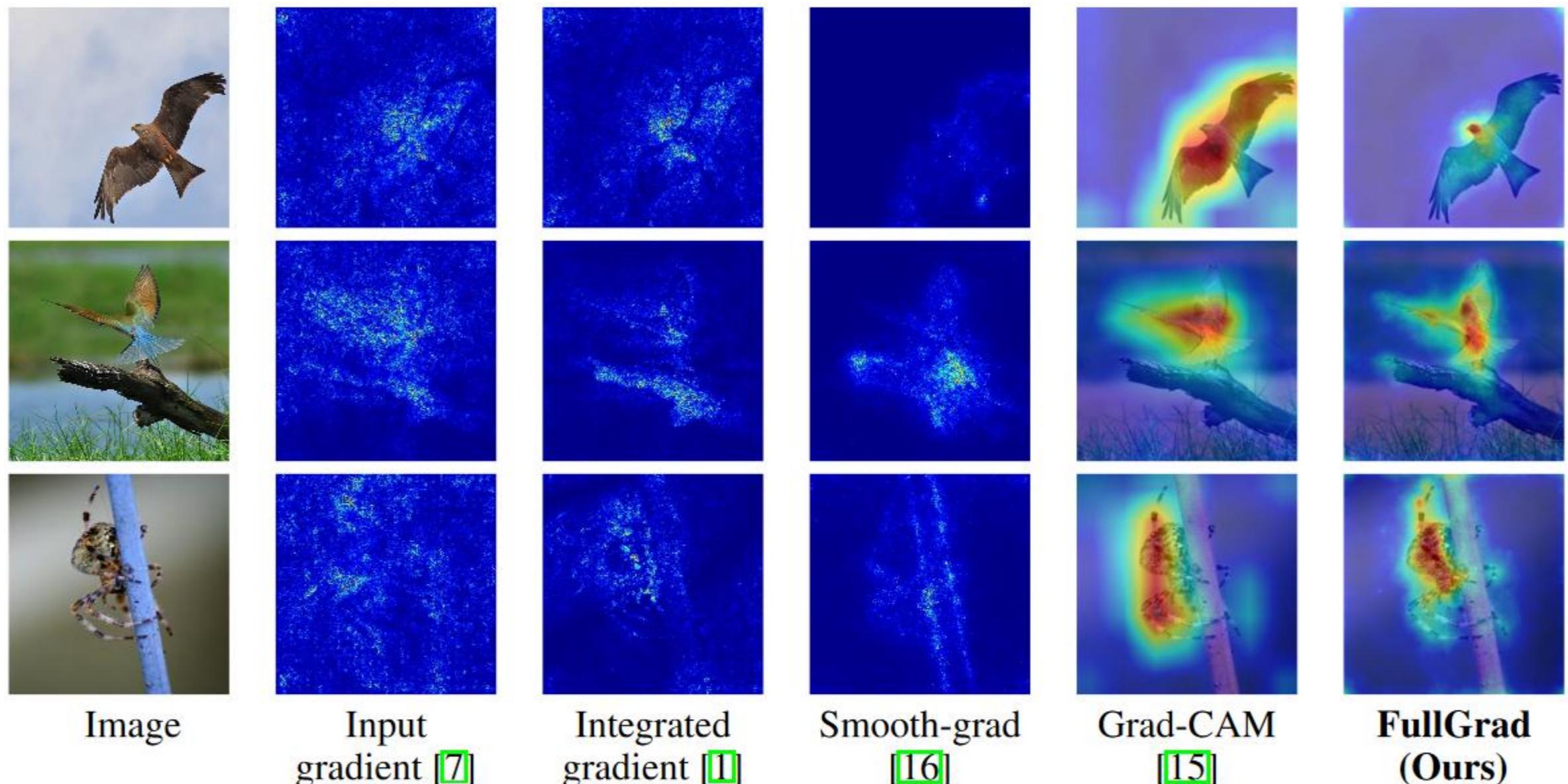
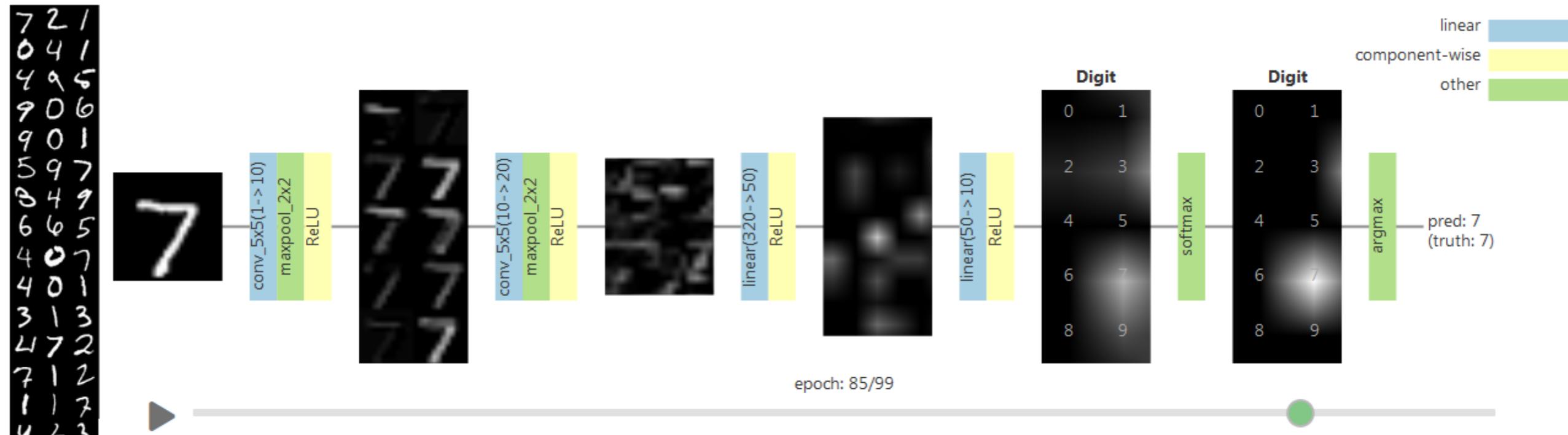


Figure 3: Comparison of different neural network saliency methods. Integrated-gradients [11] and smooth-grad [16] produce noisy object boundaries, while grad-CAM [15] indicates important regions without adhering to boundaries. FullGrad combine both desirable attributes by highlighting salient regions while being tightly confined within objects. For more results, please see supplementary

Современные методы



Neural network opened. The colored blocks are building-block functions (i.e. neural network layers), the gray-scale heatmaps are either the input image or intermediate activation vectors after some layers.

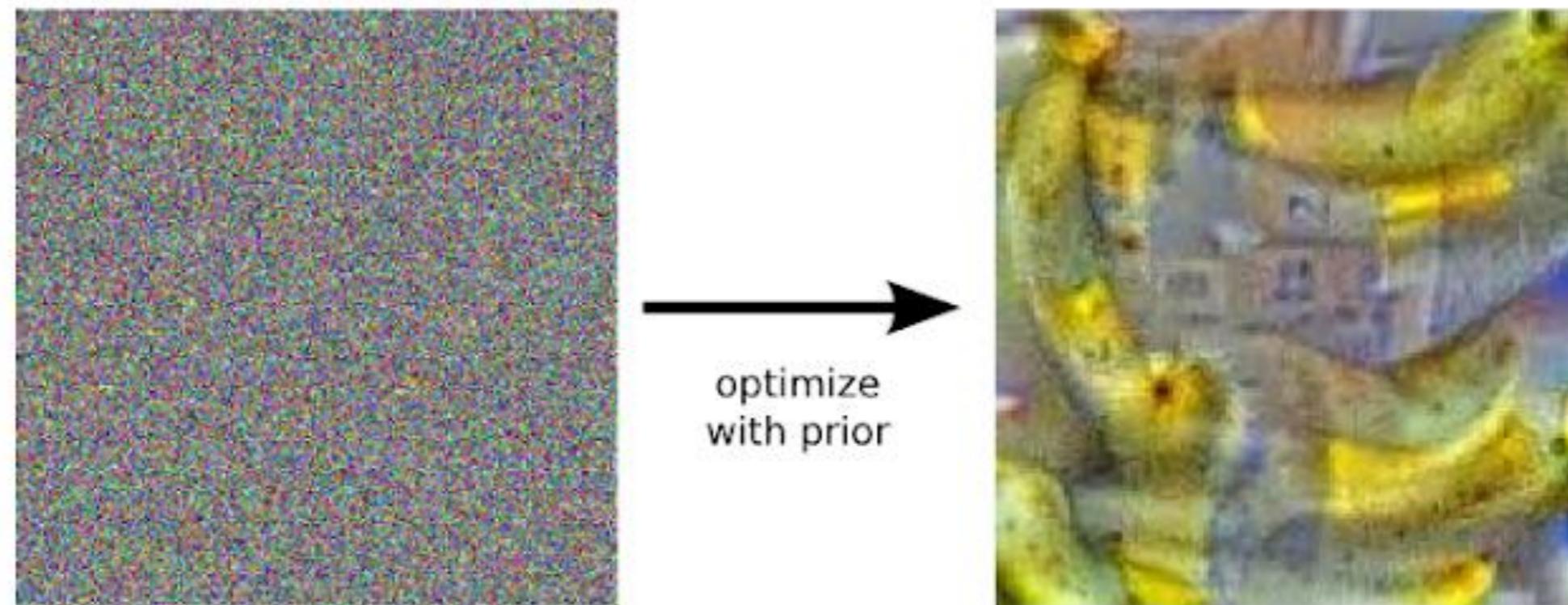
<https://distill.pub/2020/grand-tour/>

Генерация изображений

Натренировали НС-классификатор

**Дали на вход случайный шум и стали градиентно менять изображение:
увеличить уверенность, что это «банан»**

Есть хак: соседние пиксели коррелированы



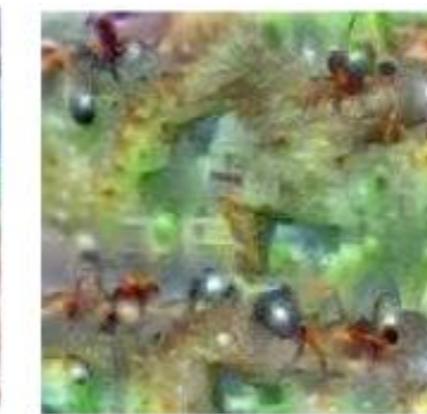
Генерация изображений



Hartebeest



Measuring Cup



Ant



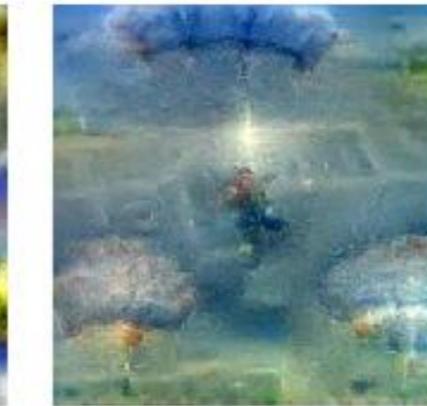
Starfish



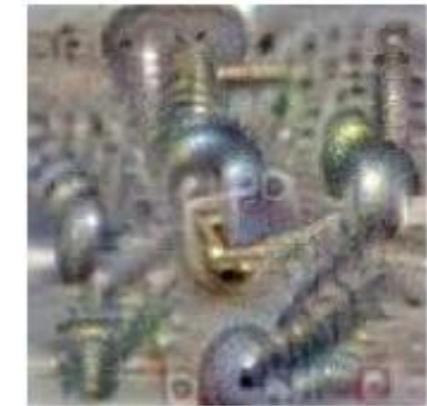
Anemone Fish



Banana



Parachute



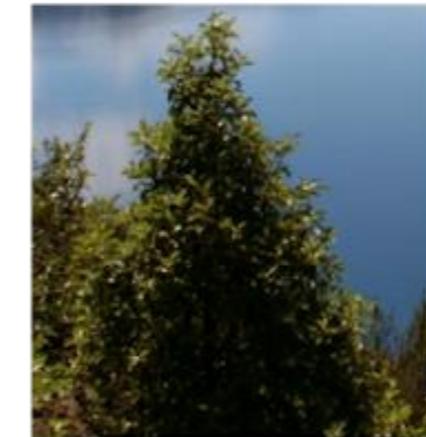
Screw

**Можно максимизировать не вероятность класса,
а активацию какого-то нейрона**

Генерация изображений



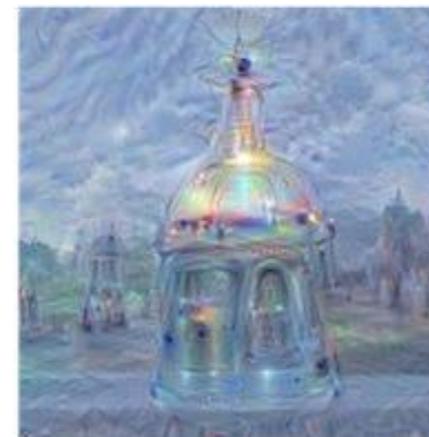
Horizon



Trees



Leaves



Towers & Pagodas



Buildings



Birds & Insects

Можем превращать изображения в изображения другого класса!

Генерация изображений

Зачем: чтобы понять, что выучились тому...



Класс «гантель» – всегда генерируется гантель с рукой!

<https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>

Генерация изображений: восстановление из признаков

по стилю, ориентации (относительно камеры), цвету, яркости и т.п.

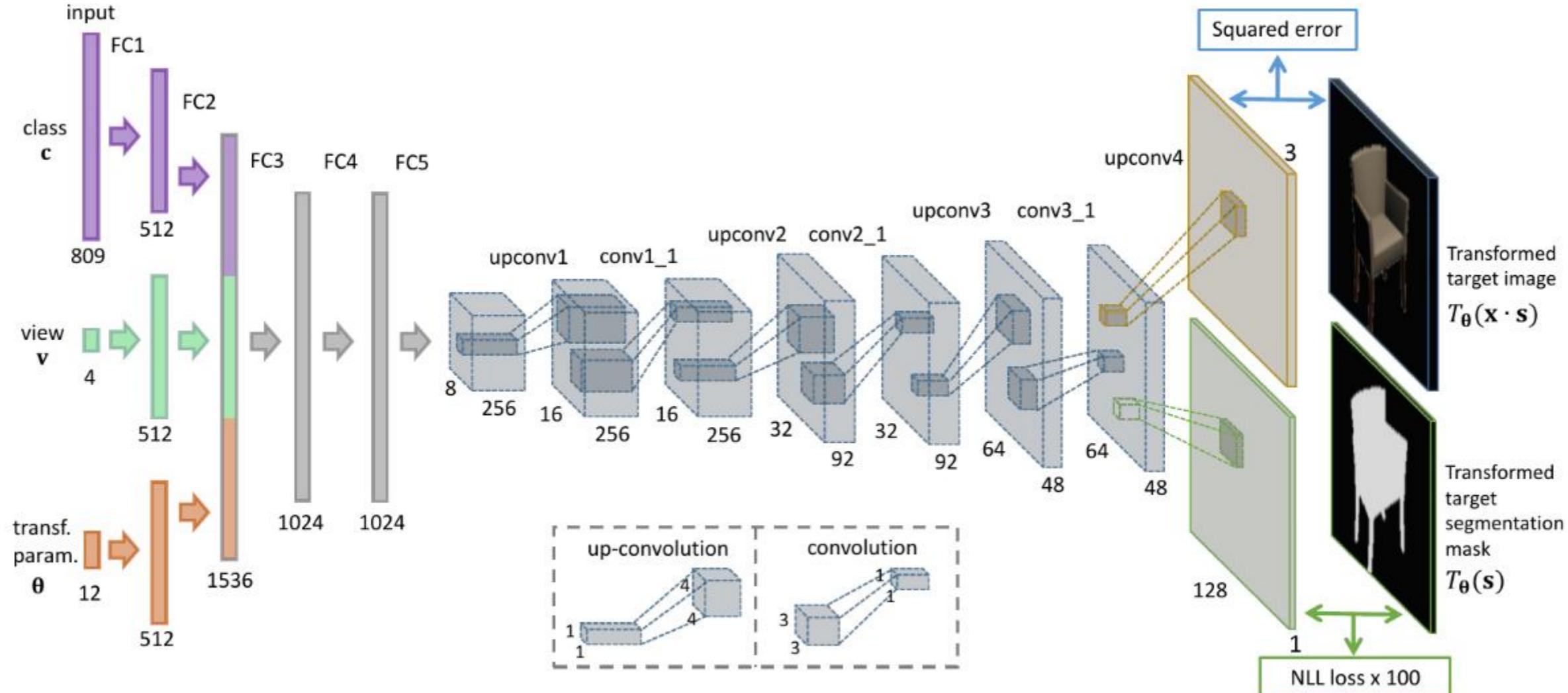


Fig. 1. Architecture of a 1-stream deep network ("1s-S-deep") that generates 128×128 pixel images. Layer names are shown above: FC - fully connected, upconv - upsampling+convolution, conv - convolution.

«развернутая CNN»: параметры → 1024 признака → изображение и маска сегментации

Генерация изображений: восстановление из признаков



Выборка:

(ONE-стиль, угол камеры, доп. параметры)



(изображение, маска сегментации)

**Маска для простоты
(можно забелить фон)**

Деконволюционные слои (deconvolutional layers)

Деконволюция (обратная операция к свёртке)

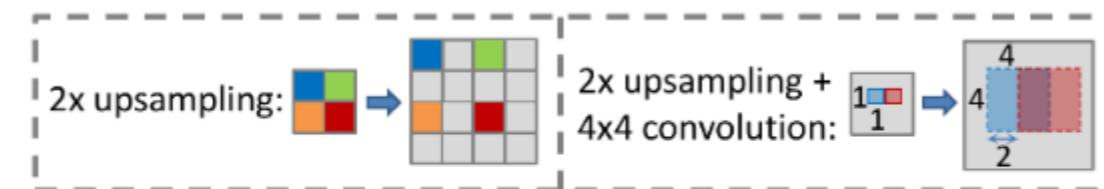


Fig. 2. Illustration of upsampling (left) and upsampling+convolution (right) as used in the generative network.

Dosovitskiy A. и др. «Learning to Generate Chairs, Tables and Cars with Convolutional Networks», 2017

// <https://arxiv.org/pdf/1411.5928.pdf>

Генерация изображений – преобразование эталонов



Fig. 7. Generation of chair images while activating various transformations. Each row shows one transformation: translation, rotation, zoom, stretch, saturation, brightness, color. The middle column shows the reconstruction without any transformation.

Генерация изображений – линейная комбинация стилей



Fig. 12. Examples of morphing different chairs, one morphing per row. Leftmost and rightmost chairs in each row are present in the training set, all intermediate ones are “invented” by the network. Rows are ordered by decreasing subjective quality of the morphing, from top to bottom.

Генерация изображений – Арифметика над признаками

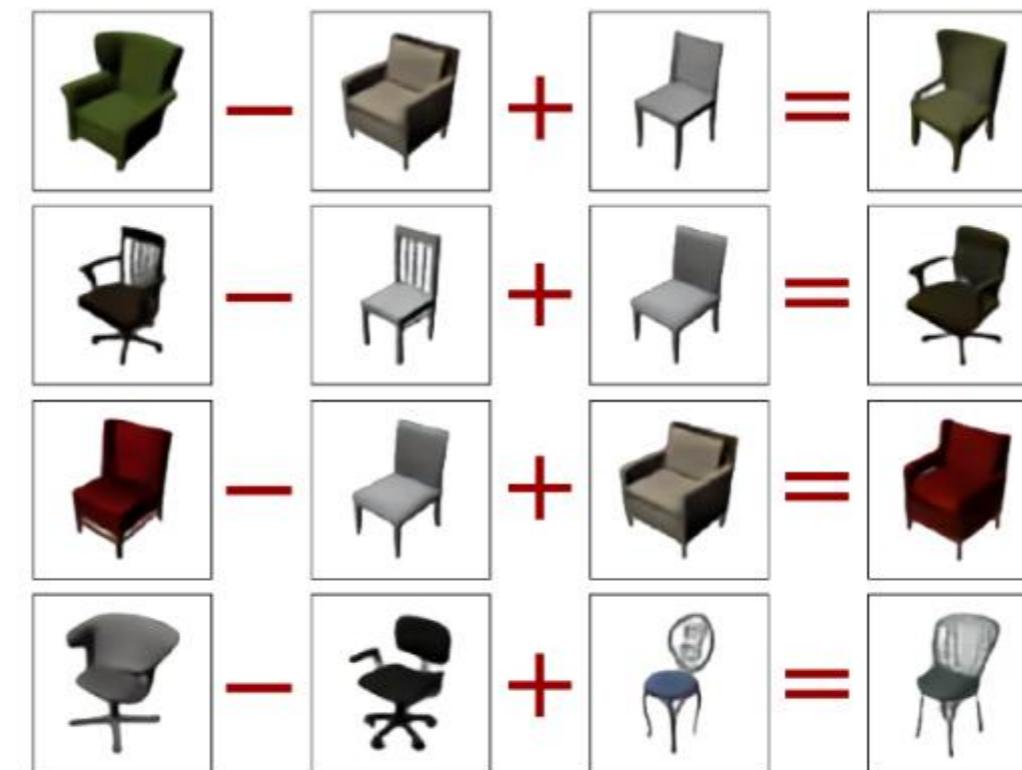
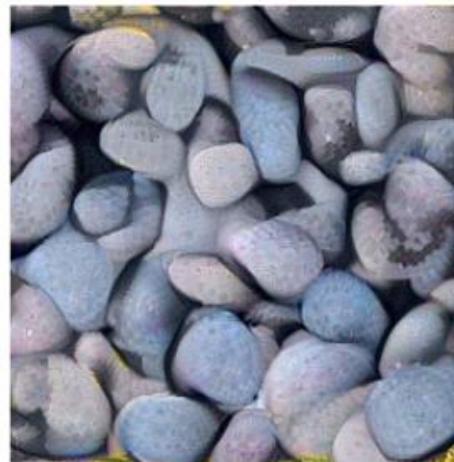


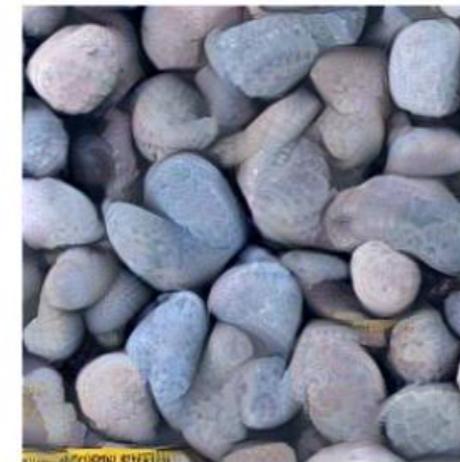
Fig. 16. Feature arithmetics: simple operations in the feature space lead to interpretable changes in the image space.

Генерация текстур

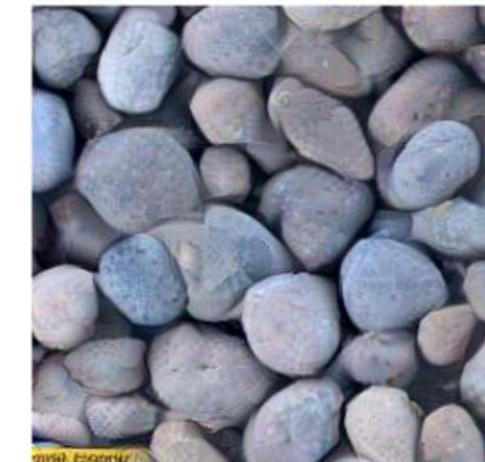
~1k parameters



~10k parameters



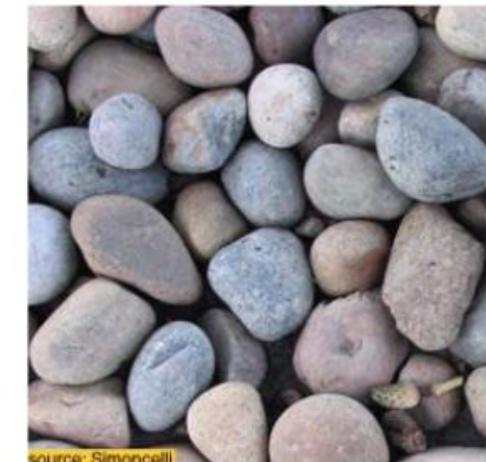
~177k parameters



~852k parameters



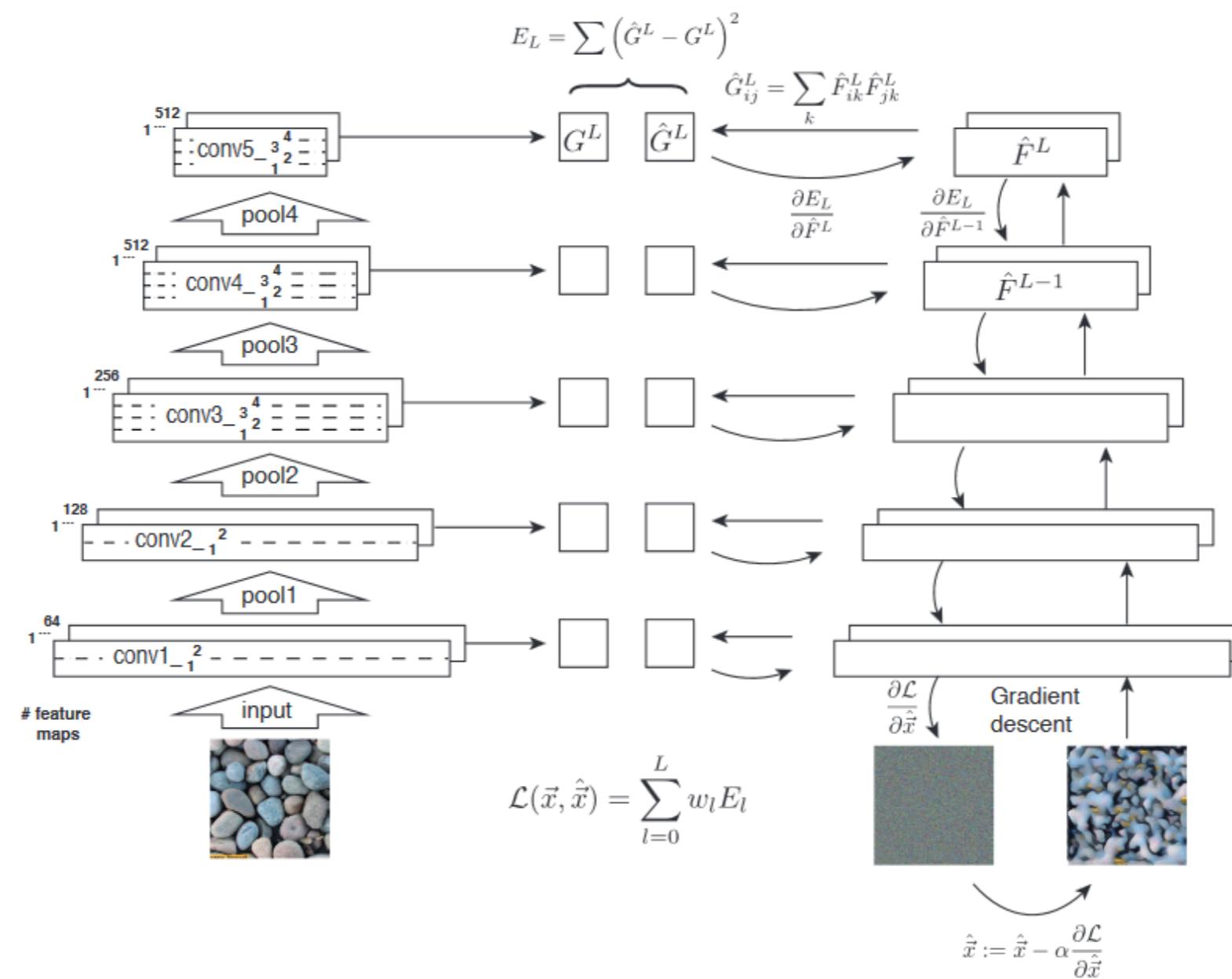
original



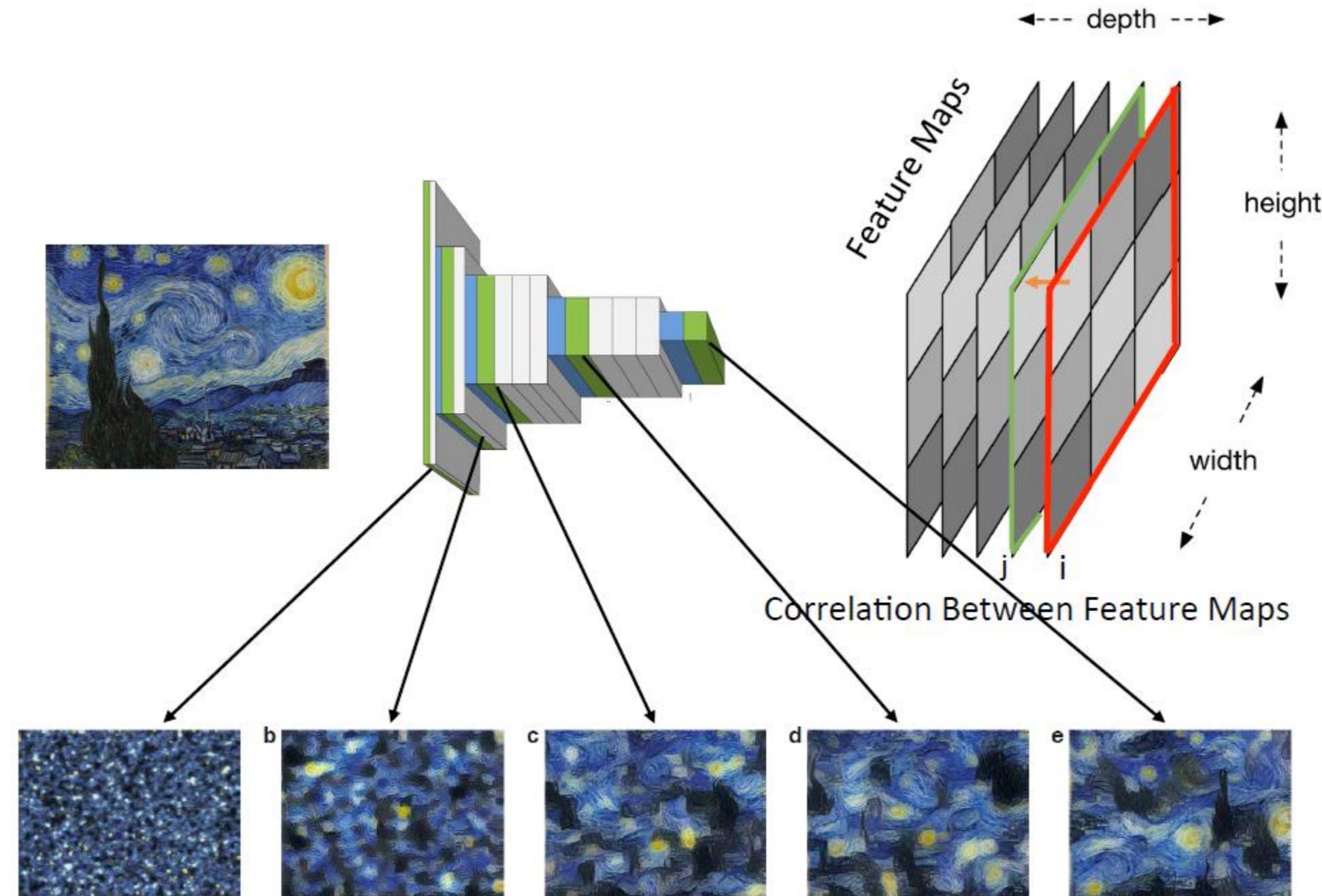
эксперимент по изменению числа параметров в модели

Leon A. Gatys «Texture Synthesis Using Convolutional Neural Networks» <https://arxiv.org/pdf/1505.07376.pdf>

Генерация текстур



Потом: «стиль изображения»



Генерация текстур

Идея:

Берём какую-нибудь CNN, например VGG-19 (её свёрточную часть)

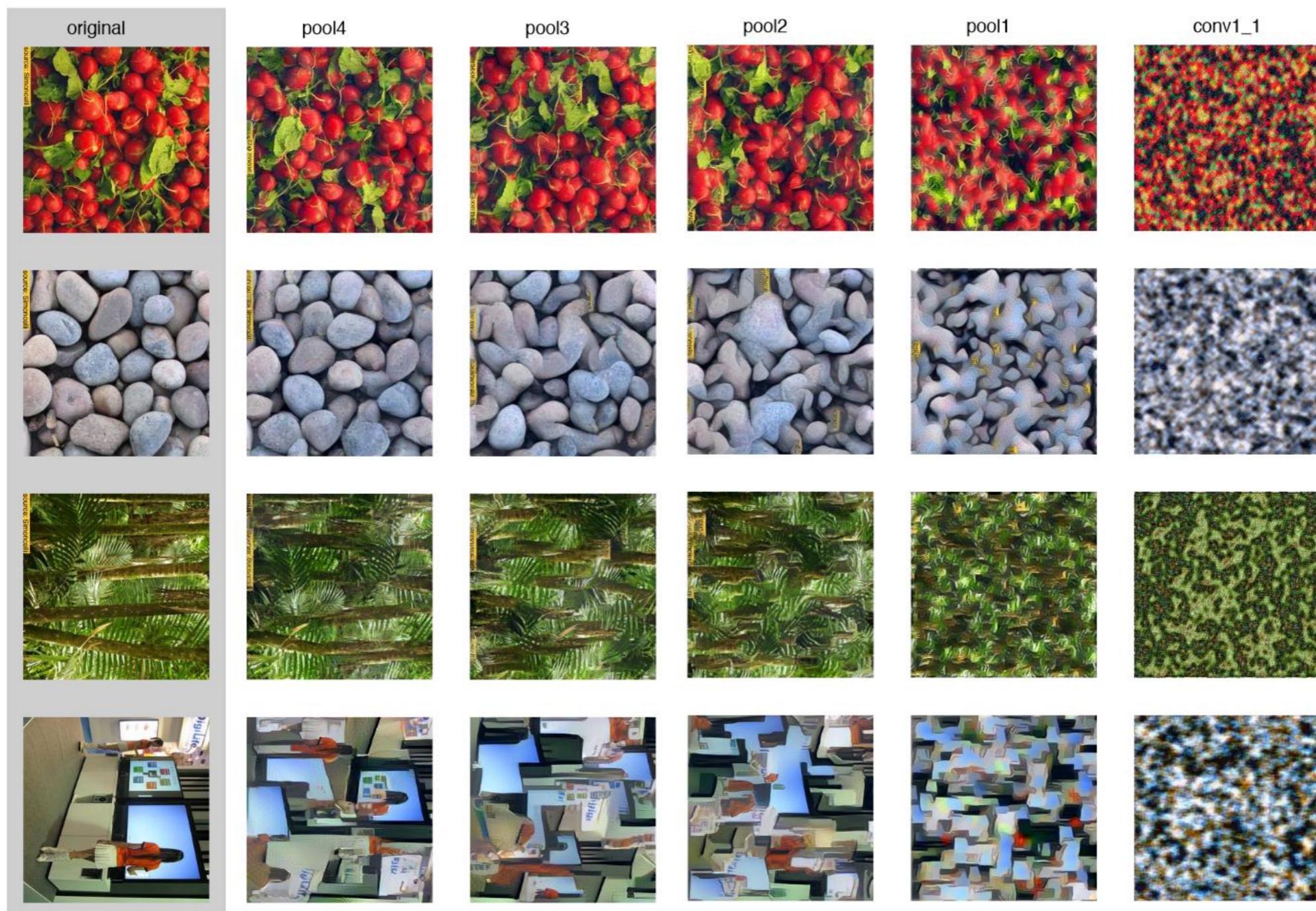
**На этой части изображение ~ тензор $w \times h \times k$
k – число каналов (м.б. 512)**

Считаем матрицу Грама $k \times k$

i,j -й элемент – корреляция wh -мерных векторизаций i -го канала и j -го

Вторая точно такая же сеть – на вход шум – тоже считаем матрицы Грама

**Теперь обучаем вторую НС (обучаемые параметры – само изображение) так,
чтобы её матрицы Грама были похожи на матрицы Грама первой НС**



Генерация пейзажей



<https://github.com/DmitryUlyanov/fast-neural-doodle>

Генерация пейзажей

Вход: картина + сегментация + новая сегментация

**Для каждого сегмента получаем описание стиля (матрицы Грама разных слоёв) и
генерируем новое изображение**

Стилизация

Что такое стиль?

Не должен зависеть от координат...

- 1) усредним признаки по пространственным измерениям;)**
- 2) сложнее – посчитаем ковариации (больше информации)**

**Идея: похожесть на фотографию + похожесть на стиль рисунка
= похожесть признакового описания + похожесть матриц Грама**

**Leon A. Gatys и др. «A Neural Algorithm of Artistic Style», 2015 //
<https://arxiv.org/pdf/1508.06576.pdf>**

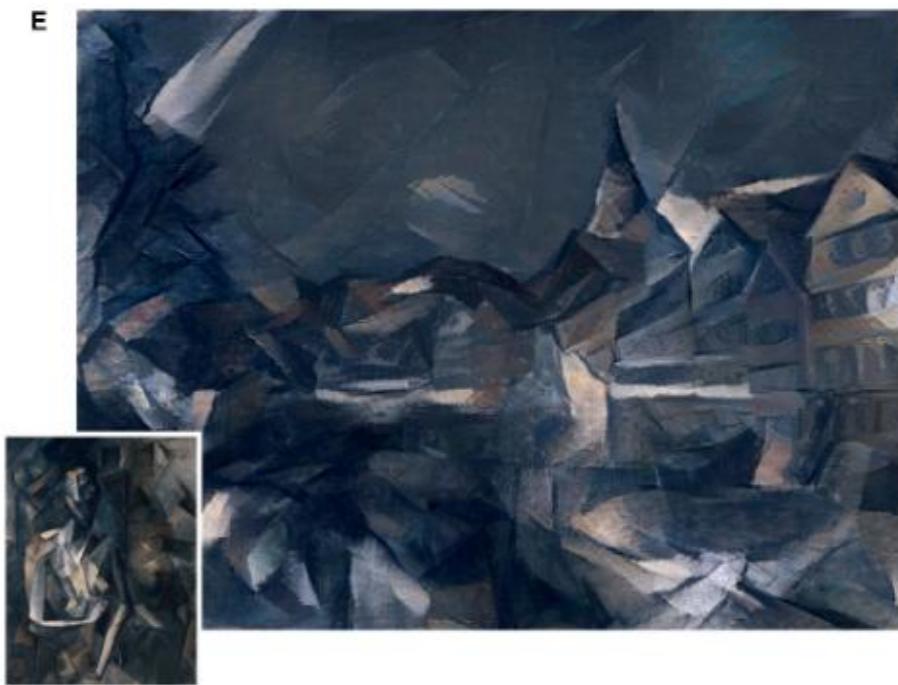
A



B



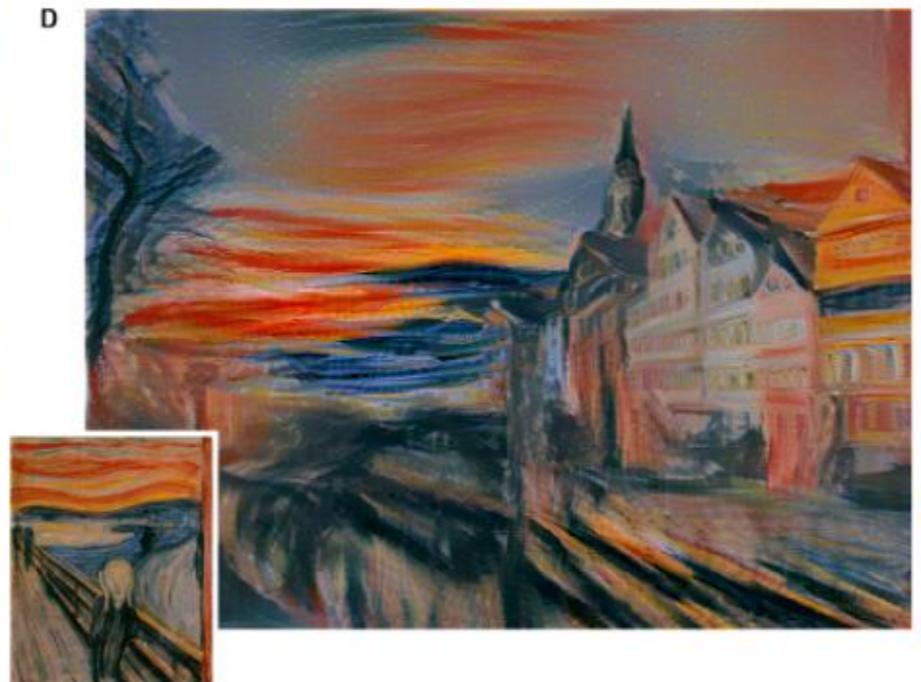
E



C



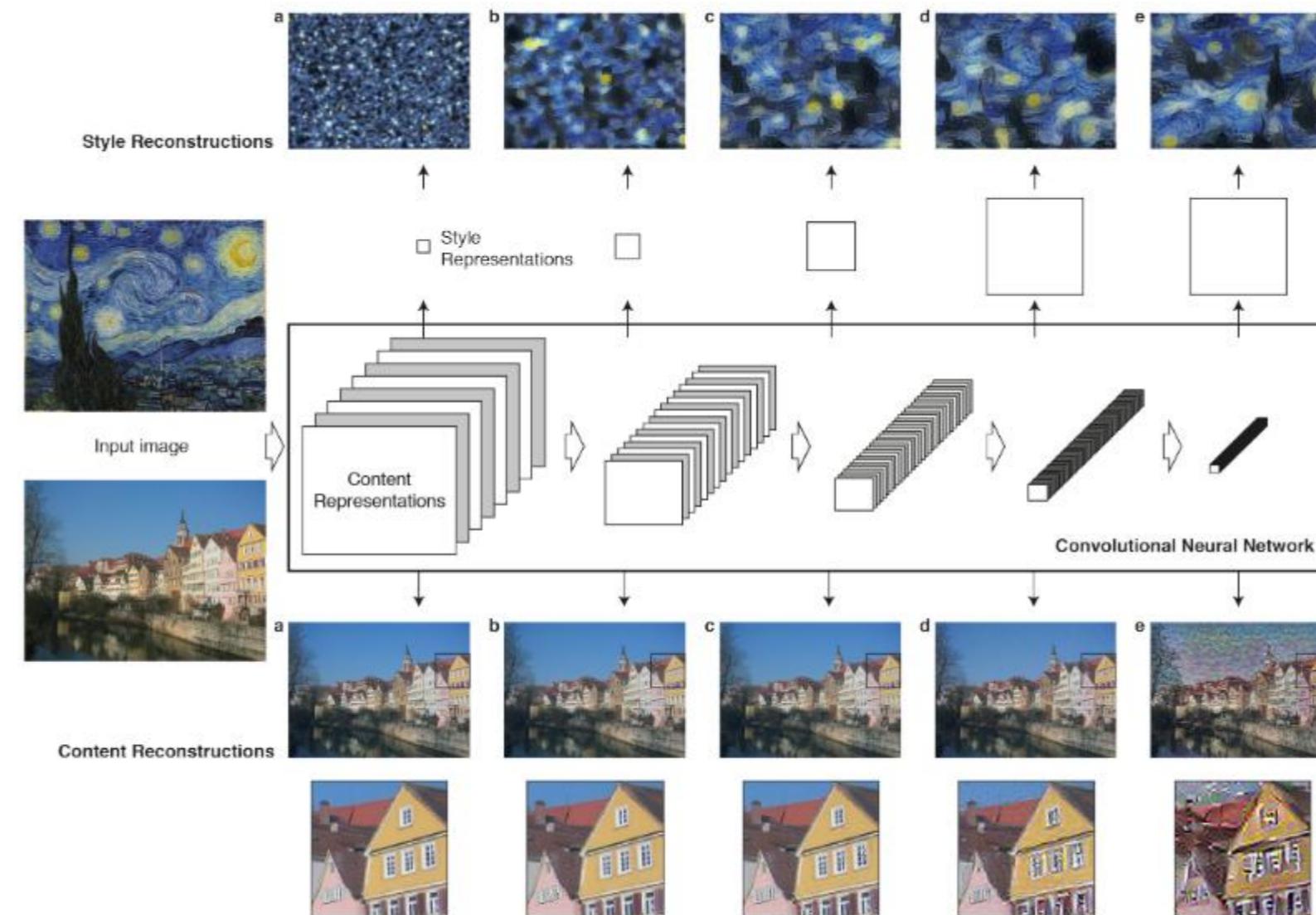
D



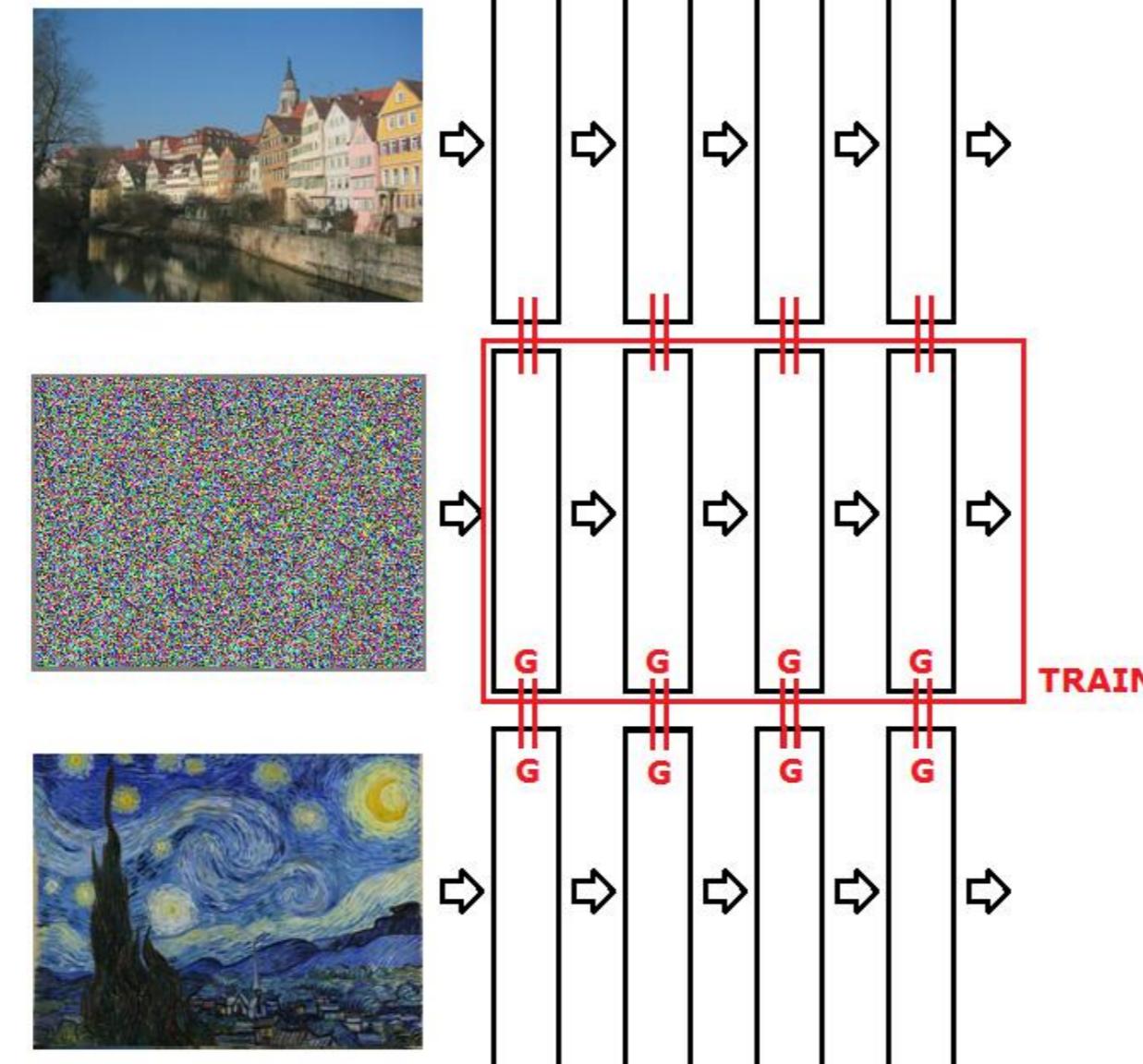
F



Стилизация

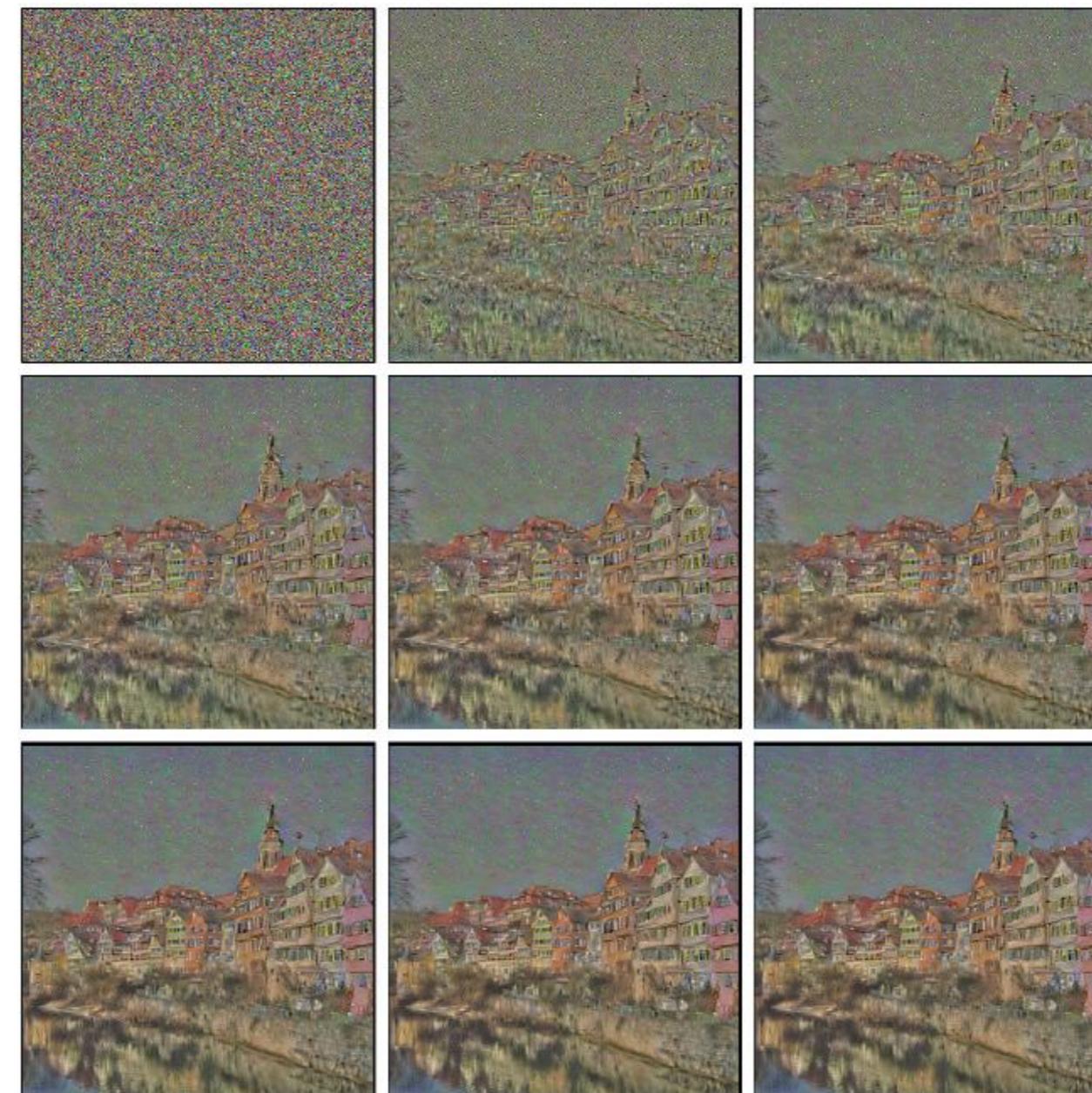


Стилизация

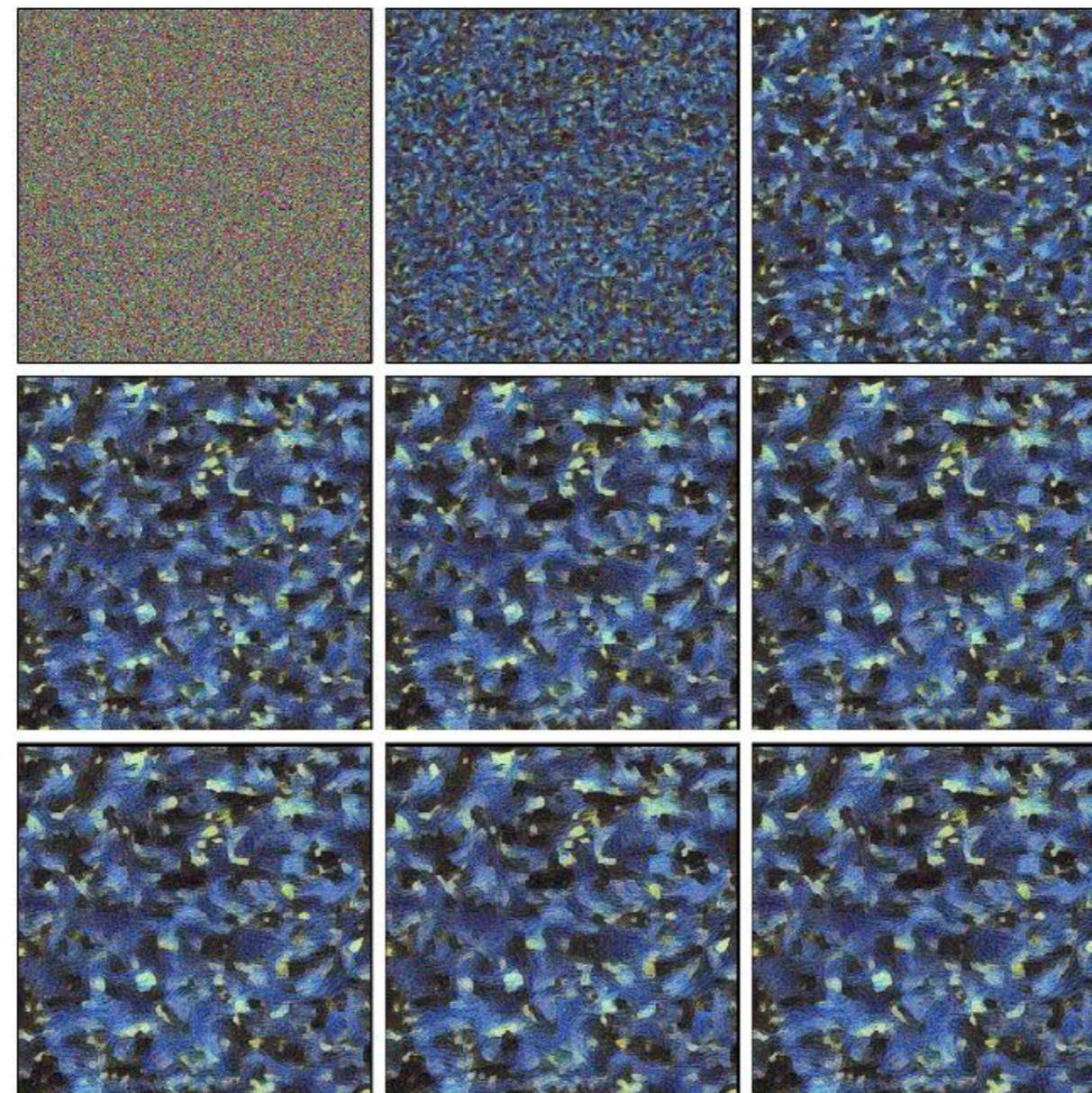


на самом деле с равенствами не совсем так...

Оптимизация по содержанию

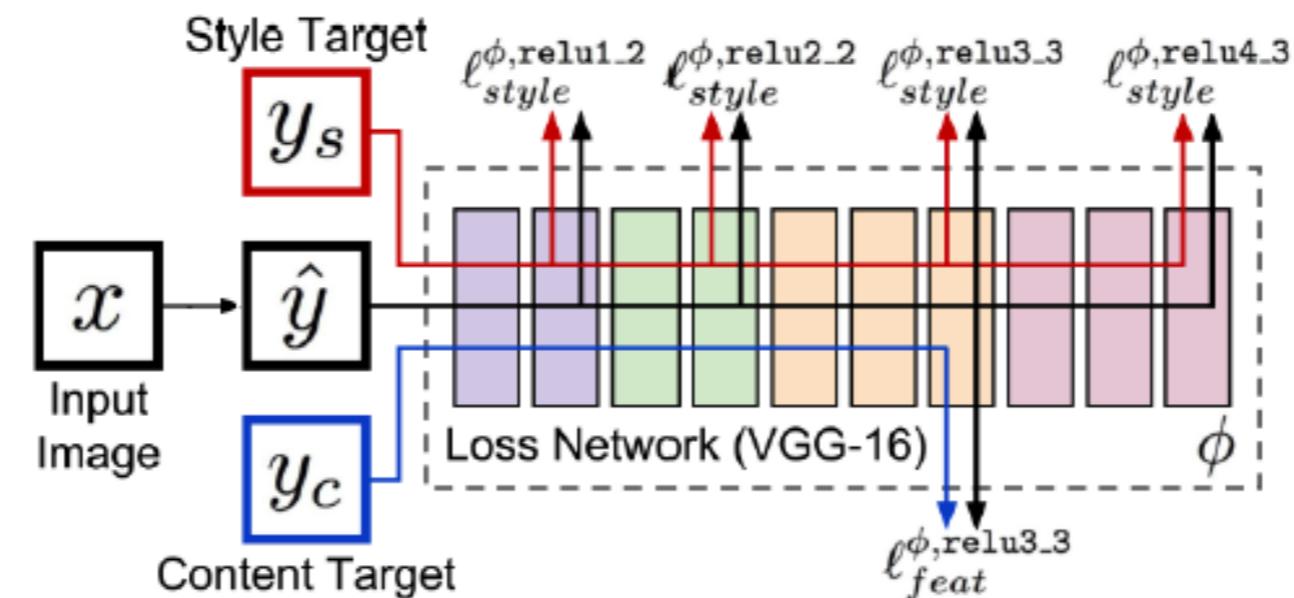


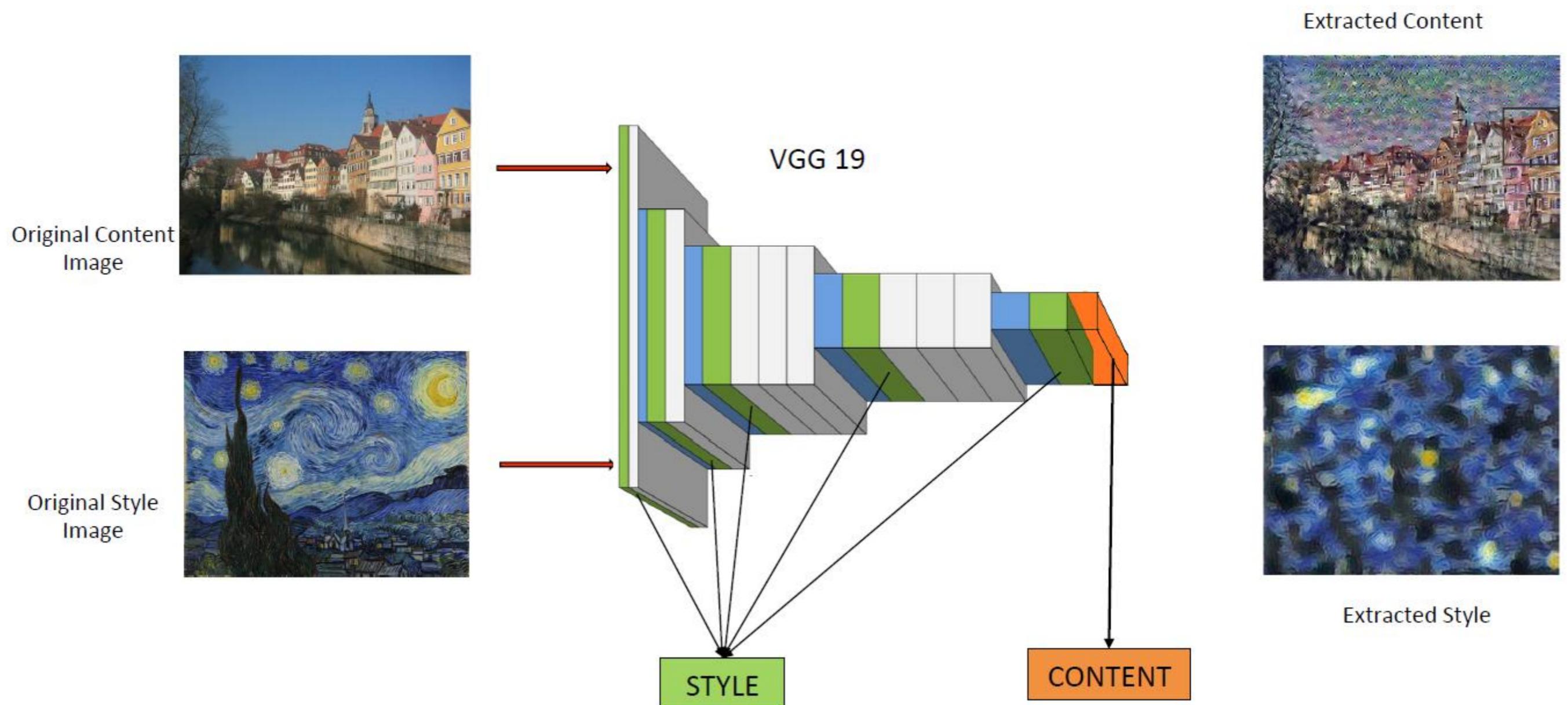
Оптимизация по стилю



Стилизация

$$\mathcal{L}_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{content}(\vec{p}, \vec{x}) + \beta \mathcal{L}_{style}(\vec{a}, \vec{x})$$





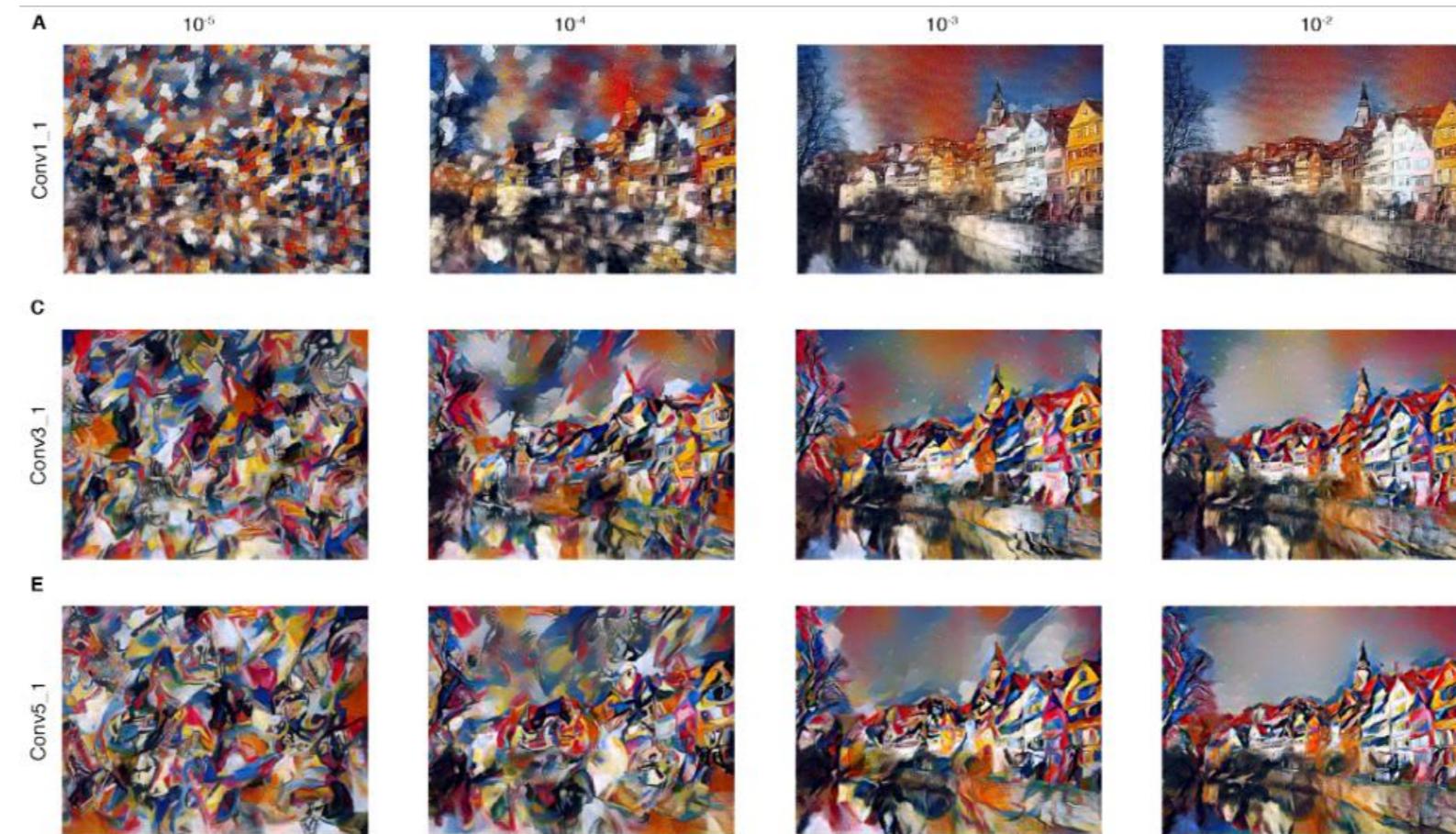
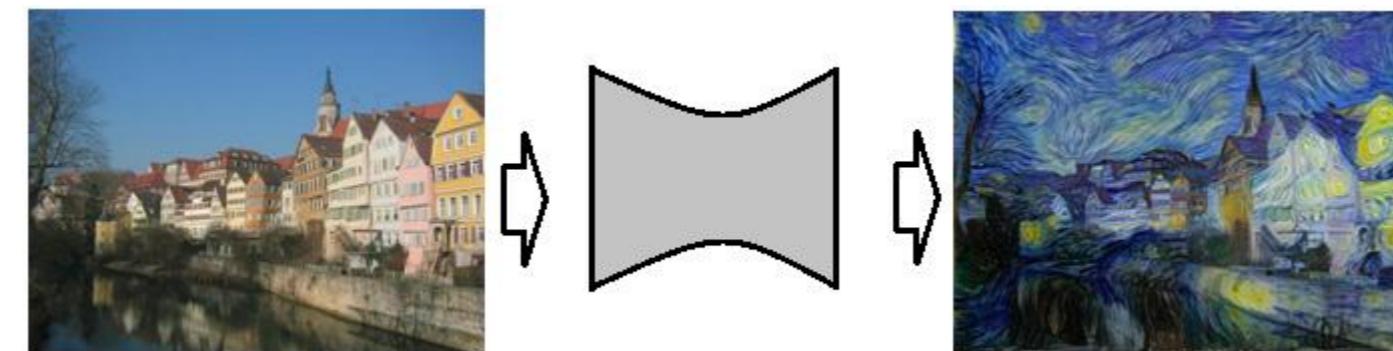


Figure 3: Detailed results for the style of the painting *Composition VII* by Wassily Kandinsky. The rows show the result of matching the style representation of increasing subsets of the CNN layers (see Methods). We find that the local image structures captured by the style representation increase in size and complexity when including style features from higher layers of the network. This can be explained by the increasing receptive field sizes and feature complexity along the network's processing hierarchy. The columns show different relative weightings between the content and style reconstruction. The number above each column indicates the ratio α/β between the emphasis on matching the content of the photograph and the style of the artwork (see Methods).

Быстрая стилизация



**пусть будет всего одна сеть,
но она умеет делать конкретную стилизацию!**

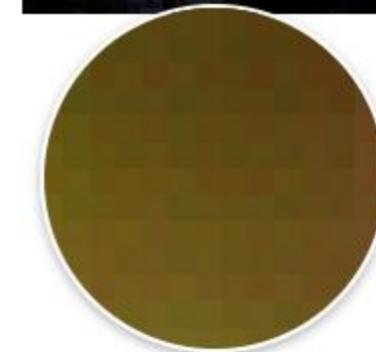
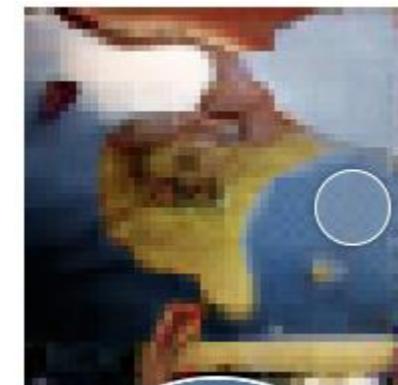
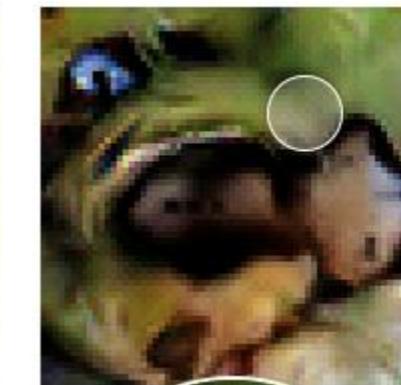
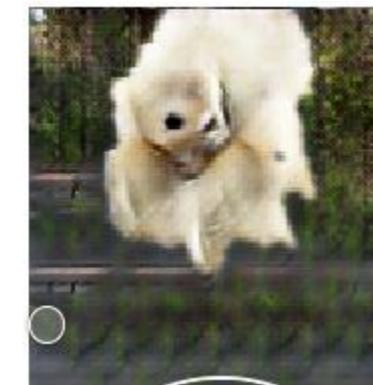
Кстати,

- **зашумлять изображения, чтобы была устойчива к шуму**

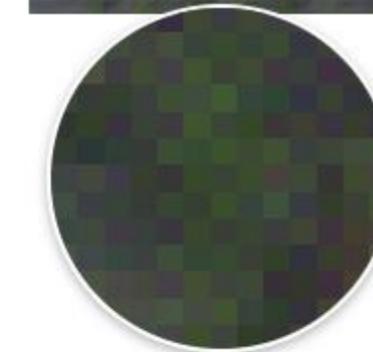
[Дмитрий Ульянов]

Дальше – GAN

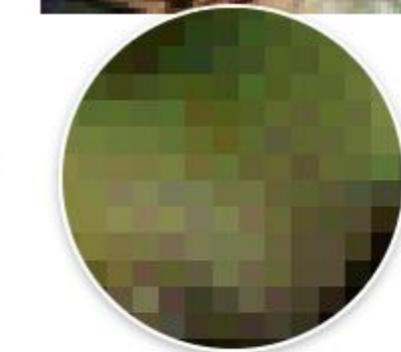
Причина шахматных сеток на изображениях



Radford, et al., 2015 [1]



Salimans et al., 2016 [2]



Donahue, et al., 2016 [3]

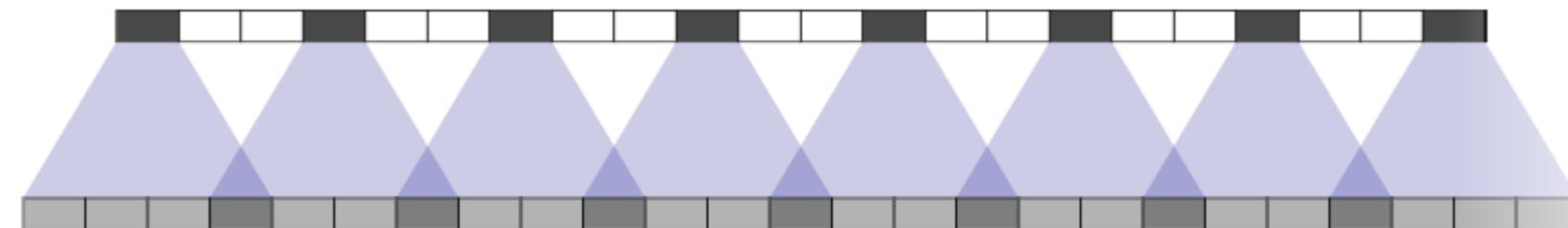


Dumoulin, et al., 2016 [4]

<https://distill.pub/2016/deconv-checkerboard/>

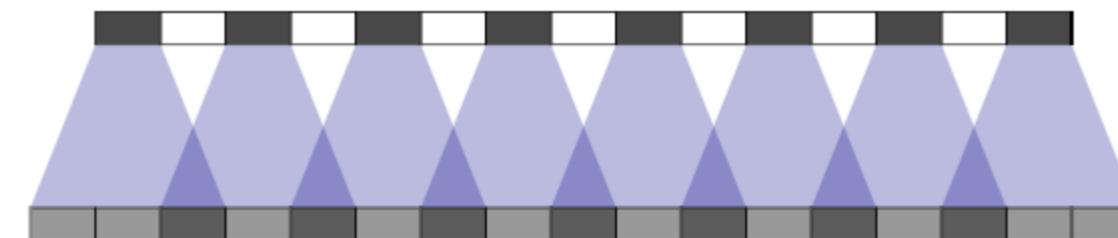
Причина шахматных сеток на изображениях

Как происходит обратная свёртка:



stride = 3

size = 4



stride = 2

size = 3

Дистилляция данных

Model distillation – построение простой модели на основе сложной / нескольких моделей связано с компрессией (model compression) Hinton et al. (2015)

Dataset Distillation – фиксируем модель и пытаемся построить (!) простой набор данных, который заменит имеющийся сложный

**связано с сокращением набора данных
(dataset pruning / core-set construction / instance selection)**

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, Alexei A. Efros «Dataset Distillation» //

<https://arxiv.org/abs/1811.10959>

<https://github.com/SsnL/dataset-distillation>

Дистилляция данных: зачем

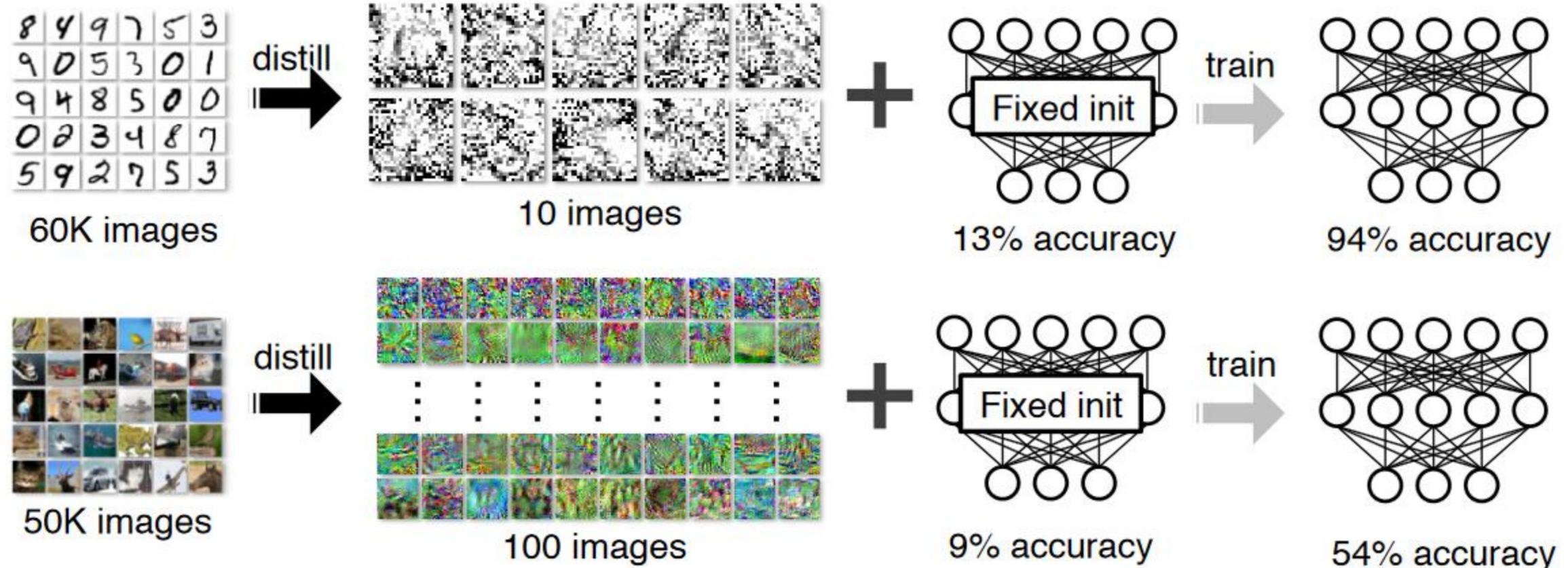
- **ускорение обучения**
- **сокращение объёма для хранения данных**
- **научный вопрос: сколько информации содержится в данных**
 - ~ насколько сможем «ужать»

Что удалось:

**сжать 60 000 MNIST в 10 синтетических изображений,
по одному на каждый класс**

Для фиксированной инициализации данная сеть сходится на этом наборе за несколько шагов к максимальному (изначальному) качеству

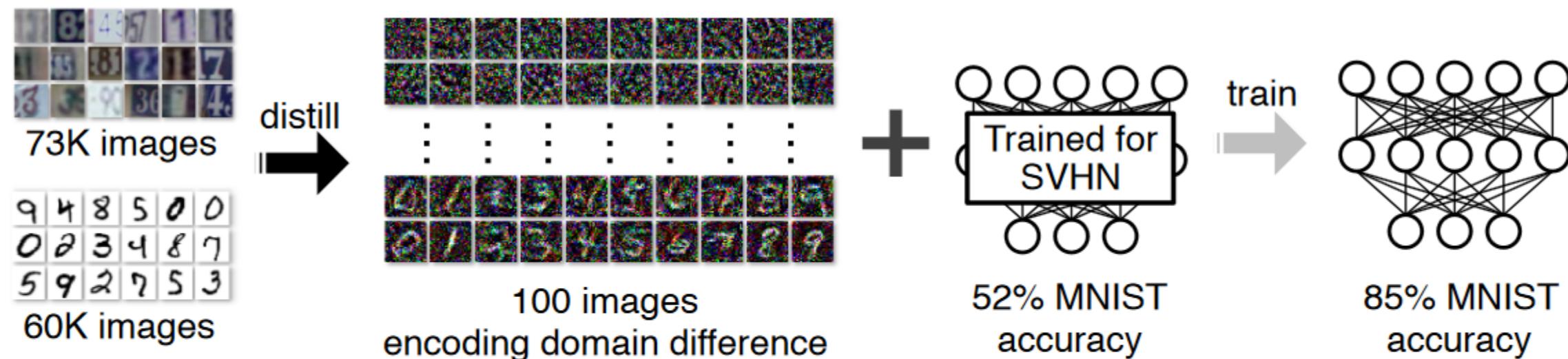
Дистилляция данных: результаты



(a) Dataset distillation on MNIST and CIFAR10

Figure 1: We distill the knowledge of tens of thousands of images into a few synthetic training images called distilled images. (a) On MNIST, 10 distilled images can train a standard LENET with a fixed initialization to 94% test accuracy, compared to 99% when fully trained. On CIFAR10, 100 distilled images can train a network with a fixed initialization to 54% test accuracy, compared to 80% when fully trained.

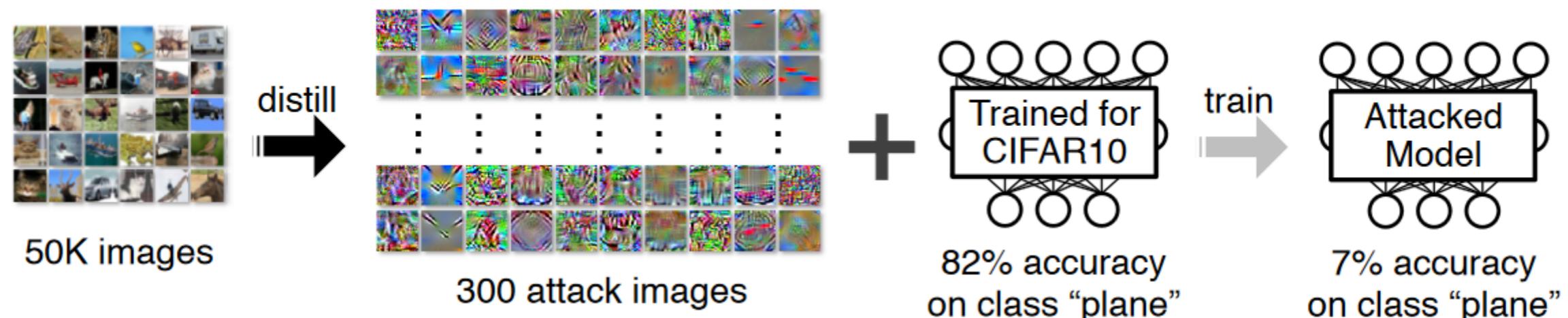
Дистилляция данных: результаты



(b) Dataset distillation can quickly fine-tune pre-trained networks on new datasets

(b) We distill the domain difference between SVHN and MNIST into 100 images. These images can quickly fine-tune pre-trained SVHN networks to achieve high accuracy on MNIST.

Дистилляция данных: результаты



(c) Dataset distillation can maliciously attack classifier networks

(c) Our formulation can create dataset poisoning images. After trained with these images for *one single* gradient step, networks will catastrophically misclassify one category.

**Ещё эксперименты по созданию изображений,
которые за один шаг портят настройку по конкретному классу**

Дистилляция данных: подход

$$\theta^* = \arg \min_{\theta} \frac{1}{N} \sum_{i=1}^N \ell(x_i, \theta) \triangleq \arg \min_{\theta} \ell(\mathbf{x}, \theta),$$

Standard training

At each step t ,
a minibatch of training data $\mathbf{x}_t = \{x_{t,j}\}_{j=1}^n$ is sampled to update the current parameters as

$$\theta_{t+1} = \theta_t - \eta \nabla_{\theta_t} \ell(\mathbf{x}_t, \theta_t),$$

Instead, we aim to learn a tiny set of synthetic distilled training data $\tilde{\mathbf{x}} = \{\tilde{x}_i\}_{i=1}^M$ with $M \ll N$ and a corresponding learning rate $\tilde{\eta}$ so that a single GD step such as

$$\theta_1 = \theta_0 - \tilde{\eta} \nabla_{\theta_0} \ell(\tilde{\mathbf{x}}, \theta_0) \tag{2}$$

using these learned synthetic data $\tilde{\mathbf{x}}$ can greatly boost the performance on the real test set.

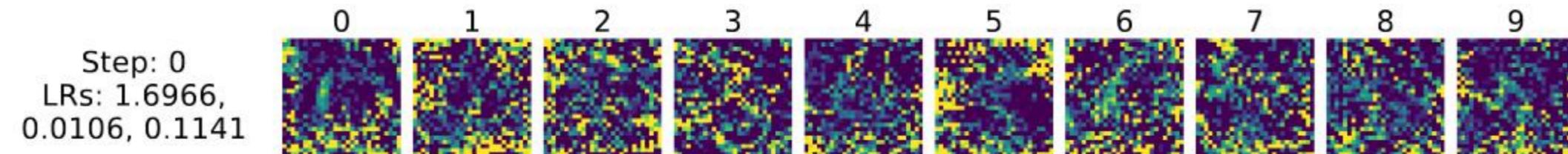
СКОЛЬКО ШАГОВ

Given an initial θ_0 , we obtain these synthetic data $\tilde{\mathbf{x}}$ and learning rate $\tilde{\eta}$ by minimizing the objective below \mathcal{L} :

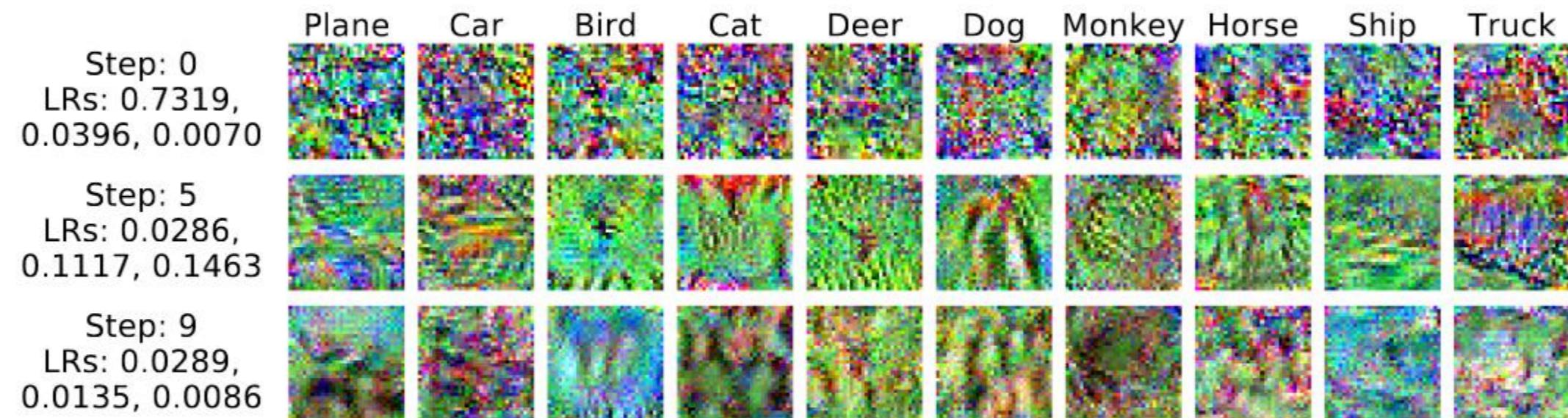
$$\tilde{\mathbf{x}}^*, \tilde{\eta}^* = \arg \min_{\tilde{\mathbf{x}}, \tilde{\eta}} \mathcal{L}(\tilde{\mathbf{x}}, \tilde{\eta}; \theta_0) = \arg \min_{\tilde{\mathbf{x}}, \tilde{\eta}} \ell(\mathbf{x}, \theta_1) = \arg \min_{\tilde{\mathbf{x}}, \tilde{\eta}} \ell(\mathbf{x}, \theta_0 - \tilde{\eta} \nabla_{\theta_0} \ell(\tilde{\mathbf{x}}, \theta_0)), \quad (3)$$

where we derive the new weights θ_1 as a function of distilled data $\tilde{\mathbf{x}}$ and learning rate $\tilde{\eta}$ using Equation 2 and then evaluate the new weights over all the training data \mathbf{x} .

Дистилляция: результаты для фиксированной инициализации



(a) MNIST. These distilled images train a fixed initialization from 12.90% test accuracy to 93.76%.



(b) CIFAR10. These distilled images train a fixed initialization from 8.82% test accuracy to 54.03%.

Figure 2: Dataset distillation for fixed initializations: MNIST distilled images use one GD step and three epochs (10 images in total). CIFAR10 distilled images use ten GD steps and three epochs (100 images in total). For CIFAR10, only selected steps are shown. At left, we report the corresponding learning rates for all three epochs.

Дистилляция данных: подход

Недостаток: нереалистичные картинки

А что если сделать для любых инициализаций?

Зададимся распределением на весах сети и будем брать веса из этого распределения

$$\tilde{\mathbf{x}}^*, \tilde{\eta}^* = \arg \min_{\tilde{\mathbf{x}}, \tilde{\eta}} \mathbb{E}_{\theta_0 \sim p(\theta_0)} \mathcal{L}(\tilde{\mathbf{x}}, \tilde{\eta}; \theta_0),$$

Дистилляция данных: подход

Algorithm 1 Dataset Distillation

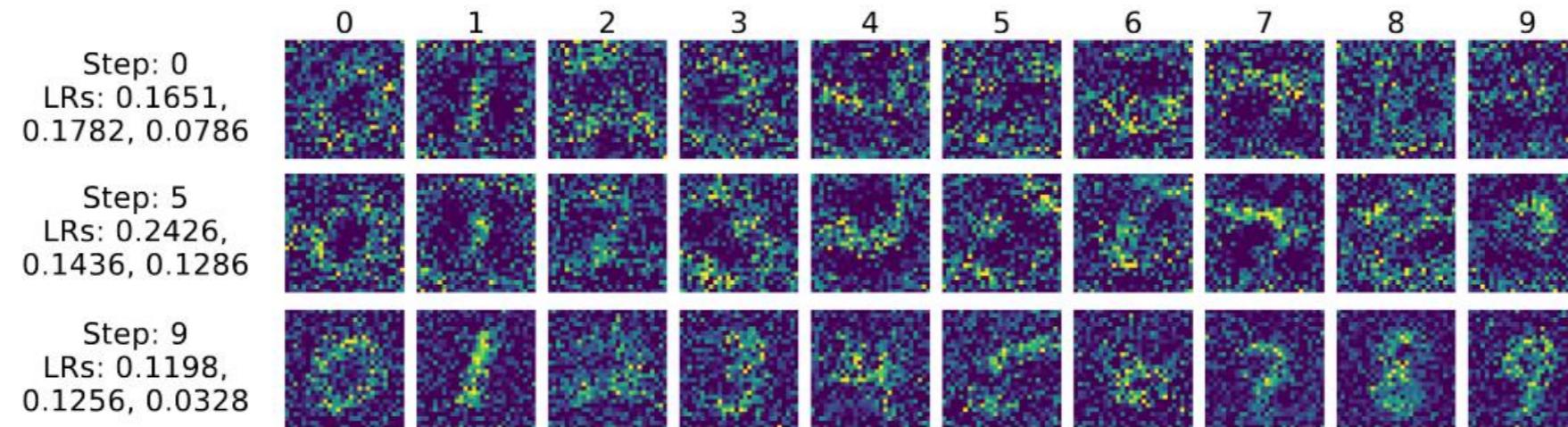
Input: $p(\theta_0)$: distribution of initial weights; M : the number of distilled data

Input: α : step size; n : batch size; T : the number of optimization iterations; $\tilde{\eta}_0$: initial value for $\tilde{\eta}$

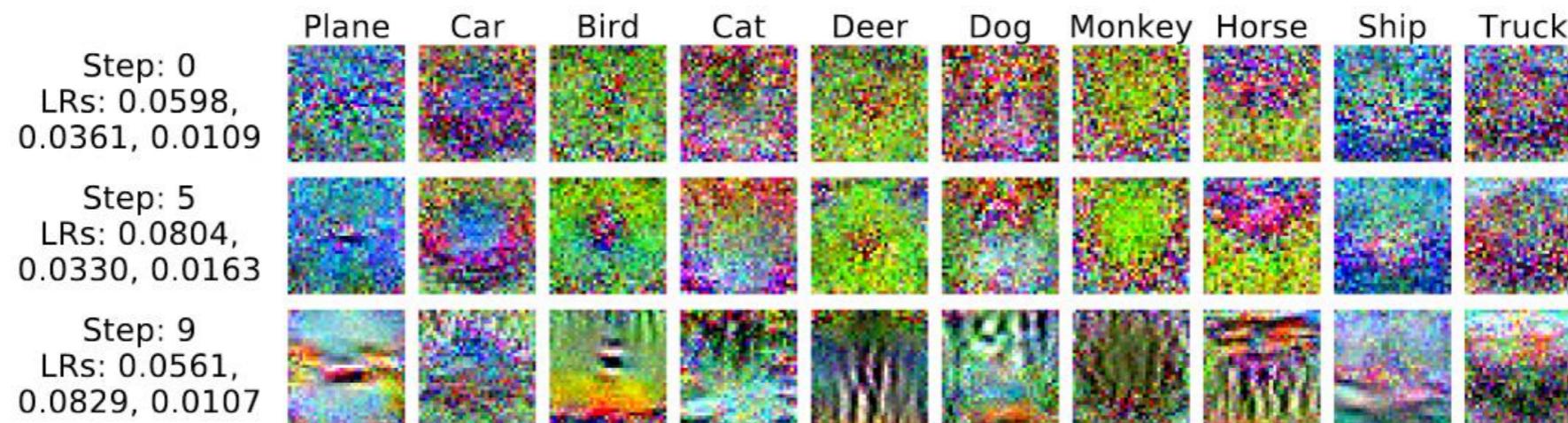
- 1: Initialize $\tilde{\mathbf{x}} = \{\tilde{x}_i\}_{i=1}^M$ randomly, $\tilde{\eta} \leftarrow \tilde{\eta}_0$
- 2: **for each** training step $t = 1$ to T **do**
- 3: Get a minibatch of real training data $\mathbf{x}_t = \{x_{t,j}\}_{j=1}^n$
- 4: Sample a batch of initial weights $\theta_0^{(j)} \sim p(\theta_0)$
- 5: **for each** sampled $\theta_0^{(j)}$ **do**
- 6: Compute updated parameter with GD: $\theta_1^{(j)} = \theta_0^{(j)} - \tilde{\eta} \nabla_{\theta_0^{(j)}} \ell(\tilde{\mathbf{x}}, \theta_0^{(j)})$
- 7: Evaluate the objective function on real training data: $\mathcal{L}^{(j)} = \ell(\mathbf{x}_t, \theta_1^{(j)})$
- 8: **end for**
- 9: Update $\tilde{\mathbf{x}} \leftarrow \tilde{\mathbf{x}} - \alpha \nabla_{\tilde{\mathbf{x}}} \sum_j \mathcal{L}^{(j)}$, and $\tilde{\eta} \leftarrow \tilde{\eta} - \alpha \nabla_{\tilde{\eta}} \sum_j \mathcal{L}^{(j)}$
- 10: **end for**

Output: distilled data $\tilde{\mathbf{x}}$ and optimized learning rate $\tilde{\eta}$

Дистилляция данных: результат для произвольных инициализаций



(a) MNIST. These distilled images unknown random initializations to $79.50\% \pm 8.08\%$ test accuracy.



(b) CIFAR10. These distilled images unknown random initializations to $36.79\% \pm 1.18\%$ test accuracy.

Figure 3: Distilled images trained for *random initialization* with ten GD steps and three epochs (100 images in total). We show images from selected GD steps and the corresponding learning rates for all three epochs.

Результаты

	Ours		Baselines					
	Fixed init.	Random init.	Used as training data in same number of GD steps				Used in K-NN classification	
			Random real	Optimized real	<i>k</i> -means	Average real	Random real	<i>k</i> -means
MNIST	96.6	79.5 ± 8.1	68.6 ± 9.8	73.0 ± 7.6	76.4 ± 9.5	77.1 ± 2.7	71.5 ± 2.1	92.2 ± 0.1
CIFAR10	54.0	36.8 ± 1.2	21.3 ± 1.5	23.4 ± 1.3	22.5 ± 3.1	22.3 ± 0.7	18.8 ± 1.3	29.4 ± 0.3

Table 1: Comparison between our method trained for ten GD steps and three epochs and various baselines. For baselines using K-Nearest Neighbor (K-NN), best result among all combinations of distance metric $\in \{l_1, l_2\}$ and $K \in \{1, 3\}$ is reported. In K-NN and *k*-means, K and k can have different values. All methods use ten images per category (100 in total), except for the average real images baseline, which reuses the same images in different GD steps.

Random real images – randomly sample the same number of real images per category

Optimized real images – sample different sets of random real images as above, and choose the top 20% best performing sets

***k*-means** – apply *k*-means clustering to each category, and use the cluster centroids as training images

Average real images – compute the average image for each category, which is reused indifferent GD steps

BTW

Не было dropout-а
(чтобы избежать шума)

distilled learning rates = 0.02
Adam solver + learning rate = 0.001

NVIDIA Titan Xp and V100 GPUs
one GPU for fixed initial weights
four GPUs for random initial weights
1-4 часа обучения

«Мягкая дистилляция»

**Если использовать не чёткие метки, а распределения,
датасет можно ещё больше сжать / получить лучше качество!**

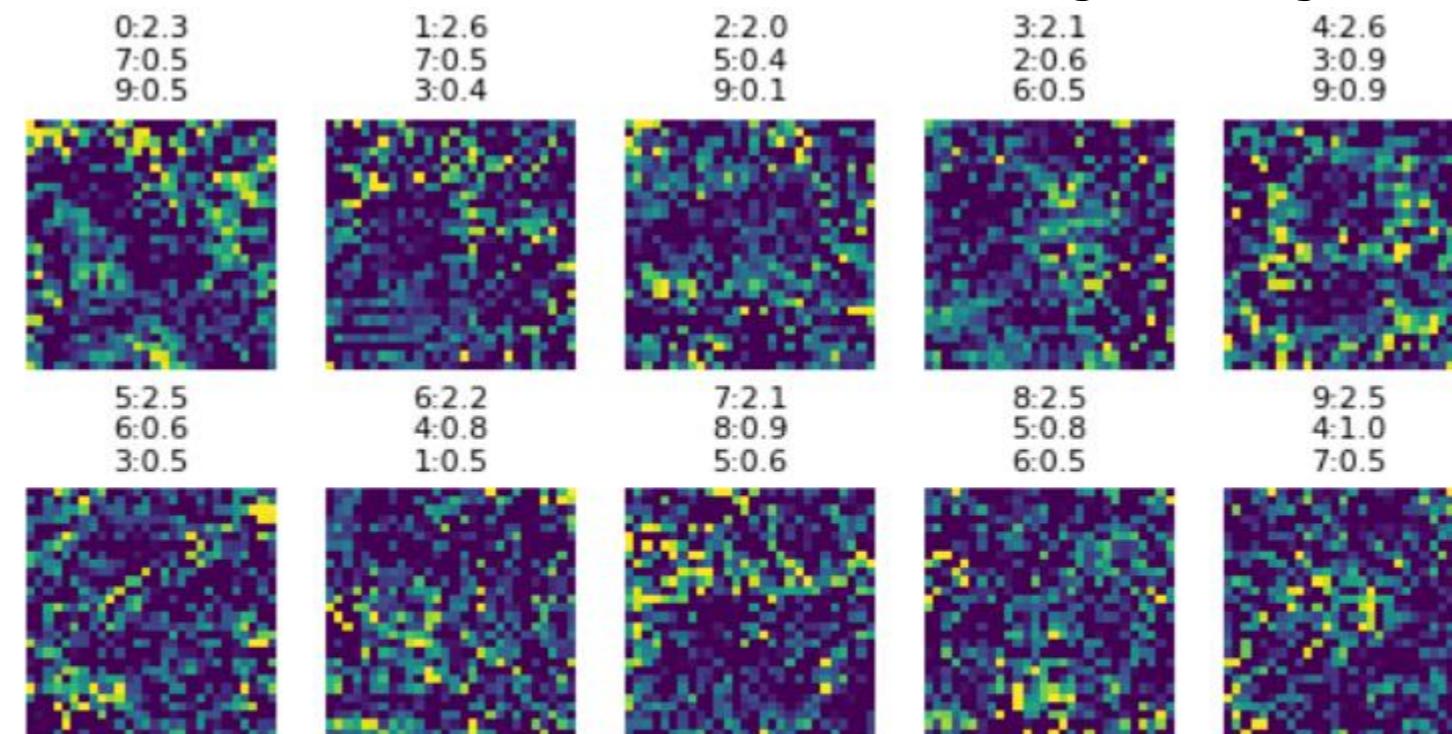


Figure 1: 10 MNIST images learned by SLDD can train networks with fixed initializations from 11.13% distillation accuracy to 96.13%. Each image is labelled with its top 3 classes and their associated logits. The full labels for these 10 images can be found in Table 1.

Ilya Sucholutsky, Matthias Schonlau «Soft-Label Dataset Distillation and Text Dataset Distillation» // <https://paperswithcode.com/paper/improving-dataset-distillation>

«Мягкая дистилляция»

96% MNIST – 10 дистилированных изображений (лучше, чем в чётком случае)

точность ~ 92% только 5 дистилированных изображений! (Менее 1 объекта на класс!)

Идея – минимизировать и по у

With our modified notation, we simply need to change equation (4) to also minimize over \tilde{y} .

$$\tilde{x}^*, \tilde{y}^*, \tilde{\eta}^* = \arg \min_{\tilde{x}, \tilde{y}, \tilde{\eta}} \mathcal{L}(\tilde{x}, \tilde{y}, \tilde{\eta}; \theta_0) = \arg \min_{\tilde{x}, \tilde{\eta}} \ell(x, y, \theta_0 - \tilde{\eta} \nabla_{\theta_0} \ell(\tilde{x}, \tilde{y}, \theta_0)) \quad (5)$$

Algorithm 1 Soft-Label Dataset Distillation (SLDD)

Input: $p(\theta_0)$: distribution of initial weights; M : the number of distilled data; α : step size; n : batch size; T : number of optimization iterations; \tilde{y}_0 : initial value for \tilde{y} ; $\tilde{\eta}_0$: initial value for $\tilde{\eta}$

- 1: Initialize distilled data
 $\tilde{\mathbf{x}} = \{\tilde{x}_i\}_{i=1}^M$ randomly,
 $\tilde{\mathbf{y}} = \{\tilde{y}_i\}_{i=1}^M \leftarrow \tilde{y}_0$,
 $\tilde{\eta} \leftarrow \tilde{\eta}_0$
- 2: **for** each training step $t = 1$ to T **do**
- 3: Get a minibatch of real training data
 $(\mathbf{x}_t, \mathbf{y}_t) = \{x_{t,j}, y_{t,j}\}_{j=1}^n$
- 4: One-hot encode the labels
 $(\mathbf{x}_t, \mathbf{y}^*_t) = \{x_{t,j}, \text{Encode}(y_{t,j})\}_{j=1}^n$
- 5: Sample a batch of initial weights
 $\theta_0^{(j)} \sim p(\theta_0)$
- 6: **for** each sampled $\theta_0^{(j)}$ **do**
- 7: Compute updated model parameter with GD
 $\theta_1^{(j)} = \theta_0^{(j)} - \tilde{\eta} \nabla_{\theta_0^{(j)}} \ell(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta_0^{(j)})$
- 8: Evaluate the objective function on real training data: $\mathcal{L}^{(j)} = \ell(\mathbf{x}_t, \mathbf{y}^*_t, \theta_1^{(j)})$
- 9: **end for**
- 10: Update distilled data
 $\tilde{\mathbf{x}} \leftarrow \tilde{\mathbf{x}} - \alpha \nabla_{\tilde{\mathbf{x}}} \sum_j \mathcal{L}^{(j)}$,
 $\tilde{\mathbf{y}} \leftarrow \tilde{\mathbf{y}} - \alpha \nabla_{\tilde{\mathbf{y}}} \sum_j \mathcal{L}^{(j)}$, and
 $\tilde{\eta} \leftarrow \tilde{\eta} - \alpha \nabla_{\tilde{\eta}} \sum_j \mathcal{L}^{(j)}$
- 11: **end for**

Output: distilled data $\tilde{\mathbf{x}}$; distilled labels $\tilde{\mathbf{y}}$; optimized learning rate $\tilde{\eta}$

Дистилляция для текстов: Text Dataset Distillation (TDD)

Немного скажем, т.к. текст дискретный

GloVe-кодировки слов

Пусть каждый текст длины 400 (обрежем или дополним)

Тогда текст = изображение-матрица [размерность представления] × 400

**CNN точность 91% в IMDB sentiment classification task (Maas et al., 2011),
используя 20 синтетических предложений**

«Мягкая дистилляция»

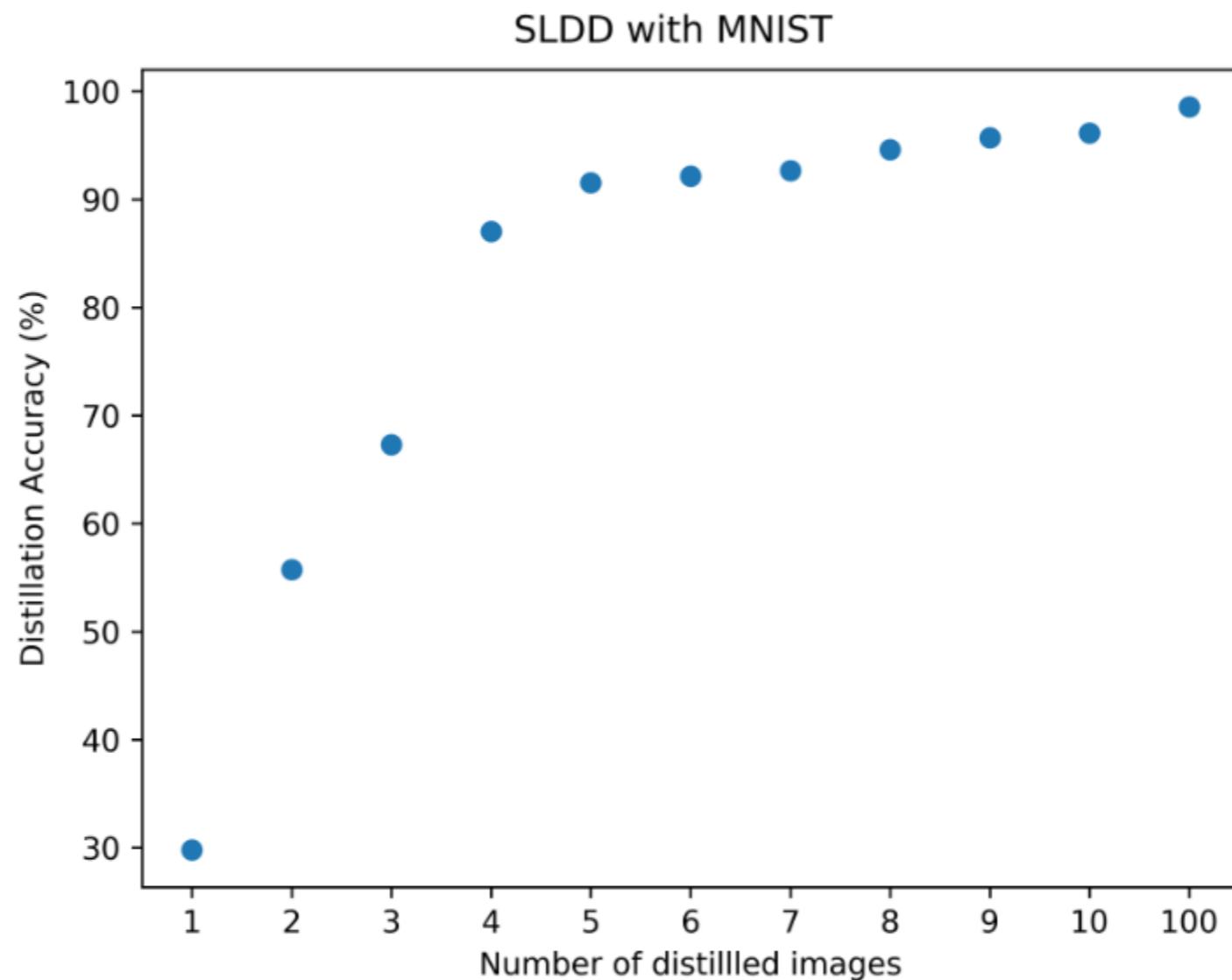


Figure 3: Distillation accuracy on MNIST with LeNet for different distilled dataset sizes.

Ссылки

Интерфейс для визуализации и интерпретации

<https://distill.pub/2018/building-blocks/>

Итог

**С помощью визуализации можно понять,
что делают НС**

Стиль ~ матрица Грама

Интересное направление – дистилляция данных