



ТЕХНОСФЕРА

Лекция 12 Variational Autoencoders Нейронные сети для искусства

Храбров Кузьма

18 ноября 2019 г.

План лекции

VAE

Трепанация нейронных сетей

Deep Dream

Artistic Style

Generative adversarial networks

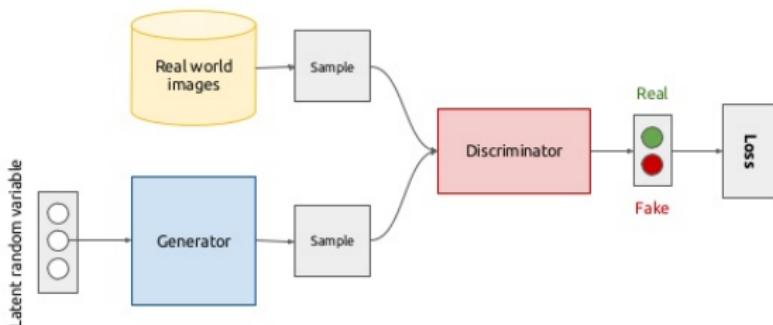
$z \sim p_z(z)$ - noise vector

$p_g(z)$ - распределение сгенерированных картинок из noise

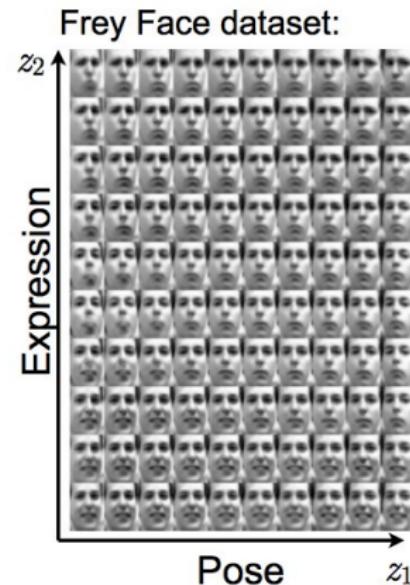
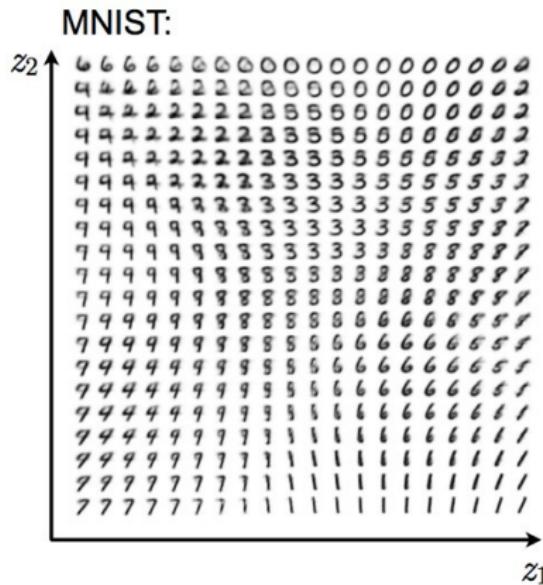
$p_{data}(x)$ - распределение настоящих картинок

$G(z)$ - генератор (генерирует картинку из z)

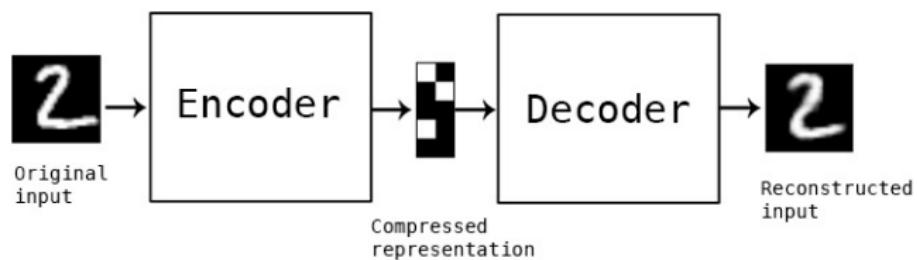
$D(x)$ - дискриминатор (отличает реальные от сгенерированных)



Autoencoder



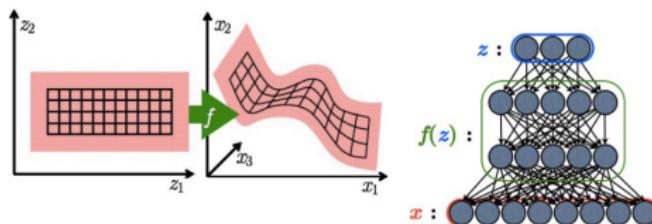
Autoencoder



Latent variable model

Найдем отображение из пространства латентных переменных (z) в распределение исходных данных (x).

$$p(x) = \int p(x, z) dz$$
$$p(x, z) = p(x|z)p(z)$$



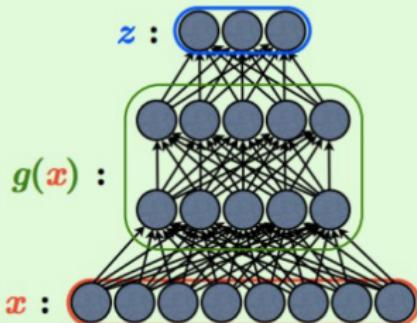
VAE

- ▶ Откуда взять z и $p(z|x)$?
- ▶ Подход VAE: будем аппроксимировать $p_\theta(z|x)$ с помощью $q_\phi(z|x)$, оптимизируя вариационную нижнюю границу $\mathcal{L}(\theta, \phi, x)$

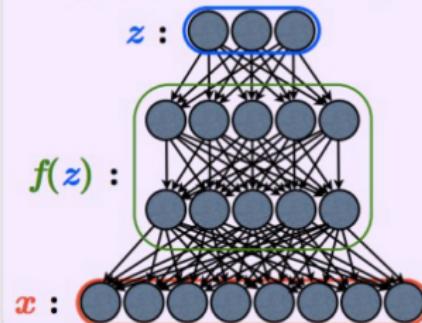
$$\begin{aligned}\mathcal{L}(\theta, \phi, x) &= \mathbb{E}_{q_\phi(z|x)} [\log p_\theta(x, z) - \log q_\phi(z | x)] \\ &= \mathbb{E}_{q_\phi(z|x)} [\log p_\theta(x | z) + \log p_\theta(z) - \log q_\phi(z | x)] \\ &= -D_{\text{KL}}(q_\phi(z | x) \| p_\theta(z)) + \mathbb{E}_{q_\phi(z|x)} [\log p_\theta(x | z)]\end{aligned}$$

VAE

$$q_{\phi}(z \mid x) = q(z; g(x, \phi))$$

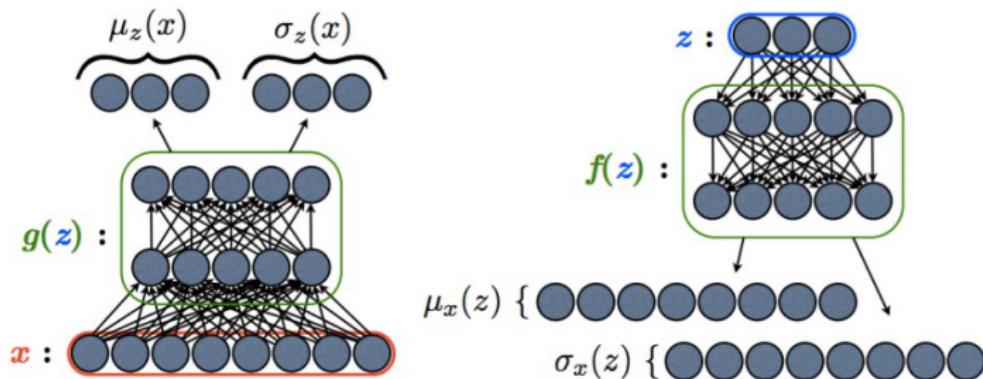


$$p_{\theta}(x \mid z) = p(x; f(z, \theta))$$

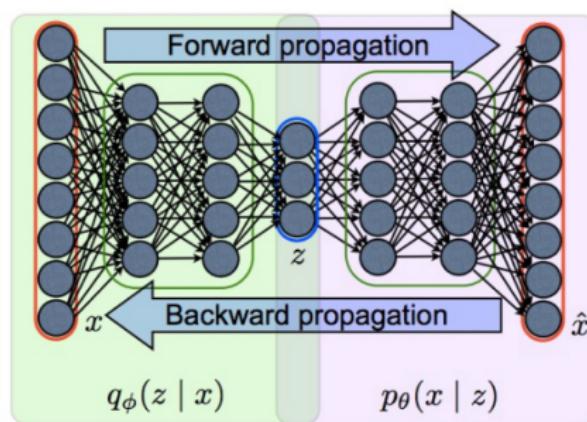


VAE

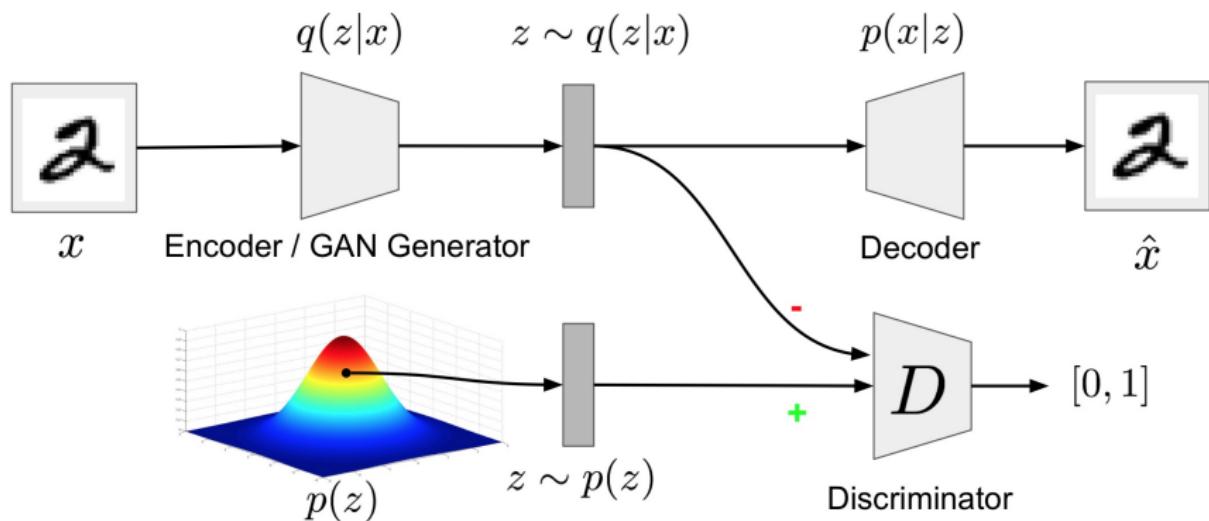
- ▶ Возьмем в качестве $q_\phi(z|x)$ нормальное распределение $\mathcal{N}(z; \mu_z(x), \sigma_z(x))$
- ▶ Параметризуем $z = \mu_z(x)\varepsilon_z$, где теперь $\varepsilon_z \sim \mathcal{N}(0, 1)$



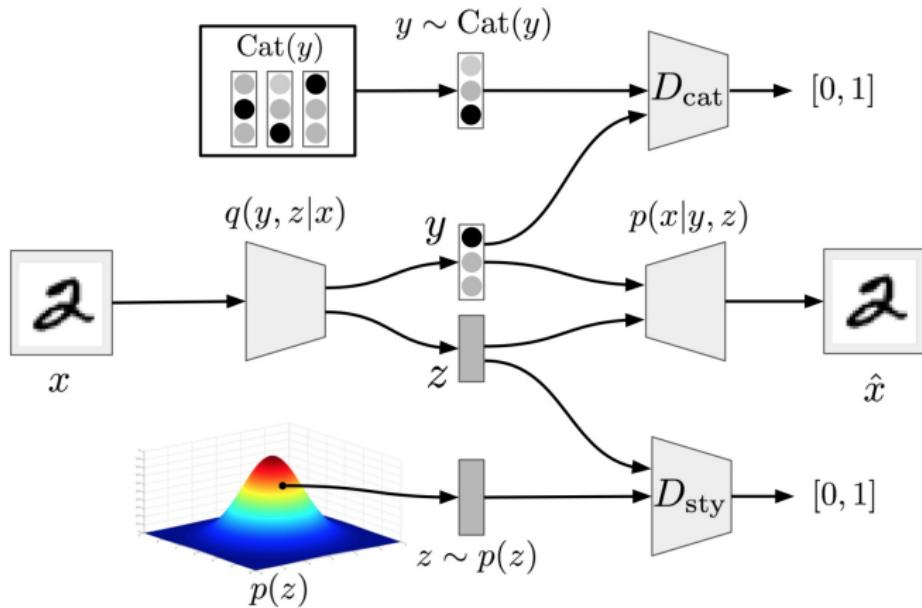
VAE



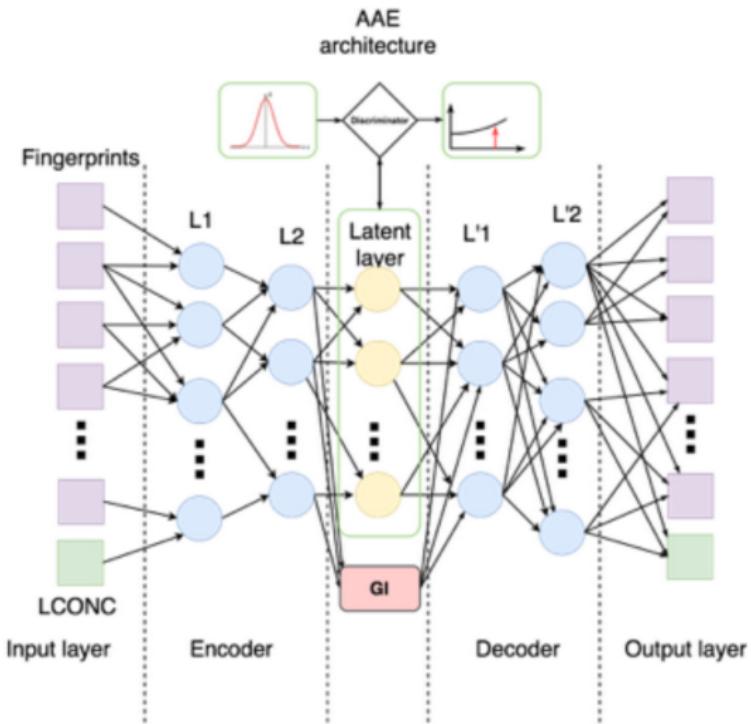
Adversarial autoencoder



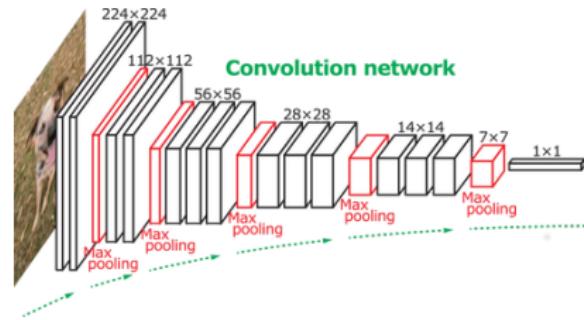
AAE



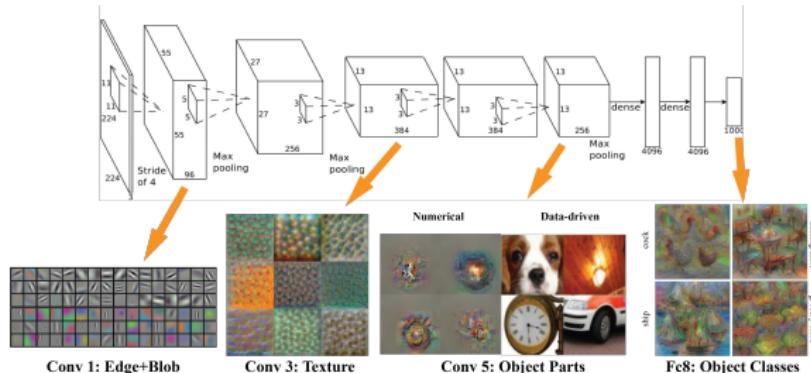
AAE



Сверточные сети - Сложный черный ящик ?



Визуализация фильтров нейронной сети



На самом деле мы можем каждому фильтру в данном слое сопоставить фрагмент картинки и более того выделить конкретные признаки.

Вопрос: как?

Модель Hubel & Weisel

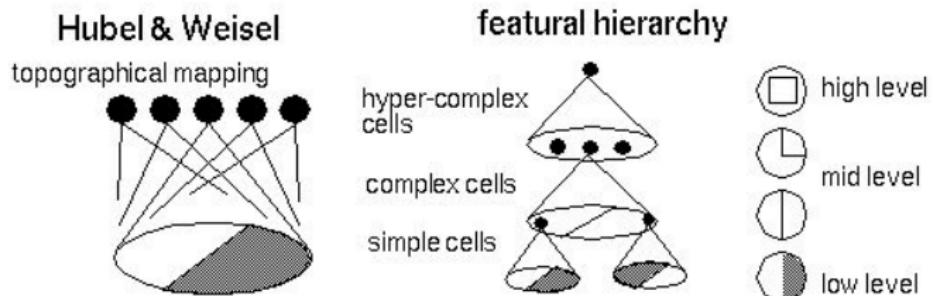


Рис.: Модель Hubel & Weisel¹

- ▶ предложена нейрофизиологами Дэвидом Хьюбелом и Торстеном Визельем в 1959 году
- ▶ в 1981 году награждены Нобелевской премией «за открытия, касающиеся принципов переработки информации в нейронных структурах»
- ▶ фундаментальные работы Визеля и Хьюбела по нейрофизиологии зрения открыли основы организации и развития нейронных цепей, ответственных за зрительное распознавание объектов

¹<http://cns-alumni.bu.edu/~slehar/webstuff/pcave/hubel.html>

Deconvolutional networks #1

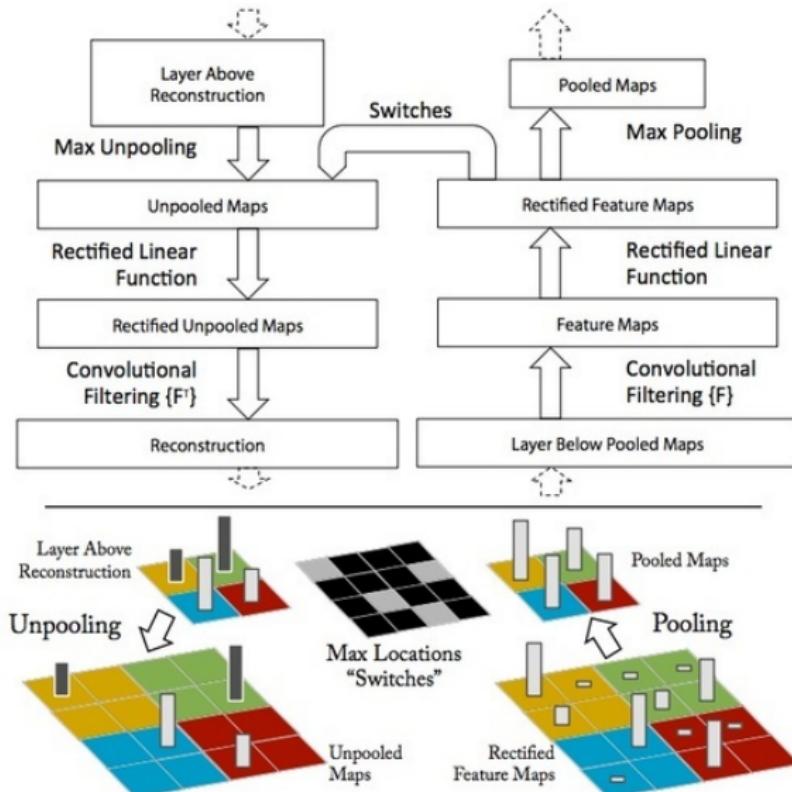


Рис.: Схема deconvolution сети

Deconvolutional networks #2 Inverse nonlinearity

Типичная нелинейность в сверточных сетях - *relu*. Как обратную операцию используют тот же *relu*.

Deconvolutional networks #3 Deconvolution

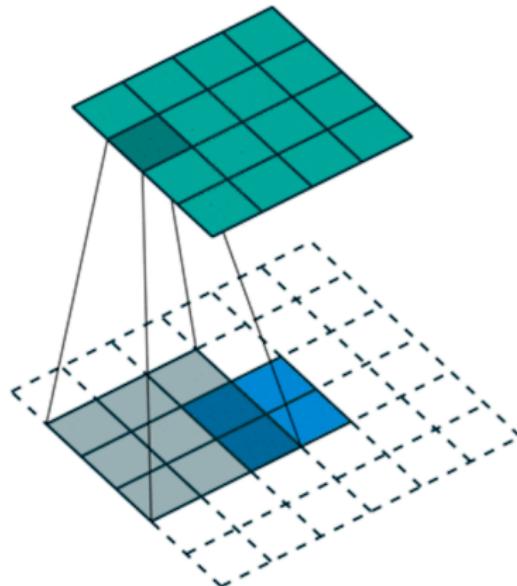
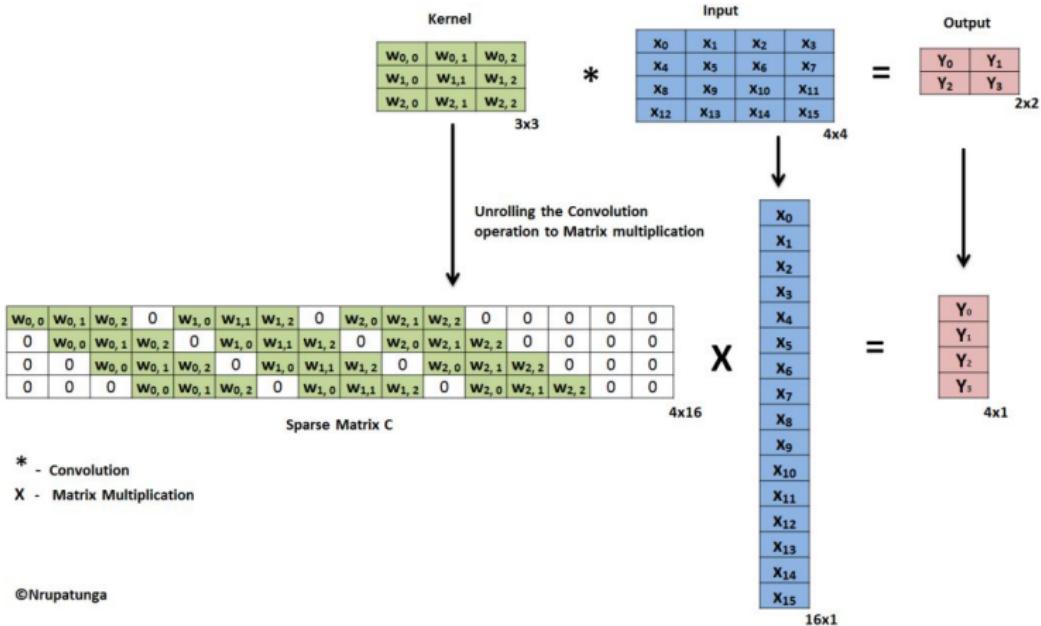


Рис.: Convolution transposed²

Формула:

$$\hat{y}_i^c = \sum_{k=1}^{K_i} z_{k,i} * f_{k,1}^c \quad (1)$$



$w_{0,0}$	0	0	0
$w_{0,1}$	$w_{0,0}$	0	0
$w_{0,2}$	$w_{0,1}$	$w_{0,0}$	0
0	$w_{0,2}$	$w_{0,1}$	$w_{0,0}$
$w_{1,0}$	0	$w_{0,2}$	$w_{0,1}$
$w_{1,1}$	$w_{1,0}$	0	$w_{0,2}$
$w_{1,2}$	$w_{1,1}$	$w_{1,0}$	0
0	$w_{1,2}$	$w_{1,1}$	$w_{1,0}$
$w_{2,0}$	0	$w_{1,2}$	$w_{1,1}$
$w_{2,1}$	$w_{2,0}$	0	$w_{1,2}$
$w_{2,2}$	$w_{2,1}$	$w_{2,0}$	0
0	$w_{2,2}$	$w_{2,1}$	$w_{2,0}$
0	0	$w_{2,2}$	$w_{2,1}$
0	0	0	$w_{2,2}$
0	0	0	0
0	0	0	0

16x4

Sparse Matrix C^T \times

Y_0
Y_1
Y_2
Y_3

4x1

=

x_0
x_1
x_2
x_3
x_4
x_5
x_6
x_7
x_8
x_9
x_{10}
x_{11}
x_{12}
x_{13}
x_{14}
x_{15}

16x1

Deconvolutional networks #4 Inverse pooling

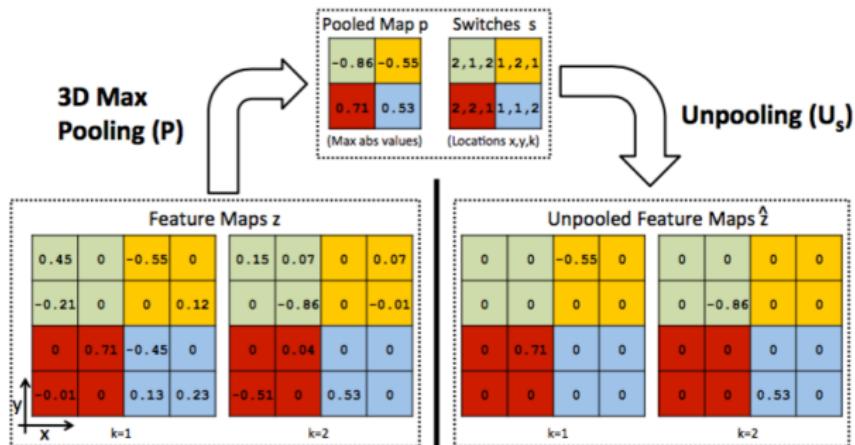


Рис.: Inverse max pooling

Визуализация признаков, #0

Алгоритм:

- ▶ Сделать прямой проход.
- ▶ Выбрать интересующий нас слой.
- ▶ Зафиксировать активации одного или нескольких нейронов и обнулить остальные.
- ▶ Сделать обратный вывод.

Визуализация признаков, #1

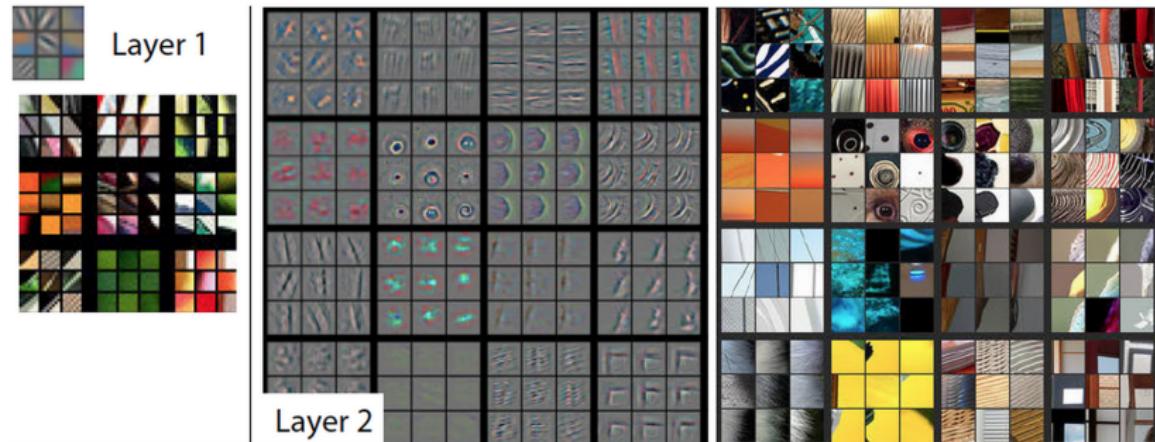


Рис.: Visualizing and Understanding Convolutional Networks³

³Matthew D. Zeiler and Rob Fergus

Визуализация признаков, #2

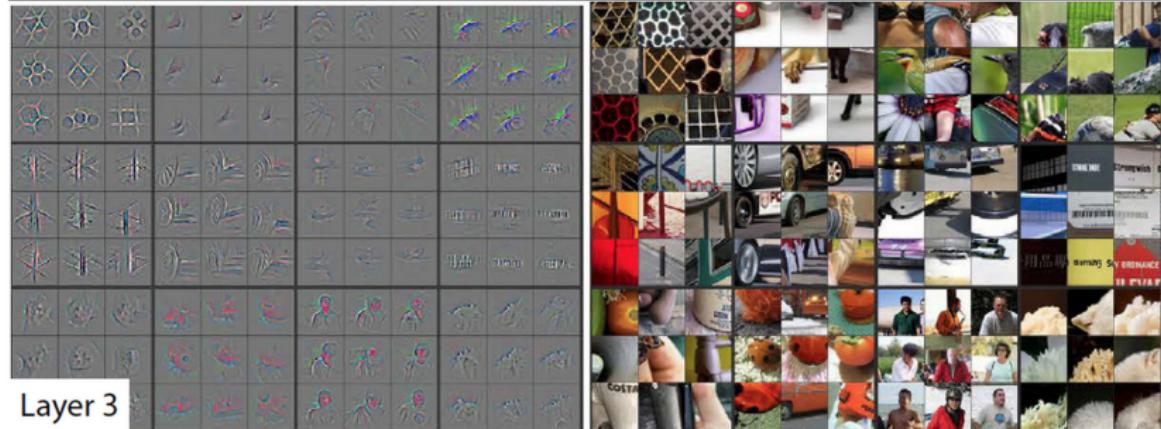


Рис.: Visualizing and Understanding Convolutional Networks⁴

⁴Matthew D. Zeiler and Rob Fergus

Визуализация признаков, #3

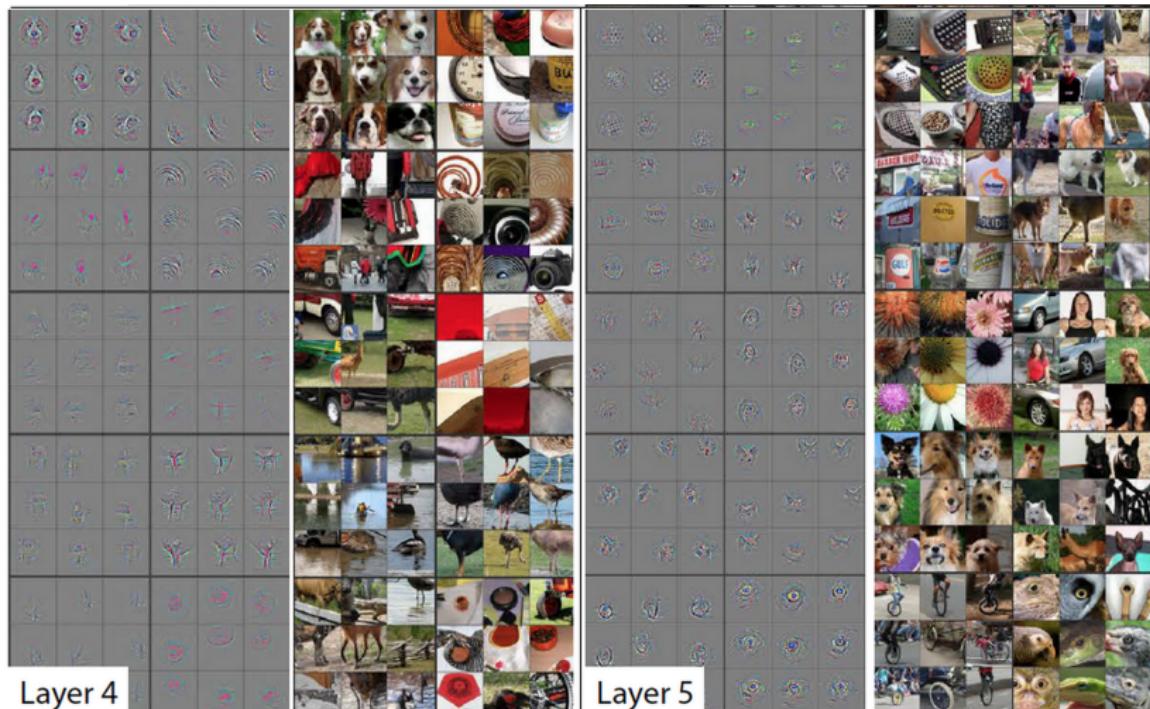


Рис.: Visualizing and Understanding Convolutional Networks⁵

⁵Matthew D. Zeiler and Rob Fergus

Deep Dream, #1

В задаче классификации, на примере ImageNet. Обозначим за $S_c(I)$ - значение активации выходного нейрона, соответствующего классу c . Рассмотрим задачу оптимизации:

$$\arg \max_I S_c(I) - \lambda \|I\|_2^2$$

Ее решение - изображение наиболее "соответствующее" классу.

Deep Dream, #2

Последнюю задачу оптимизации можно решать стандартным способом, то есть используя градиентный спуск.

1. Инициализировать начальное изображение нулями.
2. Вычислить значение производной по этому изображению.
3. Изменить изображение, прибавив к нему полученное изображение от производной.
4. Вернуться к пункту 2 или выйти из цикла.

Deep Dream, #3

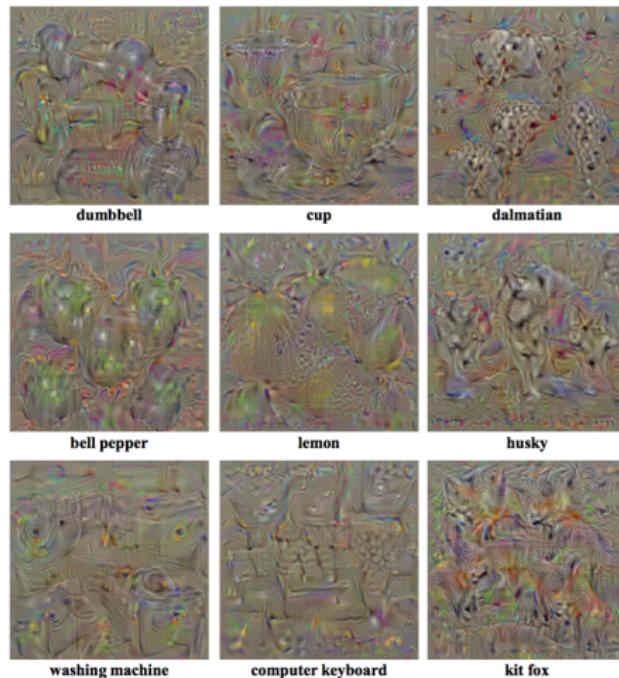


Рис.: Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps⁶

⁶Karen Simonyan, Andrea Vedaldi, Andrew Zisserman

Deep Dream, #4

Теперь инициализируем процесс реальной фотографией и на каждой итерации будем выбирать случайный класс и обнулять остальные.

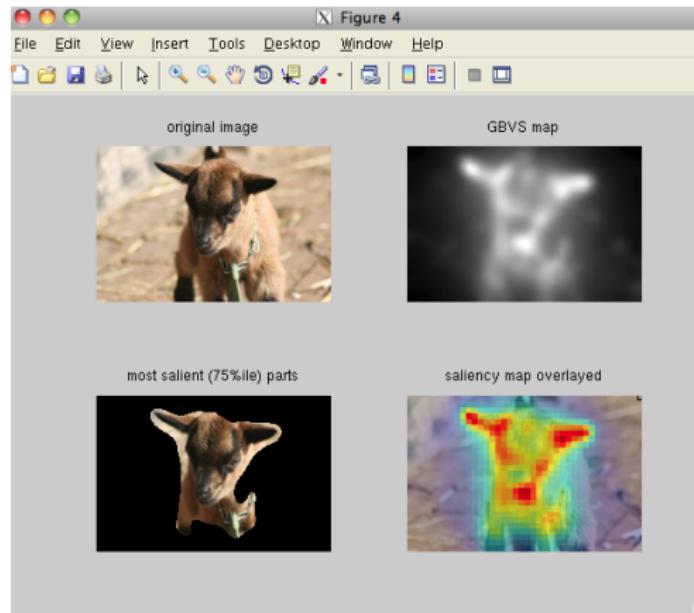
Google deep dream web page

Пример deep dream'a

Вопрос: почему столько морд собак и глаз?

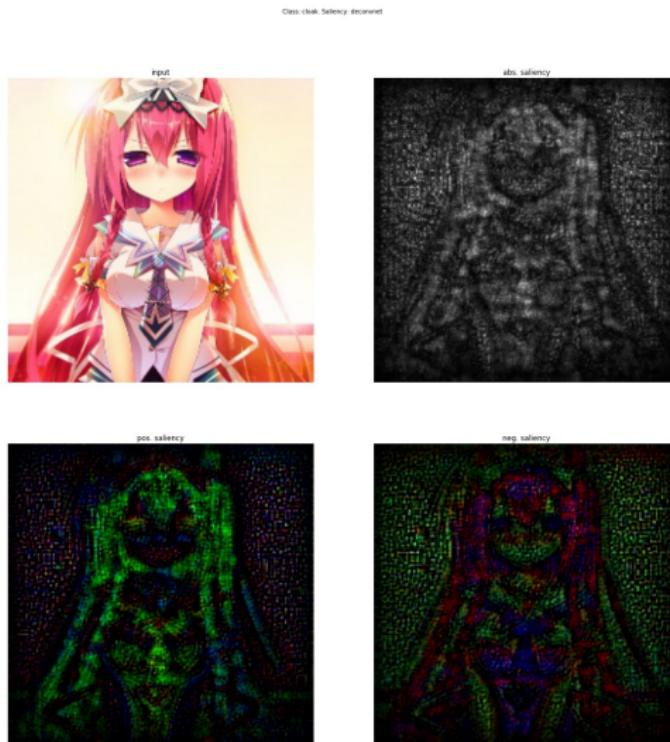
Saliency Maps #1

Saliency map - изображение показывающее выделенность(полезность) каждой точки исходного изображения.



Saliency Maps #2

Вернемся теперь к производной по картинке. Теперь вместо поиска оптимального изображения просто выведем саму производную. Получим вариант saliency map.



Saliency Maps #3

Наконец, можно заметить, что deconvolution тесно связана со взятием производной.

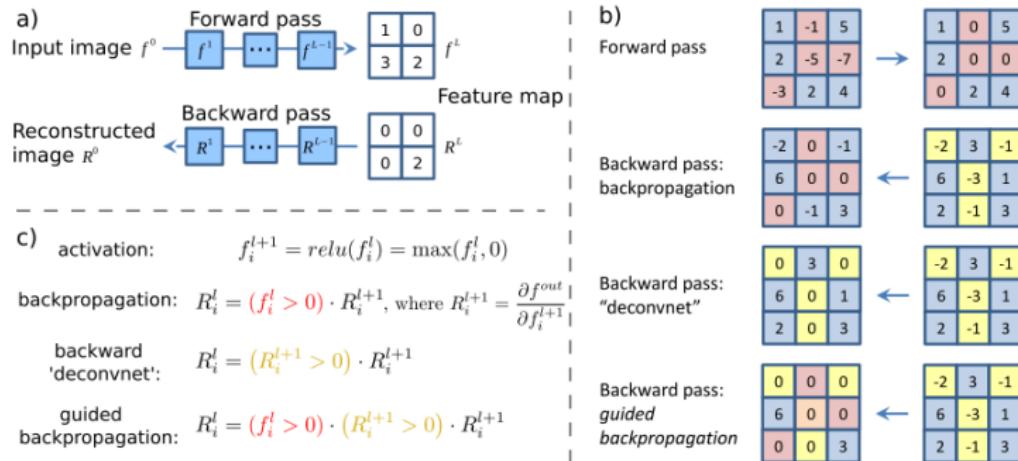


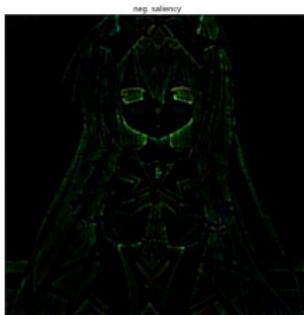
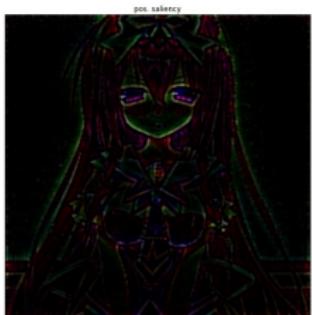
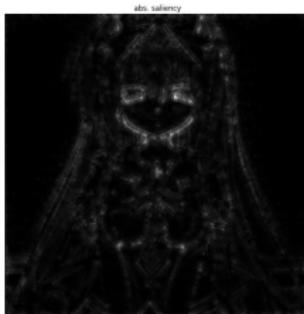
Рис.: Striving for simplicity: the all convolutional net ⁷

⁷ Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, Martin Riedmiller

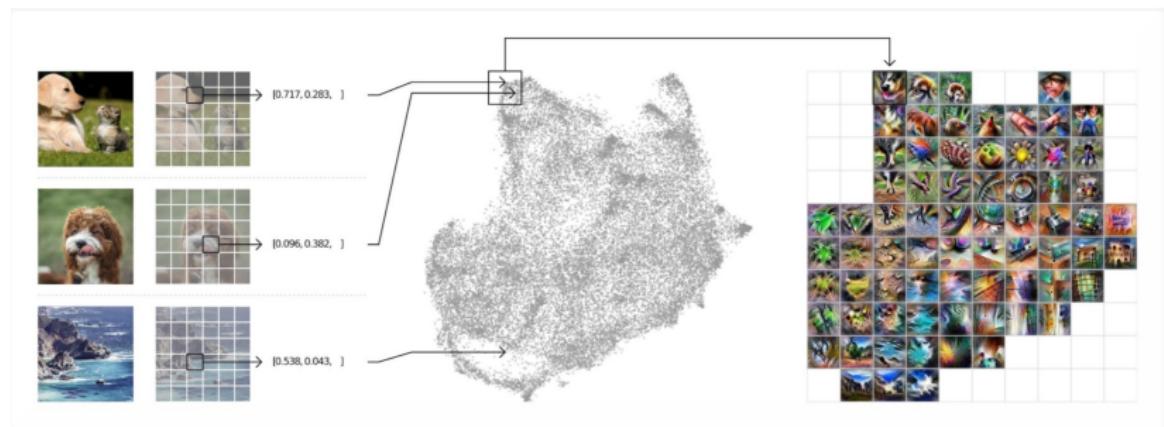
Saliency Maps #4

Используя модифицированный градиентный спуск получаем более четкие saliency maps

Class: chick. Saliency-guided backproj



Activation Atlases #1



Сайт проекта

Artistic Style, #1

Хотим научиться на уровне каждого признака контролировать содержание и стиль картинки

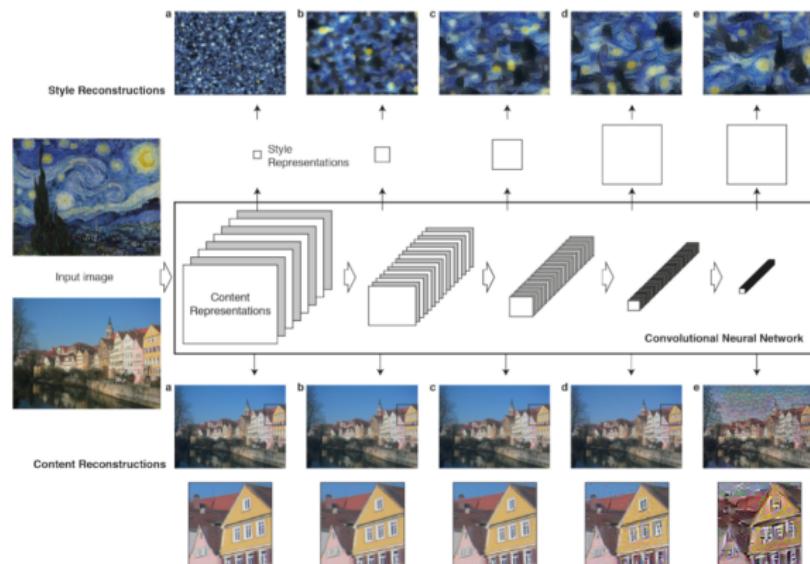


Рис.: A Neural Algorithm of Artistic Style⁸

⁸Leon A. Gatys, Alexander S. Ecker, Matthias Bethge

Artistic Style, #2 Content

Пусть задано изображение c , к которому мы стремимся и изображение x , которое мы генерируем.

Пусть F_{ij}^l - активация i -го фильтра в позиции j на слое l для изображения x . Определим C_{ij}^l для изображения c аналогично. Тогда запишем квадратичную функцию потерь между представлениями для x и c .

$$\mathcal{L}_{content}(x, c) = \sum_l w_l \sum_{i,j} \frac{1}{2} (F_{ij}^l - C_{ij}^l)^2$$

Artistic Style, #3 Content

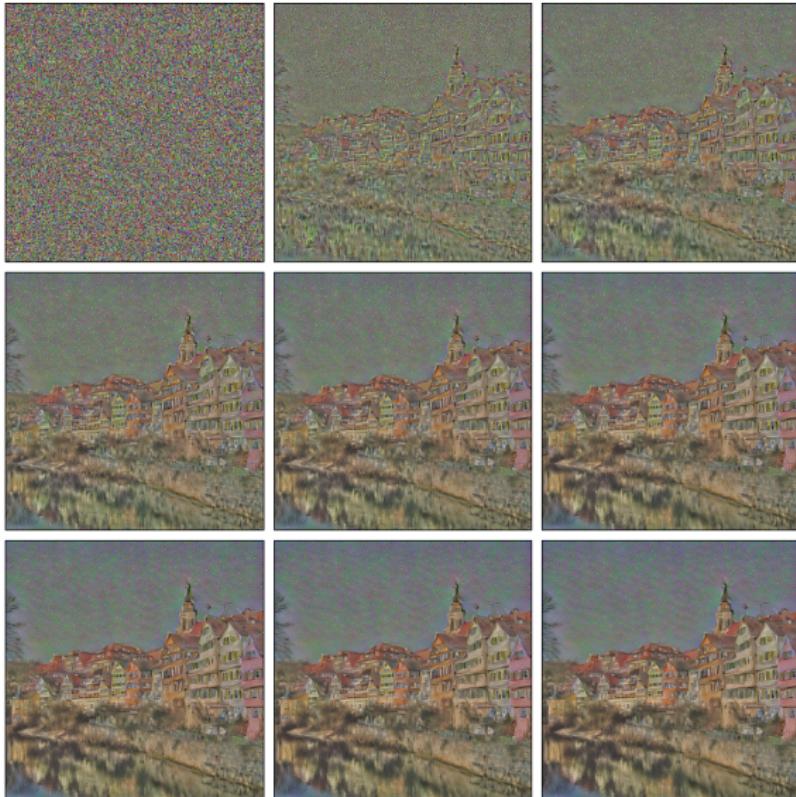


Рис.: artistic style notebook

Artistic Style, #4 Content

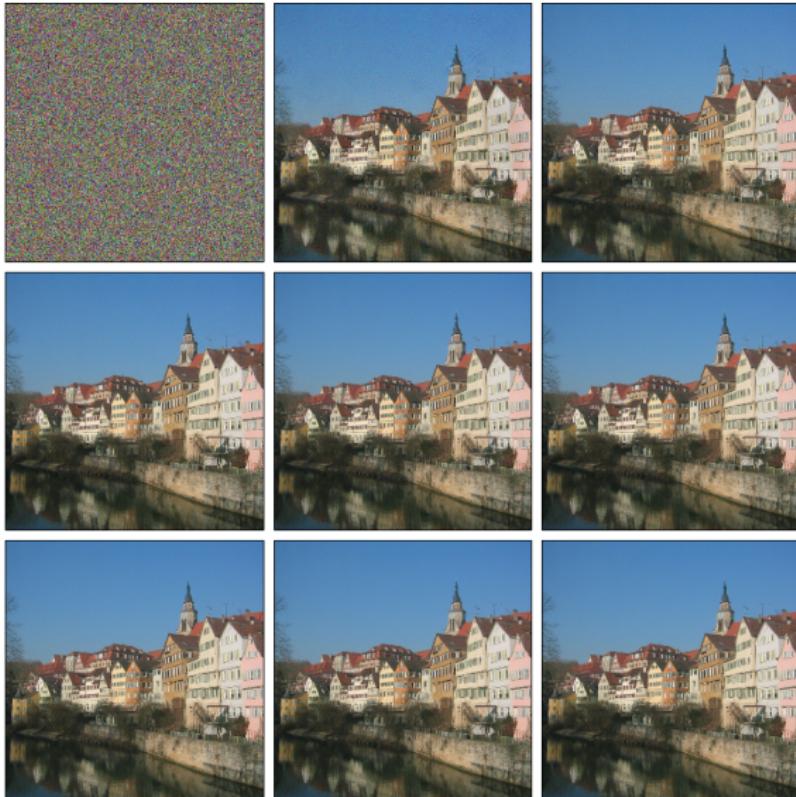


Рис.: artistic style notebook

Artistic Style, #5 Style

Стиль - общая характеристика изображения. Авторы статьи определяют его как корреляцию фильтров нейронной сети.

$$G_{ij}^I = \sum_k F_{ik}^I F_{jk}^I$$

Аналогично определяются A_{ij}^I для исходного изображения.
Тогда функция потерь может быть записана в виде:

$$\sum_{l=0}^L \frac{w_l}{4N_l^2 M_l^2} \sum_{i,j} (G_{ij}^l - A_{ij}^l)^2$$

А ее производная может быть вычислена аналитически:

$$\frac{\partial E_l}{\partial F_{ij}^l} = \begin{cases} \frac{1}{N_l^2 M_l^2} ((F^l)^\top (G^l - A^l))_{ji} & \text{if } F_{ij}^l > 0 \\ 0 & \text{if } F_{ij}^l < 0 . \end{cases} \quad (2)$$

Artistic Style, #6 Style

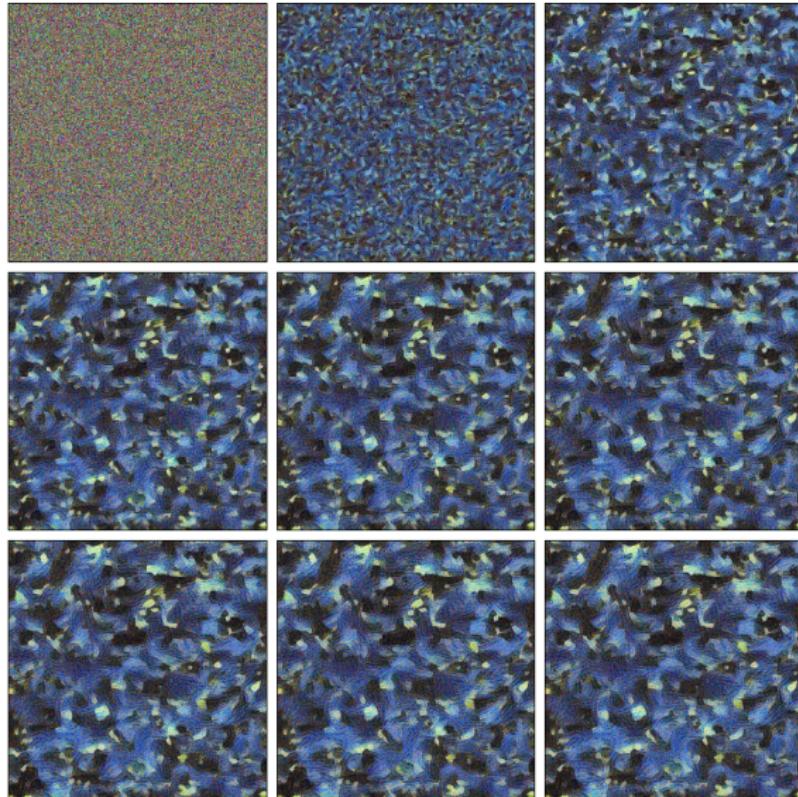
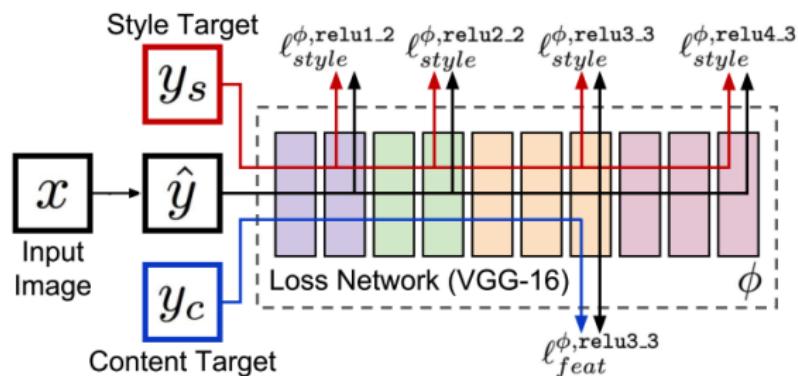


Рис.: artistic style notebook

Artistic Style, #7 Total loss

Наконец, если мы будем оптимизировать комбинированную функцию потерь \mathcal{L}_{total} то на выходе получим изображение соответствующее требуемому стилю и содержанию.

$$\mathcal{L}_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{content}(\vec{p}, \vec{x}) + \beta \mathcal{L}_{style}(\vec{a}, \vec{x}) \quad (3)$$



Artistic Style, #7 Total loss

A



B



C



D



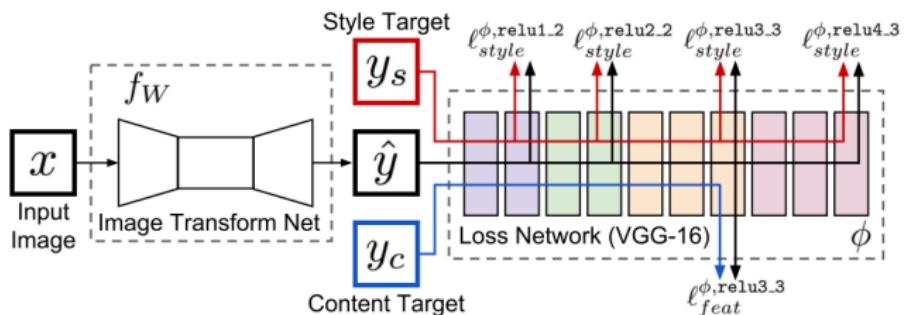
E



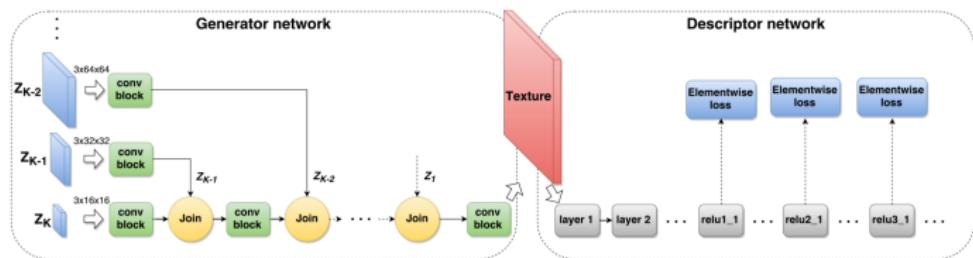
F



Artistic Style, #8 Как ускорить?



Artistic Style, #9 Как ускорить?



Вопросы

