

# Основы глубинного обучения

Лекция 6

Задачи компьютерного зрения

Евгений Соколов

[esokolov@hse.ru](mailto:esokolov@hse.ru)

НИУ ВШЭ, 2020

# Интерпретация моделей

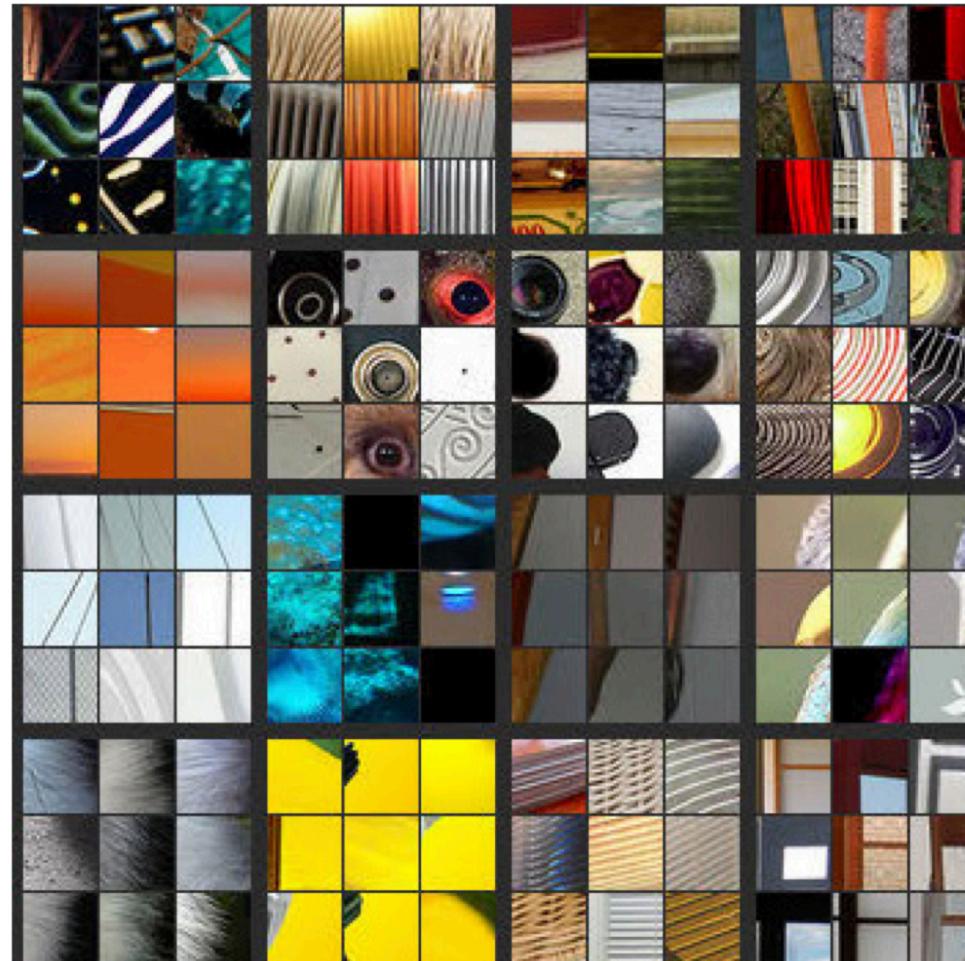
# Что находят свёрточные сети?

- Можно для каждого фильтра найти кусочки картинок, дающие самый сильный отклик
- Сделаем это для AlexNet

# Слой 1



# Слой 2



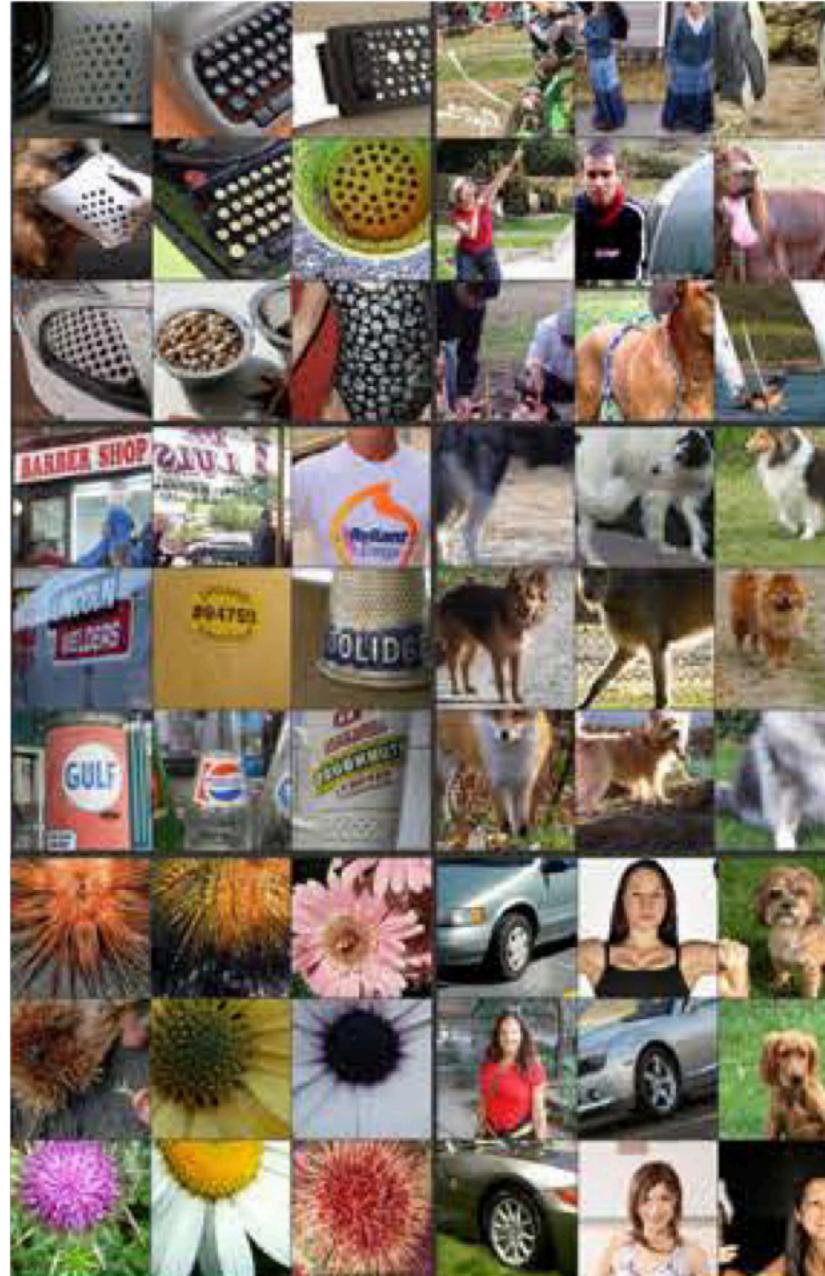
# Слой 3



# Слой 4



# Слой 5

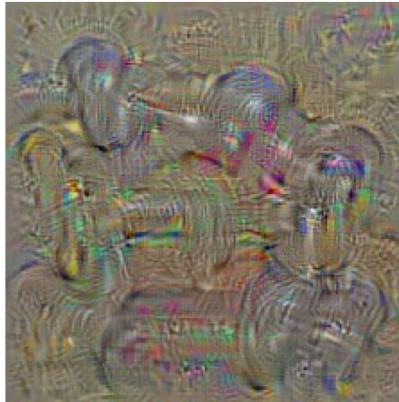


# Максимизация вероятности класса

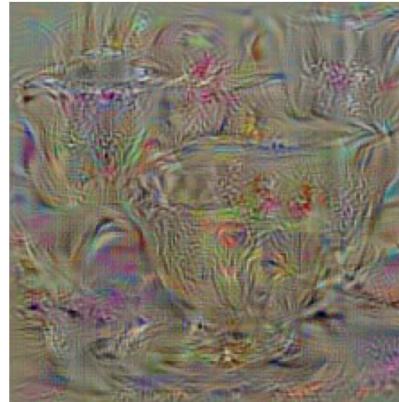
$$a_y(x) - \lambda \|x\|_2^2 \rightarrow \max_x$$

- Инициализируем картинку случайным шумом
- Ищем оптимальную для данного класса картинку градиентным спуском

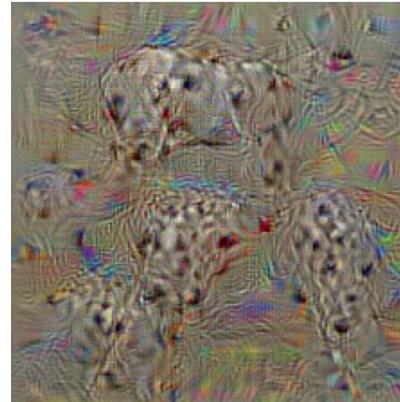
# Максимизация вероятности класса



dumbbell



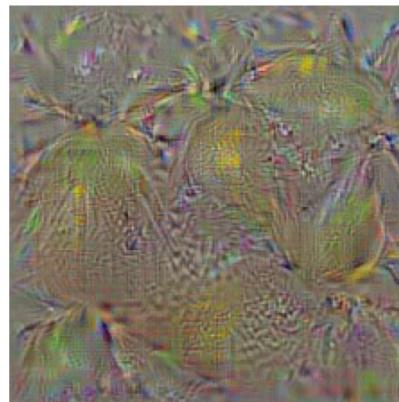
cup



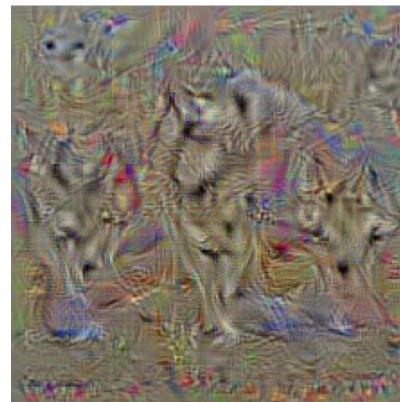
dalmatian



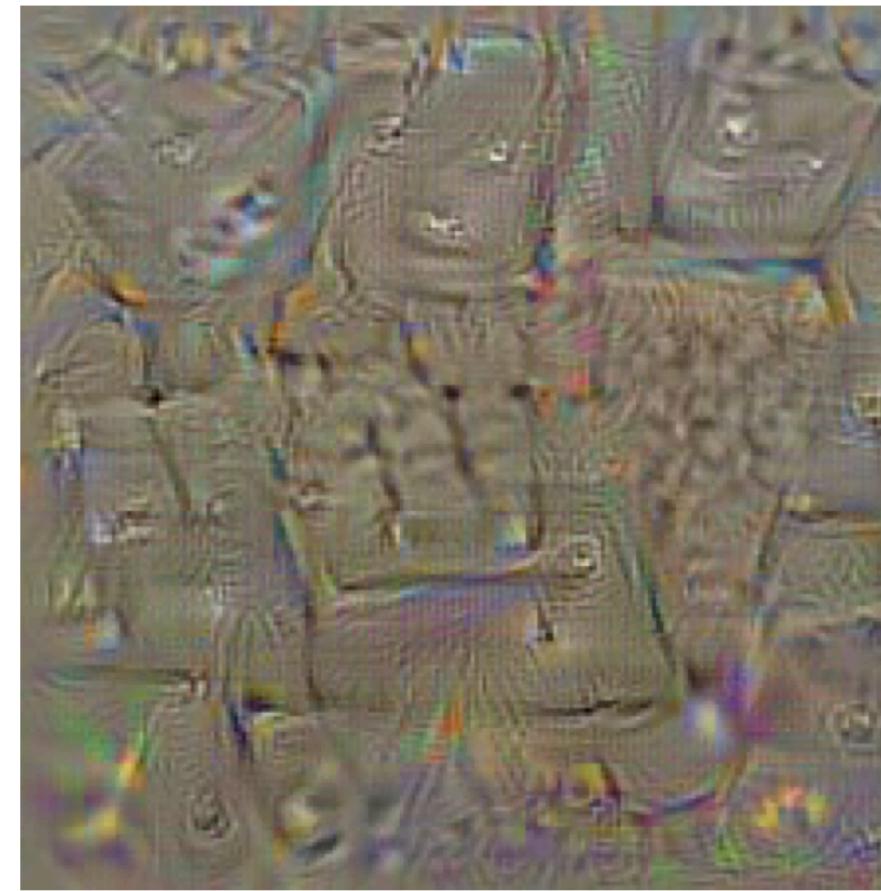
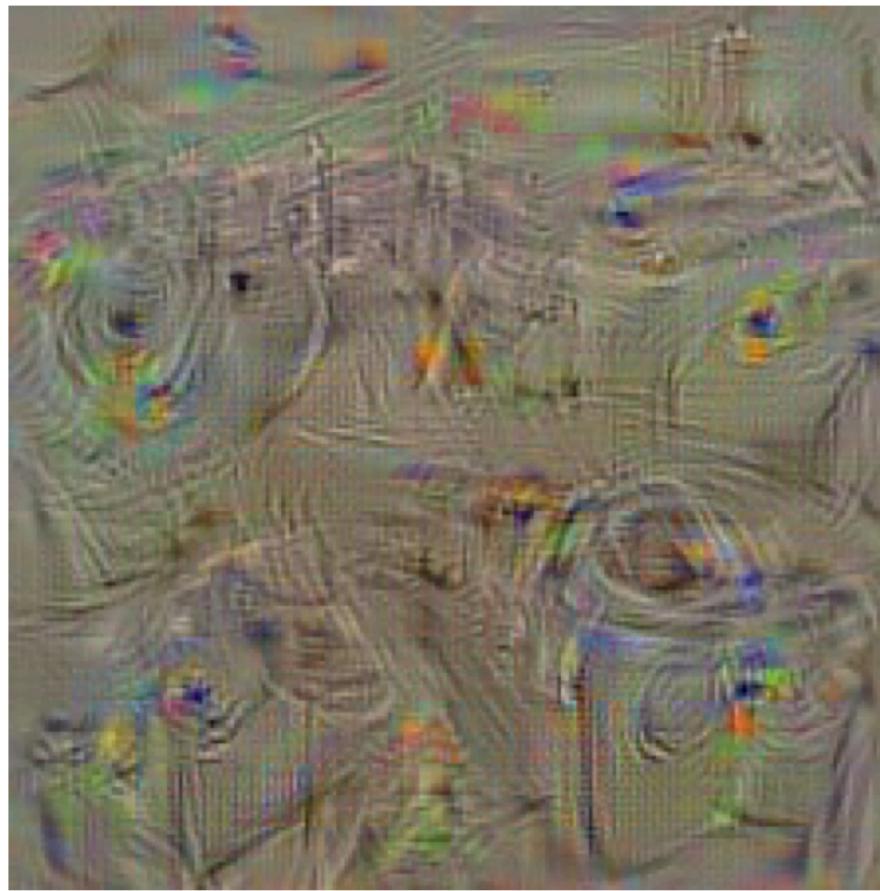
bell pepper



lemon



husky



# Максимизация вероятности класса



Hartebeest



Measuring Cup



Ant



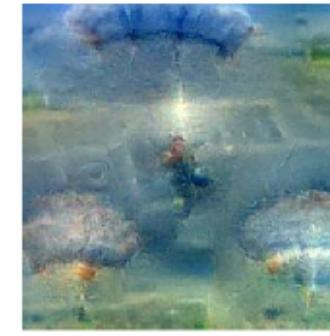
Starfish



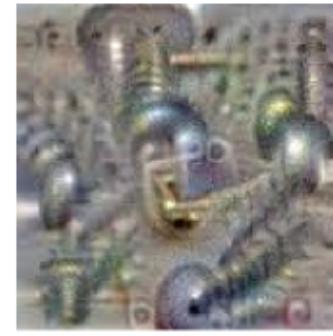
Anemone Fish



Banana

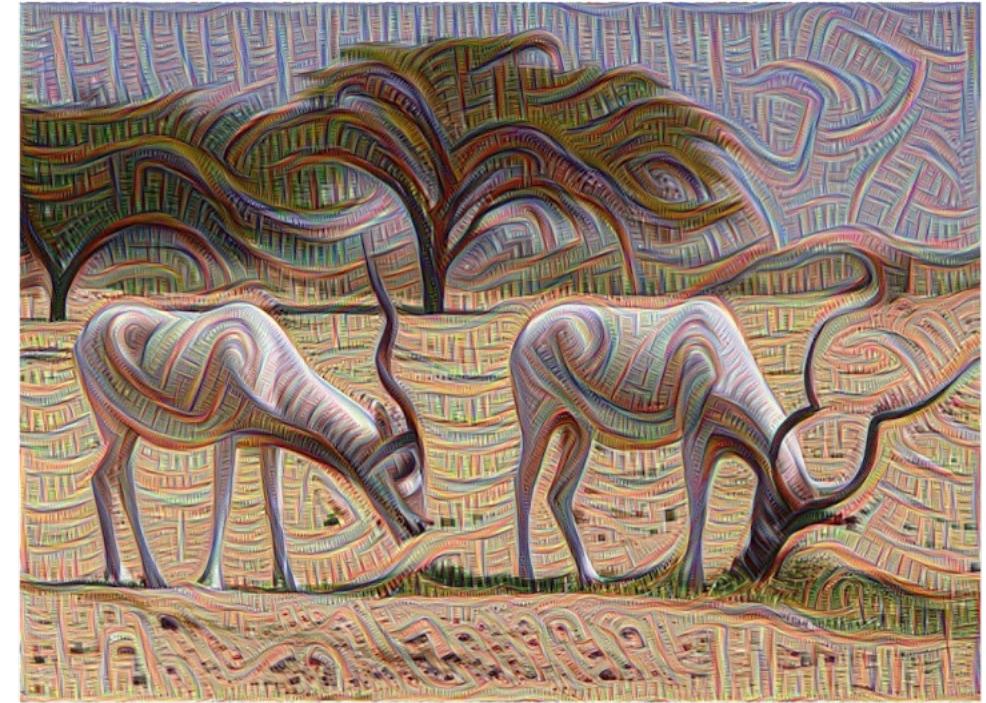


Parachute

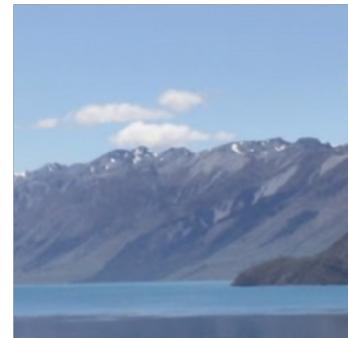


Screw

# Максимизация вероятности класса



# Максимизация вероятности класса



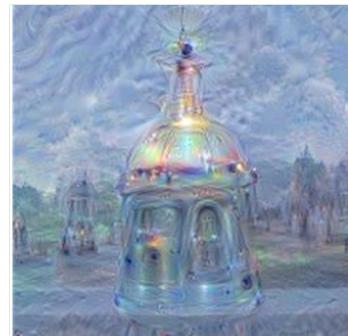
Horizon



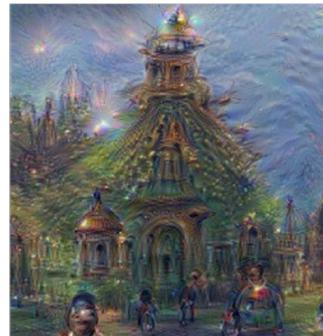
Trees



Leaves



Towers & Pagodas



Buildings



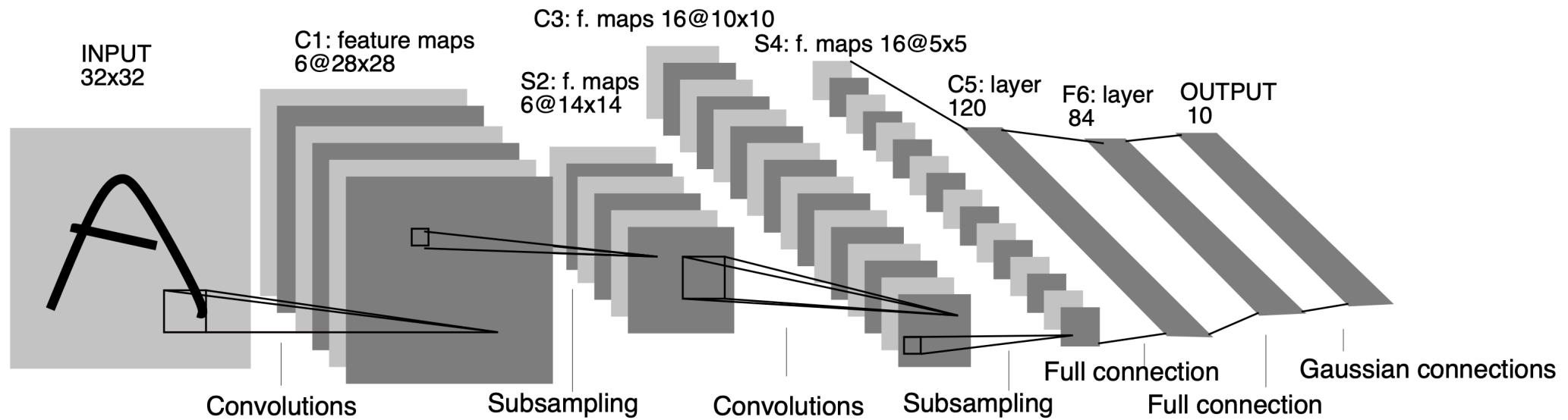
Birds & Insects

# Максимизация вероятности класса



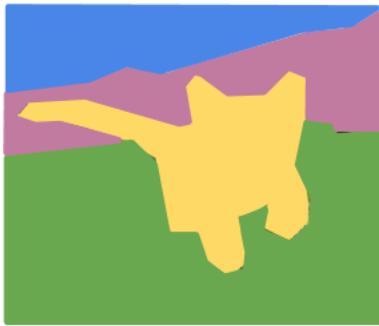
# Компьютерное зрение

# Классификация изображений



# Обычно хочется другого

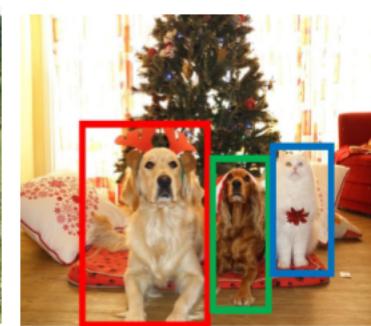
Semantic  
Segmentation



Classification  
+ Localization



Object  
Detection



Instance  
Segmentation

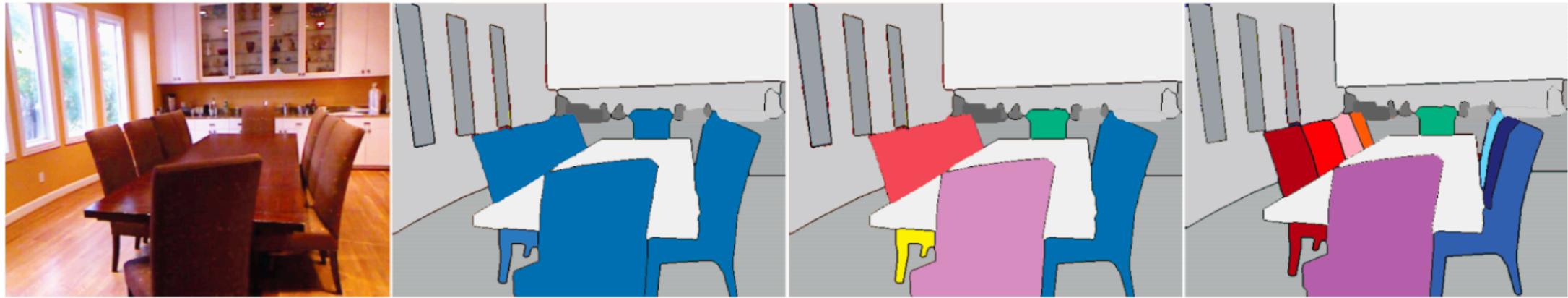


Обычно хочется другого

Но мы будем сводить все задачи к классификации!

# Семантическая сегментация

# Терминология



Semantic  
segmentation

Instance  
segmentation

# Постановка задачи

- Данные: изображения и их корректные сегментации
- Пример: PASCAL VOC 2012



# Постановка задачи

Table 2: Statistics of the segmentation image sets.

	train		val		trainval		test	
	img	obj	img	obj	img	obj	img	obj
<b>Aeroplane</b>	88	108	90	110	178	218	—	—
<b>Bicycle</b>	65	94	79	103	144	197	—	—
<b>Bird</b>	105	137	103	140	208	277	—	—
<b>Boat</b>	78	124	72	108	150	232	—	—
<b>Bottle</b>	87	195	96	162	183	357	—	—
<b>Bus</b>	78	121	74	116	152	237	—	—
<b>Car</b>	128	209	127	249	255	458	—	—
<b>Cat</b>	131	154	119	132	250	286	—	—
<b>Chair</b>	148	303	123	245	271	548	—	—
<b>Cow</b>	64	152	71	132	135	284	—	—
<b>Diningtable</b>	82	86	75	82	157	168	—	—
<b>Dog</b>	121	149	128	150	249	299	—	—
<b>Horse</b>	68	100	79	104	147	204	—	—
<b>Motorbike</b>	81	101	76	103	157	204	—	—
<b>Person</b>	442	868	445	865	887	1733	—	—
<b>Pottedplant</b>	82	151	85	171	167	322	—	—
<b>Sheep</b>	63	155	57	153	120	308	—	—
<b>Sofa</b>	93	103	90	106	183	209	—	—
<b>Train</b>	83	96	84	93	167	189	—	—
<b>Tvmonitor</b>	84	101	74	98	158	199	—	—
<b>Total</b>	1464	3507	1449	3422	2913	6929	—	—

# Постановка задачи

- Каждый объект содержит сильно больше информации, чем при классификации!
- Можно хорошо обучаться на небольших выборках

# Метрики качества

- Попиксельная доля верных ответов:

$$L(y, a) = \frac{1}{n} \sum_{i=1}^n [y_i = a_i]$$

- Мера Жаккара (считается отдельно для каждого класса):

$$J_k(y, a) = \frac{\sum_{i=1}^n [y_i = k][a_i = k]}{\sum_{i=1}^n \max([y_i = k], [a_i = k])}$$

(можно усреднить по всем классам)

# ФУНКЦИЯ ПОТЕРЬ

- Для одного изображения:

$$L(y, a) = \sum_{i=1}^n \sum_{k=1}^K [y_i = k] \log a_{ik}$$

сумма по пикселям

сумма по классам

истинный класс в  $i$ -м пикселе

вероятность  $k$ -го класса в  $i$ -м пикселе  
(из модели)

# ФУНКЦИЯ ПОТЕРЬ

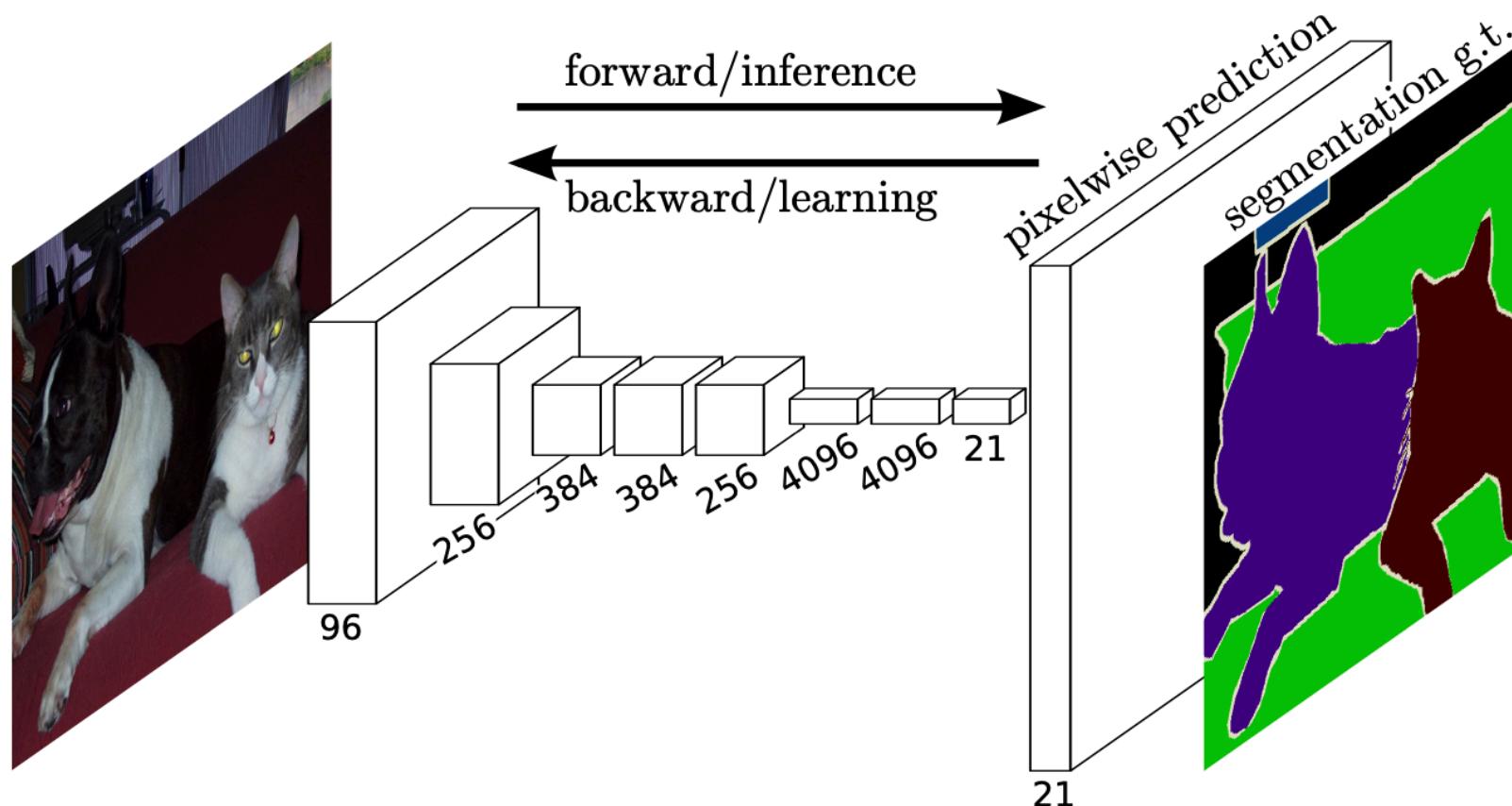
- Для одного изображения (categorical cross-entropy, CCE):

$$L(y, a) = \sum_{i=1}^n \sum_{k=1}^K [y_i = k] \log a_{ik}$$

- Если модель в  $i$ -м пикселе выдаёт какие-то числа  $b_{i1}, \dots, b_{iK}$ , то их можно превратить в вероятности через softmax:

$$a_{ik} = \frac{\exp(b_{ik})}{\sum_{m=1}^K \exp(b_{im})}$$

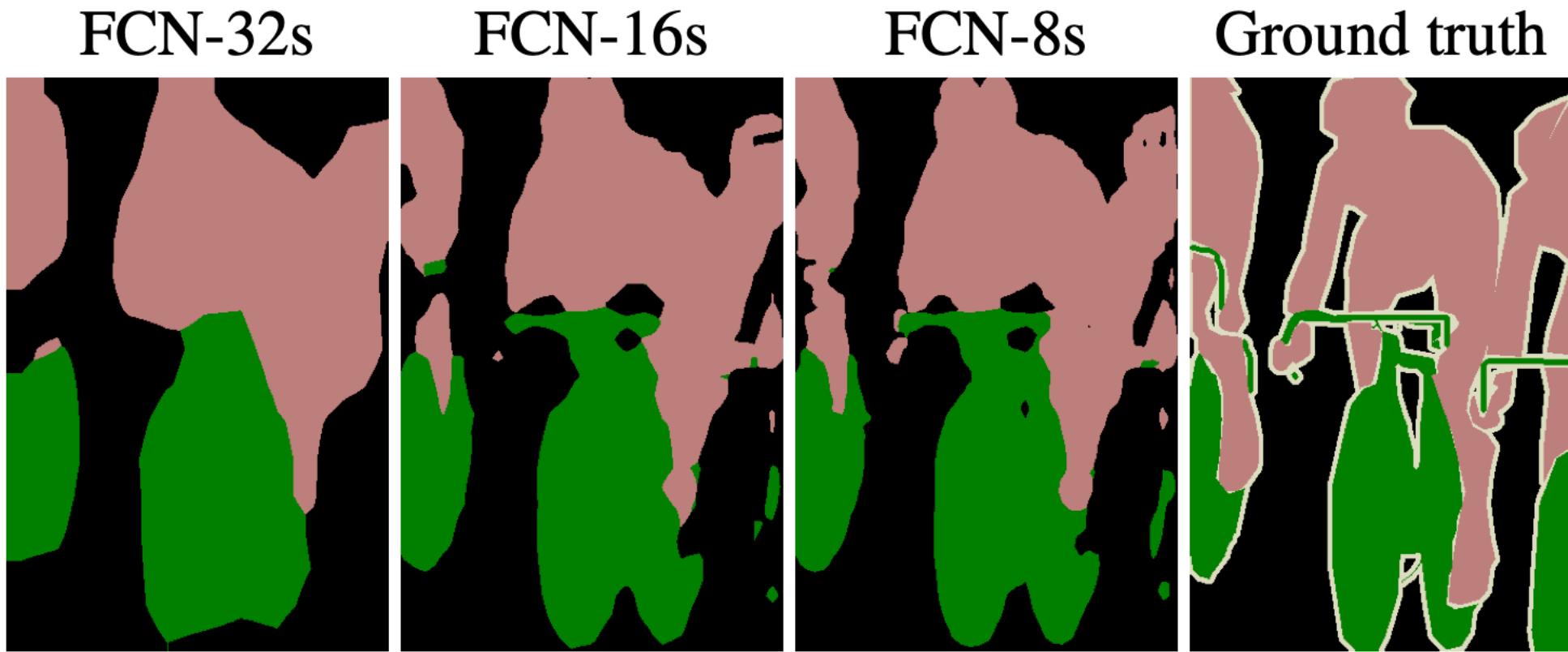
# Fully Convolutional Network



# Fully Convolutional Network

- Убираем полносвязные слои
- Остаются только свёртки
- Тензор с последнего слоя преобразуем свёртками  $1 \times 1$  в тензор такого же размера, но с  $K$  каналами (по числу классов)
- Повышаем разрешение
  - Можно простым размножением пикселей
  - Можно хитрее

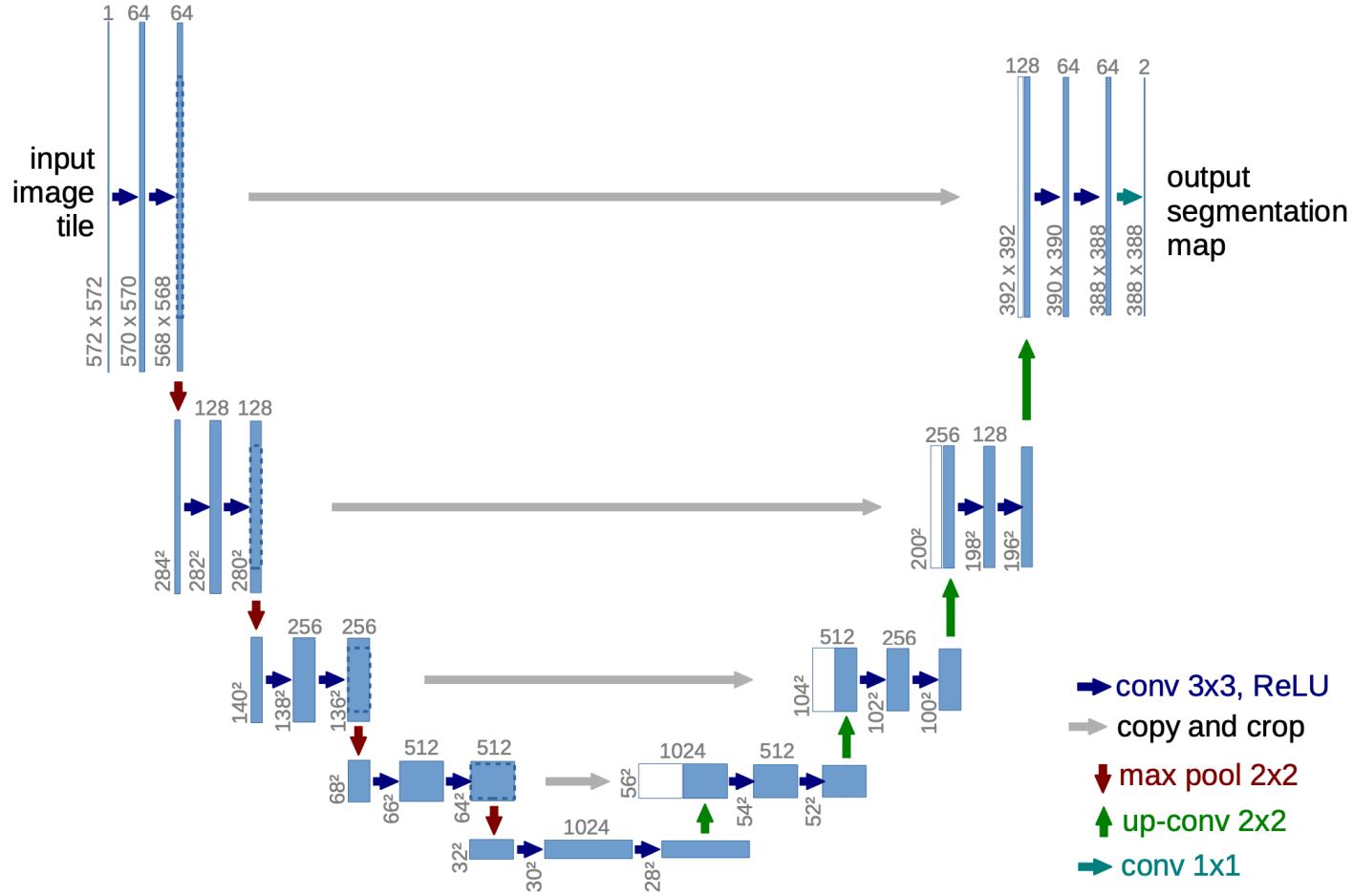
# Fully Convolutional Network



## Чем это плохо?

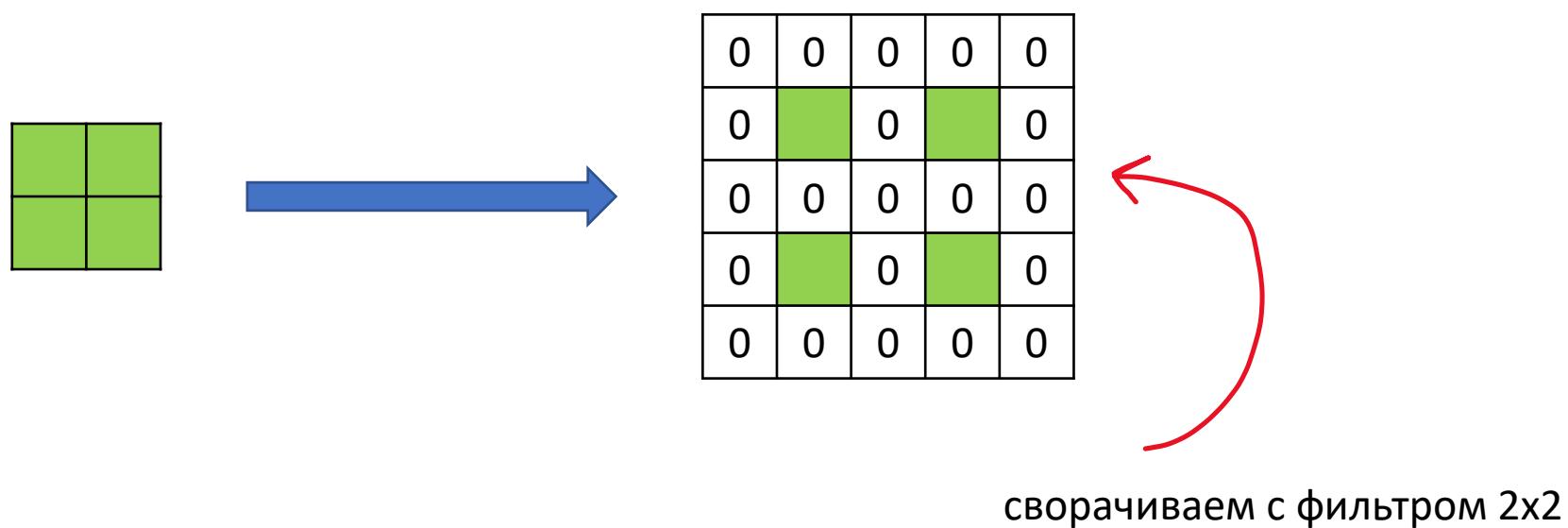
- Тензор на последнем слое довольно маленький
- Классы предсказываются грубо, у объектов нечёткие границы
- Можно делать меньше пулингов
- Но тогда будут проблемы с размером поля восприятия

# U-Net





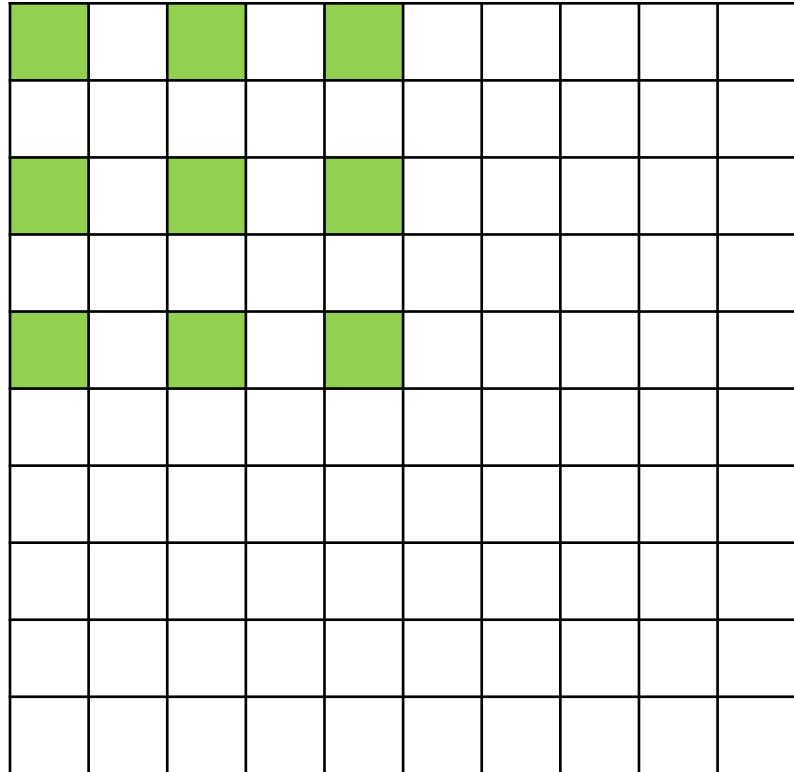
# Upsampling, вариант 2 (up-convolution)



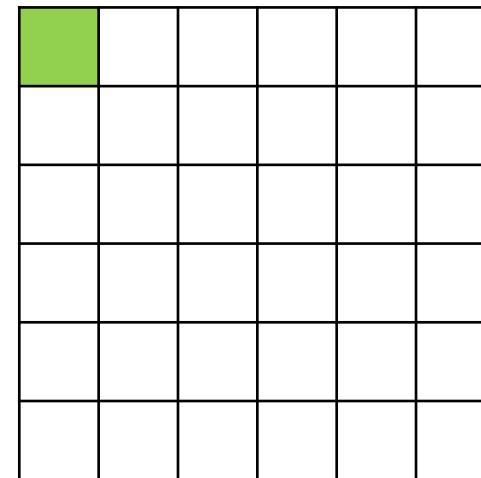
# U-Net

- «Декодировщик» имеет очень большое поле восприятия
- Главное отличие от FCN — копирование слоёв между ветками сети

# Dilated convolutions («раздутые» свёртки)

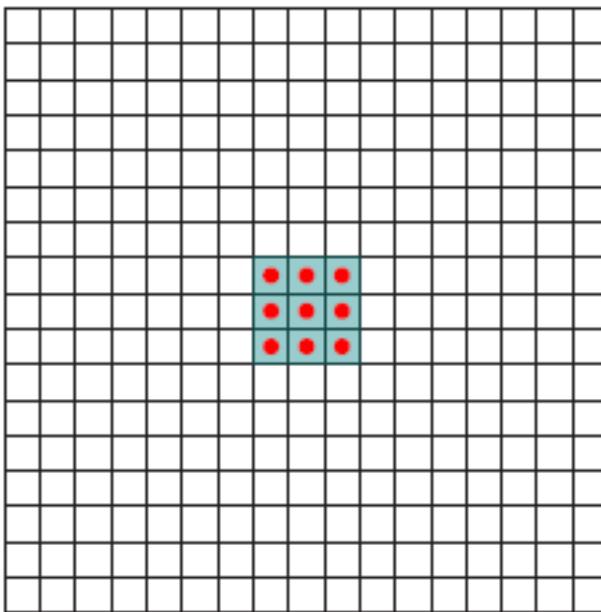


$$\begin{matrix} * & \begin{matrix} \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \end{matrix} & = \end{matrix}$$

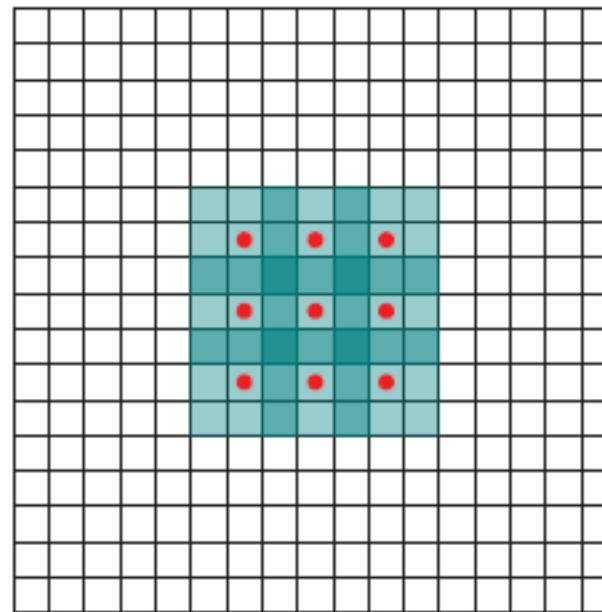


$$l = 2$$

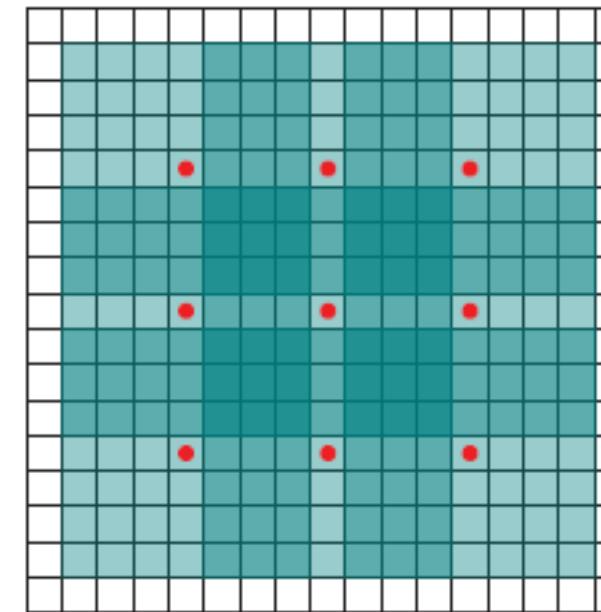
# Dilated convolutions («раздутые» свёртки)



$$l = 1$$



$$l = 2$$



$$l = 4$$

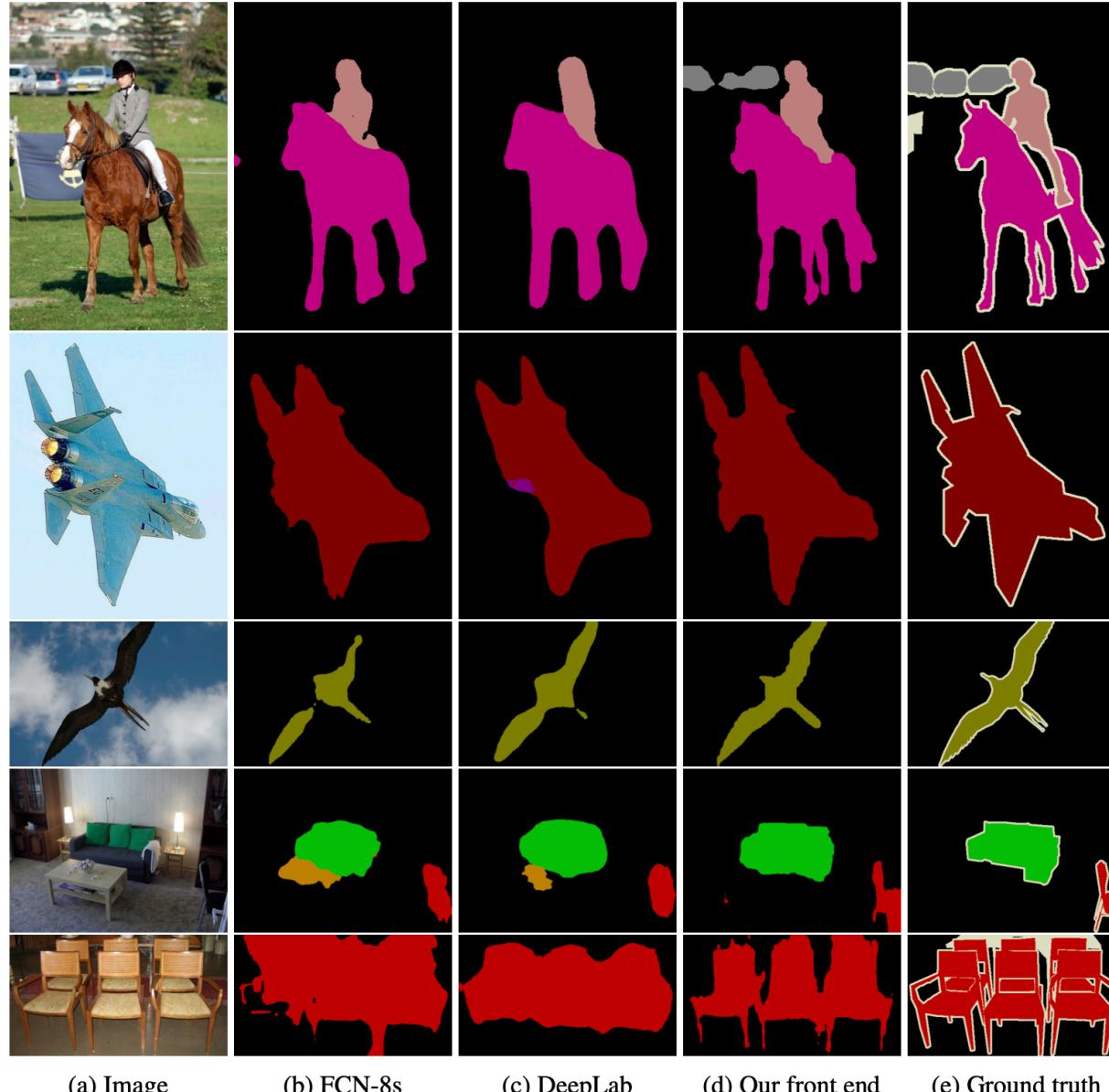
# Dilated convolutions

Layer	1	2	3	4	5	6	7	8
Convolution	$3 \times 3$	$3 \times 3$	$3 \times 3$	$3 \times 3$	$3 \times 3$	$3 \times 3$	$3 \times 3$	$1 \times 1$
Dilation	1	1	2	4	8	16	1	1
Truncation	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Receptive field	$3 \times 3$	$5 \times 5$	$9 \times 9$	$17 \times 17$	$33 \times 33$	$65 \times 65$	$67 \times 67$	$67 \times 67$
Output channels								
Basic	$C$	$C$	$C$	$C$	$C$	$C$	$C$	$C$
Large	$2C$	$2C$	$4C$	$8C$	$16C$	$32C$	$32C$	$C$

Table 1: Context network architecture. The network processes  $C$  feature maps by aggregating contextual information at progressively increasing scales without losing resolution.

# Dilated convolutions

- Можем сохранять размер тензора и при этом увеличивать поле восприятия



(a) Image

(b) FCN-8s

(c) DeepLab

(d) Our front end

(e) Ground truth