

Основы глубинного обучения

Лекция 6

Компьютерное зрение. Работа с текстами.

Евгений Соколов

esokolov@hse.ru

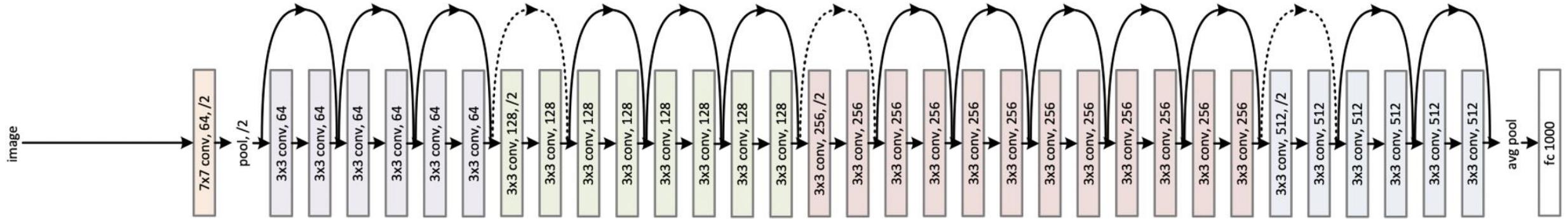
НИУ ВШЭ, 2024

Transfer learning

Перенос знаний

- ImageNet:
 - Много данных (которые сложно собрать!)
 - Годы улучшений
- Не хотелось бы повторять это для новых задач

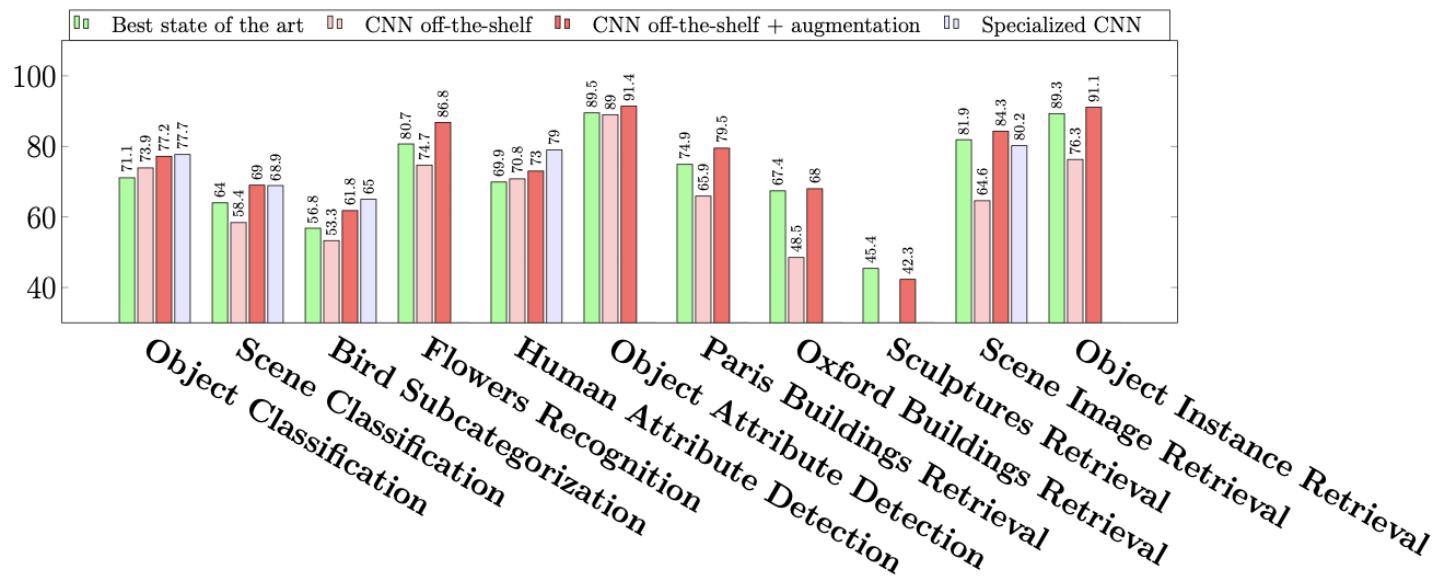
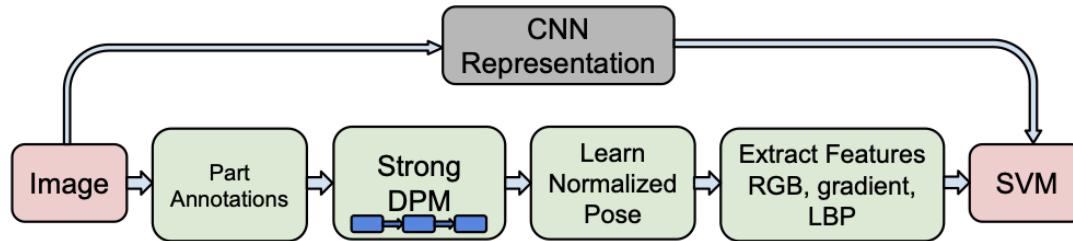
Дообучение



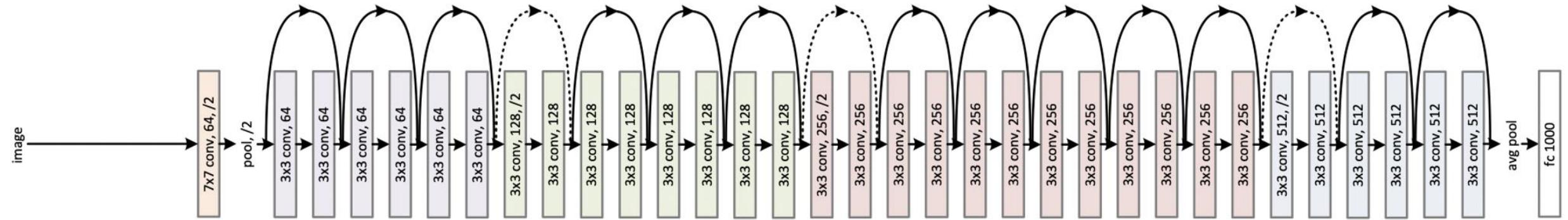
Если данных совсем мало:

- Берём модель из другой задачи
- Заменяем последний слой на слой с нужным числом выходов
- Обучаем только его
- По сути, это обучение линейной модели

Представления с последних слоёв



Дообучение



Если данных не очень мало:

- Берём модель из другой задачи
- Заменяем последний слой на слой с нужным числом выходов
- Обучаем его и несколько слоёв до него
- Чем ближе к началу слой, тем ниже стоит делать градиентный шаг

Дообучение

- Как правило, на первых слоях фильтры похожие для всех задач
- Чем сильнее новая задача отличается от исходной, тем больше слоёв нужно переучивать
- В любом случае выходы последних слоёв модели, обученной на ImageNet, лучше случайных признаков

Интерпретация моделей

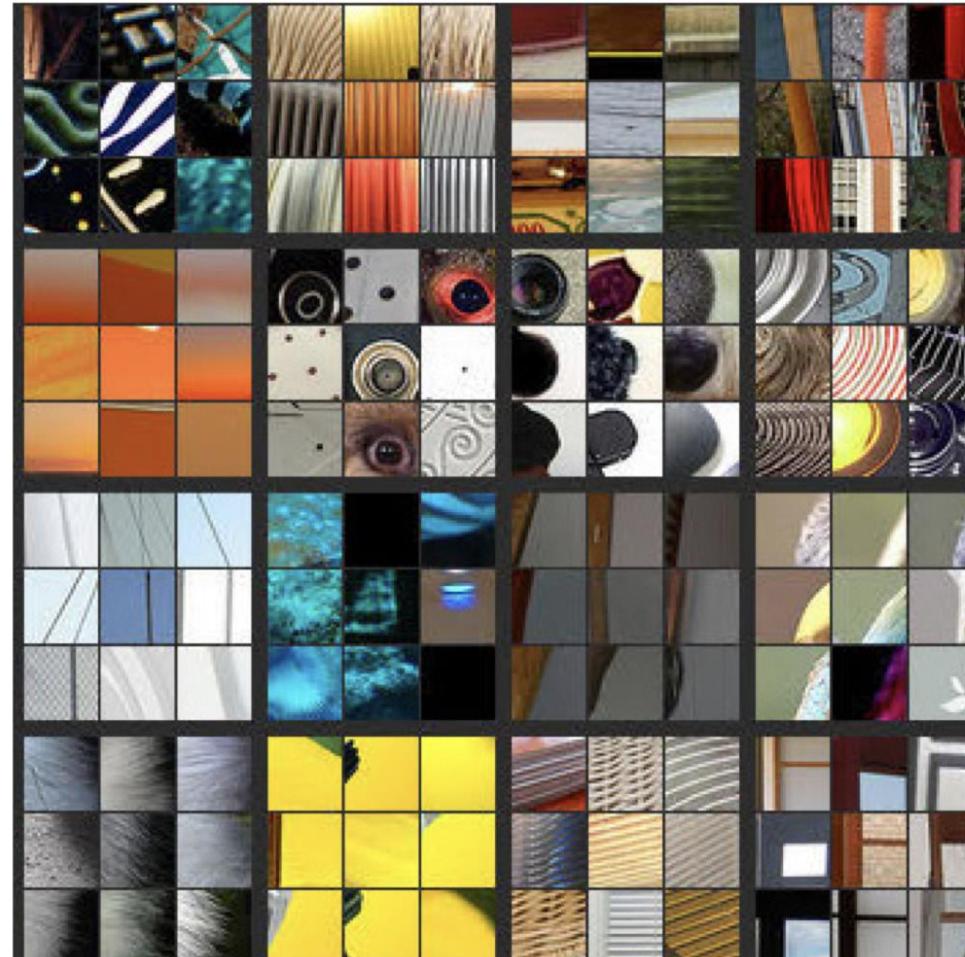
Что находят свёрточные сети?

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

Слой 1



Слой 2



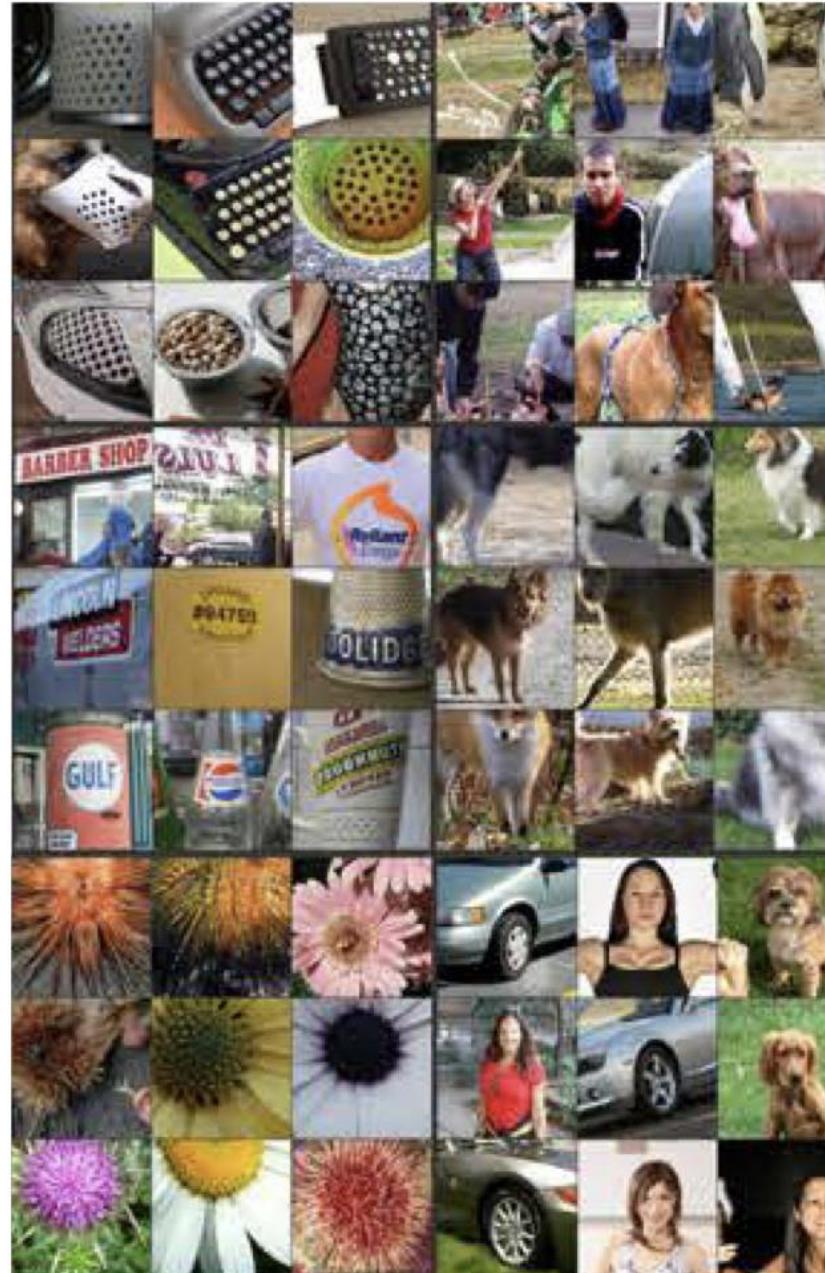
Слой 3



Слой 4



Слой 5

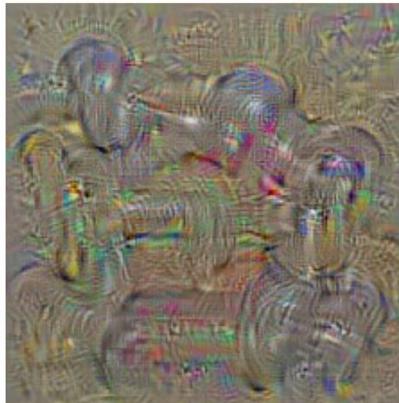


Максимизация вероятности класса

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

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

Максимизация вероятности класса



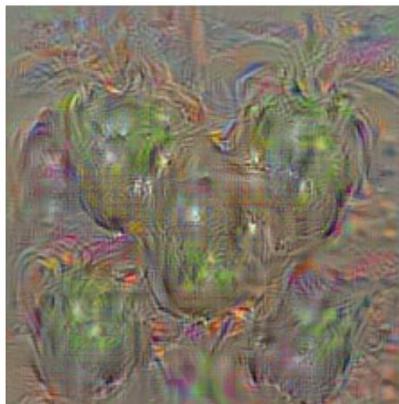
dumbbell



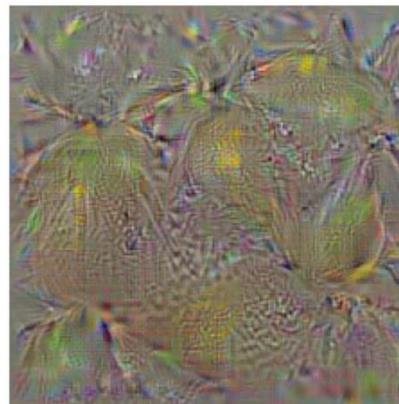
cup



dalmatian



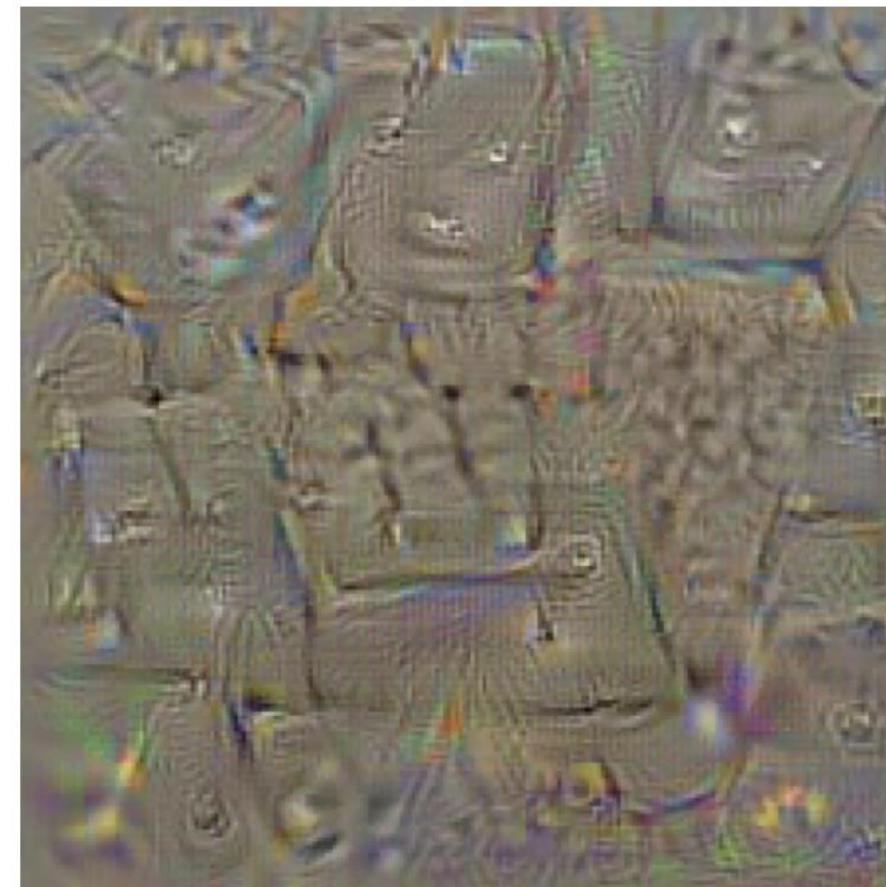
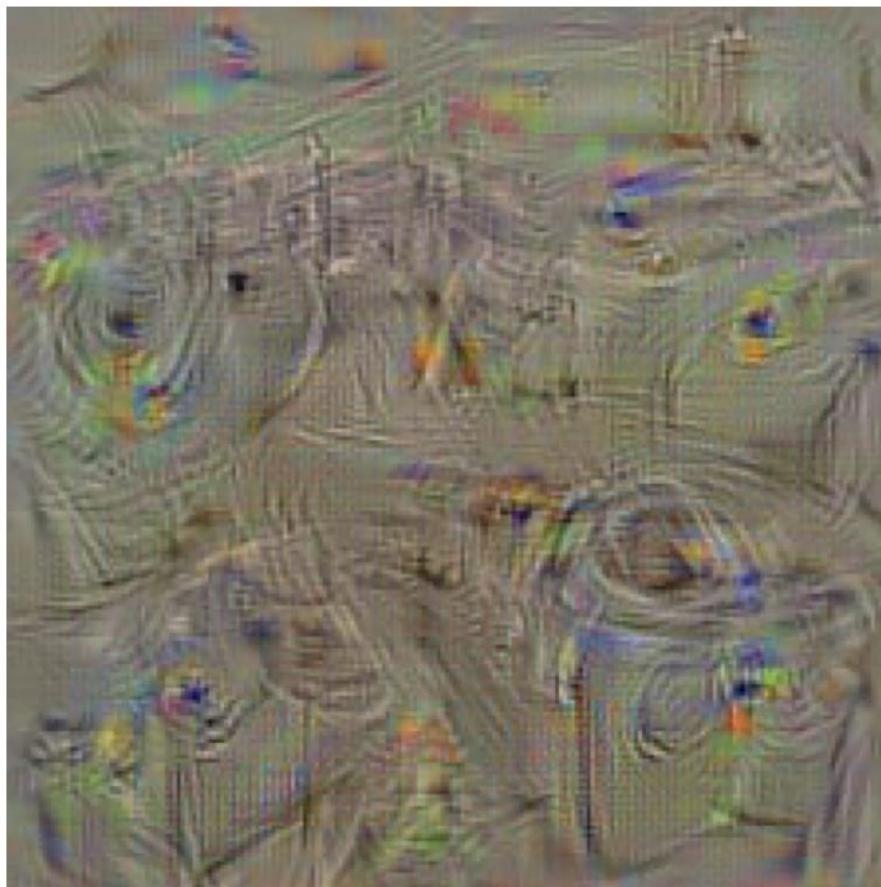
bell pepper



lemon



husky



Максимизация вероятности класса



Hartebeest



Measuring Cup



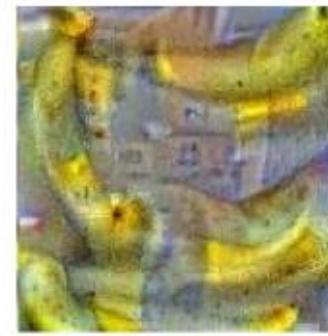
Ant



Starfish



Anemone Fish



Banana

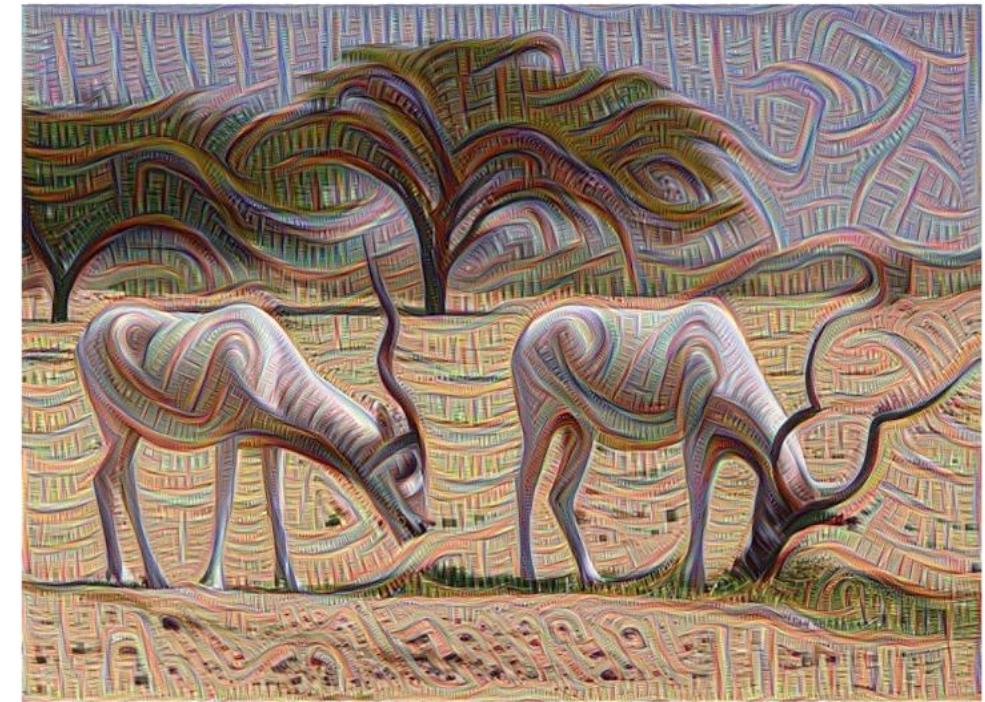


Parachute

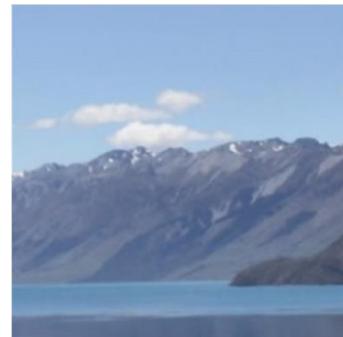


Screw

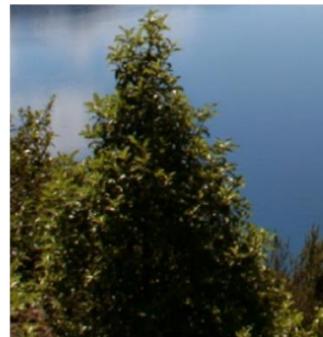
Максимизация вероятности класса



Максимизация вероятности класса



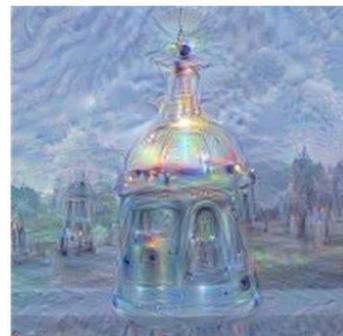
Horizon



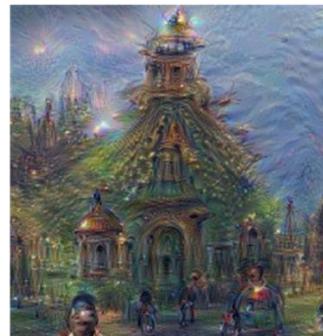
Trees



Leaves



Towers & Pagodas



Buildings



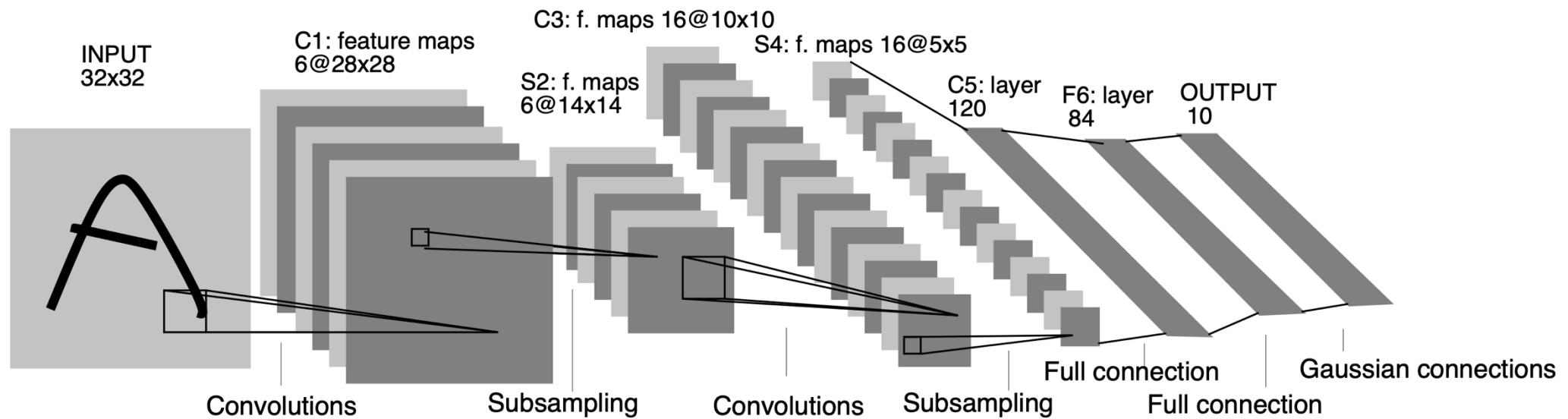
Birds & Insects

Максимизация вероятности класса



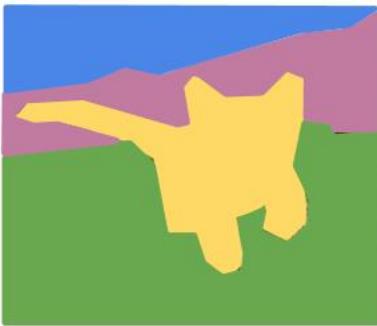
Компьютерное зрение

Классификация изображений



Обычно хочется другого

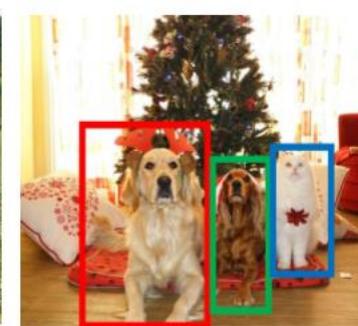
**Semantic
Segmentation**



**Classification
+ Localization**



**Object
Detection**



**Instance
Segmentation**

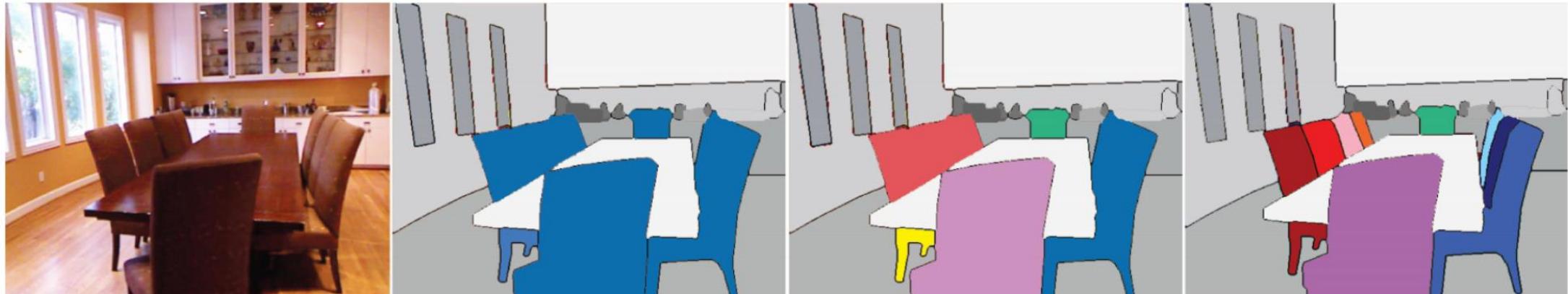


Обычно хочется другого

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

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

The diagram shows the mathematical expression for the cross-entropy loss function. Red arrows point from the labels below the equation to specific parts of the formula:

- The first red arrow points to the inner summation over i , labeled "сумма по пикселям" (sum over pixels).
- The second red arrow points to the inner summation over k , labeled "сумма по классам" (sum over classes).
- The third red arrow points to the term $[y_i = k]$, labeled "истинный класс в i -м пикселе" (true class in i -th pixel).
- The fourth red arrow points to the term $\log a_{ik}$, labeled "вероятность k -го класса в i -м пикселе (из модели)" (probability of k -th class in i -th pixel from the model).

ФУНКЦИЯ ПОТЕРЬ

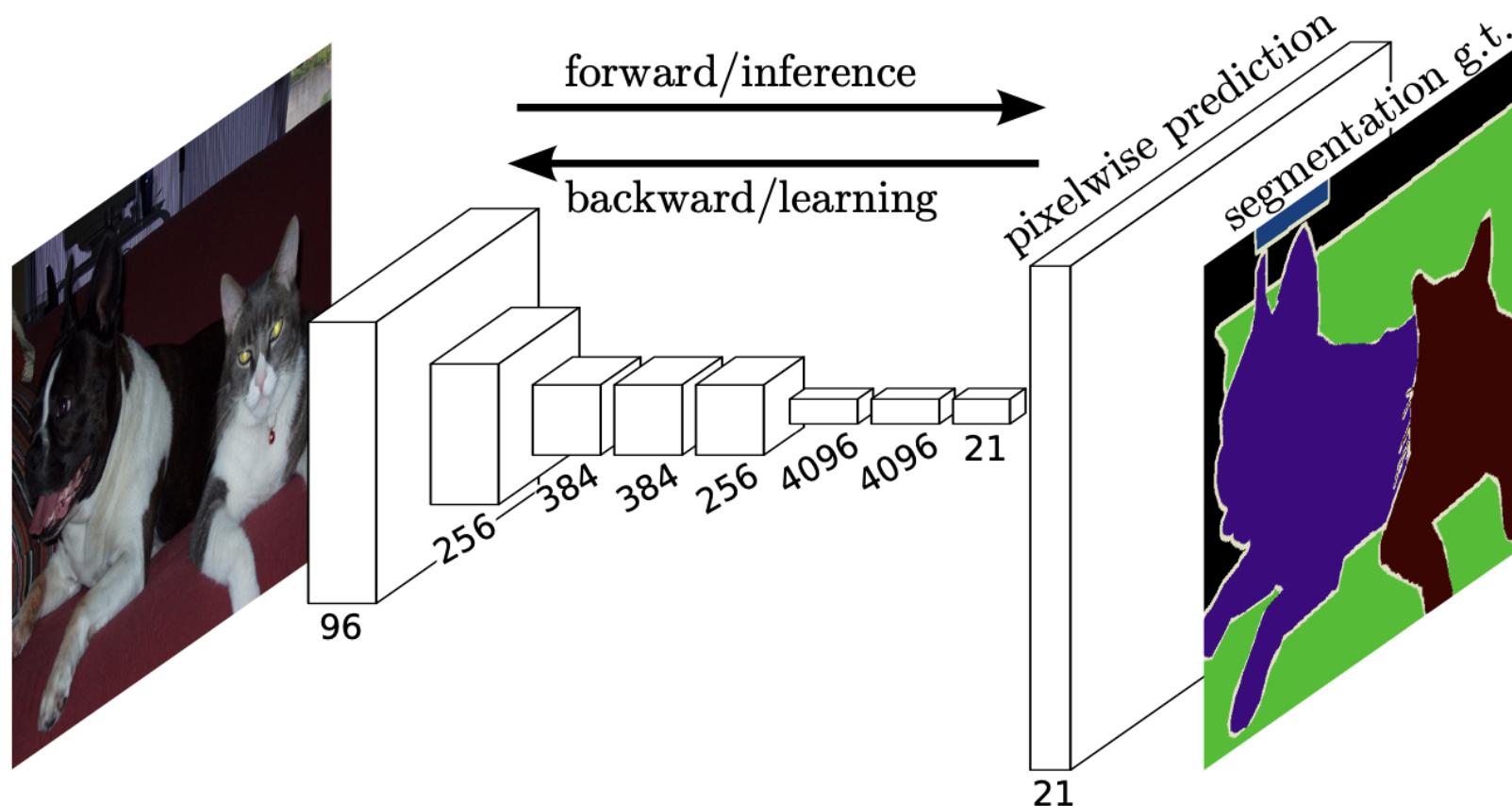
- Для одного изображения (categorical cross-entropy, CCE):

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

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

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

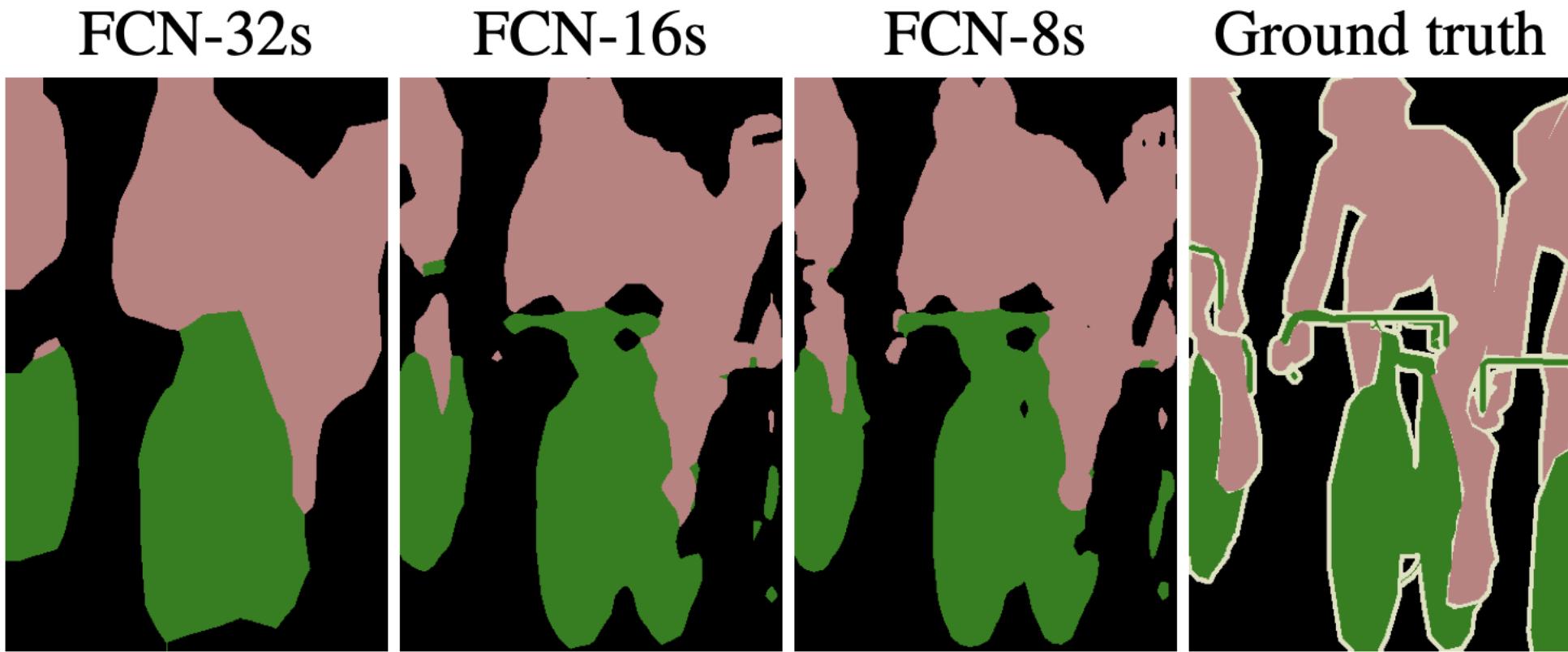
Fully Convolutional Network



Fully Convolutional Network

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

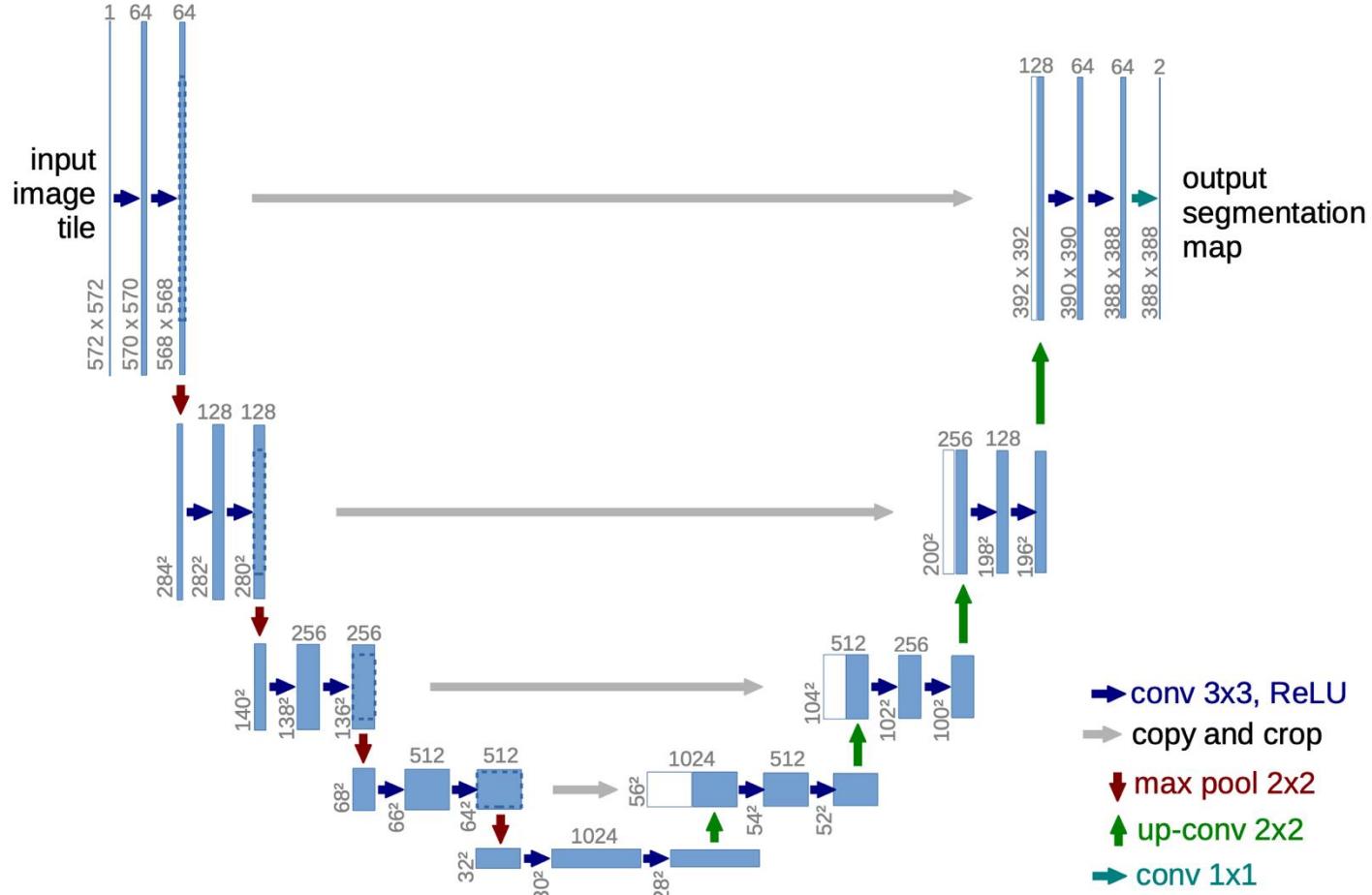
Fully Convolutional Network



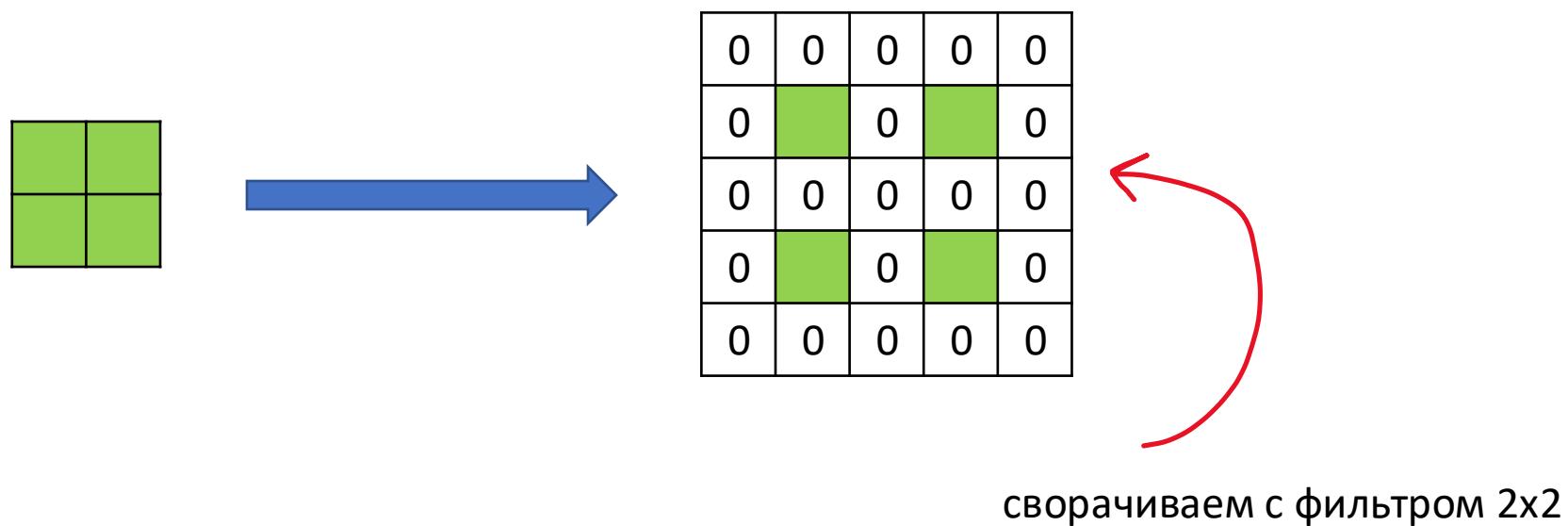
Чем это плохо?

- 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



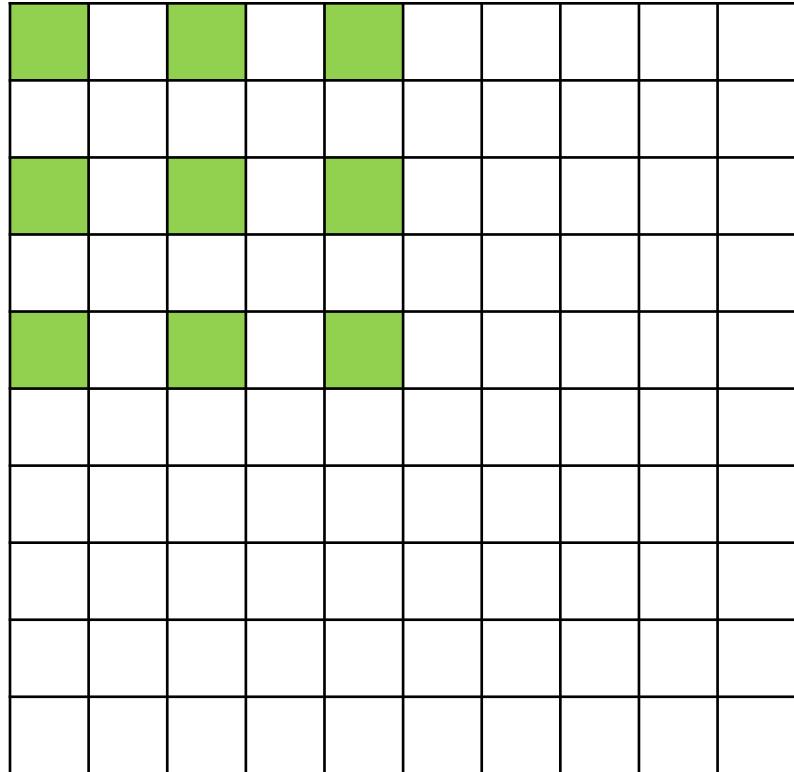
Upsampling, вариант 2 (up-convolution)



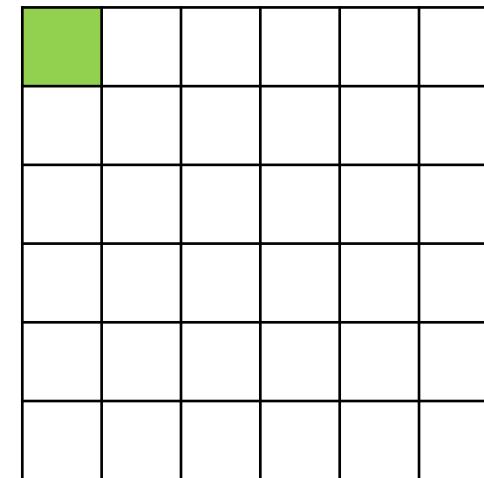
U-Net

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

Dilated convolutions («раздутые» свёртки)

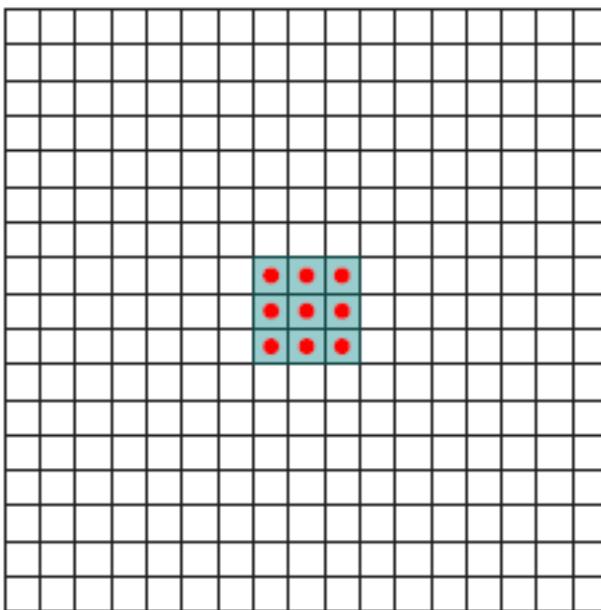


$$\begin{matrix} * & \begin{matrix} & & \\ & & \\ & & \end{matrix} & = \end{matrix}$$

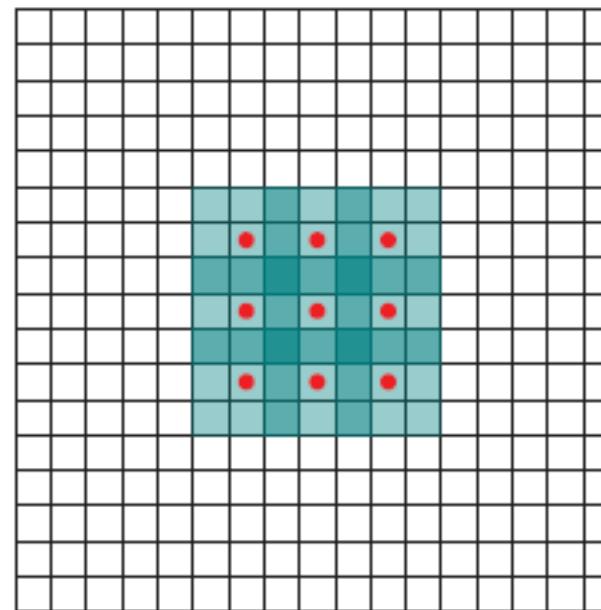


$$l = 2$$

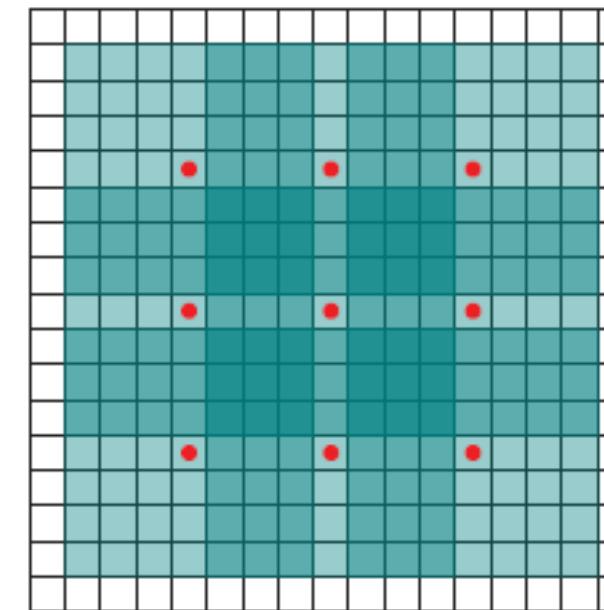
Dilated convolutions («раздутые» свёртки)



$$l = 1$$



$$l = 2$$



$$l = 4$$

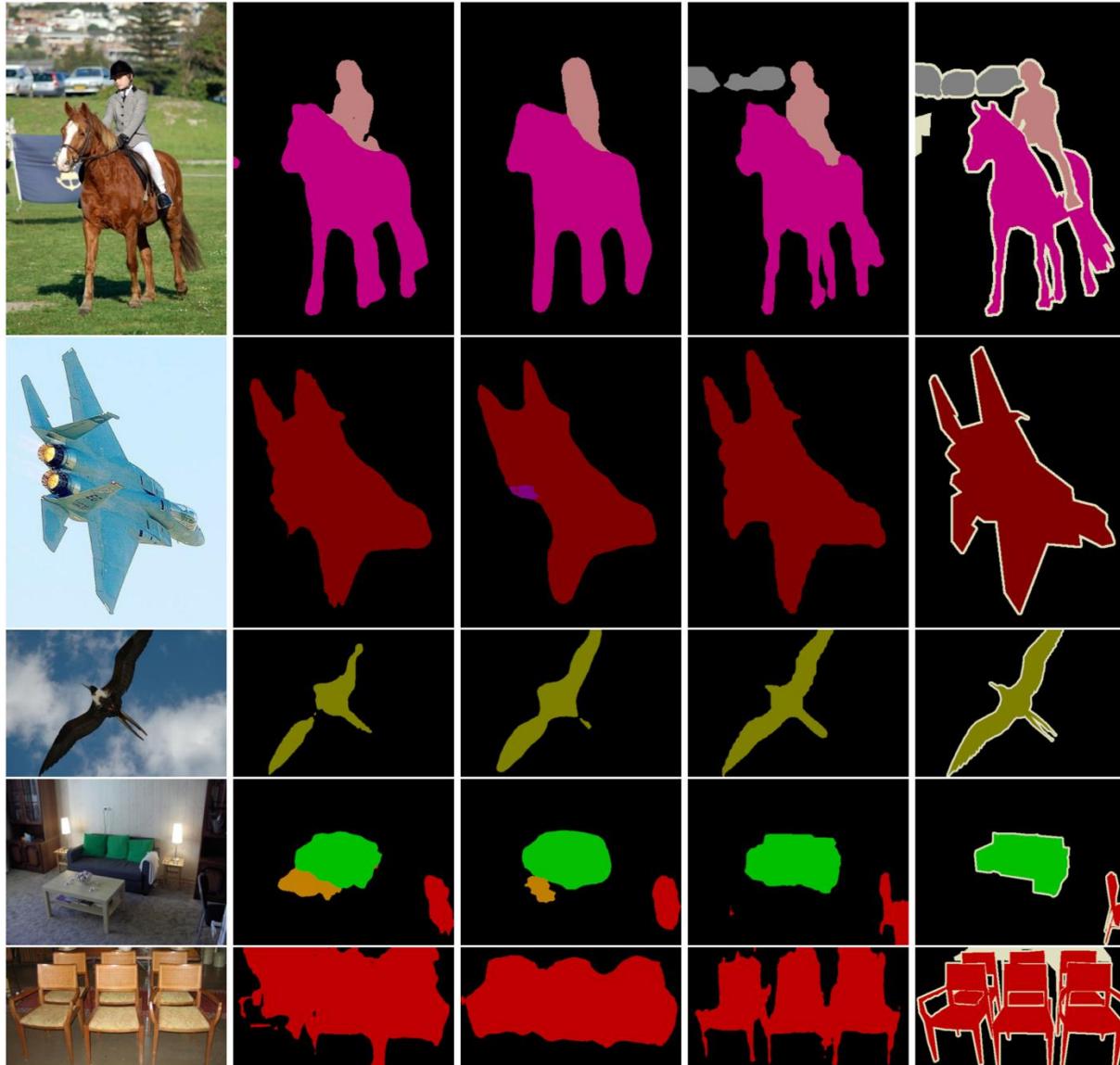
Dilated convolutions

Layer	1	2	3	4	5	6	7	8
Convolution	3×3	3×3	3×3	3×3	3×3	3×3	3×3	1×1
Dilation	1	1	2	4	8	16	1	1
Truncation	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Receptive field	3×3	5×5	9×9	17×17	33×33	65×65	67×67	67×67
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

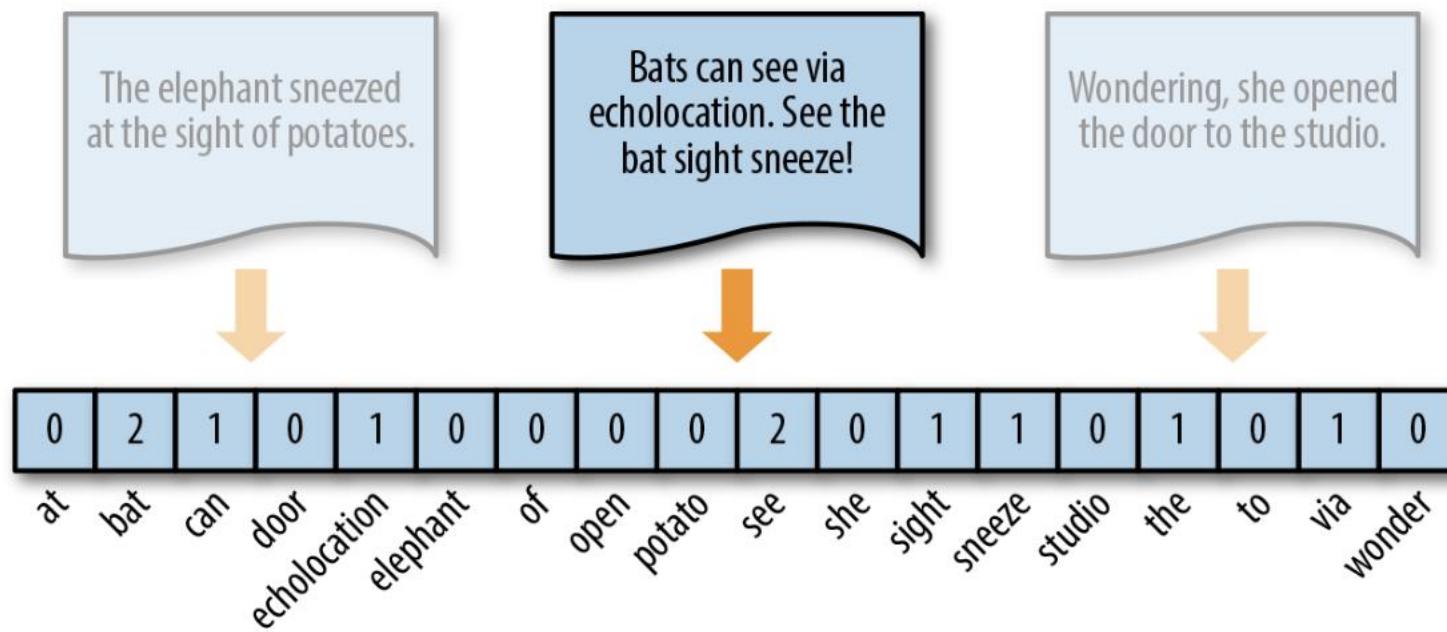
(e) Ground truth

Представления для текстов

Bag of words

- Заводим словарь, состоящий из всех слов в выборке
- Делаем признак-индикатор для каждого слова из словаря
- Можно добавлять n-граммы

Bag of words



Bag of words

- Слишком много признаков
- Не учитываем смыслы слов
- Семантически похожие тексты могут иметь очень разные представления

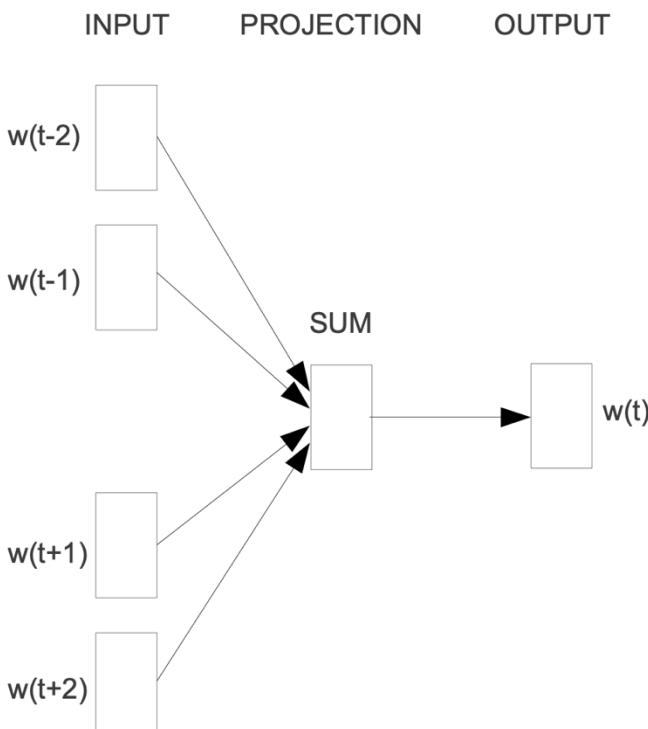
word2vec

- Попробуем обучить вектор-представление для каждого слова
- Что потребовать от такого представления?

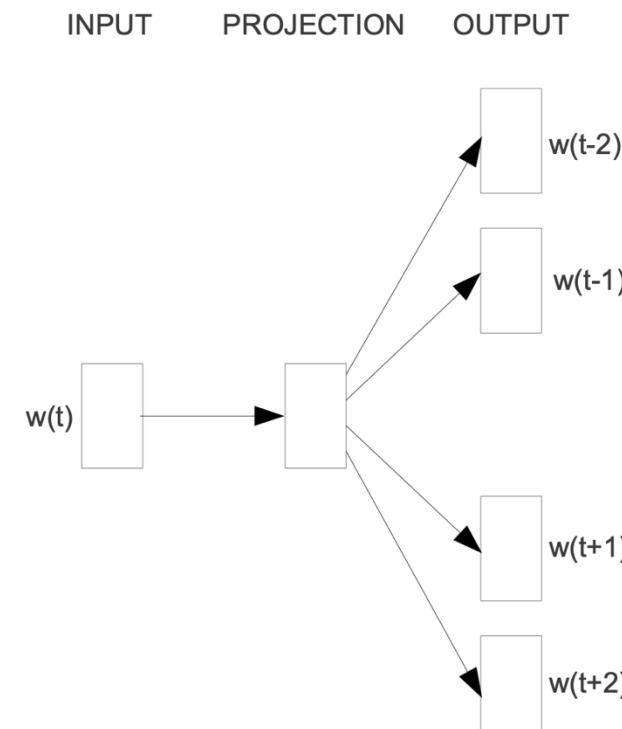
word2vec

- Попробуем обучить вектор-представление для каждого слова
- Что потребовать от такого представления?
- Важная идея: если выкинуть слово, то оно должно хорошо восстанавливаться по представлениям соседних слов
- Может применять и при работе с изображениями

word2vec



CBOW



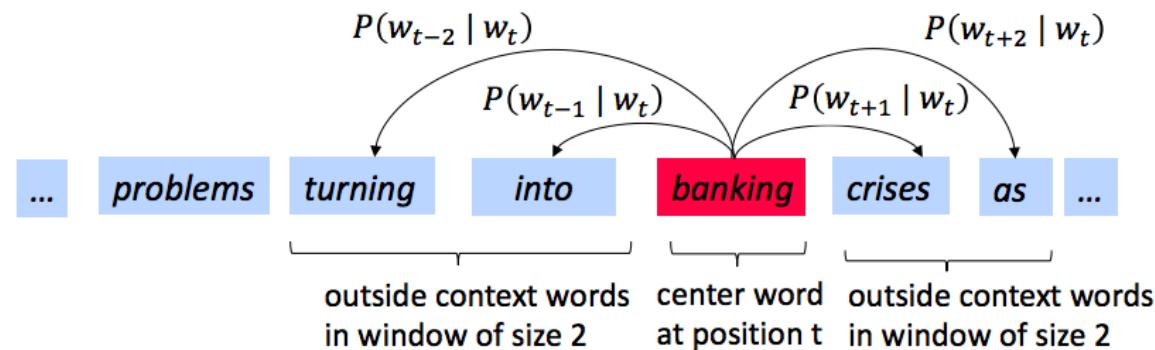
Skip-gram

Skip-gram model

- Вероятность встретить слово w_O рядом со словом w_I :

$$p(w_O | w_I) = \frac{\exp(\langle v'_{w_O}, v_{w_I} \rangle)}{\sum_{w \in W} \exp(\langle v'_w, v_{w_I} \rangle)}$$

- W — словарь
- v_w — «центральное» представление слова
- v'_w — «контекстное» представление слова



Skip-gram model

- Вероятность встретить слово w_O рядом со словом w_I :

$$p(w_O|w_I) = \frac{\exp(\langle v'_{w_O}, v_{w_I} \rangle)}{\sum_{w \in W} \exp(\langle v'_w, v_{w_I} \rangle)}$$

- Функционал для текста $T = (w_1 w_2 \dots w_n)$:

$$\sum_{i=1}^n \sum_{\substack{-c \leq j \leq c \\ j \neq 0}} \log p(w_{i+j}|w_i) \rightarrow \max$$

Skip-gram model

- Вероятность встретить слово w_O рядом со словом w_I :

$$p(w_O | w_I) = \frac{\exp(\langle v'_{w_O}, v_{w_I} \rangle)}{\sum_{w \in W} \exp(\langle v'_w, v_{w_I} \rangle)}$$

- Считать знаменатель ОЧЕНЬ затратно
- Значит, и производные считать тоже долго

Negative sampling

$$p(w_O | w_I) = \log \sigma(\langle v'_{w_O}, v_{w_I} \rangle) + \sum_{i=1}^k \log \sigma(-\langle v'_{w_i}, v_{w_I} \rangle)$$

- w_i — случайно выбранные слова
- Слово w генерируется с вероятностью $P(w)$ — шумовое распределение
- $P(w) = \frac{U(w)^{\frac{3}{4}}}{\sum_{v \in W} U(v)^{\frac{3}{4}}}$, $U(v)$ — частота слова v в корпусе текстов

word2vec: особенности обучения

- Положительные примеры — слова, стоящие рядом
- Отрицательные примеры: подбираем к слову «шум», то есть другое слово, которое не находится рядом
- Важно семплировать в SGD слова с учётом их популярности — иначе будем обучаться только на самые частые слова

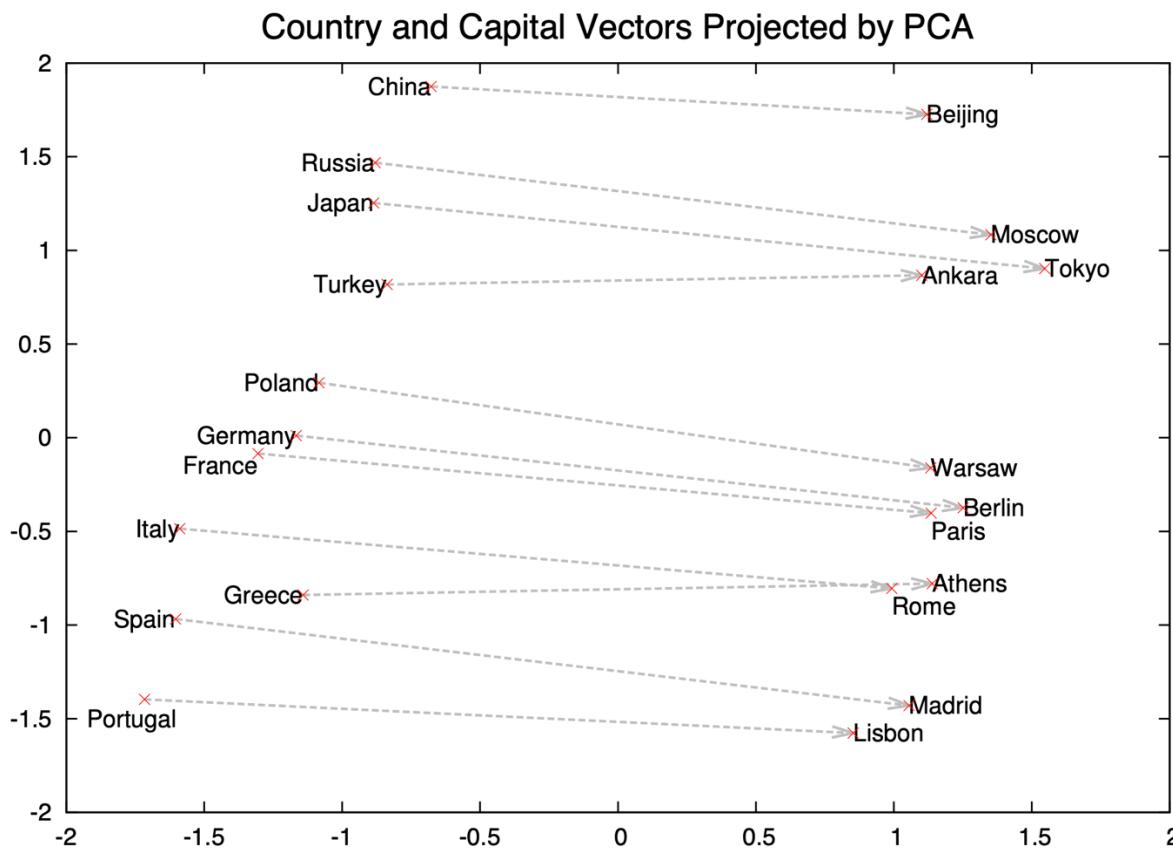
Как это использовать?

- Можно искать похожие слова
- Можно менять формы слов
- Можно искать определённые отношения
- Можно использовать как признаки для моделей

word2vec

Czech + currency	Vietnam + capital	German + airlines	Russian + river	French + actress
koruna	Hanoi	airline Lufthansa	Moscow	Juliette Binoche
Check crown	Ho Chi Minh City	carrier Lufthansa	Volga River	Vanessa Paradis
Polish zolty	Viet Nam	flag carrier Lufthansa	upriver	Charlotte Gainsbourg
CTK	Vietnamese	Lufthansa	Russia	Cecile De

word2vec



word2vec

- Яркий пример self-supervision
- Сейчас находит применения для изображений и даже для табличных данных
- Оказывается, в данных очень много информации даже без разметки

Проблемы word2vec

- Не учитываем структуру слов
- Не закладываем никакой априорной информации о разных формах одного слова
- Не умеем обрабатывать опечатки

FastText

- Заменим каждое слово на «мешок»
- «руслан» -> (<руслан>, <ру, рус, усл, сла, лан, ан>)
- Слово w заменяется на набор токенов t_1, \dots, t_n
- Мы обучаем векторы токенов: v_{t_1}, \dots, v_{t_n} (на самом деле есть «центральные» и «контекстные» версии всех векторов)
- $z_w = \sum_{i=1}^n v_{t_i}$ — вектор слова
- Все остальные детали — как в word2vec

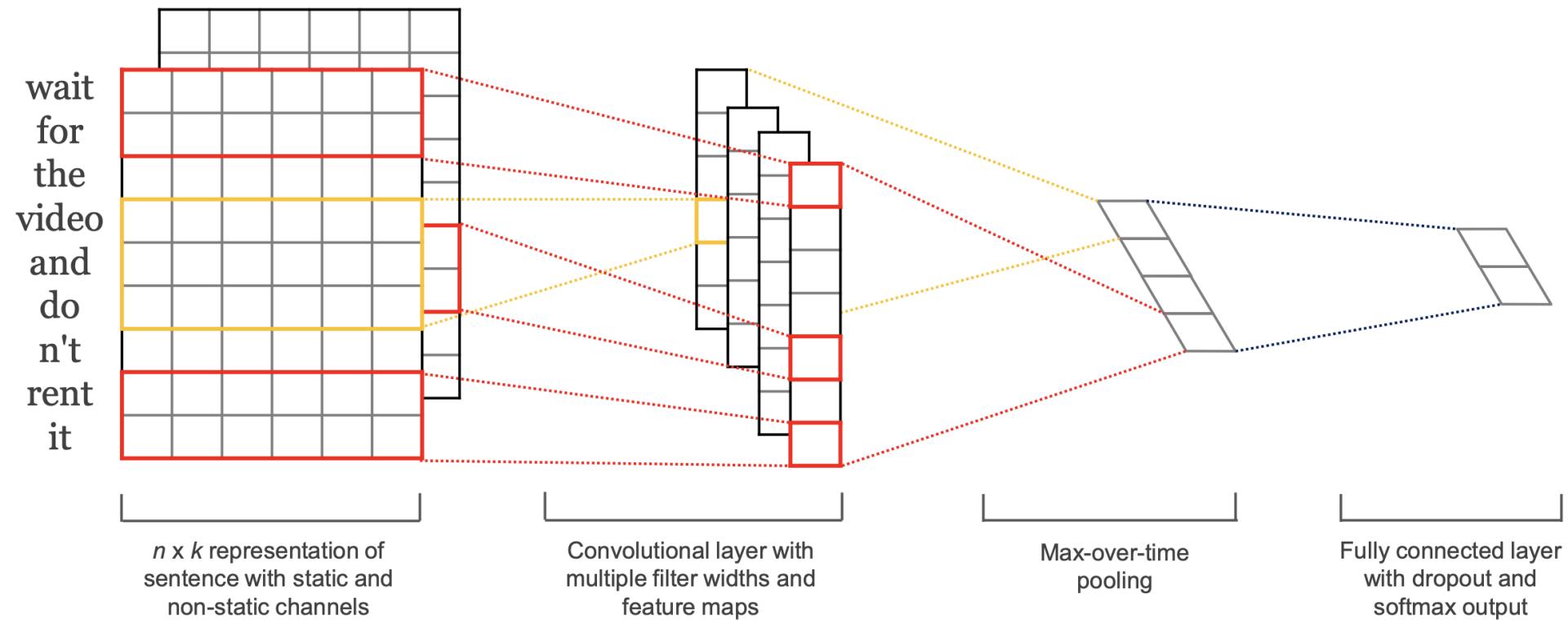
Что бывает ещё?

- GloVe
- ELMo/BERT (в следующих лекциях)

Работа с текстом

- Векторные представления строятся для слов
- Можно просто усреднить по всем словам — получим признаки для текста
- Можно усреднять с весами
- Можно ли умнее?

CNN для последовательностей



CNN для последовательностей

- Можно обучать представления слов с нуля
- А можно инициализировать с помощью w2v — это сильно лучше!

CNN для последовательностей

Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
CNN-rand	76.1	45.0	82.7	89.6	91.2	79.8	83.4
CNN-static	81.0	45.5	86.8	93.0	92.8	84.7	89.6
CNN-non-static	81.5	48.0	87.2	93.4	93.6	84.3	89.5
CNN-multichannel	81.1	47.4	88.1	93.2	92.2	85.0	89.4
RAE (Socher et al., 2011)	77.7	43.2	82.4	—	—	—	86.4
MV-RNN (Socher et al., 2012)	79.0	44.4	82.9	—	—	—	—
RNTN (Socher et al., 2013)	—	45.7	85.4	—	—	—	—
DCNN (Kalchbrenner et al., 2014)	—	48.5	86.8	—	93.0	—	—
Paragraph-Vec (Le and Mikolov, 2014)	—	48.7	87.8	—	—	—	—
CCAE (Hermann and Blunsom, 2013)	77.8	—	—	—	—	—	87.2
Sent-Parser (Dong et al., 2014)	79.5	—	—	—	—	—	86.3
NBSVM (Wang and Manning, 2012)	79.4	—	—	93.2	—	81.8	86.3
MNB (Wang and Manning, 2012)	79.0	—	—	93.6	—	80.0	86.3
G-Dropout (Wang and Manning, 2013)	79.0	—	—	93.4	—	82.1	86.1
F-Dropout (Wang and Manning, 2013)	79.1	—	—	93.6	—	81.9	86.3
Tree-CRF (Nakagawa et al., 2010)	77.3	—	—	—	—	81.4	86.1
CRF-PR (Yang and Cardie, 2014)	—	—	—	—	—	82.7	—
SVM _S (Silva et al., 2011)	—	—	—	—	95.0	—	—

Минусы

- Ищем выразительные «локальные» комбинации
- Не пытаемся понять смысл текста в целом