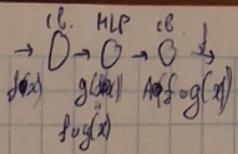


Лекция 4.



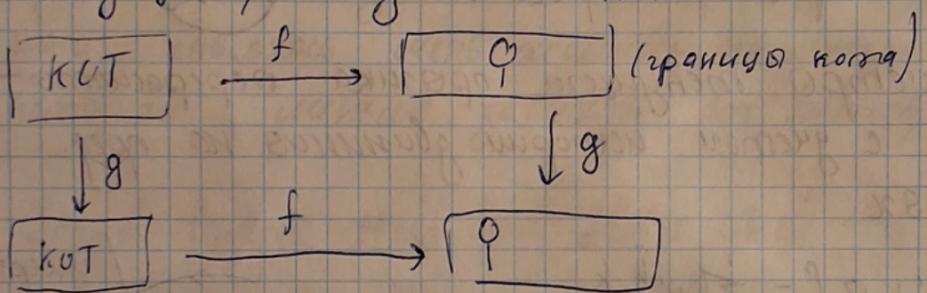
$f(g \circ x) = f(x)$ - инвариантность, относительно операции g . Но такого добиться изображено.

Если построим границу, затем сдвинуть картинку, и опять построим границу, то результаты не будут совпадать.

Эквивариантность.

$$f(g \circ x) = g \circ f(x)$$

Чтобы не, получать картинки этого:



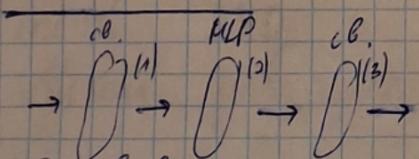
Все действия над объектами описываются группой.

Другими словами:

$$\forall g \in G, \forall x \in X. f(g \circ x) = g \circ f(x)$$

Ключевой факт:

Композиция эквивариантных функций также является эквивариантной функцией.



НЕ эквивариантна потому что в (3) сдвиг происходит относительно (1) , а там уже получили что-то однозначное. Но если нет сдвига относительно входа

Сверточная нейросеть — ~~один~~ способ получения продвинутых признаков и последующего применения MLP (линейн. регрессии)

В нейронных сетях фильтр изгibtается свёртками

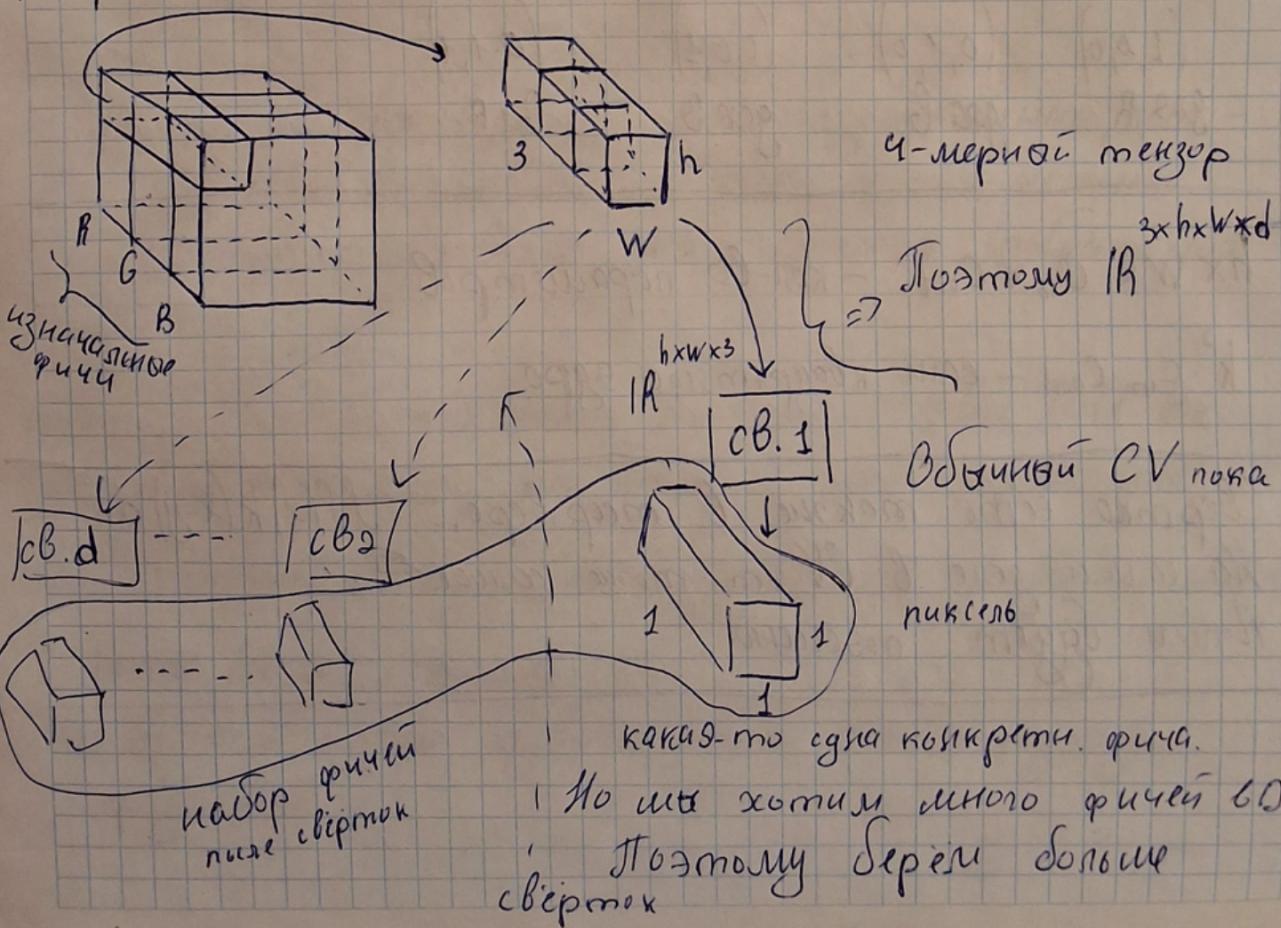
Операция свёртки многоканального изображения U с ядром K:

$$U(x, y, t) = \sum_{i=x-\delta}^{x+\delta} \sum_{j=y-\delta}^{y+\delta} \sum_{s=1}^S K(i-x+\delta, j-y+\delta, s, t) U(i, j, s)$$

$$U \in \mathbb{R}^{W \times H \times S}, \quad V \in \mathbb{R}^{W \times H \times T}, \quad K \in \mathbb{R}^{k \times k \times S \times T}, \quad k = 2\delta + 1$$

K - не матрица, а тензор!

Пример:

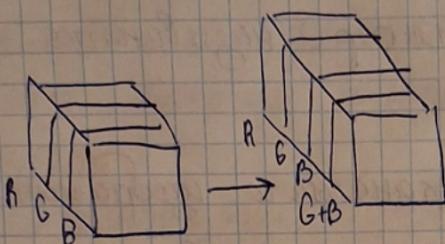


RGB → 4B

ПРИМЕР В СВЕРТОК

$$\begin{cases} h=1 \\ w=1 \\ in=3 \\ out=1 \end{cases}$$

если мы получим один пиксель, то есть перейти в 4B



$$h=1$$

$$w=1$$

$$in=3$$

$$out=1$$

$$\begin{array}{cccc} (1, 0, 0) & (0, 1, 0) & (0, 0, 1) & (0, 1, 1) \\ g_{R,R} & g_{R,G} & g_{R,B} & G+B \end{array}$$

$h \times w \times C_{in} \times C_{out}$ - кол-во параметров

$k^2 \cdot C_{in} \cdot C_{out}$ - если квадратное ядро

Свертка есть такое в теор. пере.: $\int f(x) k(x-y) dx$
На самом деле в CV это тоже такое. ↗
Но там фиги постепенно

Параметры:

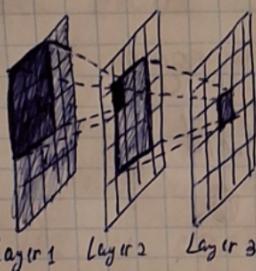
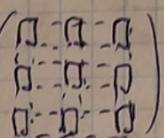
kernel_size: размер ядра

in_channel: число каналов in

out_channel: число каналов результата K

Есть другие параметры:

padding, stride, dilation



Каждый слоёкский слой имеет более широкое receptive поле

Receptive field - размер части изображения, которую вовлекает на значение нейрона слоёк изображения. Иными словами, зона покрытия изображения нейроном

Пример:

Если 3x3 имеют почти такой же receptive field как 1 сверточка 5x5, то при этом меньше параметров.

Увеличение слоёв позволяет уменьшить кол-во параметров!

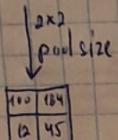
Почти всегда свёртки 3x3

Чем больше глубина, тем сильнее тухнет градиент.

Pooling Layer

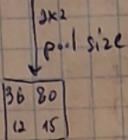
Max Pooling

29	15	28	18
0	100	90	38
12	12	7	2
12	12	45	6



Average Pooling

31	15	98	184
0	100	90	38
12	12	7	2
12	12	45	6



о обучаемых параметров

Обычно: CB, PL, что-то и так по кругу.

PL подбрасывает увеличить Receptive field, но не увеличивает обучаемые параметры.

Любой пулинг можно заменить для конкретной сверточки, со $\text{stride} = 2$

Свертки реализуются как матричное умножение

2. Winograd для 3х3

3. FFT, для больших размеров

В статье (irunv.net) наглядно показано как извлекаются признаки, которые вычленяют сверточные слои.

На поздних слоях видно уже что-то похожее на операции на определенные части изображения

То есть их выходы можно интерпретировать

Препр

1CB.

B
арх

(8,5%) Res/

Image
Разус
Болни

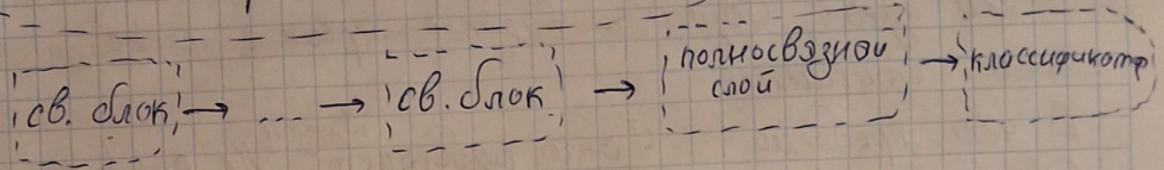
в. зап

кимор

Do

режко

Сверточная архитектура → Нелинейность → Пузыри
Сверточный блок



В ноутбуке пример первой нейросети и других архитектур.

²⁰¹⁵
ImageNet поднял Human Performance в задаче классификации изображений для лидера качества моделей.
Разумеется это не особо корректно, т.к. было слишком мало классов, которое тогда удивляло в голове и было много редких типов объектов, которые человек раньше не видел

До DL в 2011 был пероменс 25,8, затем резко ворас до 16.4%

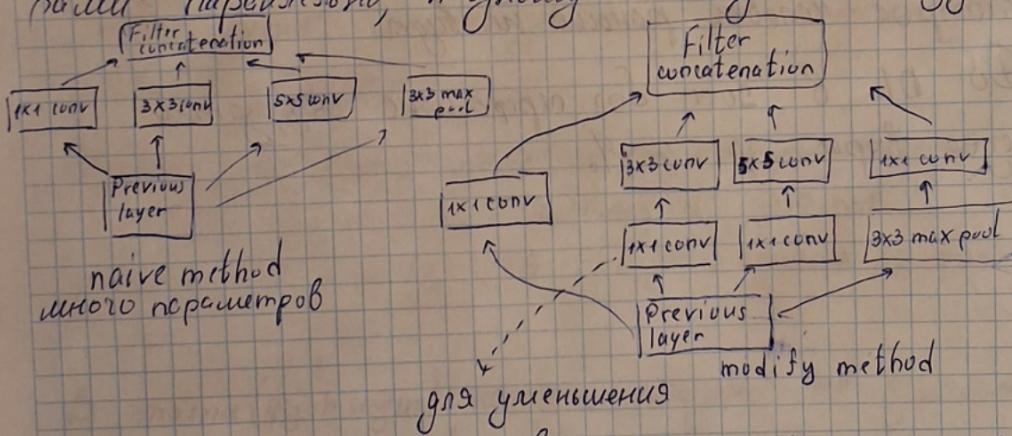
Была проблема, что гарантированно обучались 8 слоёв. Но в 2015 удалось обучить 16.

IGG → В чём идея? Обучить вначале первое 8 слоёв. Затем оставшиеся.

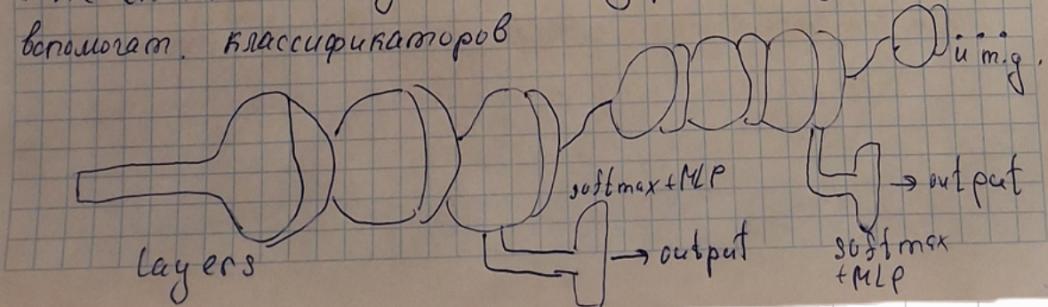
В чём проблема? Долго работает обучение (не end-to-end обучение (это что это)). Работает с из-меняющимися размерами, приходится например сжимать исходное end-to-endзначит обучать не всю нейронку сразу.

GoogLeNet (a.k.a. Inception, 2014)

- Идея применения свертоек с различными параметрами параллельно, к одному и тому же выходу.

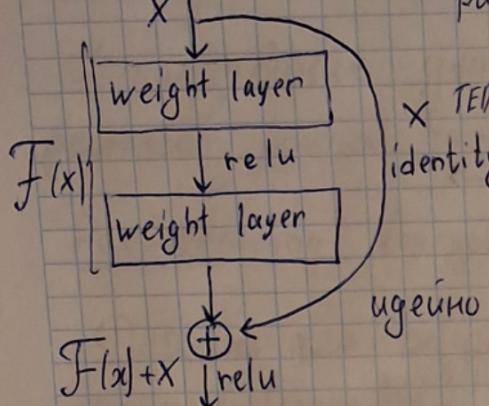


- Не end-to-end обучение благодаря использованию вспомогат. классификаторов



(skip) Residual connections

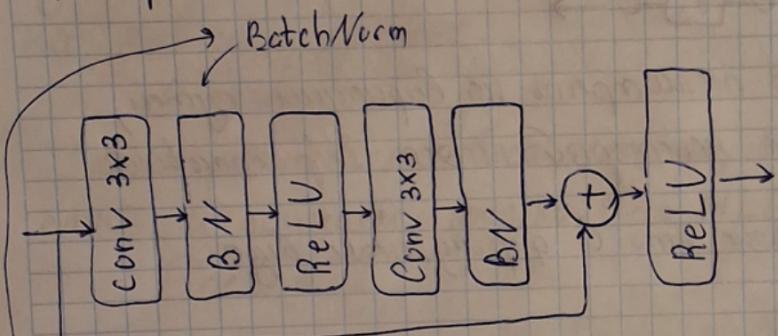
Решаем проблему затухания градиентов, без отказа от end-to-end обучения, т.к. из-за него обучение усложняется



$$\text{рассматривая } y = \tilde{F}(x) \Rightarrow \frac{d\ell}{dx} = \frac{d\ell}{dy} \cdot \frac{d\tilde{F}}{dx}$$

$$\times \text{ ТЕПЕРЬ: } y = \tilde{F}(x) + x \Rightarrow \frac{d\ell}{dx} = \frac{d\ell}{dy} \cdot \frac{d\tilde{F}}{dx} + \frac{d\ell}{dy}$$

Пример:



Но текущий метод максимум 100 слоёв, которых можно обучить — ~1000 слоёв

ResNet (2015) Top 1. Accuracy 98.31% Top 5 Accuracy 94.64%

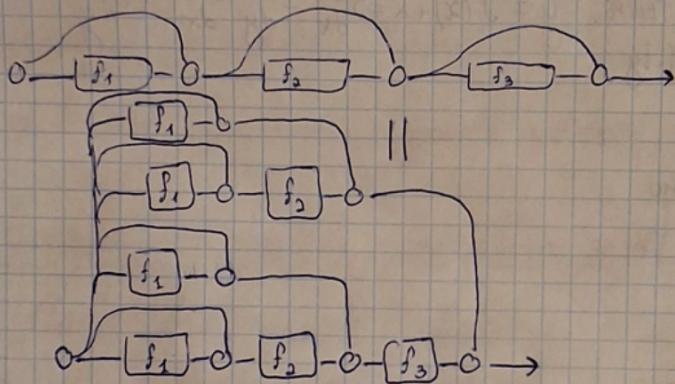
Нейросеть с использованием skip connection, 152 слоя

- BN для стабильного обучения

- Блеск pooling — выделки с stride=kernel_size (как обсуждалось ранее)

- Global average pooling на последнем слое, то есть на выходе вектор размерности равной числу каналов \Rightarrow обработка изображений

Каких-то исчерпывающих объяснений, почему residual connections работают и дают крутые результаты — нет. Но есть исследование говорящее некоторое объяснение и инициирующее понимание части вещей.



Если внимательно посмотреть на вариации путей, становится понятен некоторый вид skip connection

В группе есть такие это в формулированном виде.

avg pool усредняет один канал. То есть из одного канала достает общую информацию о факте. Это позволяет наводить изображения произвольного размера.

SOTA-система, которой виновата за лучшие показатели, качество. В 2015 например это был ResNet

Сама по себе задача ImageNet не особо практиче-
ски полезна. Но модели, решающие эту задачу,
способны хорошо видеть признаки и факты.

Transfer Learning

Использование уже предобученной модели при
решении другой задачи

Pre-training (предобучение)

Для начала обучаем модель на какой-то общей
задаче (р. ImageNet). Предполагаем, что модель научилась
извлекать признаки, которые хорошо обобщаются
на схожие данные

Linear probing

Замораживаем веса модели, меняем последнее
слой и обучаем только веса MLP.

Finetuning

Также заменяем голову, но обучаем уже всю
модель целиком

