

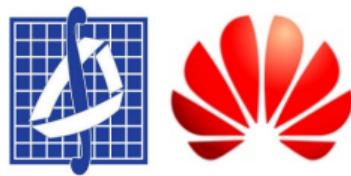
Введение в искусственный интеллект.

Современное компьютерное зрение

Тема: Основные классификационные СНС

Бабин Д.Н., Иванов И.Е., Петюшко А.А.

кафедра Математической Теории Интеллектуальных Систем



① Основные базы данных для обучения



План лекции

- ① Основные базы данных для обучения
- ② Об унитарном кодировании и классификационной функции потерь



План лекции

- ① Основные базы данных для обучения
- ② Об унитарном кодировании и классификационной функции потерь
- ③ AlexNet



План лекции

- ① Основные базы данных для обучения
- ② Об унитарном кодировании и классификационной функции потерь
- ③ AlexNet
- ④ VGG



План лекции

- ① Основные базы данных для обучения
- ② Об унитарном кодировании и классификационной функции потерь
- ③ AlexNet
- ④ VGG
- ⑤ Network-in-Network

План лекции

- ① Основные базы данных для обучения
- ② Об унитарном кодировании и классификационной функции потерь
- ③ AlexNet
- ④ VGG
- ⑤ Network-in-Network
- ⑥ Inception

План лекции

- ① Основные базы данных для обучения
- ② Об унитарном кодировании и классификационной функции потерь
- ③ AlexNet
- ④ VGG
- ⑤ Network-in-Network
- ⑥ Inception
- ⑦ ResNet

План лекции

- ① Основные базы данных для обучения
- ② Об унитарном кодировании и классификационной функции потерь
- ③ AlexNet
- ④ VGG
- ⑤ Network-in-Network
- ⑥ Inception
- ⑦ ResNet
- ⑧ DenseNet

- ① Основные базы данных для обучения
- ② Об унитарном кодировании и классификационной функции потерь
- ③ AlexNet
- ④ VGG
- ⑤ Network-in-Network
- ⑥ Inception
- ⑦ ResNet
- ⑧ DenseNet
- ⑨ Xception, ResNeXt и ShuffleNet

- ① Основные базы данных для обучения
- ② Об унитарном кодировании и классификационной функции потерь
- ③ AlexNet
- ④ VGG
- ⑤ Network-in-Network
- ⑥ Inception
- ⑦ ResNet
- ⑧ DenseNet
- ⑨ Xception, ResNeXt и ShuffleNet
- ⑩ (P)NasNet

О задаче классификации

Обучение

Процесс обучения — это процесс подстройки весов фильтров СНС для (суб)оптимального решения определенной задачи на основе предложенного набора данных.



О задаче классификации

Обучение

Процесс обучения — это процесс подстройки весов фильтров СНС для (суб)оптимального решения определенной задачи на основе предложенного набора данных.

Задача классификации

Это задача обучения с учителем (supervised learning), когда есть обучающая **размеченная** выборка данных $(x^i, y^i)_{i=1}^M$, где $x^i \in X$, а $y^i \in C = \{1, \dots, N\}$ — метка класса для экземпляра x^i .

При этом обучаемый классификатор F по входному экземпляру $x \in X$ должен выдавать корректную метку, т.е. $F : X \rightarrow C$.



О задаче классификации

Обучение

Процесс обучения — это процесс подстройки весов фильтров СНС для (суб)оптимального решения определенной задачи на основе предложенного набора данных.

Задача классификации

Это задача обучения с учителем (supervised learning), когда есть обучающая **размеченная** выборка данных $(x^i, y^i)_{i=1}^M$, где $x^i \in X$, а $y^i \in C = \{1, \dots, N\}$ — метка класса для экземпляра x^i .

При этом обучаемый классификатор F по входному экземпляру $x \in X$ должен выдавать корректную метку, т.е. $F : X \rightarrow C$.

Пример. Если выход СНС $Net(x) = p \in [0, 1]^N$, $\sum_{i=1}^N p_i = 1$ — это выход SoftMax-слоя, то $F(x) = \arg \max_i p_i$.

О задаче классификации

Обучение

Процесс обучения — это процесс подстройки весов фильтров СНС для (суб)оптимального решения определенной задачи на основе предложенного набора данных.

Задача классификации

Это задача обучения с учителем (supervised learning), когда есть обучающая **размеченная** выборка данных $(x^i, y^i)_{i=1}^M$, где $x^i \in X$, а $y^i \in C = \{1, \dots, N\}$ — метка класса для экземпляра x^i .

При этом обучаемый классификатор F по входному экземпляру $x \in X$ должен выдавать корректную метку, т.е. $F : X \rightarrow C$.

Пример. Если выход СНС $Net(x) = p \in [0, 1]^N$, $\sum_{i=1}^N p_i = 1$ — это выход SoftMax-слоя, то $F(x) = \arg \max_i p_i$.

Начнем с рассмотрения наиболее известных размеченных баз данных изображений для задачи классификации.

MNIST¹

- Modified National Institute of Standards and Technology
- База данных рукописных цифр 0–9, оттенки серого
- На данный момент лучшие классификаторы СНС дают порядка 99.9% распознавания на тесте

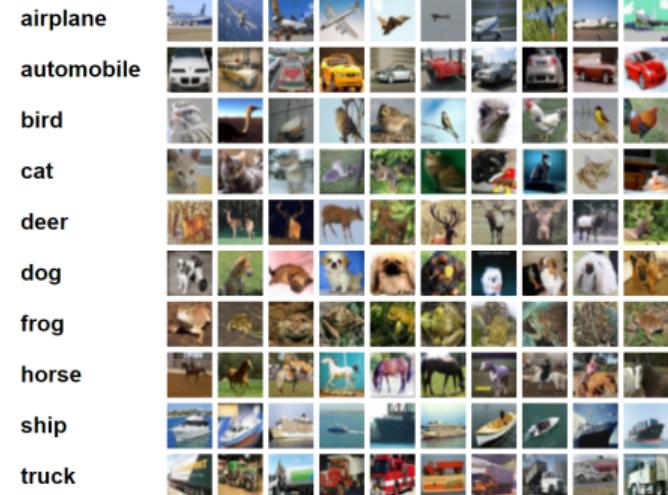


| База данных | train, объем | valid, объем | test, объем | Разрешение | Классы |
|-------------|--------------|--------------|-------------|------------|--------|
| MNIST | 60K | - | 10K | 28x28x1 | 10 |

¹<http://yann.lecun.com/exdb/mnist/>

CIFAR²

- Canadian Institute For Advanced Research
- База данных объектов из 10 (CIFAR-10) или 100 (CIFAR-100) классов на цветных изображениях
- На данный момент лучшие классификаторы СНС дают порядка 99.5% распознавания на тесте для CIFAR-10 и 96% для CIFAR-100

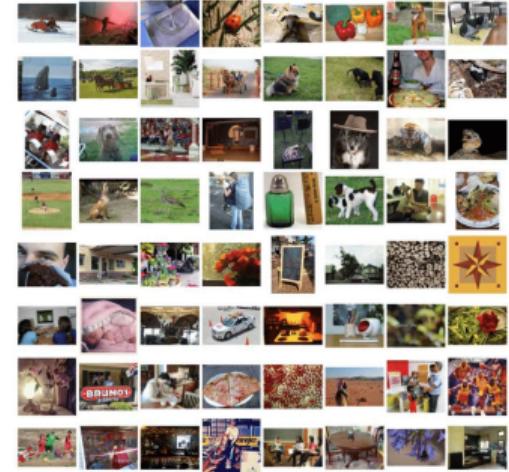


| База данных | train, объем | valid, объем | test, объем | Разрешение | Классы |
|-------------|--------------|--------------|-------------|------------|--------|
| CIFAR-10 | 50K | - | 10K | 32x32x3 | 10 |
| CIFAR-100 | 50K | - | 10K | 32x32x3 | 100 |

²<https://www.cs.toronto.edu/~kriz/cifar.html>

ImageNet-1K³

- Одна из самых сложных и больших баз данных для классификации изображений
- Изображения в основном цветные, разных размеров
- Из-за сложности (и порой неоднозначности меток) классификации вводятся две метрики качества:
 - top-1: правильный ответ имеет максимальную вероятность
 - top-5: правильный ответ содержится в 5 максимальных по вероятности
- На данный момент лучшие классификаторы СНС дают порядка 91% распознавания top-1 и 99% для top-5



| База данных | train, объем | valid, объем | test, объем | Разрешение | Классы |
|-------------|--------------|--------------|-------------|------------|--------|
| ImageNet-1K | 1200K | 50K | 100K | ≈400x350x3 | 1000 |

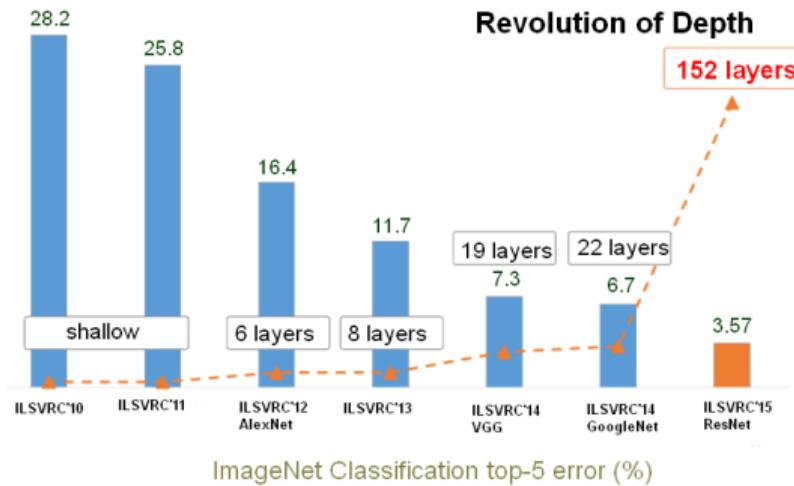
³<http://image-net.org/>

Сводная таблица по базам данных

| База данных | train, объем | valid, объем | test, объем | Разрешение | Классы |
|-------------|--------------|--------------|-------------|------------|--------|
| MNIST | 60K | - | 10K | 28x28x1 | 10 |
| CIFAR-10 | 50K | - | 10K | 32x32x3 | 10 |
| CIFAR-100 | 50K | - | 10K | 32x32x3 | 100 |
| ImageNet-1K | 1200K | 50K | 100K | ≈400x350x3 | 1000 |

Прогресс СНС на ImageNet

- Начиная с 2012 года, в конкурсе ILSVRC⁴, произошел прорыв благодаря СНС AlexNet
- Наблюдается вполне ожидаемая ситуация: с уменьшением ошибки распознавания увеличивается глубина СНС⁵



⁴ImageNet Large Scale Visual Recognition Competition

⁵http://image-net.org/challenges/talks/ilsvrc2015_deep_residual_learning_kaiminghe.pdf

Человеческий уровень на ImageNet

- Интересно сравнить человеческий уровень распознавания с лучшими СНС
- Сложность в:
 - ① Несбалансированных классах — вряд ли обычный человек знает разницу между средиземноморским и южновосточным зябликом :)
 - ② Наличии нескольких объектов на фотографии — и порой они из разных классов
- Andrej Karpathy⁶ замерил человеческий уровень ошибки top-5: **5.1%**

⁶<http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/>



- Интересно сравнить человеческий уровень распознавания с лучшими СНС
- Сложность в:
 - ① Несбалансированных классах — вряд ли обычный человек знает разницу между средиземноморским и южновосточным зябликом :)
 - ② Наличии нескольких объектов на фотографии — и порой они из разных классов
- Andrej Karpathy⁶ замерил человеческий уровень ошибки top-5: **5.1%**

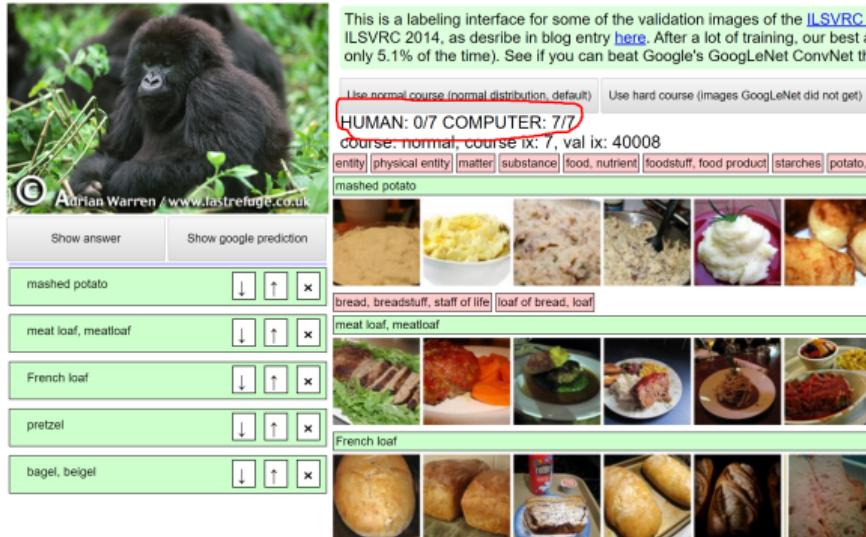
Вывод

Начиная с 2015 года, классификаторы-СНС превосходят человека в задаче классификации изображений.

⁶<http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/>

Разметка ImageNet

- Andrej Karpathy⁷ создал сайт, где можно посоревноваться с компьютером (в данном случае — с СНС GoogleNet) в области распознавания изображений
 - Для входной картинки нужно выбрать 5 наиболее подходящих классов из списка справа



⁷<https://cs.stanford.edu/people/karpathy/ilsvrc/>

Унитарное кодирование

- Пусть задано множество классов мощности N : $C = \{1, 2, \dots, N\}$
- Метка принадлежности к классу $y_c \in C$

Унитарное кодирование

- Пусть задано множество классов мощности N : $C = \{1, 2, \dots, N\}$
- Метка принадлежности к классу $y_c \in C$

Унитарное кодирование (one-hot encoding)

Для обучения нейросетей используется другая запись метки $y_{one-hot} \in \{0, 1\}^N$, причем если $y_{one-hot} = (y_1, \dots, y_N)$, то существует ровно одна единица на месте y_c :
 $y_{one-hot} = (0, \dots, 1, \dots, 0) : y_i = 0, i \neq y_c, y_i = 1, i = y_c.$



Унитарное кодирование

- Пусть задано множество классов мощности N : $C = \{1, 2, \dots, N\}$
- Метка принадлежности к классу $y_c \in C$

Унитарное кодирование (one-hot encoding)

Для обучения нейросетей используется другая запись метки $y_{one-hot} \in \{0, 1\}^N$, причем если $y_{one-hot} = (y_1, \dots, y_N)$, то существует ровно одна единица на месте y_c :
 $y_{one-hot} = (0, \dots, 1, \dots, 0) : y_i = 0, i \neq y_c, y_i = 1, i = y_c.$

Замечание. При этом вектор $y_{one-hot} = (y_1, \dots, y_N)$ является корректным вектором вероятностей:

$$\sum_{k=1}^N y_k = 1, \quad 0 \leq y_i \leq 1 \quad \forall i = 1 \dots N$$

Классификационная функция потерь

- Пусть задано множество классов мощности N : $C = \{1, 2, \dots, N\}$
- $(x^i, y^i), i = 1, \dots, M$ – обучающее множество
- Работаем только с унитарным кодированием меток y
- Выход SoftMax-слоя нейросети F : $p^i = F(x^i)$

Классификационная функция потерь

- Пусть задано множество классов мощности N : $C = \{1, 2, \dots, N\}$
- $(x^i, y^i), i = 1, \dots, M$ — обучающее множество
- Работаем только с унитарным кодированием меток y
- Выход SoftMax-слоя нейросети F : $p^i = F(x^i)$

Перекрестная энтропия (cross entropy)

Перекрестная энтропия для задания функции потерь при классификации:

$$H(y, p) = \sum_{i=1}^M (H(y^i) + D_{KL}(y^i || p^i)) = - \sum_{i=1}^M \sum_{j=1}^N y_j^i \log p_j^i$$

где $H(y)$ — энтропия y , $D_{KL}(y || p)$ — расстояние Кульбака-Лейблера между распределениями y и p .

Классификационная функция потерь

- Пусть задано множество классов мощности N : $C = \{1, 2, \dots, N\}$
- $(x^i, y^i), i = 1, \dots, M$ — обучающее множество
- Работаем только с унитарным кодированием меток y
- Выход SoftMax-слоя нейросети F : $p^i = F(x^i)$

Перекрестная энтропия (cross entropy)

Перекрестная энтропия для задания функции потерь при классификации:

$$H(y, p) = \sum_{i=1}^M (H(y^i) + D_{KL}(y^i || p^i)) = - \sum_{i=1}^M \sum_{j=1}^N y_j^i \log p_j^i$$

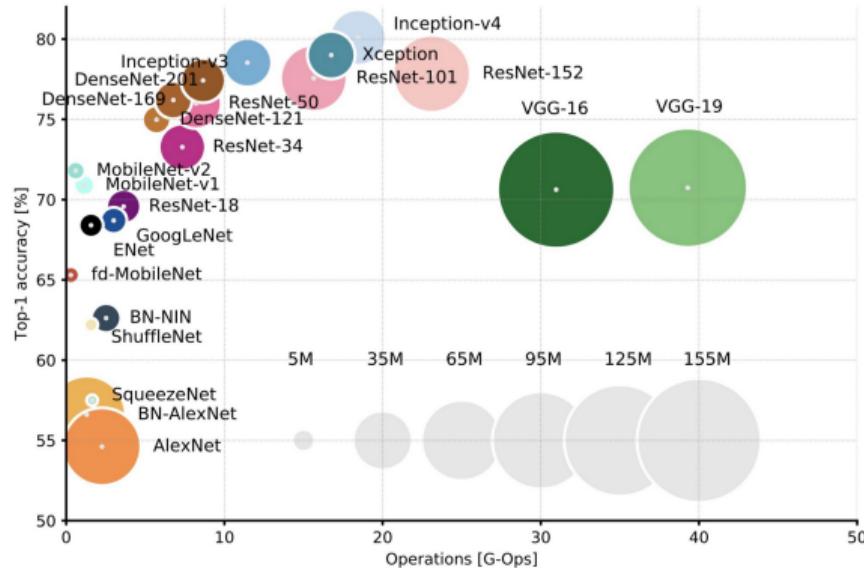
где $H(y)$ — энтропия y , $D_{KL}(y || p)$ — расстояние Кульбака-Лейблера между распределениями y и p .

Вопрос. Как будет выглядеть перекрестная энтропия для унитарного кодирования y ?



Основные характеристики СНС⁸

- Качество работы (процент распознавания) — ось Y
- Скорость работы (млрд операций) — ось X
- Размер (количество весов) — размер соотв. кружка



⁸<https://towardsdatascience.com/neural-network-architectures-156e5bad51ba>

Первая современная сверточная сеть

Первая современная сверточная сеть была
придумана в 1989 Яном Лекуном⁹, и
сочетает в себе все необходимые вещи,
используемые и по сей день:

- Свертка (из Неокогнитрона)
- Метод обратного распространения ошибки (от Хинтона и др.)
- Современная попытка воспроизвести работу от Karpathy¹⁰

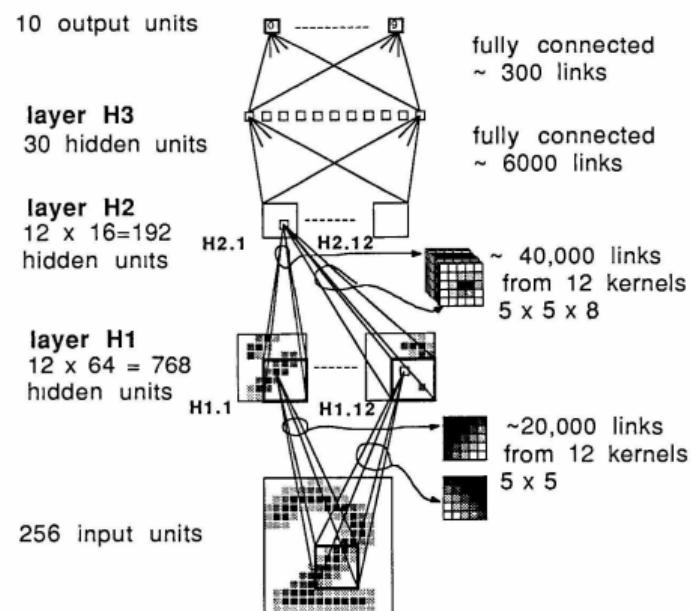
⁹Yann LeCun et al. Backpropagation Applied to Handwritten Zip Code Recognition, 1989

¹⁰<http://karpathy.github.io/2022/03/14/lecun1989/>

Первая современная сверточная сеть

Первая современная сверточная сеть была придумана в 1989 Яном Лекуном⁹, и сочетает в себе все необходимые вещи, используемые и по сей день:

- Свертка (из Неокогнитрона)
- Метод обратного распространения ошибки (от Хинтона и др.)
- Современная попытка воспроизвести работу от Karpathy¹⁰



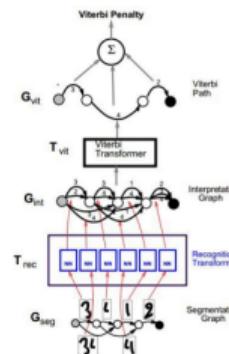
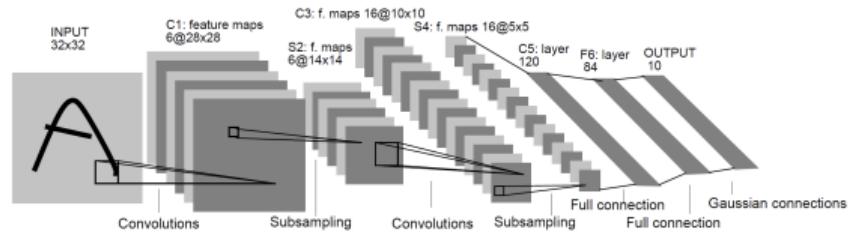
⁹Yann LeCun et al. Backpropagation Applied to Handwritten Zip Code Recognition, 1989

¹⁰<http://karpathy.github.io/2022/03/14/lecun1989/>

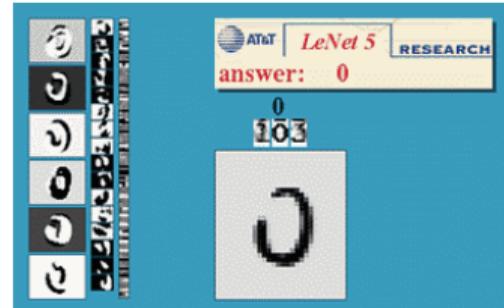
Первая известная сверточная сеть

Первой известной сверточной сетью стала т.н. LeNet-5¹¹ от того же Лекуна. Изменения по сравнению с версией 1989 года:

- Появились два слоя субдискретизации
- Постобработка на выходе (алгоритм Витерби и т.п.)



Демо работы системы¹²



¹¹Yann LeCun et al. Gradient-based learning applied to document recognition, 1998

¹²<http://yann.lecun.com/exdb/lenet/>

- Именно AlexNet прославил СНС и положил начало их повсеместному использованию в компьютерном зрении
- По сравнению с сетью LeNet, привнес множество новых трюков

Особенности AlexNet

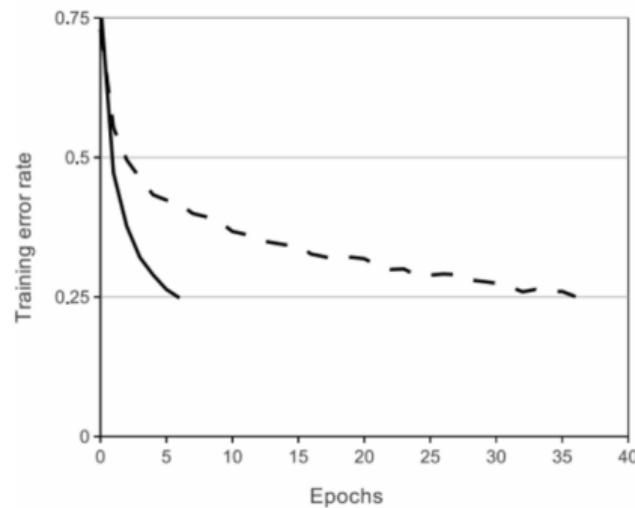
- Использование ReLU вместо sigmoid
- Локальная нормализация активаций
- Субдискретизация с перекрытием
- Аугментация обучающей выборки
- Дропаут (выброс) для уменьшения переобучения
- Аугментация на teste (TTA)

¹³Krizhevsky A., Sutskever I., Hinton G. E. Imagenet classification with deep convolutional neural networks
2012

AlexNet и ReLU

Оказывается, если отказаться от классических на тот момент (2012) функций активации (sigmoid, tanh), то можно ускорить обучение СНС на порядок.

К примеру, СНС глубины 4 с функцией активации ReLU сходится в 6 раз быстрее на CIFAR-10 по сравнению с СНС той же архитектуры, но с функцией активации tanh.



AlexNet — локальная нормализация активаций

- Будем нормализовывать карты активации
- Пусть a_{xy}^i, b_{xy}^i — активация и нормализованная активация в точке (x, y) для карты i
- Всего карт признаков N
- Нормализация:

$$b_{xy}^i = \frac{a_{xy}^i}{\left(k + \alpha \sum_{j=\max(0, i-n/2)}^{\min(N-1, i+n/2)} (a_{xy}^j)^2 \right)^\beta}$$

- где $k = 2, n = 5, \alpha = 10^{-4}, \beta = 0.75$

Эта нормализация уменьшила ошибки: top-1 на 1.4%, top-5 на 1.2%.

AlexNet — локальная нормализация активаций

- Будем нормализовывать карты активации
- Пусть a_{xy}^i, b_{xy}^i — активация и нормализованная активация в точке (x, y) для карты i
- Всего карт признаков N
- Нормализация:

$$b_{xy}^i = \frac{a_{xy}^i}{\left(k + \alpha \sum_{j=\max(0, i-n/2)}^{\min(N-1, i+n/2)} (a_{xy}^j)^2 \right)^\beta}$$

- где $k = 2, n = 5, \alpha = 10^{-4}, \beta = 0.75$

Эта нормализация уменьшила ошибки: top-1 на 1.4%, top-5 на 1.2%.

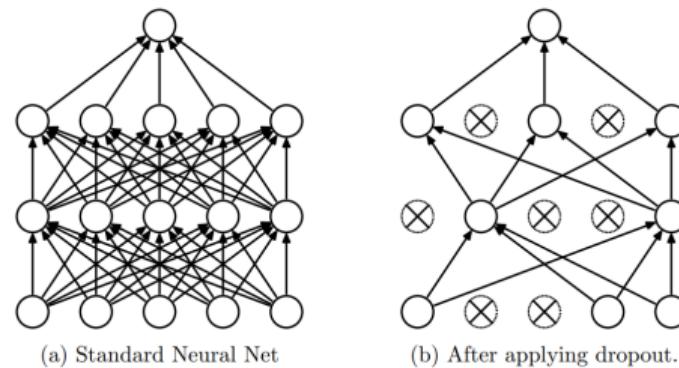
Замечание. Это — вариант нормализации по части слоя (**group normalization**) из-за распараллеливания на 2 карты (см. далее).

AlexNet — субдискретизация с перекрытием

- Обычно субдискретизация (пулинг) используется с окном размера $z = n$ со сдвигом, равным размеру окна $s = n$
- В AlexNet применяется пулинг с перекрытием в 1 клетку: $z = n, s = n - 1$

При $n = 3$ пулинг с перекрытием уменьшает ошибки: top-1 на 0.4%, top-5 на 0.3%.

AlexNet — дропаут

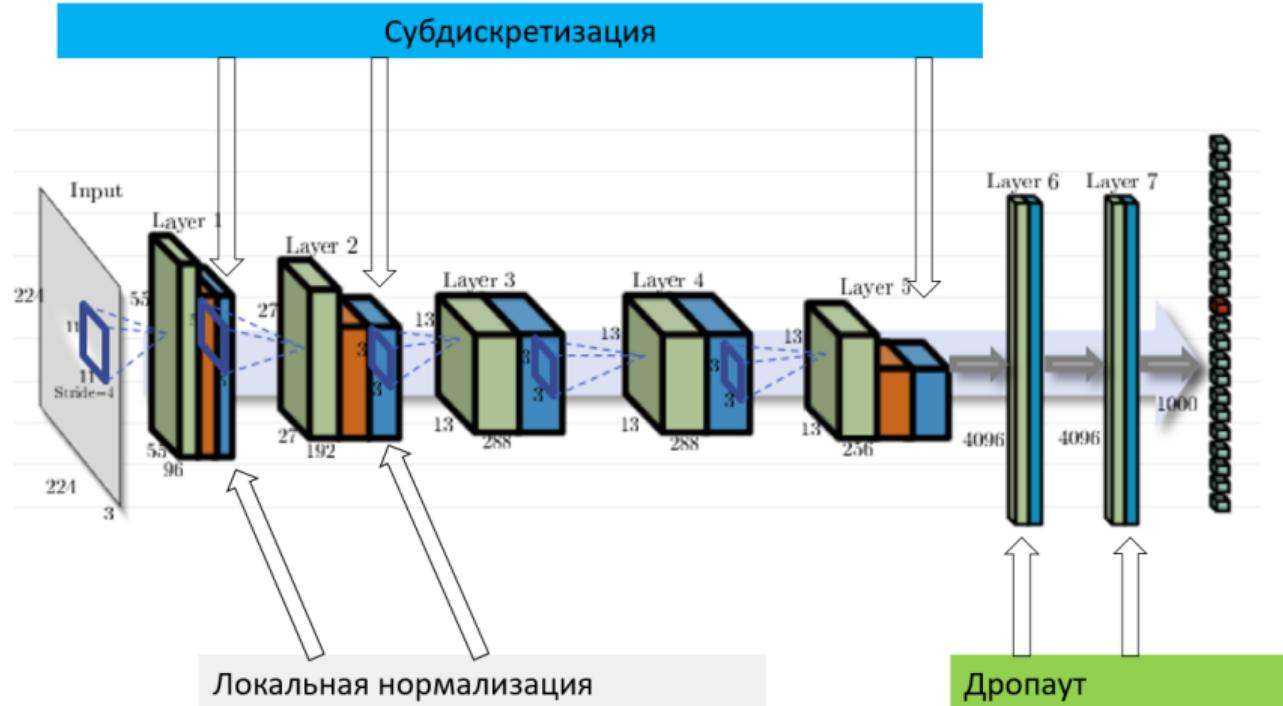


В AlexNet использовалось значение $p = 0.5$ на полносвязных слоях, на сверточных дропаут обычно не используется.

Дропаут помог с переобучением, однако время обучения возросло в $1/p = 2$ раз.



Итоговая архитектура AlexNet



Инициализация

- Веса сверточ задаются случайными числами из нормального распределения с нулевым средним и с.к.о. 0.01
- Коэффициент сдвига в некоторых слоях инициализируется нулем, в некоторых (например, полносвязных) - единицей

Инициализация

- Веса сверточ задаются случайными числами из нормального распределения с нулевым средним и с.к.о. 0.01
- Коэффициент сдвига в некоторых слоях инициализируется нулем, в некоторых (например, полносвязных) - единицей

Оптимизатор

- Динамическое управление коэффициентом скорости обучения: коэффициент делится на 10, когда качество перестает улучшаться на валидационной выборке
- Оптимизатор - momentum (накопление градиента)
- L_2 -регуляризация на веса (которое также называется “weight decay”)



Предобработка

- Изображение сначала интерполируется так, чтобы меньшая сторона стала 256 пикселей, после чего вырезается центральный квадрат 256×256
- Размер входного изображения для СНС — 224×224 (об этом позже)
- Вычитаем средний цвет RGB ([123.68, 116.779, 103.939])



Предобработка

- Изображение сначала интерполируется так, чтобы меньшая сторона стала 256 пикселей, после чего вырезается центральный квадрат 256×256
- Размер входного изображения для СНС — 224×224 (об этом позже)
- Вычитаем средний цвет RGB ([123.68, 116.779, 103.939])

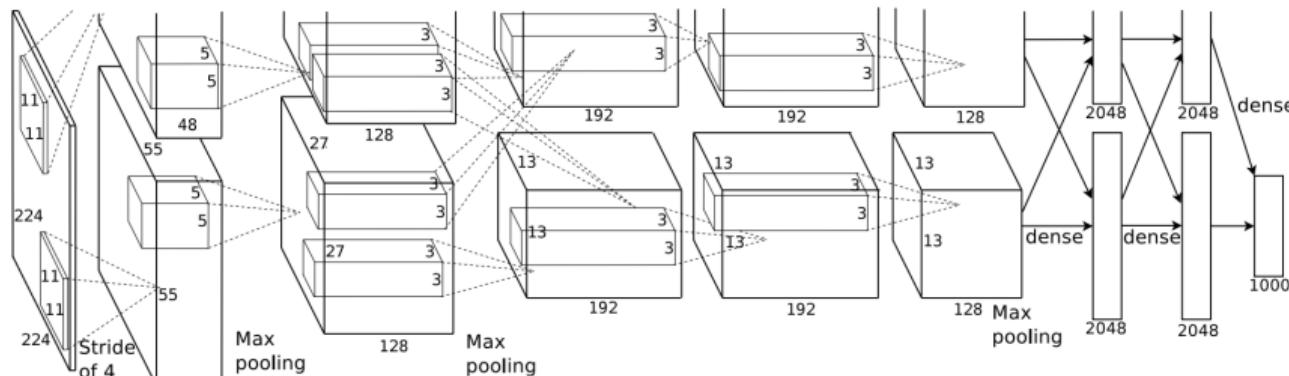
Аугментация

- Случайный вырез (кроп, англ. “crop”) размера 224×224 из 256×256
- Горизонтальное отражение
- Изменение интенсивности RGB цветов

- Для AlexNet начали применять т.н. *n-crop*, где после подачи на вход СНС n изображений получается n k -мерных (в случае ImageNet $k = 1000$) векторов вероятностей: $p^1 = (p_1^1, \dots, p_k^1), \dots, p^n = (p_1^n, \dots, p_k^n)$
- Выходной вектор вероятности вычисляется как среднее: $p = \frac{1}{n} \sum_{i=1}^n p^i$
- 10 кропов из картинки 256×256 получаются так: берется центральный кроп и 4 угловых, а также их горизонтальные зеркальные отражения — $(1 + 4) * 2 = 10$.

AlexNet — Распараллеливание по GPU

- Вся СНС не помещалась на одной карте GTX 580 с 3 GB памяти \Rightarrow использовались две такие карты
- Было использовано **распараллеливание по модели**: сверточные фильтры делились в каждом слое на две равные группы (см. **групповые свертки**), и эти группы обсчитывались независимо на разных картах, иногда синхронизируя результаты (в сверточном слое 3, а также во всех полно связных слоях)
- Выигрыш от использования двух карт вместо одной: ошибки top-1 и top-5 снизились на 1.7% и 1.2% соответственно.

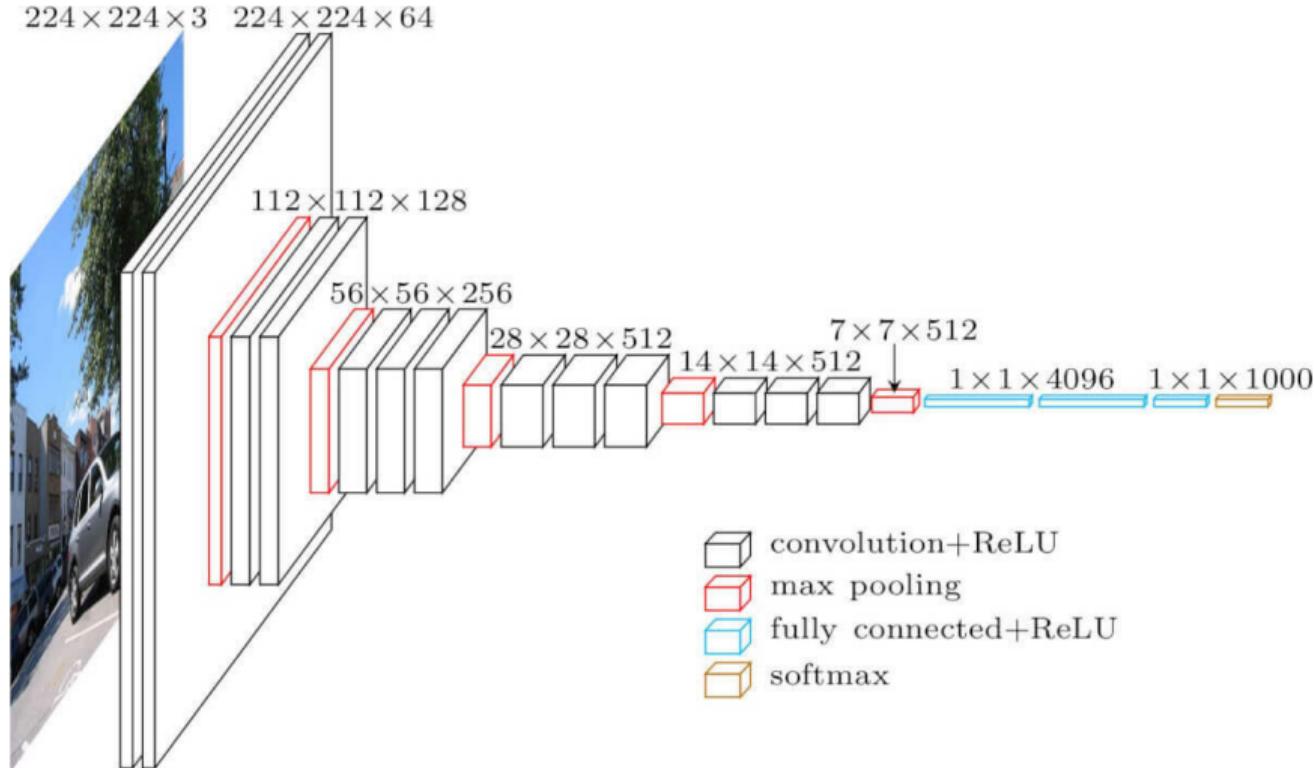


- Предложена в **Visual Geometry Group** (Университет Оксфорда)
- Идея простой и глубокой сверточной сети доведена до максимума
 - Существуют 2 версии: VGG-16 и VGG-19 (по числу слоев с обучаемыми весами)
 - Содержит огромное число параметров (138 млн и 144 млн), дальнейшее увеличение глубины и параметров не дало прироста на ImageNet
- Используются только стандартные свертки размера 3×3 (к примеру, в AlexNet первые две свертки были размеров 11×11 и 5×5)
- Субдискретизация без перекрытия
- Входной размер тот же — 224×224
- 150 кропов (равномерная решетка 5×5 , горизонтальное зеркальное отражение $\times 2$, и все это на $\times 3$ масштабах 256, 384, 512)

¹⁴Simonyan K., Zisserman A. Very deep convolutional networks for large-scale image recognition. 2014.



Архитектура VGG-16



Network-in-Network — важные находки¹⁶

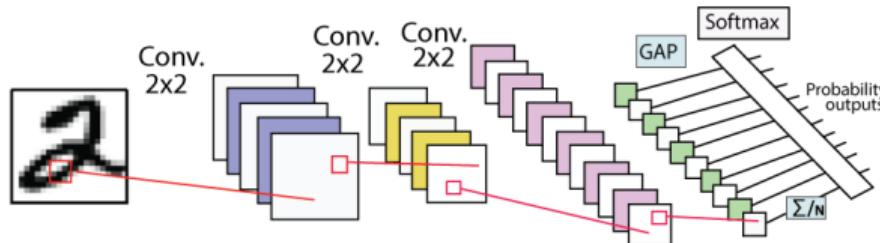
- Вопрос 1: Можно ли вместо обычной свертки использовать более общую модель?
- Вопрос 2: Можно ли адаптировать классификационную модель к любому размеру входной картинки?

¹⁵<https://principlesofdeeplearning.com/index.php/a-tutorial-on-global-average-pooling/> 

¹⁶Lin M., Chen Q., Yan S. Network in network. 2013.

Network-in-Network — важные находки¹⁶

- Вопрос 1: Можно ли вместо обычной свертки использовать более общую модель?
- Вопрос 2: Можно ли адаптировать классификационную модель к любому размеру входной картинки?
- Идея 1: Использовать многослойный перцептрон для того же рецептивного поля
- Идея 2: Использовать глобальное усреднение (global average pooling¹⁵)



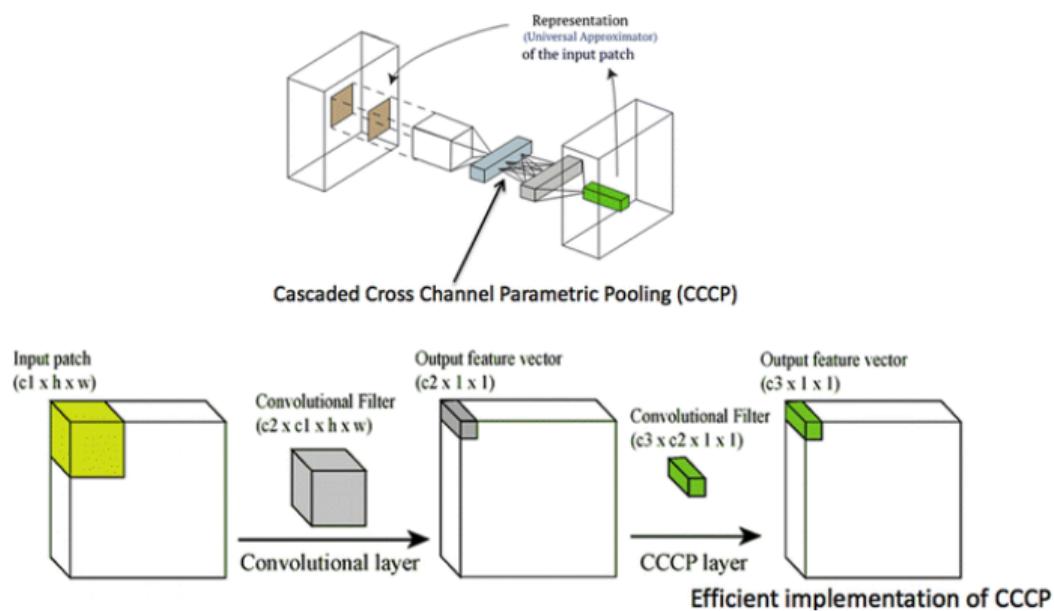
¹⁵<https://principlesofdeeplearning.com/index.php/a-tutorial-on-global-average-pooling/>

¹⁶Lin M., Chen Q., Yan S. Network in network. 2013.

Network-in-Network — Mlpconv

Если подумать, то MLP над рецептивным полем можно реализовать как каскад:

$CONV_{h \times w} \rightarrow CONV_{1 \times 1} \rightarrow CONV_{1 \times 1} \rightarrow \dots \rightarrow CONV_{1 \times 1}$ — это и называется Mlpconv¹⁷



¹⁷ King, S. Feature Learning and Deep Learning Architecture Survey. 2016

Network-in-Network — GAP

- **Полносвязный слой.** Пусть после последней свертки имеем тензор $d \times h \times w$. В простейшем случае мы его “выпрямляем” (flatten) в вектор длины dhw , после чего с помощью матрицы размера $N \times dhw$ преобразуем в вектор логитов длины N , где N — число классов. Число параметров — $Ndhw$

- **Полносвязный слой.** Пусть после последней свертки имеем тензор $d \times h \times w$. В простейшем случае мы его “выпрямляем” (flatten) в вектор длины dhw , после чего с помощью матрицы размера $N \times dhw$ преобразуем в вектор логитов длины N , где N — число классов. Число параметров — $Ndhw$
- **Глобальное усреднение.** С помощью обычного усреднения преобразуем карты признаков в скалярные значения, после чего (если $d \neq N$) делаем еще одну 1×1 свертку из d средних в N логитов. Число параметров (в худшем случае) — Nd

Вывод

В полносвязном случае мы должны иметь фиксированный размер входа, чтобы иметь фиксированный размер последней матрицы ($N \times dhw$), в то время как для глобального усреднения нам входной размер неважен — все равно потом “схлопнем” всю карту признаков в 1 значение

Разреженная архитектура

- **Проблема:** линейное наращивание сверточных слоев достаточно быстро себя исчерпывает (вспомните VGG)
- **Идея:** использование нелинейной разреженной архитектуры

¹⁸Szegedy C. et al. Going Deeper with Convolutions. 2014.

Разреженная архитектура

- **Проблема:** линейное наращивание сверточных слоев достаточно быстро себя исчерпывает (вспомните VGG)
- **Идея:** использование нелинейной разреженной архитектуры

Конкатенация фильтров

- **Проблема:** при линейной структуре у нас признаки с рецептивного поля одного размера (ограниченного размером свертки)
- **Идея:** конкатенировать выходы сверток разного размера на слоях одной глубины

¹⁸Szegedy C. et al. Going Deeper with Convolutions. 2014.

Разреженная архитектура

- **Проблема:** линейное наращивание сверточных слоев достаточно быстро себя исчерпывает (вспомните VGG)
- **Идея:** использование нелинейной разреженной архитектуры

Конкатенация фильтров

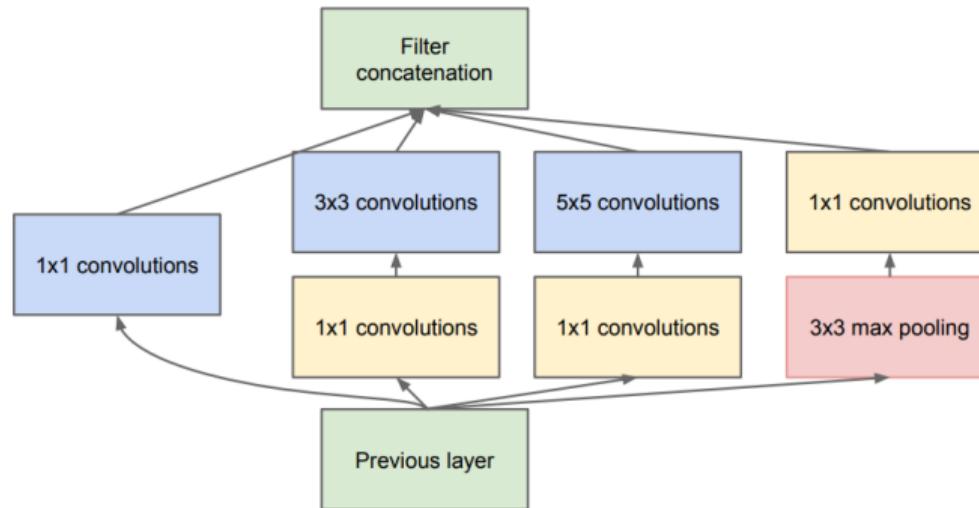
- **Проблема:** при линейной структуре у нас признаки с рецептивного поля одного размера (ограниченного размером свертки)
- **Идея:** конкатенировать выходы сверток разного размера на слоях одной глубины

Уменьшение сложности

- **Проблема:** при большом количестве карт много вычислений свертки
- **Идея:** с помощью свертки 1×1 предварительно уменьшить количество карт

¹⁸Szegedy C. et al. Going Deeper with Convolutions. 2014.

Inception модуль



Все свертки и субдискретизации производятся с шагом 1.

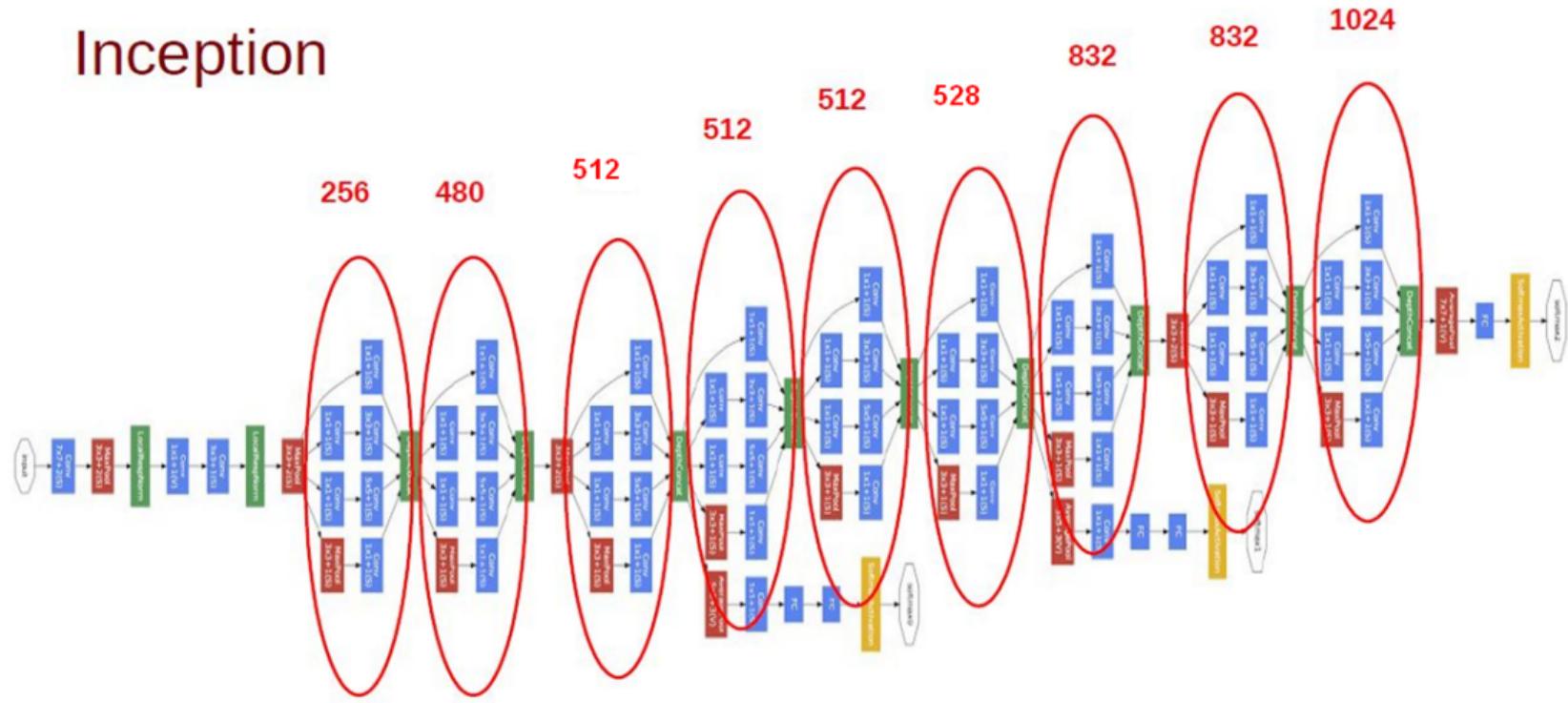
Архитектурные особенности

- Количество составных блоков около 100
- Глубина — 27 слоев, из них 22 — с обучаемыми параметрами (сравните с VGG)
- Перед последним полносвязным слоем — GAP (+0.6%)
- Входной размер — $224 \times 224 \times 3$ (хотя благодаря GAP можно сделать произвольно больше)
- Введены два дополнительных классификатора в середине сети (борьба с затухающим градиентом, регуляризация)
- Функции потерь для доп. классификаторов домножаются на 0.3 (+0.5%)
- Все еще используется дропаут



Inception V1 также известна под именем GoogleNet.

Inception



Inception, параметры

| type | patch size/ stride | output size | depth | #1×1 | #3×3 reduce | #3×3 | #5×5 reduce | #5×5 | pool proj | params | ops |
|----------------|-----------------------|----------------|-------|------|----------------|------|----------------|------|--------------|--------|------|
| convolution | 7×7/2 | 112×112×64 | 1 | | | | | | | 2.7K | 34M |
| max pool | 3×3/2 | 56×56×64 | 0 | | | | | | | | |
| convolution | 3×3/1 | 56×56×192 | 2 | | 64 | 192 | | | | 112K | 360M |
| max pool | 3×3/2 | 28×28×192 | 0 | | | | | | | | |
| inception (3a) | | 28×28×256 | 2 | 64 | 96 | 128 | 16 | 32 | 32 | 159K | 128M |
| inception (3b) | | 28×28×480 | 2 | 128 | 128 | 192 | 32 | 96 | 64 | 380K | 304M |
| max pool | 3×3/2 | 14×14×480 | 0 | | | | | | | | |
| inception (4a) | | 14×14×512 | 2 | 192 | 96 | 208 | 16 | 48 | 64 | 364K | 73M |
| inception (4b) | | 14×14×512 | 2 | 160 | 112 | 224 | 24 | 64 | 64 | 437K | 88M |
| inception (4c) | | 14×14×512 | 2 | 128 | 128 | 256 | 24 | 64 | 64 | 463K | 100M |
| inception (4d) | | 14×14×528 | 2 | 112 | 144 | 288 | 32 | 64 | 64 | 580K | 119M |
| inception (4e) | | 14×14×832 | 2 | 256 | 160 | 320 | 32 | 128 | 128 | 840K | 170M |
| max pool | 3×3/2 | 7×7×832 | 0 | | | | | | | | |
| inception (5a) | | 7×7×832 | 2 | 256 | 160 | 320 | 32 | 128 | 128 | 1072K | 54M |
| inception (5b) | | 7×7×1024 | 2 | 384 | 192 | 384 | 48 | 128 | 128 | 1388K | 71M |
| avg pool | 7×7/1 | 1×1×1024 | 0 | | | | | | | | |
| dropout (40%) | | 1×1×1024 | 0 | | | | | | | | |
| linear | | 1×1×1000 | 1 | | | | | | | 1000K | 1M |
| softmax | | 1×1×1000 | 0 | | | | | | | | |

- Вычислений больше в начальных слоях, а параметров — в конечных!
- Всего параметров чуть меньше 7 млн

Inception — детали обучения и тестирования

- Процедура обучения — SGD с momentum
- Скорость обучения уменьшается на 4% каждые 8 эпох
- Аугментация по размеру и пропорциям картинок в обучении
- 144 кропа в teste: 4 масштаба (256, 288, 320 и 352), берем 3 квадрата (крайние и центральный с длиной меньшей стороны), в каждом квадрате берем 4 угла и центральный подквадрат размера 224×224 + уменьшенный квадрат до 224×224 , ну и плюс все их горизонтально отраженные варианты. Итого: $4 \times 3 \times 6 \times 2 (+2.18\%)$
- Для уменьшения дисперсии использовался ансамбль из 7 идентичных по архитектуре моделей, разница в которых была только в порядке подачи картинок обучающей базы ($+1.27\%$ дополнительно; итого делается $7 \times 144 = 1008$ запусков)

Определение

Эпоха — это время, в течение которого на вход обучаемой НС будет подана вся обучающая база

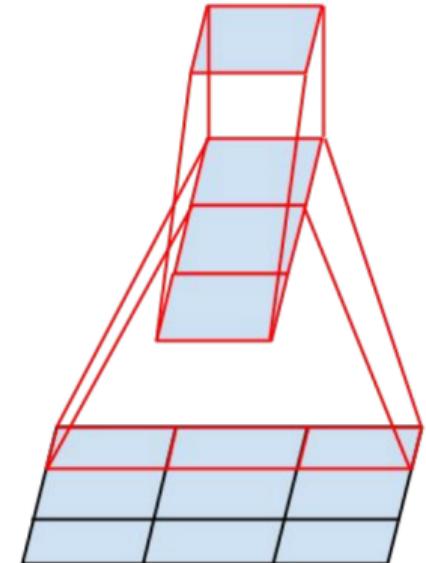
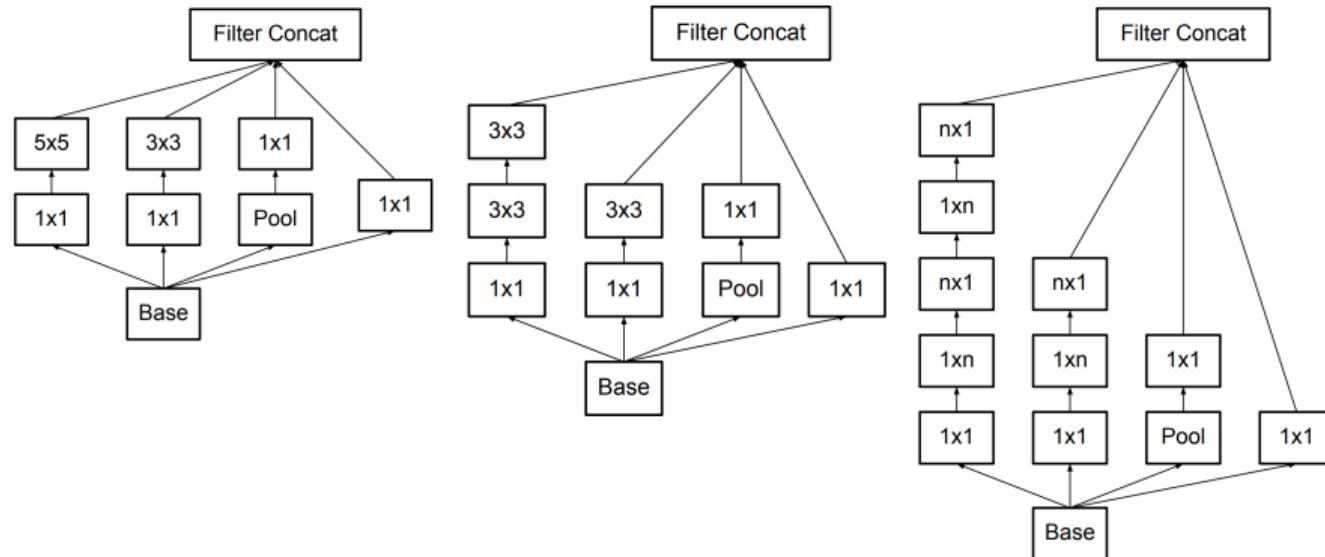
Inception V2 и V3 — новые идеи²⁰

- Факторизация двумерных сверток $n \times n$ на последовательные “одномерные” $n \times 1$ и $1 \times n$ (выигрыш по параметрам: вместо n^2 параметров используем $2n$)
- Повышение входного размера изображения с 224×224 до 299×299 (наиболее распространенный сейчас вариант для классификатора)
- Оптимизатор RMSProp вместо простого SGD
- Использование технологии обучения под названием “сглаживание меток” (label smoothing)
 - Обычная практика — т.е. one-hot encoding — когдациальному (ground truth) классу присваивается метка (вероятность) “1”, а остальным $K - 1$ классам — “0”
 - При сглаживаниициальному классу дается метка $1 - \frac{K-1}{K}\epsilon$, а остальным $K - 1$ классам $\frac{1}{K}\epsilon$, где $0 < \epsilon \ll 1$
- Добавление BatchNorm¹⁹ позволило еще немного улучшить показатели (и назвать модель Inception V3)

¹⁹Ioffe S., Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. 2015.

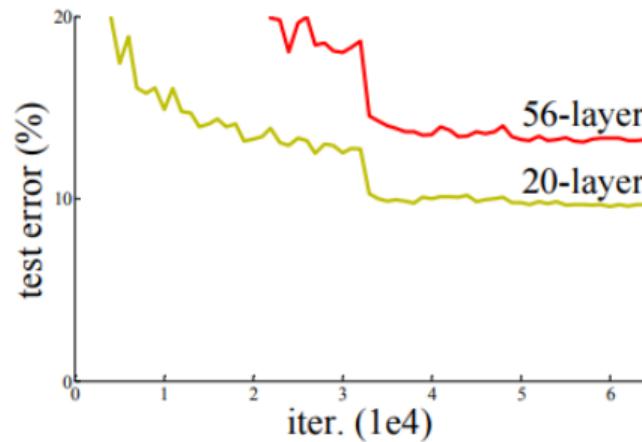
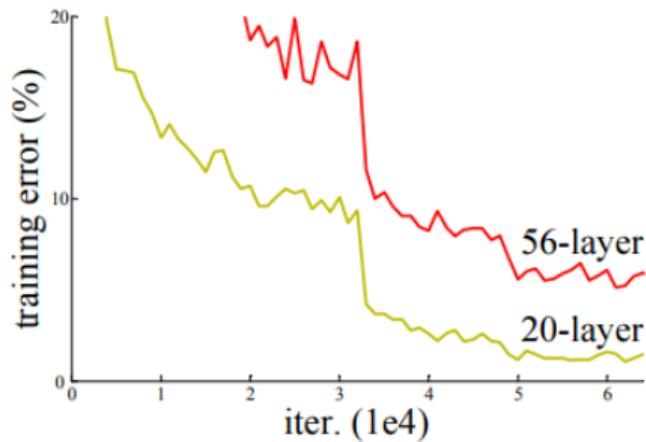
²⁰Szegedy C. et al. Rethinking the inception architecture for computer vision. 2016.

Inception V2 и V3 – схема inception-блока



Проблема

При увеличении “плоских” слоев в стиле VGG качество работы сети падает, а не растет (даже на обучающей выборке, не говоря уже о тестовой!)



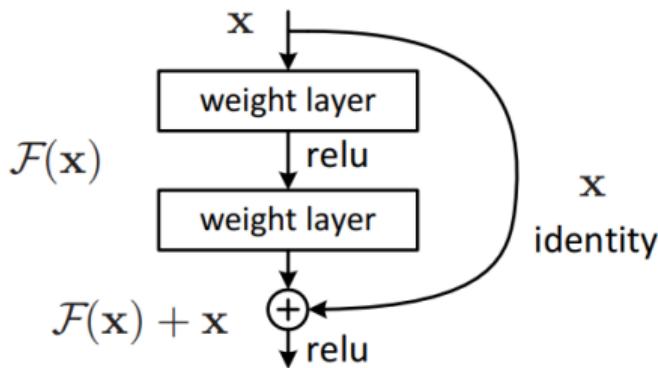
²¹ He K. et al. Deep residual learning for image recognition. 2016.

ResNet — новая архитектура

Решение

Будем вместо целевой функции $H(x)$ обучать остаточную (**residual** \Rightarrow ResNet) функцию $F(x) = H(x) - x$, а целевую рассчитывать как поточечную сумму входа x и остаточной функции $F(x)$: $H(x) = F(x) + x$.

При этом, если размерности остаточной функции и входа различны, нужно использовать линейную проекцию W_s : $H(x) = F(x) + W_s x$.



Таким образом, мы дополнительно прокидываем тождественные связи (*identity*) между входом и выходом остаточной функции.
Эти тождественные связи также носят название “skip-connections” или “short-cuts”.

ResNet – о Res-блоках

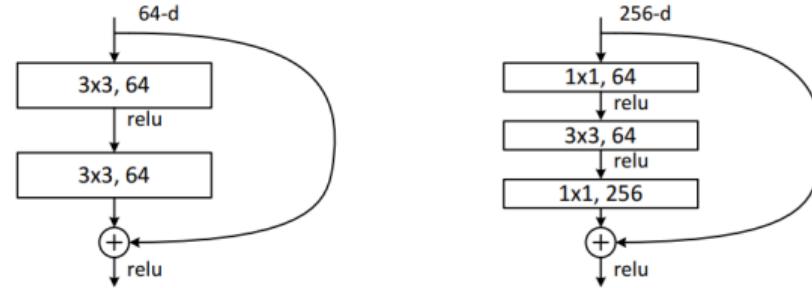
- Наиболее известны следующие разновидности ResNet (по числу сверточных слоев): Resnet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-152
- Соответственно, для более глубоких сетей используется большее число карт признаков \Rightarrow многократно возрастает количество весов фильтров и число операций
- Авторы предлагают использовать дизайн бутылочного горлышка (**bottleneck**): сначала уменьшить количество карт с помощью сверток 1×1 , затем обычная свертка 3×3 , и наконец возвращаем количество карт свертками 1×1

ResNet — о Res-блоках

- Наиболее известны следующие разновидности ResNet (по числу сверточных слоев): Resnet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-152
- Соответственно, для более глубоких сетей используется большее число карт признаков \Rightarrow многократно возрастает количество весов фильтров и число операций
- Авторы предлагают использовать дизайн бутылочного горлышка (**bottleneck**): сначала уменьшить количество карт с помощью сверток 1×1 , затем обычная свертка 3×3 , и наконец возвращаем количество карт свертками 1×1

Слева — дизайн Res-блока для ResNet-18/34.

Справа — дизайн Res-блока для ResNet-50/101/152.



ResNet — сравнение глубины

AlexNet, 8 layers
(ILSVRC 2012)



VGG, 19 layers
(ILSVRC 2014)

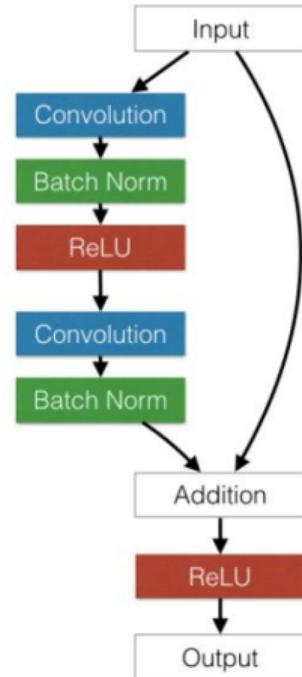


ResNet, **152 layers**
(ILSVRC 2015)



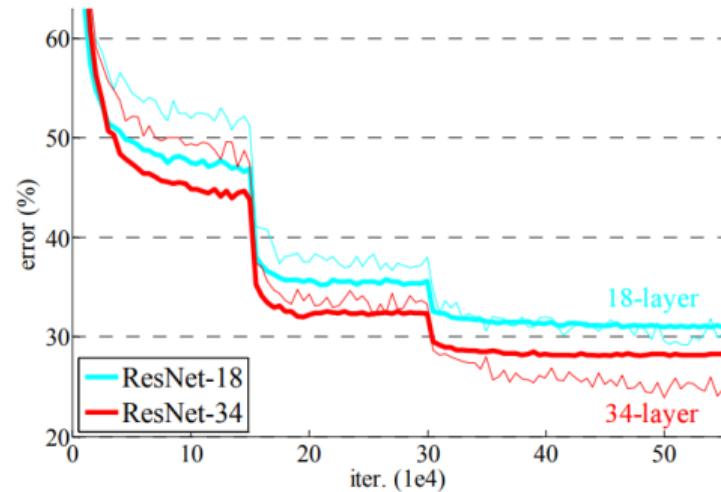
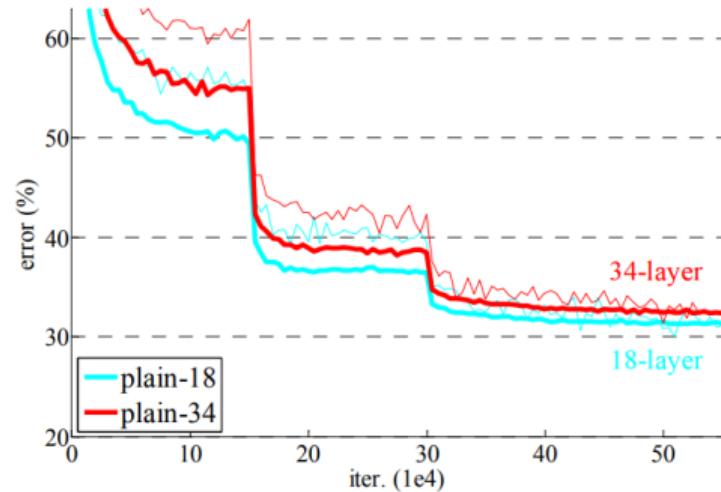
Архитектурные особенности

- Входной размер — $224 \times 224 \times 3$
- Перед последним полносвязным слоем — GAP
- Принцип сохранения сложности (уменьшаем пространственный размер в 2 раза \Rightarrow одновременно увеличиваем в 2 раза количество карт признаков)
- Почти везде уменьшение размерности через свертки с шагом 2 (и соответственно, использование проекций для тождественных связей W_s)
- BatchNorm используется после каждой свертки



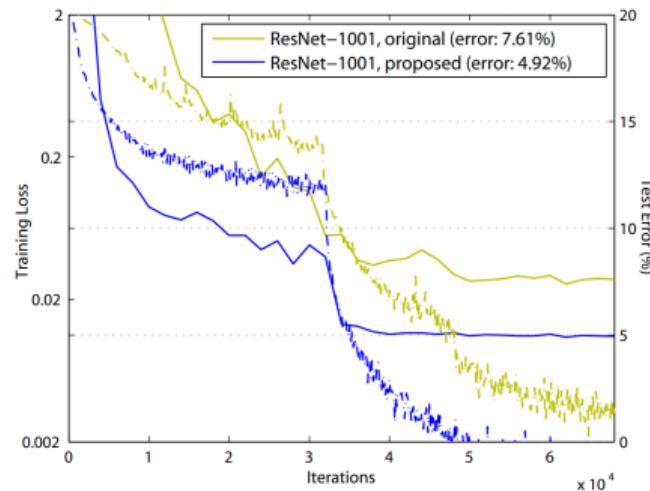
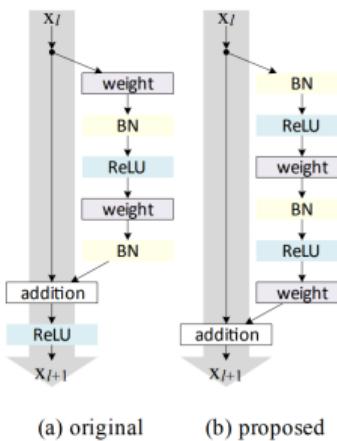
- Инициализация: $x \sim N(0, \sqrt{\frac{2}{n}})$, где n — число входных нейронов (для светки размером $h \times w$ на количестве карт с имеем $n = cwh$)
- Процедура обучения — SGD с momentum и L_2 -регуляризацией
- 10 кропов в teste, как в AlexNet
- Ошибка top-5: 3.57% на ансамбле, включающем 2 ResNet-152

ResNet — проблема решена!



Развитие идей Inception и ResNet

- При использовании обычного Res-блока дальнейшее увеличение количества слоев не давало прироста
- ResNet-V2: при небольшом изменении порядка блоков ReLU, BatchNorm и CONV оказалось возможным обучить хорошую сеть даже на 1001 слой²²

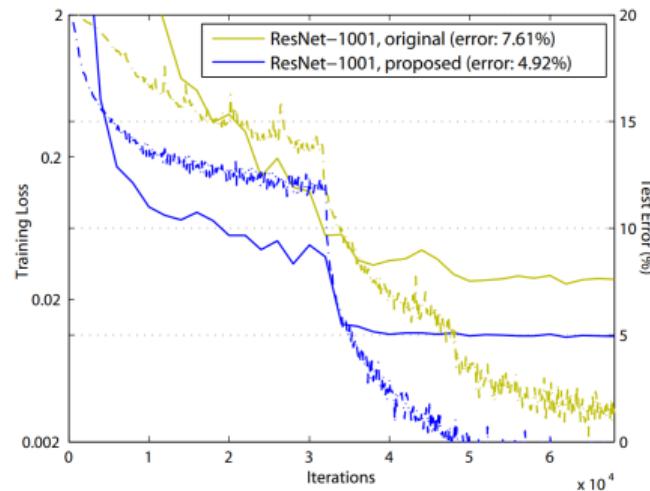
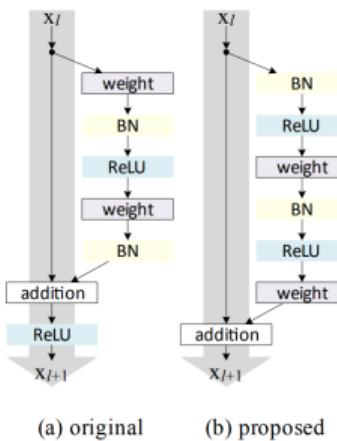


²²He K. et al. Identity mappings in deep residual networks. 2016.

²³Szegedy C. et al. Inception-v4, inception-resnet and the impact of residual connections on learning. 2017.

Развитие идей Inception и ResNet

- При использовании обычного Res-блока дальнейшее увеличение количества слоев не давало прироста
- ResNet-V2: при небольшом изменении порядка блоков ReLU, BatchNorm и CONV оказалось возможным обучить хорошую сеть даже на 1001 слой²²



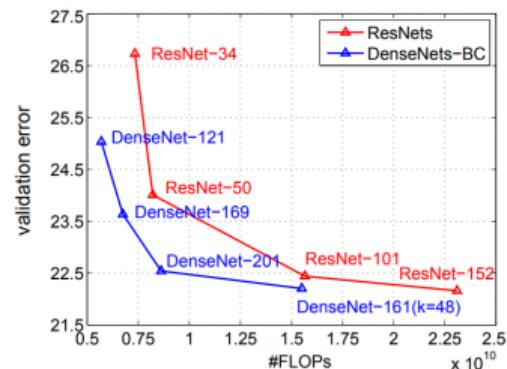
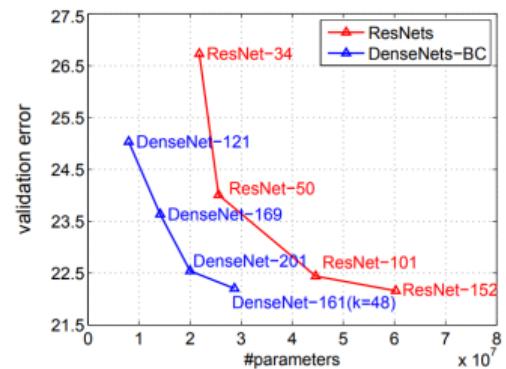
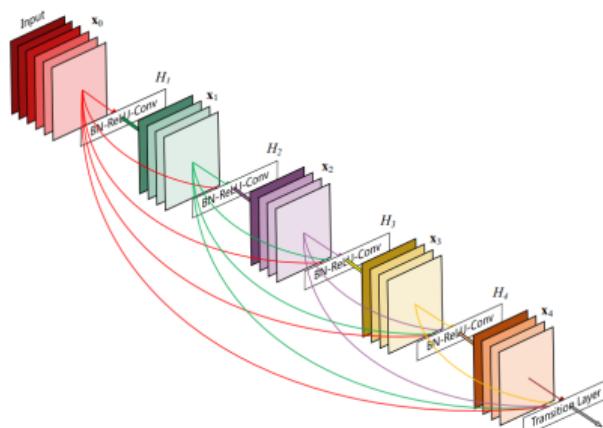
А если объединить идеи последних Inception и ResNet, то получится еще лучше (например, Inception-ResNet-V2)²³

²²He K. et al. Identity mappings in deep residual networks. 2016.

²³Szegedy C. et al. Inception-v4, inception-resnet and the impact of residual connections on learning. 2017.

DenseNet: еще больше соединений²⁴

- Было предложено в рамках слоев одинакового пространственного разрешения использовать вообще все возможные связи с помощью конкатенации
- Таким образом, если слоев L , то таких связей будет $L(L - 1)/2$
- Такая сеть называется DenseNet и выигрывает у ResNet



²⁴Huang G. et al. Densely connected convolutional networks. 2017.

Xception, ResNeXt и ShuffleNet — игра с типами сверток

Также были получены хорошие результаты от использования разных видов сверток:

- ResNeXt²⁵ — улучшение res-блока с помощью групповых сверток (grouped convolution)
- Xception²⁶ — улучшение inception-блока с помощью поканально разделяемых сверток (depthwise separable convolution)
- ShuffleNet²⁷ — объединение групповых и поканально разделяемых сверток с помощью операции перемешивания (shuffle)

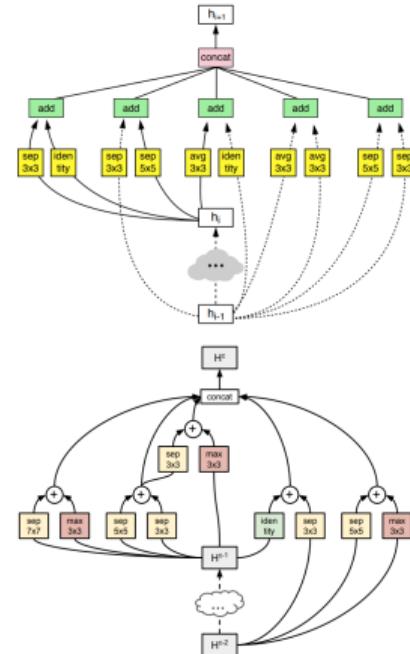
²⁵Xie S. et al. Aggregated residual transformations for deep neural networks. 2017.

²⁶Chollet F. Xception: Deep learning with depthwise separable convolutions. 2017.

²⁷Zhang X. et al. Shufflenet: An extremely efficient convolutional neural network for mobile devices. 2018. 

Автоматические методы нахождения архитектур

- В последнее время все большую популярность приобрели методы автоматического поиска архитектуры нейросетей (Neural Architecture Search, NAS)
- Используется весь арсенал накопленных методов: от эволюционных алгоритмов и обучения с подкреплением (NASNet²⁸) до дифференцируемых предсказателей качества по архитектуре (PNASNet²⁹)
- При этом получающиеся нейросети очень сложно интерпретировать



²⁸Zoph B. et al. Learning transferable architectures for scalable image recognition. 2018.

²⁹Liu C. et al. Progressive neural architecture search. 2018.

Заключительные выводы

- СНС на данный момент классифицируют лучше человека

Заключительные выводы

- СНС на данный момент классифицируют лучше человека
- Глубина и сложность СНС увеличиваются год от года

Заключительные выводы

- СНС на данный момент классифицируют лучше человека
- Глубина и сложность СНС увеличиваются год от года
- Первая классическая СНС появилась еще 30 лет назад (в 1989 году)

Заключительные выводы

- СНС на данный момент классифицируют лучше человека
- Глубина и сложность СНС увеличиваются год от года
- Первая классическая СНС появилась еще 30 лет назад (в 1989 году)
- AlexNet совместила в себе большое число находок, используемых и поныне

Заключительные выводы

- СНС на данный момент классифицируют лучше человека
- Глубина и сложность СНС увеличиваются год от года
- Первая классическая СНС появилась еще 30 лет назад (в 1989 году)
- AlexNet совместила в себе большое число находок, используемых и поныне
- Иногда важные находки появляются вовсе не у чемпионов соревнований

Заключительные выводы

- СНС на данный момент классифицируют лучше человека
- Глубина и сложность СНС увеличиваются год от года
- Первая классическая СНС появилась еще 30 лет назад (в 1989 году)
- AlexNet совместила в себе большое число находок, используемых и поныне
- Иногда важные находки появляются вовсе не у чемпионов соревнований
- Разреженная архитектура эффективнее плотной

Заключительные выводы

- СНС на данный момент классифицируют лучше человека
- Глубина и сложность СНС увеличиваются год от года
- Первая классическая СНС появилась еще 30 лет назад (в 1989 году)
- AlexNet совместила в себе большое число находок, используемых и поныне
- Иногда важные находки появляются вовсе не у чемпионов соревнований
- Разреженная архитектура эффективнее плотной
- Тождественные связи крайне полезны

Заключительные выводы

- СНС на данный момент классифицируют лучше человека
- Глубина и сложность СНС увеличиваются год от года
- Первая классическая СНС появилась еще 30 лет назад (в 1989 году)
- AlexNet совместила в себе большое число находок, используемых и поныне
- Иногда важные находки появляются вовсе не у чемпионов соревнований
- Разреженная архитектура эффективнее плотной
- Тождественные связи крайне полезны
- Разные типы сверток могут повысить качество классификации

Заключительные выводы

- СНС на данный момент классифицируют лучше человека
- Глубина и сложность СНС увеличиваются год от года
- Первая классическая СНС появилась еще 30 лет назад (в 1989 году)
- AlexNet совместила в себе большое число находок, используемых и поныне
- Иногда важные находки появляются вовсе не у чемпионов соревнований
- Разреженная архитектура эффективнее плотной
- Тождественные связи крайне полезны
- Разные типы сверток могут повысить качество классификации
- На данный момент автоматический поиск архитектур крайне востребован

Время для вопросов



Спасибо за внимание!

