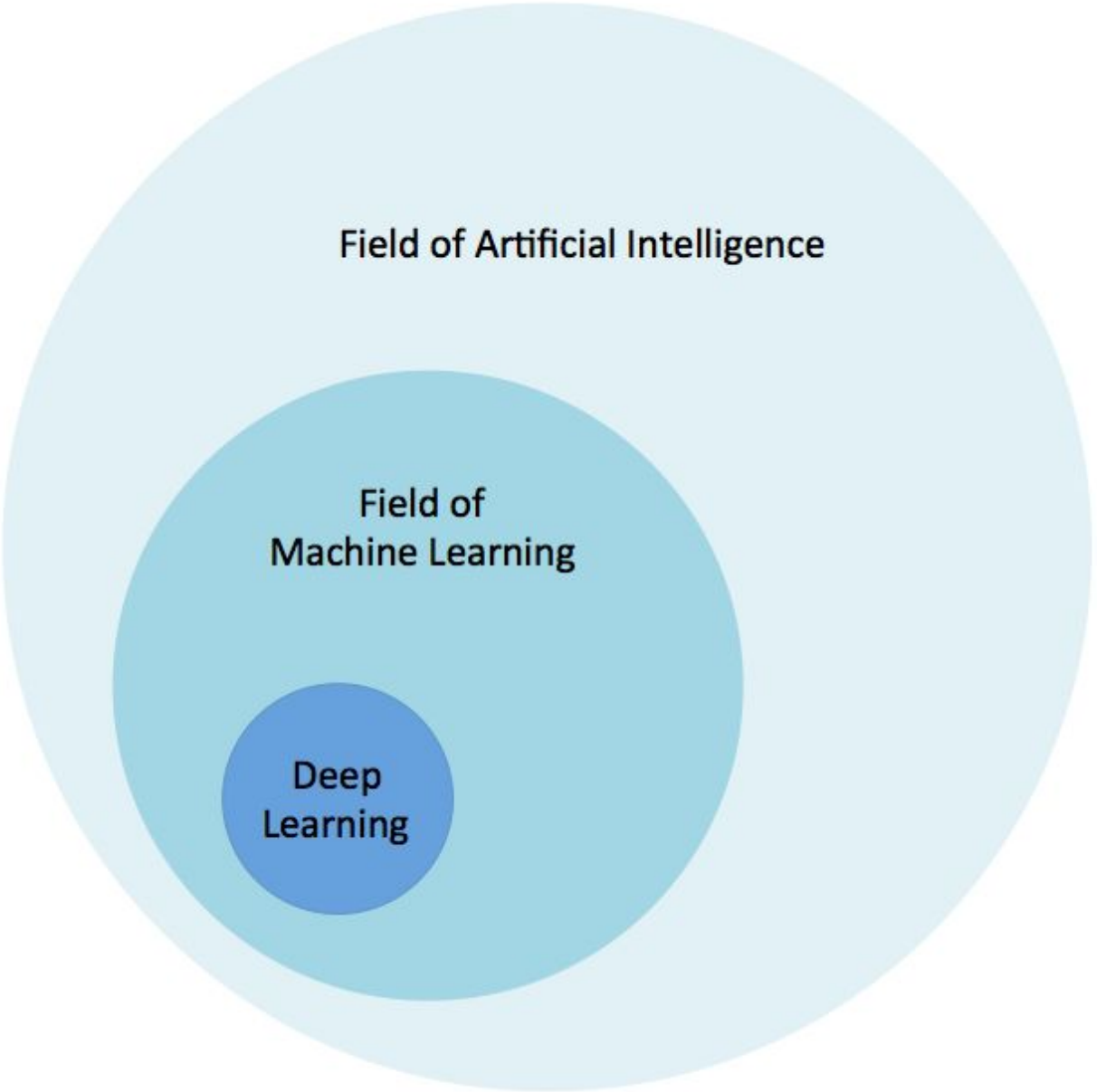
Deep Learning

Сапунов Григорий
CTO / Intento (inten.to)

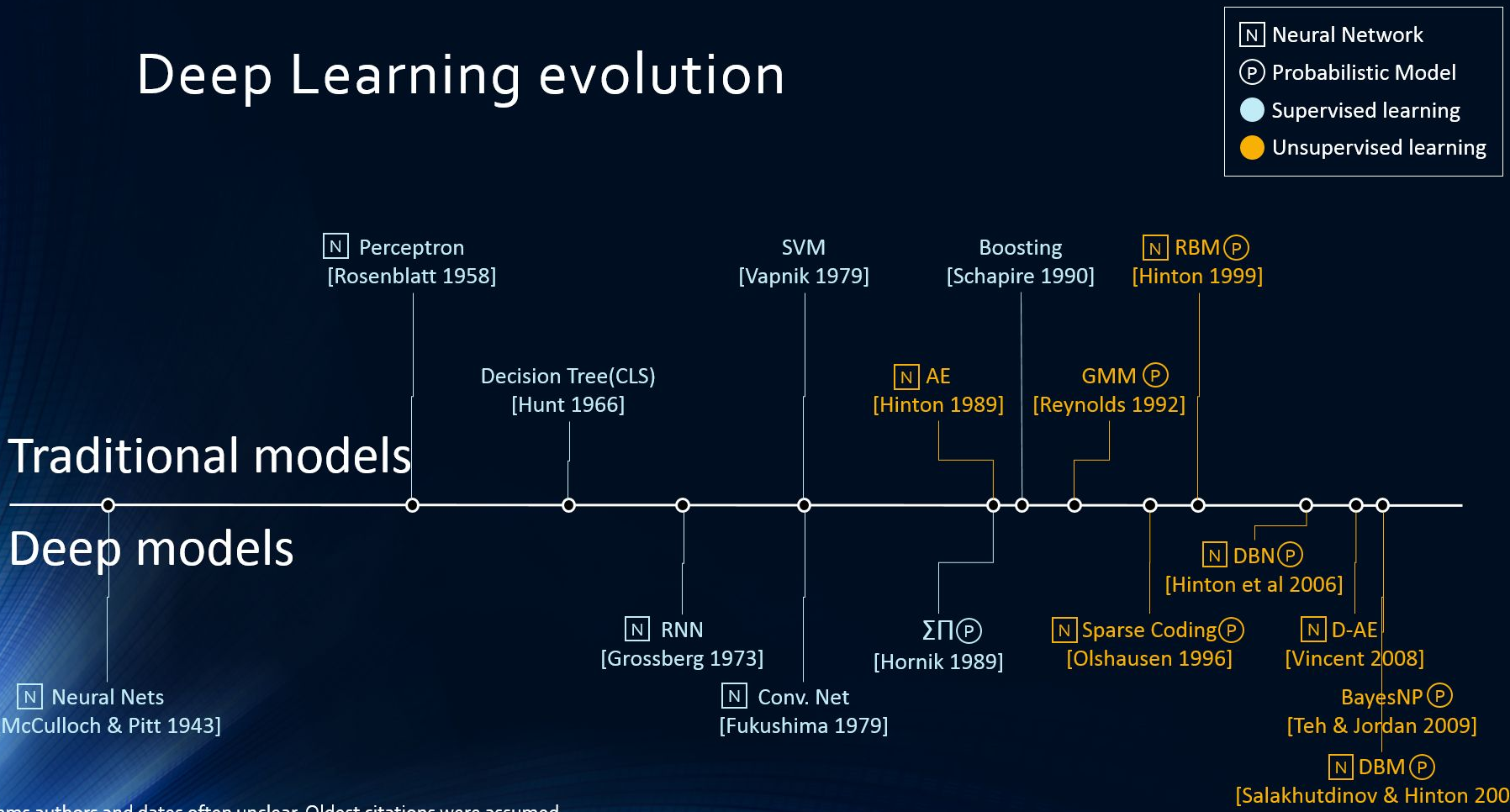
Deep Learning vs. Machine Learning

В чём различия и что общее?

Место Deep Learning среди других областей



Deep Learning evolution



Algorithms authors and dates often unclear. Oldest citations were assumed
Classifications based on Yann LeCun's Deep Learning class at NYU – spring 2014

ML / DL

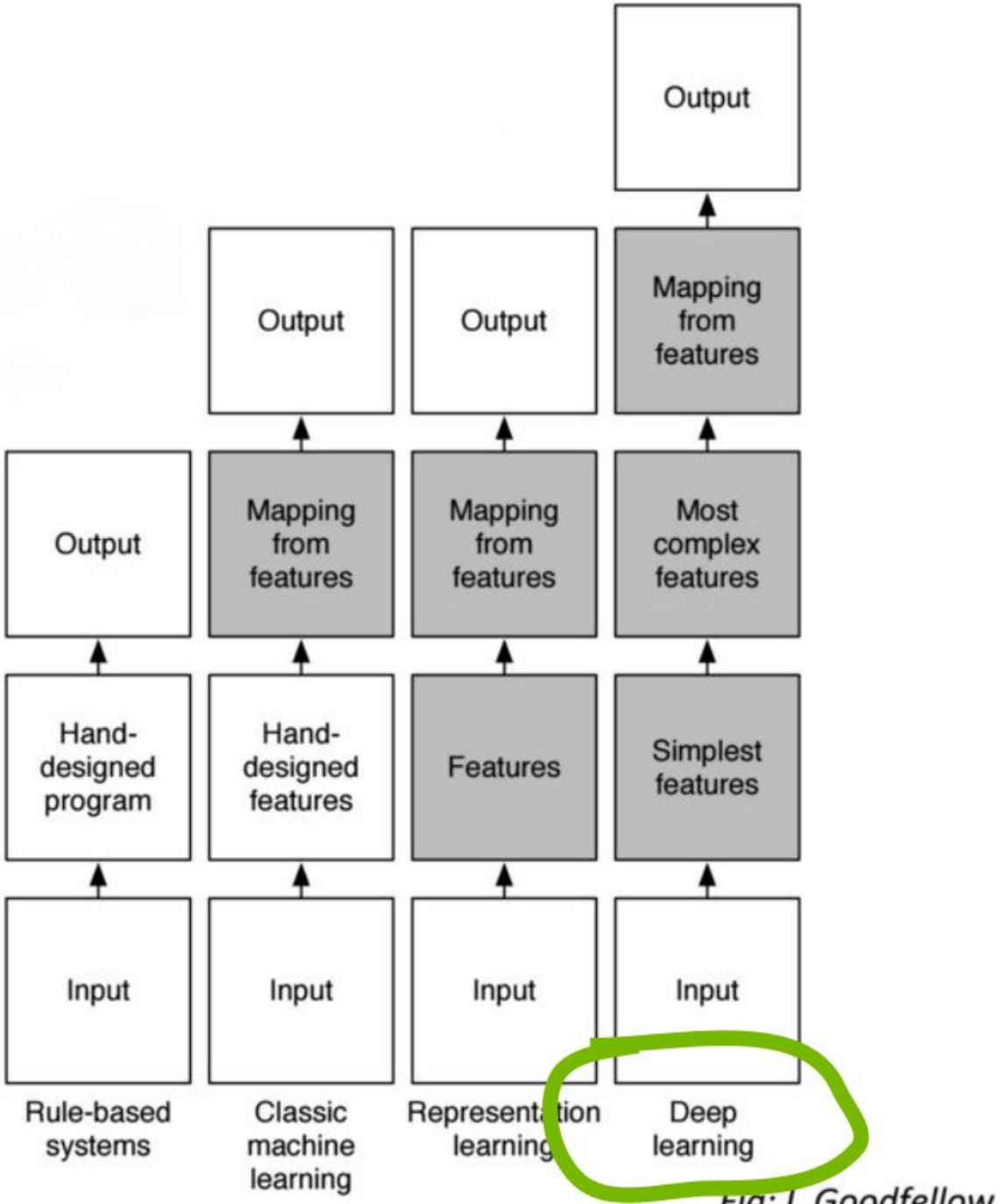
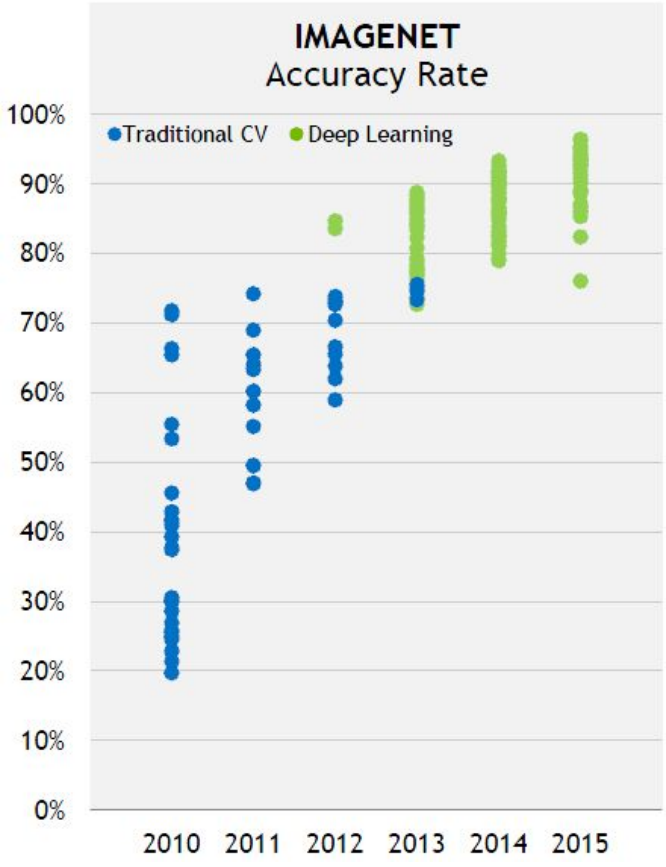
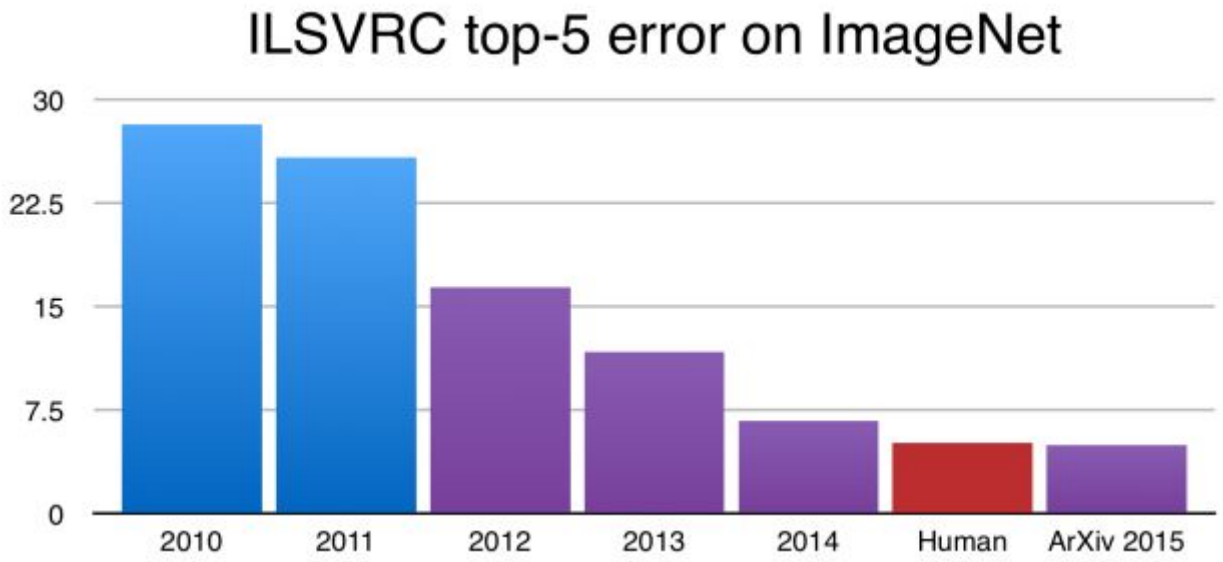


Fig: I. Goodfellow

Важные тренды

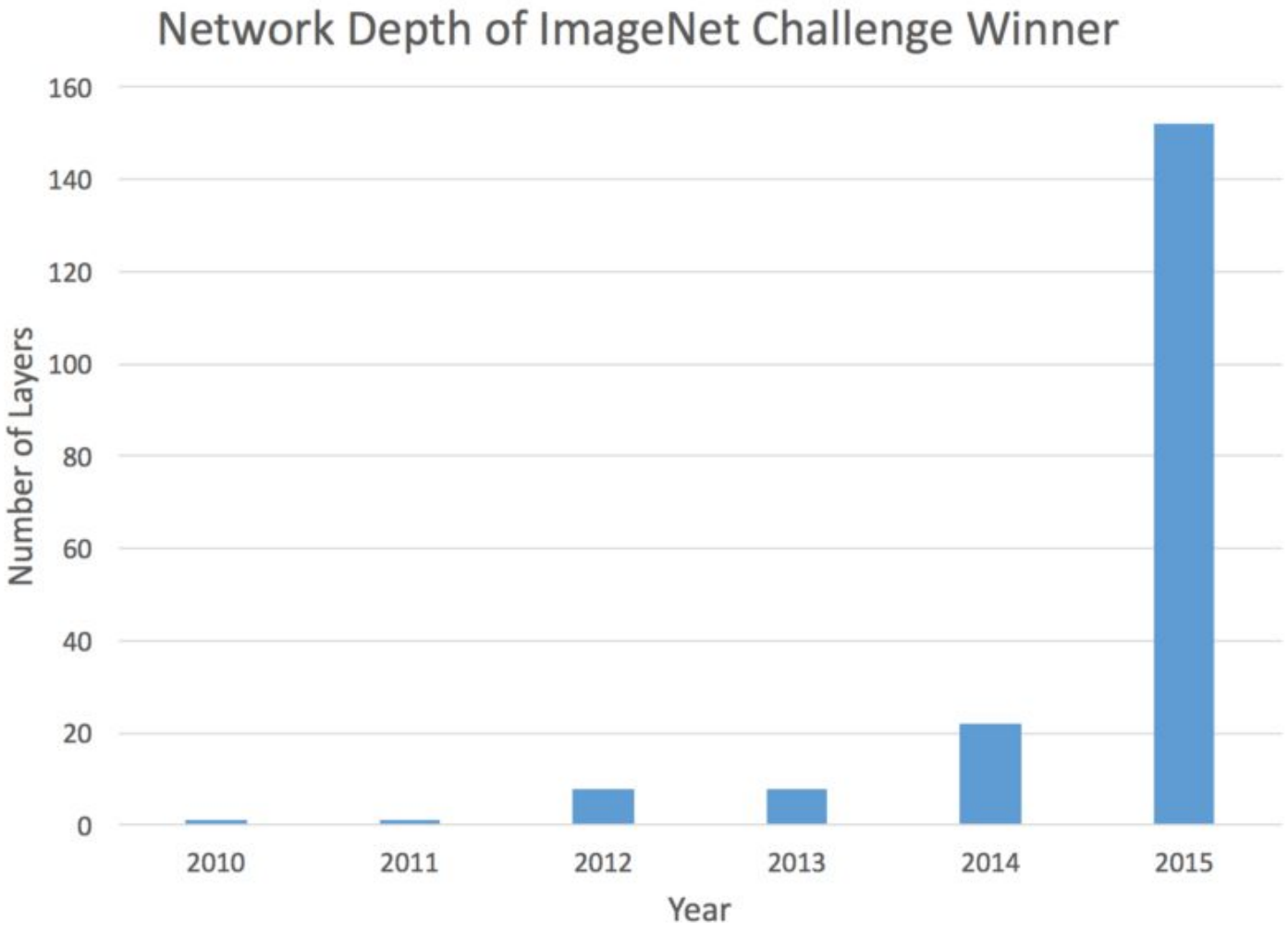
#1. Точность сетей растёт



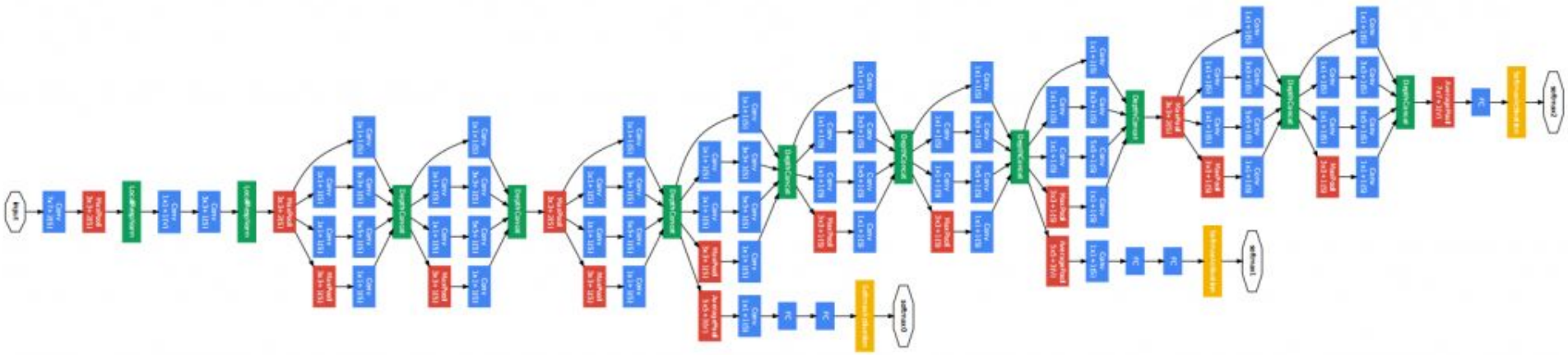


- Blue: Traditional CV
- Purple: Deep Learning
- Red: Human

#2. Сложность сетей растёт

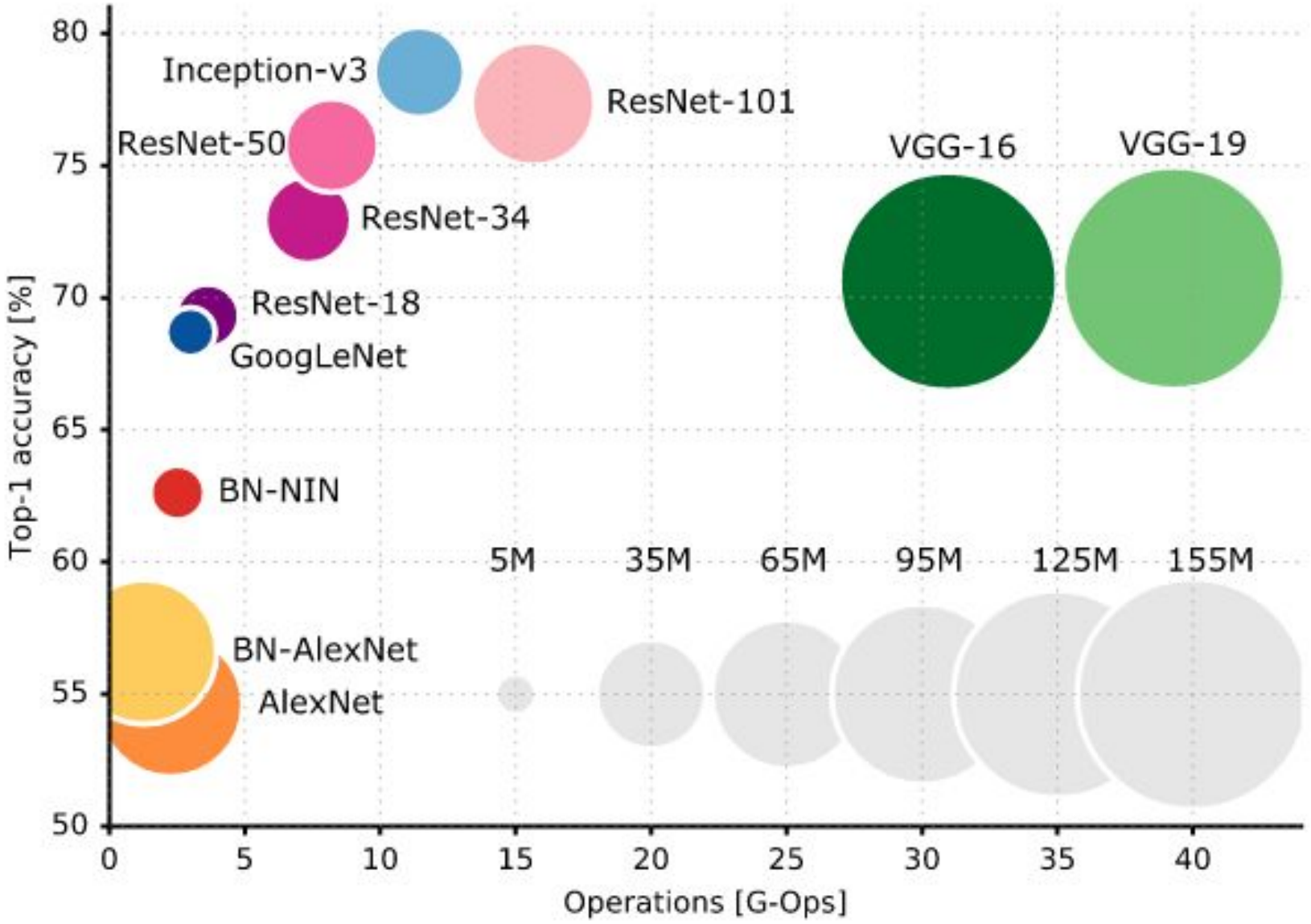


#2. Сложность сетей растёт



Реальная нейросеть: GoogLeNet (2014)
http://cs.unc.edu/~wliu/papers/GoogLeNet.pdf

#2. Сложность сетей растёт

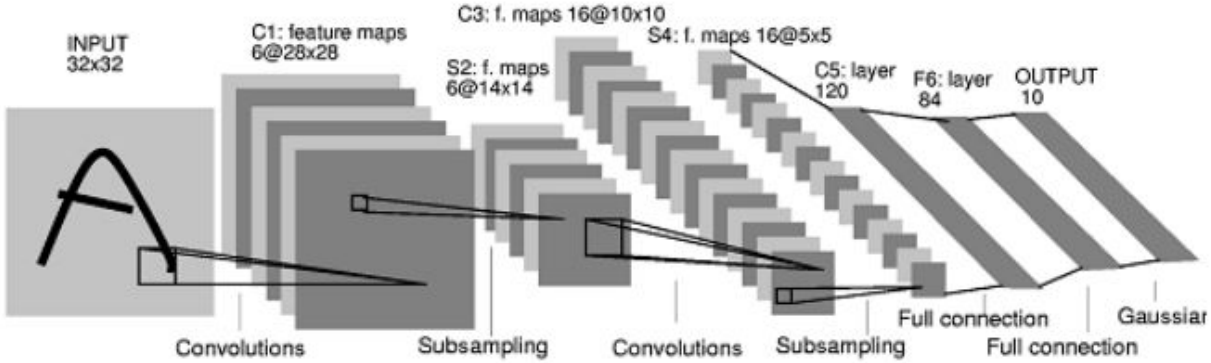


https://culurciello.github.io/tech/2016/06/04/nets.html

#3. Объёмы данных растут

1998

LeCun et al.



# of transistors

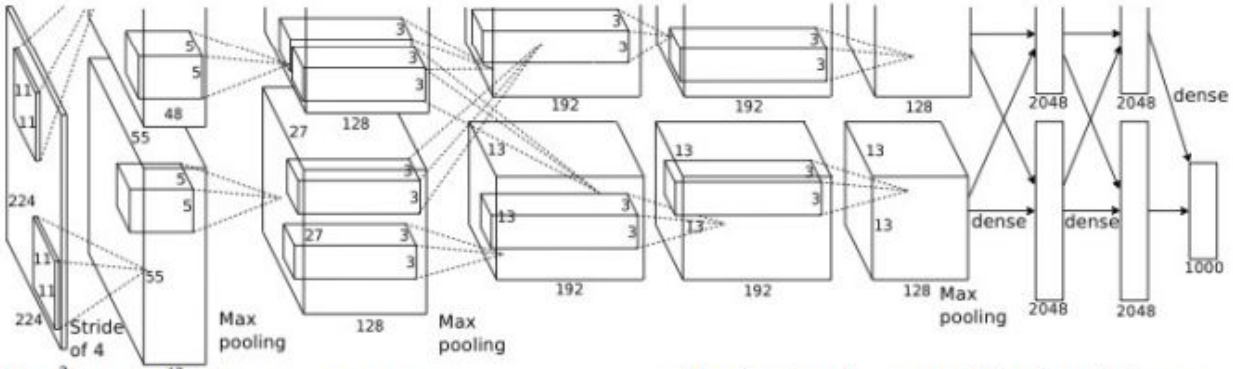
10^6

# of pixels used in training

10^7 NIST

2012

Krizhevsky et al.



# of transistors GPUs

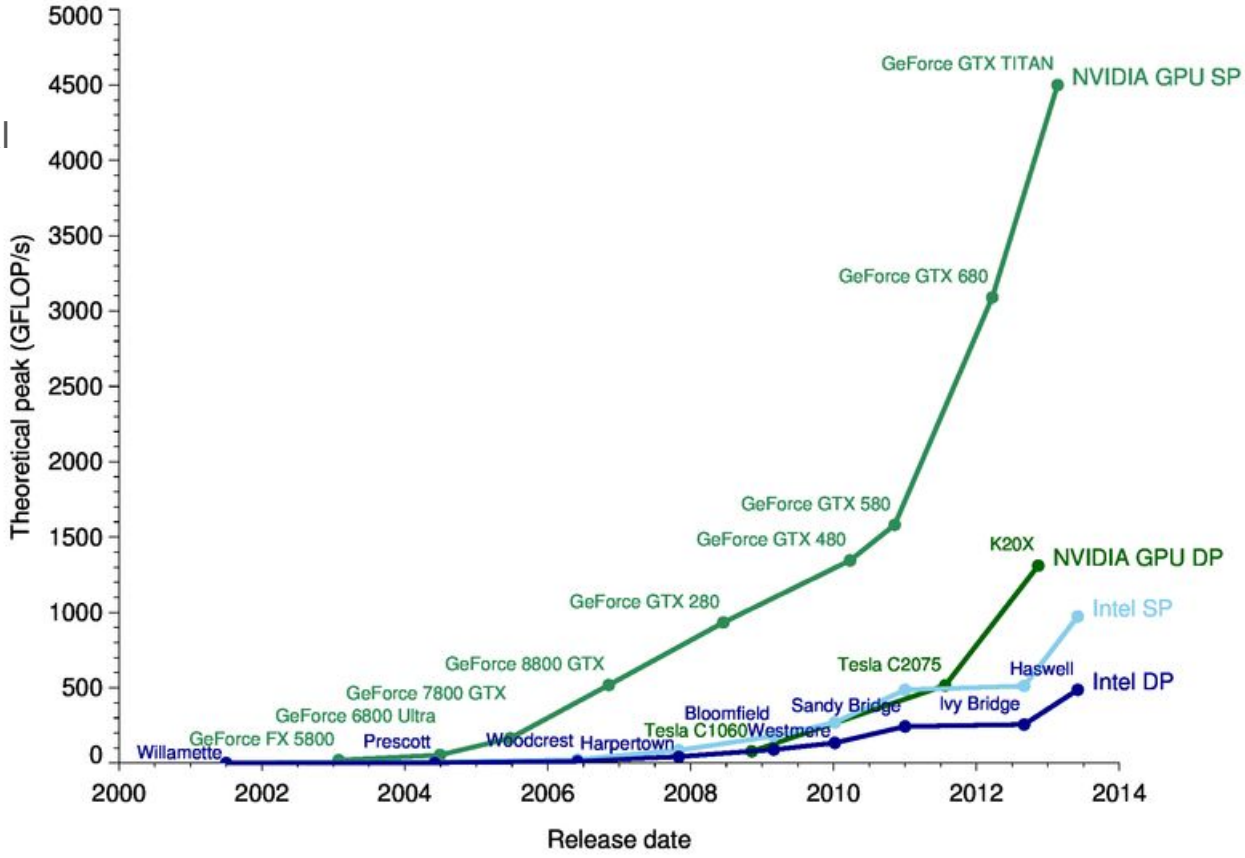
10^9

# of pixels used in training

10^{14} IMAGENET

#4. Вычислительные мощности растут

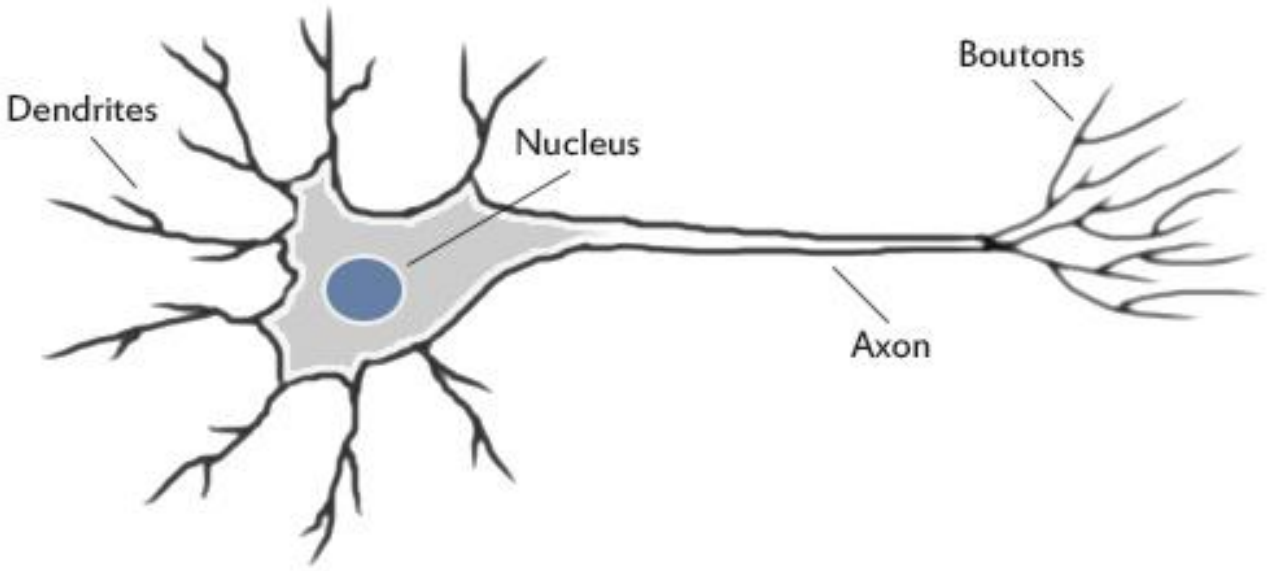
- NVIDIA DGX-1 ($129,000)
  - 170 TFLOPS (FP16)
  - 85 TFLOPS (FP32)
- NVIDIA GTX Titan X Pascal ($1000)
  - 11 TFLOPS (FP32)
- NVIDIA GTX 1080
  - 8 TFLOPS (FP32)
- NVIDIA GTX Titan X Old
  - 6.1 TFLOPS (FP32)
- NVIDIA Drive PX-2
  - 8.0 TFLOPS
- NVIDIA Drive PX
  - 2.3 TFLOPS
- Intel Core i7-6700K
  - ~0.1-0.2 TFLOPS



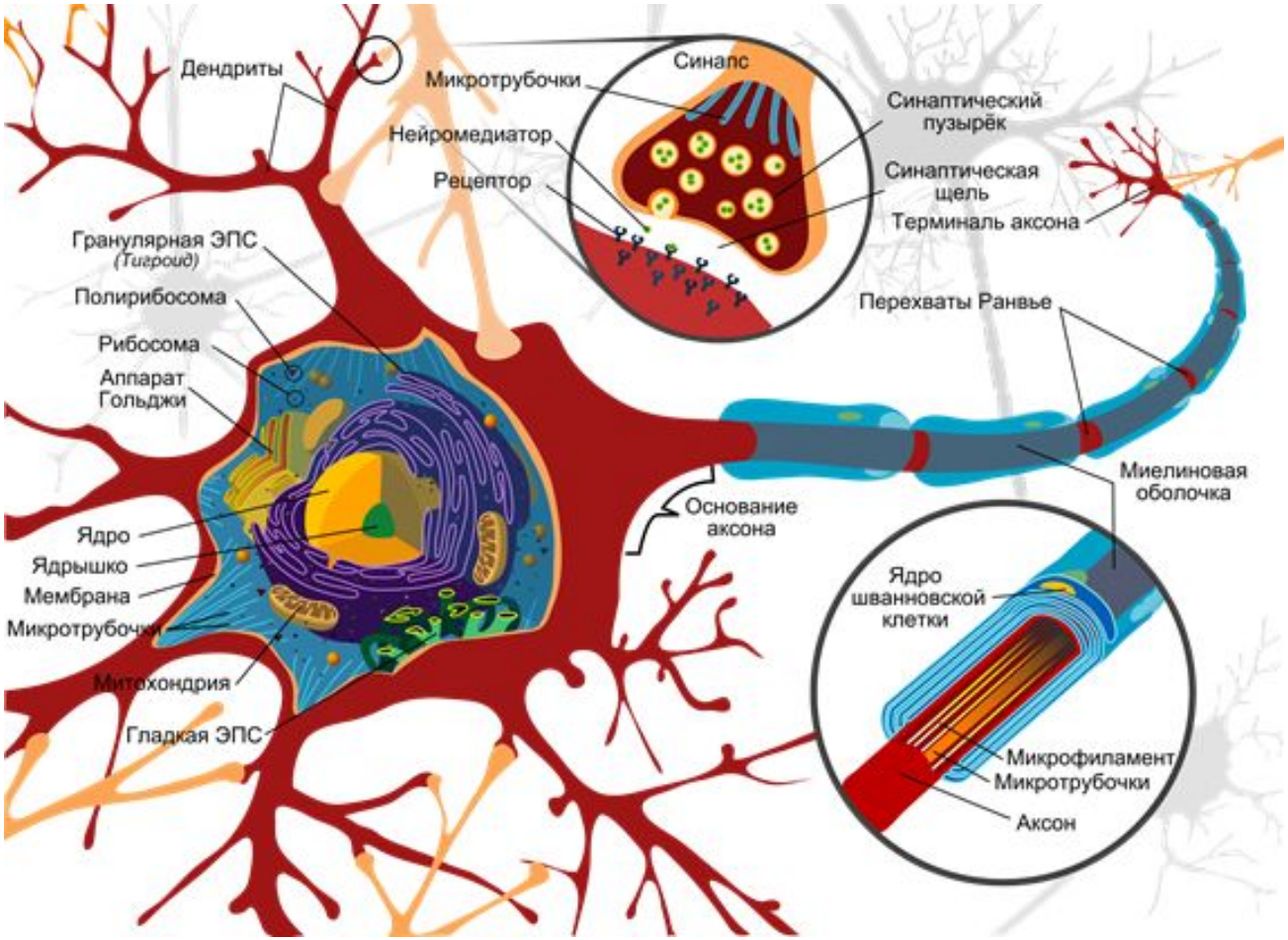
Нейрон и нейросеть

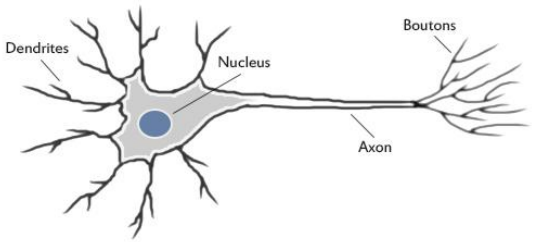
Естественный нейрон

Нейрон — клетка нервной системы, способная к возбуждению и передаче сигнала.



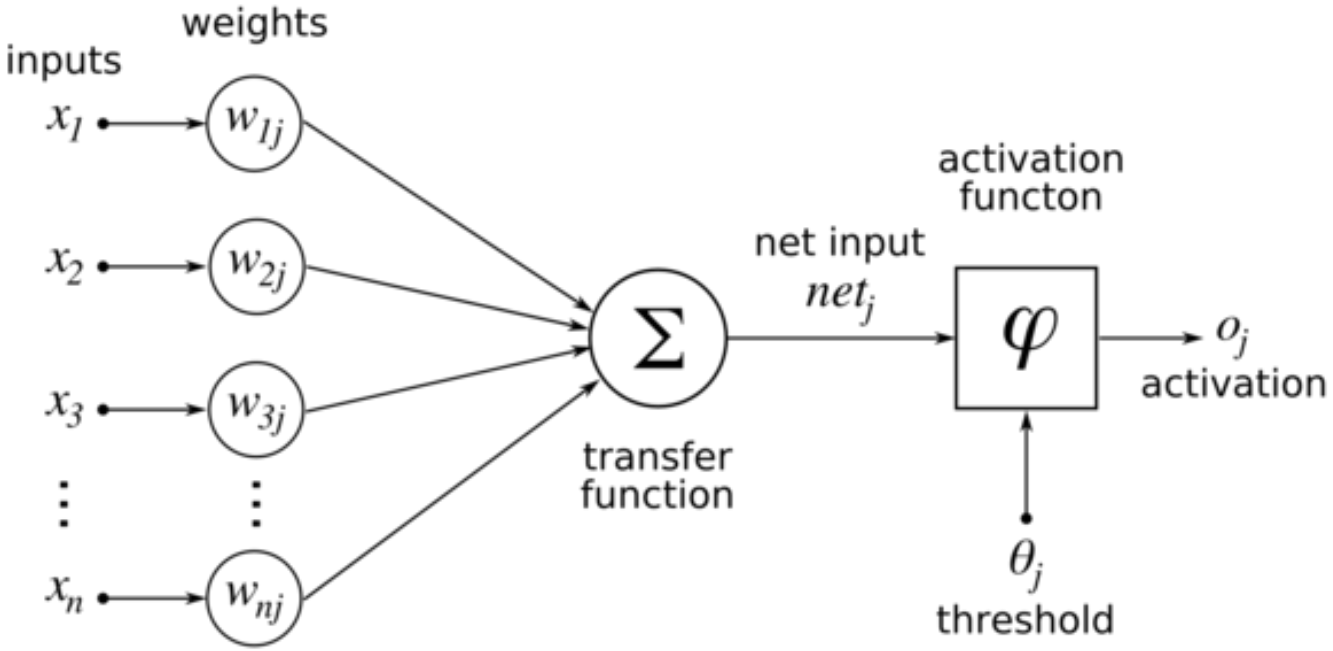
В реальности всё посложнее...

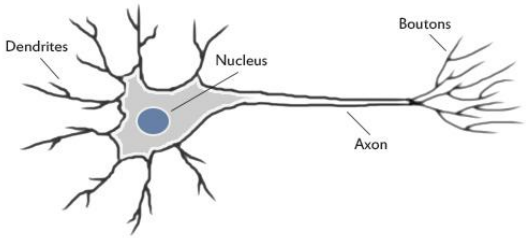




Искусственный нейрон

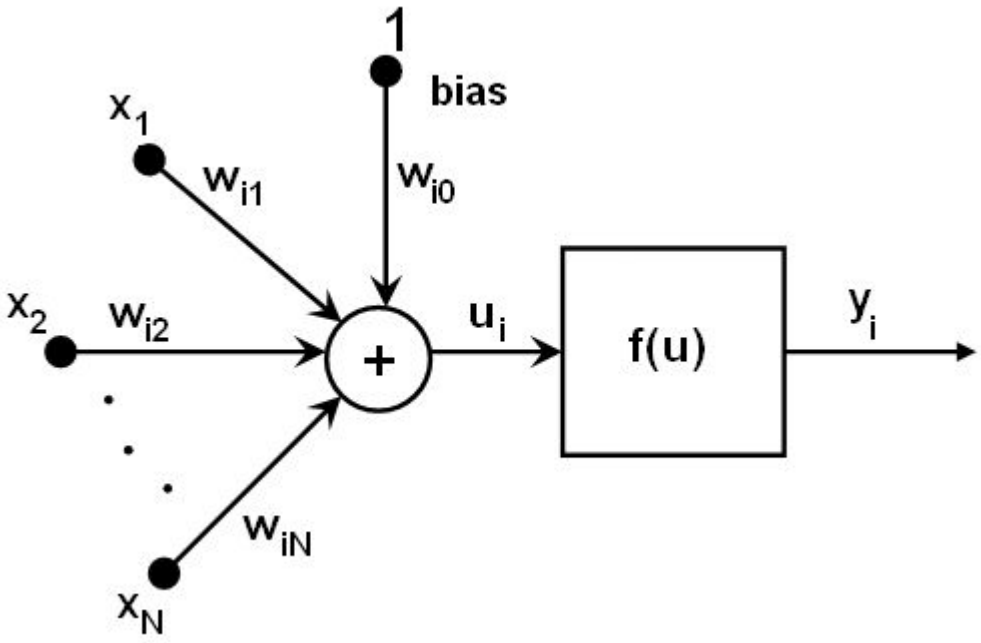
Искусственный нейрон — отдалённое подобие биологического.
Базовый элемент искусственной нейронной сети





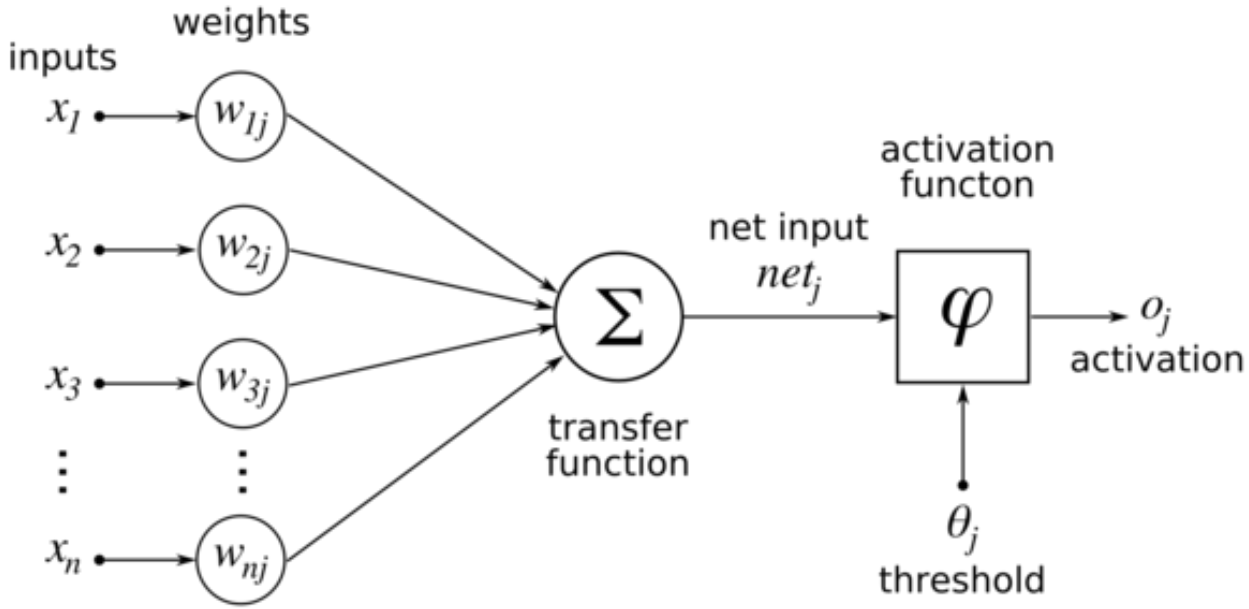
Модель искусственного нейрона

Обычно у нейрона также есть bias-вход, константное смещение, вес на котором также обучается.



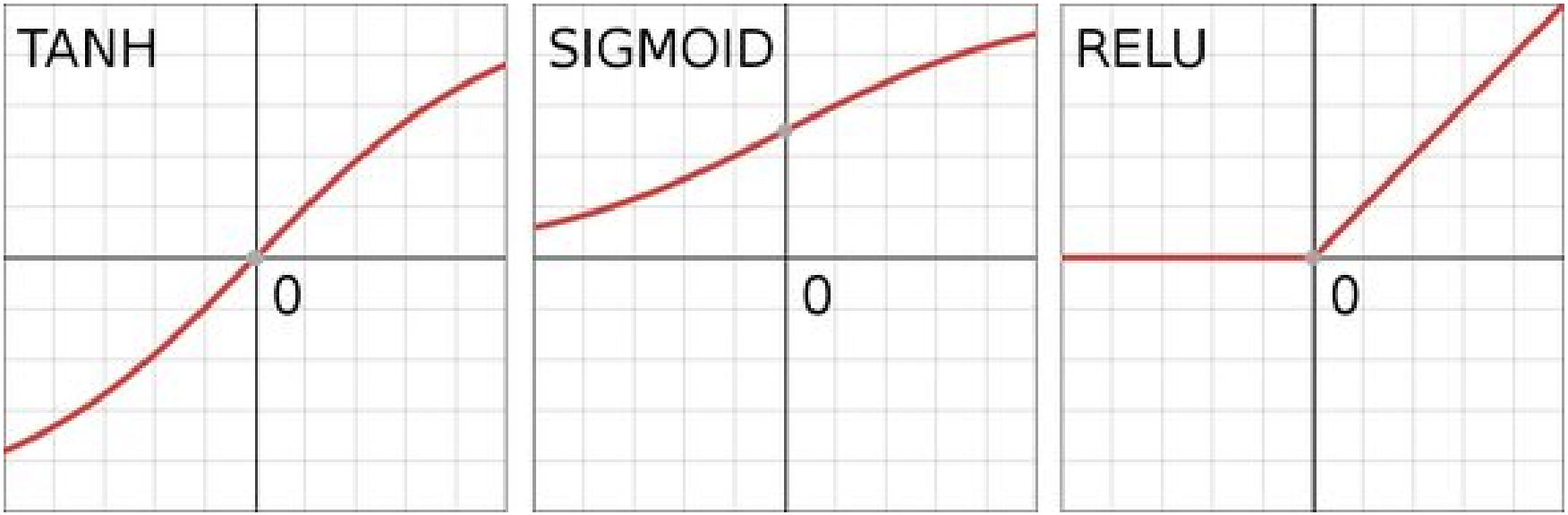
Модель искусственного нейрона

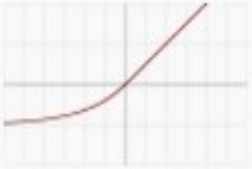
Отдельный нейрон математически полностью аналогичен логистической регрессии (в случае если используется функция активации сигмоида)



$$f(z) = \frac{1}{1 + e^{-z}}.$$

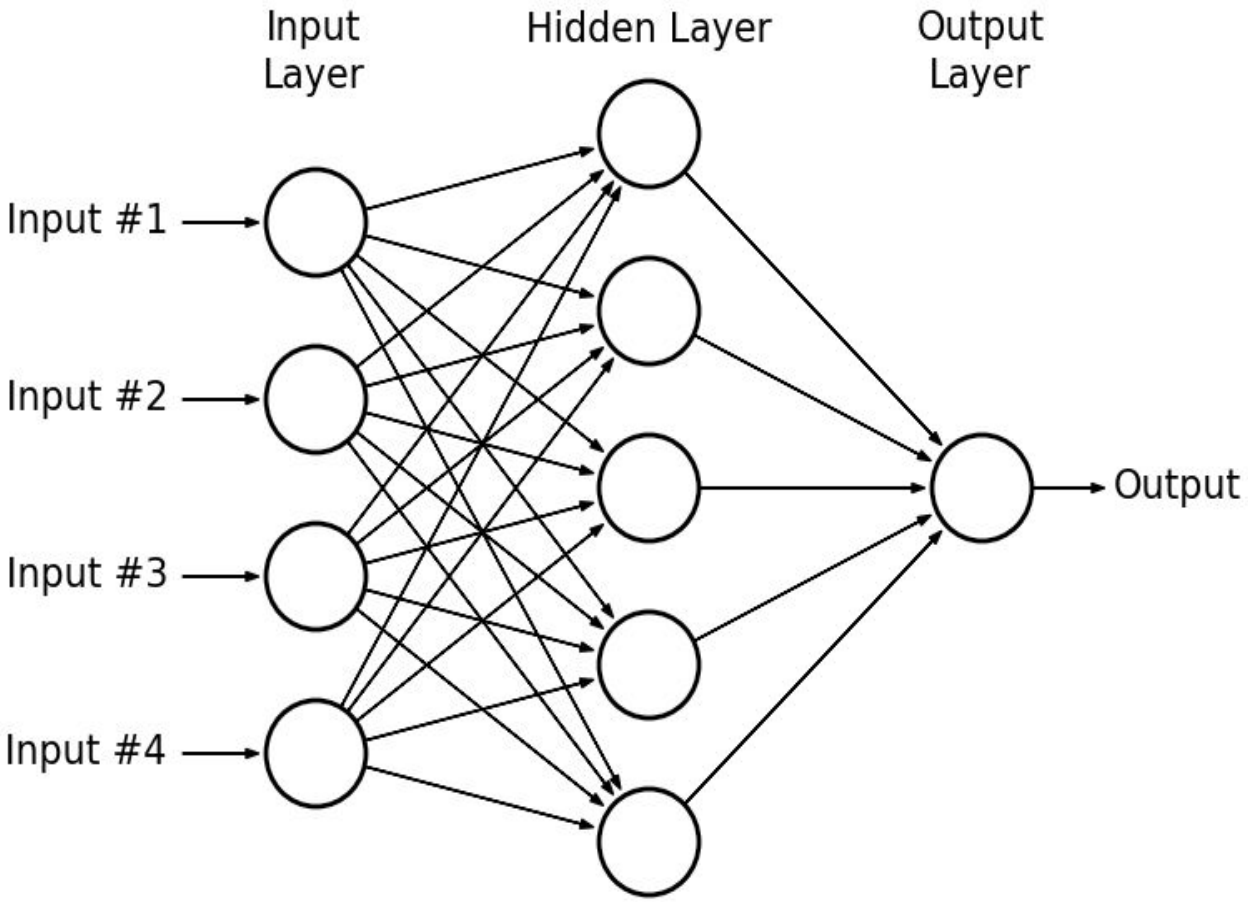
Функции активации: Tanh, Sigmoid, ReLU



Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1-f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2+1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU)[2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU)[3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$

https://en.wikipedia.org/wiki/Activation_function

Искусственная нейросеть: терминология

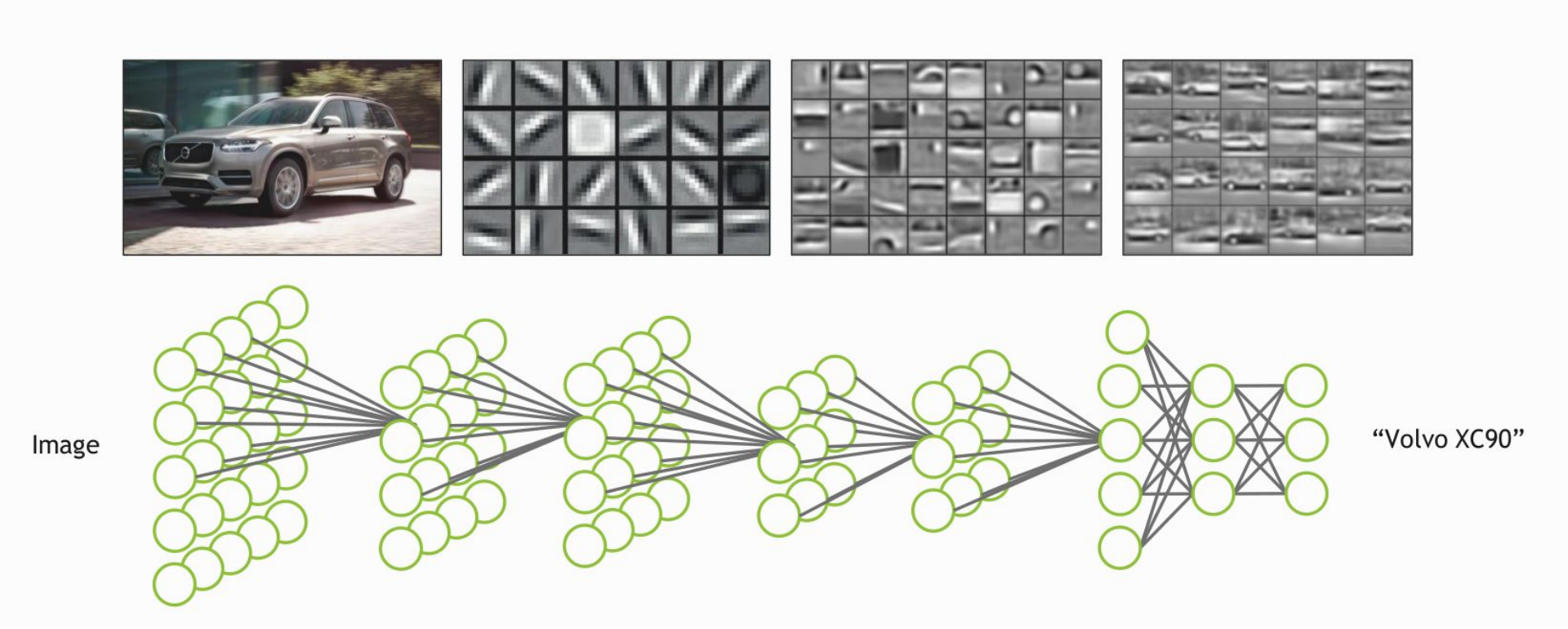


Использование нейросети

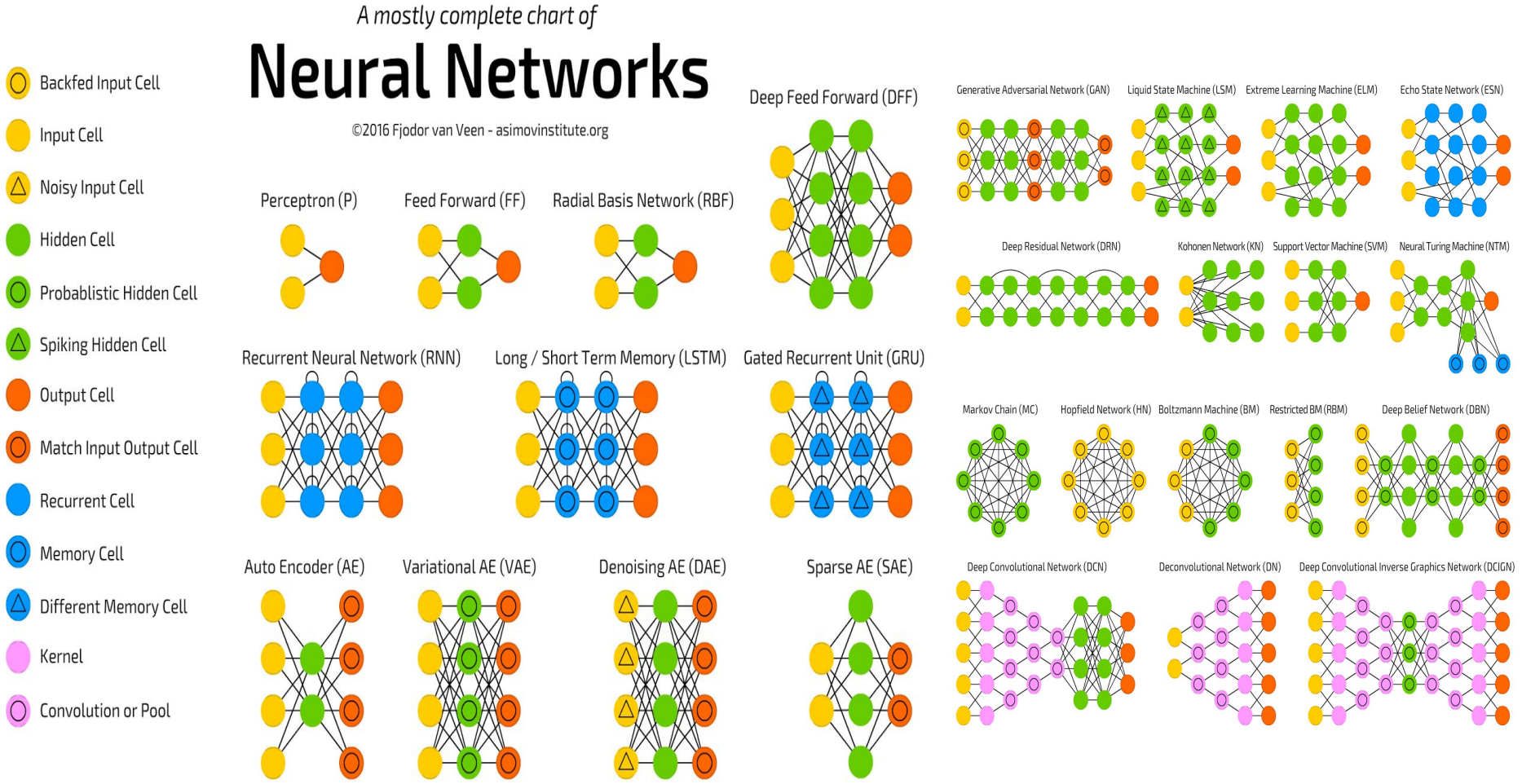
- На вход нейросети для каждого объекта подаётся вектор признаков
- На выходе нейросети получается результат. Вид результата определяется архитектурой нейросети
  - Например, в случае многоклассовой классификации (n классов) выходной слой содержит n нейронов, каждый из которых выдаёт “степень принадлежности” к данному классу (если активация была сигмоидальной)
  - Если нужно получить распределение вероятностей классов, используется функция softmax, нормализующая все выходные значения, так чтобы их сумма была равна 1.

Глубокая нейросеть

Наличие более одного скрытого слоя даёт возможность строить иерархические представления.

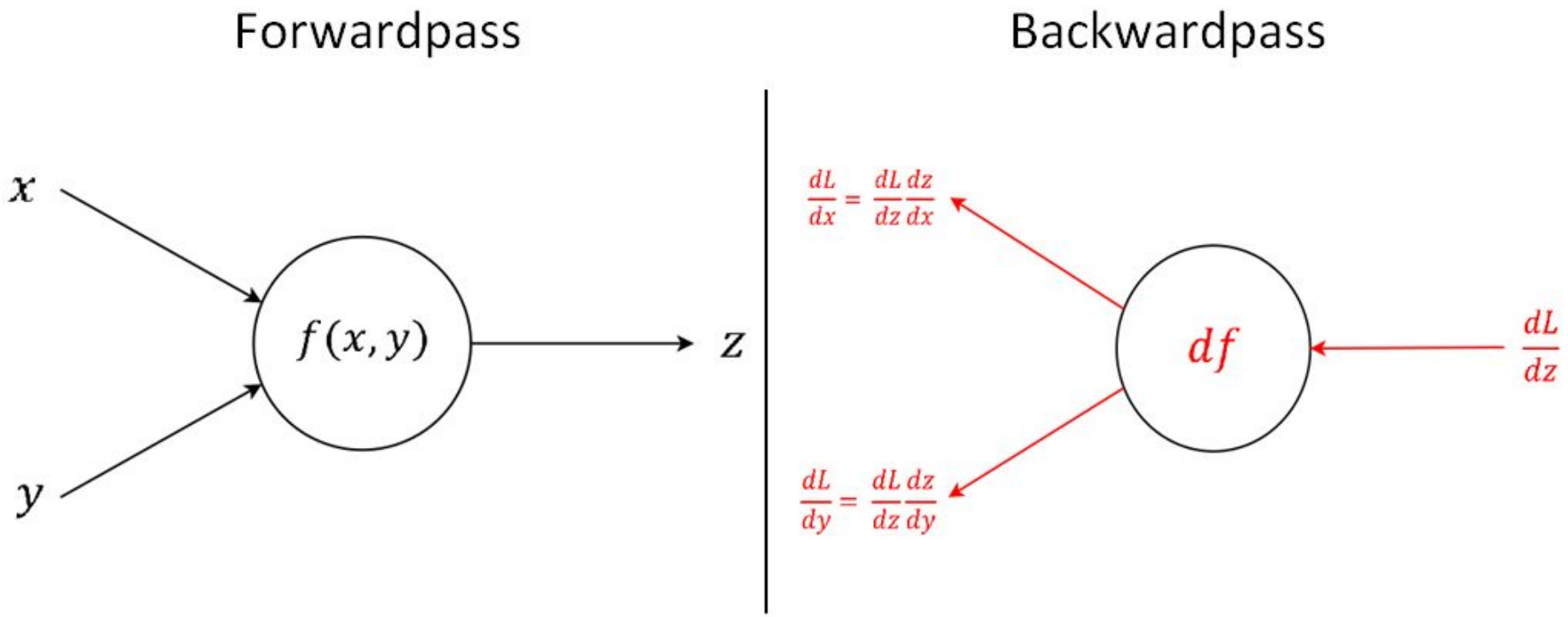


Существует многообразие архитектур



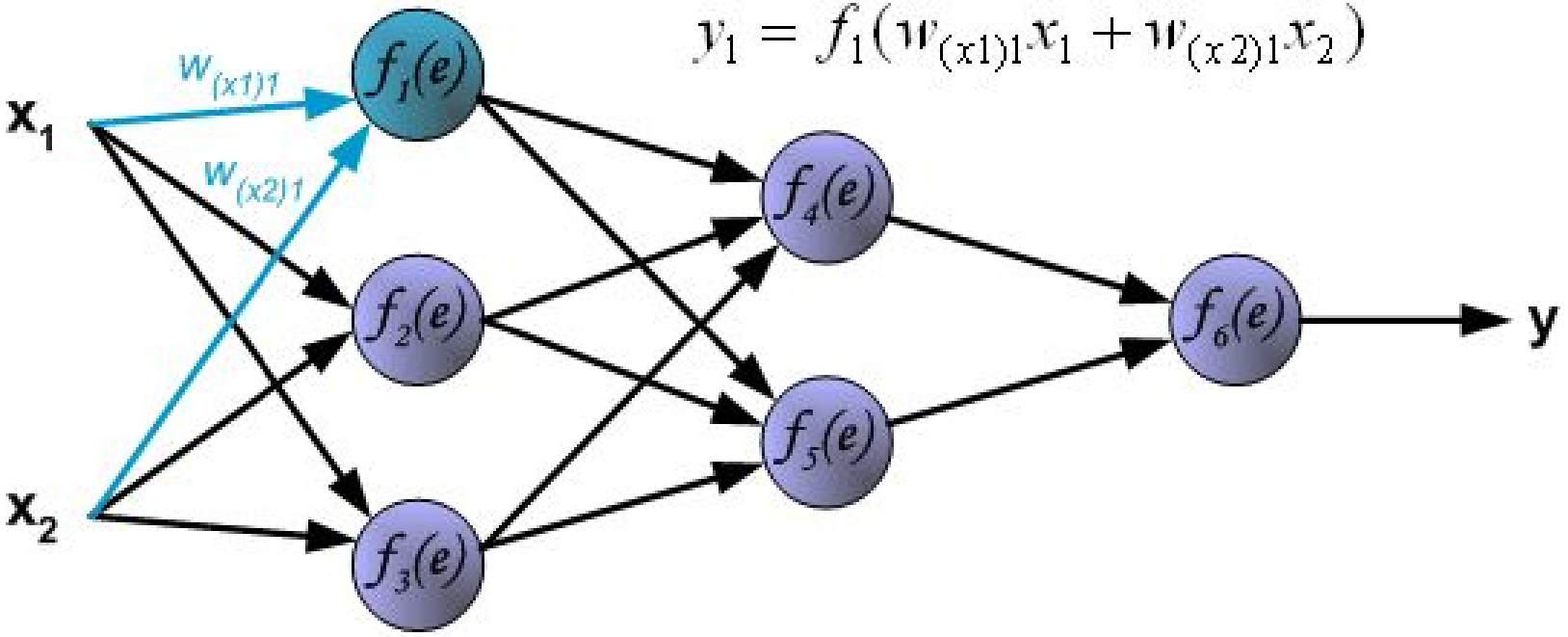
http://www.asimovinstitute.org/neural-network-zoo/

Два режима работы нейрона и нейросети



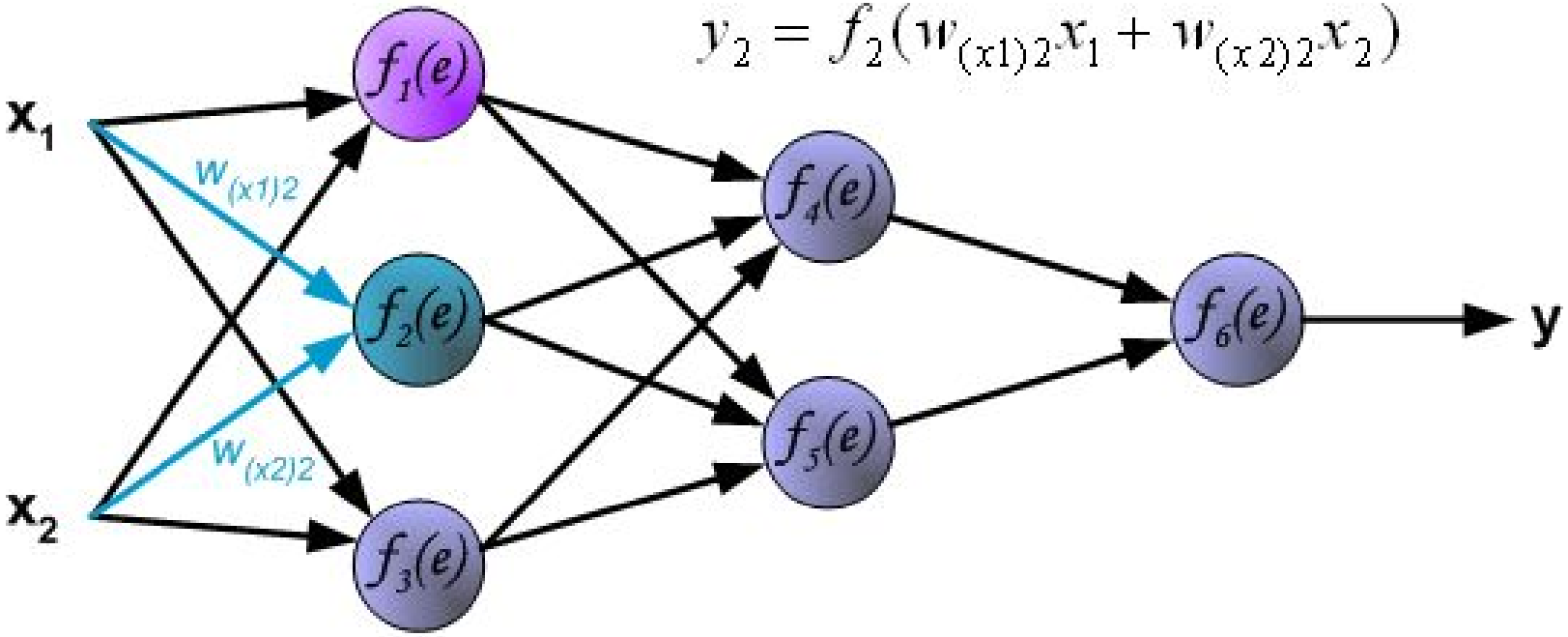
Механизмы работы
1. Forward propagation

Forward propagation



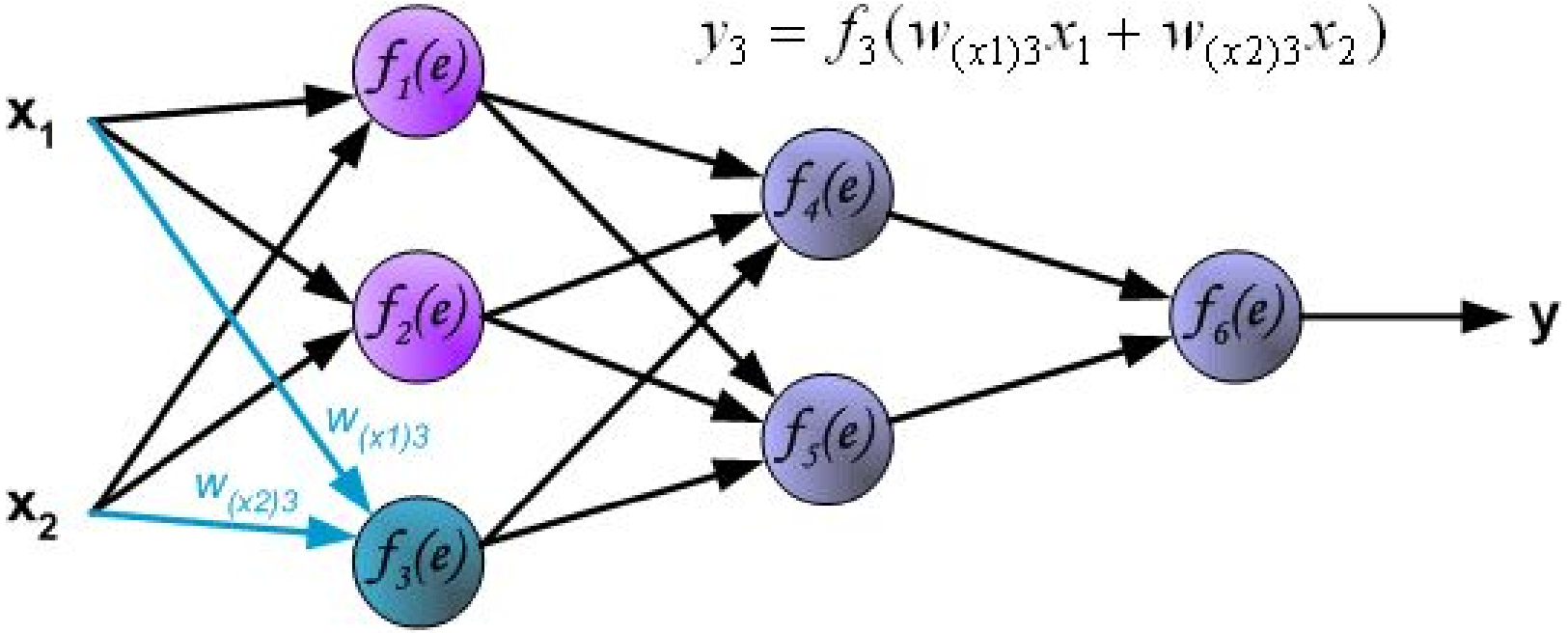
http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Forward propagation



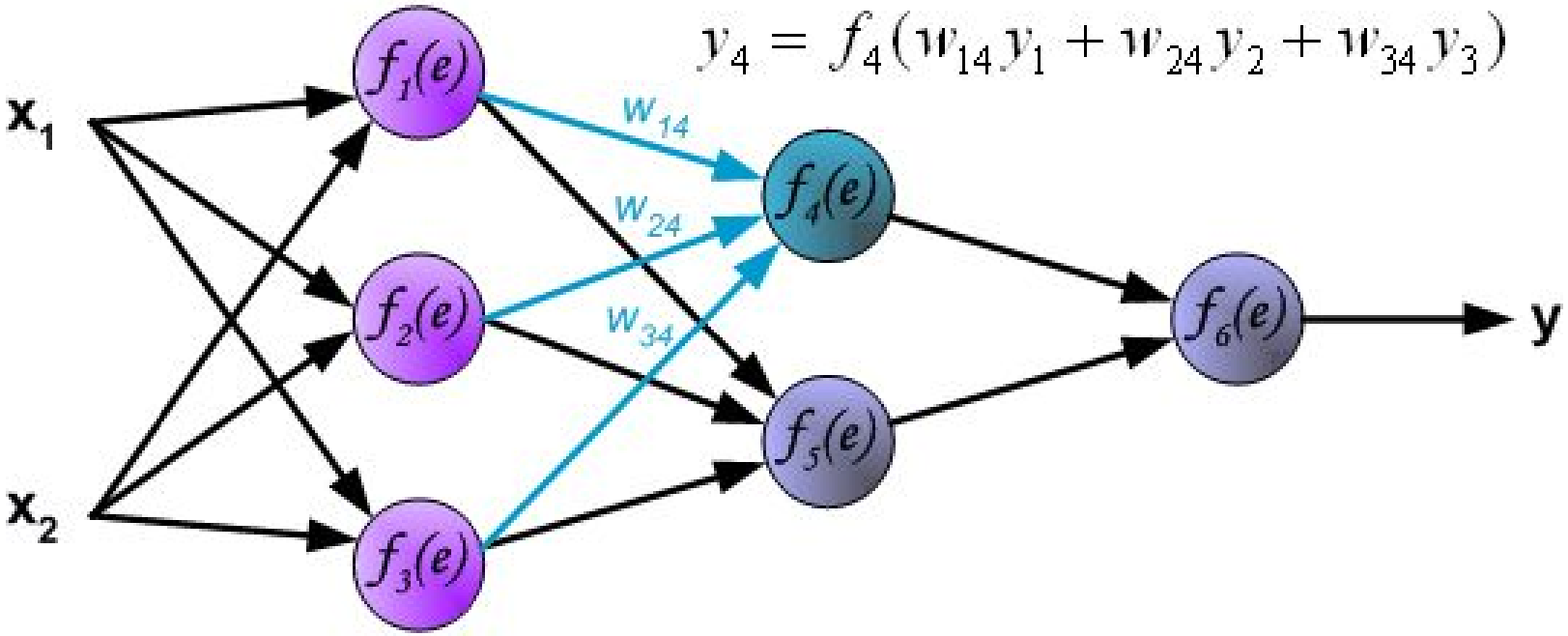
http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Forward propagation



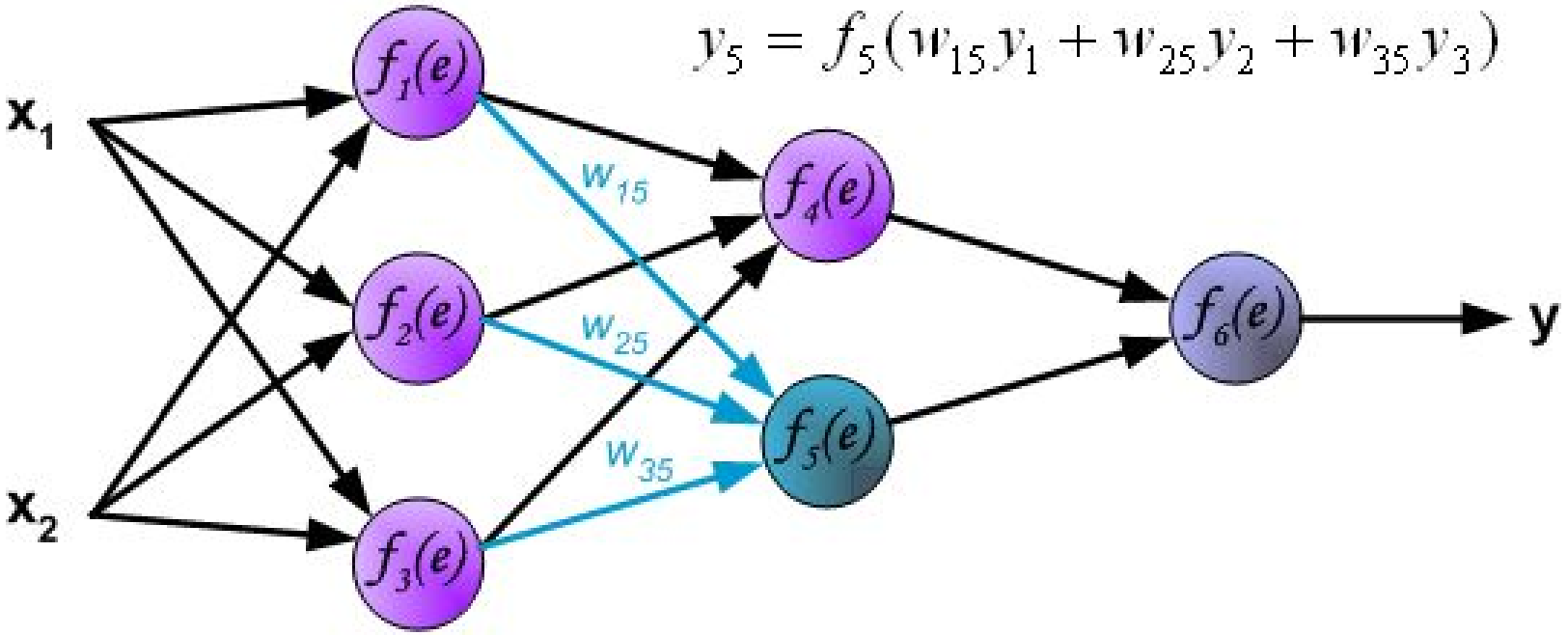
http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Forward propagation



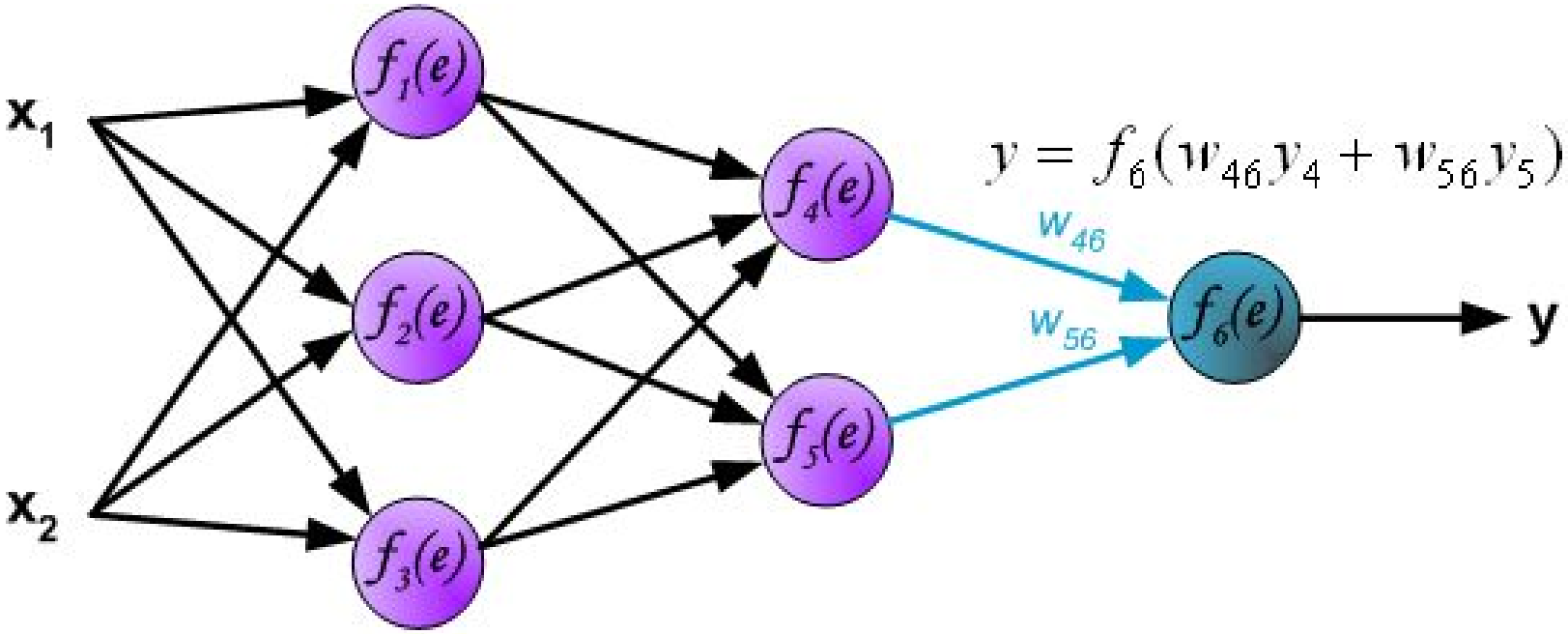
http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Forward propagation



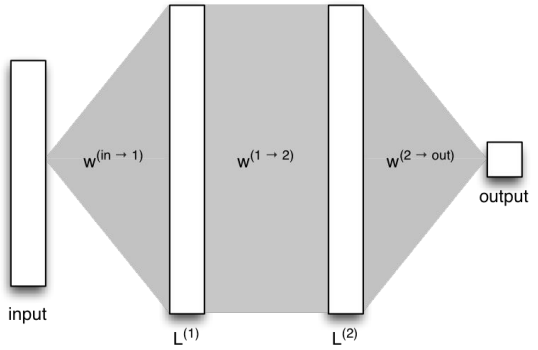
http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Forward propagation

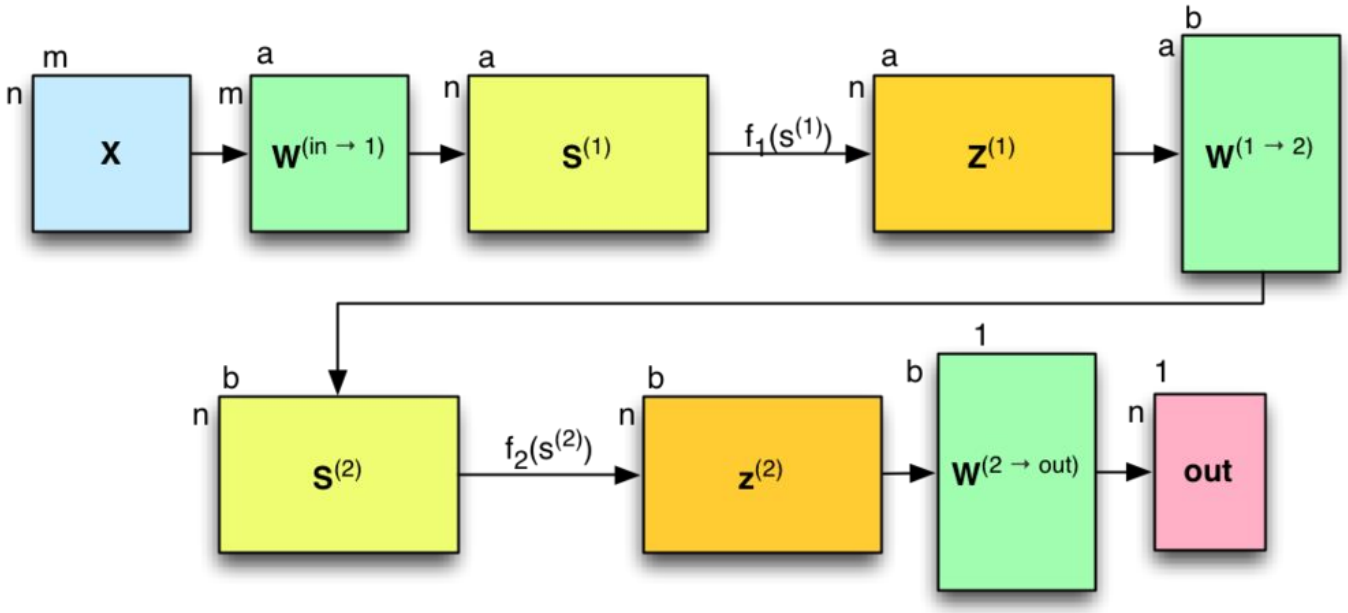


http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Forward propagation: matrix form



В реальности код векторизуется и вычисляется с помощью матричных операций:



$$S^{(1)} = XW^{(in \to 1)}$$
$$Z^{(1)} = f_1(S^{(1)})$$
$$S^{(2)} = Z^{(1)}W^{(1 \to 2)}$$
$$Z^{(2)} = f_2(S^{(2)})$$
$$\hat{y} = f_{out}\left(Z^{(2)}W^{(2 \to out)}\right)$$

http://briandolhansky.com/blog/2014/10/30/artificial-neural-networks-matrix-form-part-5

Механизмы работы
2. Backward propagation

Оптимизационная задача

Цель обучения: например, научиться хорошо классифицировать объекты (задача обучения с учителем, классификация) или предсказывать значения переменной (задача обучения с учителем, регрессия)

Задача обучения: минимизировать некую функцию потерь (Loss function, Cost function, Objective), которая отражает близость к идеалу классификации, регрессии или иной задачи.

С помощью хитрого выбора функции потерь можно решать довольно сложные задачи. В будущем увидим это на примере CTC в seq2seq или на примере переноса стиля изображения.

Функции потерь

В зависимости от типа задачи выбирается архитектура нейросети и в частности функция потерь (http://cs231n.github.io/neural-networks-2/#losses)

Примеры для классификации (http://cs231n.github.io/linear-classify/#loss)

- Multi-class SVM (hinge) loss
$$L_i = \sum_{j \neq y_i} \max(0, f_j - f_{y_i} + 1)$$
http://www.pyimagesearch.com/2016/09/05/multi-class-svm-loss/

- Multi-class squared SVM (hinge) loss
$$L_i = \sum_{j \neq y_i} max(0, s_j - s_{y_i} + 1)^2$$

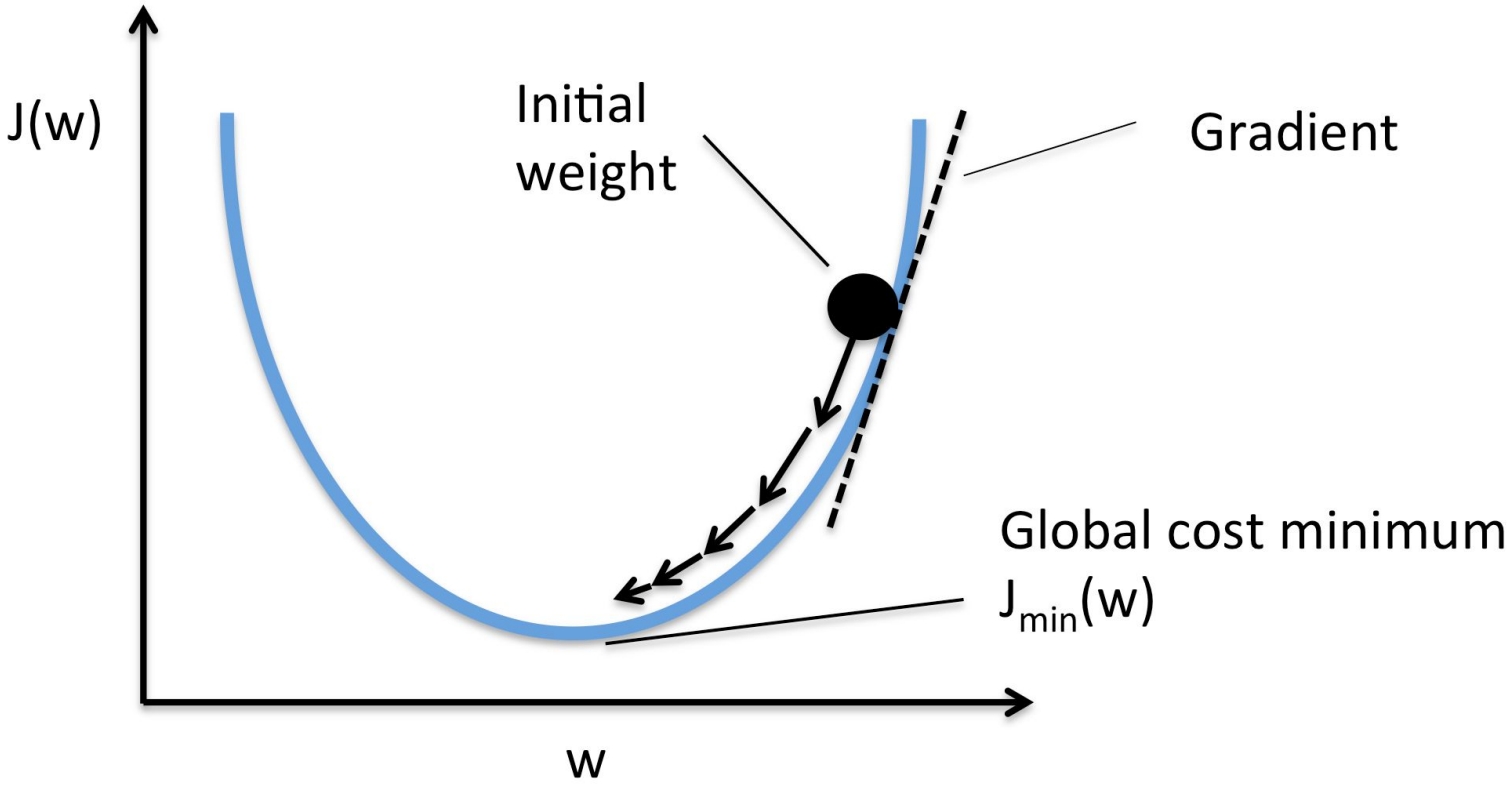
- Multi-class cross-entropy (softmax) loss
$$L_i = -\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right)$$
http://www.pyimagesearch.com/2016/09/12/softmax-classifiers-explained/

К функции потерь может добавляться регуляризация (L1/L2) для предотвращения переобучения.

Градиентный спуск

Градиентный спуск — это способ оптимизации дифференцируемых функций.

Идея: если мы можем определить направление наибольшего роста функции потерь (а это и есть градиент этой функции), то, двигаясь в противоположном направлении, мы уменьшим эту функцию (что приблизит нас к идеалу)



Градиентный спуск

Обычный градиентный спуск (GD или batch gradient descent) подразумевает расчёт градиента по полному датасету. Это довольно дорогая процедура, которая приводит к медленной сходимости алгоритма.

Есть несколько разновидностей градиентного спуска:

- **Stochastic gradient descent (SGD, стохастический градиентный спуск)**: обновления происходят на каждом обучающем примере.
- **Mini-batch gradient descent (MB-GD, градиентный спуск с мини-батчами)**: обновления происходят по некоторому числу обучающих примеров. Промежуточный вариант между обычным и стохастическим градиентным спуском.

http://sebastianraschka.com/faq/docs/closed-form-vs-gd.html

Общий вид алгоритма градиентного спуска

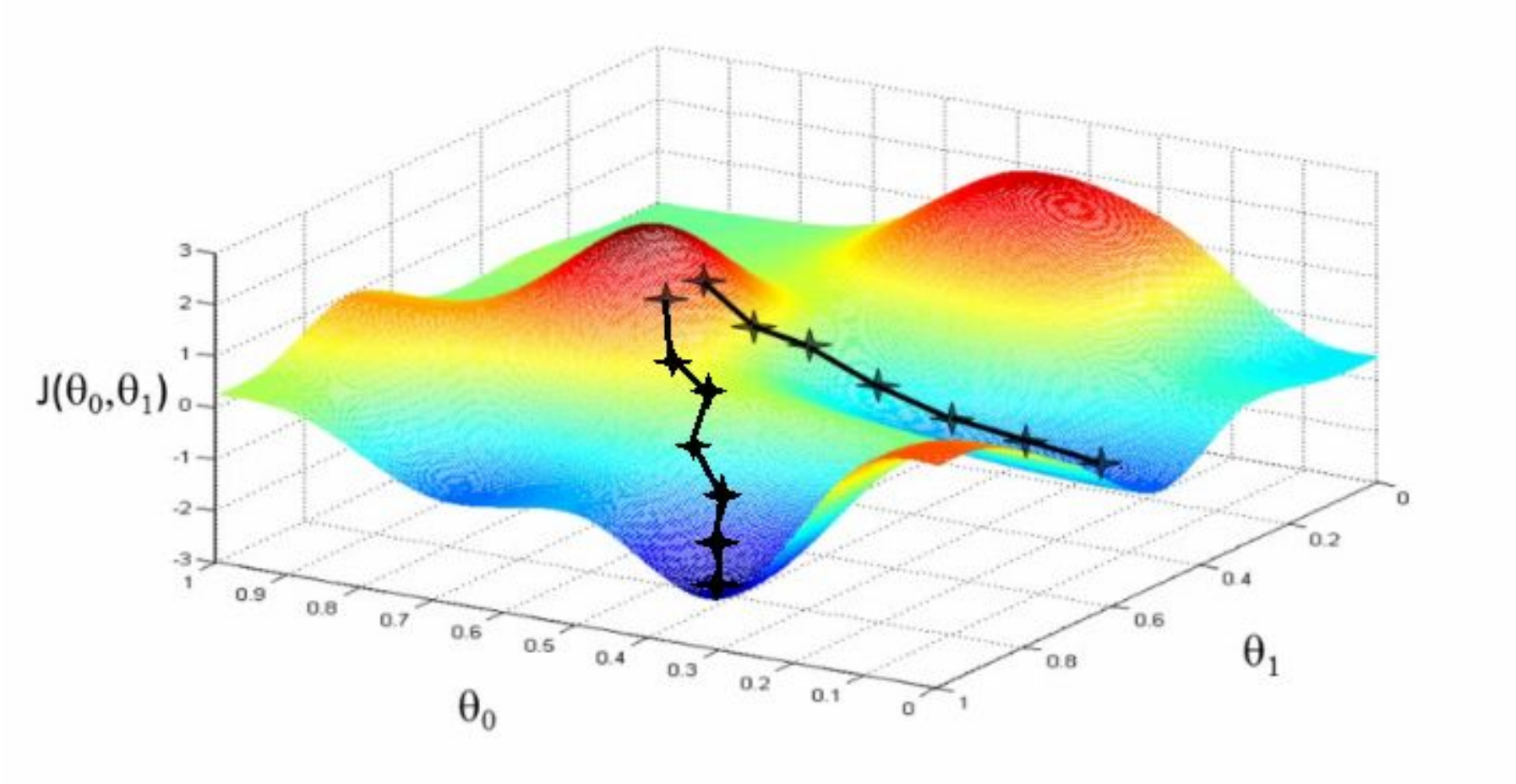
- Задаём параметр η (скорость обучения, learning rate)
- Инициализируем веса нейросети W
- Повторяем пока не достигнут критерий остановки:
  - Получаем мини-батч обучающих примеров <X,y>
  - Вычисляем градиент функции потерь g
  - Корректируем веса $W = W - \eta * g$

Правильный выбор скорости обучения очень важен.

http://www.deeplearningbook.org/contents/optimization.html

Градиентный спуск

В случае нейросетей ландшафт функции потерь очень сложный, поэтому с градиентным спуском много тонкостей.

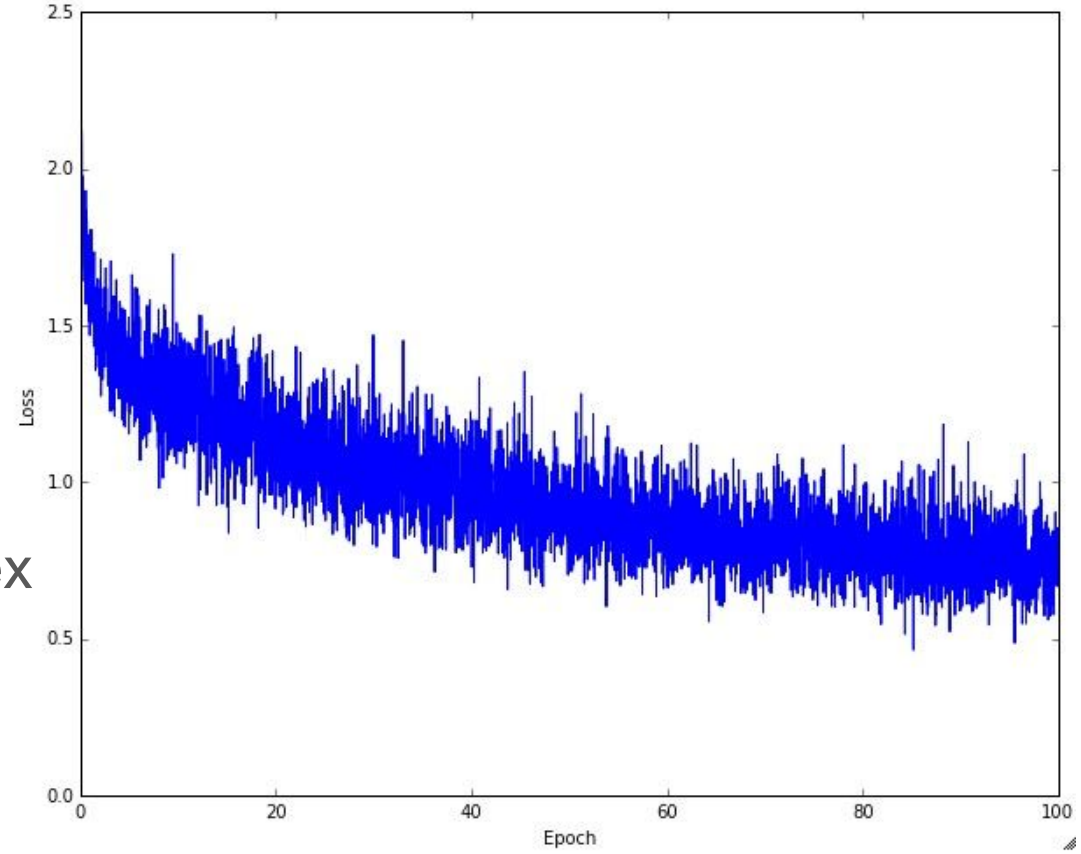


http://blog.datumbox.com/tuning-the-learning-rate-in-gradient-descent/

Градиентный спуск

Реальный пример уменьшения со временем функции потерь при обучении нейросети градиентным спуском.

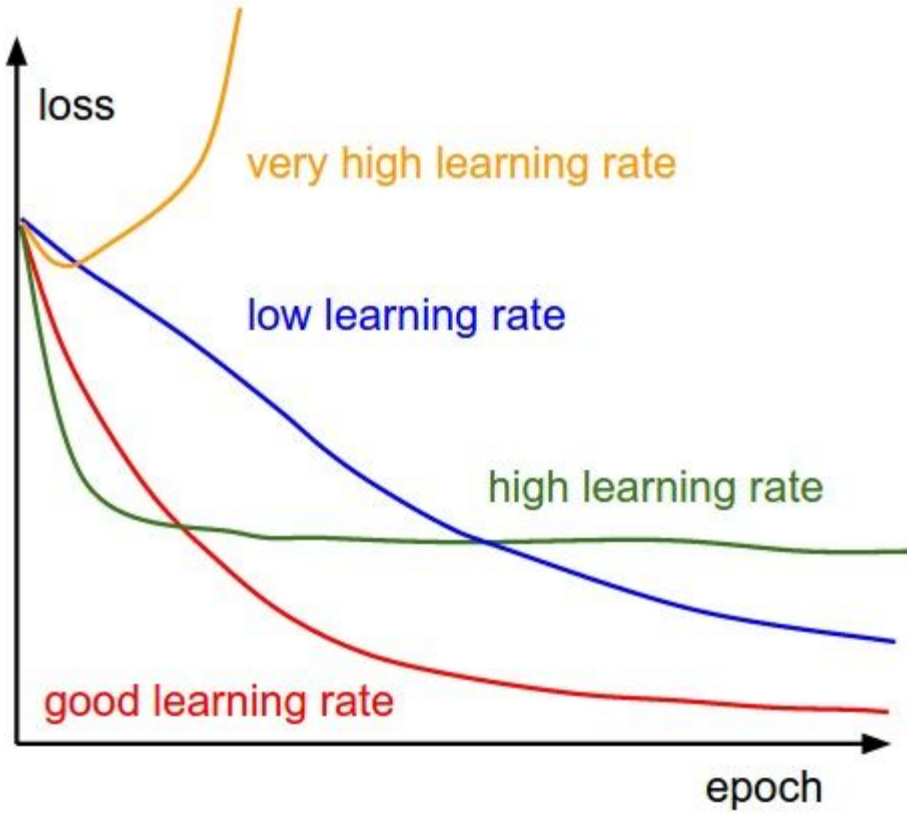
Эпоха = один полный цикл обучения нейросети на всех примерах обучающей выборки.



http://cs231n.github.io/neural-networks-3/

Градиентный спуск

Важность правильного выбора параметра скорости обучения (learning rate) и влияние его на скорость обучения нейросети.

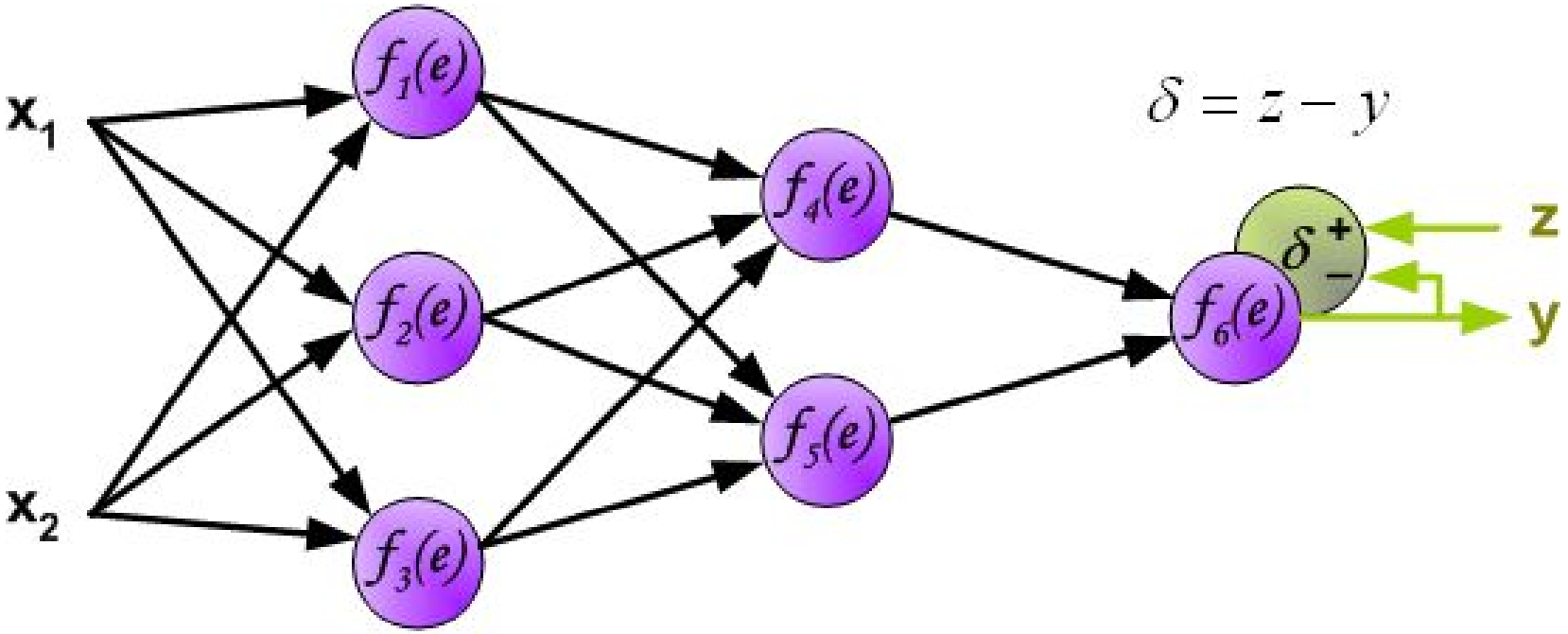


http://cs231n.github.io/neural-networks-3/

Алгоритм back-propagation

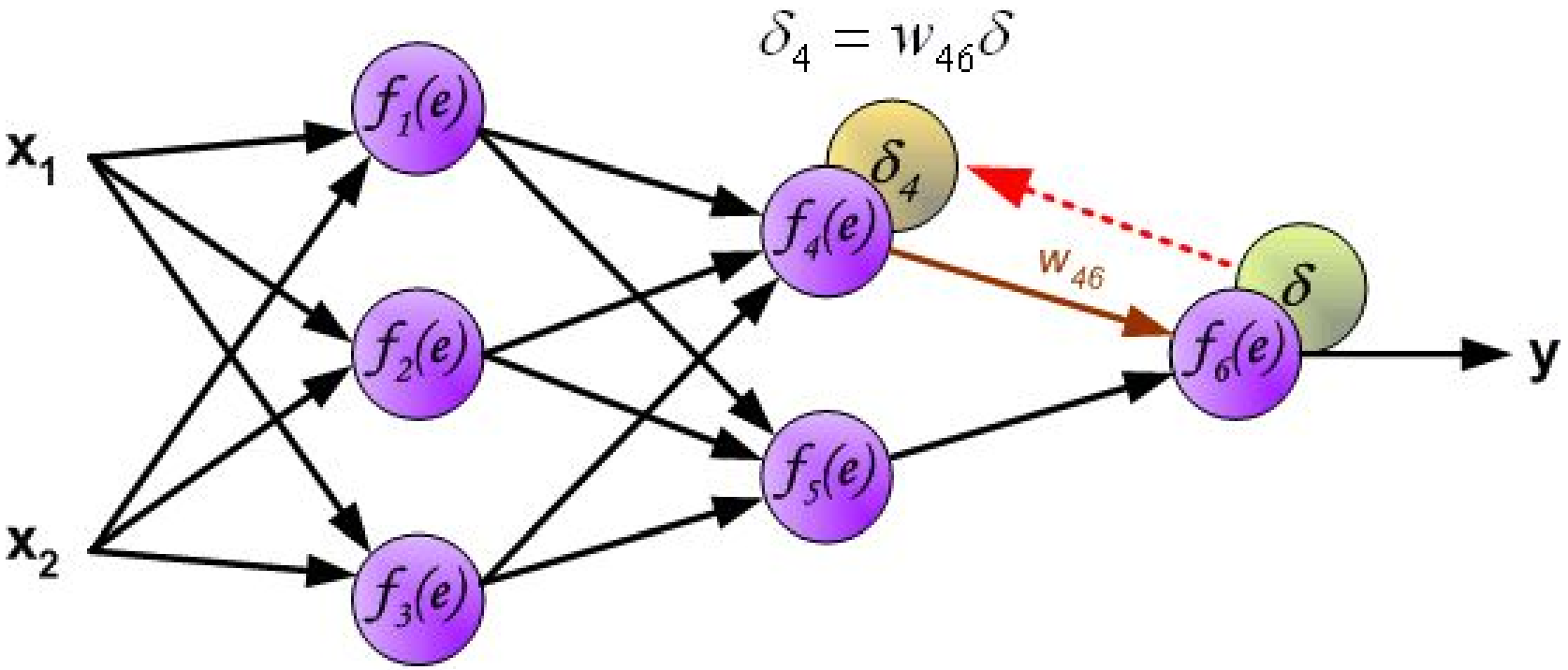
Алгоритм обратного распространения ошибки (back-propagation, backprop) — это способ расчёта градиента ошибки по каждому из весов в нейросети, чтобы затем каждый вес скорректировать в соответствии с алгоритмом градиентного спуска.

Backward propagation



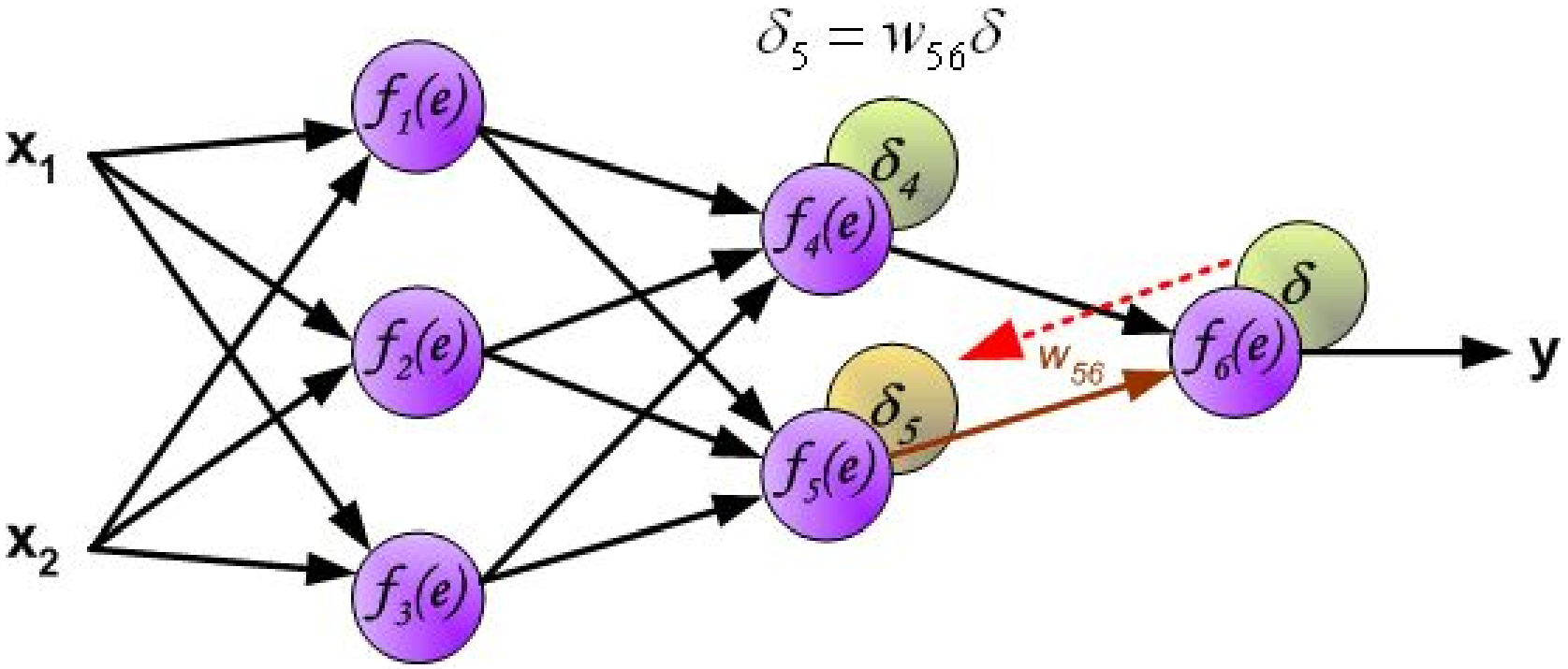
http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Backward propagation



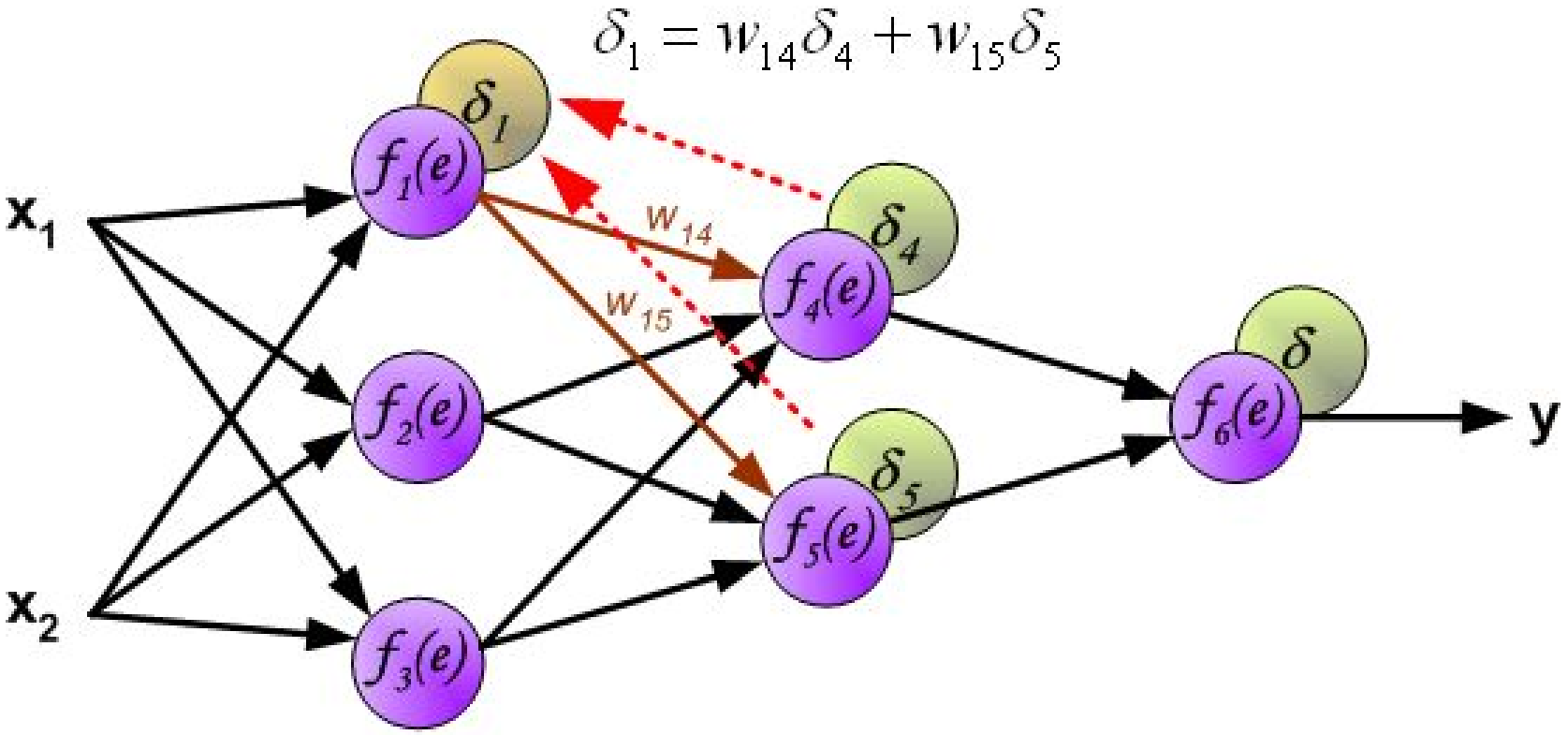
http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Backward propagation



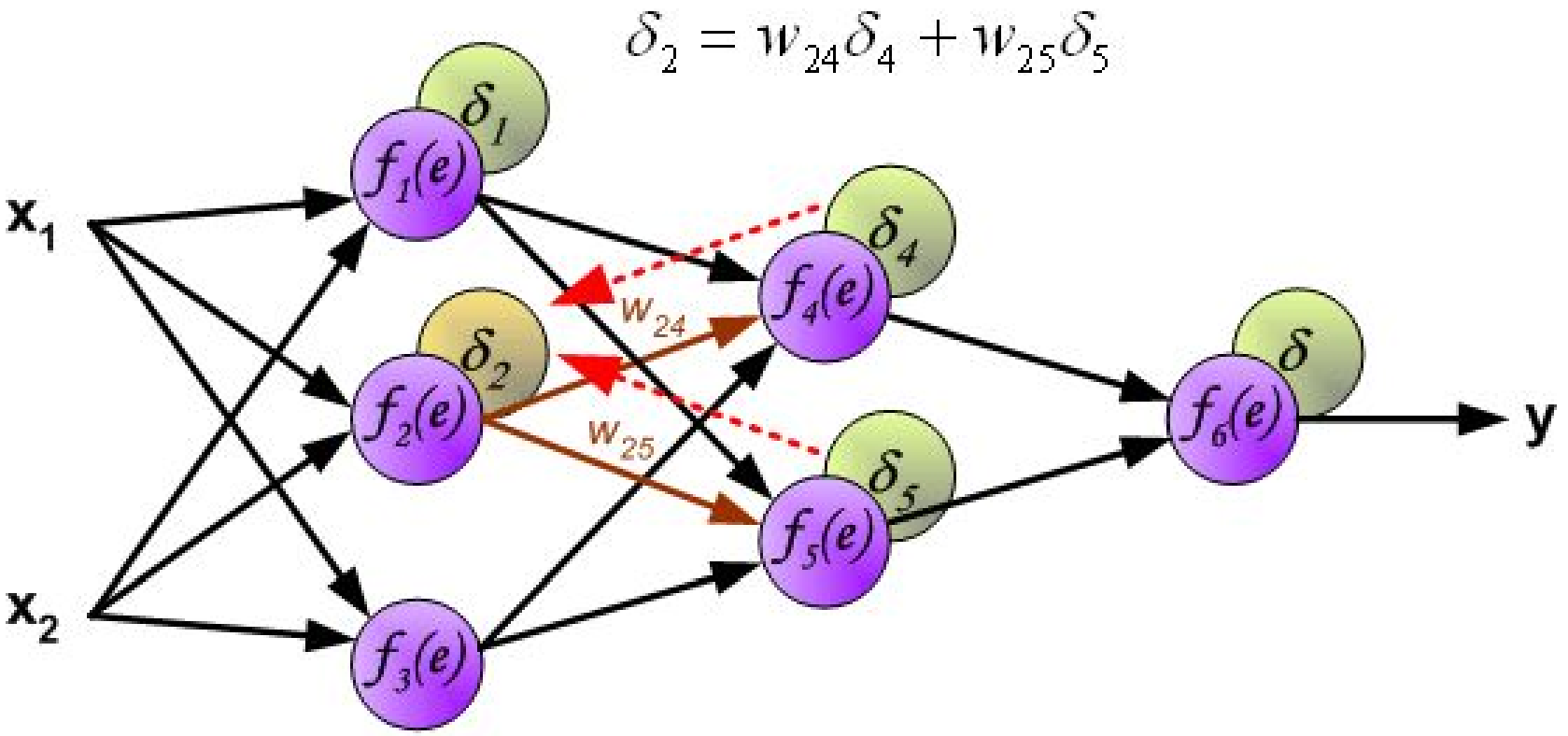
http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Backward propagation



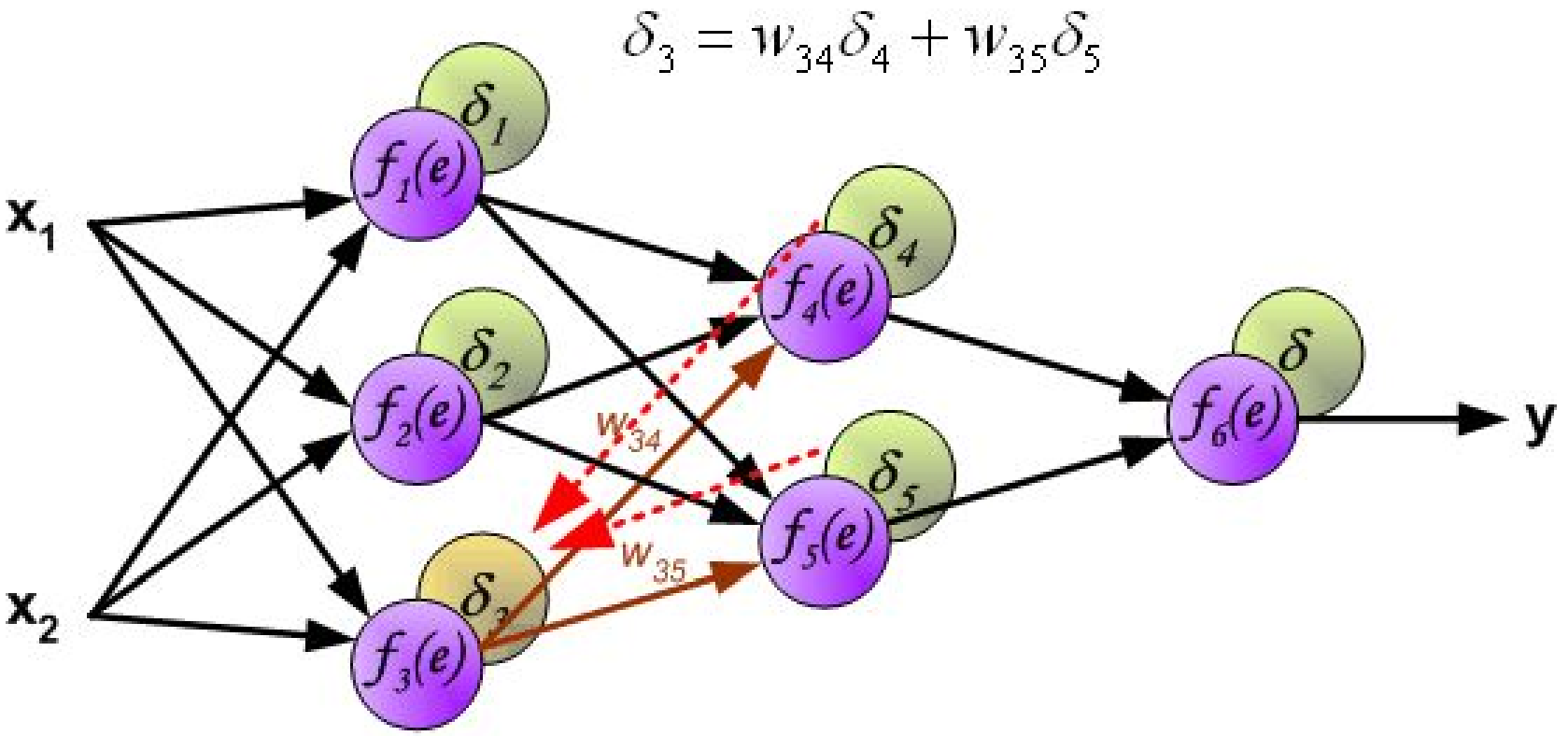
http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Backward propagation



http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Backward propagation

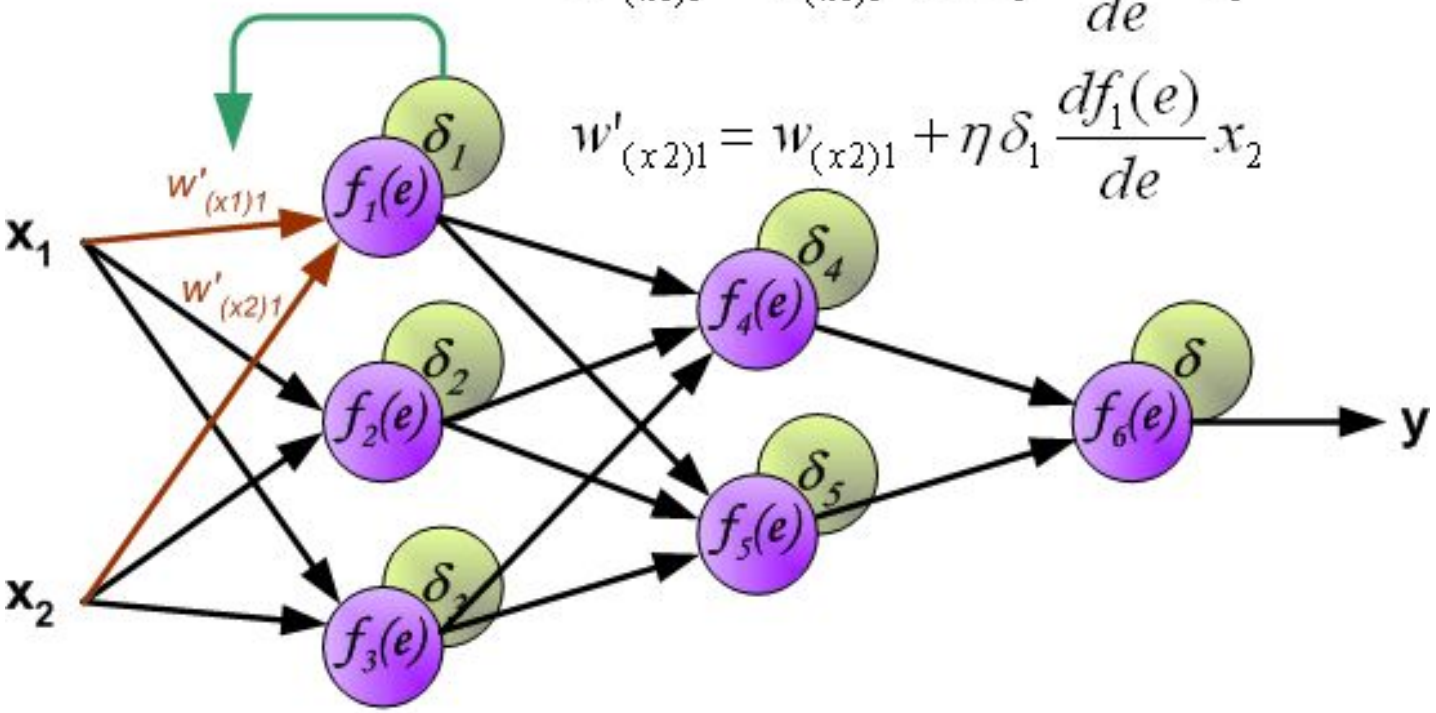


http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Backward propagation

$$w'_{(x1)1} = w_{(x1)1} + \eta \delta_1 \frac{df_1(e)}{de} x_1$$

$$w'_{(x2)1} = w_{(x2)1} + \eta \delta_1 \frac{df_1(e)}{de} x_2$$

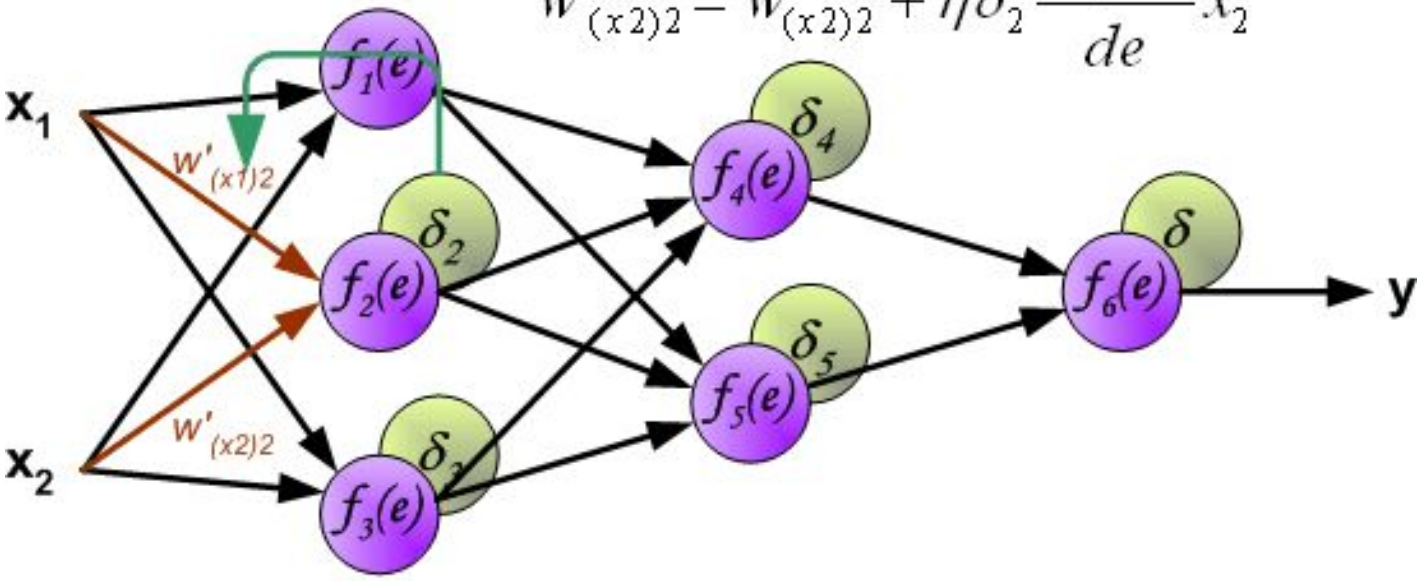


http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Backward propagation

$$w'_{(x1)2} = w_{(x1)2} + \eta \delta_2 \frac{df_2(e)}{de} x_1$$

$$w'_{(x2)2} = w_{(x2)2} + \eta \delta_2 \frac{df_2(e)}{de} x_2$$

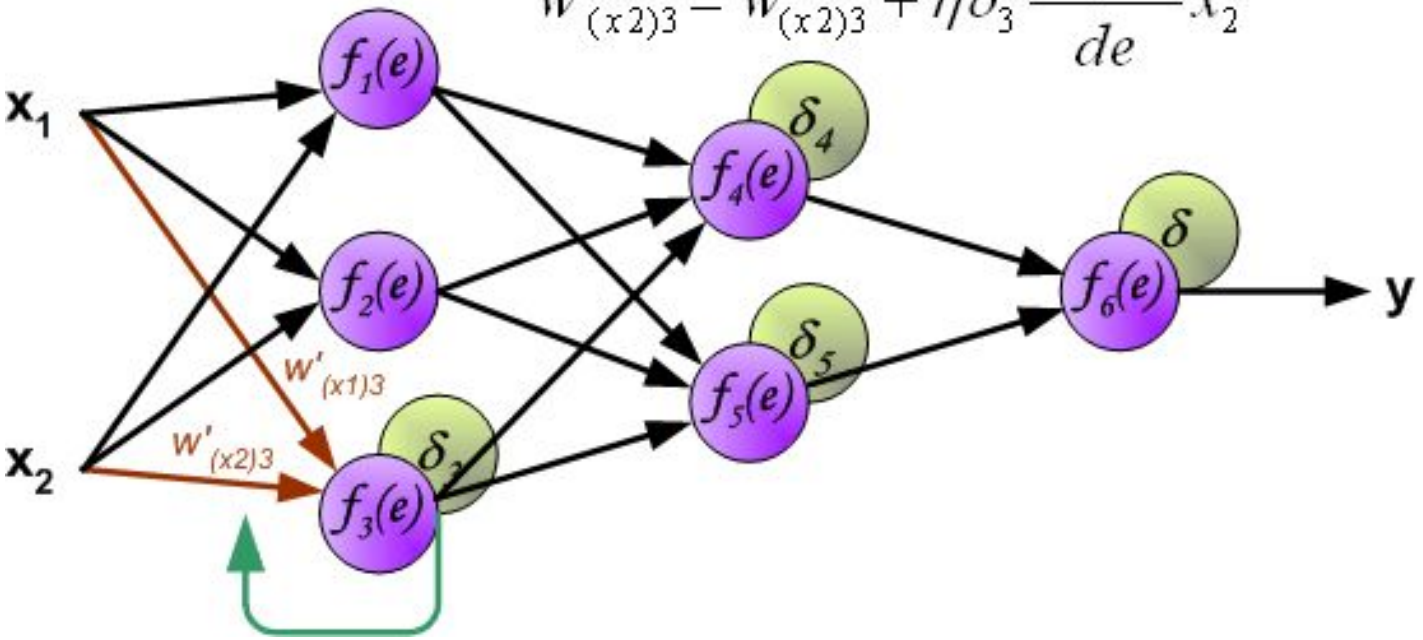


http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Backward propagation

$$w'_{(x1)3} = w_{(x1)3} + \eta \delta_3 \frac{df_3(e)}{de} x_1$$

$$w'_{(x2)3} = w_{(x2)3} + \eta \delta_3 \frac{df_3(e)}{de} x_2$$



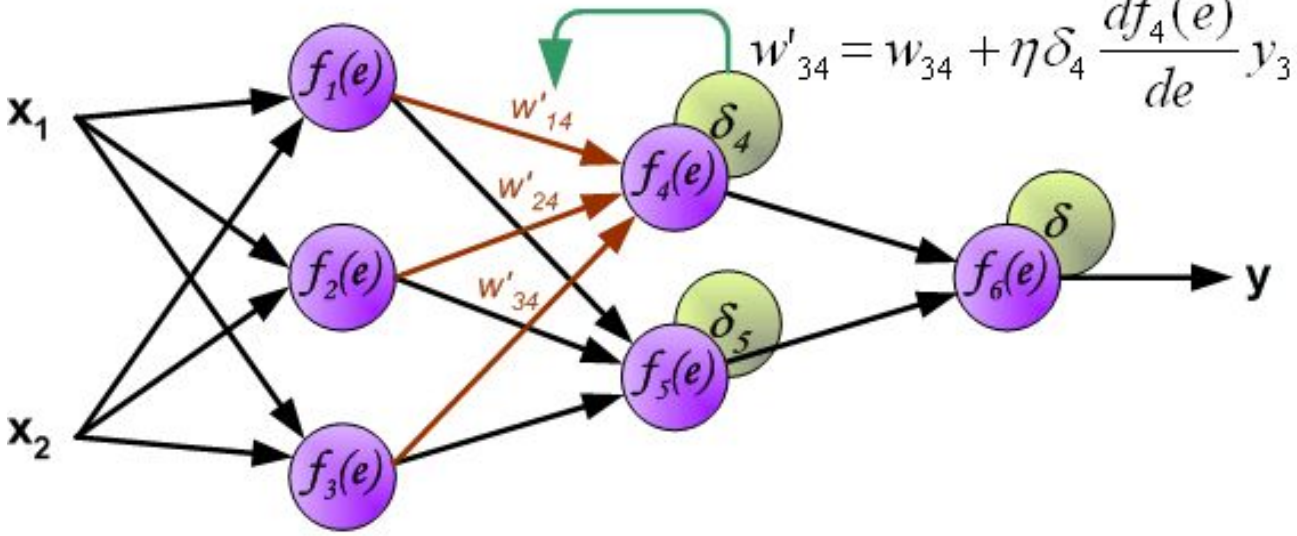
http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Backward propagation

$$w'_{14} = w_{14} + \eta\delta_4 \frac{df_4(e)}{de} y_1$$

$$w'_{24} = w_{24} + \eta\delta_4 \frac{df_4(e)}{de} y_2$$

$$w'_{34} = w_{34} + \eta\delta_4 \frac{df_4(e)}{de} y_3$$



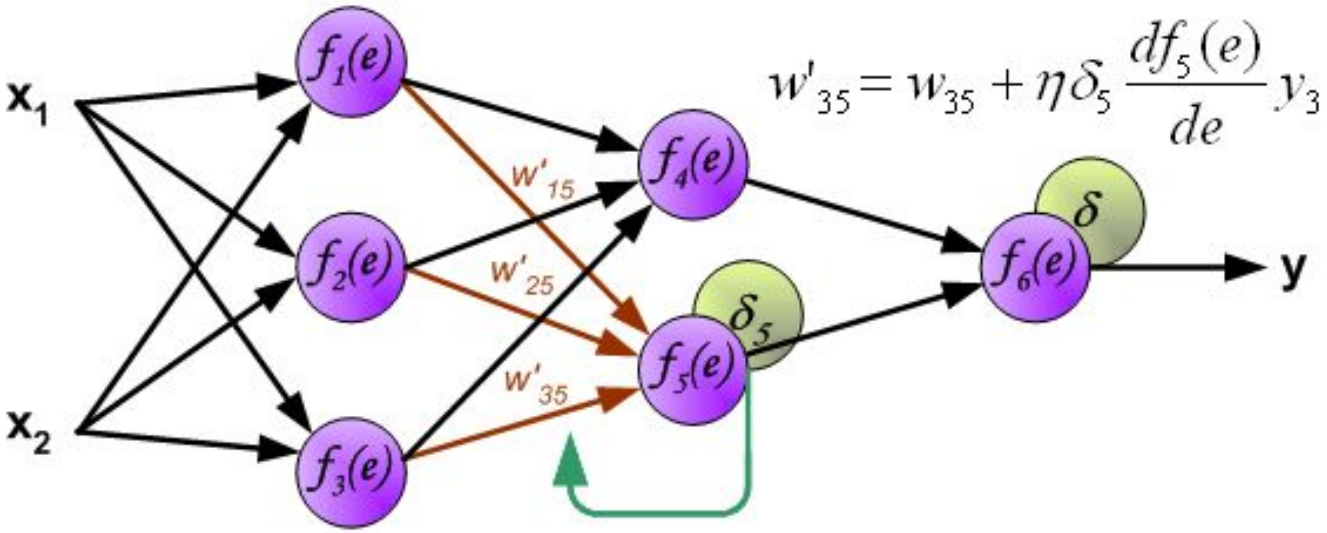
http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Backward propagation

$$w'_{15} = w_{15} + \eta \delta_5 \frac{df_5(e)}{de} y_1$$

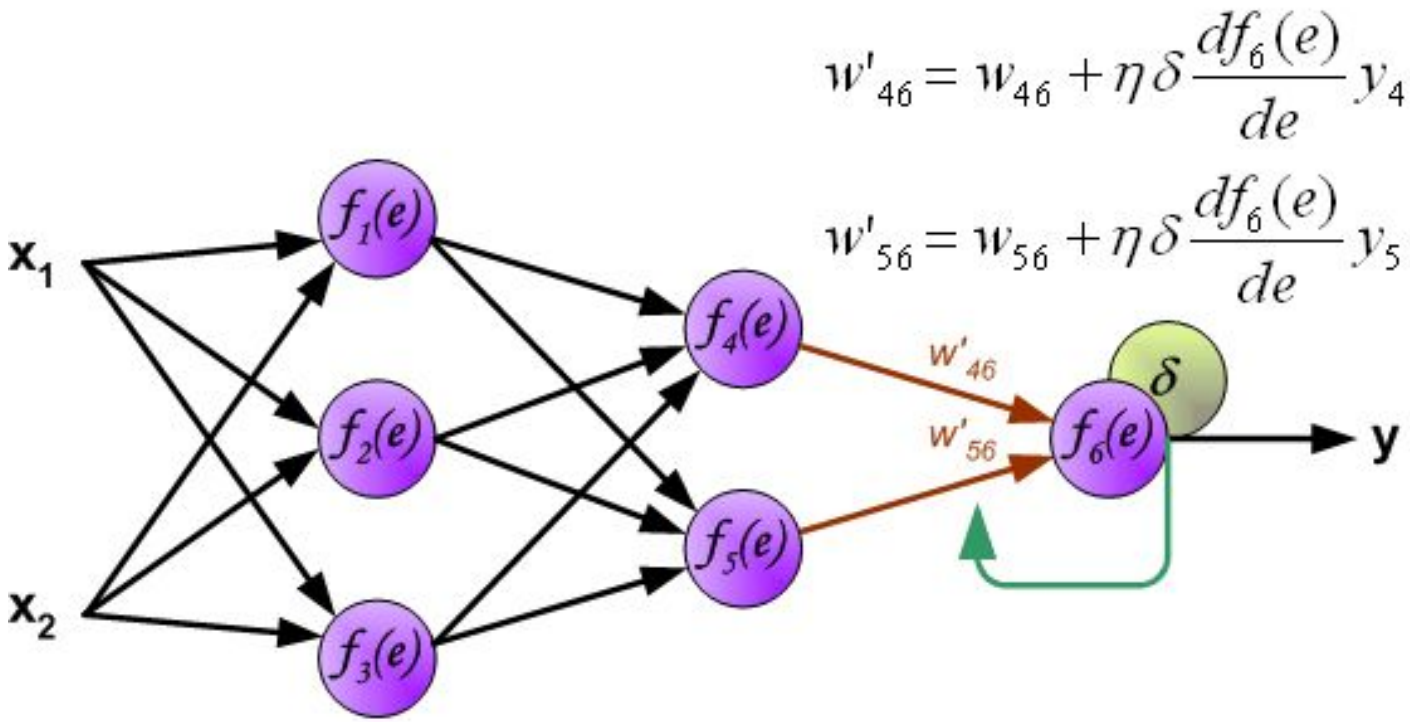
$$w'_{25} = w_{25} + \eta \delta_5 \frac{df_5(e)}{de} y_2$$

$$w'_{35} = w_{35} + \eta \delta_5 \frac{df_5(e)}{de} y_3$$



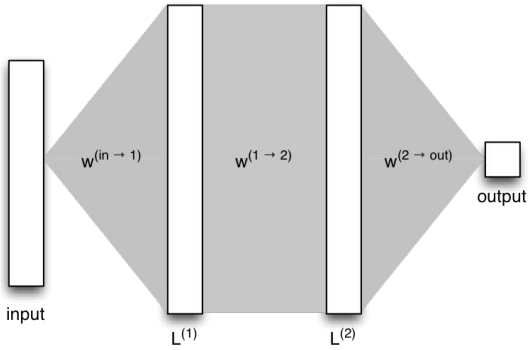
http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Backward propagation



http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50321873

Backward propagation: matrix form

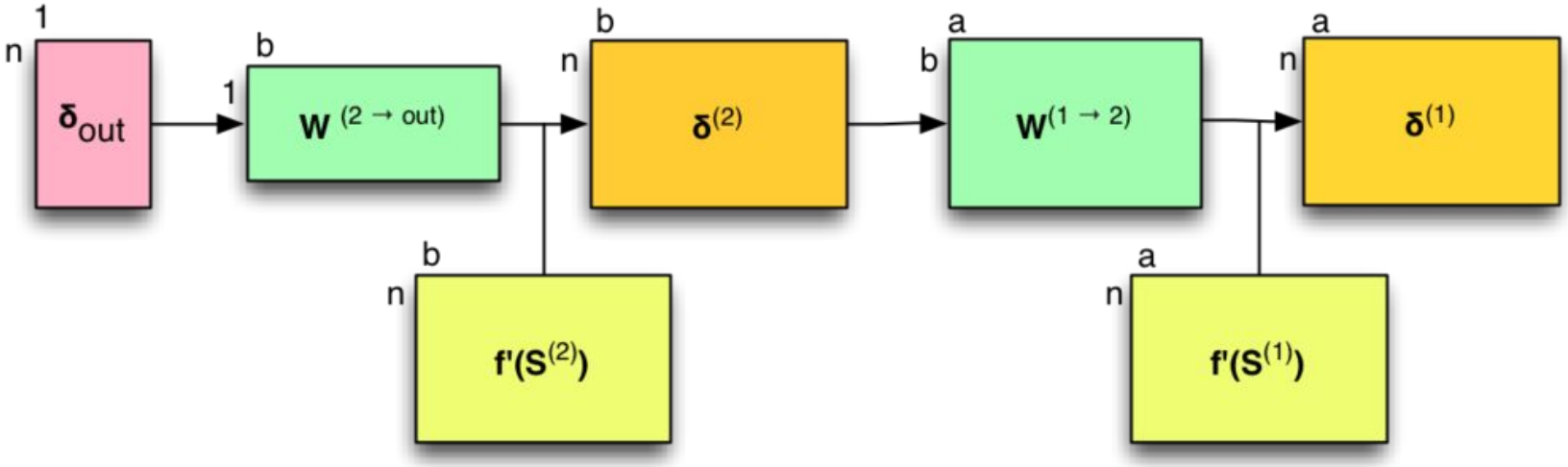


$$D^{(out)} = (Z^{(out)} - Y)^T$$

$$D^{(i)} = F^{(i)} \odot W^{(i)} D^{(j)}$$

$$\Delta W^{(in \to 1)} = -\eta (D^{(1)} X)^T$$

$$\Delta W^{(i \to j)} = -\eta (D^{(j)} Z^{(i)})^T$$



http://briandolhansky.com/blog/2014/10/30/artificial-neural-networks-matrix-form-part-5

Дифференцирование

Ещё несколько лет назад программирование нейросети заключалось в ручном задании всех слоёв, функций активации и функции потерь, аналитического расчёта их производных, а также проверки корректности этих производных с помощью численного дифференцирования. Эта процедура была подвержена ошибкам и имела высокий порог входа.

В последние годы необходимости в ручном задании производных нет, так как большинство библиотек поддерживают один из двух вариантов автоматизации этого процесса:

- Символьное дифференцирование (Theano)
- Автоматическое дифференцирование (Tensorflow, Torch)

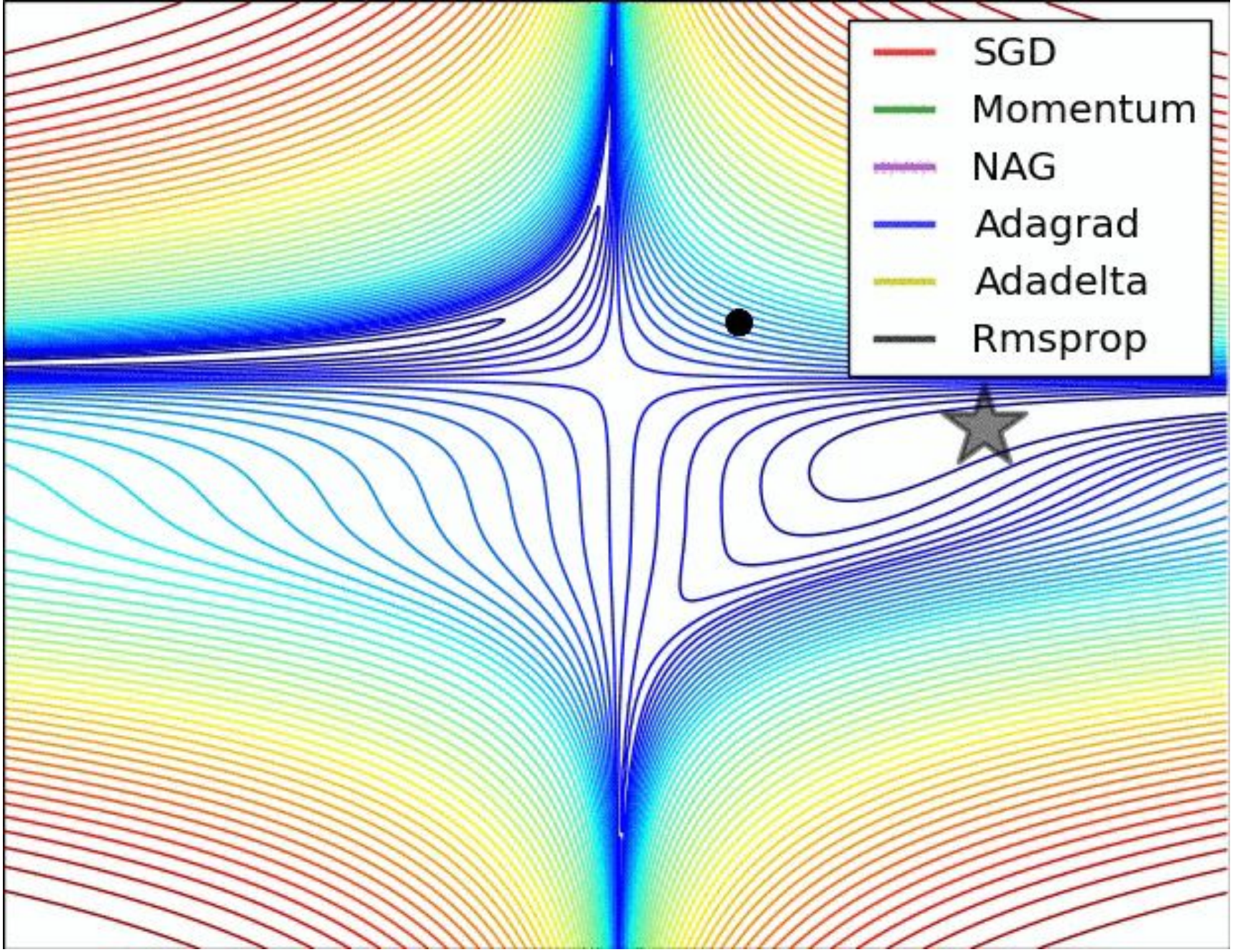
https://domluna.me/automatic-differentiation-a.k.a-please-save-me-from-backprop/

Вариации и улучшения backprop

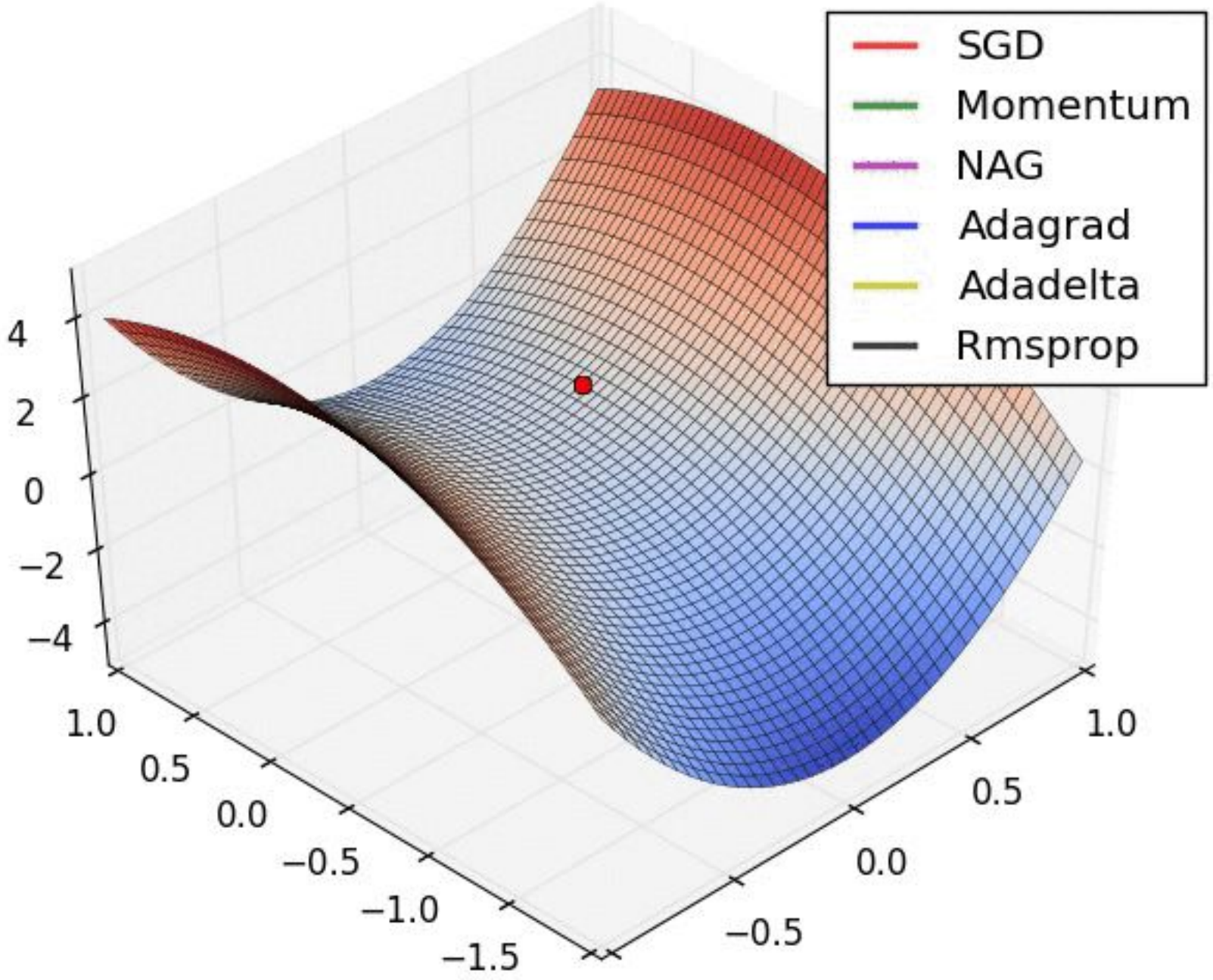
Существует множество модификаций backprop

- Изменяемый learning rate (постепенное уменьшение, уменьшение на порядок каждые N эпох, …)
- Использование момента инерции (momentum update, Nesterov momentum)
- Адаптивные методы (Adagrad, Adadelta, RMSprop, Adam, …)
- Методы второго порядка (L-BFGS, Conjugate gradient, Hessian Free)

http://cs231n.github.io/neural-networks-3/

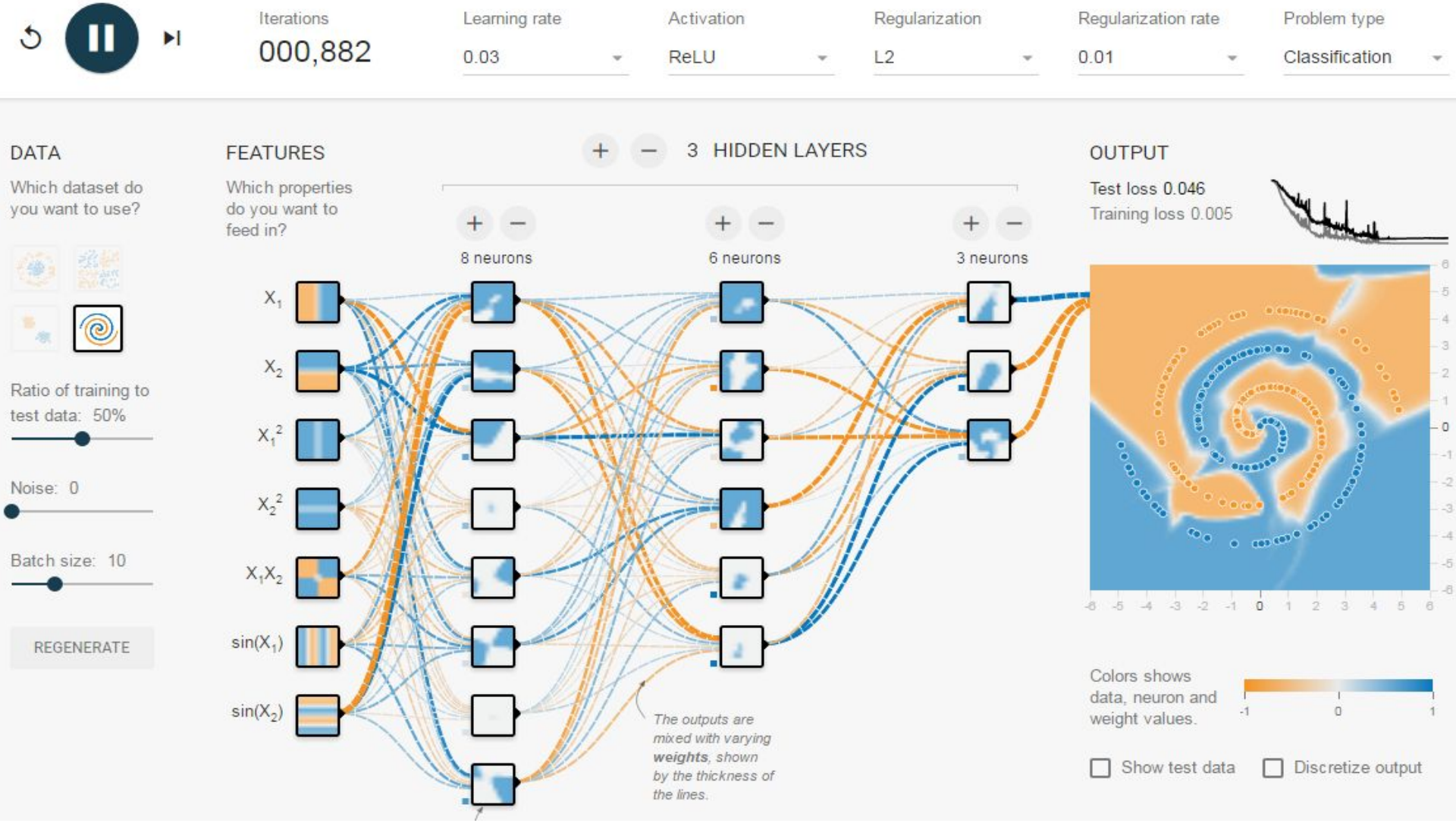


http://cs231n.github.io/neural-networks-3/



http://cs231n.github.io/neural-networks-3/

Демонстрация работы

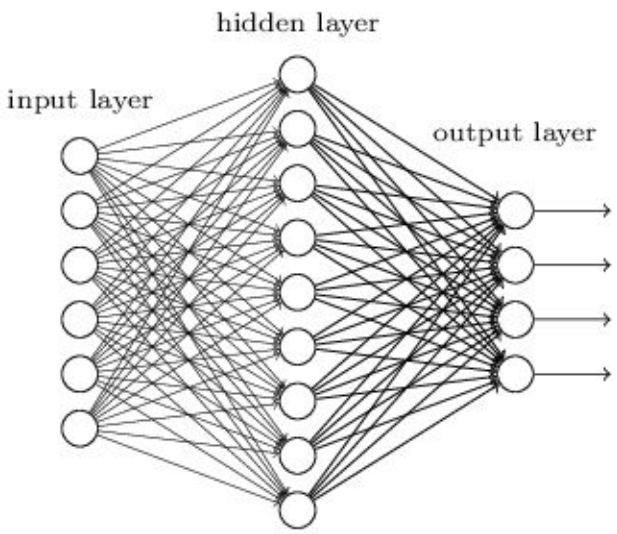


http://playground.tensorflow.org/

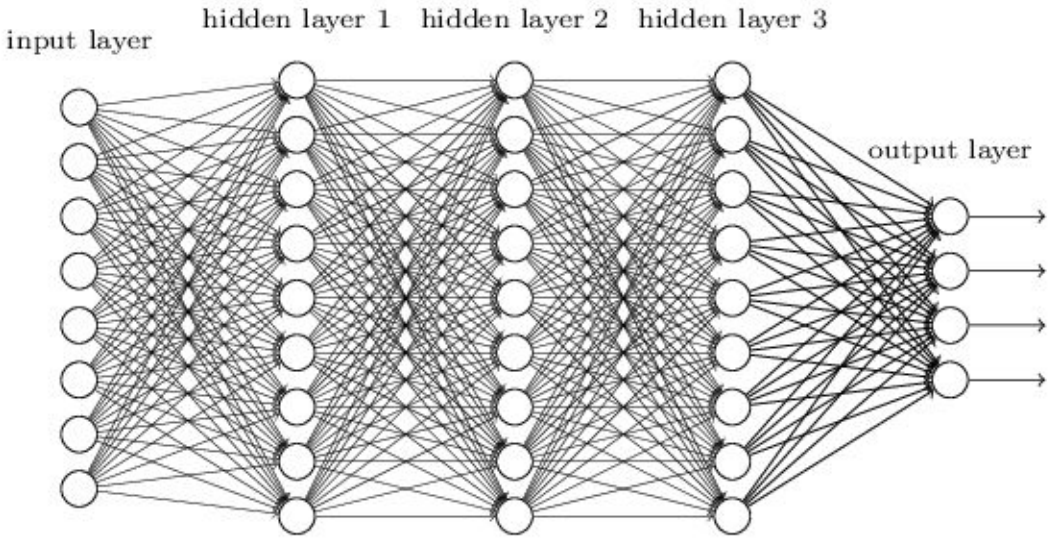
Классические нейросети прямого распространения (Feed-Forward Neural Networks, FNN)

Полносвязная нейросеть

"Non-deep" feedforward neural network



Deep neural network

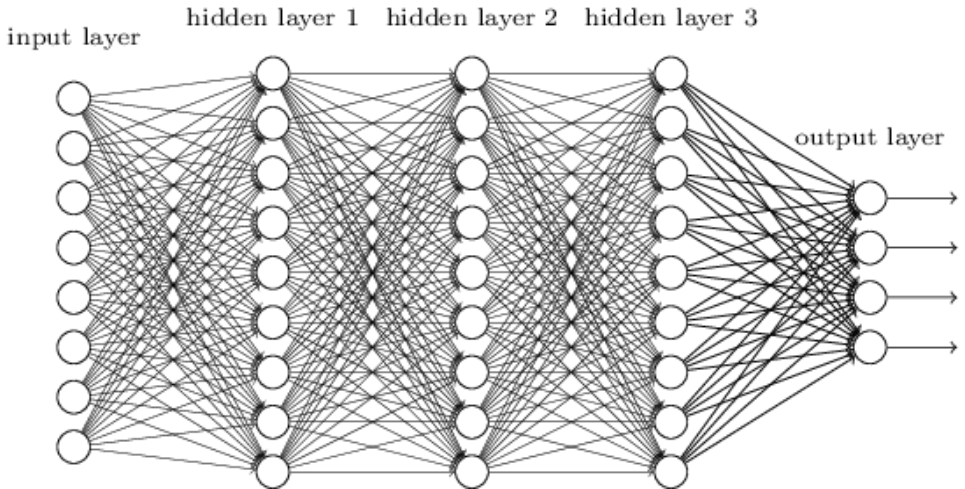


Multilayer Perceptron (MLP)

Классика FNN. Хорошо работают для классификации, но есть трудности:

- Много параметров
  - Для сети, у которой на входе картинка 100x100, три скрытых слоя по 100 нейронов каждый, и выходом на 10 классов, число параметров будет примерно 1M
    (10000*100 + 100*100 + 100*100 + 100*10)
- Затухающие градиенты (если слоёв много)

Как следствие — трудно обучать.



Пример с MLP

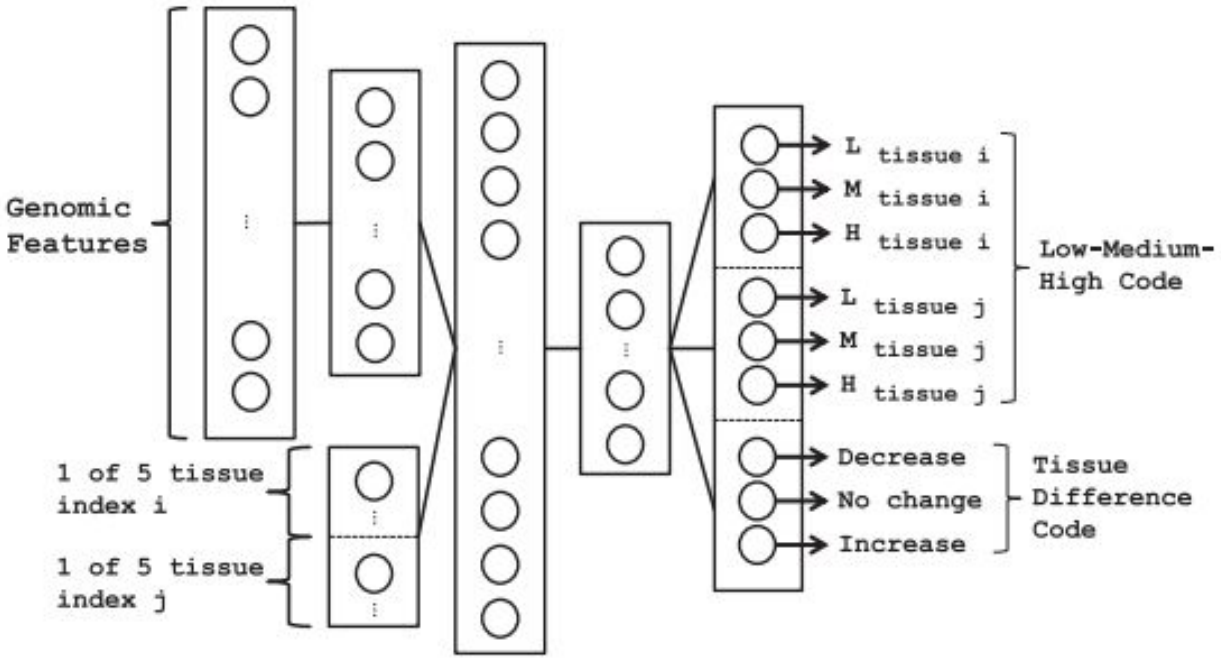


Fig. 1. Architecture of the DNN used to predict AS patterns. It contains three hidden layers, with hidden variables that jointly represent genomic features and cellular context (tissue types)

Deep learning of the tissue-regulated splicing code

http://www.psi.toronto.edu/publications/2014/DeepSplicingCode.pdf

Пример с MLP

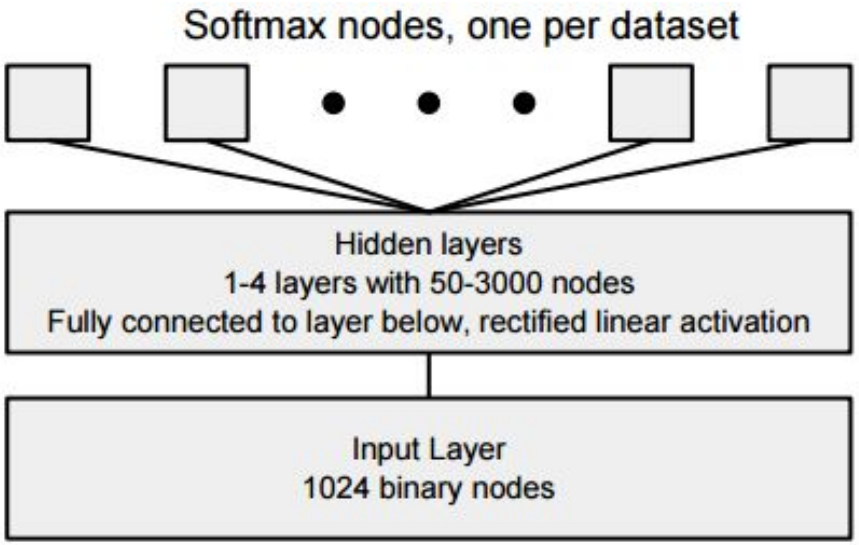
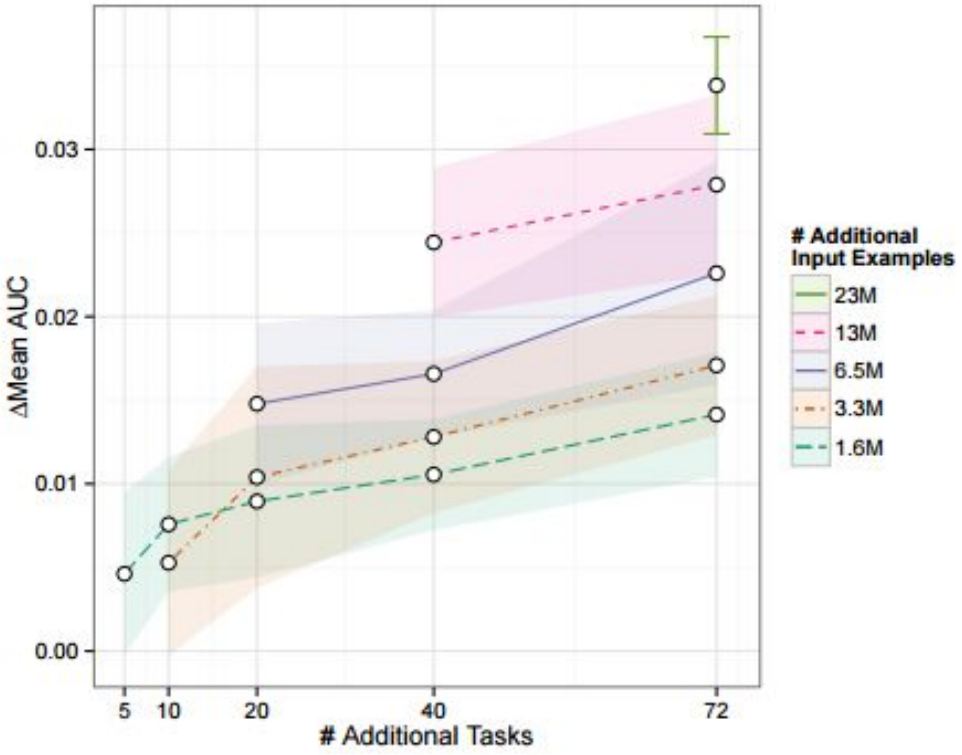


Figure 1. Multitask neural network.



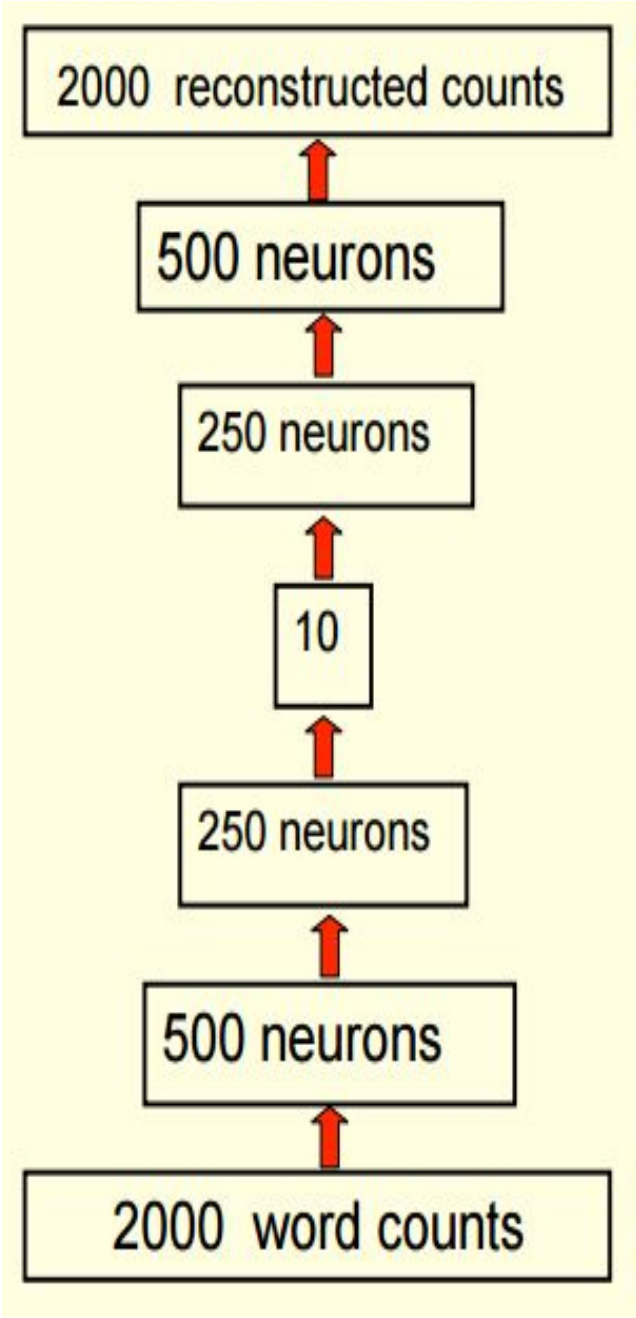
Massively Multitask Networks for Drug Discovery
https://arxiv.org/abs/1502.02072

Вариации FNN: Автоэнкодер (AE)

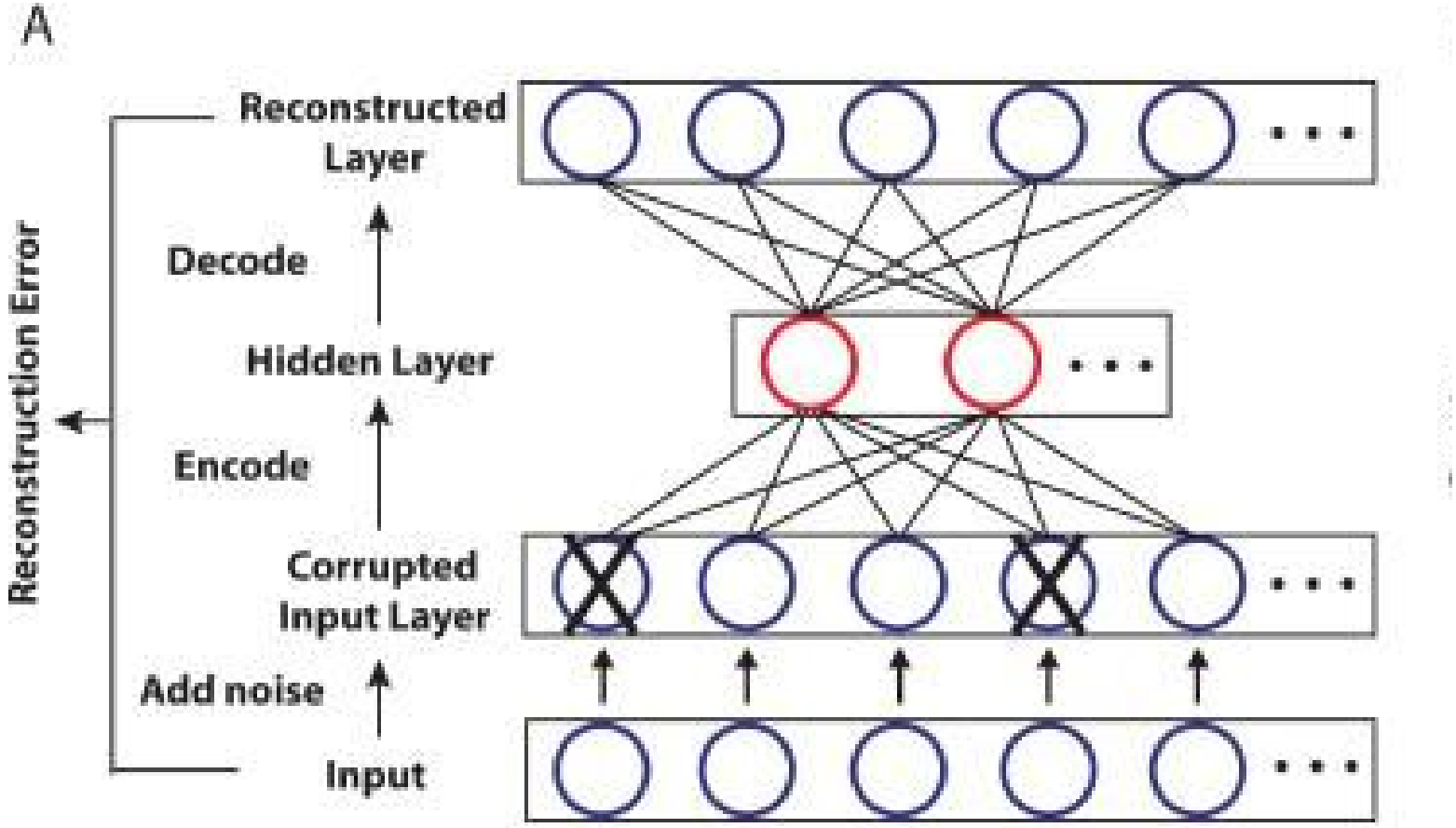
Учится создавать компактное описание входных данных.

Используется для уменьшения размерности и получения новых высокоуровневых признаков.

Может быть глубоким (многослойным).



Примеры с AE



UNSUPERVISED FEATURE CONSTRUCTION AND KNOWLEDGE EXTRACTION FROM GENOME-WIDE ASSAYS OF BREAST CANCER WITH DENOISING AUTOENCODERS, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4299935/

Примеры с AE

UNSUPERVISED FEATURE CONSTRUCTION AND KNOWLEDGE EXTRACTION FROM GENOME-WIDE ASSAYS OF BREAST CANCER WITH DENOISING AUTOENCODERS
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4299935/

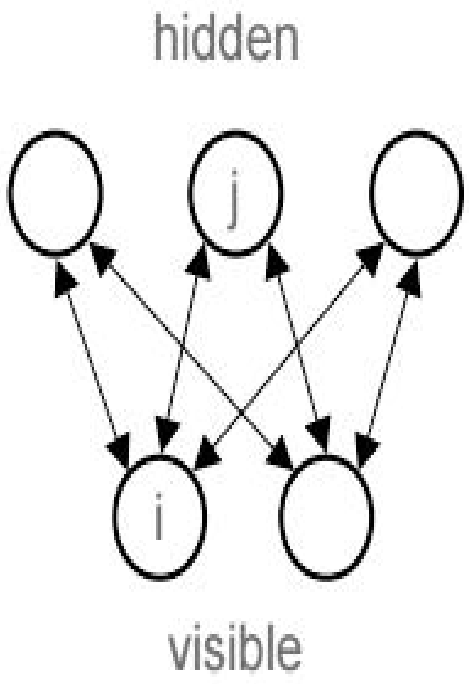
“We evaluate the performance of DAs by applying them to a large collection of breast cancer gene expression data. Results show that DAs successfully construct features that contain both clinical and molecular information. There are features that represent tumor or normal samples, estrogen receptor (ER) status, and molecular subtypes. Features constructed by the autoencoder generalize to an independent dataset collected using a distinct experimental platform.”

Learning structure in gene expression data using deep architectures, with an application to gene clustering
http://biorxiv.org/content/early/2015/11/16/031906

Вариации FNN: RBM

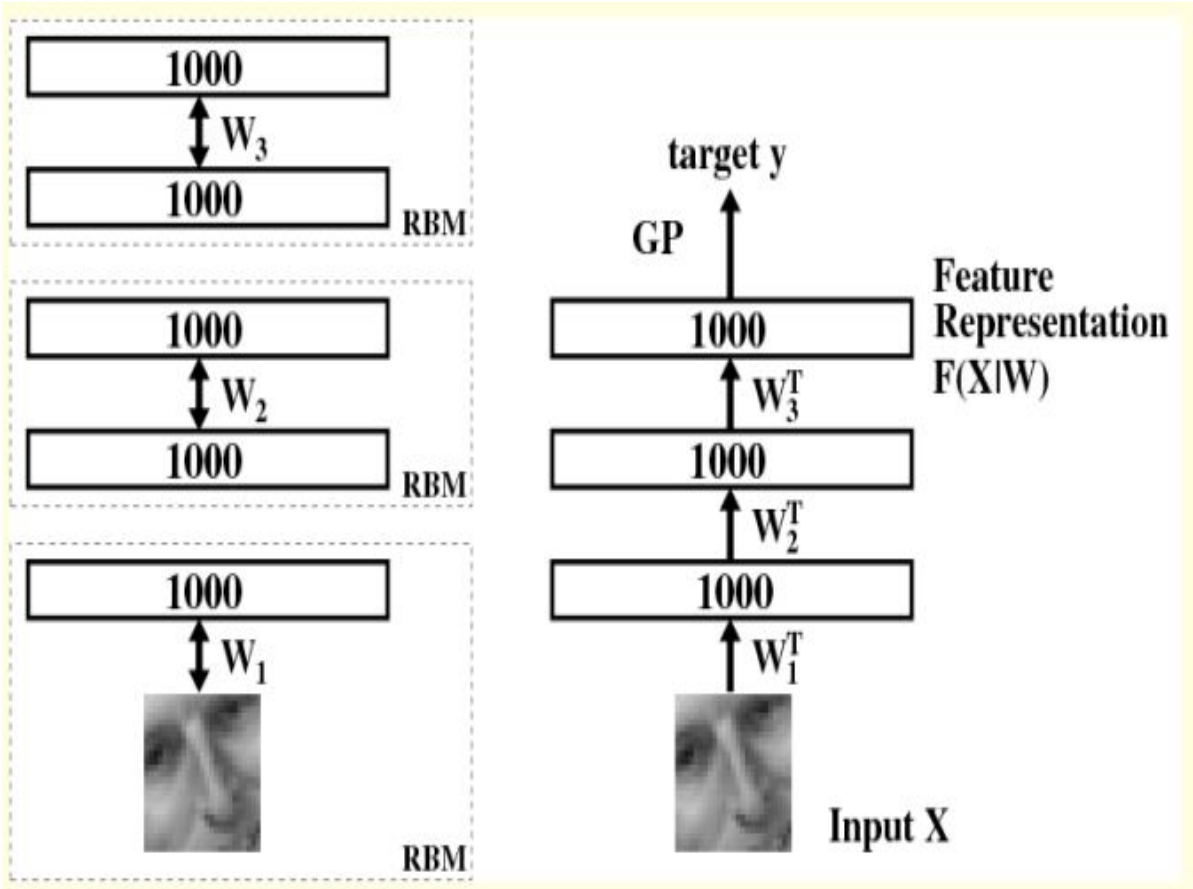
Restricted Boltzmann Machine (RBM)

Неглубокая генеративная модель, которая учится генерировать данные с заданными характеристиками. По факту очень похожа на автоэнкодер, но в отличие от автоэнкодера стохастическая.



Вариации FNN: DBN

Deep Belief Networks (DBN) — фактически способ обучения глубоких сетей, при котором каждый уровень сети учится как отдельная RBM.

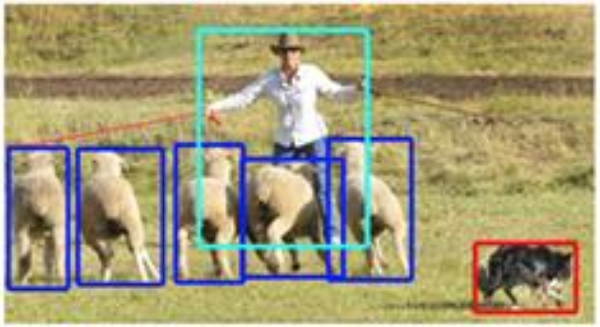


Свёрточные нейросети

Convolutional Neural Networks, CNN

Классические задачи для CNN

(a) classification



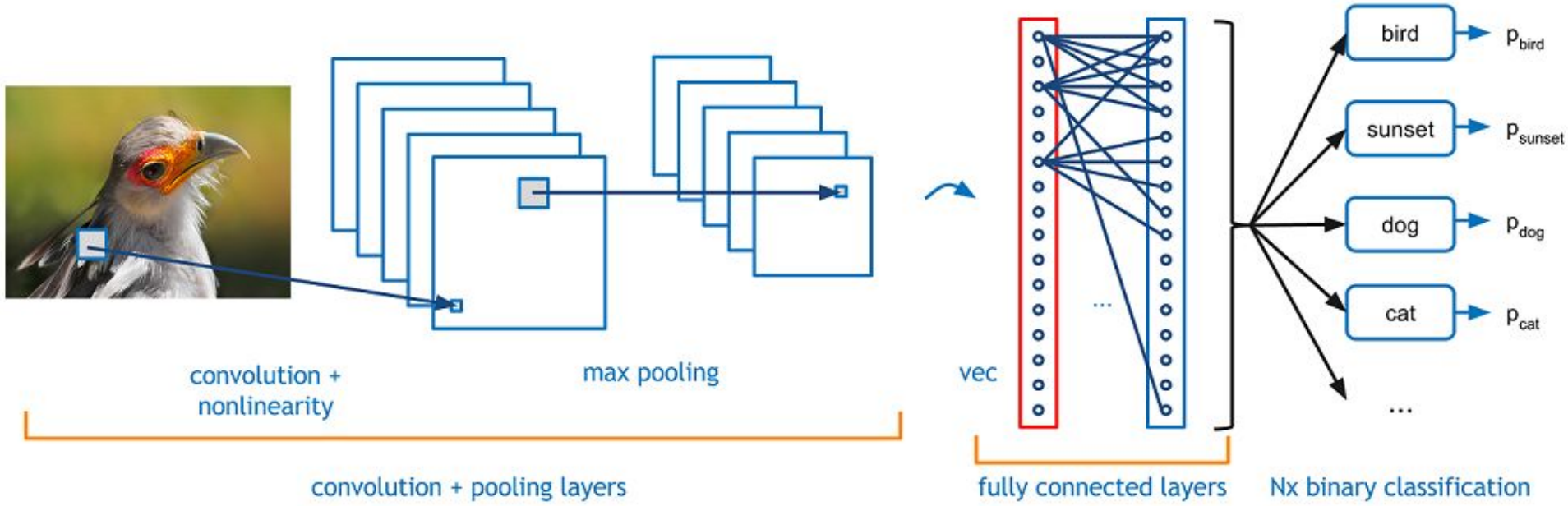
(b) detection

(c) segmentation

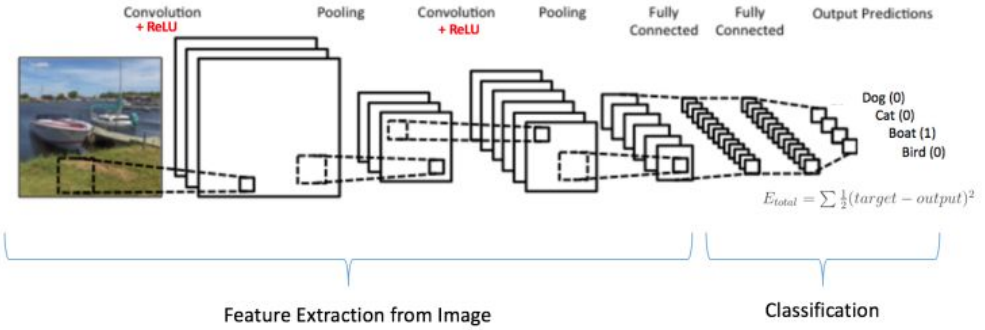
https://research.facebook.com/blog/learning-to-segment/

Свёрточная нейросеть: общий вид

Свёрточная нейросеть (CNN) — это Feed-Forward сеть специального вида:



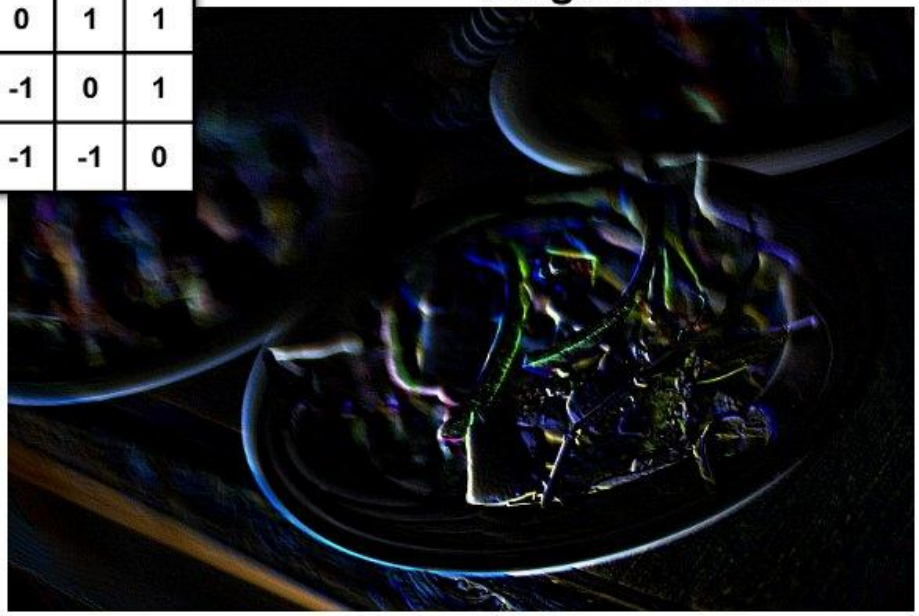
Состав CNN



- **Свёрточные слои**: каждая плоскость в свёрточном слое — это один нейрон, реализующий операцию свёртки (convolution) и являющийся матричным фильтром небольшого размера (например, 5x5).
- **Слои субдискретизации** (subsampling, spatial pooling): уменьшают размер изображения (например, в 2 раза).
- **Полносвязные слои** (MLP) на выходе модели (используются для классификации).

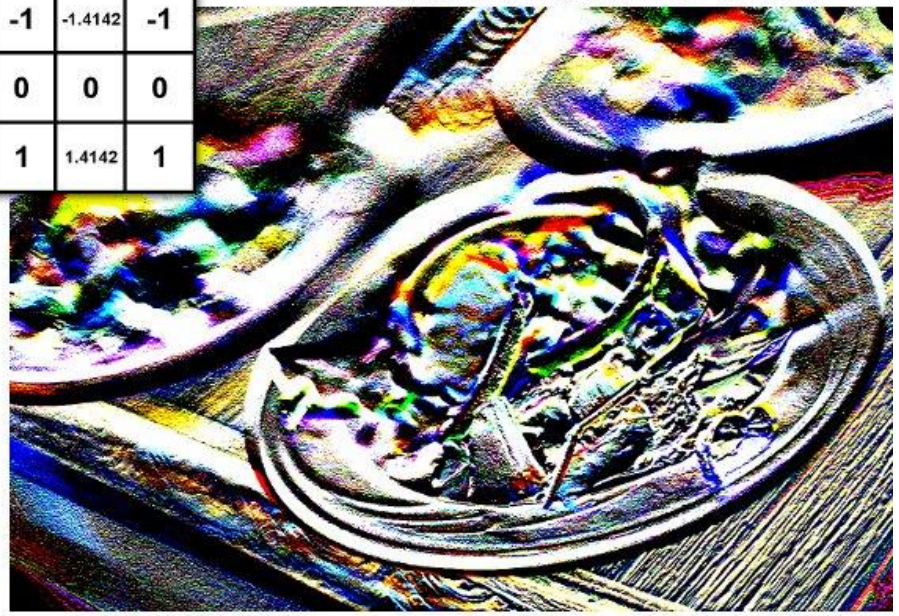
Diagonal Prewitt

0	1	1
-1	0	1
-1	-1	0



Horizontal Frei-Chen

-1	-1.4142	-1
0	0	0
1	1.4142	1

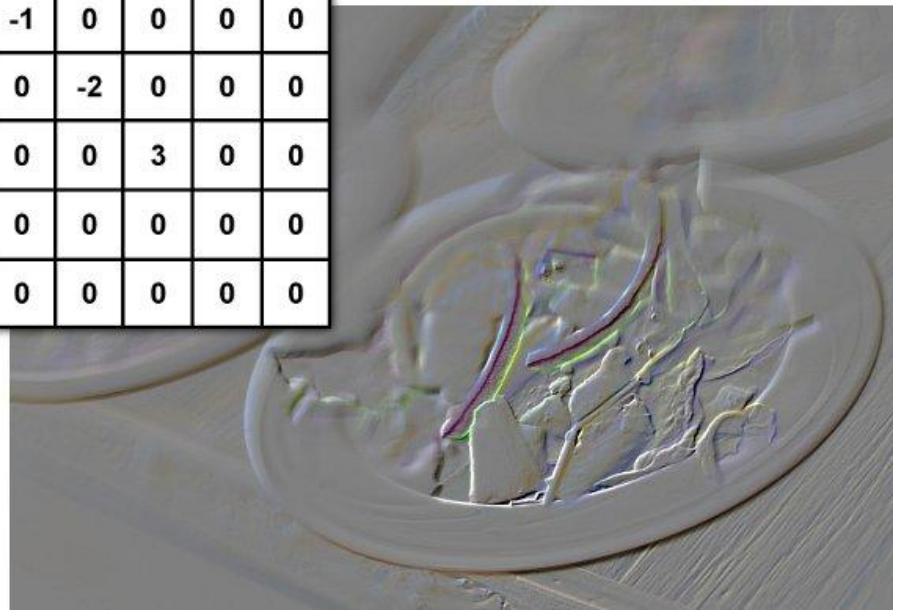


Sharpen

0	-1	0
-1	5	-1
0	-1	0

Emboss

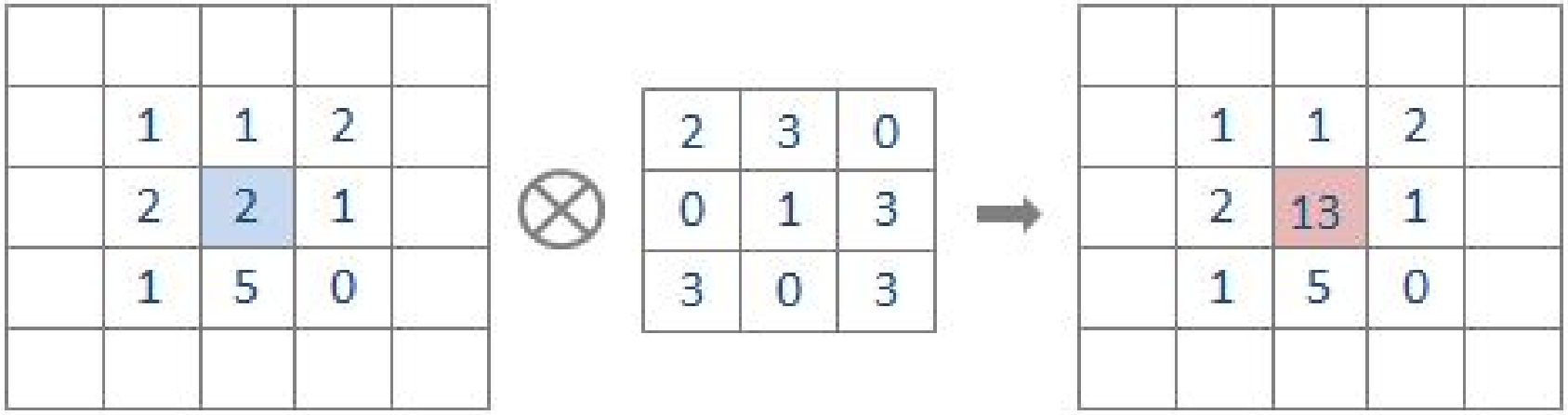
-1	0	0	0	0
0	-2	0	0	0
0	0	3	0	0
0	0	0	0	0
0	0	0	0	0



http://www.gimpbible.com/files/convolution-matrix/

Визуализация операции свёртки

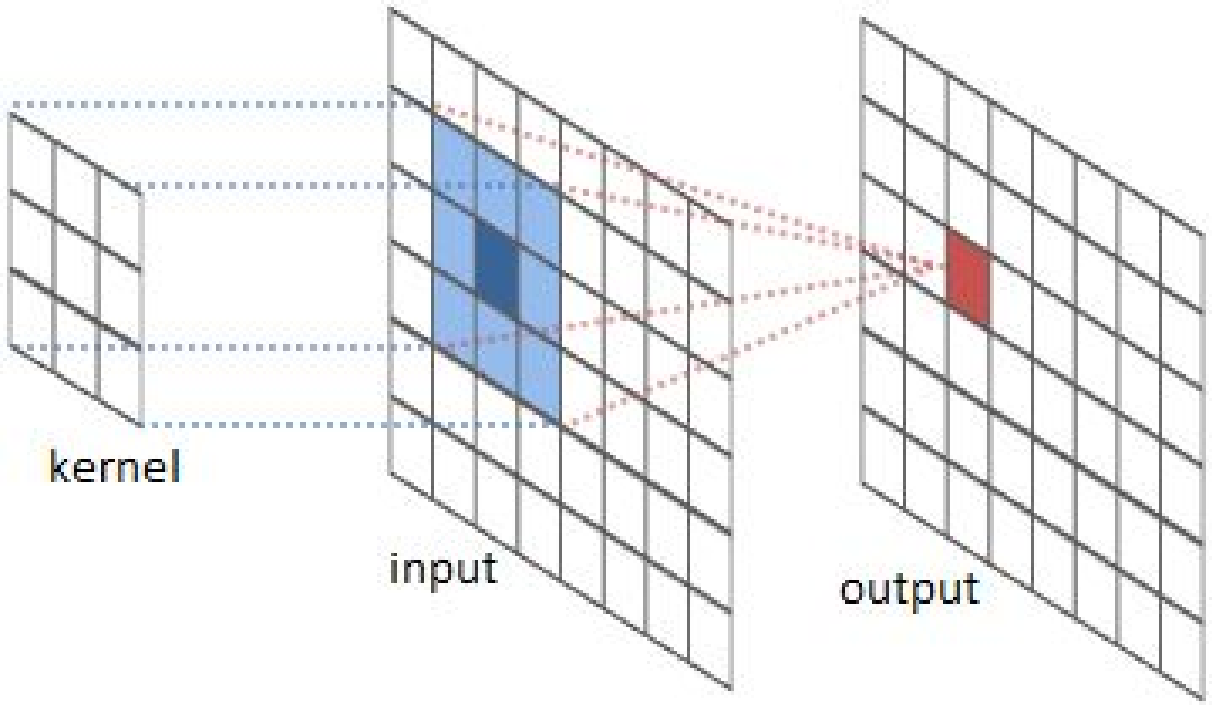
Знакомые по фотошопу фильтры blur, emboss, sharpen и другие — это именно матричные фильтры.



input kernel output

http://intellabs.github.io/RiverTrail/tutorial/

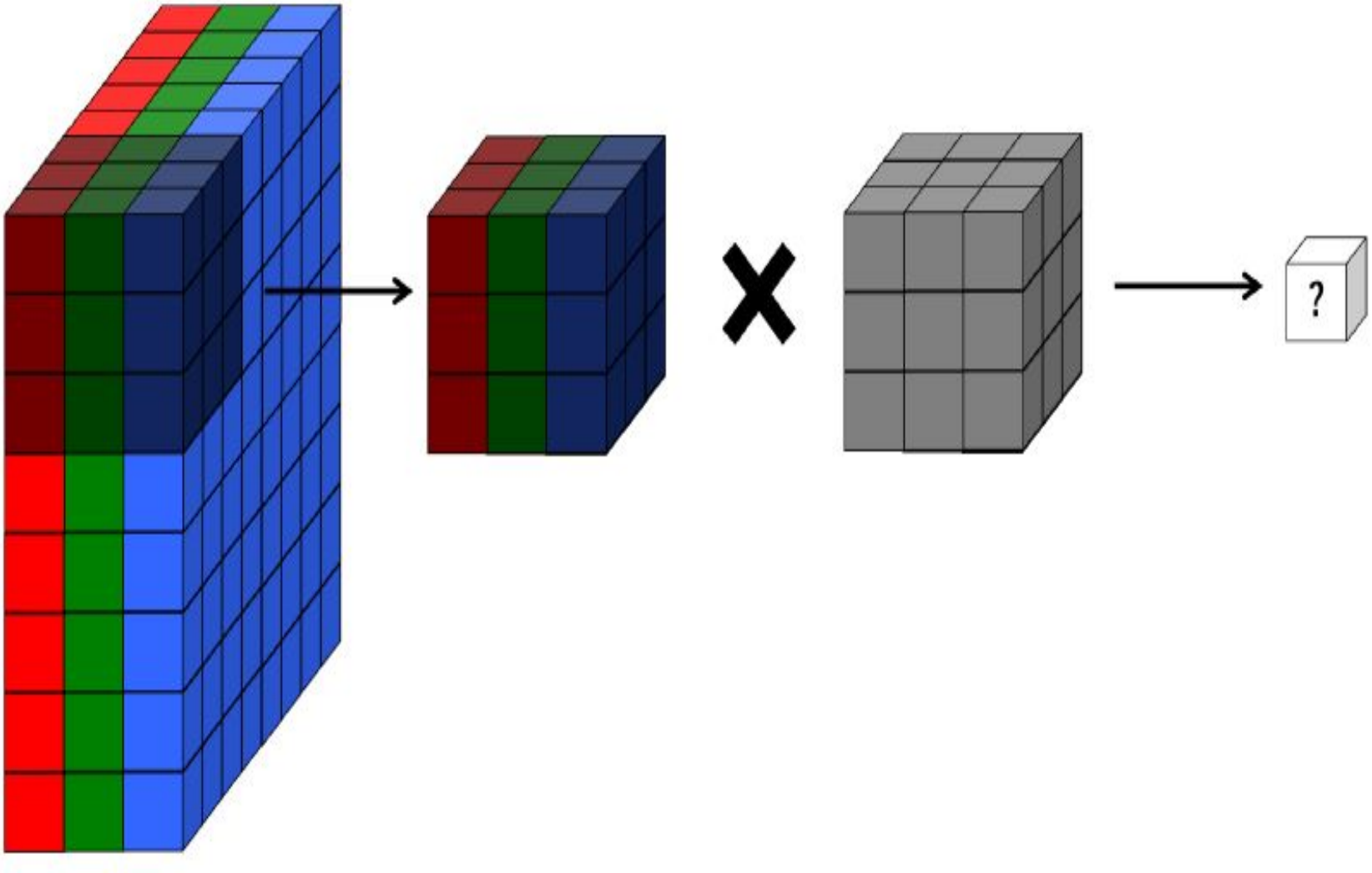
Визуализация операции свёртки



http://intellabs.github.io/RiverTrail/tutorial/

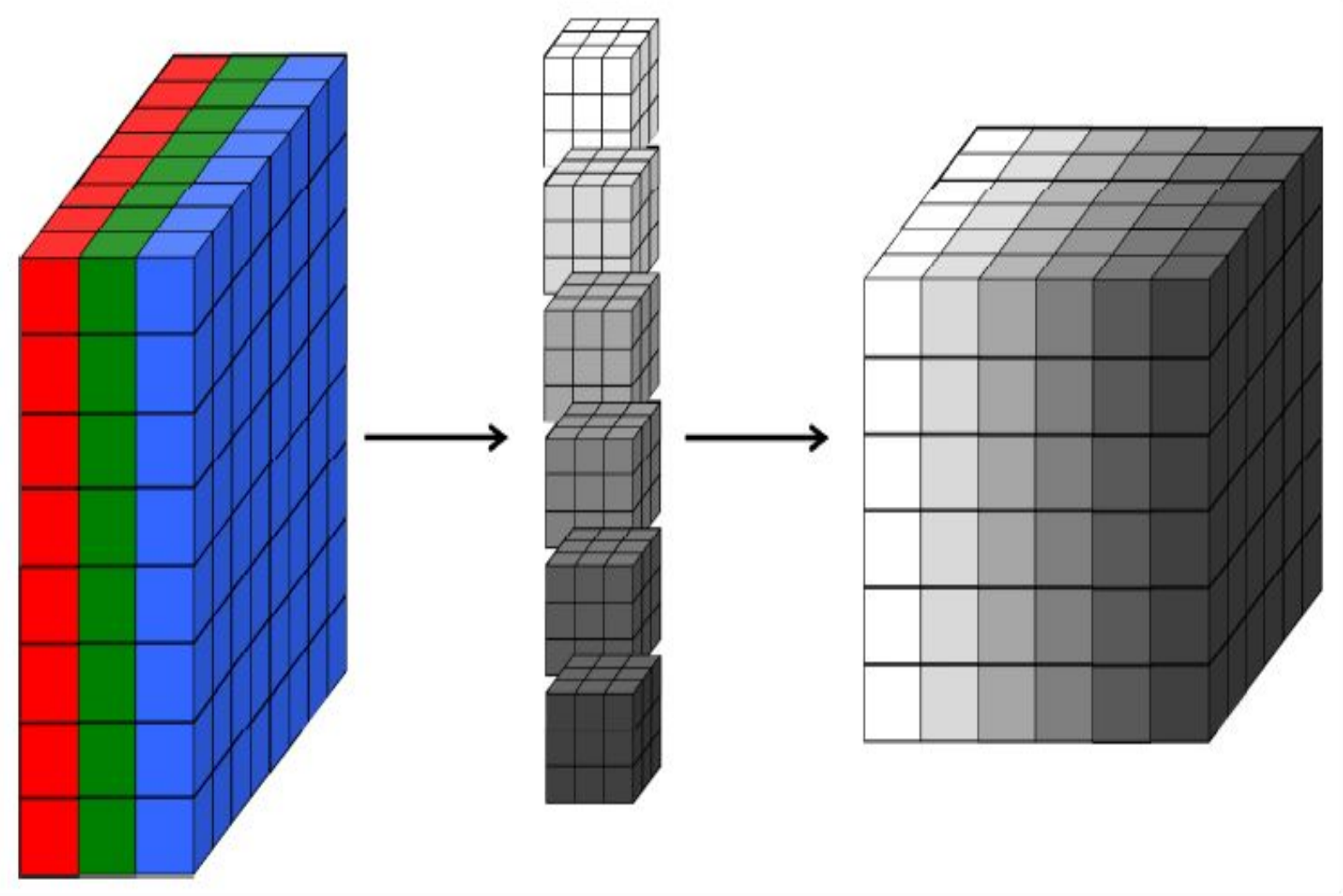
Свёртка работает над объёмами

В реальности фильтр трёхмерный.

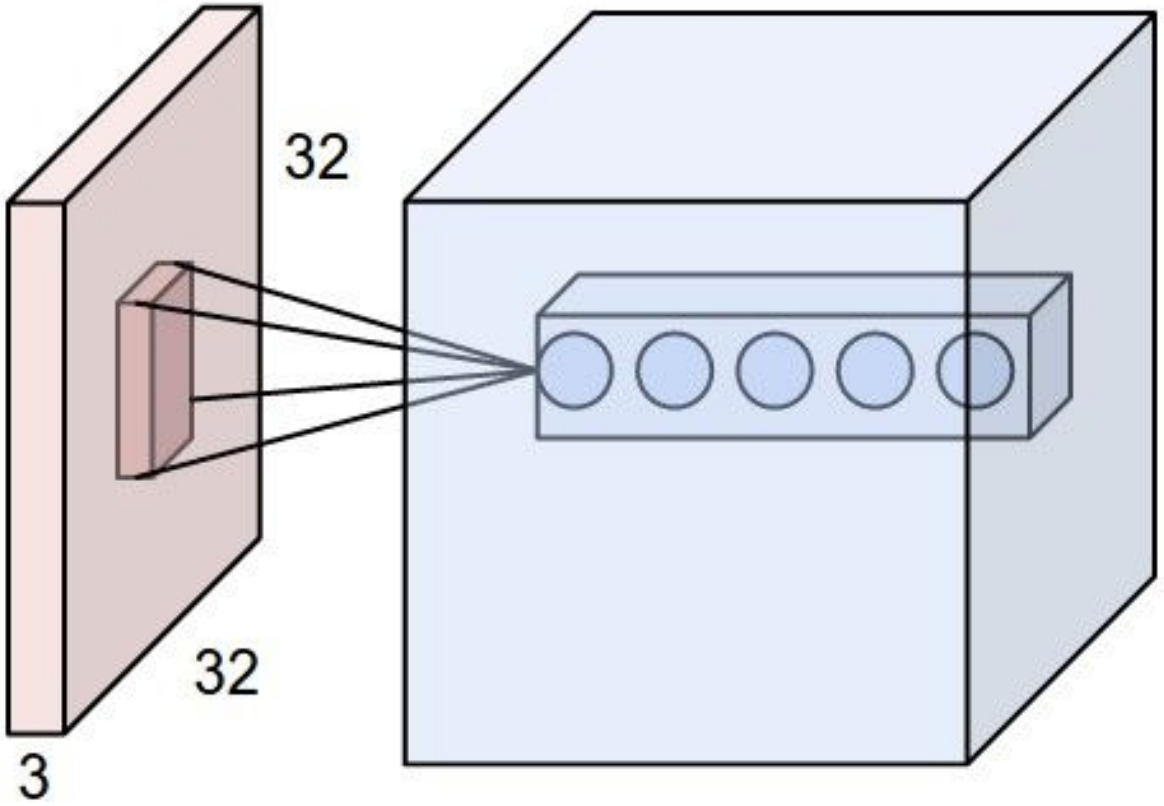


Свёртка работает с объёмами

Несколько фильтров (образующих один слой) создают “объём”



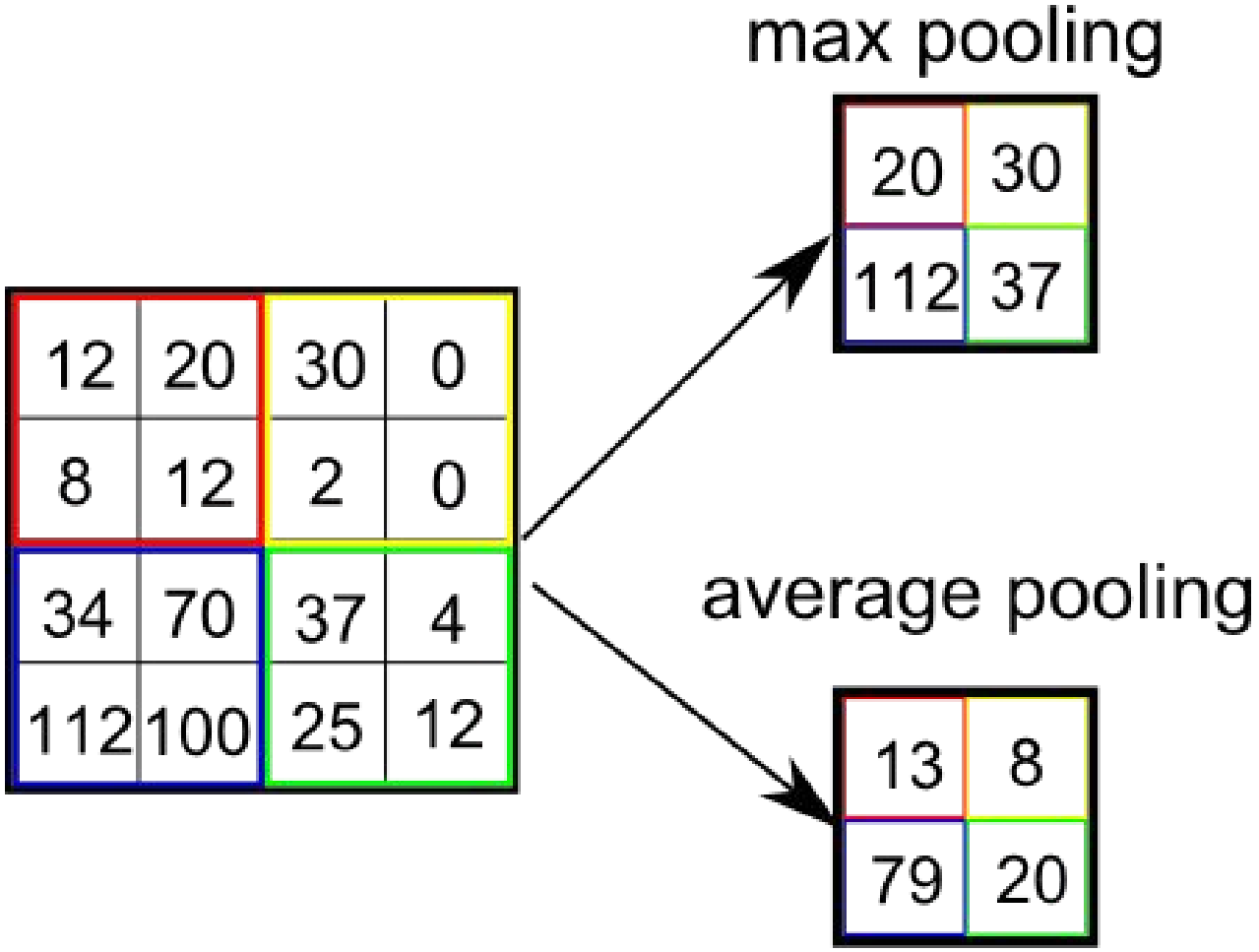
CNN: Свёрточный слой (5 нейронов)



Веса нейронов — это коэффициенты ядра свёртки. Каждая “обучаемая” свёртка выделяет одинаковые локальные признаки во всех частях изображения.

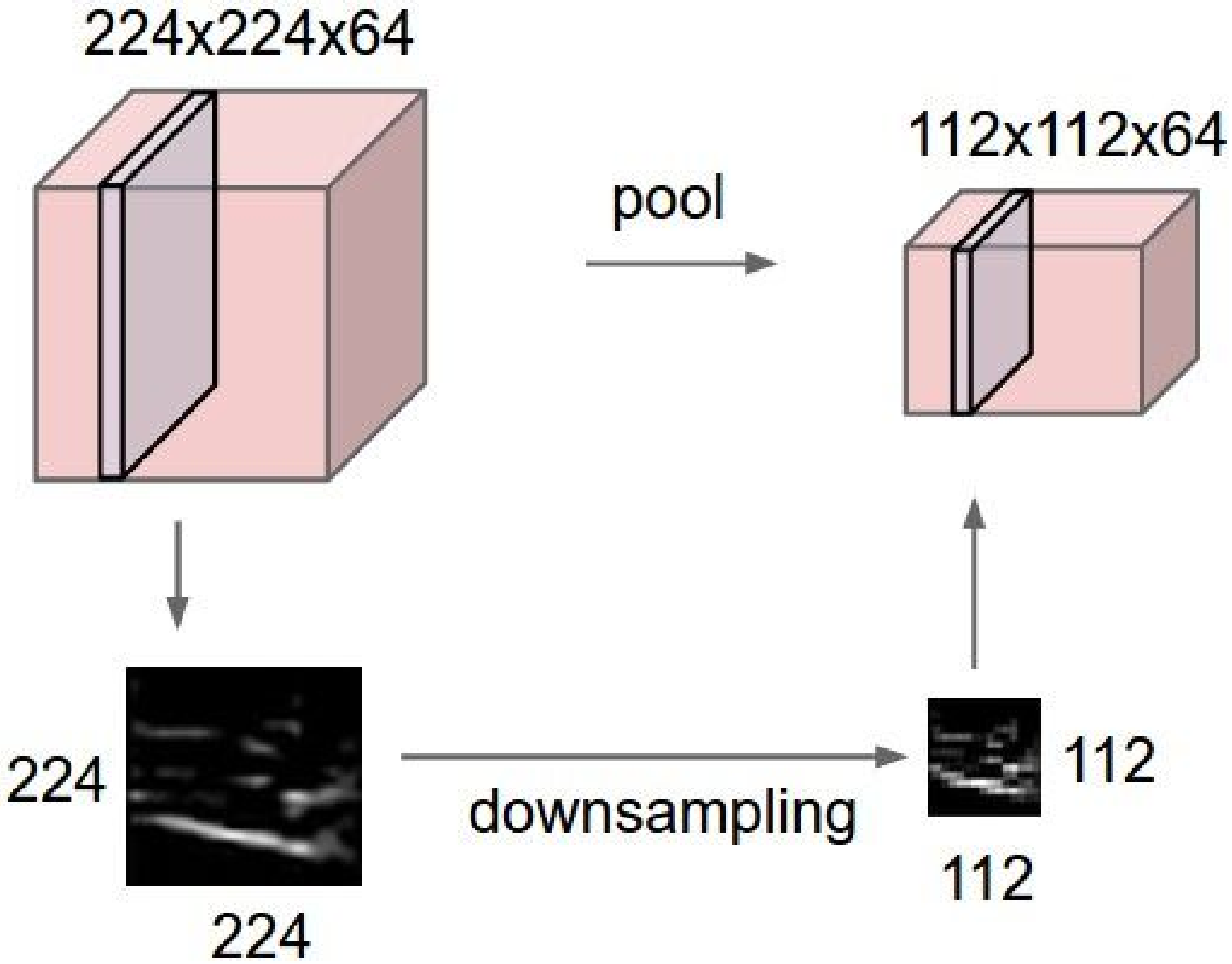
http://cs231n.github.io/convolutional-networks/

Операция pooling (max pool, avg pool)



http://vaaaaaanquish.hatenablog.com/entry/2015/01/26/060622

CNN: Pooling слой (downsampling)



http://cs231n.github.io/convolutional-networks/

Пример сети: LeNet-5

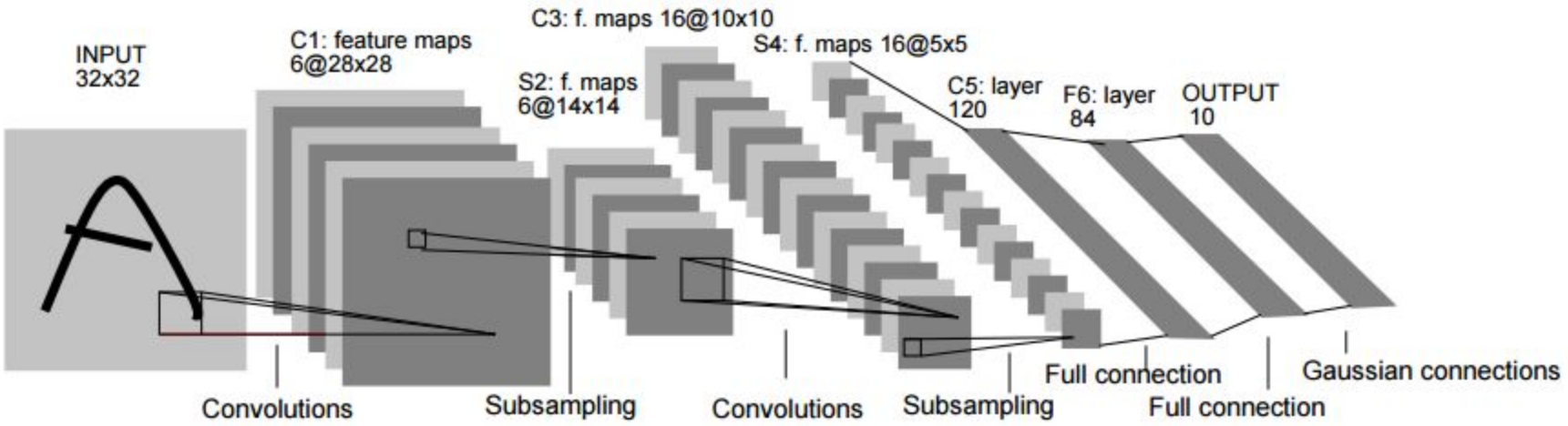


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

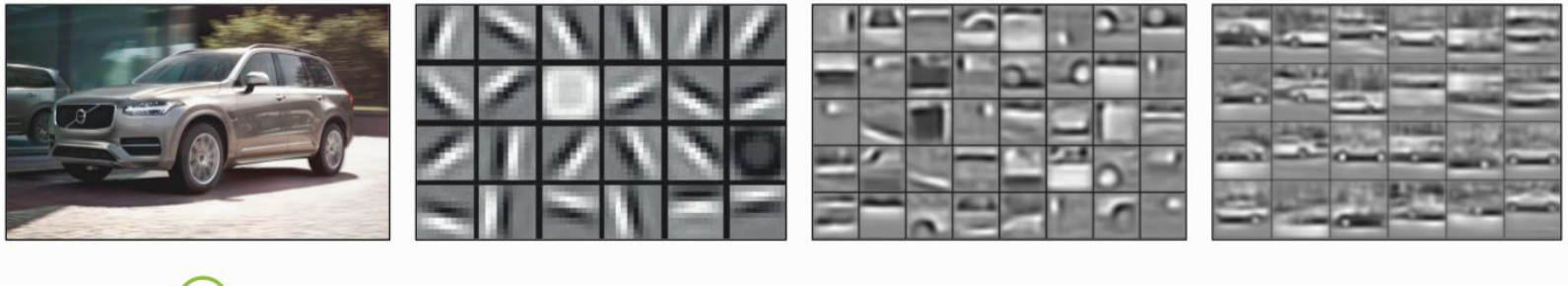
http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

Пример: объявление сети (Keras)

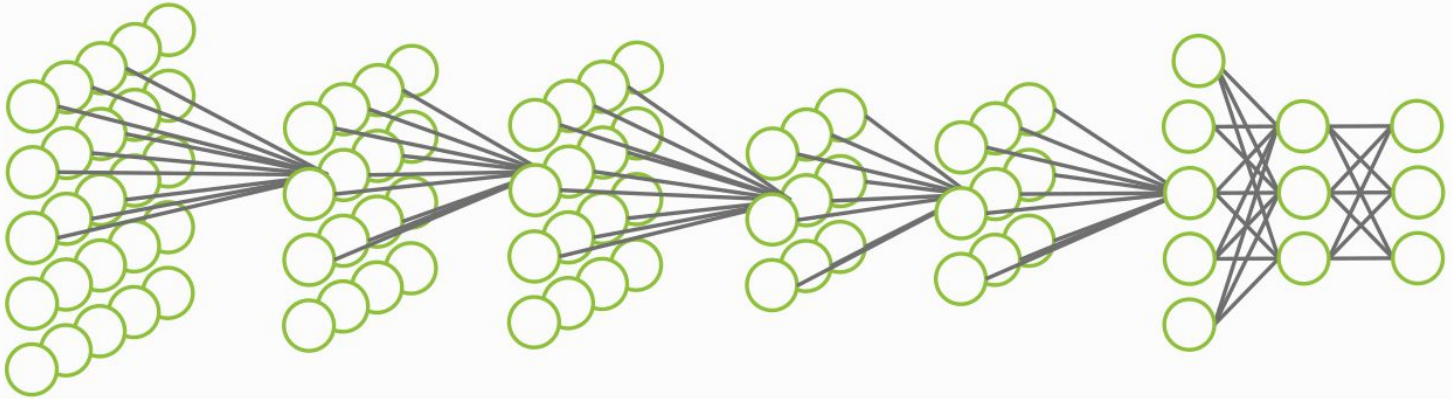
```
model = Sequential()

model.add(Convolution2D(20, 5, 5, border_mode="same",
  input_shape=(depth, height, width)))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Convolution2D(50, 5, 5, border_mode="same"))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Flatten())
model.add(Dense(500))
model.add(Activation("relu"))
model.add(Dense(classes))
model.add(Activation("softmax"))

model.compile(loss=keras.losses.categorical_crossentropy,
    optimizer=keras.optimizers.Adadelta(), metrics=['accuracy'])

model.fit(x_train, y_train, batch_size=batch_size, epochs=epochs,
    verbose=1, validation_data=(x_test, y_test))
score = model.evaluate(x_test, y_test, verbose=0)
```

Свёрточная нейросеть

Свёрточные слои учат иерархические признаки для изображений, а spatial pooling даёт некоторую инвариантность к перемещениям.



Image



“Volvo XC90”

У CNN меньше параметров, чем у FNN

CNN

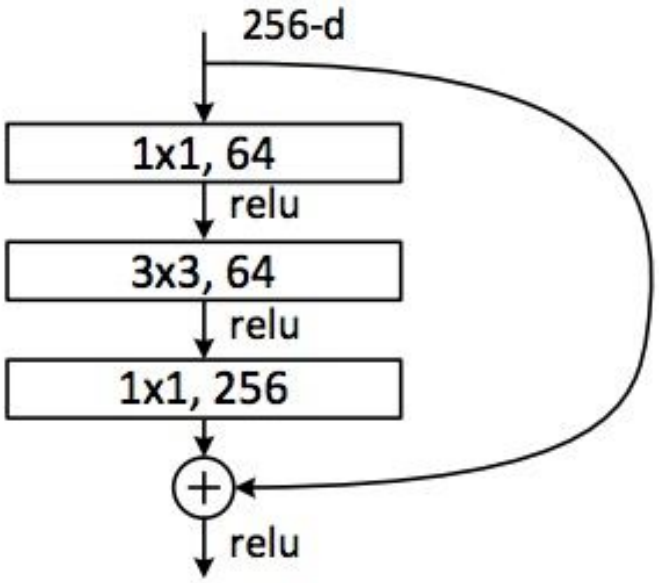
- вход ч/б картинка 100x100
- три свёрточных слоя по 100 плоскостей каждый (conv 5x5 и subsampling 2)
- выход: 10 классов
- число параметров примерно 6**50K** (5*5*1*100 + 5*5*100*100 + 5*5*100*100 + 12*12*100*10)

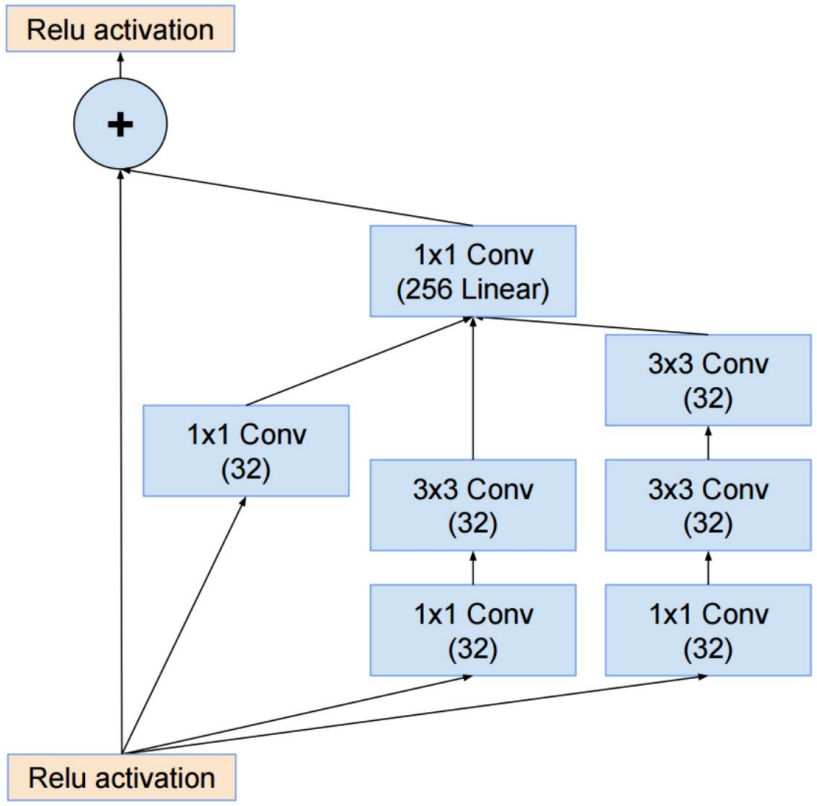
FNN

- вход: ч/б картинка 100x100
- три скрытых слоя по 100 нейронов каждый
- выход: 10 классов
- число параметров примерно **1M** (10000*100 + 100*100 + 100*100 + 100*10)

Современные архитектуры

Inception, ResNet и другие современные архитектуры содержат специальные блоки слоёв.





Пример с обработкой изображений

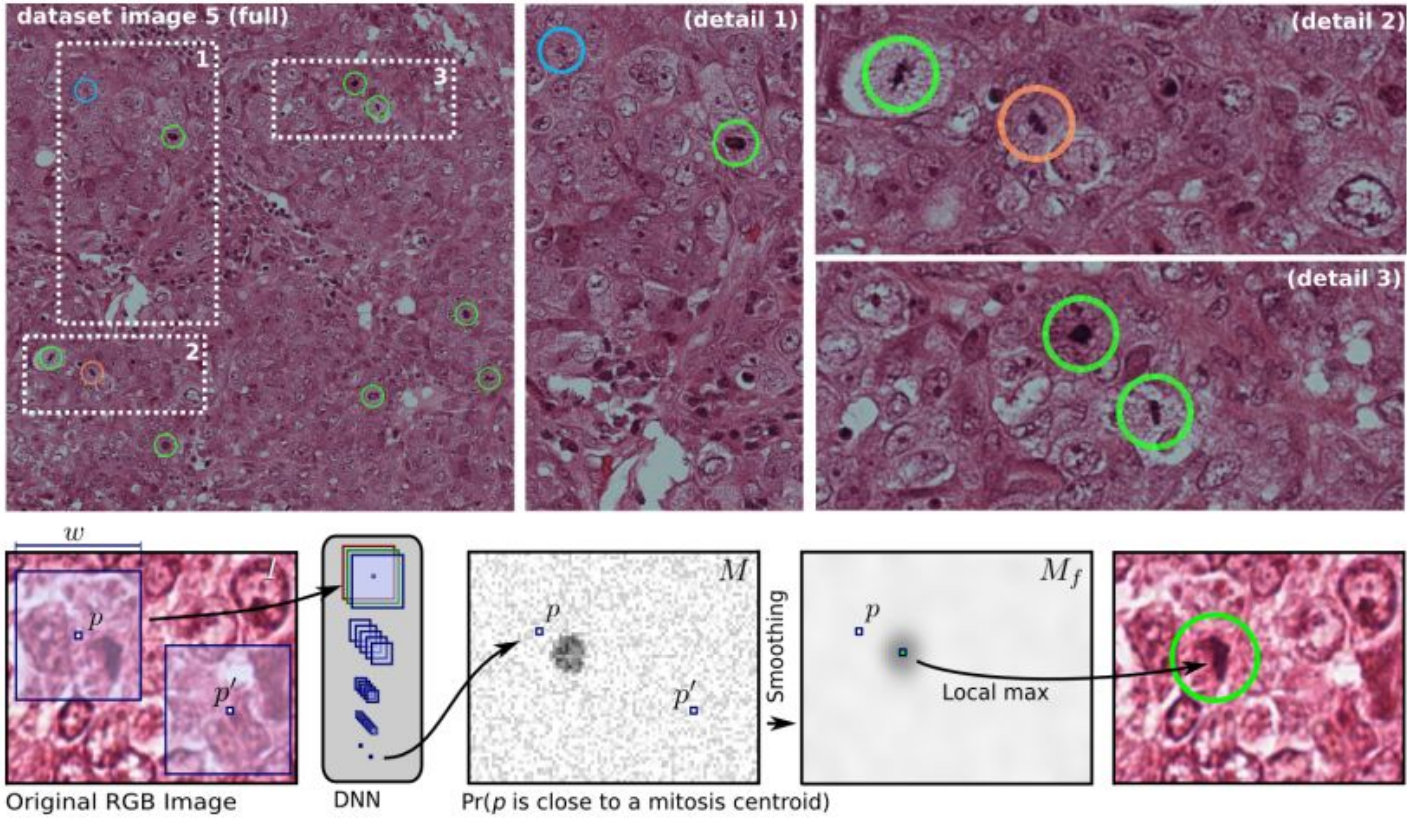


Fig. 1. *Top left:* one image (4 MPixels) corresponding to one of the 50 high power fields represented in the dataset. Our detected mitosis are circled green (true positives) and red (false positives); cyan denotes mitosis not detected by our approach. *Top right:* details of three areas (full-size results on the whole dataset in supplementary material). Note the challenging appearance of mitotic nuclei and other very similar non-mitotic structures. *Bottom:* overview of our detection approach.

Mitosis detection in breast cancer histology images with deep neural networks.
http://people.idsia.ch/~juergen/miccai2013.pdf

Пример с обработкой изображений

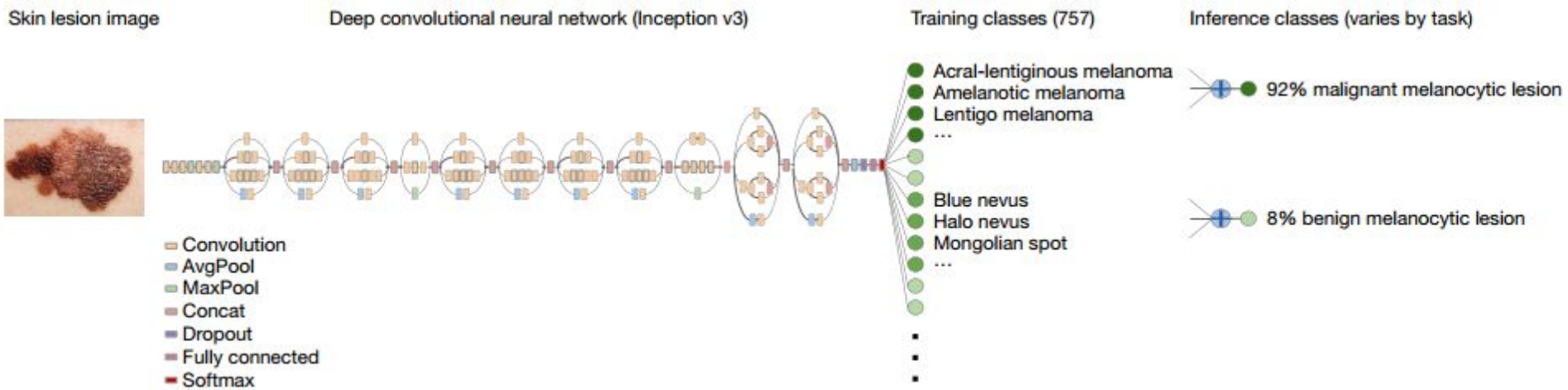
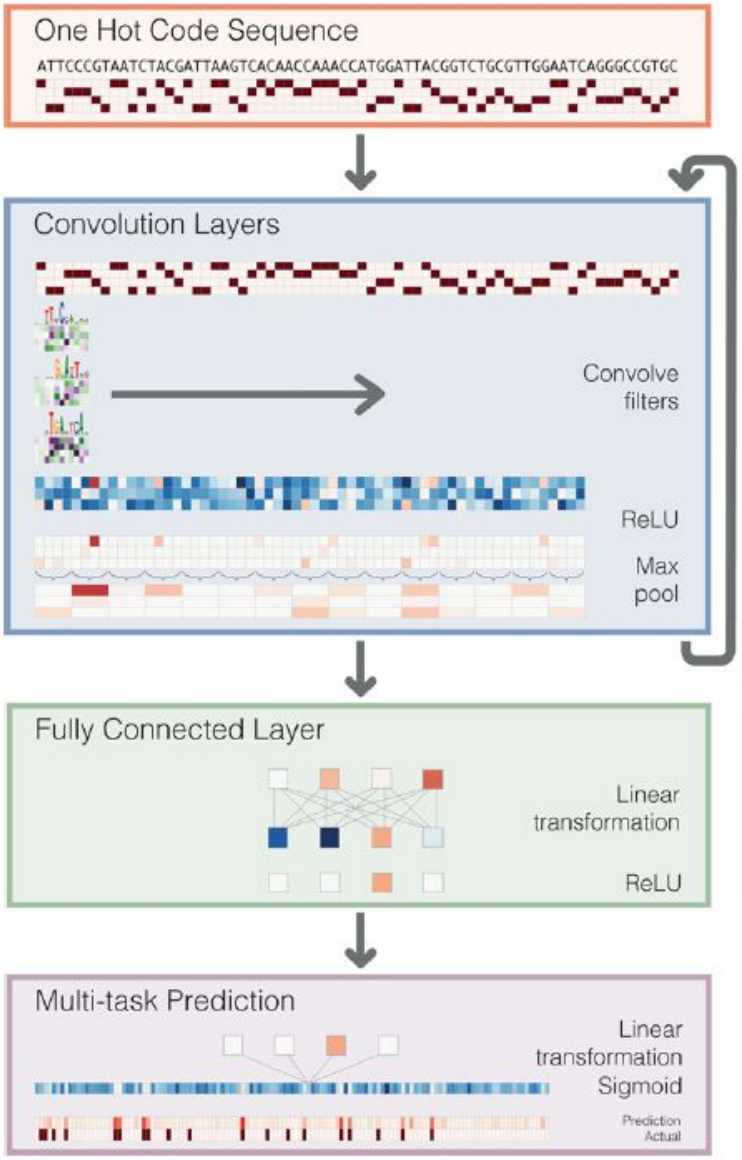


Figure 1 | Deep CNN layout. Our classification technique is a deep CNN. Data flow is from left to right: an image of a skin lesion (for example, melanoma) is sequentially warped into a probability distribution over clinical classes of skin disease using Google Inception v3 CNN architecture pretrained on the ImageNet dataset (1.28 million images over 1,000 generic object classes) and fine-tuned on our own dataset of 129,450 skin lesions comprising 2,032 different diseases. The 757 training classes are defined using a novel taxonomy of skin disease and a partitioning algorithm that maps diseases into training classes (for example, acrolentiginous melanoma, amelanotic melanoma, lentigo melanoma). Inference classes are more general and are composed of one or more training classes (for example, malignant melanocytic lesions—the class of melanomas). The probability of an inference class is calculated by summing the probabilities of the training classes according to taxonomy structure.

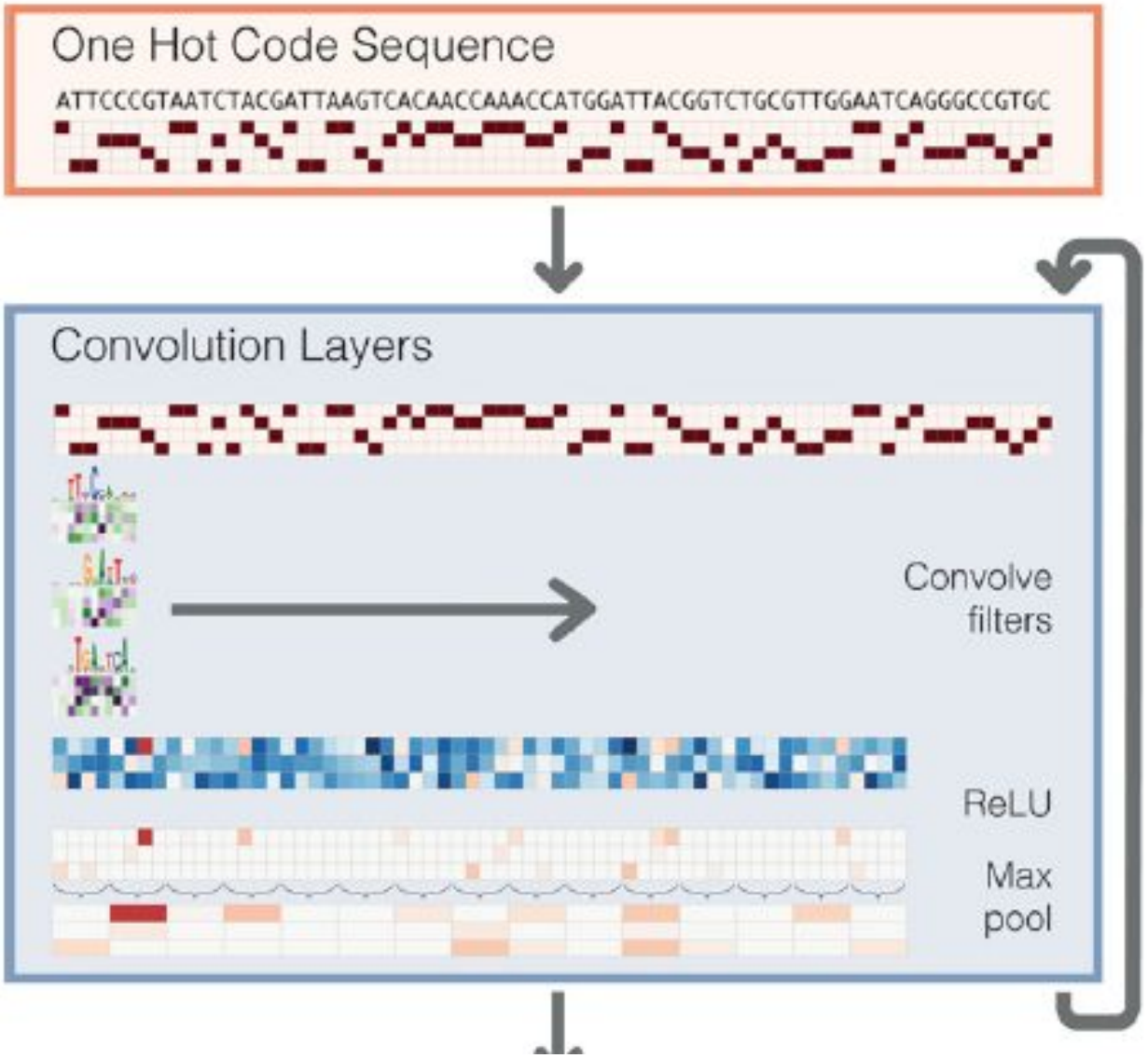
Dermatologist-level classification of skin cancer with deep neural networks
http://www.nature.com/nature/journal/v542/n7639/full/nature21056.html

Пример с геномными данными



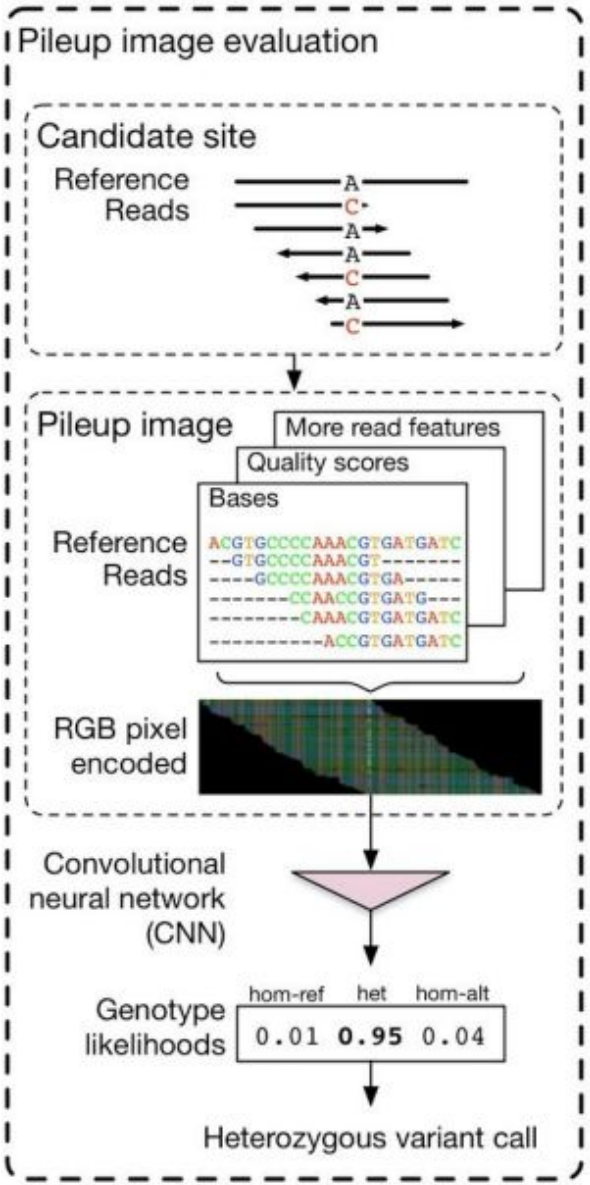
Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks
http://genome.cshlp.org/content/26/7/990

Пример с геномными данными



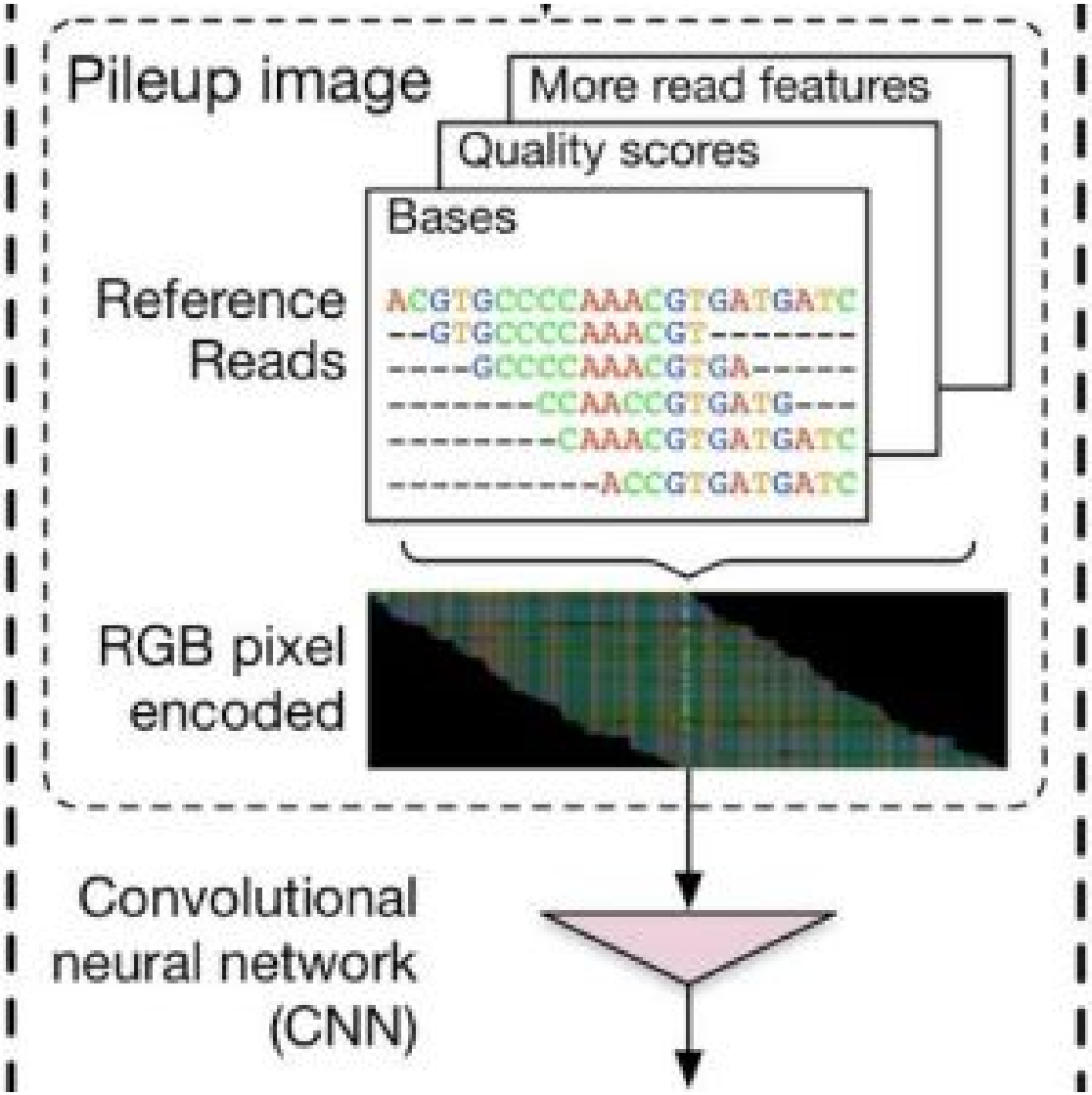
Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks
http://genome.cshlp.org/content/26/7/990

Пример с геномными данными



Creating a universal SNP and small indel variant caller with deep neural networks
http://biorxiv.org/content/early/2016/12/21/092890

Пример с геномными данными

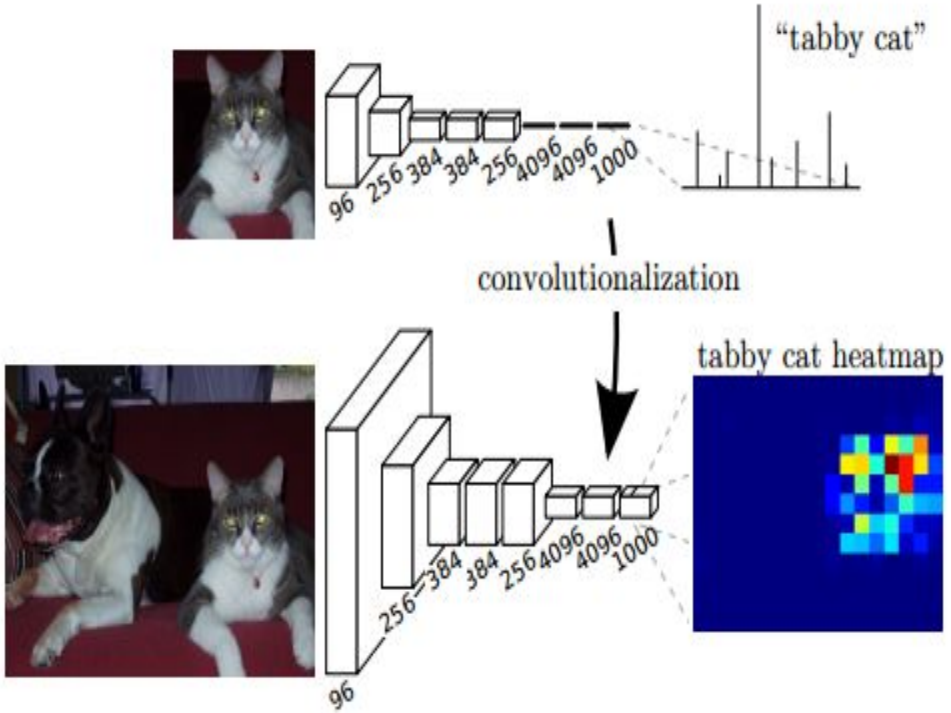


Creating a universal SNP and small indel variant caller with deep neural networks
http://biorxiv.org/content/early/2016/12/21/092890

Fully-convolutional networks (FCN)

Обычная свёрточная сеть, но без MLP сверху (нет полносвязных слоёв).

Позволяет работать с изображениями произвольного размера и выдавать на выходе тепловую карту классификации.

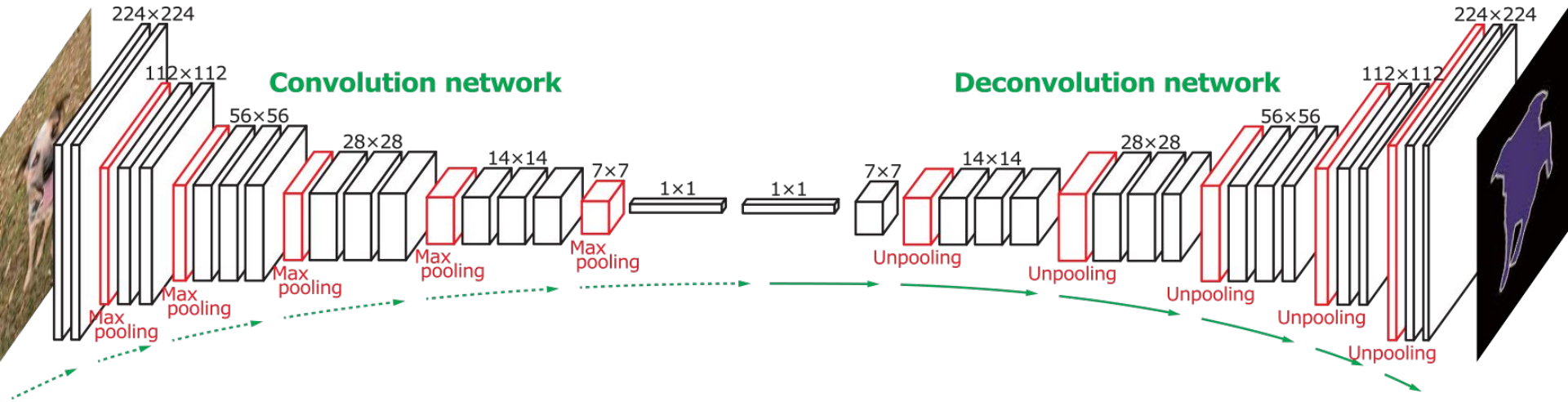


https://arxiv.org/abs/1411.4038

Deconvolution networks

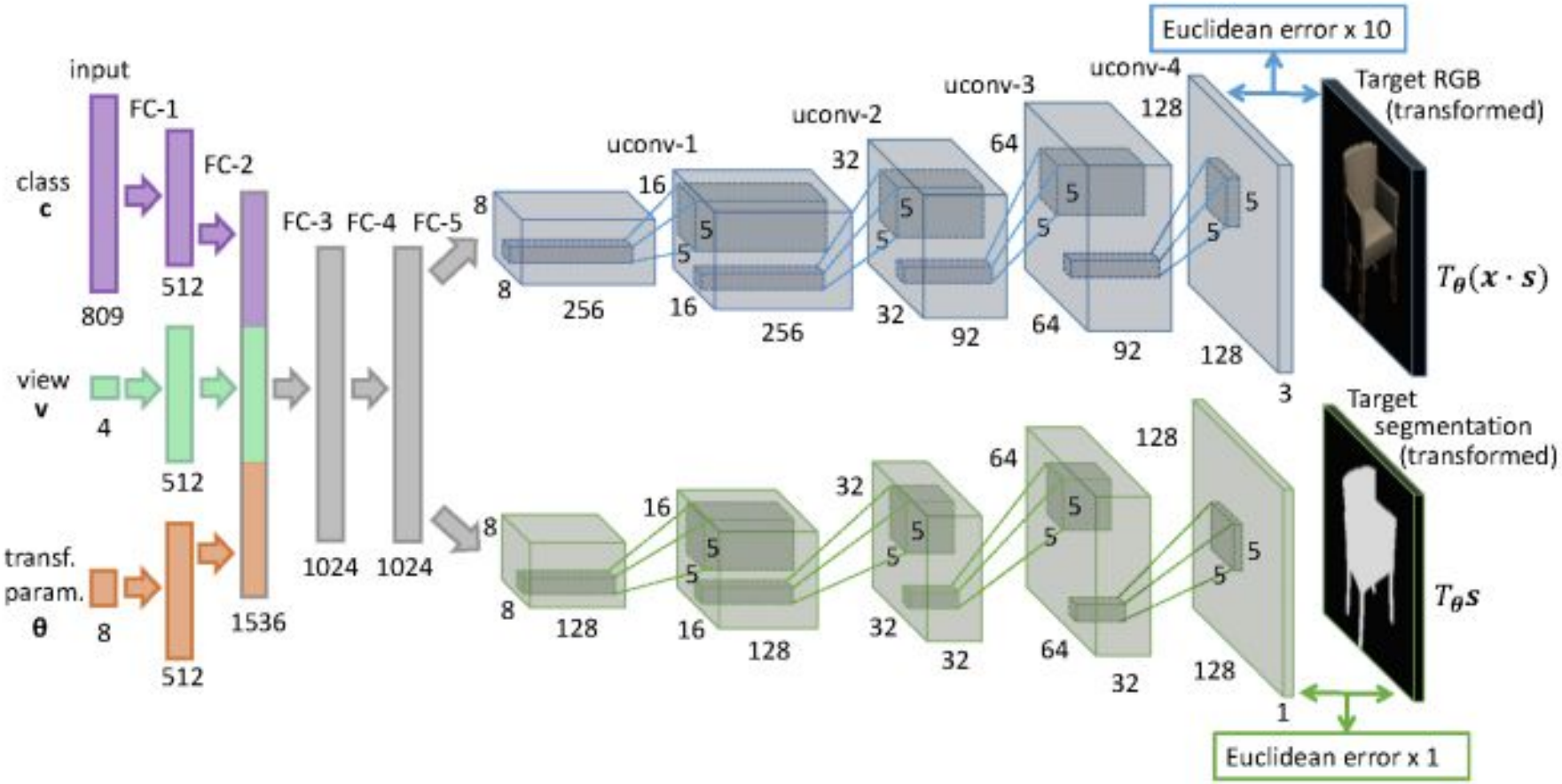
Правильнее называть это Transposed convolution, а не Deconvolution (это слово уже занято в цифровой обработке сигналов для обратной операции).

По сути, реализован обучаемый upsampling.



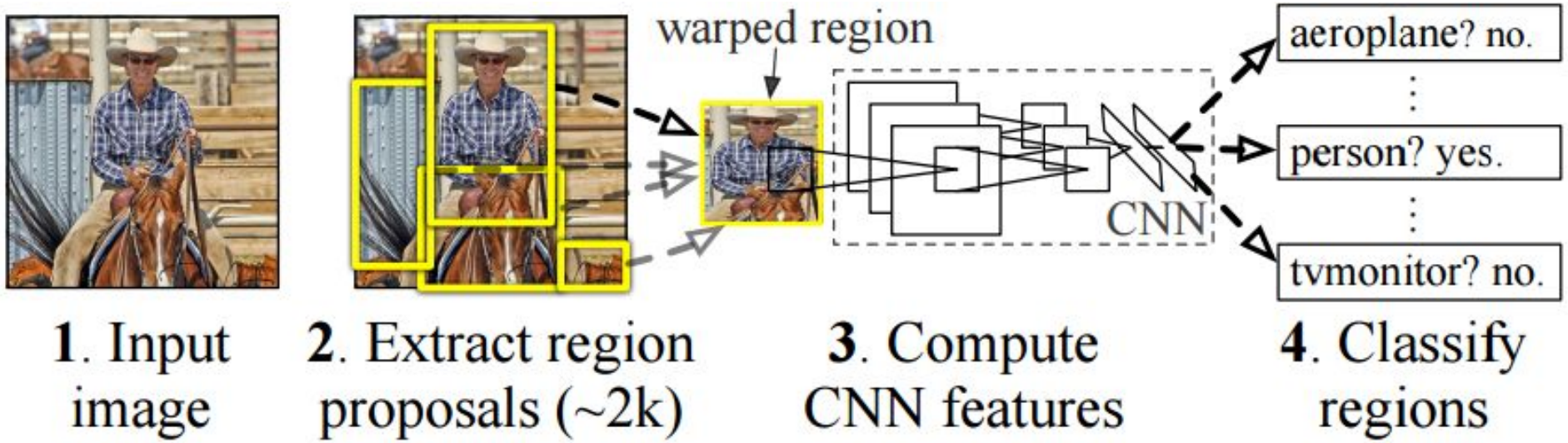
http://cvlab.postech.ac.kr/research/deconvnet/

Генерация изображений



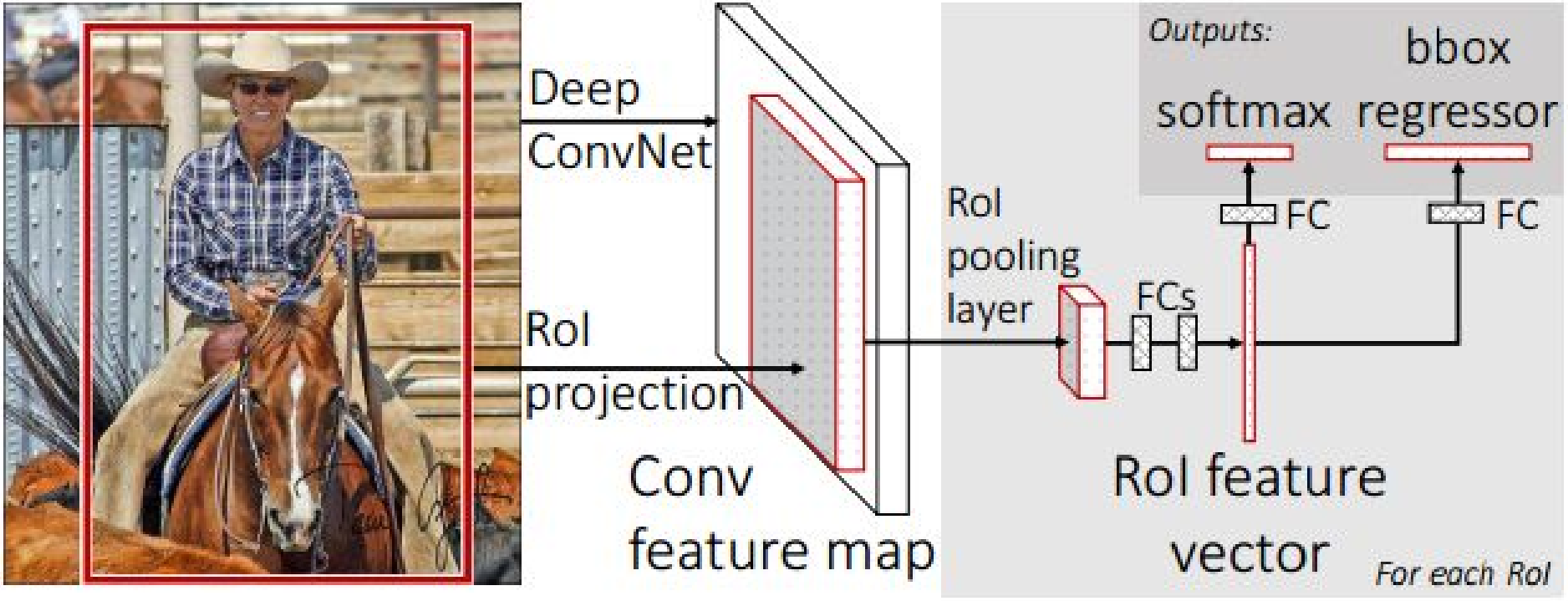
https://arxiv.org/abs/1411.5928

R-CNN: Region-based Convolutional Network



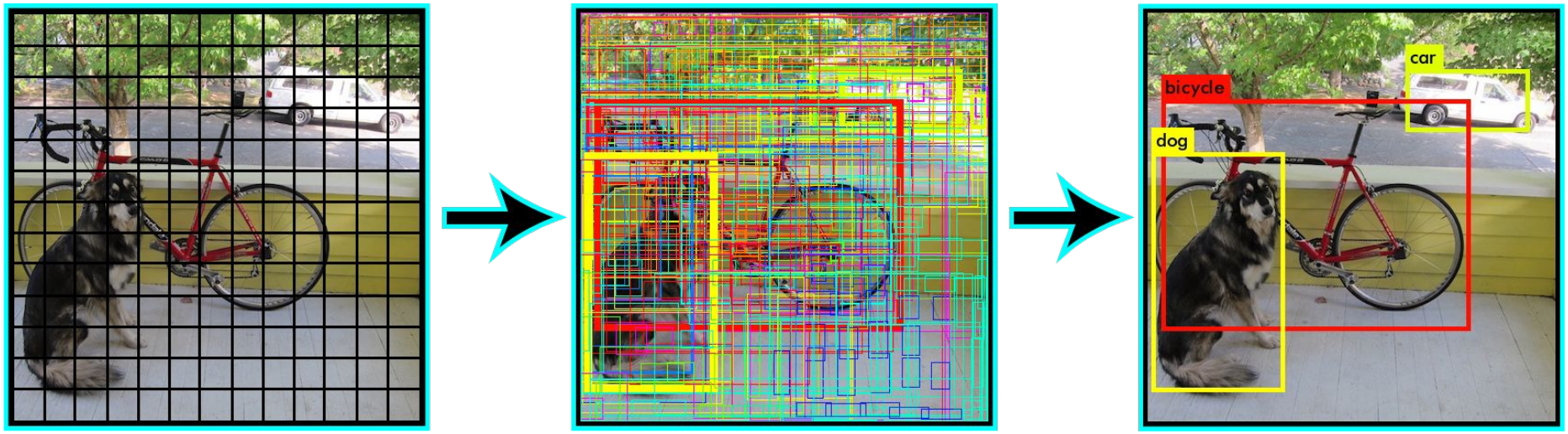
https://github.com/rbgirshick/rcnn
https://people.eecs.berkeley.edu/~rbg/papers/pami/rcnn_pami.pdf

Fast R-CNN



http://tutorial.caffe.berkeleyvision.org/caffe-cvpr15-detection.pdf

YOLO: Real-Time Object Detection

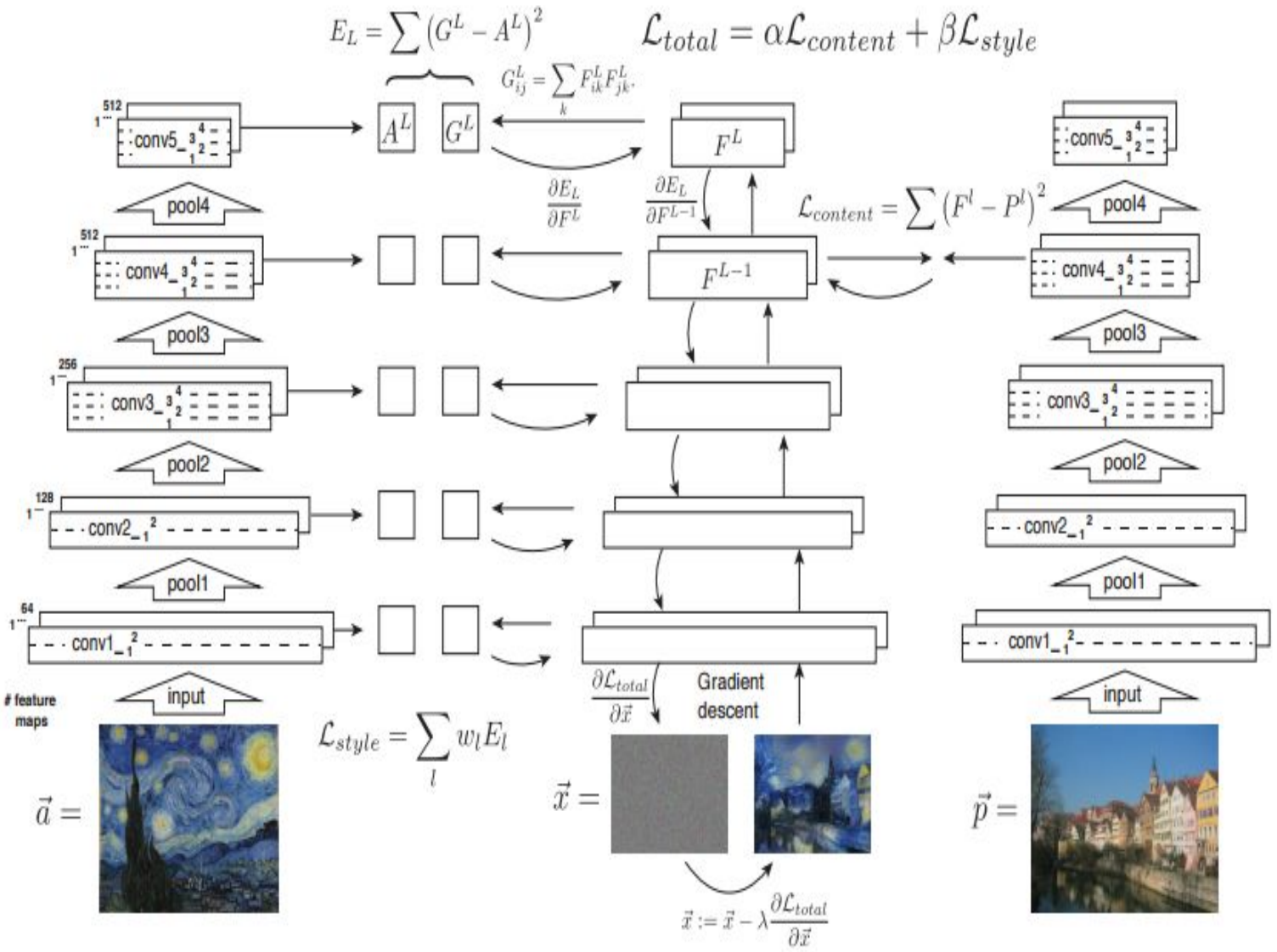


https://pjreddie.com/darknet/yolo/

Неклассические задачи: перенос стиля

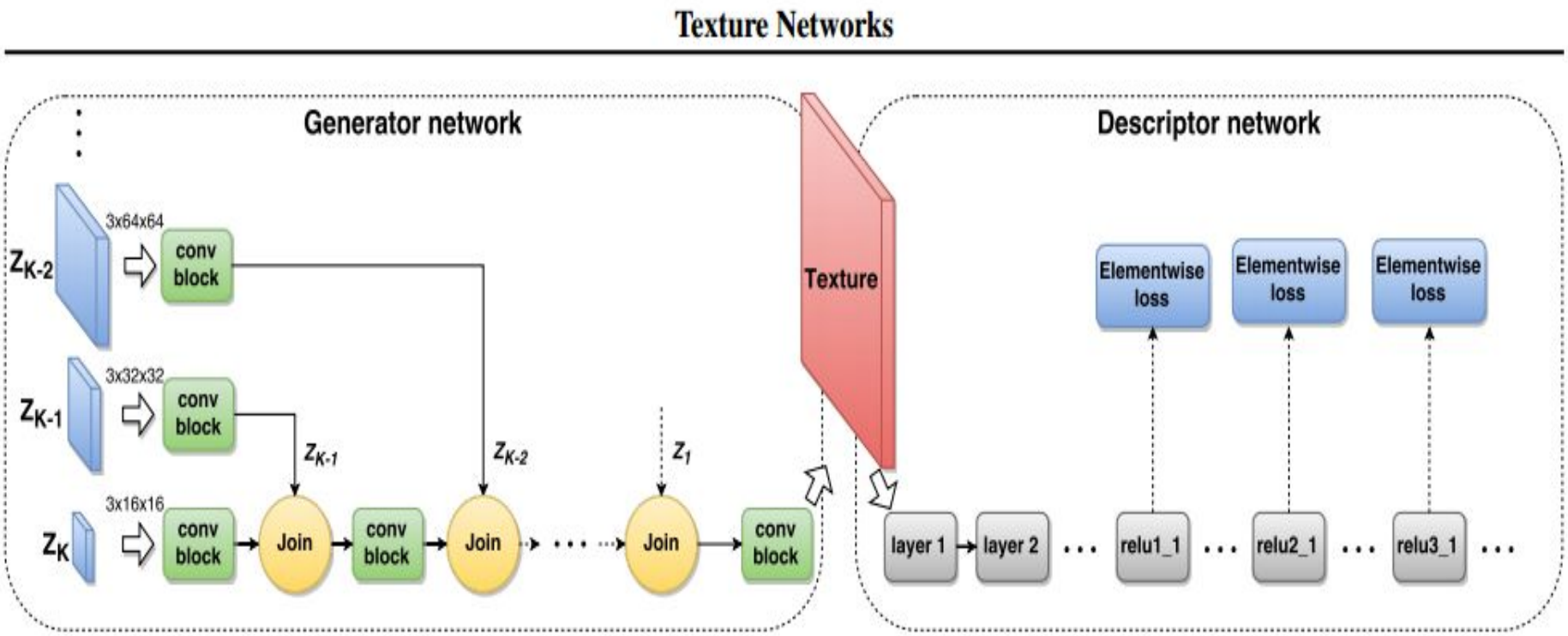
https://arxiv.org/abs/1508.06576

Перенос стиля: оригинальный алгоритм



https://arxiv.org/abs/1508.06576

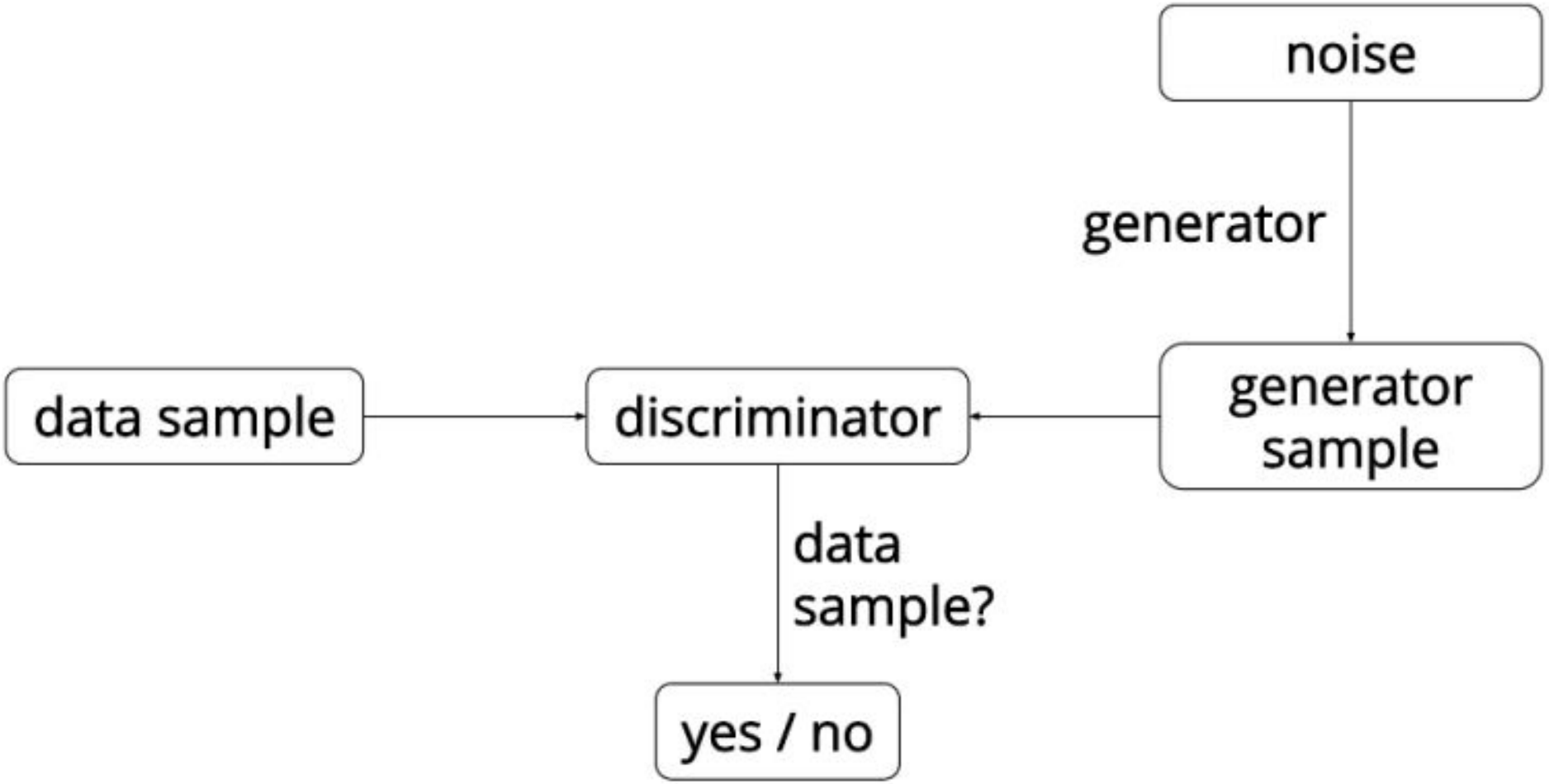
Перенос стиля: быстрый алгоритм



https://arxiv.org/abs/1603.03417

Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs)



https://arxiv.org/abs/1411.5928

Generative Adversarial Networks (GANs)

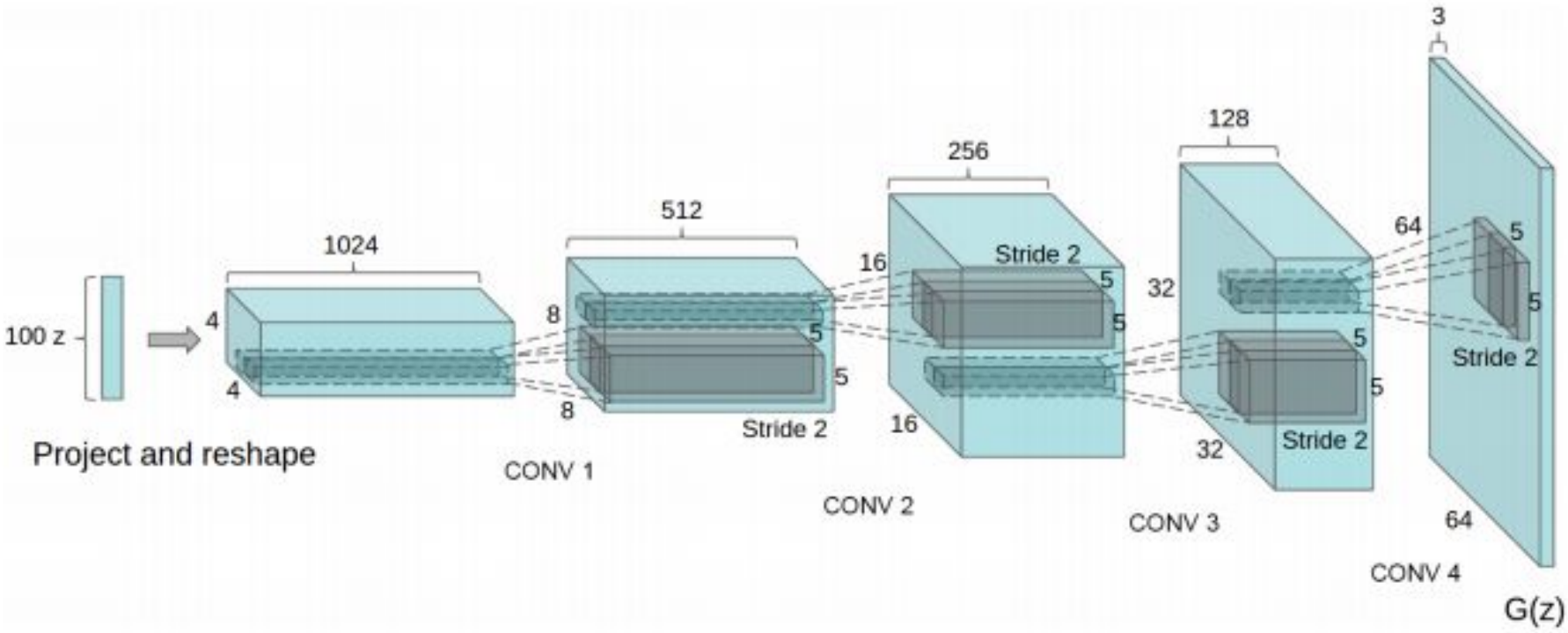


Figure 1: DCGAN generator used for LSUN scene modeling. A 100 dimensional uniform distribution Z is projected to a small spatial extent convolutional representation with many feature maps. A series of four fractionally-strided convolutions (in some recent papers, these are wrongly called deconvolutions) then convert this high level representation into a 64×64 pixel image. Notably, no fully connected or pooling layers are used.

https://arxiv.org/pdf/1511.06434v2.pdf

Generative Adversarial Networks (GANs)

Figure 2: Generated bedrooms after one training pass through the dataset. Theoretically, the model could learn to memorize training examples, but this is experimentally unlikely as we train with a small learning rate and minibatch SGD. We are aware of no prior empirical evidence demonstrating memorization with SGD and a small learning rate.

https://arxiv.org/pdf/1511.06434v2.pdf

Generative Adversarial Networks (GANs)

volcano

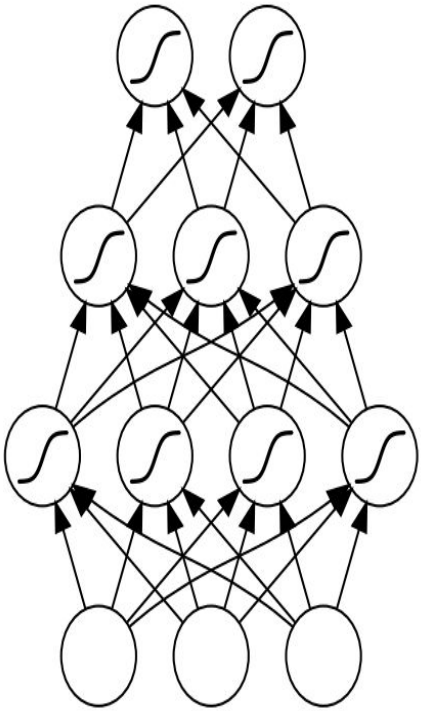
http://www.evolvingai.org/ppgn

Рекуррентные нейросети

Recurrent Neural Networks, RNN

Основы RNN

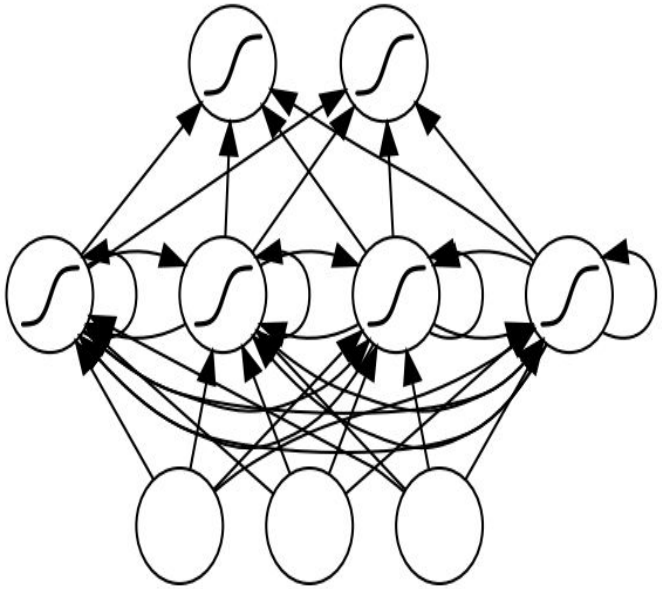
Feedforward NN vs. Recurrent NN



Output Layer

Hidden Layers

Input Layer



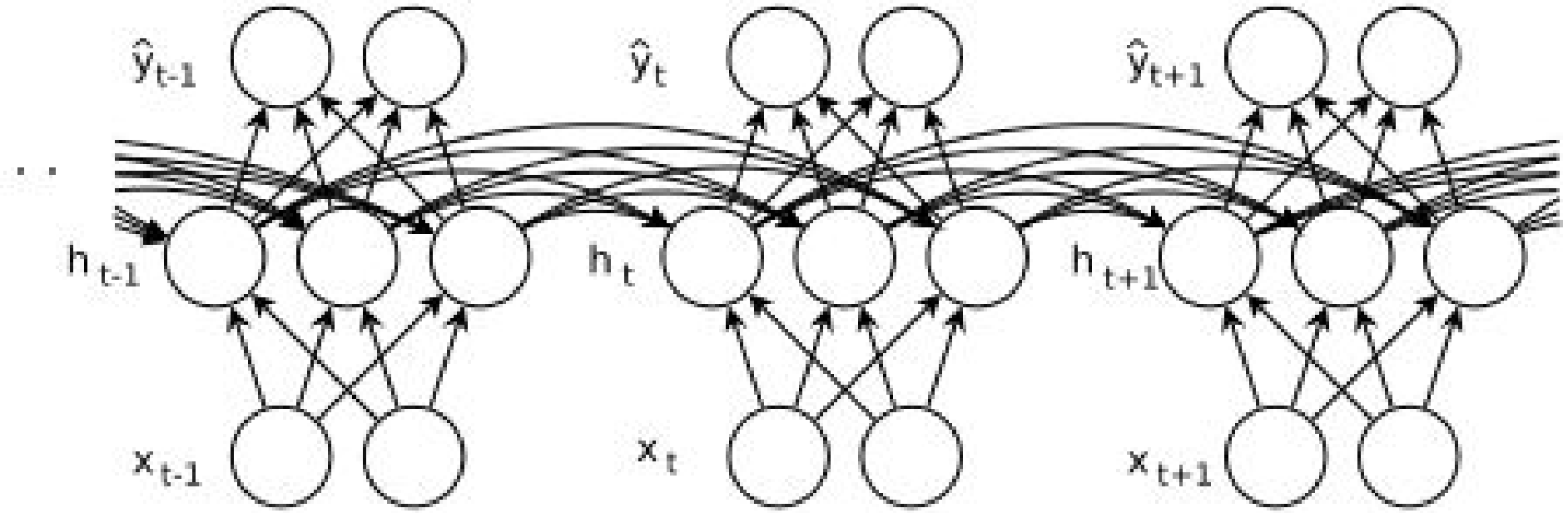
Output Layer

Hidden Layer

Input Layer

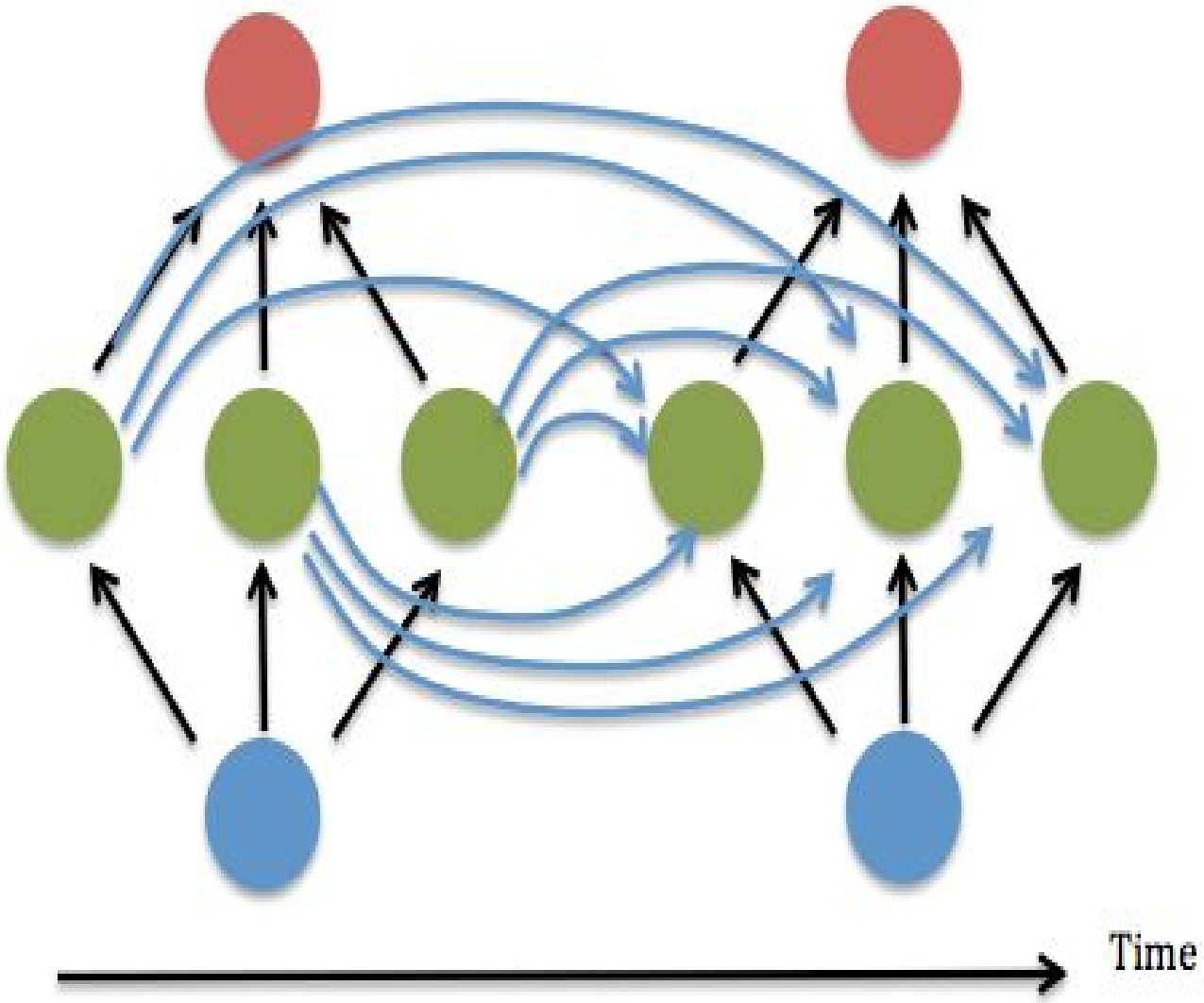
В RNNs разрешены циклические связи

Что такое циклическая связь?



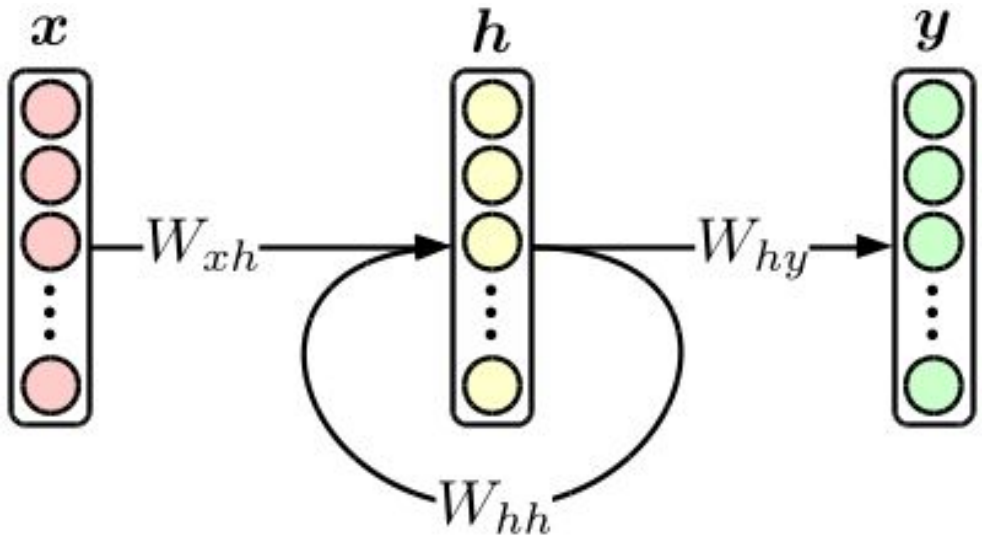
Циклические связи существуют во времени

Что такое циклическая связь?

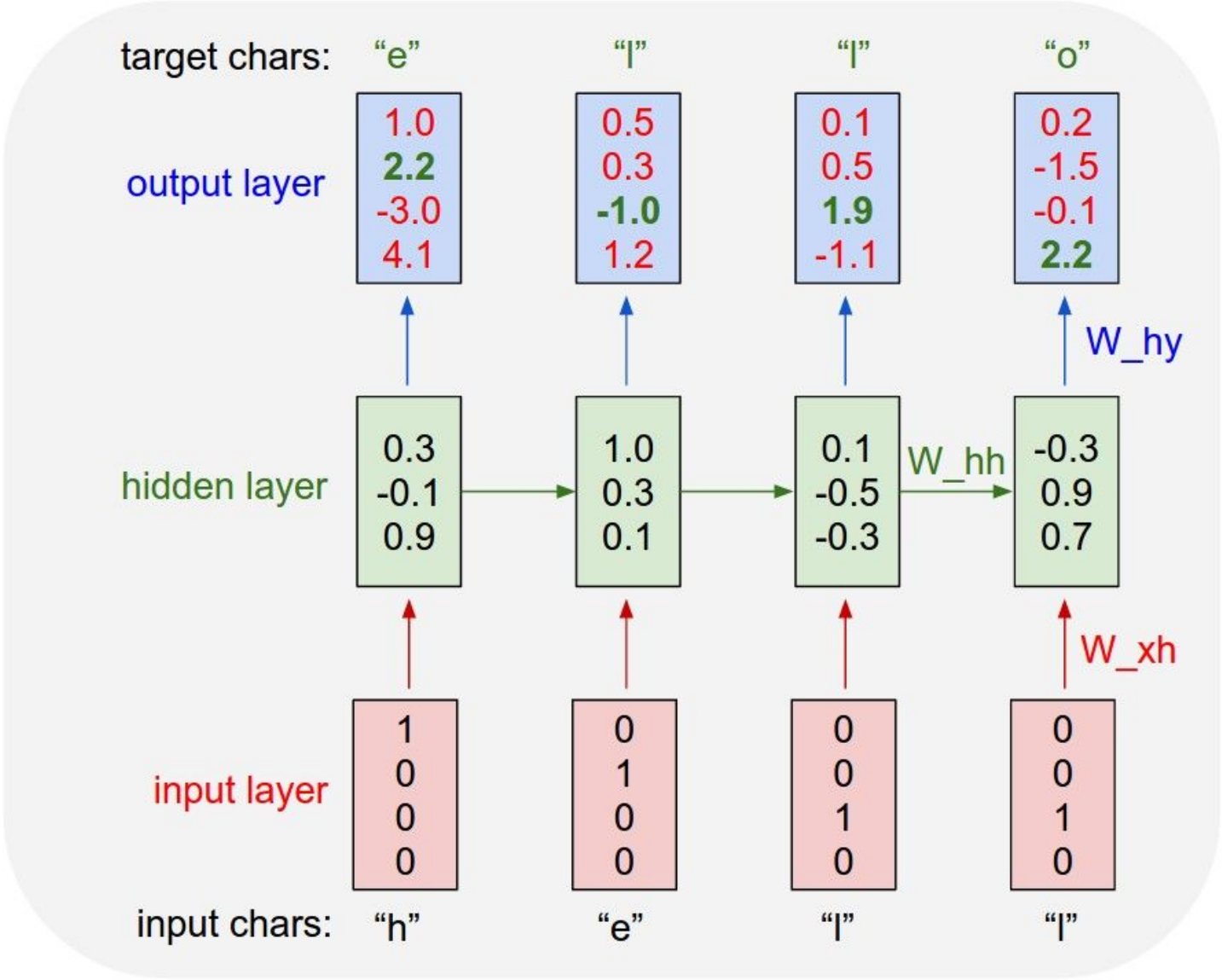


RNN: Матричная формулировка

Главное отличие RNN от FNN в наличии циклических связей (W_{hh} на рисунке)



Пример работы на генерации текста



http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Пример с RNN

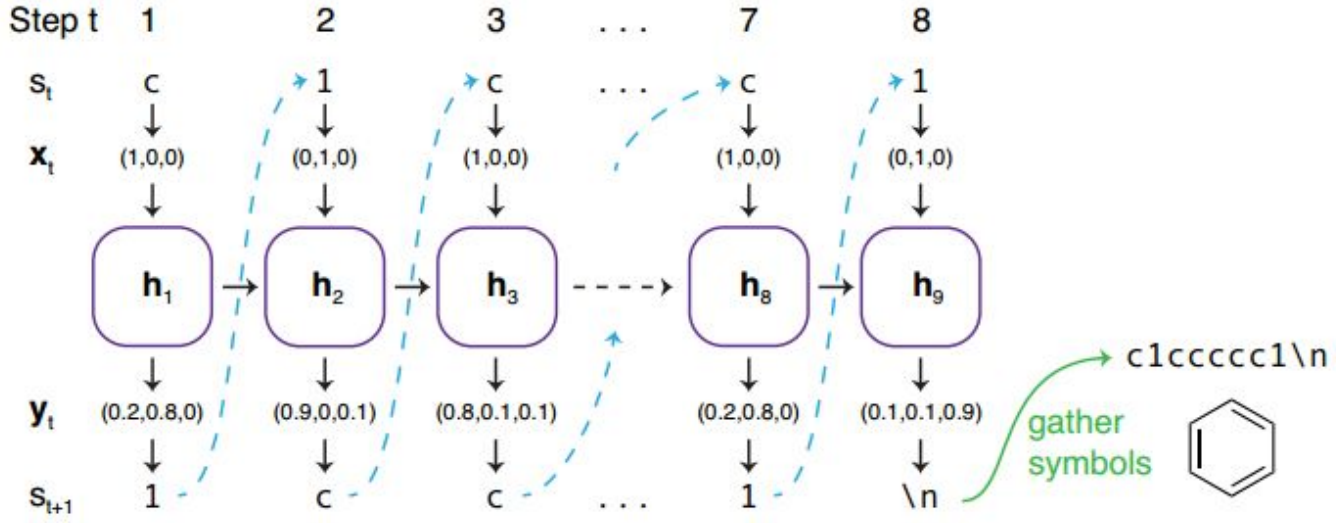
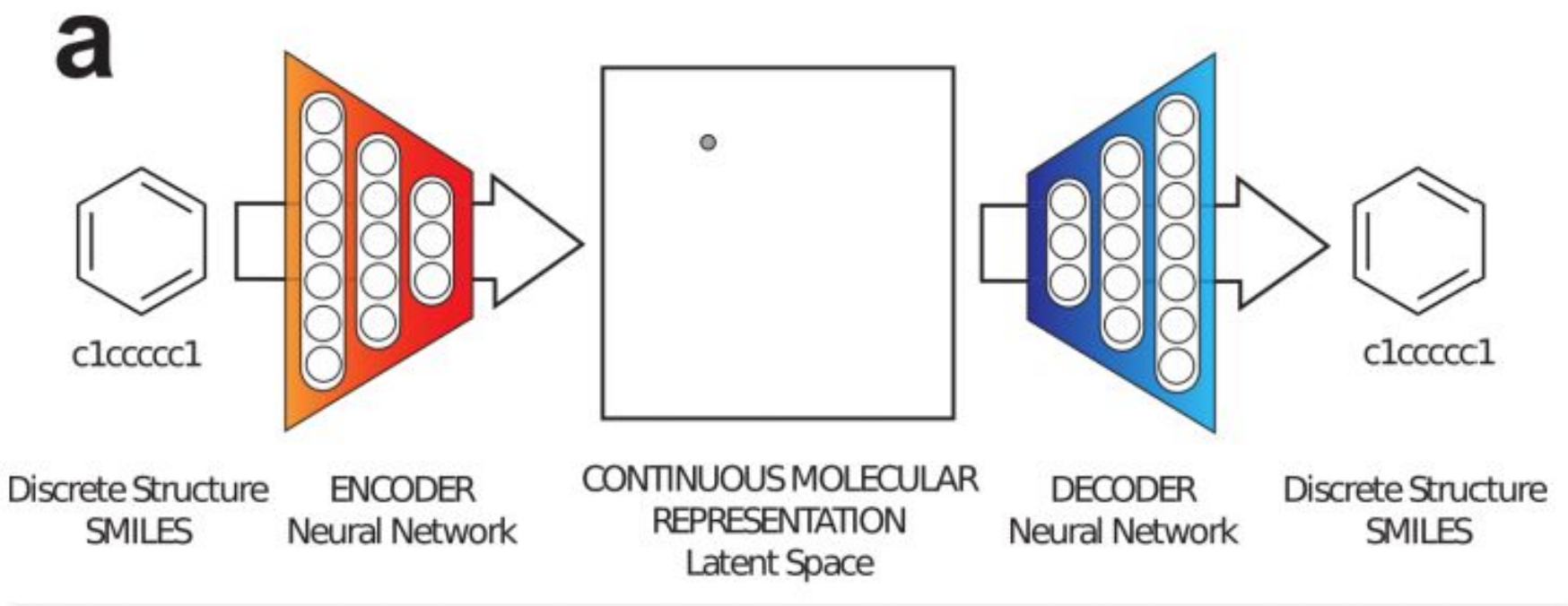


Figure 3 The Symbol Generation and Sampling Process. We start with a random seed symbol s_1, here c, which gets converted into a one-hot vector x_1 and input into the model. The model then updates its internal state h_0 to h_1 and outputs y_1, which is the probability distribution over the next symbols. Here, sampling yields $s_2 = 1$. Converting s_2 to x_2, and feeding it to the model leads to updated hidden state h_2 and output y_2, from which can sample again. This iterative symbol-by-symbol procedure can be continued as long as desired. In this example, we stop it after observing an EOL (\n) symbol, and obtain the SMILES for benzene. The hidden state h_i allows the model to keep track of opened brackets and rings, to ensure that they will be closed again later.

Batch	Generated Example	valid
0	0c.BK5i%ur+7oAFc7L3T=F8B5e=n)CS6RCTAR((0VCp1CApb)	no
1000	0F=CCC20CCCC)C2)C1CNC2CCCCCCCCCCCCCCCCCCCCCC	no
2000	0=C(N)C(=0)N(c1occc10C)c2ccccc20C	yes
3000	0=C1C=2N(c3cc(ccc30C2CCC1)CCCc4cn(c5c(Cl)cccc54)C)C	yes

Generating Focussed Molecule Libraries for Drug Discovery with Recurrent Neural Networks, https://arxiv.org/abs/1701.01329

Пример с RNN (Recurrent Autoencoder)



Automatic chemical design using a data-driven continuous representation of molecules, https://arxiv.org/abs/1610.02415

Свойства нейросетей

Feedforward NN (FNN):

- FFN — это универсальный аппроксиматор: однослойная нейросеть с конечным числом нейронов может аппроксимировать непрерывную функцию на компактных подмножествах R^n (Теорема Цыбенко, универсальная теорема аппроксимации).
- FFN не имеют естественной возможности учесть порядок во времени.
- FFN не обладают памятью, кроме полученной во время обучения.

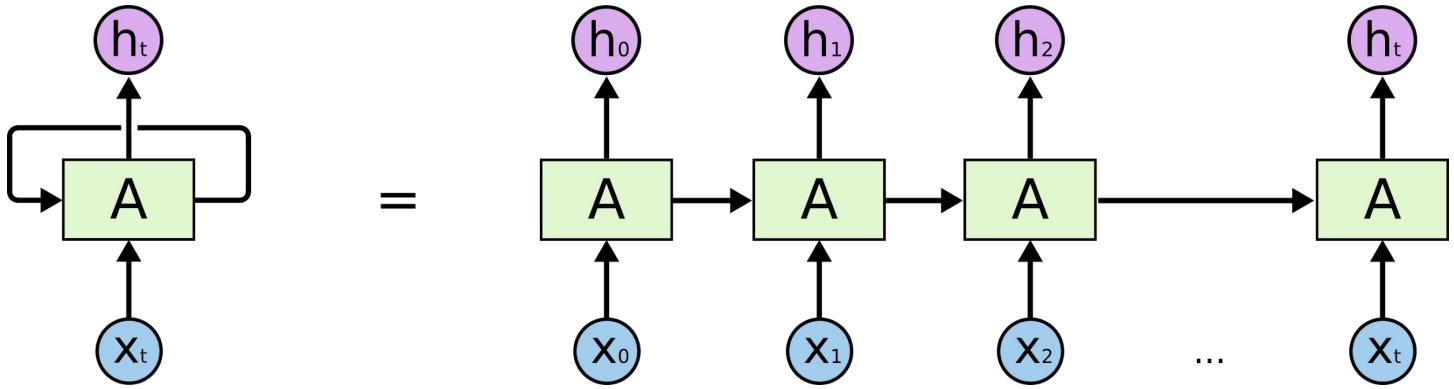
Recurrent NN (RNN):

- RNN Тьюринг-полны: можно реализовать любую вычислимую функцию.
- RNN обладают определённым видом памяти и гораздо лучше подходят для работы с последовательностями, моделированием контекста и временными зависимостями.

Backpropagation through time (BPTT)

Для обучения RNN используется специальный вариант backpropagation: (backpropagation through time, BPTT) и “разворачивание” нейросети.

Из-за этого есть проблема с затуханием градиентов при большой глубине. Для её решения вместо простых нейронов используют более сложные ячейки памяти — LSTM или GRU.



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Unfolding the RNN and training using BPTT

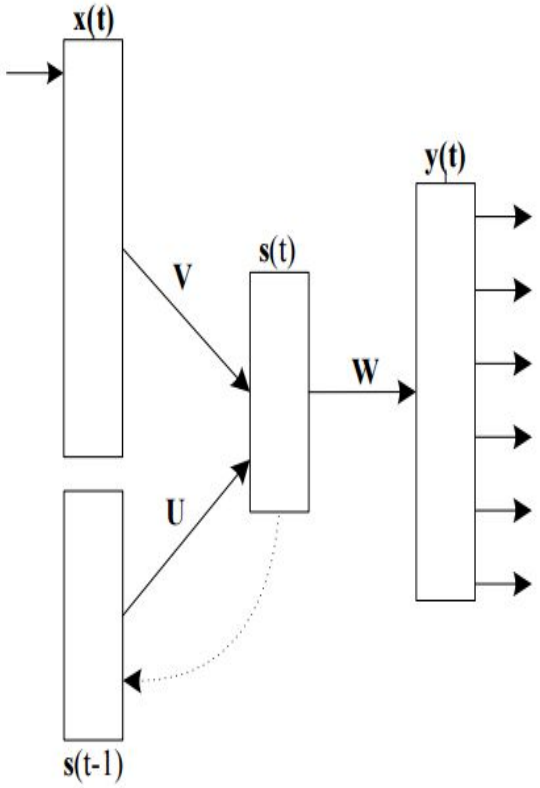


Figure 1: A simple recurrent neural network.

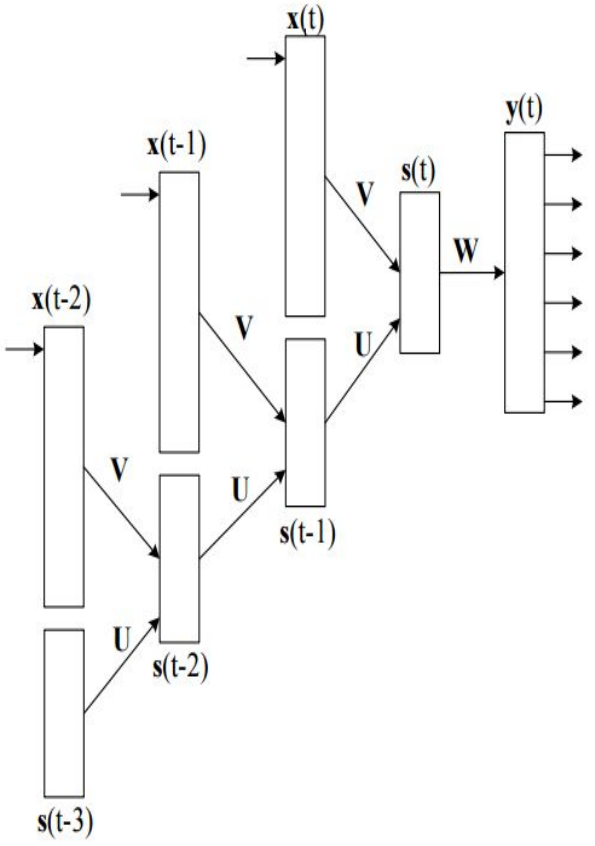


Figure 2: An unfolded recurrent neural network.

Can do backprop on the unfolded network: Backpropagation through time (BPTT)

http://ir.hit.edu.cn/~jguo/docs/notes/bptt.pdf

RNN problem: Vanishing gradients

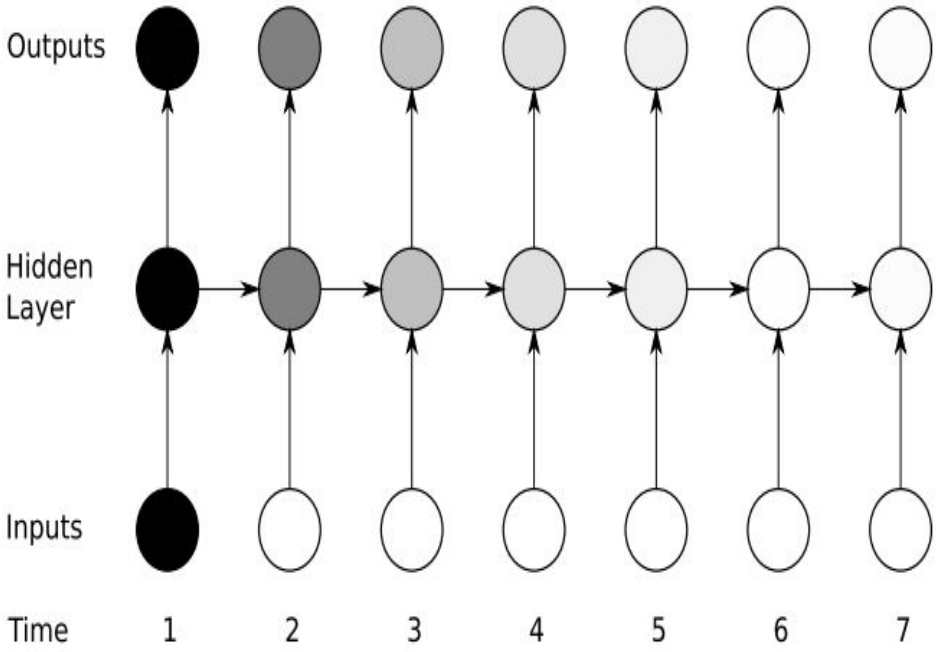
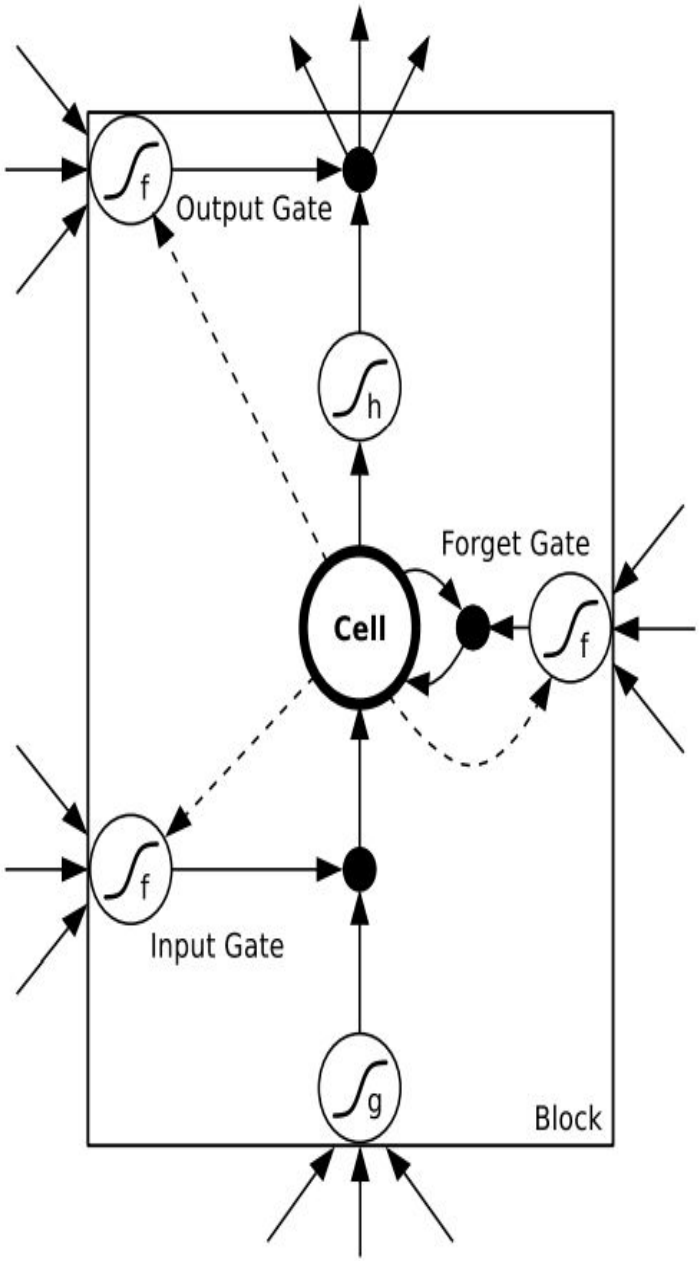


Figure 4.1: **The vanishing gradient problem for RNNs.** The shading of the nodes in the unfolded network indicates their sensitivity to the inputs at time one (the darker the shade, the greater the sensitivity). The sensitivity decays over time as new inputs overwrite the activations of the hidden layer, and the network 'forgets' the first inputs.

Solution: Long short-term memory (LSTM, Hochreiter, Schmidhuber, 1997)

LSTM cell



LSTM network

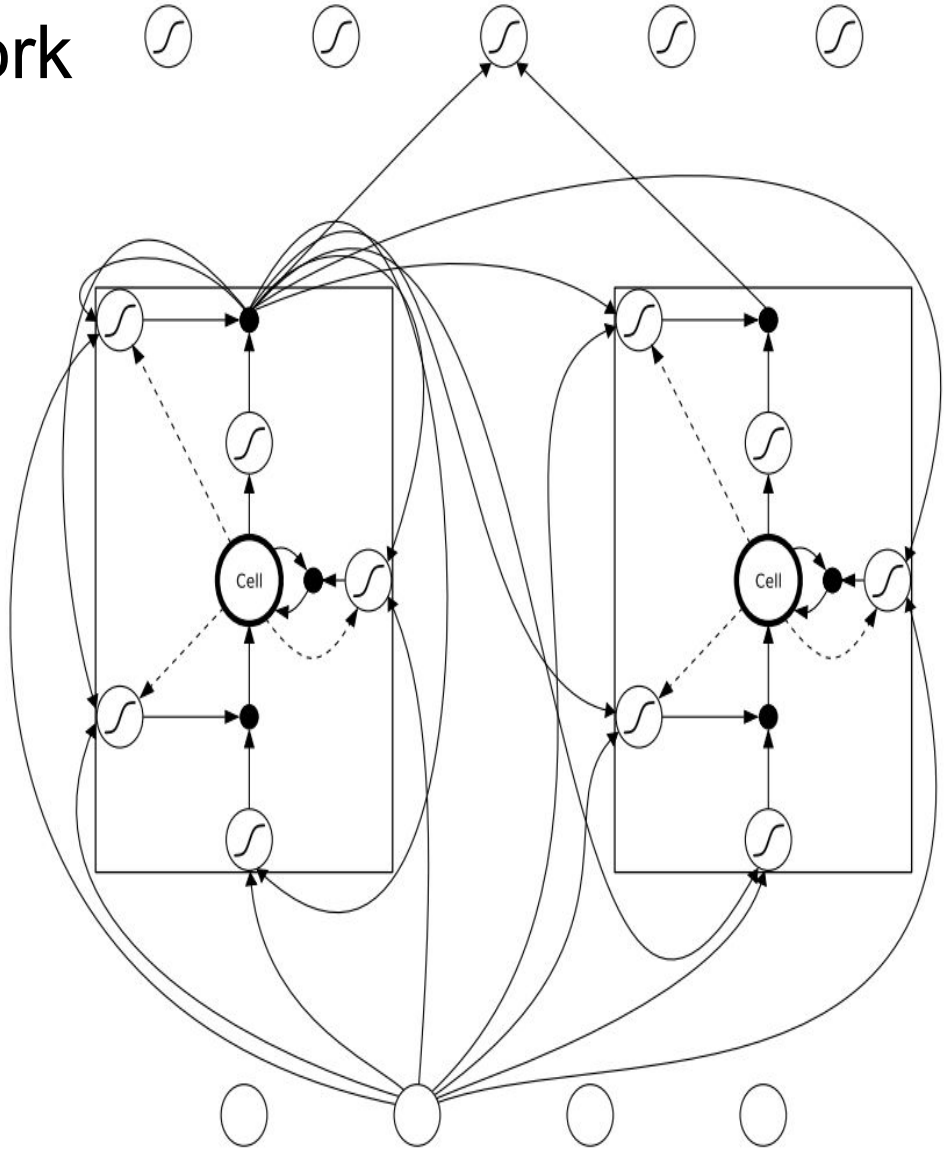


Figure 4.3: **An LSTM network.** The network consists of four input units, a hidden layer of two single-cell LSTM memory blocks and five output units. Not all connections are shown. Note that each block has four inputs but only one output.

Сравнение ячейки LSTM и обычного нейрона

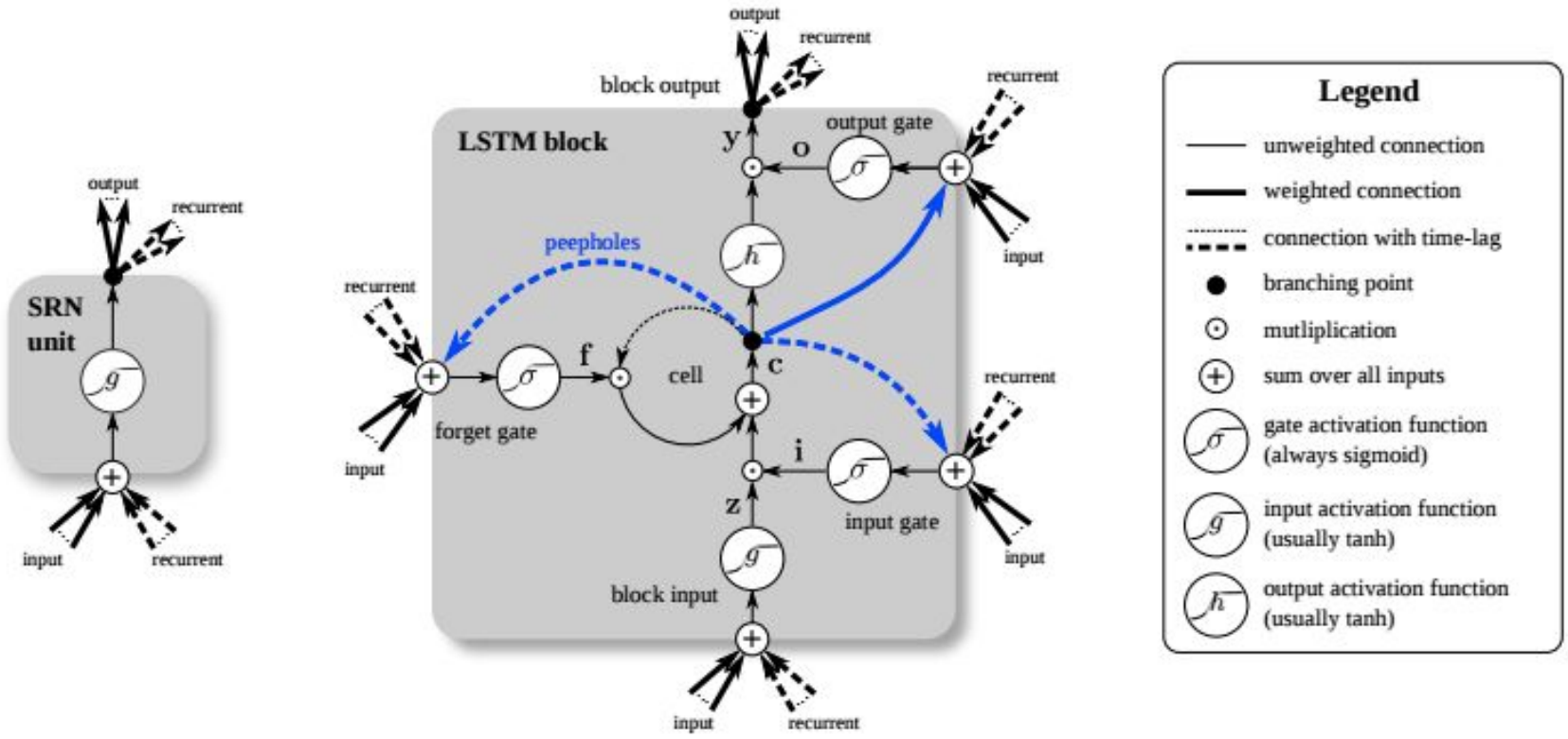


Figure 1. Detailed schematic of the Simple Recurrent Network (SRN) unit (left) and a Long Short-Term Memory block (right) as used in the hidden layers of a recurrent neural network.

https://deeplearning4j.org/lstm

LSTM: Fixing vanishing gradient problem

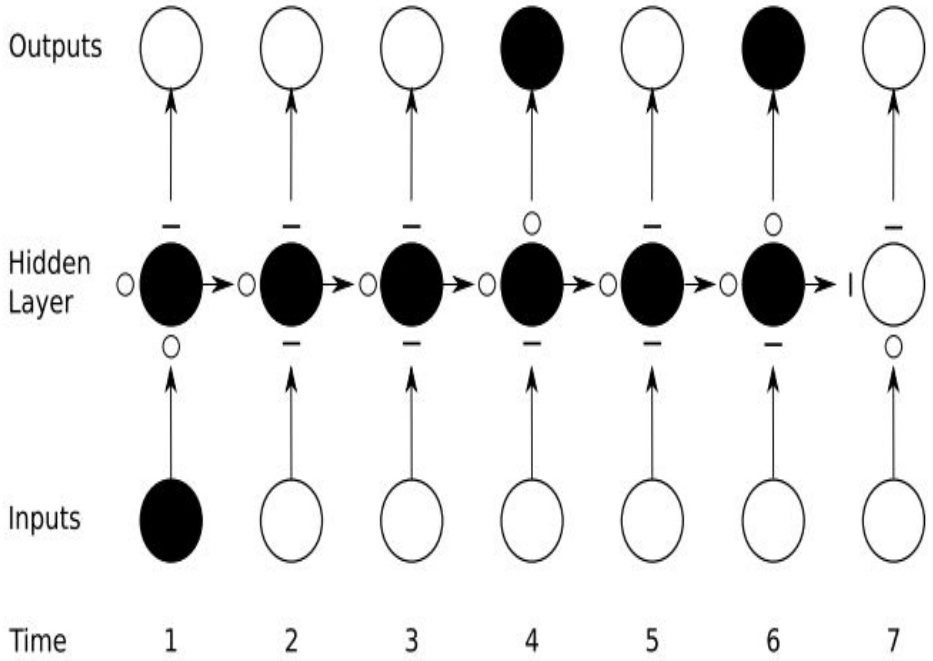
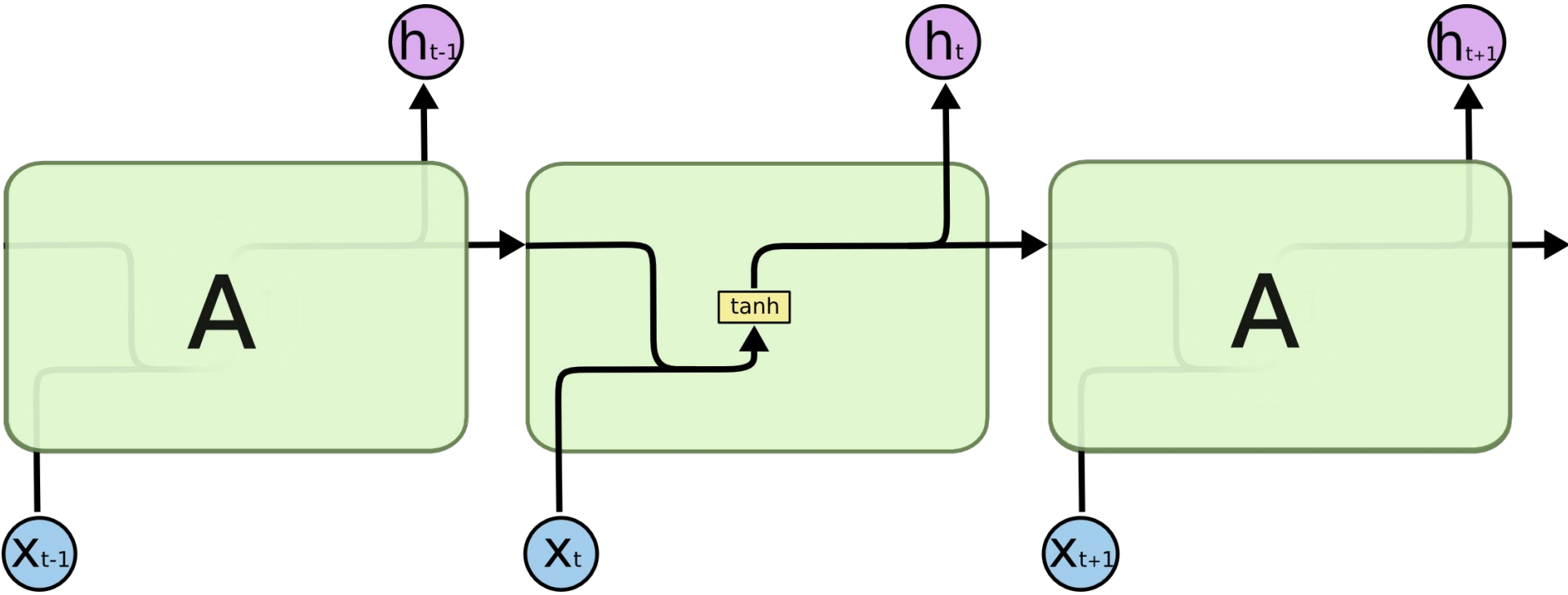


Figure 4.4: **Preservation of gradient information by LSTM.** As in Figure 4.1 the shading of the nodes indicates their sensitivity to the inputs at time one; in this case the black nodes are maximally sensitive and the white nodes are entirely insensitive. The state of the input, forget, and output gates are displayed below, to the left and above the hidden layer respectively. For simplicity, all gates are either entirely open ('O') or closed ('—'). The memory cell 'remembers' the first input as long as the forget gate is open and the input gate is closed. The sensitivity of the output layer can be switched on and off by the output gate without affecting the cell.

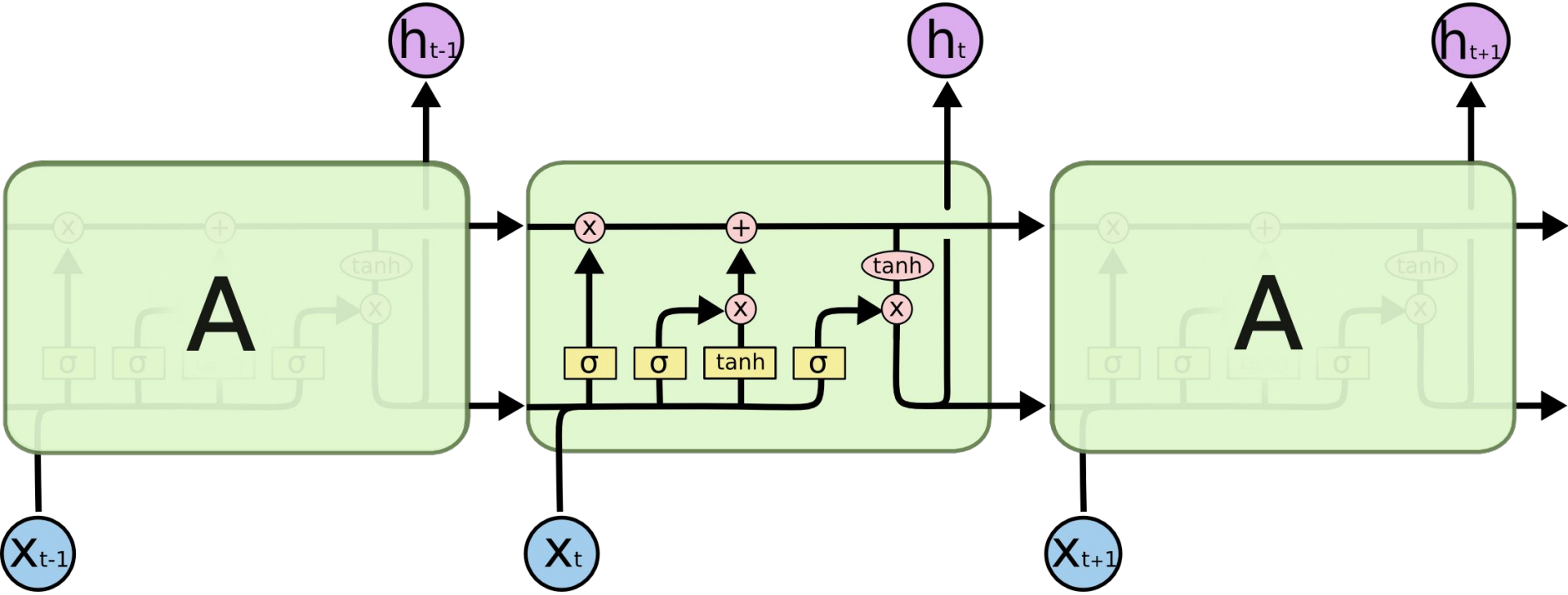
Разбор работы LSTM

Обычная RNN



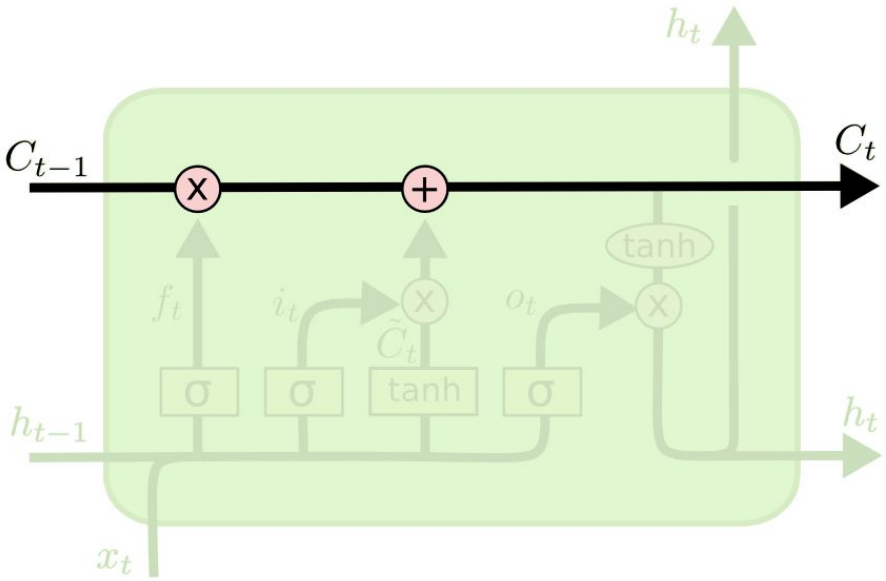
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

RNN с LSTM-ячейками



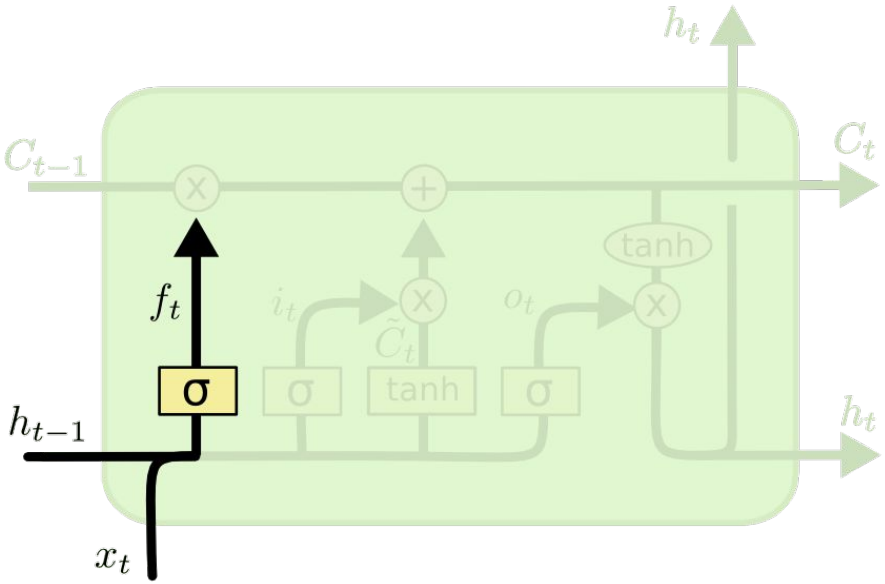
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Состояние ячейки памяти



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

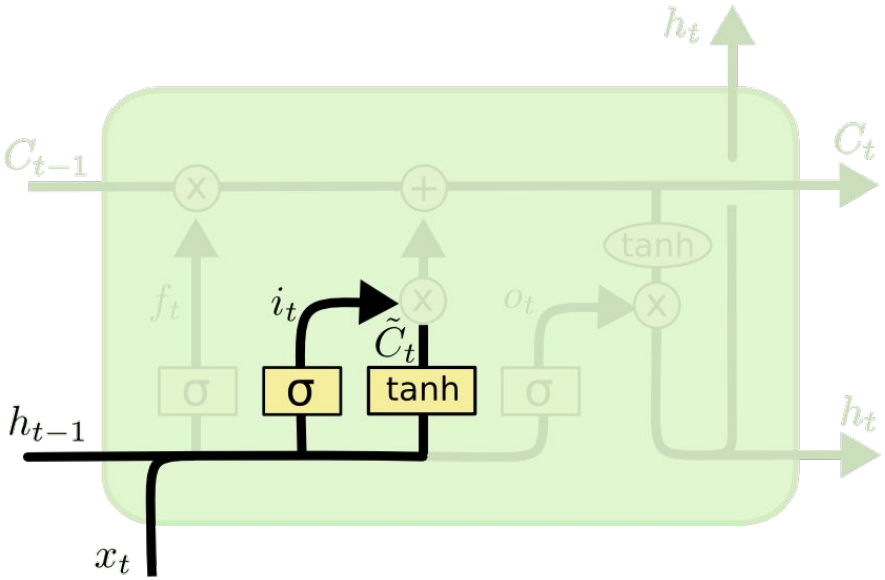
Forget-gate



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Input-gate

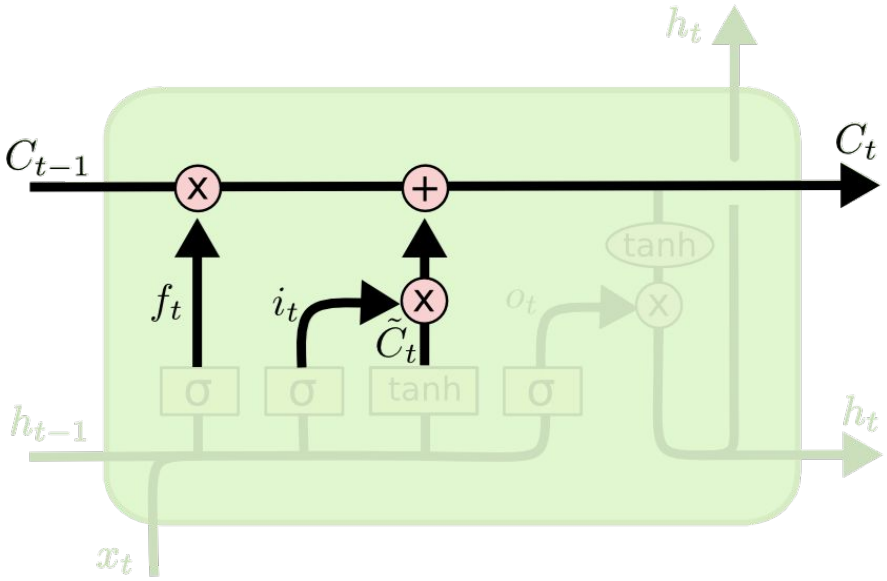


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

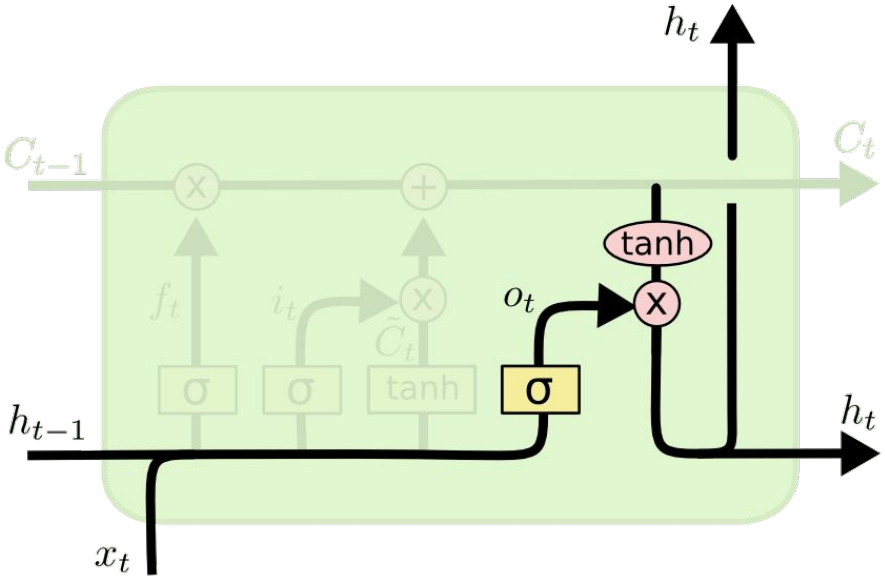
Обновление состояния памяти



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Выход ячейки

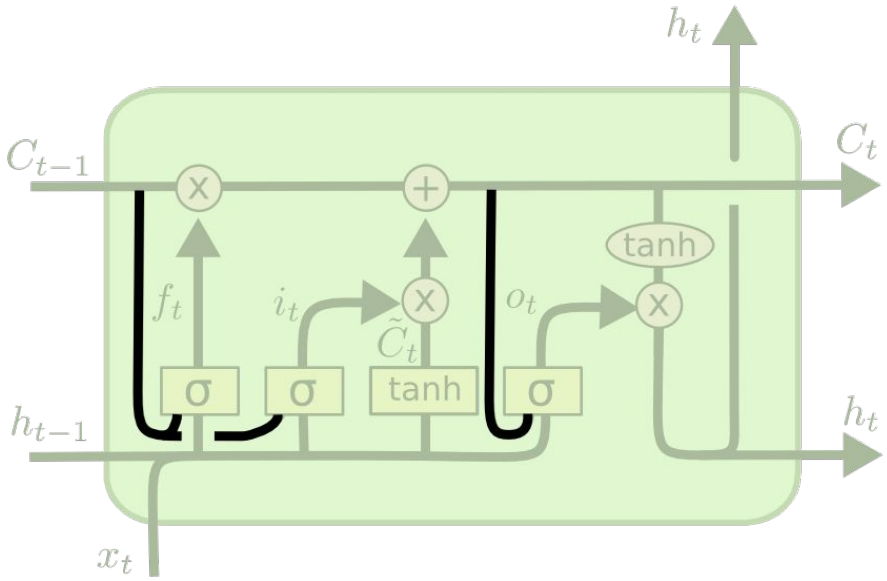


$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

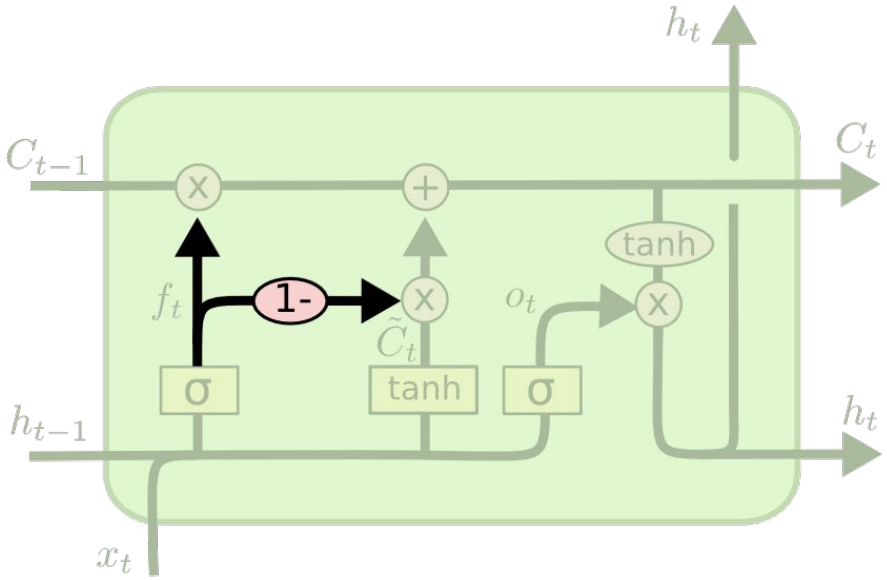
LSTM c “peephole connections”



$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \; + \; b_o\right)$$

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

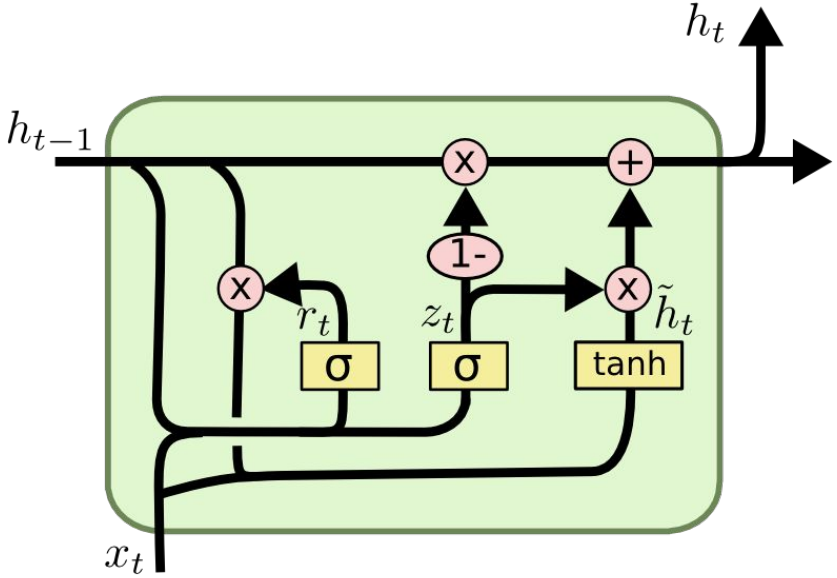
Coupled forget and input gates



$$C_t = f_t * C_{t-1} + (1 - \boldsymbol{f_t}) * \tilde{C}_t$$

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Gated Recurrent Unit (GRU)



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

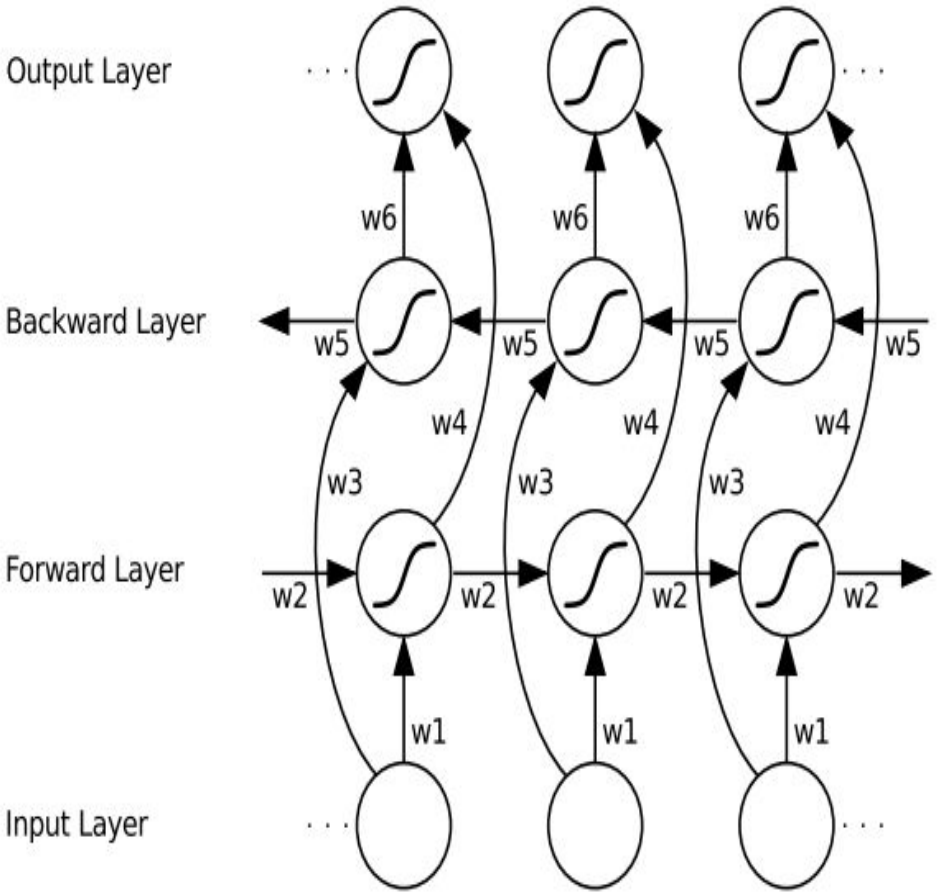
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Advanced RNN

Bidirectional RNN (BRNN/BLSTM)

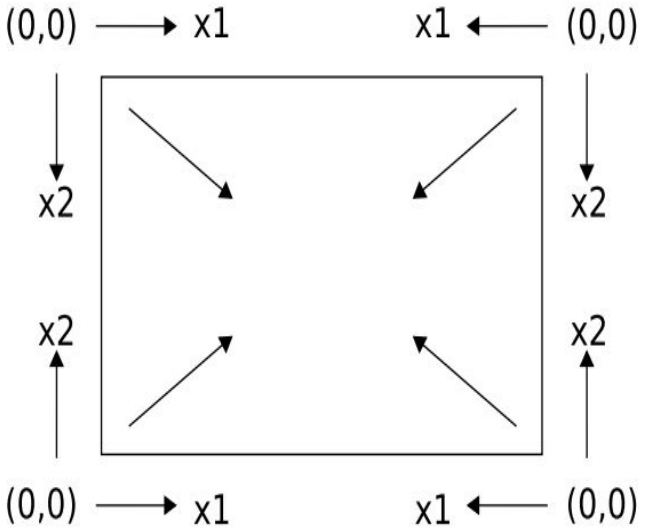
Часто последовательность доступна сразу целиком, так что её можно сканировать в обоих направлениях.

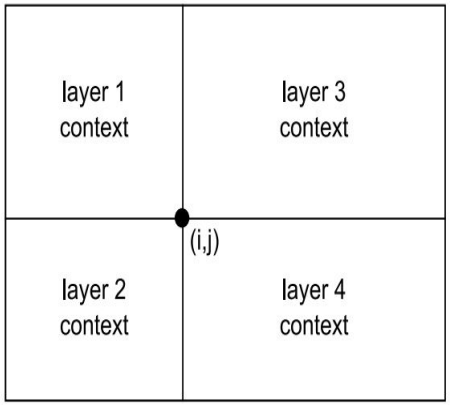


Multidimensional and Multidirectional RNNs

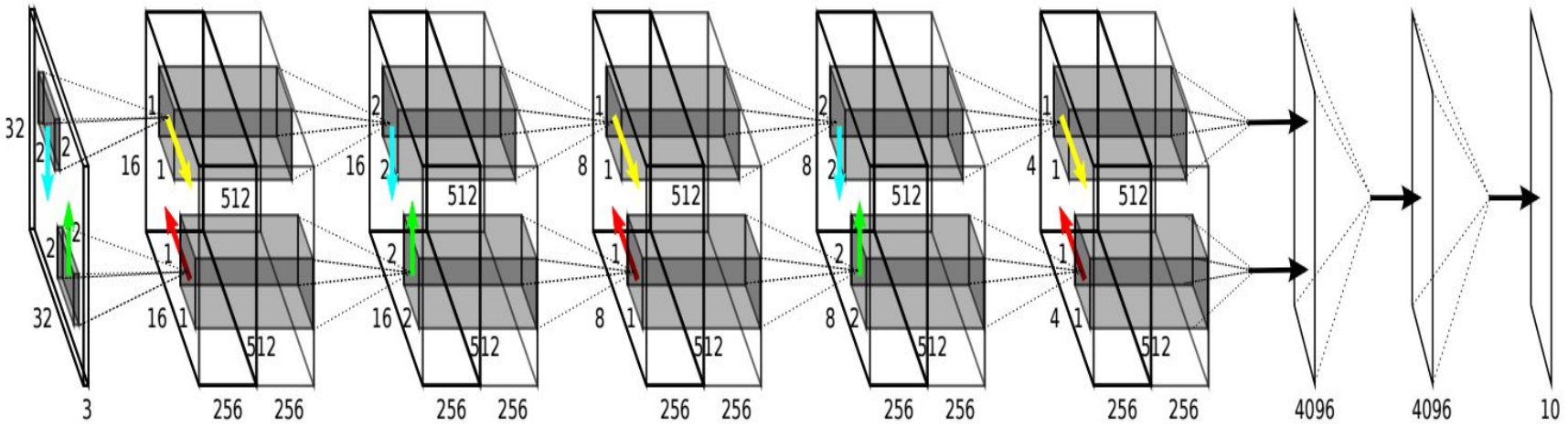
RNN могут также быть многомерными и многонаправленными.

Это более естественно для многомерных данных, например, изображений.

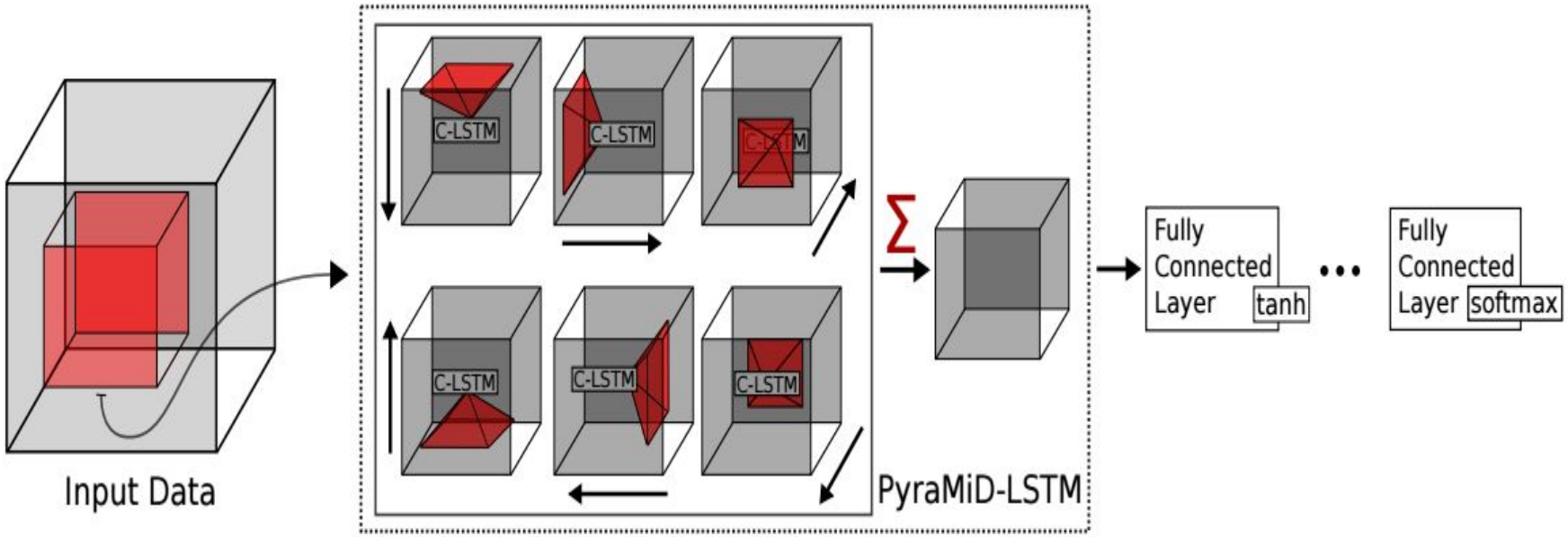




ReNet (2015)



PyraMiD-LSTM (2015)



Grid LSTM (2016)

Интересное многомерное обобщение LSTM: Grid LSTM

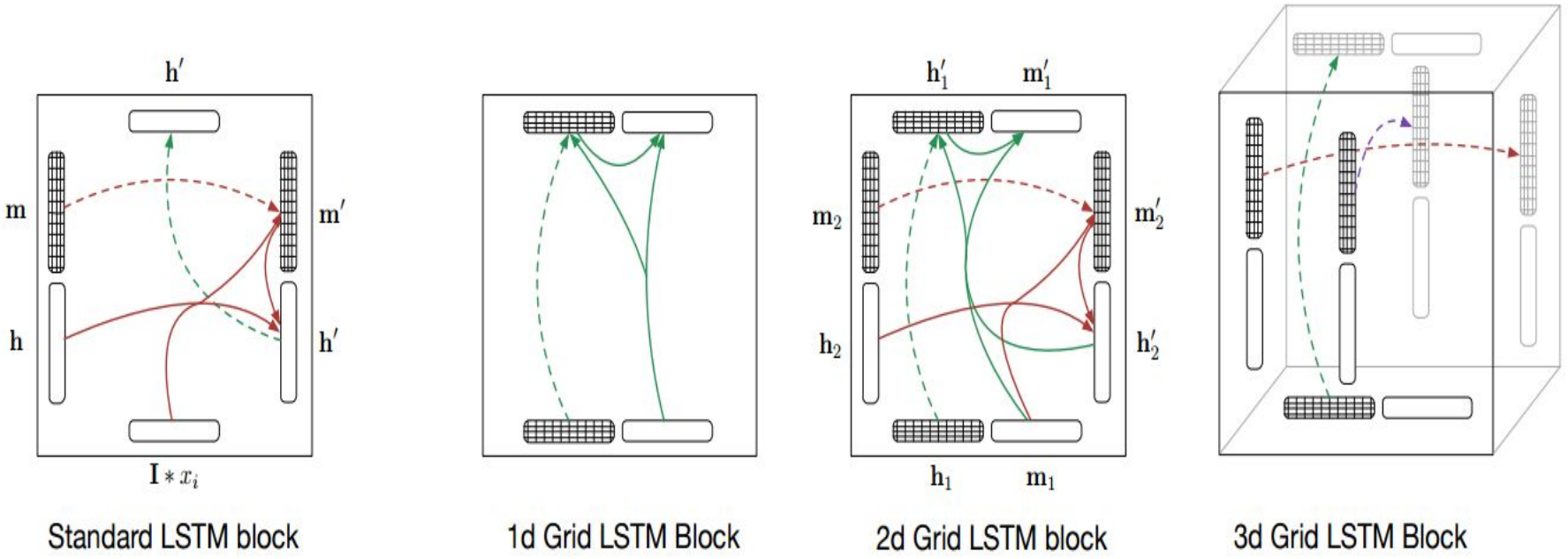
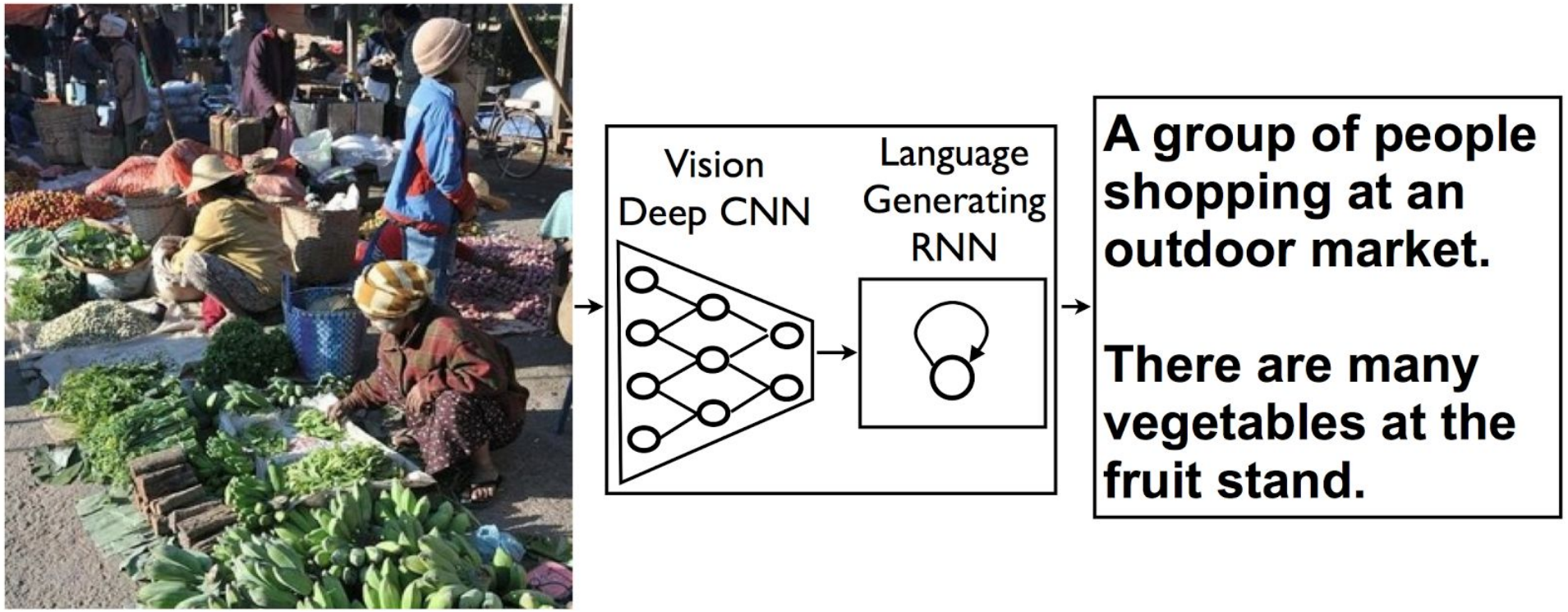


Figure 1: Blocks form the standard LSTM and those that form Grid LSTM networks of $N = 1, 2$ and 3 dimensions. The dashed lines indicate identity transformations. The standard LSTM block does not have a memory vector in the vertical dimension; by contrast, the 2d Grid LSTM block has the memory vector $\mathbf{m}_1$ applied along the vertical dimension.

https://arxiv.org/abs/1507.01526

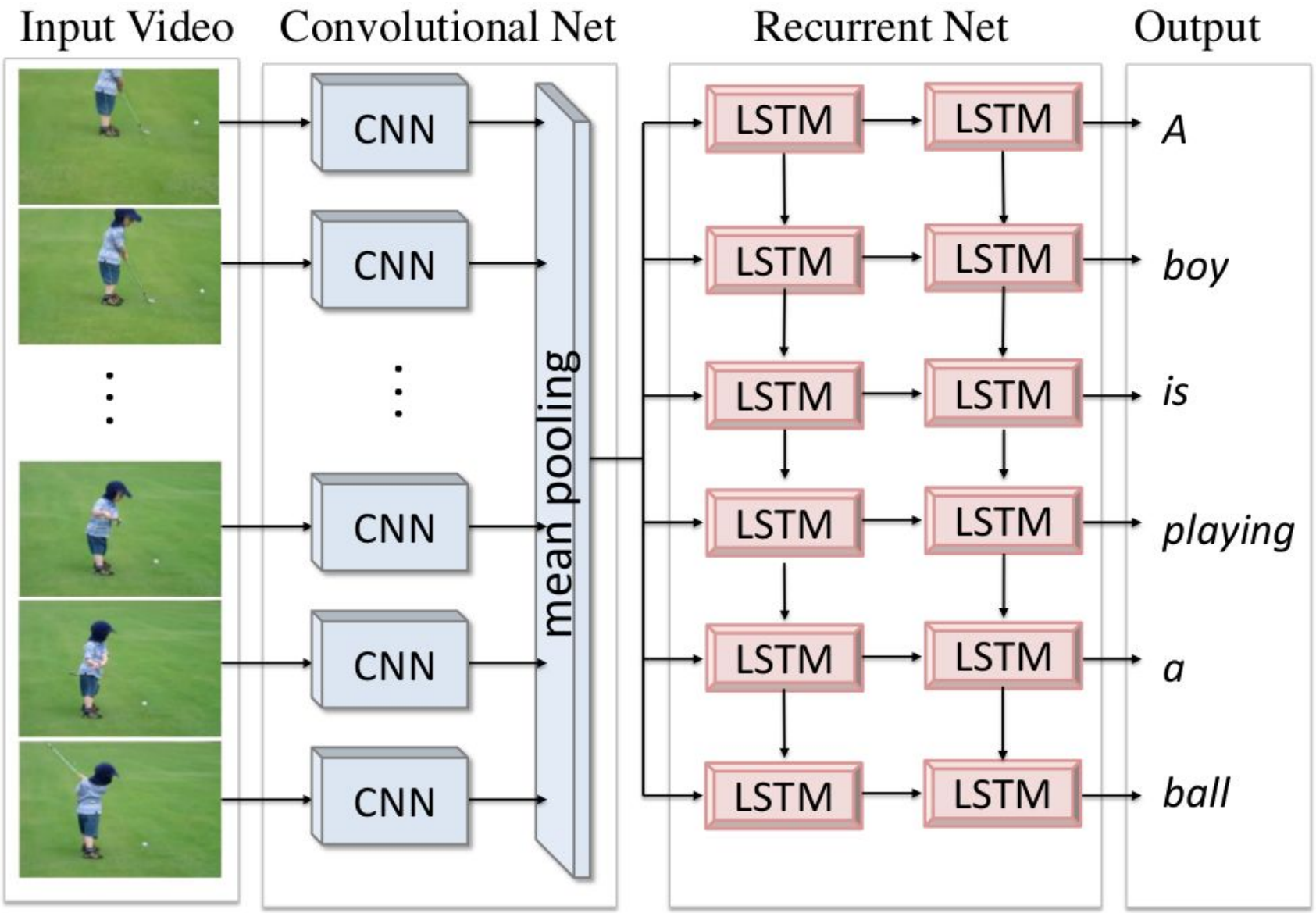
Мультимодальное обучение (Multimodal Learning)

Генерация описаний картинок



“Show and Tell: A Neural Image Caption Generator”
http://arxiv.org/abs/1411.4555

Мультимодальное обучение



https://www.cs.utexas.edu/~vsub/

Мультимодальное обучение

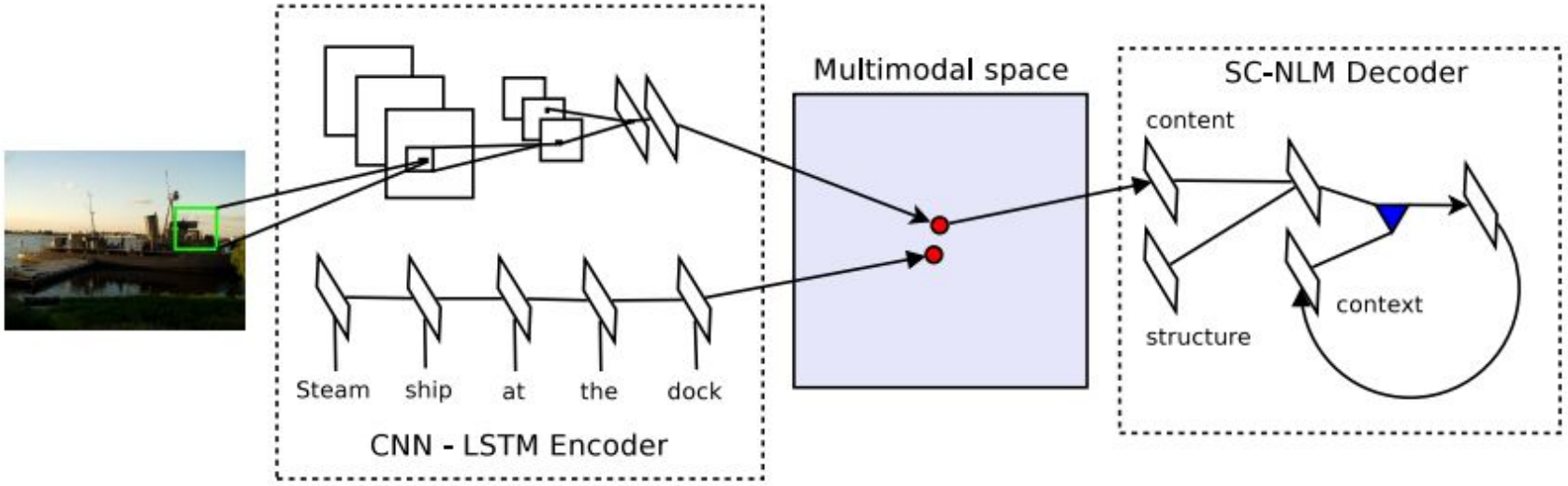
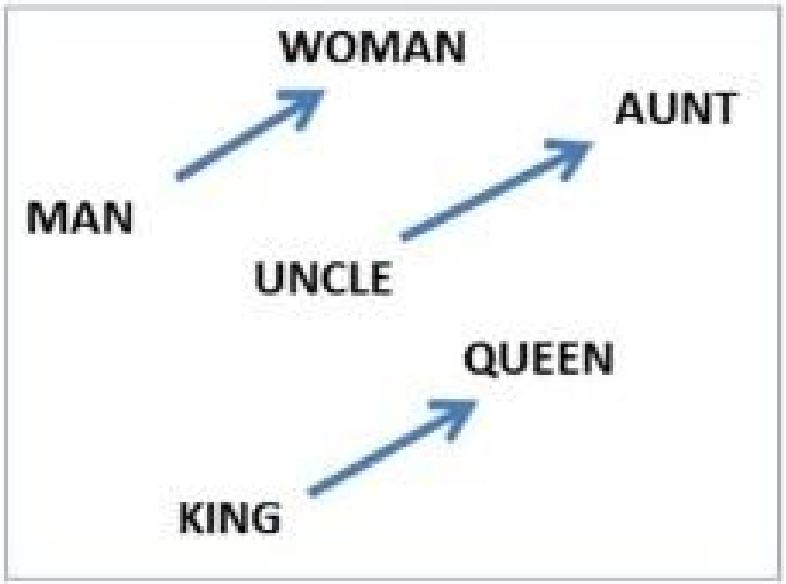


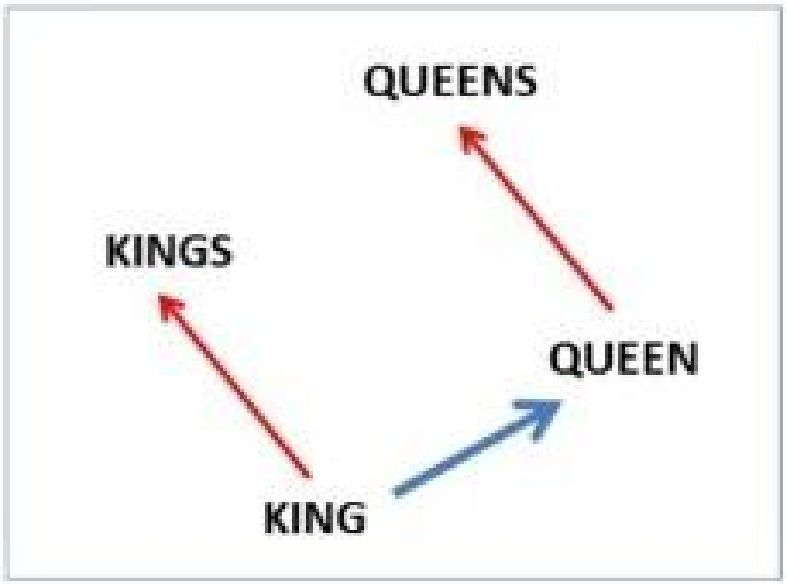
Figure 2: **Encoder:** A deep convolutional network (CNN) and long short-term memory recurrent network (LSTM) for learning a joint image-sentence embedding. **Decoder:** A new neural language model that combines structure and content vectors for generating words one at a time in sequence.

Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models
http://arxiv.org/abs/1411.2539

Example: Semantic Spaces (word2vec, GloVe)

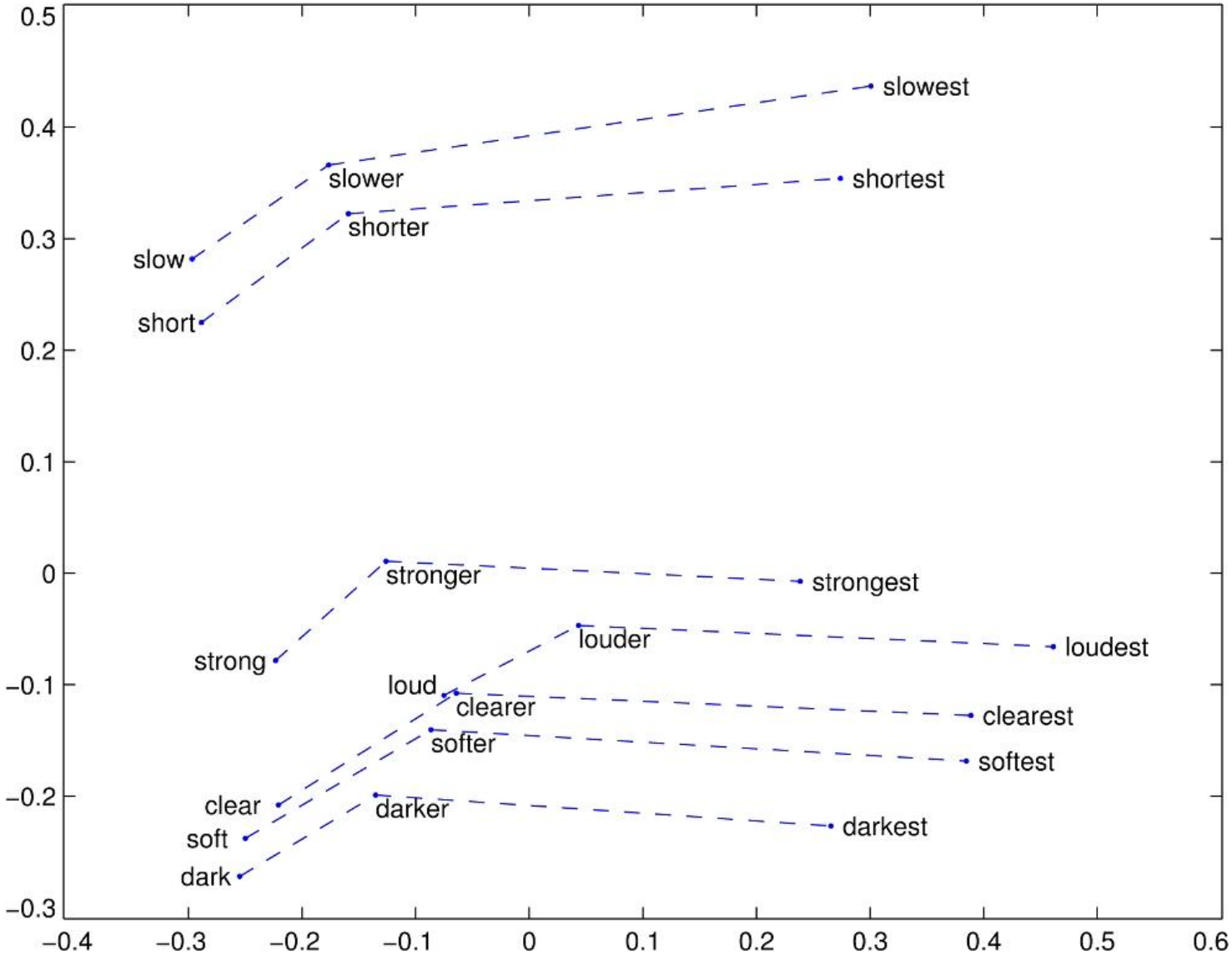
vec("man") – vec("king") + vec("woman") = vec("queen")





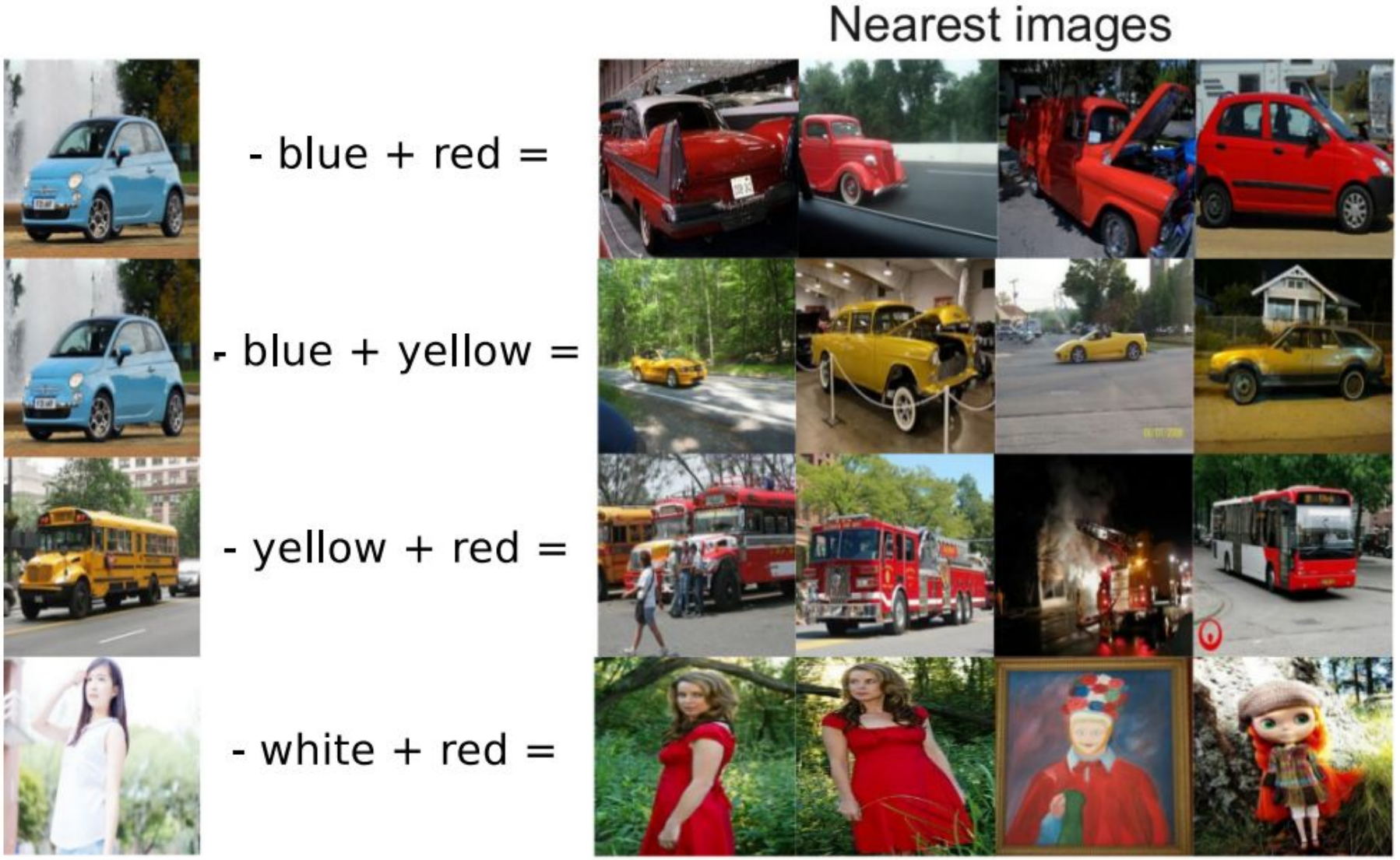
https://code.google.com/archive/p/word2vec/

Example: Semantic Spaces (word2vec, GloVe)



http://nlp.stanford.edu/projects/glove/

Example: More multi-modal learning



Nearest Images

- day + night =

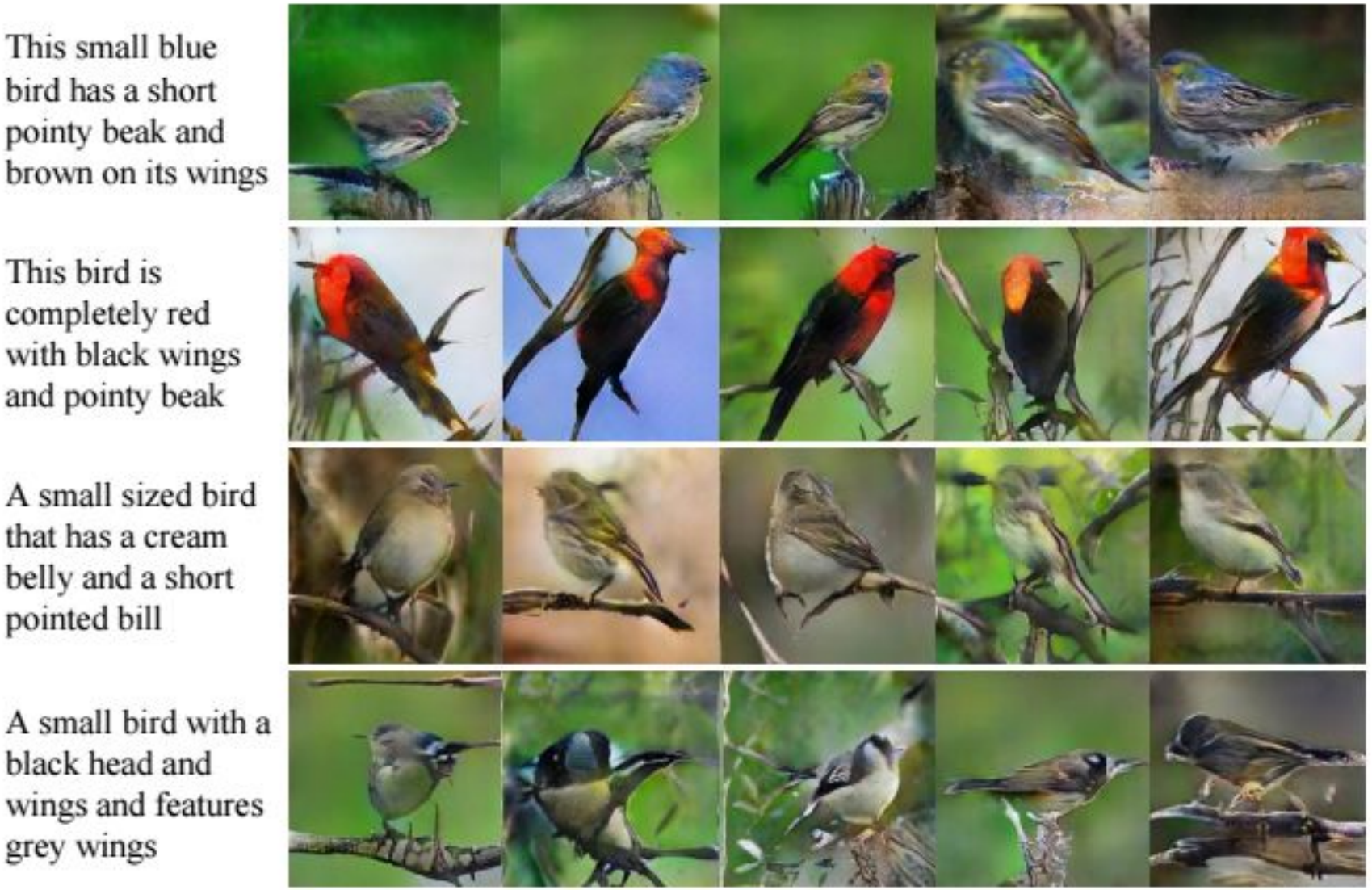
- flying + sailing =

- bowl + box =

- box + bowl =

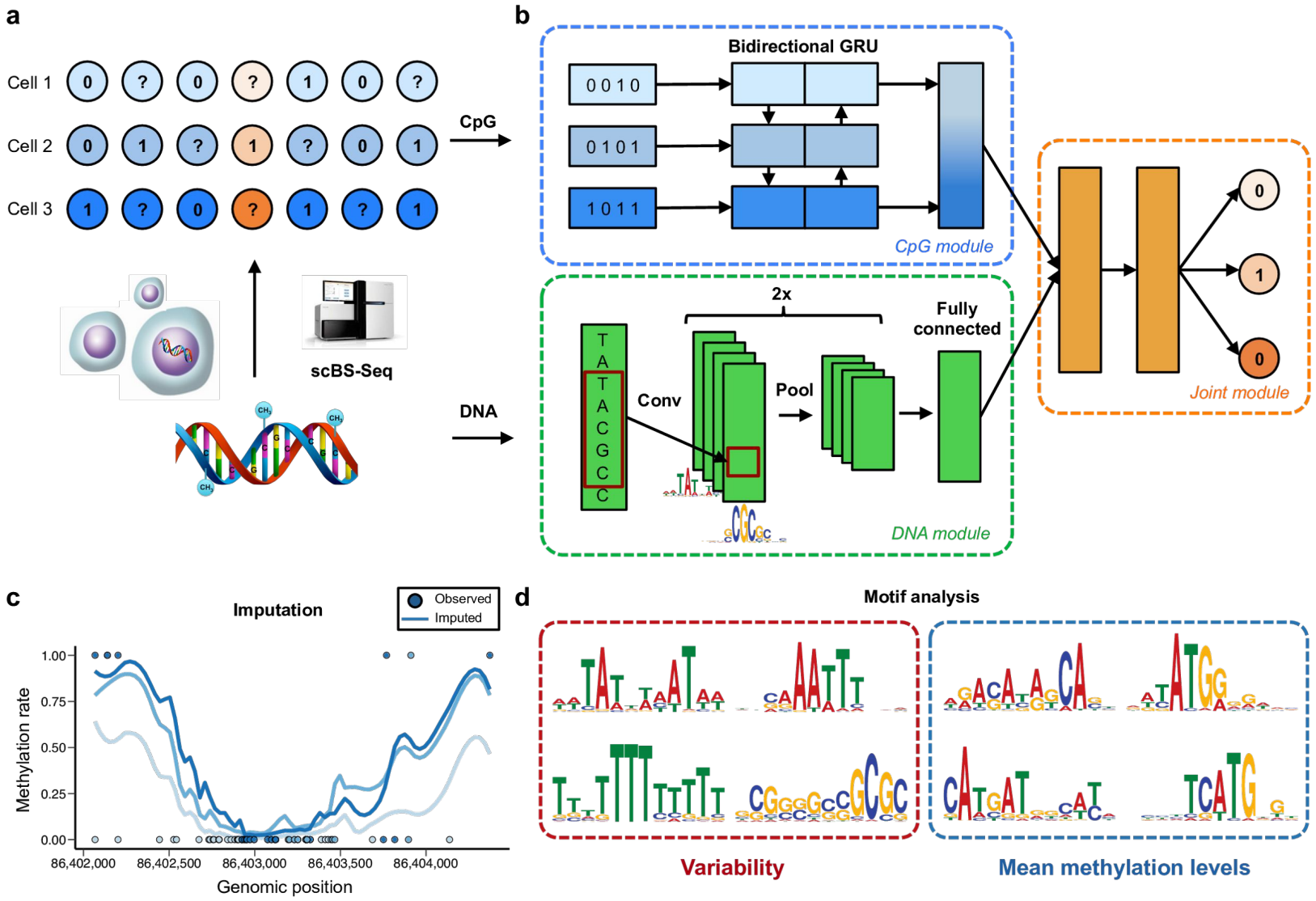
(Kiros, Salakhutdinov, Zemel, TACL 2015)

Example: Image generation by text



StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks, https://arxiv.org/abs/1612.03242

Мультимодальное обучение

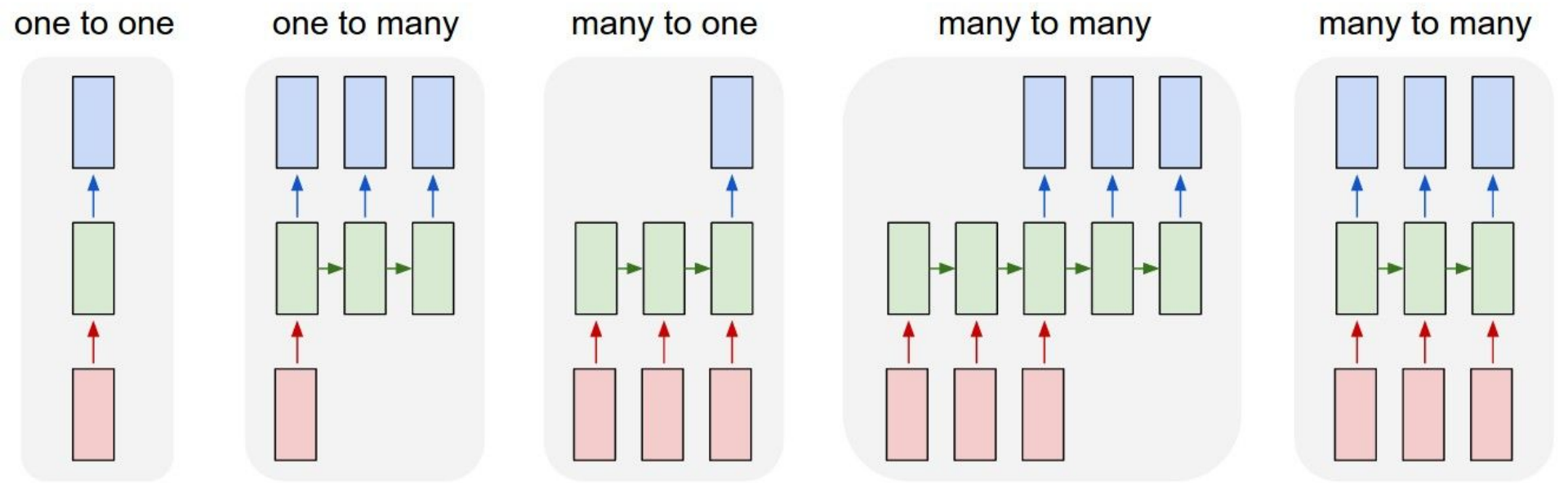


Accurate prediction of single-cell DNA methylation states using deep learning

http://biorxiv.org/content/early/2017/02/01/055715

Sequence Learning и парадигма seq2seq

Sequence to Sequence Learning (seq2seq)

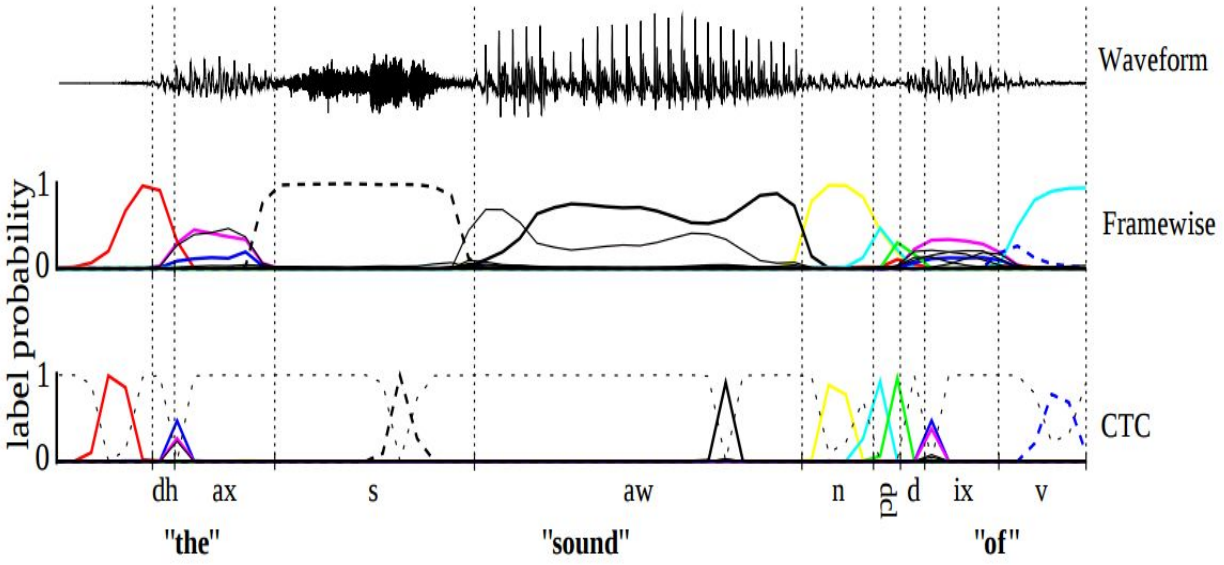


http://karpathy.github.io/2015/05/21/rnn-effectiveness/

CTC (Connectionist Temporal Classification)

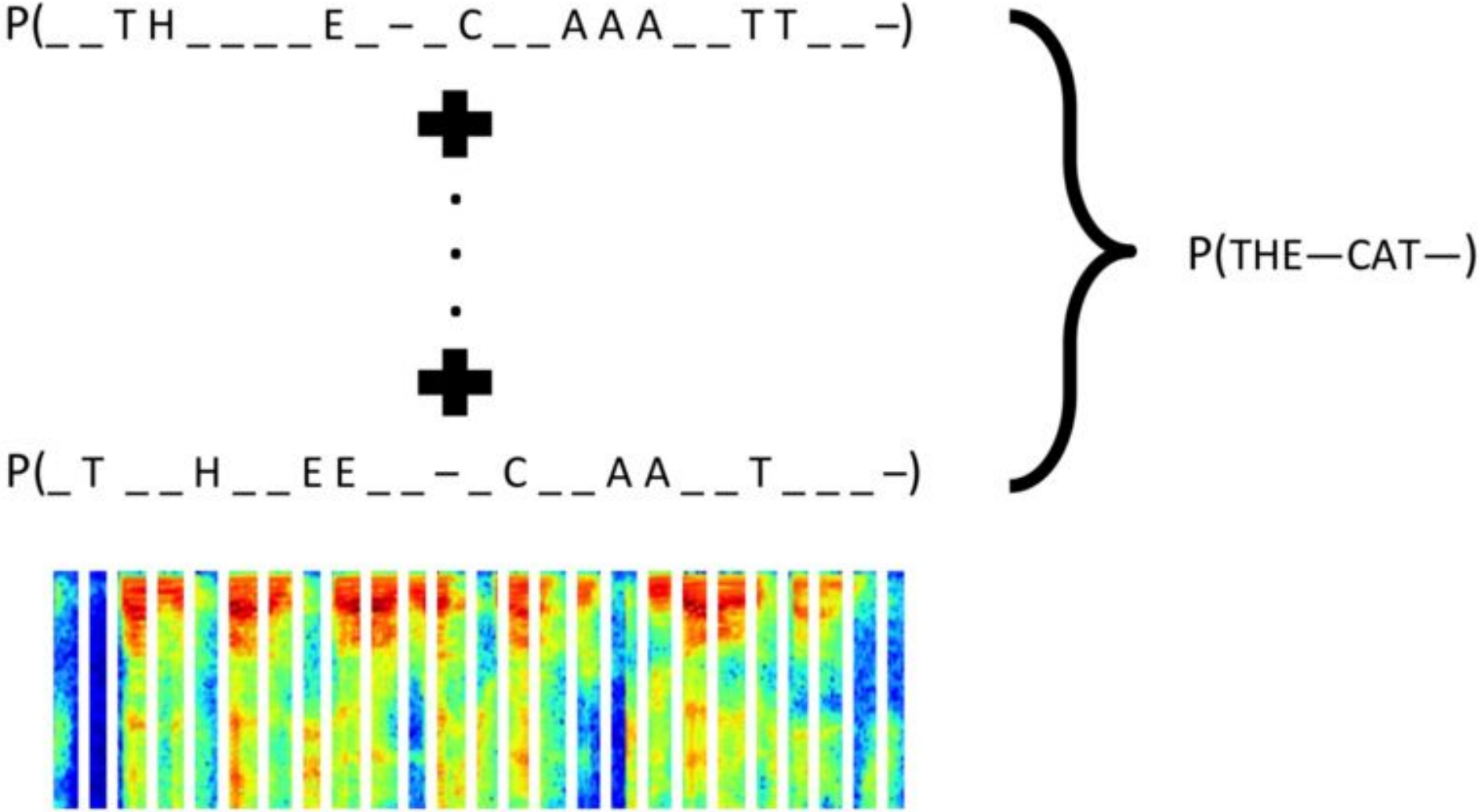
Есть много задач, где точное расположение меток неважно, а важна только их последовательность. Например, в распознавании речи, рукописного текста, автомобильных номеров.

Специальный выходной слой CTC (Graves, Fernández, Gomez, Schmidhuber, 2006) был создан для временной классификации, когда выравнивание входных данных и выходных меток неизвестно и не требуется.



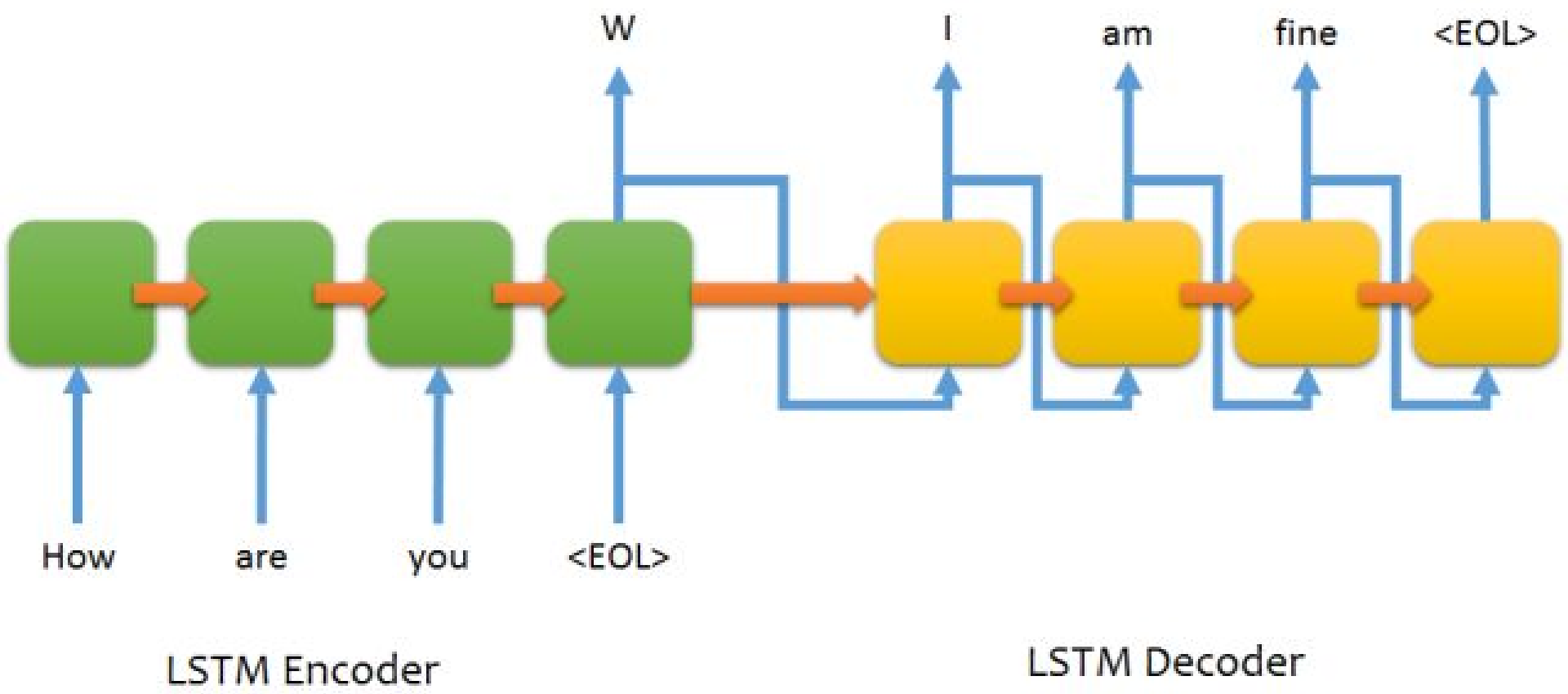
https://www.cs.toronto.edu/~graves/preprint.pdf

CTC (Connectionist Temporal Classification)



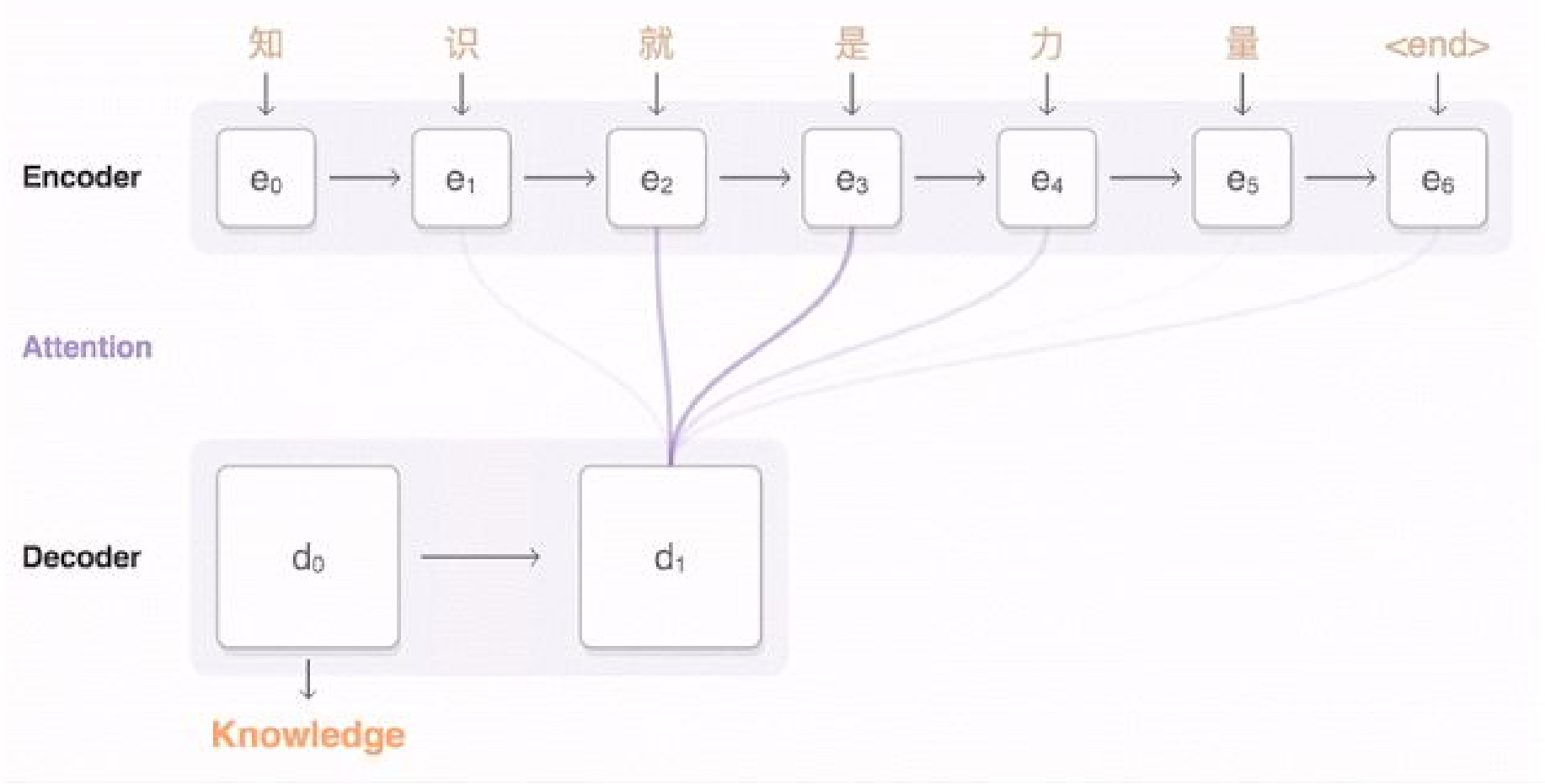
https://github.com/baidu-research/warp-ctc

Encoder-Decoder architecture



https://github.com/farizrahman4u/seq2seq

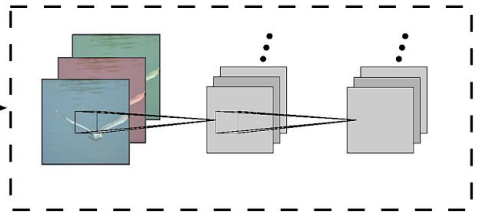
Encoder-Decoder with Attention

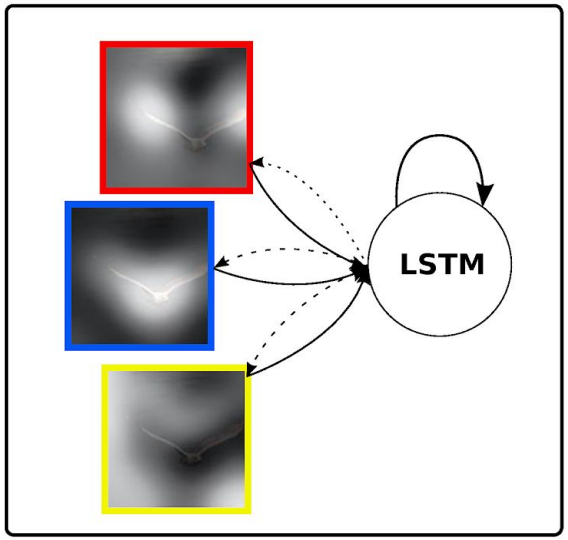


https://research.googleblog.com/2016/09/a-neural-network-for-machine.html

CNN+RNN with Attention

14x14 Feature Map





A
bird
flying
over
a
body
of
water

1. Input Image

2. Convolutional Feature Extraction

3. RNN with attention over the image

4. Word by word generation

http://kelvinxu.github.io/projects/capgen.html

CNN+RNN with Attention

A stop sign is on a road with a mountain in the background.

A woman is throwing a frisbee in a park.

http://kelvinxu.github.io/projects/capgen.html

Фреймворки и библиотеки для работы с нейросетями

Библиотеки и фреймворки

Подробный список: http://deeplearning.net/software_links/

Универсальные библиотеки и сервисы

- **Torch7, PyTorch** (http://torch.ch/) [Lua, Python]
- **TensorFlow** (https://www.tensorflow.org/) [Python, C++]
- **Theano** (http://deeplearning.net/software/theano/) [Python]
  - **Keras** (http://keras.io/)
  - **Lasagne** (https://github.com/Lasagne/Lasagne)
  - **blocks** (https://github.com/mila-udem/blocks)
  - **pylearn2** (https://github.com/lisa-lab/pylearn2)
- **Microsoft Cognitive Toolkit (CNTK)** (http://www.cntk.ai/) [Python, C++, C#, BrainScript]
- **Neon** (http://neon.nervanasys.com/) [Python]
- **Deeplearning4j** (http://deeplearning4j.org/) [Java]
- **MXNet** (http://mxnet.io/) [C++, Python, R, Scala, Julia, Matlab, Javascript]
- …

Обработка изображений и видео

- **OpenCV** (http://opencv.org/) [C, C++, Python]
- **Caffe** (http://caffe.berkeleyvision.org/) [C++, Python, Matlab]
- **Torch7** (http://torch.ch/) [Lua]
- **clarifai** (https://www.clarifai.com/)
- **Google Vision API** (https://cloud.google.com/vision/)
- …

Распознавание речи

- **Microsoft Cognitive Toolkit (CNTK)** (http://www.cntk.ai/) [Python, C++, C#, BrainScript]
- **KALDI** (http://kaldi-asr.org/) [C++]
- **Google Speech API** (https://cloud.google.com/)
- **Yandex SpeechKit** (https://tech.yandex.ru/speechkit/)
- **Baidu Speech API** (http://www.baidu.com/)
- **wit.ai** (https://wit.ai/)
- …

Обработка текстов

- **Torch7** (http://torch.ch/) [Lua]
- **Theano/Keras/**… [Python]
- **TensorFlow** (https://www.tensorflow.org/) [C++, Python]
- **Google Translate API** (https://cloud.google.com/translate/)
- …

Что читать и смотреть

- **CS231n: Convolutional Neural Networks for Visual Recognition**, Fei-Fei Li, Andrej Karpathy, Stanford (http://vision.stanford.edu/teaching/cs231n/index.html)
- **CS224d: Deep Learning for Natural Language Processing**, Richard Socher, Stanford (http://cs224d.stanford.edu/index.html)
- **Neural Networks for Machine Learning**, Geoffrey Hinton (https://www.coursera.org/course/neuralnets)
- **Подборка курсов по компьютерному зрению** (http://eclass.cc/courselists/111_computer_vision_and_navigation)
- **Подборка курсов по deep learning (**http://eclass.cc/courselists/117_deep_learning)
- "**Deep Learning**", Ian Goodfellow, Yoshua Bengio and Aaron Courville (http://www.deeplearningbook.org/)

Что читать и смотреть

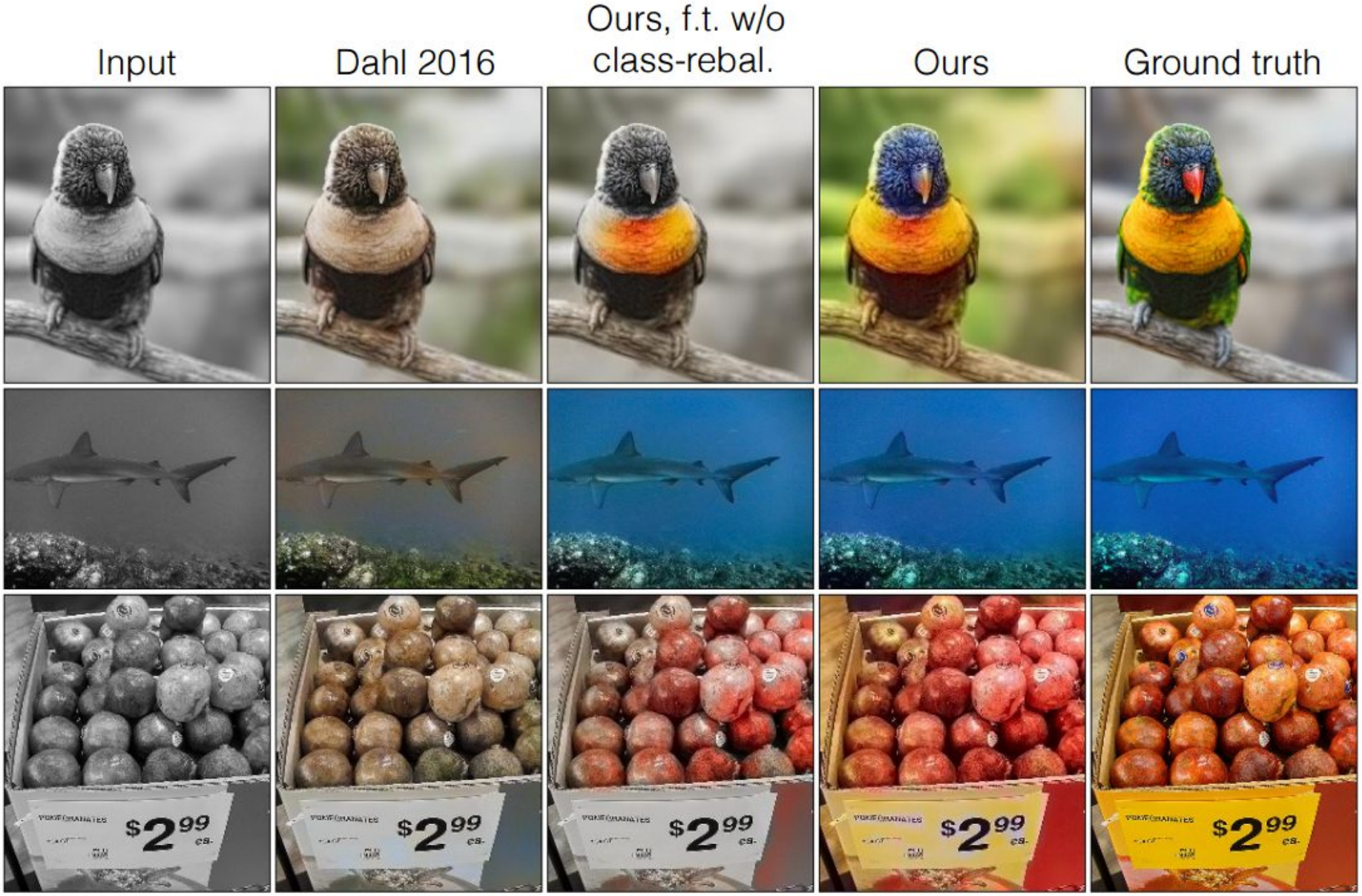
- **Google+ Deep Learning community** (https://plus.google.com/communities/112866381580457264725)
- **VK Deep Learning community** (http://vk.com/deeplearning)
- **Quora** (https://www.quora.com/topic/Deep-Learning)
- **FB Deep Learning Moscow** (https://www.facebook.com/groups/1505369016451458/)
- **Twitter Deep Learning Hub** (https://twitter.com/DeepLearningHub)
- **NVidia blog** (https://devblogs.nvidia.com/parallelforall/tag/deep-learning/)
- **IEEE Spectrum blog** (http://spectrum.ieee.org/blog/cars-that-think)
- http://deeplearning.net/
- ...

За кем следить?

- **Jürgen Schmidhuber** (http://people.idsia.ch/~juergen/)
- **Geoffrey E. Hinton** (http://www.cs.toronto.edu/~hinton/)
- **Google DeepMind** (http://deepmind.com/)
- **Yann LeCun** (http://yann.lecun.com, https://www.facebook.com/yann.lecun)
- **Yoshua Bengio** (http://www.iro.umontreal.ca/~bengioy, https://www.quora.com/profile/Yoshua-Bengio)
- **Andrej Karpathy** (http://karpathy.github.io/)
- **Andrew Ng** (http://www.andrewng.org/)
- ...

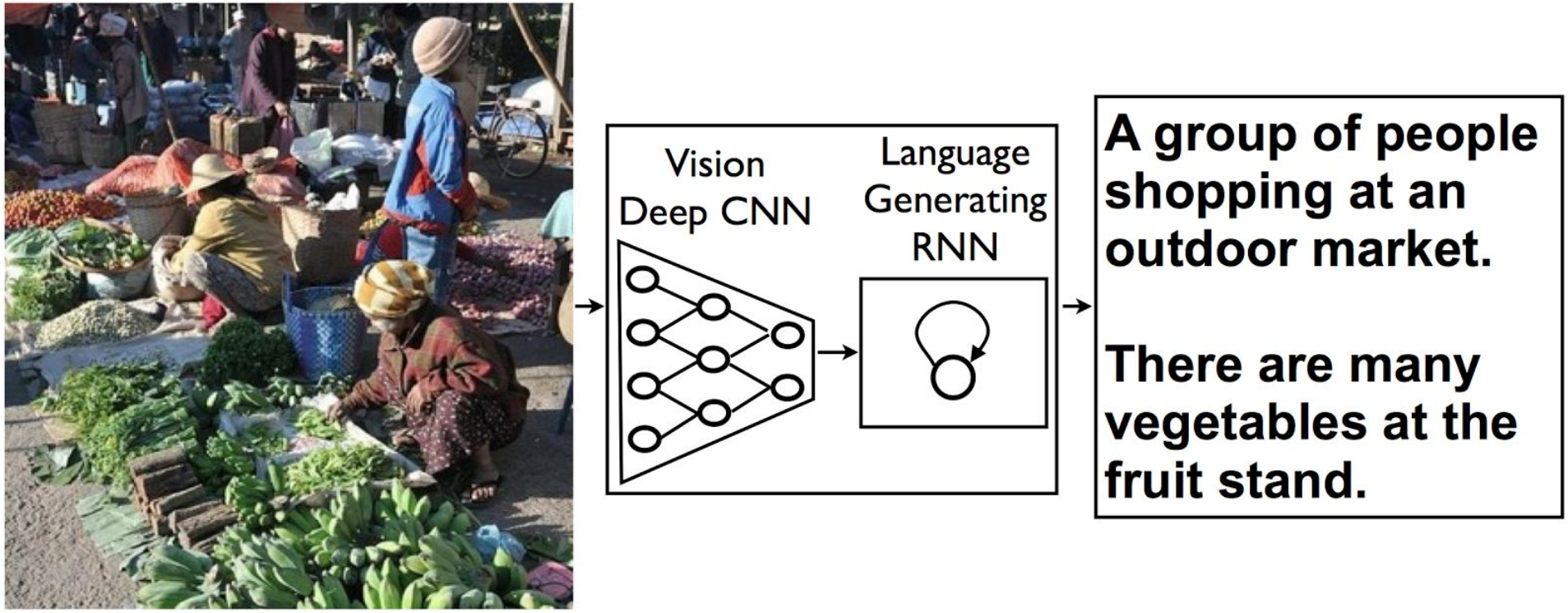
Bonus: More non-trivial examples

Image Colorization



http://richzhang.github.io/colorization/

Генерация описаний картинок



http://arxiv.org/abs/1411.4555 “Show and Tell: A Neural Image Caption Generator”

Human: "Elephants of mixed ages standing in a muddy landscape."
Model: "A herd of elephants walking across a dry grass field."

Human: "A person riding a dirt bike is covered in mud."
Computer model: "A person riding a motorcycle on a dirt road."

Human: "A group of men playing Frisbee in the park."
Computer model: "A group of young people playing a game of Frisbee."

Describes without errors	Describes with minor errors	Somewhat related to the image	Unrelated to the image
 A person riding a motorcycle on a dirt road.	 Two dogs play in the grass.	 A skateboarder does a trick on a ramp.	 A dog is jumping to catch a frisbee.
 A group of young people playing a game of frisbee.	 Two hockey players are fighting over the puck.	 A little girl in a pink hat is blowing bubbles.	 A refrigerator filled with lots of food and drinks.
 A herd of elephants walking across a dry grass field.	 A close up of a cat laying on a couch.	 A red motorcycle parked on the side of the road.	 A yellow school bus parked in a parking lot.

Easy Hacking: NeuralTalk and Walk

Ингредиенты:

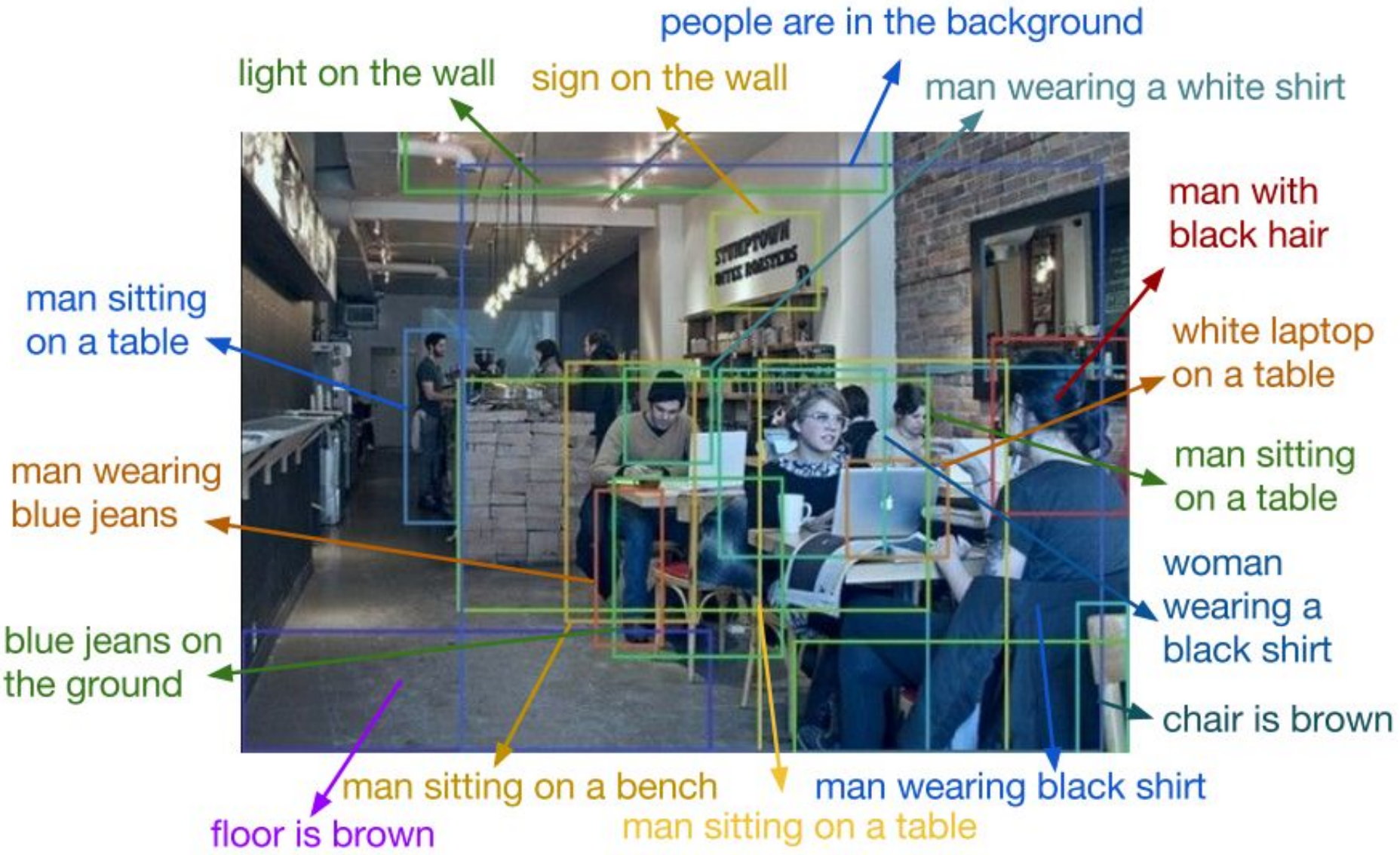
- https://github.com/karpathy/neuraltalk2
  Project for learning Multimodal Recurrent Neural Networks that describe images with sentences
- Веб-камера/ноутбук

Результат:

- https://vimeo.com/146492001


a group of people riding bikes down a street
01:56
CC
HD

More Hacking: DenseCap and ?



DenseCap: Fully Convolutional Localization Networks for Dense Captioning

http://arxiv.org/abs/1511.07571

Reinforcement Learning

Управление симулированным автомобилем на основе видеосигнала (2013)
http://people.idsia.ch/~juergen/gecco2013torcs.pdf
http://people.idsia.ch/~juergen/compressednetworksearch.html

(a) (b) (c)

Figure 4: Visual TORCS environment. (a) The 1st-person perspective used as input to the RNN controllers (figure 5) to drive the car around the track. (b), a 3rd-person perspective of car. The controllers were evolved using a track (c) of length of 714.16m and road width of 10m, that consists of straight segments of length 50 and 100m and curves with radius of 25m. The car starts at the bottom (start line) and has to drive counter-clockwise. The track boundary has a width of 14m.

Reinforcement Learning

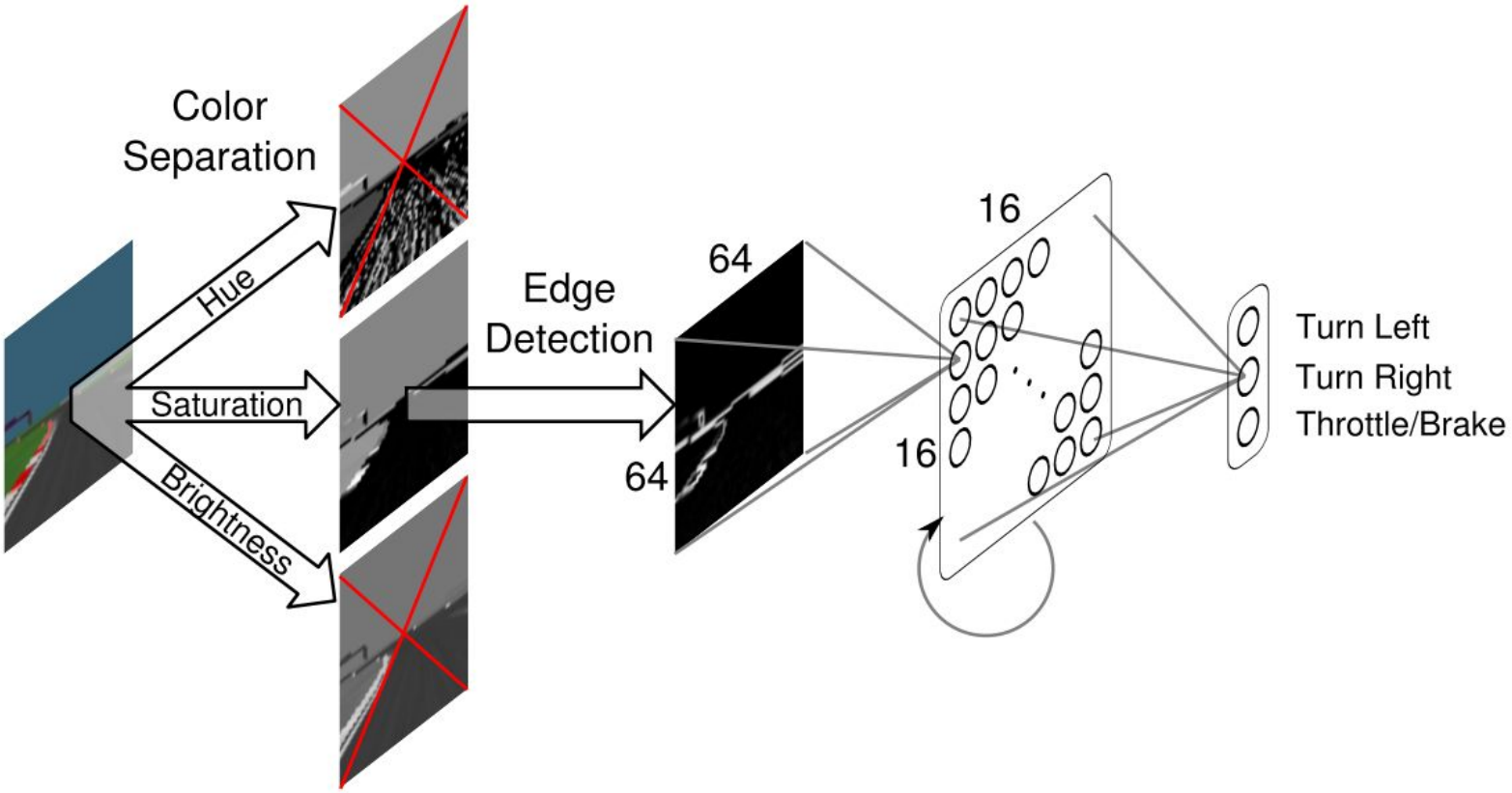


Figure 5: Visual TORCS network controller pipeline. At each time-step a raw 64×64 pixel image, taken from the driver's perspective, is split into three planes (hue, saturation and brightness). The saturation plane is then passed through Robert's edge detector [12] and then fed into the 16×16=256 recurrent neurons of the controller network, which then outputs the three driving commands.

Reinforcement Learning

Human-level control through deep reinforcement learning (2014)
http://www.nature.com/nature/journal/v518/n7540/full/nature14236.html

Playing Atari with Deep Reinforcement Learning (2013)
http://arxiv.org/abs/1312.5602

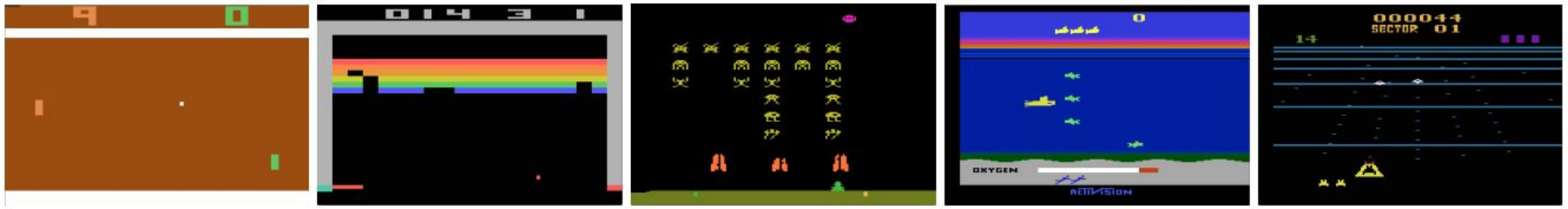
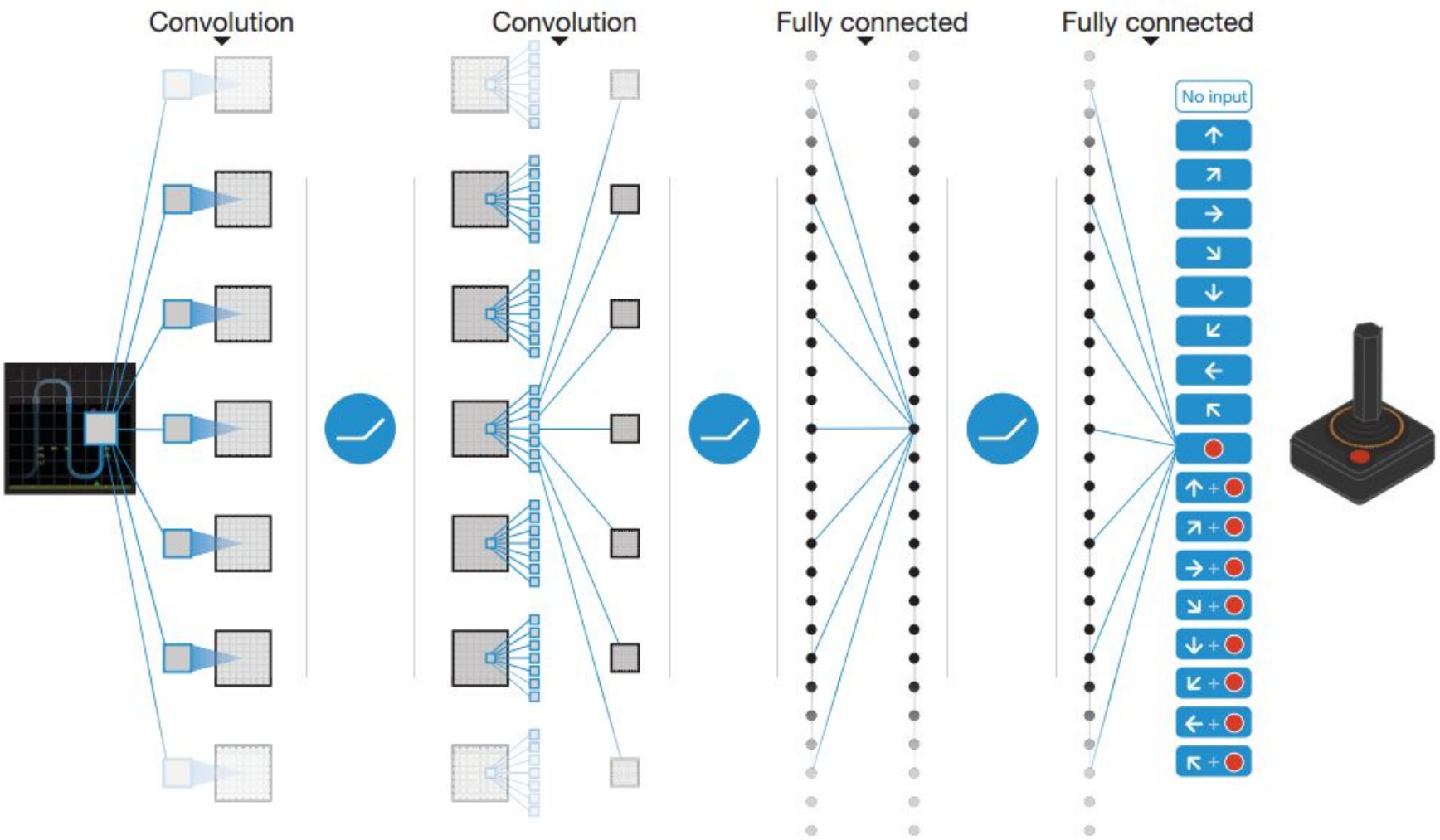
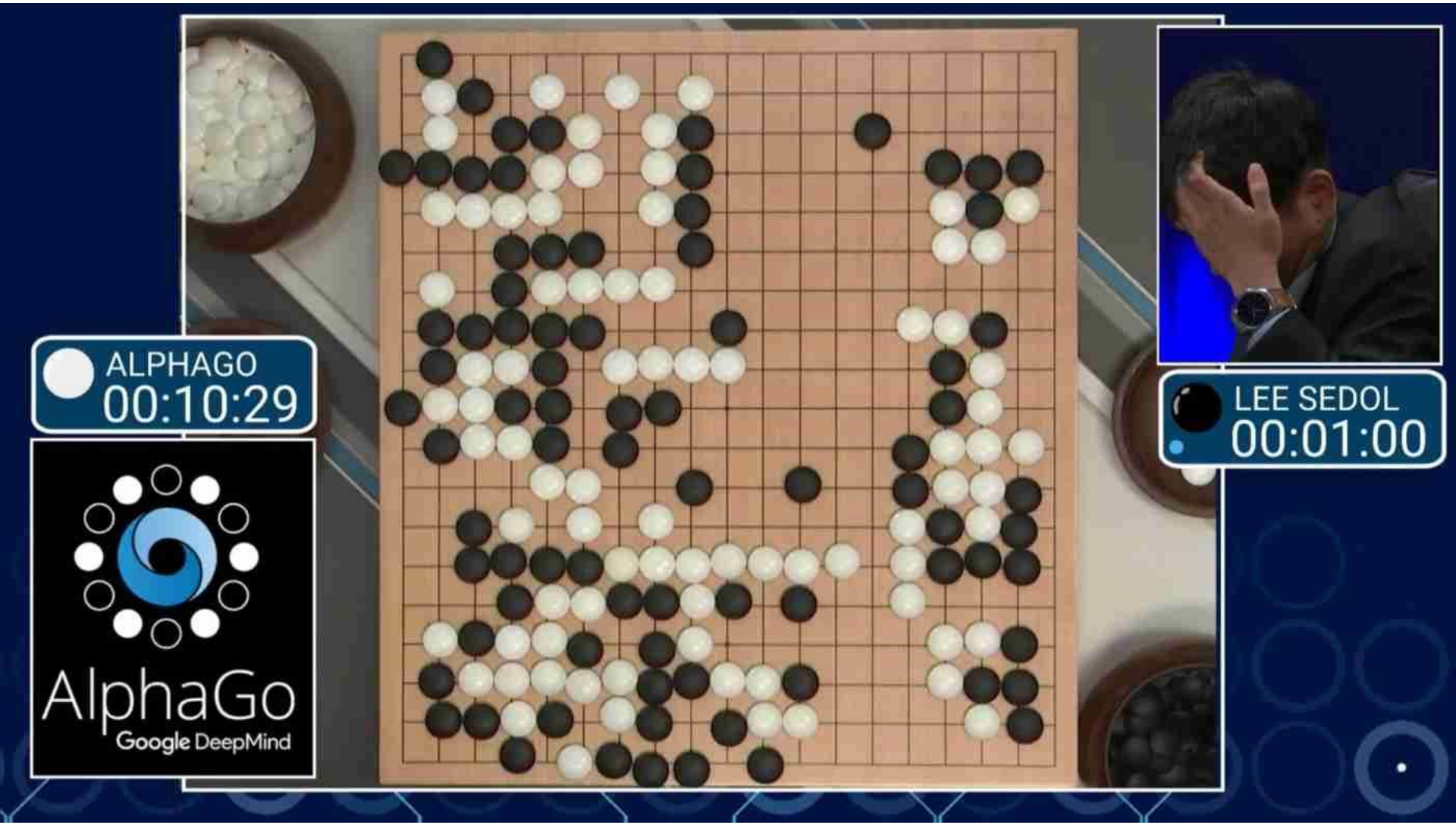


Figure 1: Screen shots from five Atari 2600 Games: (*Left-to-right*) Pong, Breakout, Space Invaders, Seaquest, Beam Rider

Reinforcement Learning



Game of Go: Computer-Human 4:1



Car driving

“Actually a “Perception to Action” system. The visual perception and control system is a Deep learning architecture trained end to end to transform pixels from the cameras into steering angles. And this car uses regular color cameras, not LIDARS like the Google cars. It is watching the driver and learns.”

https://www.youtube.com/watch?v=YuyT2SDcYrU

Example: Sensorimotor Deep Learning

“In this project we aim to develop deep learning techniques that can be deployed on a robot to allow it to learn directly from trial-and-error, where the only information provided by the teacher is the degree to which it is succeeding at the current task.”
http://rll.berkeley.edu/deeplearningrobotics/

Case: Machine Translation

Method	test BLEU score (ntst14)
Bahdanau et al. [2]	28.45
Baseline System [29]	33.30
Single forward LSTM, beam size 12	26.17
Single reversed LSTM, beam size 12	30.59
Ensemble of 5 reversed LSTMs, beam size 1	33.00
Ensemble of 2 reversed LSTMs, beam size 12	33.27
Ensemble of 5 reversed LSTMs, beam size 2	34.50
Ensemble of 5 reversed LSTMs, beam size 12	**34.81**

Table 1: The performance of the LSTM on WMT'14 English to French test set (ntst14). Note that an ensemble of 5 LSTMs with a beam of size 2 is cheaper than of a single LSTM with a beam of size 12.

Method	test BLEU score (ntst14)
Baseline System [29]	33.30
Cho et al. [5]	34.54
State of the art [9]	**37.0**
Rescoring the baseline 1000-best with a single forward LSTM	35.61
Rescoring the baseline 1000-best with a single reversed LSTM	35.85
Rescoring the baseline 1000-best with an ensemble of 5 reversed LSTMs	**36.5**
Oracle Rescoring of the Baseline 1000-best lists	~45

Table 2: Methods that use neural networks together with an SMT system on the WMT'14 English to French test set (ntst14).

Sequence to Sequence Learning with Neural Networks, http://arxiv.org/abs/1409.3215

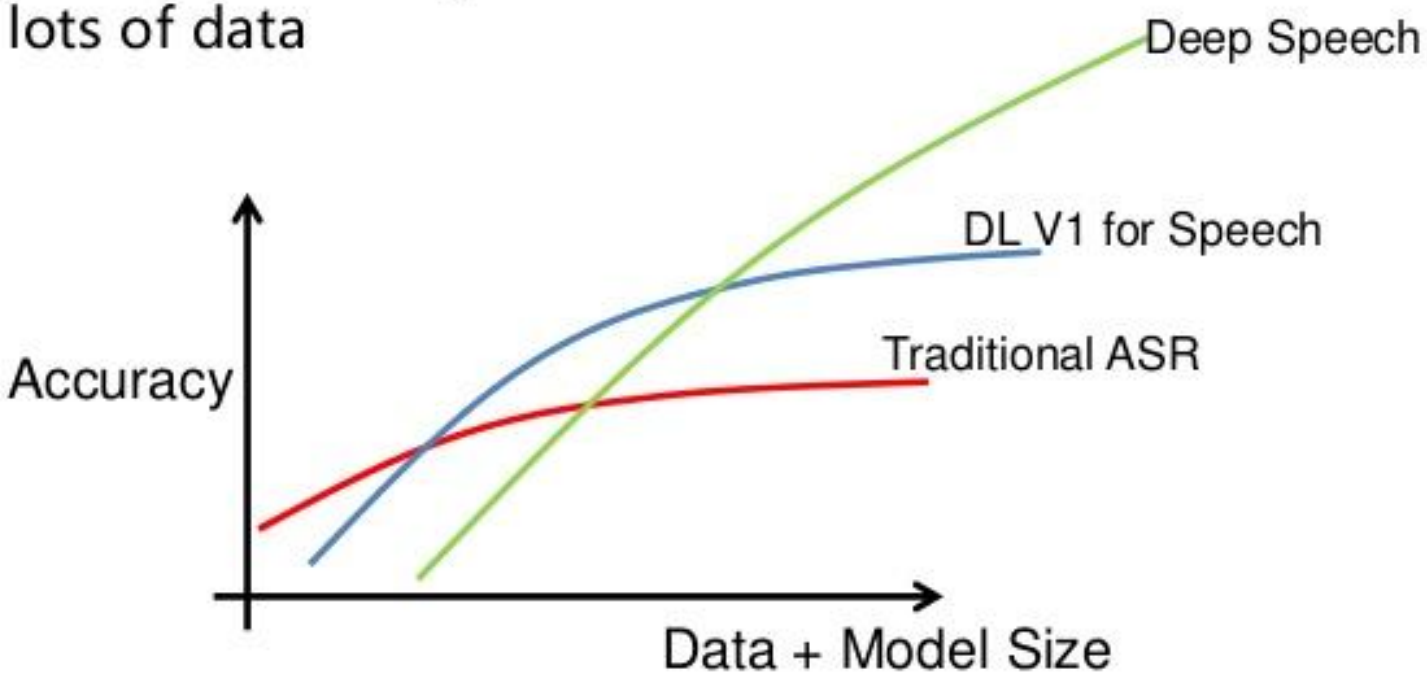
Case: Automated Speech Translation

Translating voice calls and video calls in 7 languages and instant messages in over 50.
https://www.skype.com/en/features/skype-translator/

Case: Baidu Automated Speech Recognition (ASR)

Speech Recognition 3: "Deep Speech"

- We believe end-to-end DL works better when we have big models and lots of data

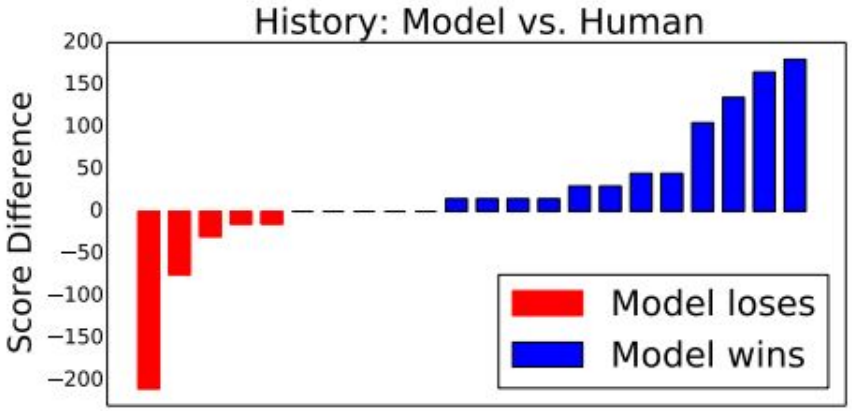


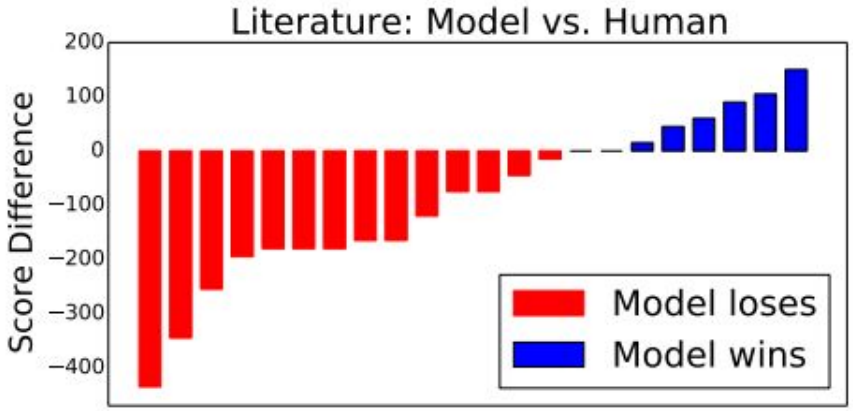
More Fun: MtG cards

http://www.escapistmagazine.com/articles/view/scienceandtech/14276-Magic-The-Gathering-Cards-Made-by-Artificial-Intelligence

Case: Question Answering

QUESTION:
He left unfinished a novel whose title character forges his father's signature to get out of school and avoids the draft by feigning desire to join. A more famous work by this author tells of the rise and fall of the composer Adrian Leverkühn. Another of his novels features the jesuit Naptha and his opponent Settembrini, while his most famous work depicts the aging writer Gustav von Aschenbach. Name this German author of The Magic Mountain and Death in Venice.
ANSWER: Thomas Mann





A Neural Network for Factoid Question Answering over Paragraphs,
https://cs.umd.edu/~miyyer/qblearn/

Case: Dialogue Systems

A Neural Conversational Model,
Oriol Vinyals, Quoc Le
http://arxiv.org/abs/1506.05869

Human: *hello !*
Machine: *hello !*
Human: *how are you ?*
Machine: *i 'm good .*
Human: *what 's your name ?*
Machine: *i 'm julia .*
Human: *when were you born ?*
Machine: *july 20th .*
Human: *what year were you born ?*
Machine: *1977 .*
Human: *where are you from ?*

Machine: *hi*
Human: *hello*
Machine: *could you please let me know what are the operating systems you are using ?*
Human: *i am using linux*
Machine: *ok it was in the ssl vpn on which you stated it was broken*
Machine: *please check if you are logged in to corp network*
Machine: *what is the error that you are running please*
Human: *i am seeing an error related to vpn*
Machine: *what is the error message that you are getting when connecting to vpn using network connect ?*
Human: *connection refused or something like that*
Machine: *may i know the version of network connect you connect ?*
Human: *i am not sure i know that*
Machine: *is the network connect prompting for any user certificate*
Machine: *?*
Human: *yes*
Machine: *are you at home or at work at the moment ?*
Machine: *office ?*
Human: *i am at home*
Machine: *try this*
Machine: *goto <URL> page and enter your name and password*
Human: *done*
Machine: *try logging in with <NAME> and check if its working fine now*
Human: *yes , now it works !*

Visual Question Answering

What vegetable is on the plate?
Neural Net: broccoli
Ground Truth: broccoli

What color are the shoes on the person's feet ?
Neural Net: brown
Ground Truth: brown

How many school busses are there?
Neural Net: 2
Ground Truth: 2

What sport is this?
Neural Net: baseball
Ground Truth: baseball

What is on top of the refrigerator?
Neural Net: magnets
Ground Truth: cereal

What uniform is she wearing?
Neural Net: shorts
Ground Truth: girl scout

What is the table number?
Neural Net: 4
Ground Truth:40

What are people sitting under in the back?
Neural Net: bench
Ground Truth: tent

https://avisingh599.github.io/deeplearning/visual-qa/

Visual Question Answering

COCOQA 33827
What is the color of the cat?
Ground truth: black
IMG+BOW: black (0.55)
2-VIS+LSTM: black (0.73)
BOW: gray (0.40)

COCOQA 33827a
What is the color of the couch?
Ground truth: red
IMG+BOW: red (0.65)
2-VIS+LSTM: black (0.44)
BOW: red (0.39)

DAQUAR 1522
How many chairs are there?
Ground truth: two
IMG+BOW: four (0.24)
2-VIS+BLSTM: one (0.29)
LSTM: four (0.19)

DAQUAR 1520
How many shelves are there?
Ground truth: three
IMG+BOW: three (0.25)
2-VIS+BLSTM: two (0.48)
LSTM: two (0.21)

COCOQA 14855
Where are the ripe bananas sitting?
Ground truth: basket
IMG+BOW: basket (0.97)
2-VIS+BLSTM: basket (0.58)
BOW: bowl (0.48)

COCOQA 14855a
What are in the basket?
Ground truth: bananas
IMG+BOW: bananas (0.98)
2-VIS+BLSTM: bananas (0.68)
BOW: bananas (0.14)

DAQUAR 585
What is the object on the chair?
Ground truth: pillow
IMG+BOW: clothes (0.37)
2-VIS+BLSTM: pillow (0.65)
LSTM: clothes (0.40)

DAQUAR 585a
Where is the pillow found?
Ground truth: chair
IMG+BOW: bed (0.13)
2-VIS+BLSTM: chair (0.17)
LSTM: cabinet (0.79)

Exploring Models and Data for Image Question Answering
http://arxiv.org/abs/1505.02074

Visual Question Answering

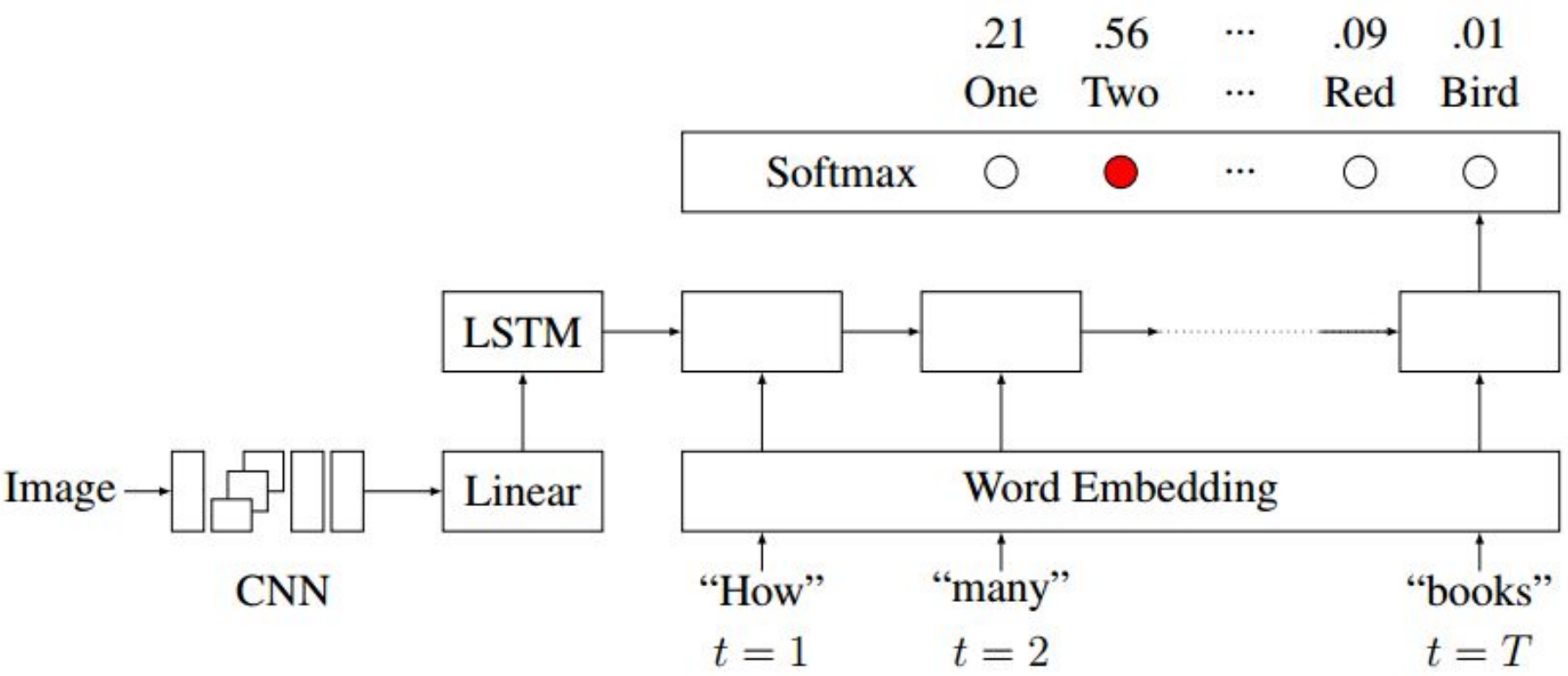


Figure 2: VIS+LSTM Model

Exploring Models and Data for Image Question Answering
http://arxiv.org/abs/1505.02074