
Deep learning


«Глубокое обучение»


Искусственные
нейронные сети

Индуктивное моделирование систем с помощью численной оптимизации структурированных функций

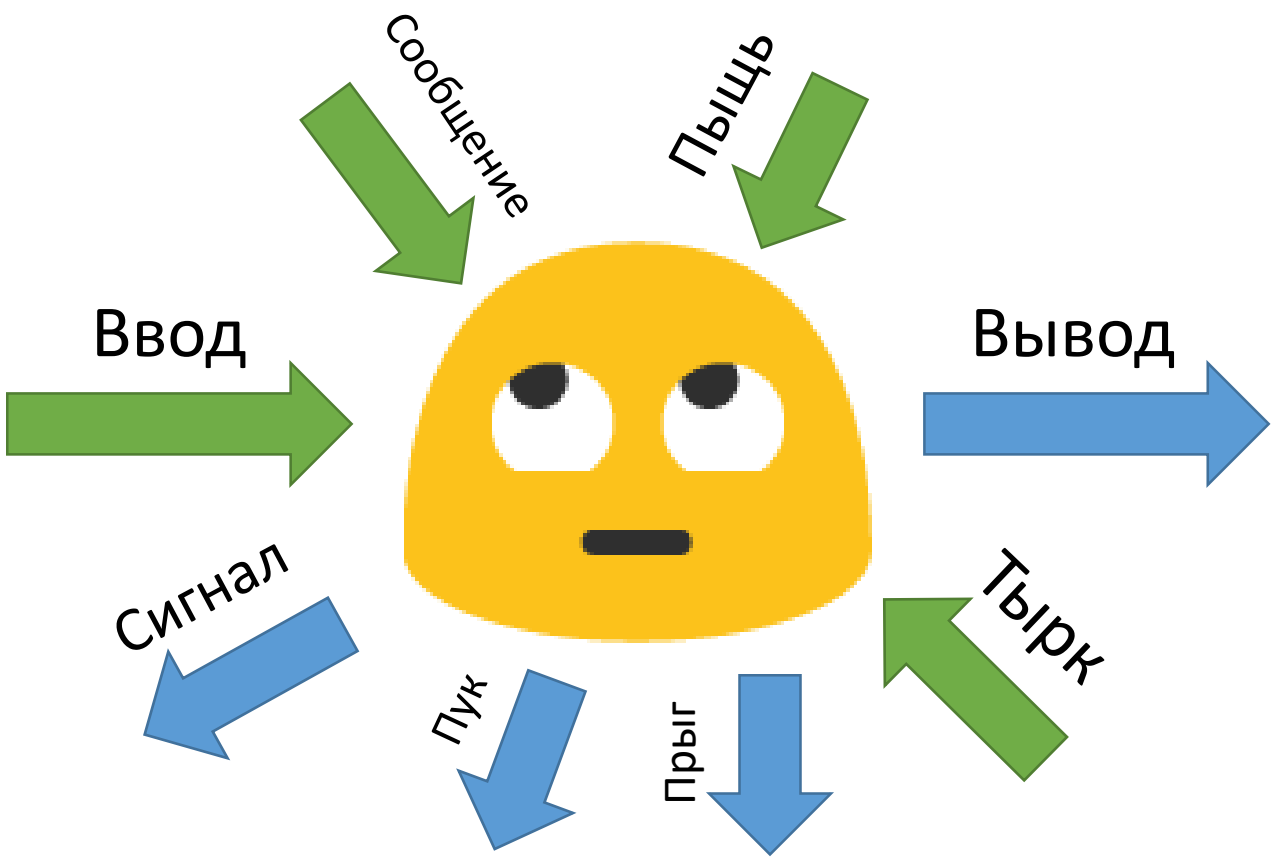
Индуктивное моделирование систем с помощью численной оптимизации структурированных функций

Индуктивное моделирование систем с помощью численной оптимизации структурированных функций

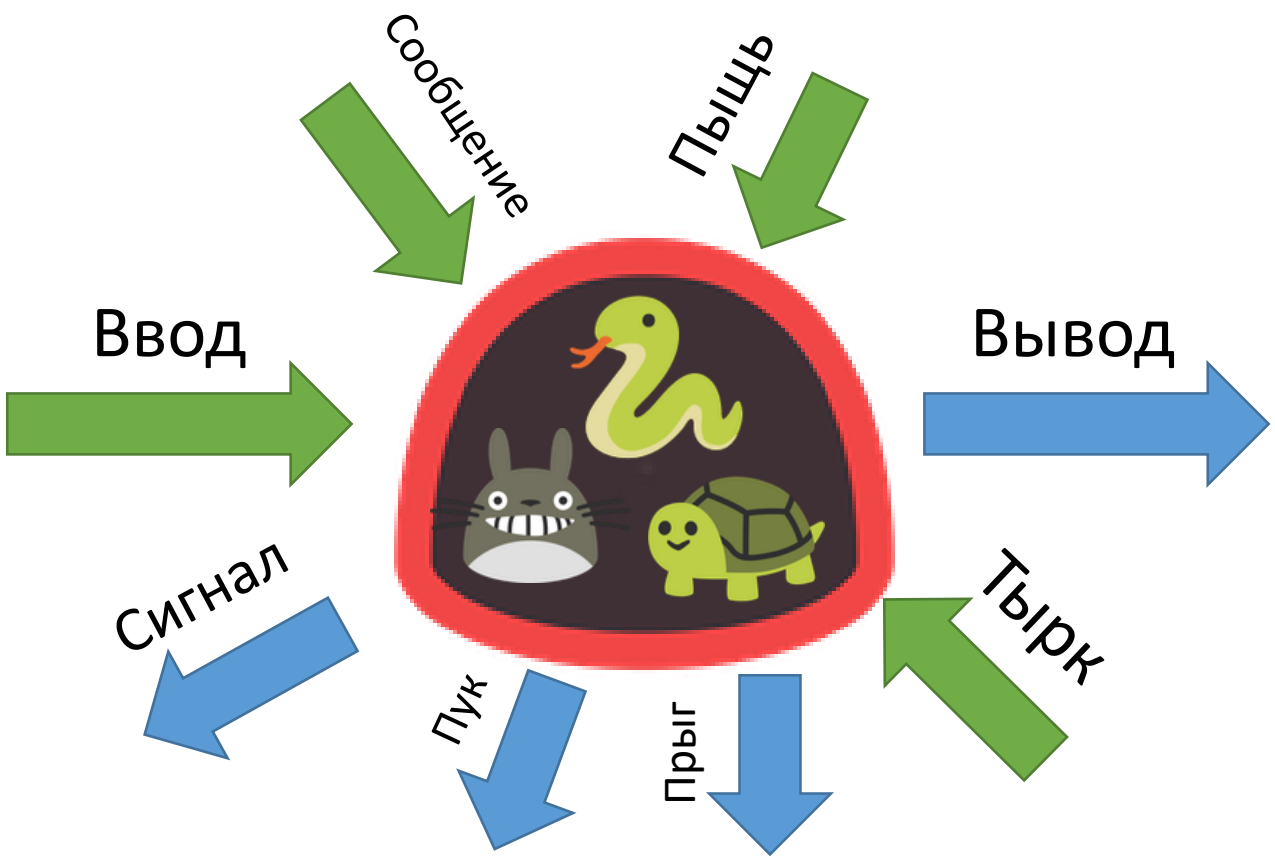
Система

Система

Система

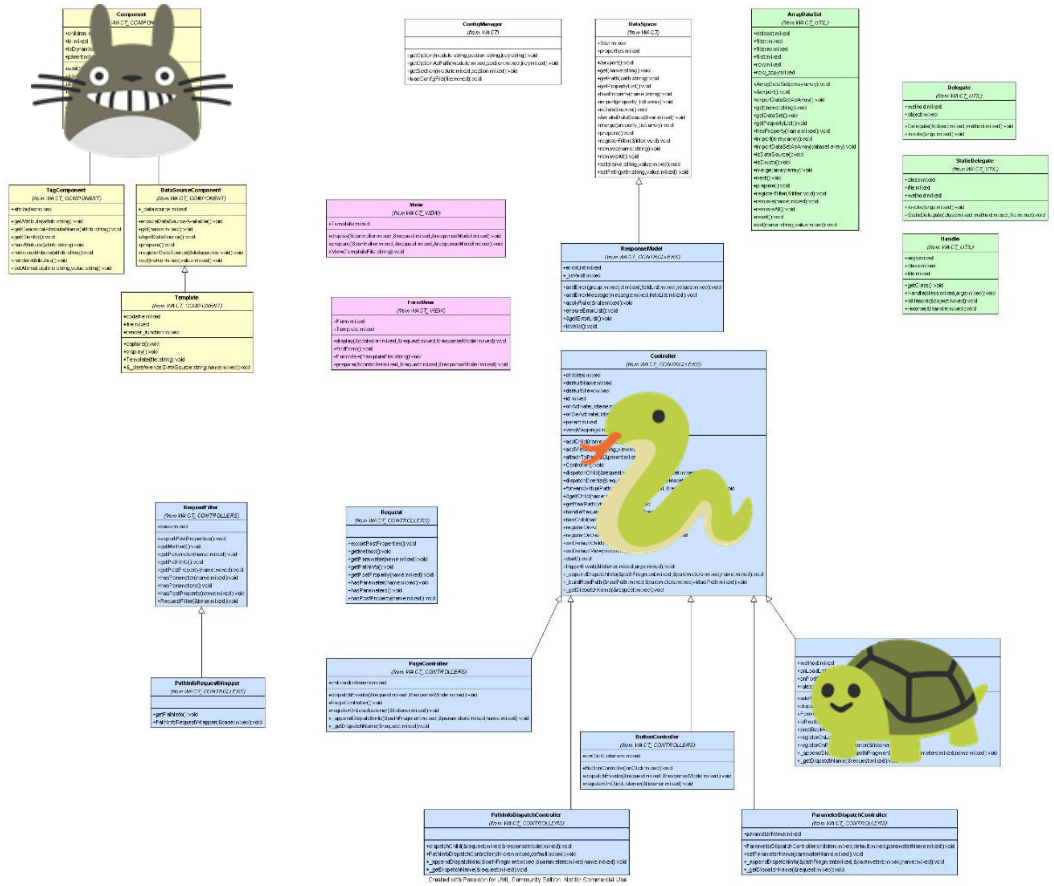


Система



Индуктивное моделирование систем с помощью численной оптимизации структурированных функций

Моделирование систем



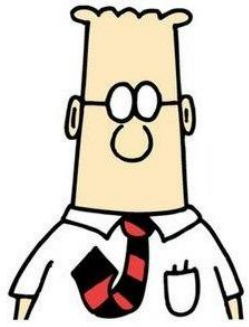
Функция

```
def f(inputs):
    ...
    interact_with_other_systems()
    ...
    return outputs
```

Моделирование систем

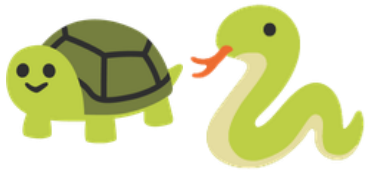
Ушастых не пускать!

```
def f(input):

    if Ушастый(input):
        return НЕ_ПУСКАТЬ
    else:
        return ПУСКАТЬ
```

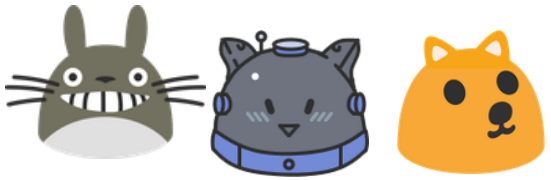


Индуктивная спецификация

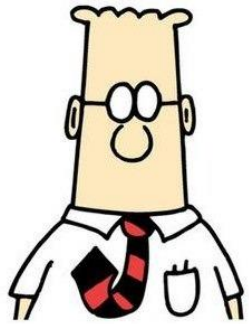
Таких пускать:



Таких - нет:

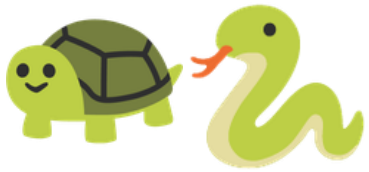


```
def f(input):

    if Ушастый(input):
        return НЕ_ПУСКАТЬ
    else:
        return ПУСКАТЬ
```

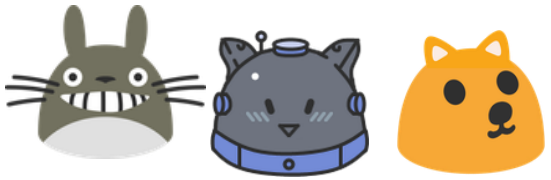


Индуктивная спецификация

Таких пускать:



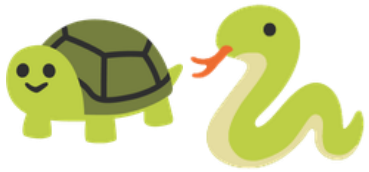
Таких - нет:



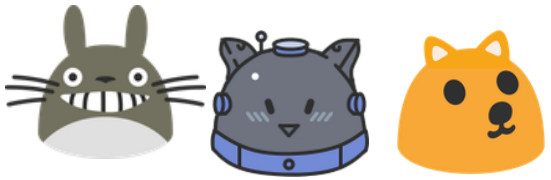
```
def f(input):

    if Зелёный(input):
        return ПУСКАТЬ
    else:
        return НЕ_ПУСКАТЬ
```

Индуктивная спецификация

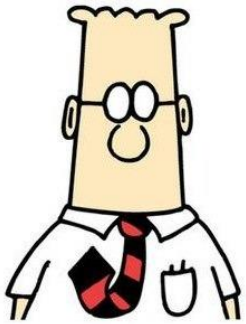
Таких пускать:



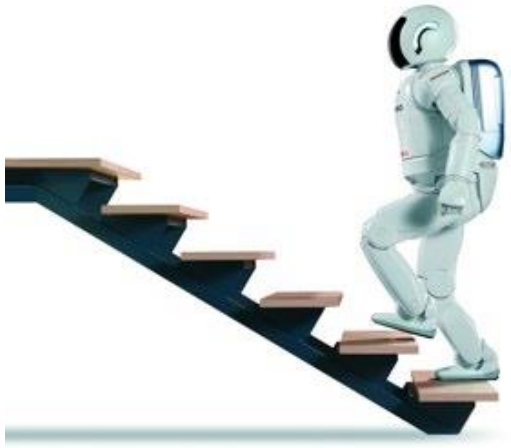
Таких - нет:



```
def f(input):

    ???
```



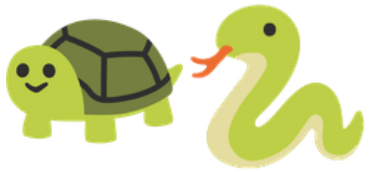
Индуктивные задачи



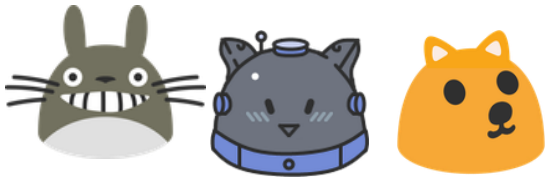
Индуктивное моделирование систем с помощью численной оптимизации структурированных функций

Индуктивное моделирование

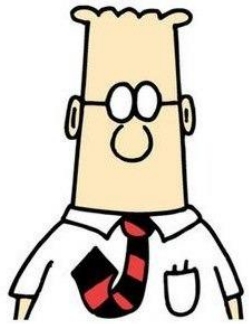
Таких пускать:



Таких - нет:



```
def f(input):

    ???
```



Мера ошибки

```
def Ошибка(f, [данные]):
    return not f(🐢) +
           not f(🐍) +
           f(🐱) +
           f(🐈) +
           f(🐻)
```

Машинное обучение

```
for f in Всякие_Функции:
    if Ошибка(f, данные) == 0:
        return f
```

Машинное обучение

$$\text{argmin}_{f \in F}\ Error(f, Data)$$

Индуктивное моделирование систем с помощью численной оптимизации структурированных функций

Линейные функции

$$f_{\boldsymbol{w},b}(x_1, x_2) = w_1 x_1 + w_2 x_2 + b$$

```
def linear_function(x, w, b):
    return x[0]*w[0] + x[1]*w[1] + b
```

Линейные функции

$$f_{\boldsymbol{w},b}(\boldsymbol{x}) := \boldsymbol{w}^T\boldsymbol{x} + b$$

```
def linear_function(x, w, b):
    return np.dot(w, x) + b
```

Машинное обучение

$$\text{argmin}_{f \in F}\, Error(f, Data)$$

Машинное обучение

$$\text{argmin}_{\boldsymbol{w},b \in R^n} \; Error(f_{\boldsymbol{w},b}, Data)$$

Машинное обучение

$$\nabla_{\boldsymbol{w},b}\, Error(f_{\boldsymbol{w},b}, Data)$$

Индуктивное моделирование систем с помощью численной оптимизации структурированных функций

Пример

```
x = [[ 2, 0.1, 3],
     [ 1, 0.1, 2],
     [ 0, 2.0, 1],
     [ 0, 1.5, 2],
     [-1, 1.2, 3]]
```





```
y = [1,
     1,
     0,
     0,
     0]
```

Пример

```
x = [[ 2, 0.1, 3],
     [ 0, 1.5, 2],
     [-1, 1.2, 3]]

y = [1,
     0,
     0]
```

f(x[i]) = y[i]

Пример

```
x = [[ 2, 0.1, 3],
 [ 0, 1.5, 2],
 [-1, 1.2, 3]]
```

```
y = [1,
 0,
 0]
```

`dot(w, x[i]) + b = y[i]`

Пример

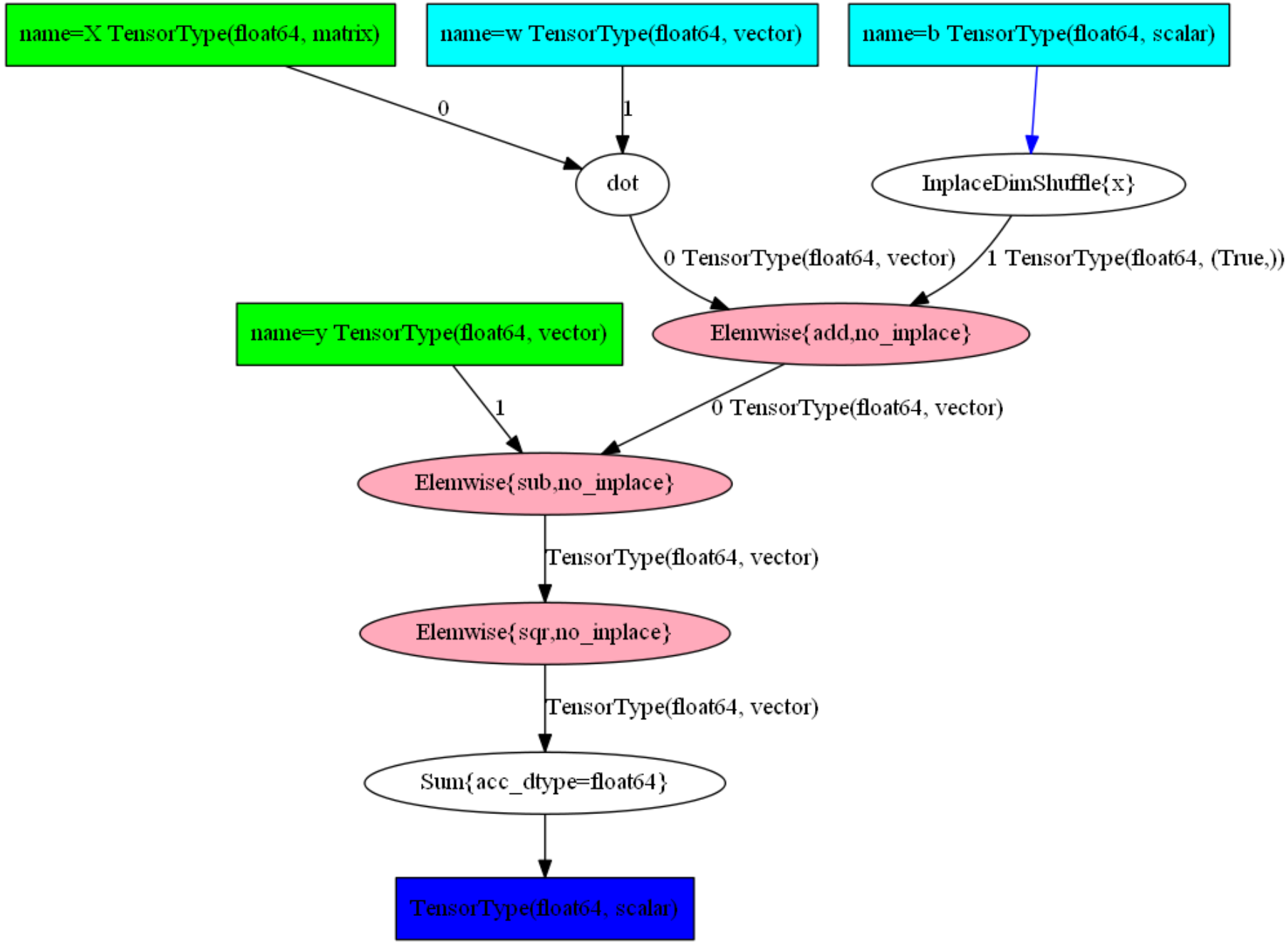
```
x = [[ 2, 0.1, 3],
 [ 0, 1.5, 2],
 [-1, 1.2, 3]]
```

```
y = [1,
 0,
 0]
```

dot(w, x[i]) + b ≅ y[i]

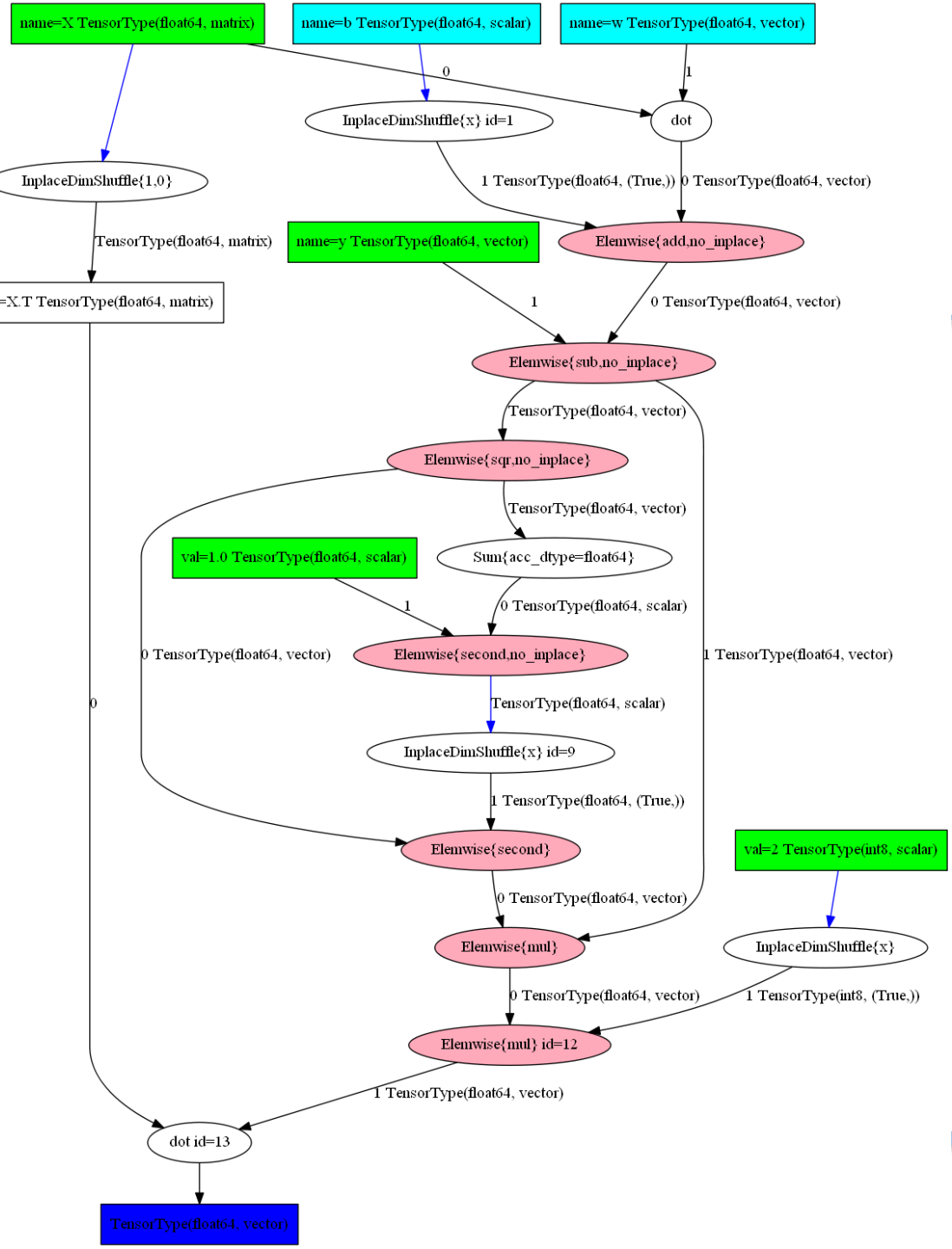
Theano

```
import theano
import theano.tensor as T
import numpy as np

X = T.matrix('X')
y = T.vector('y')

w = theano.shared(np.array([0.,0.,0.]), 'w')
b = theano.shared(0.0, 'b')

y_hat = T.dot(X, w) + b
error = T.sum(T.sqr(y_hat - y))
```


name=X TensorType(float64, matrix)
name=w TensorType(float64, vector)
name=b TensorType(float64, scalar)
0
1
dot
InplaceDimShuffle{x}
0 TensorType(float64, vector)
1 TensorType(float64, (True,))
name=y TensorType(float64, vector)
Elemwise{add,no_inplace}
1
0 TensorType(float64, vector)
Elemwise{sub,no_inplace}
TensorType(float64, vector)
Elemwise{sqr,no_inplace}
TensorType(float64, vector)
Sum{acc_dtype=float64}
TensorType(float64, scalar)

Дифференцирование

```
dw, db = T.grad(error, [w, b])
```


name=X TensorType(float64, matrix)
name=b TensorType(float64, scalar)
name=w TensorType(float64, vector)
0
1
InplaceDimShuffle{x} id=1
dot
InplaceDimShuffle{1,0}
1 TensorType(float64, (True,))
0 TensorType(float64, vector)
TensorType(float64, matrix)
name=y TensorType(float64, vector)
Elemwise{add,no_inplace}
name=X.T TensorType(float64, matrix)
1
0 TensorType(float64, vector)
Elemwise{sub,no_inplace}
TensorType(float64, vector)
Elemwise{sqr,no_inplace}
TensorType(float64, vector)
val=1.0 TensorType(float64, scalar)
Sum{acc_dtype=float64}
1
0 TensorType(float64, scalar)
0 TensorType(float64, vector)
Elemwise{second,no_inplace}
1 TensorType(float64, vector)
0
TensorType(float64, scalar)
InplaceDimShuffle{x} id=9
1 TensorType(float64, (True,))
Elemwise{second}
val=2 TensorType(int8, scalar)
0 TensorType(float64, vector)
Elemwise{mul}
InplaceDimShuffle{x}
0 TensorType(float64, vector)
1 TensorType(int8, (True,))
Elemwise{mul} id=12
1 TensorType(float64, vector)
dot id=13
TensorType(float64, vector)

Компиляция графа вычислений

```
train = theano.function(
            inputs=[X,y],
            outputs=[y_hat, error],
            updates=((w, w - 0.01 * dw),
                     (b, b - 0.01 * db)))
```

Тренировка модели

```
for i in range(1000):
    train(X_, y_)
train(X_, y_)
```

```
[array([ 1.01576331,  0.94681398, -0.0508905 ,  0.09478897, -0.02497436]),
 array(0.015275746079053545)]
```

```
w.get_value(), b.get_value()
```

```
(array([ 0.16761281, -0.48895232, -0.09882342]), array(1.025846241497001))
```

Tensorflow: Построение графа

```
import tensorflow as tf

X = tf.placeholder(tf.float32, shape=(None, 3))
y = tf.placeholder(tf.float32, shape=(None,))

w = tf.Variable(tf.zeros(3))
b = tf.Variable(tf.zeros(1))

y_hat = tf.matmul(X, tf.expand_dims(w,1)) + b
error = tf.reduce_sum(tf.square(y_hat - tf.expand_dims(y, 1)))
```

Tensorflow: Исполнение

```
optimizer = tf.train.GradientDescentOptimizer(0.005).minimize(error)

init = tf.global_variables_initializer()
sess = tf.Session()
sess.run(init)

for epoch in range(1000):
    _, c = sess.run([optimizer, error], feed_dict={X: X_, y: y_})
```

Keras

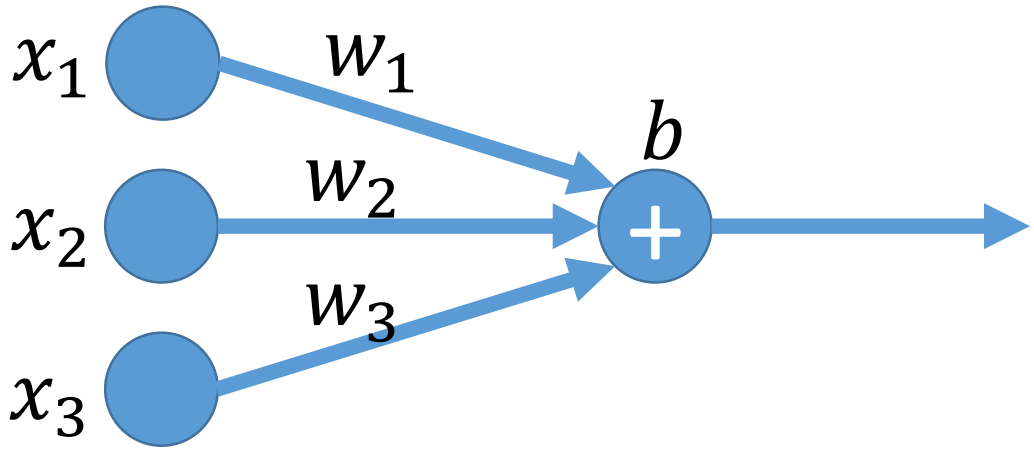
```
model = Sequential([Dense(1, input_shape=(3,))])

model.compile(loss='mean_squared_error', optimizer='sgd')
model.fit(X_, y_, epochs=1000, verbose=0)

model.predict(X_)
```

Линейная модель

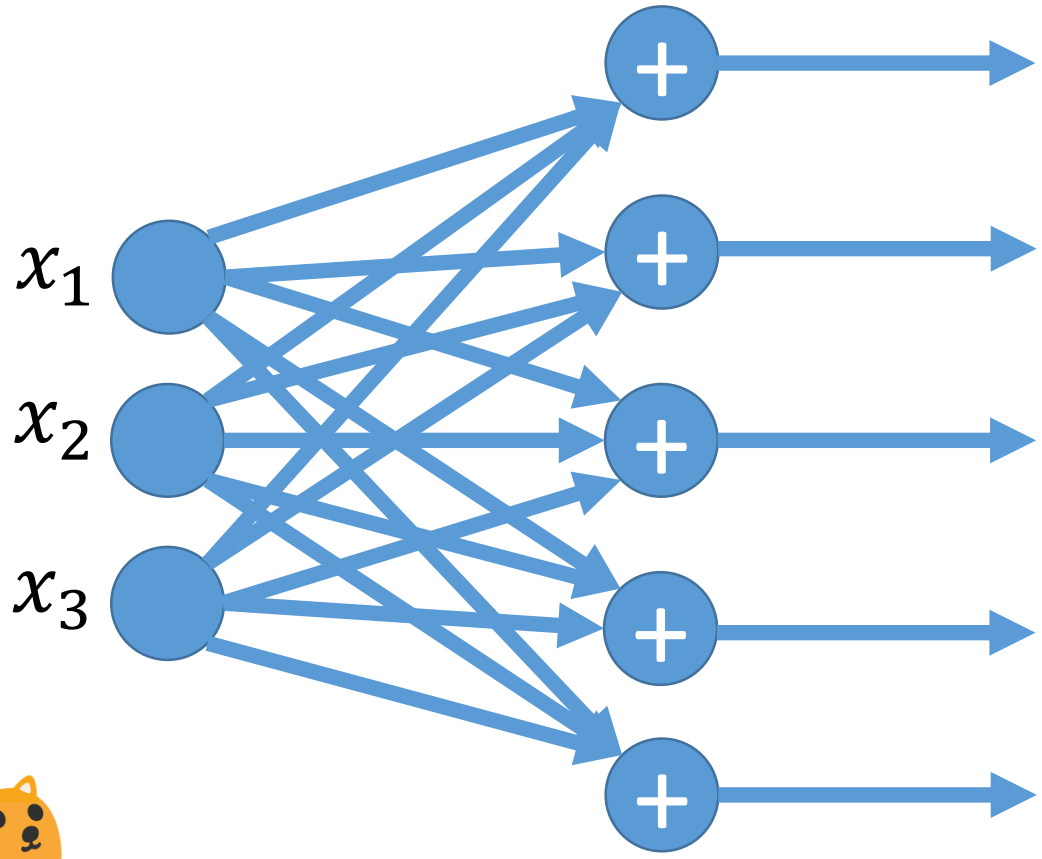
$$f_{w,b}(x_1, x_2, x_3) = w_1 x_1 + w_2 x_2 + w_3 x_3 + b$$

Линейная модель

$$f_{w,b}(x_1, x_2, x_3) = w_1 x_1 + w_2 x_2 + w_3 x_3 + b$$

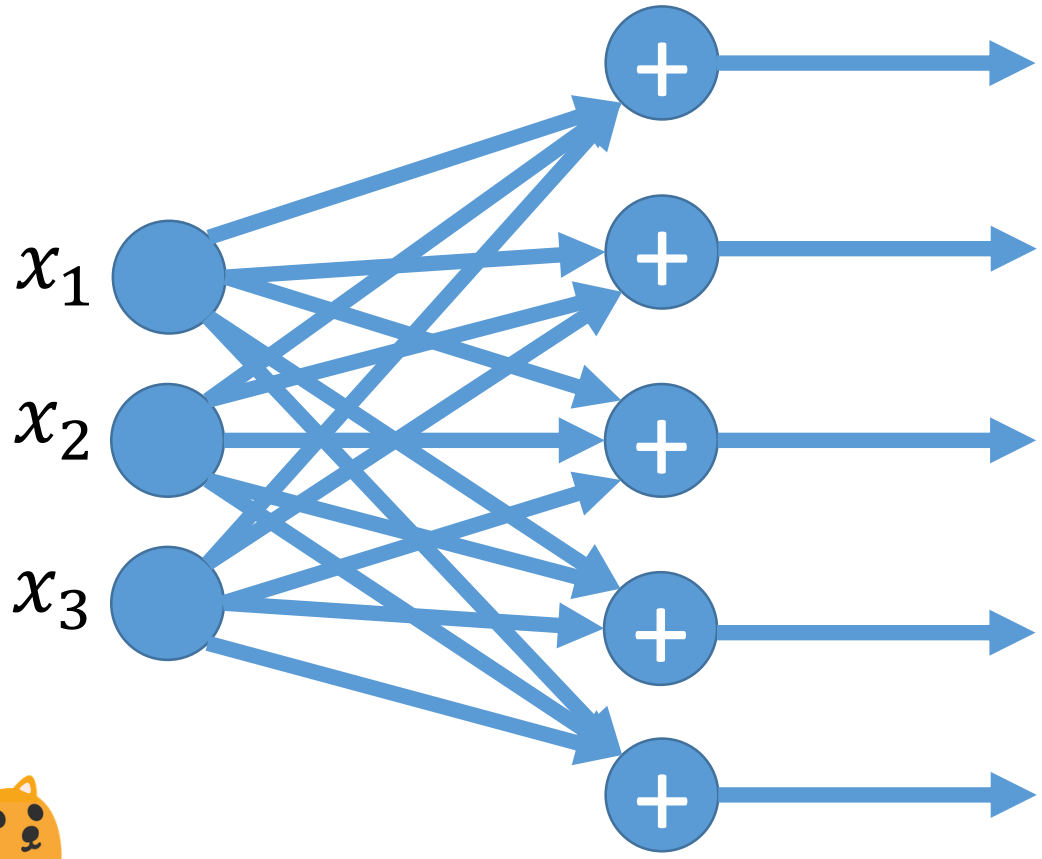


Полносвязный слой нейронов



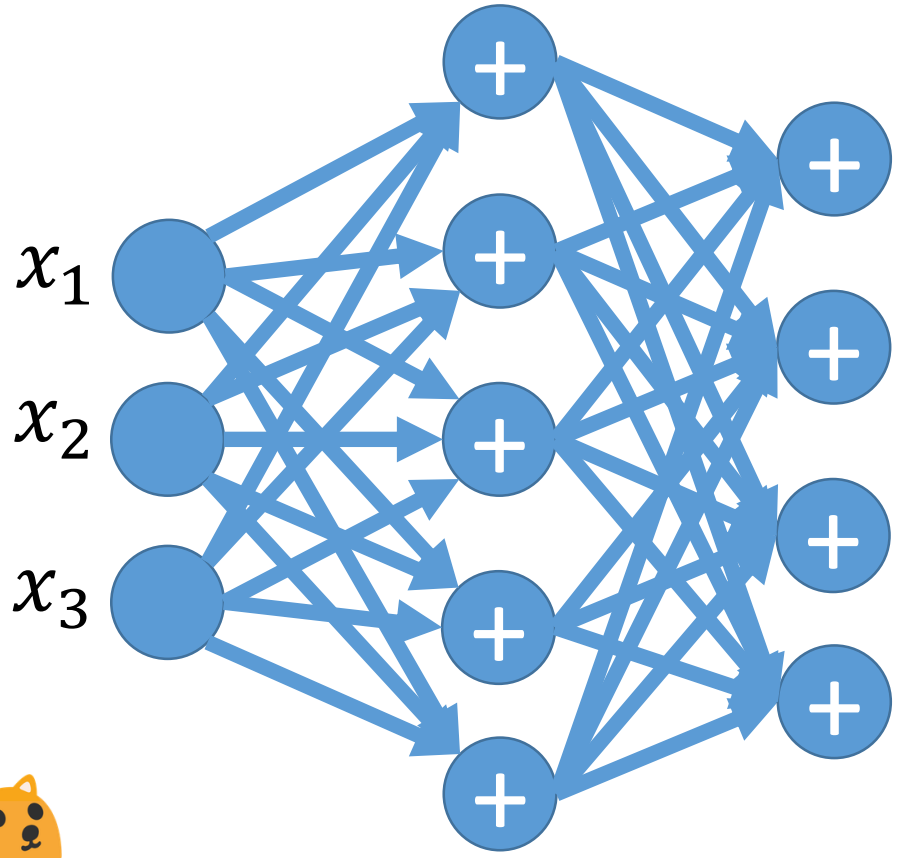

```
Sequential([Dense(5, input_shape=(3,))])
```

Полносвязный слой нейронов



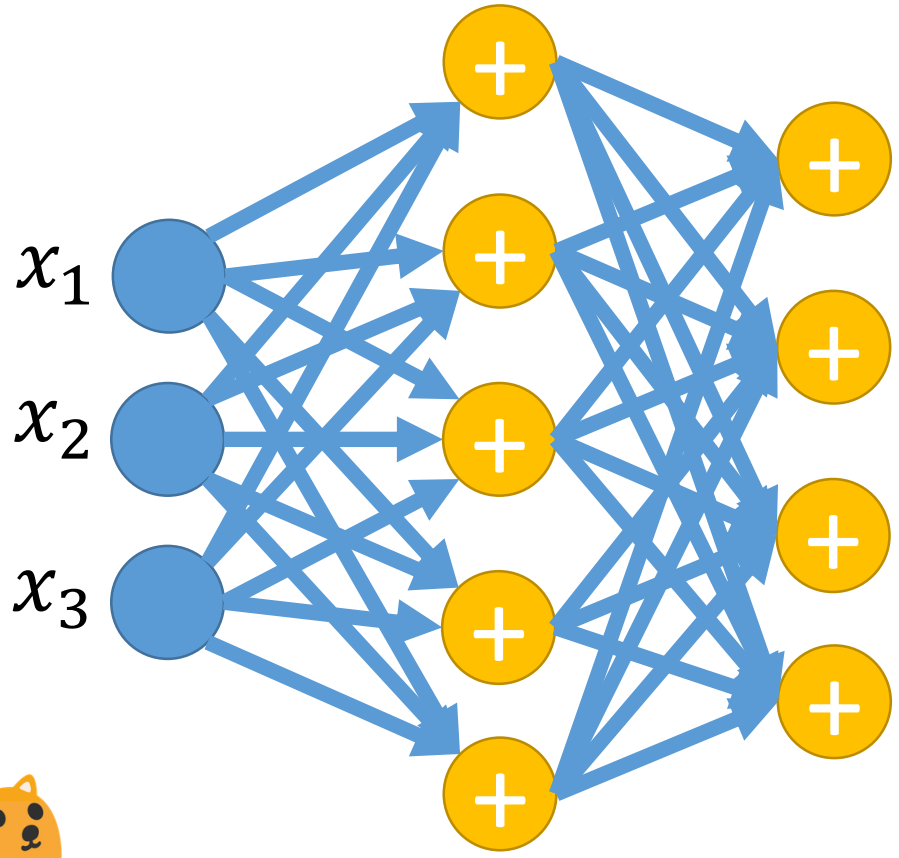

```
Sequential([InputLayer((3,)),
            Dense(5)])
```

Два полносвязных слоя



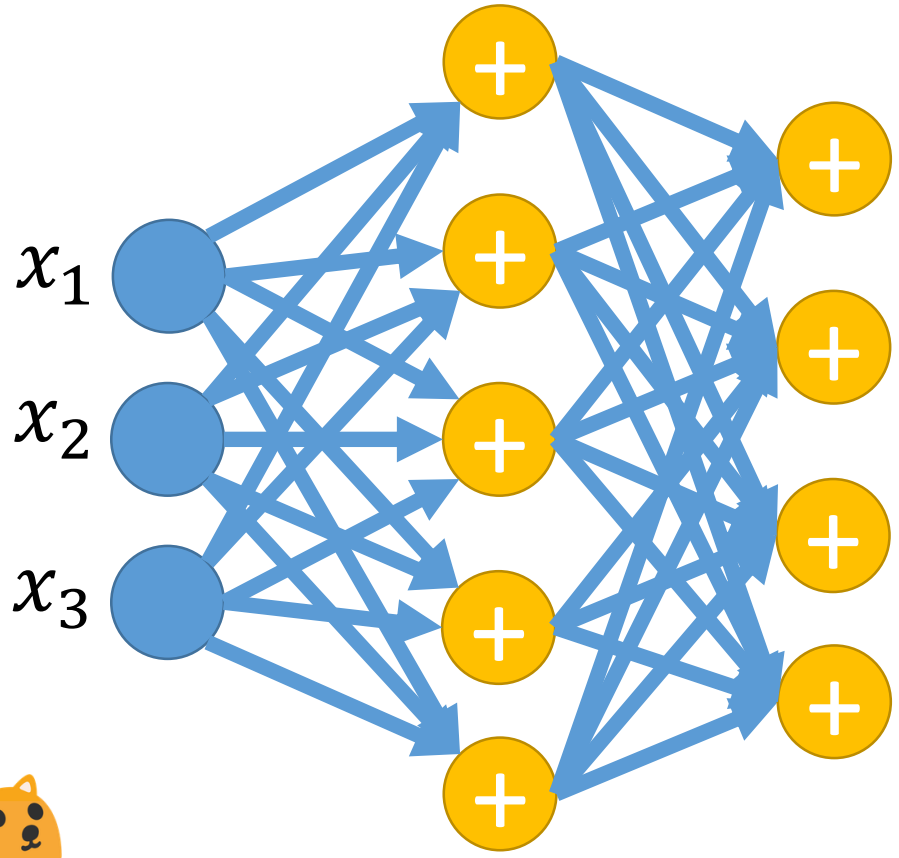

```
Sequential([InputLayer((3,)),
            Dense(5),
            Dense(4)])
```

Активации



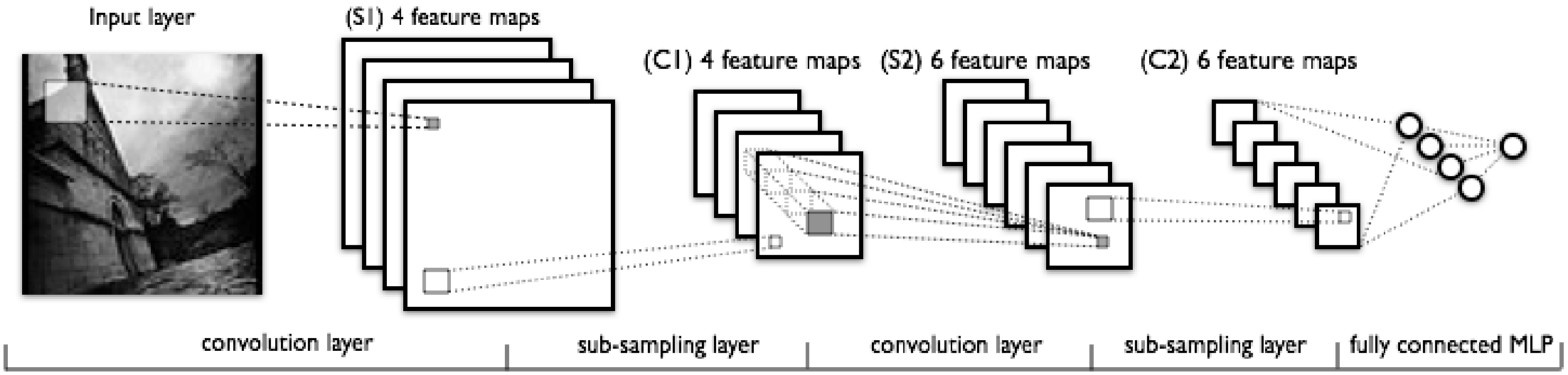

```
Sequential([InputLayer((3,)),
            Dense(5, activation='relu'),
            Dense(4, activation='sigmoid')])
```

Активации




```
Sequential([InputLayer((3,)),
            Dense(5),
            Activation('relu'),
            Dense(4),
            Activation('sigmoid')])
```

Конволюционные сети

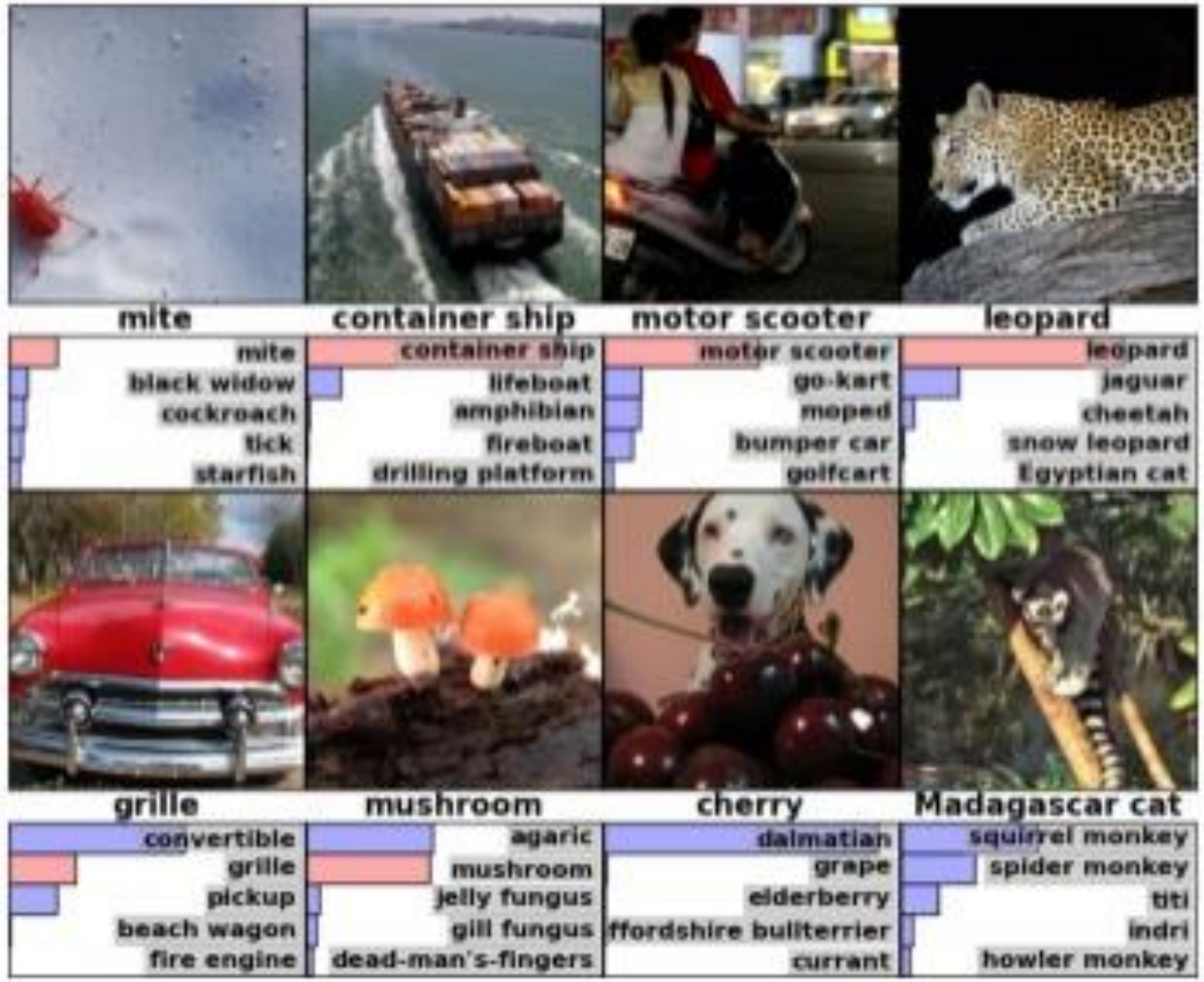



```
Sequential([Convolution2D(4, 5, 5, activation="relu",
                          input_shape=(1, w, h)),
            MaxPooling2D(pool_size=(2, 2), strides=(2, 2)),
            Convolution2D(6, 5, 5, activaton="relu"),
            MaxPooling2D(pool_size=(2, 2), strides=(2, 2)),
            Flatten(),
            Dense(4, activation="relu"),
            Dense(1, activation="softmax")])
```

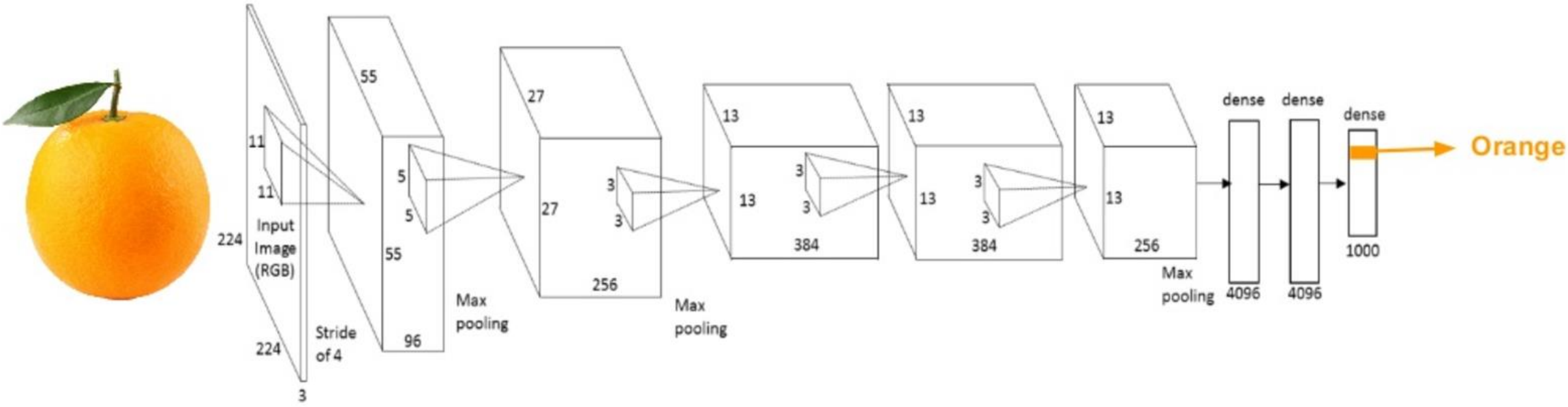
ImageNet Challenge

IMAGENET

- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.



AlexNet (2012)

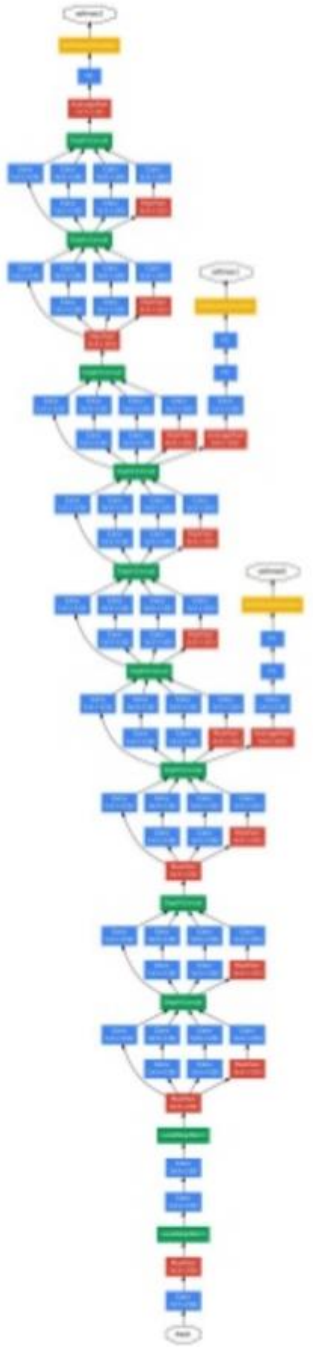


https://www.slideshare.net/xavigiro/saliency-prediction-using-deep-learning-techniques

GoogLeNet (2014)

AlexNet

- image
- conv-64
- conv-192
- conv-384
- conv-256
- conv-256
- FC-4096
- FC-4096
- FC-1000



- image
- conv-64
- conv-64
- maxpool
- conv-128
- conv-128
- maxpool
- conv-256
- conv-256
- conv-256
- conv-256
- maxpool
- conv-512
- conv-512
- conv-512
- conv-512
- maxpool
- conv-512
- conv-512
- conv-512
- conv-512
- maxpool
- FC-4096
- FC-4096
- FC-1000
- softmax

https://www.slideshare.net/xavigiro/deep-learning-for-computer-vision-imagenet-challenge-upc-2016

ResNet (2015)

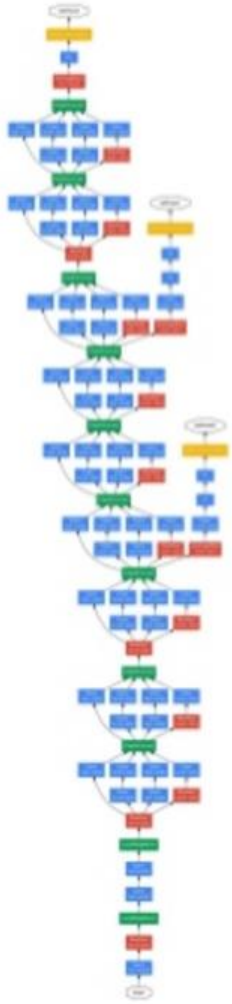
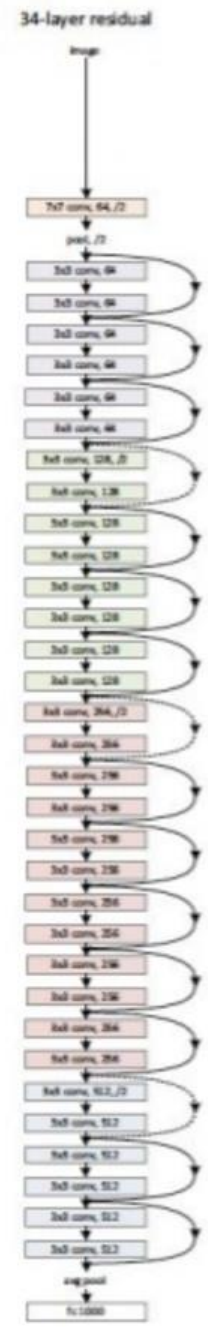
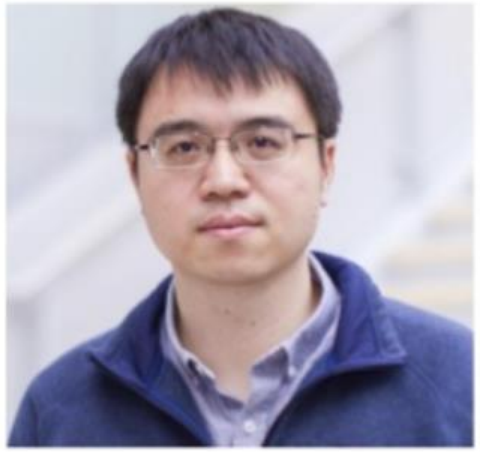


image
conv-64
conv-64
maxpool
conv-128
conv-128
maxpool
conv-256
conv-256
conv-256
conv-256
maxpool
conv-512
conv-512
conv-512
conv-512
maxpool
conv-512
conv-512
conv-512
conv-512
maxpool
FC-4096
FC-4096
FC-1000
softmax





Microsoft Research

3.6% top 5 error…
with 152 layers !!

https://www.slideshare.net/xavigiro/deep-learning-for-computer-vision-imagenet-challenge-upc-2016

Другие задачи

Image Classification

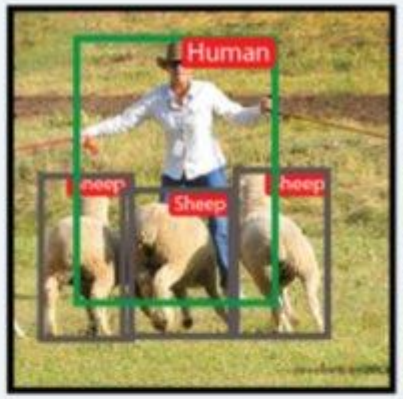
Classify an image based on the dominant object inside it.

datasets: MNIST, CIFAR, ImageNet

Object Localization

Predict the image region that contains the dominant object. Then image classification can be used to recognize object in the region

datasets: ImageNet



Object Recognition

Localize and classify all objects appearing in the image. This task typically includes: proposing regions then classify the object inside them.

datasets: PASCAL, COCO

Semantic Segmentation

Label each pixel of an image by the object class that it belongs to, such as human, sheep, and grass in the example.

datasets: PASCAL, COCO

Instance Segmentation

Label each pixel of an image by the object class and object instance that it belongs to.

datasets: PASCAL, COCO

Keypoint Detection

Detect locations of a set of predefined keypoints of an object, such as keypoints in a human body, or a human face.

datasets: COCO

Сегментация

Full-Resolution Residual Networks for Semantic Segmentation in Street Scenes

Tobias Pohlen, Alexander Hermans,
Markus Mathias, Bastian Leibe

Visual Computing Institute, Computer Vision Group
RWTH Aachen University

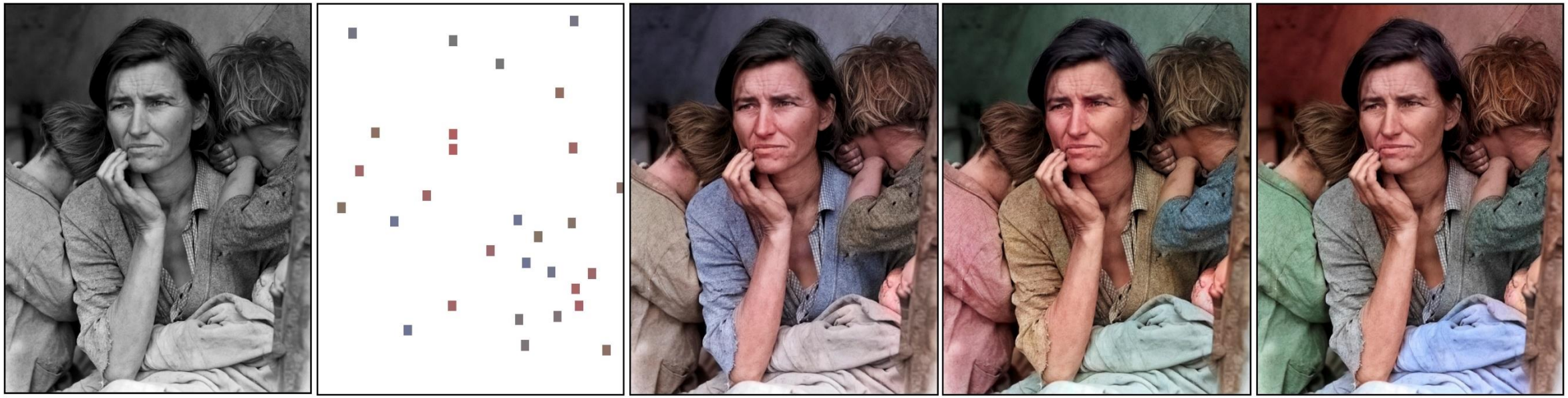
Распознавание позы

Real-time Multi-Person 2D Pose Estimation Using Part Affinity Fields

Zhe Cao, Tomas Simon, Shih-En Wei, Yaser Sheikh

Carnegie Mellon University

Раскрашивание изображений

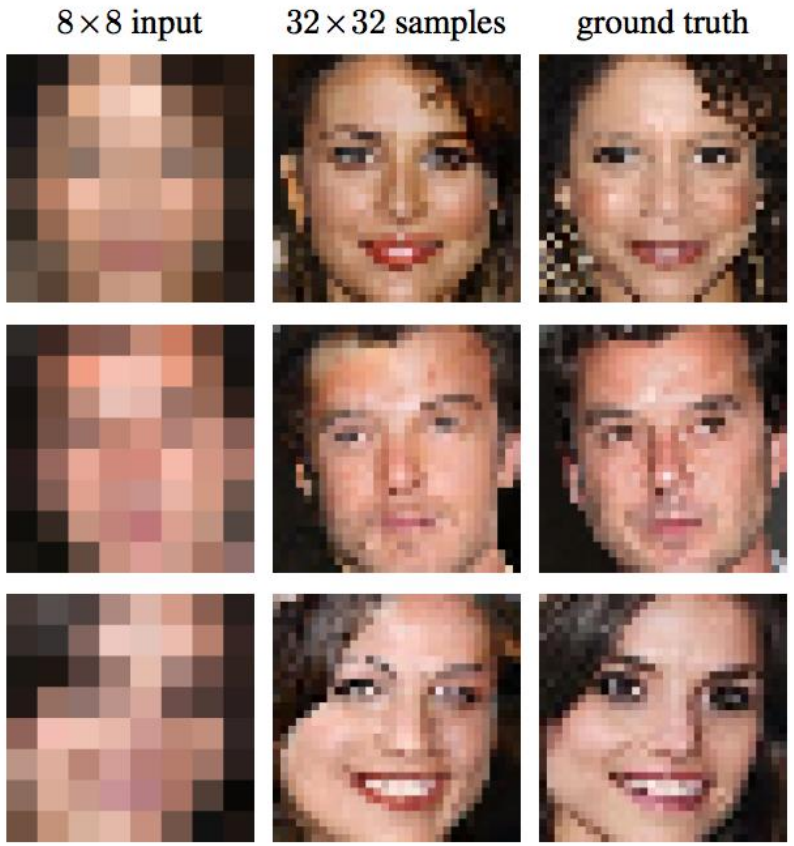


https://richzhang.github.io/ideepcolor/

Дорисовка изображений

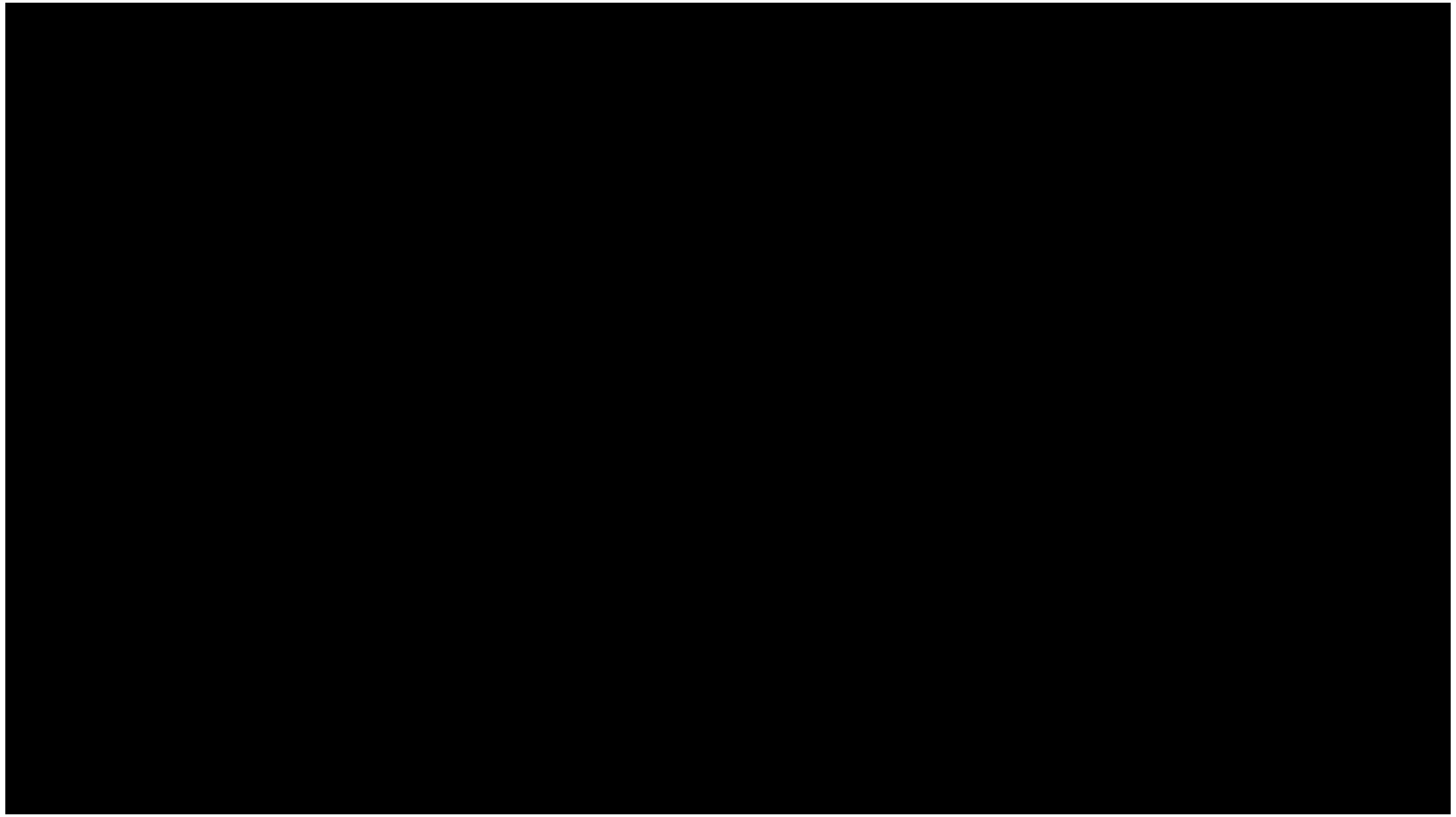
http://bamos.github.io/2016/08/09/deep-completion/

Дорисовка изображений

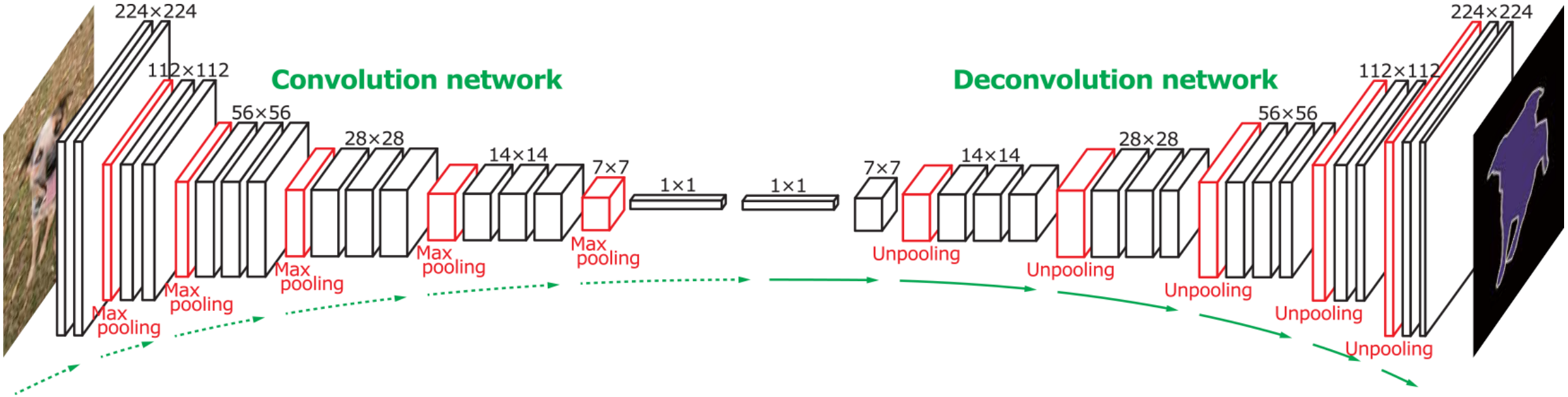


Дорисовка изображений

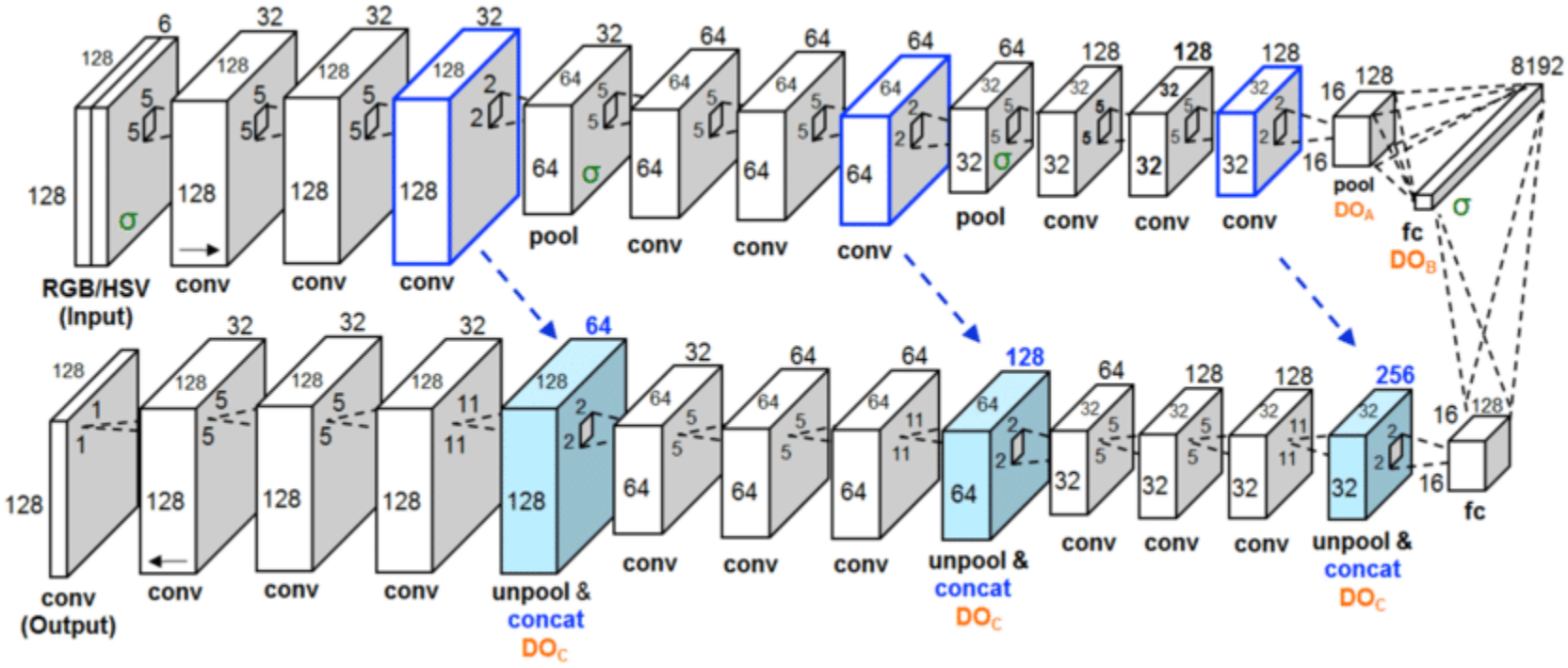
Дорисовка планов



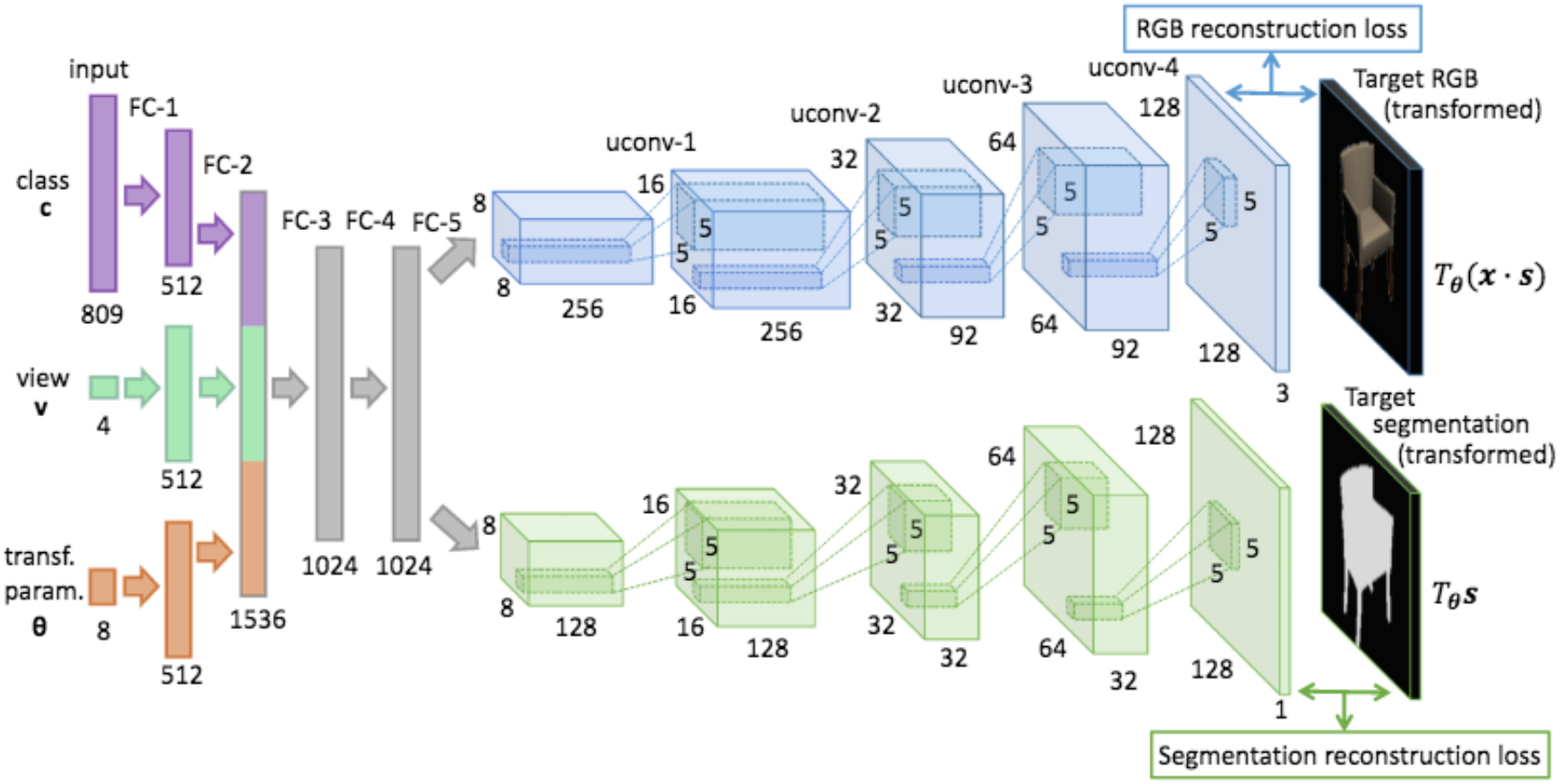
Другие архитектуры



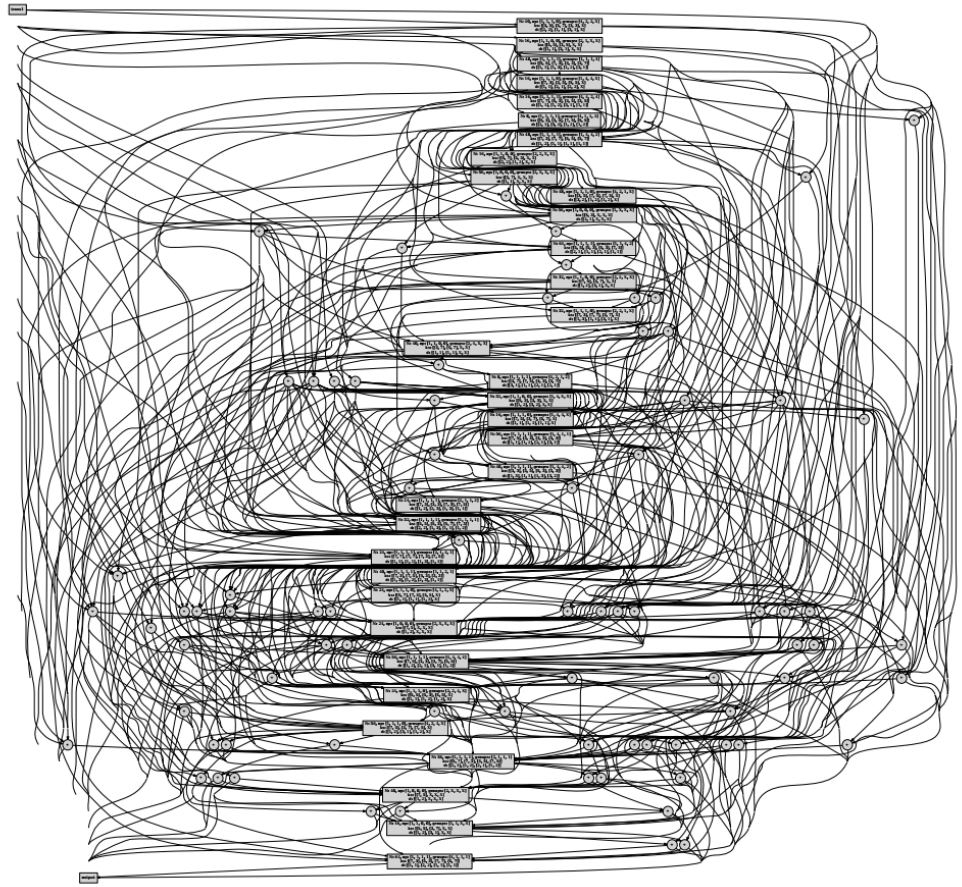
Другие архитектуры



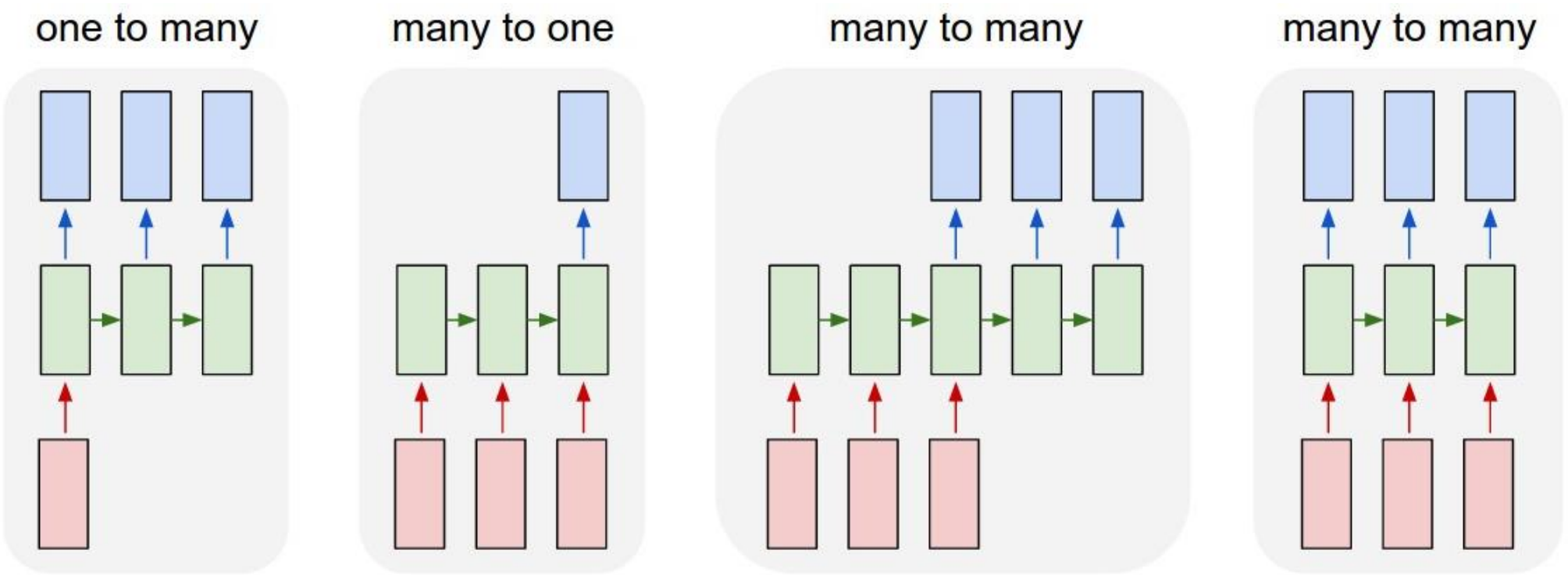
Другие архитектуры



Другие архитектуры



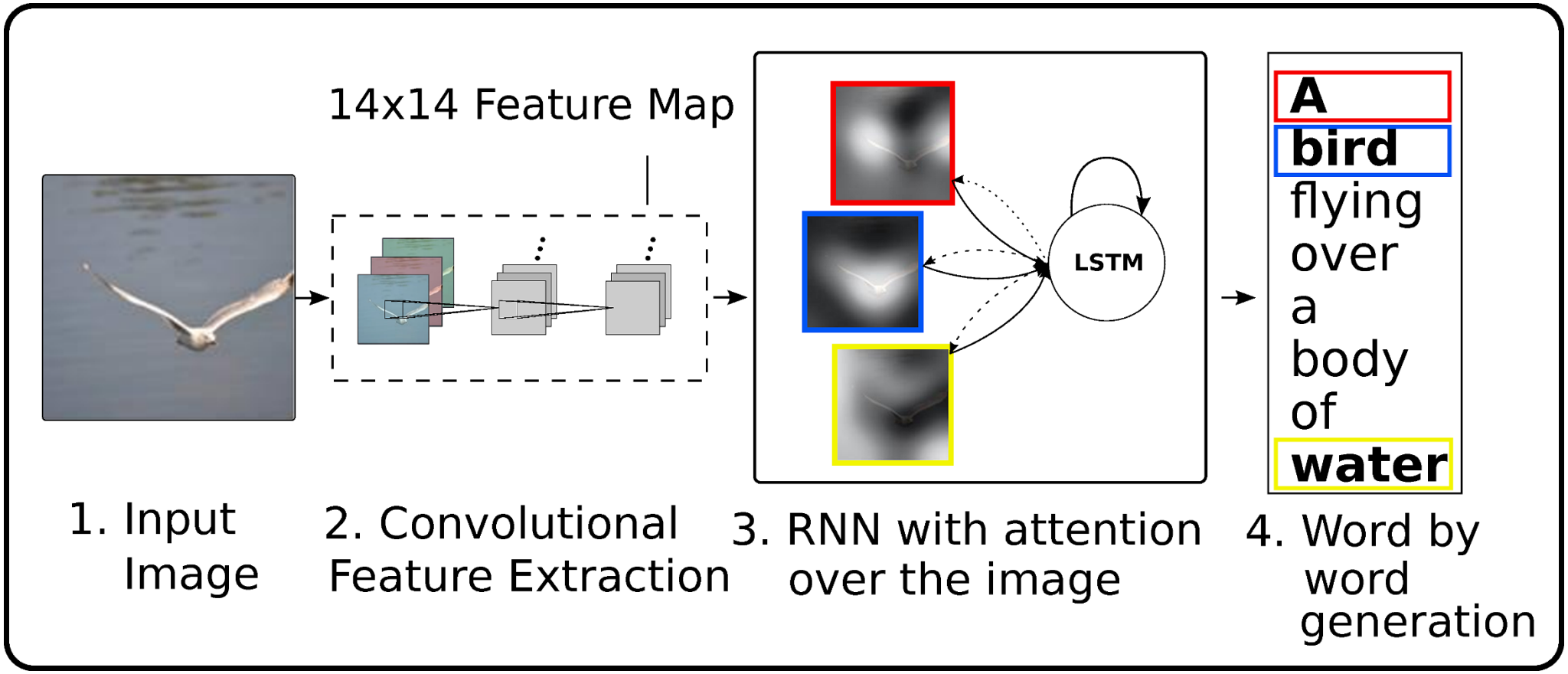
Рекуррентные сети



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

Рекуррентные сети

Генерация описаний



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

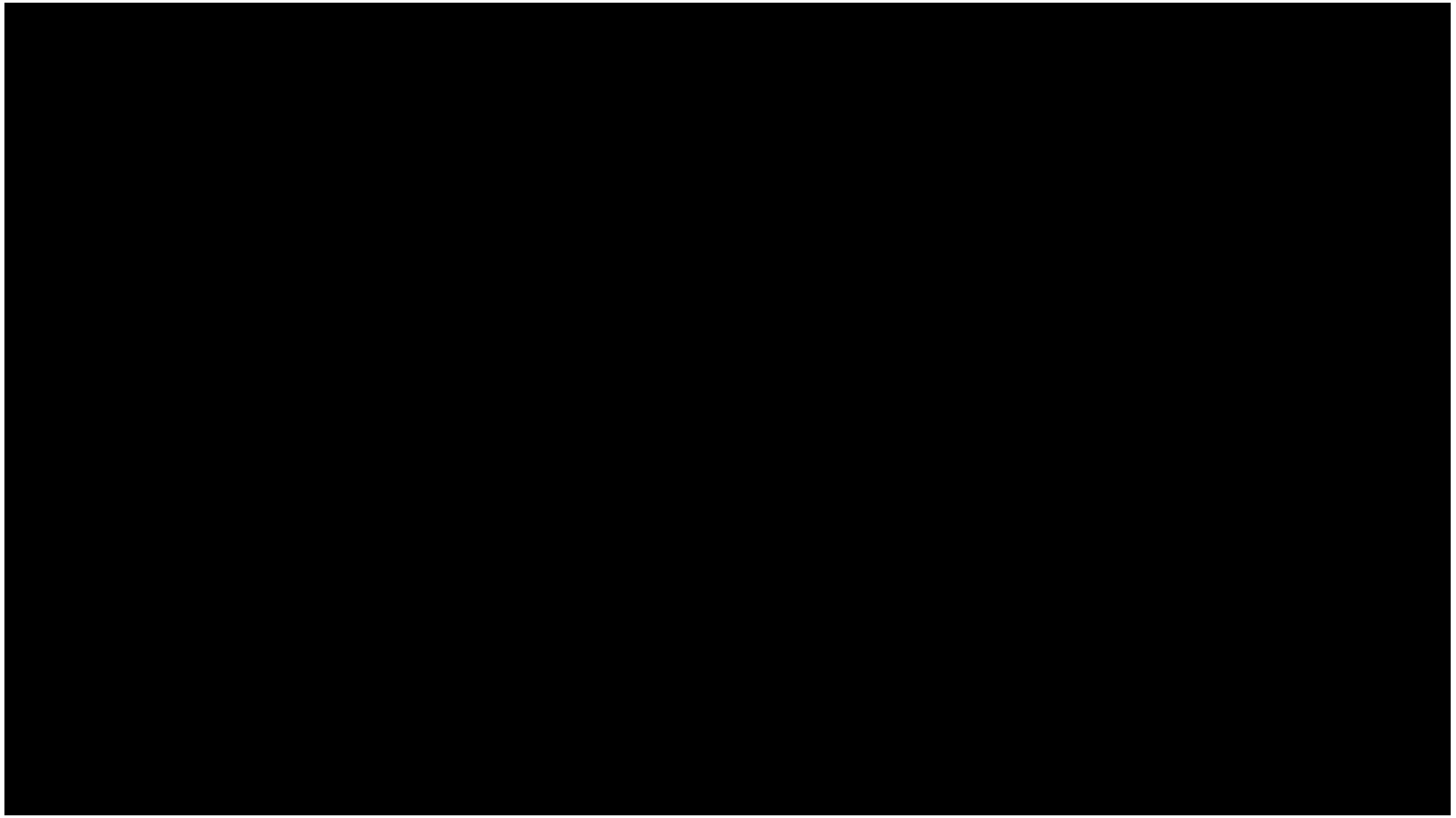
Генерация описаний

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

Генерация изображений по описанию

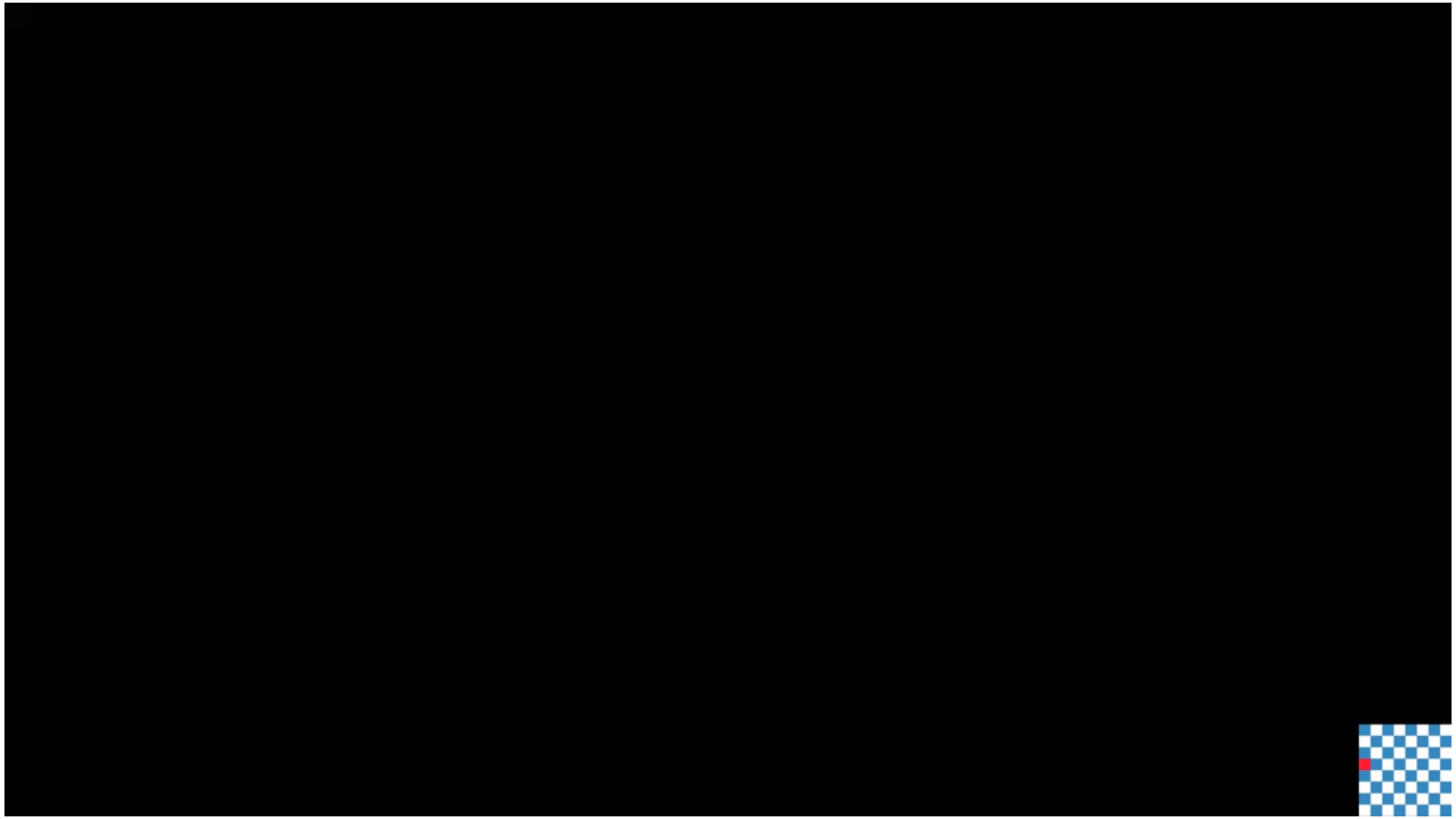
Figure 3. Example results by our proposed StackGAN, GAWWN [20], and GAN-INT-CLS [22] conditioned on text descriptions from CUB test set. GAWWN and GAN-INT-CLS generate 16 images for each text description, respectively. We select the best one for each of them to compare with our StackGAN.

LipNet



Добавление звука к видео

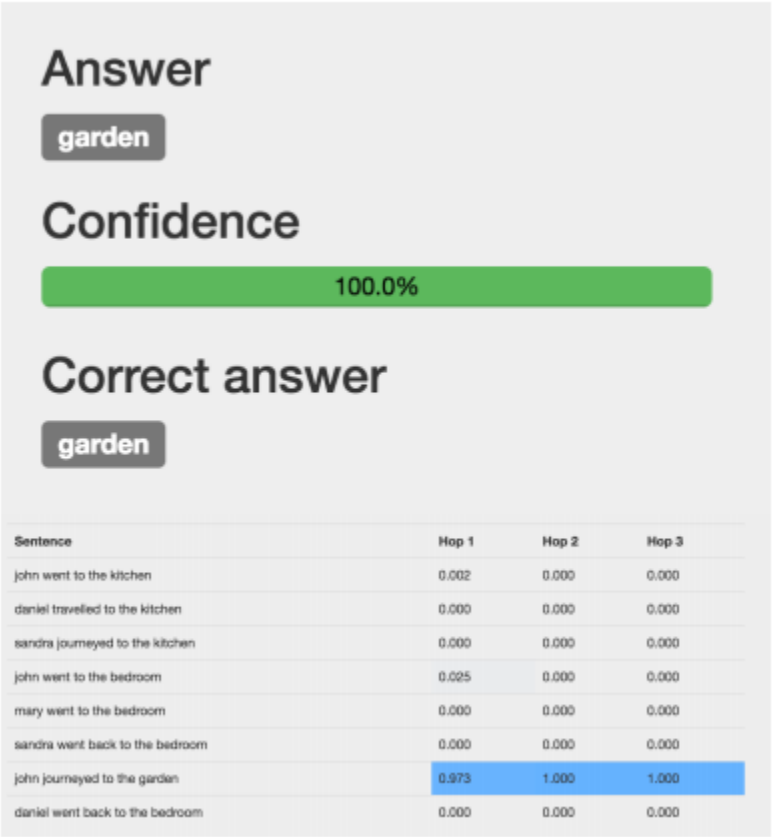
WaveNet



Л. Логика

Task 1
Story
john went to the kitchen
daniel travelled to the kitchen
sandra journeyed to the kitchen
john went to the bedroom
mary went to the bedroom
sandra went back to the bedroom
john journeyed to the garden
daniel went back to the bedroom

Question
where is john



Sentence	Hop 1	Hop 2	Hop 3
john went to the kitchen	0.002	0.000	0.000
daniel travelled to the kitchen	0.000	0.000	0.000
sandra journeyed to the kitchen	0.000	0.000	0.000
john went to the bedroom	0.025	0.000	0.000
mary went to the bedroom	0.000	0.000	0.000
sandra went back to the bedroom	0.000	0.000	0.000
john journeyed to the garden	0.973	1.000	1.000
daniel went back to the bedroom	0.000	0.000	0.000

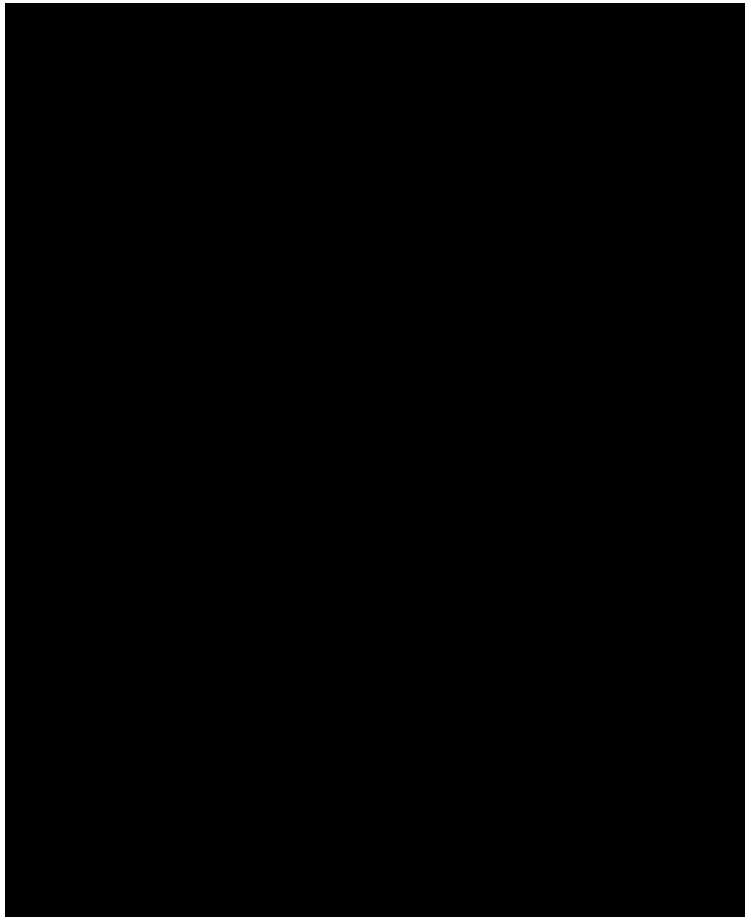
Роботика

Роботика

Роботика

Игры: Go

Игры: Atari



Игры: Doom


50
AMMO
100%
HEALTH
0
FRAG
0%
ARMOR
BULL 50 / 200
SHEL 0 / 50
RCKT 0 / 500
CELL 0 / 300

Игры: GTA5

- https://www.twitch.tv/sentdex

Индуктивное моделирование систем с помощью численной оптимизации структурированных функций


Theano, Tensorflow, Keras


Theano, Tensorflow, Keras,
PyTorch, MXNet, CNTK, Lasagne,
PyLearn2, DyNet, Chainer, Caffe,
DeepLearning4J, …

Deep Learning Book
http://www.deeplearningbook.org/

Practical Deep Learning for Coders
http://www.fast.ai/

Coursera: **Deep Learning Specialization**
(Andrew Ng, deeplearning.ai)

Coursera: **Neural Networks for Machine Learning**
(Geoffrey Hinton, U. of Toronto)

+ Kaggle,
CodaLab,
TopCoder,
…