POLYTECH

Распознавание искусственным интеллектом

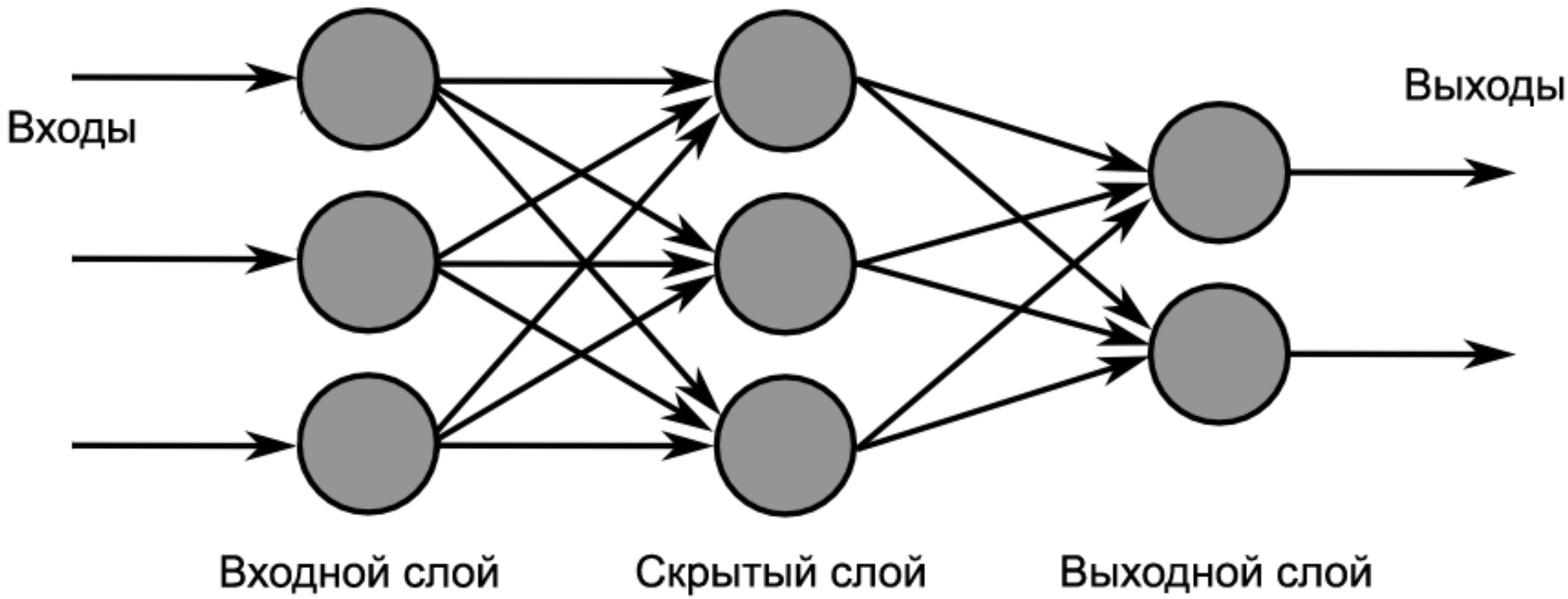
Константин Кориков | 2019

Агенда

1. Модель нейронной сети
2. Обучение нейронной сети
3. Нейронные сети с Python

Модель нейронной сети

Многослойный перцептрон (MLP)



Математическое описание

$$\vec{y} = f(W\vec{x})$$

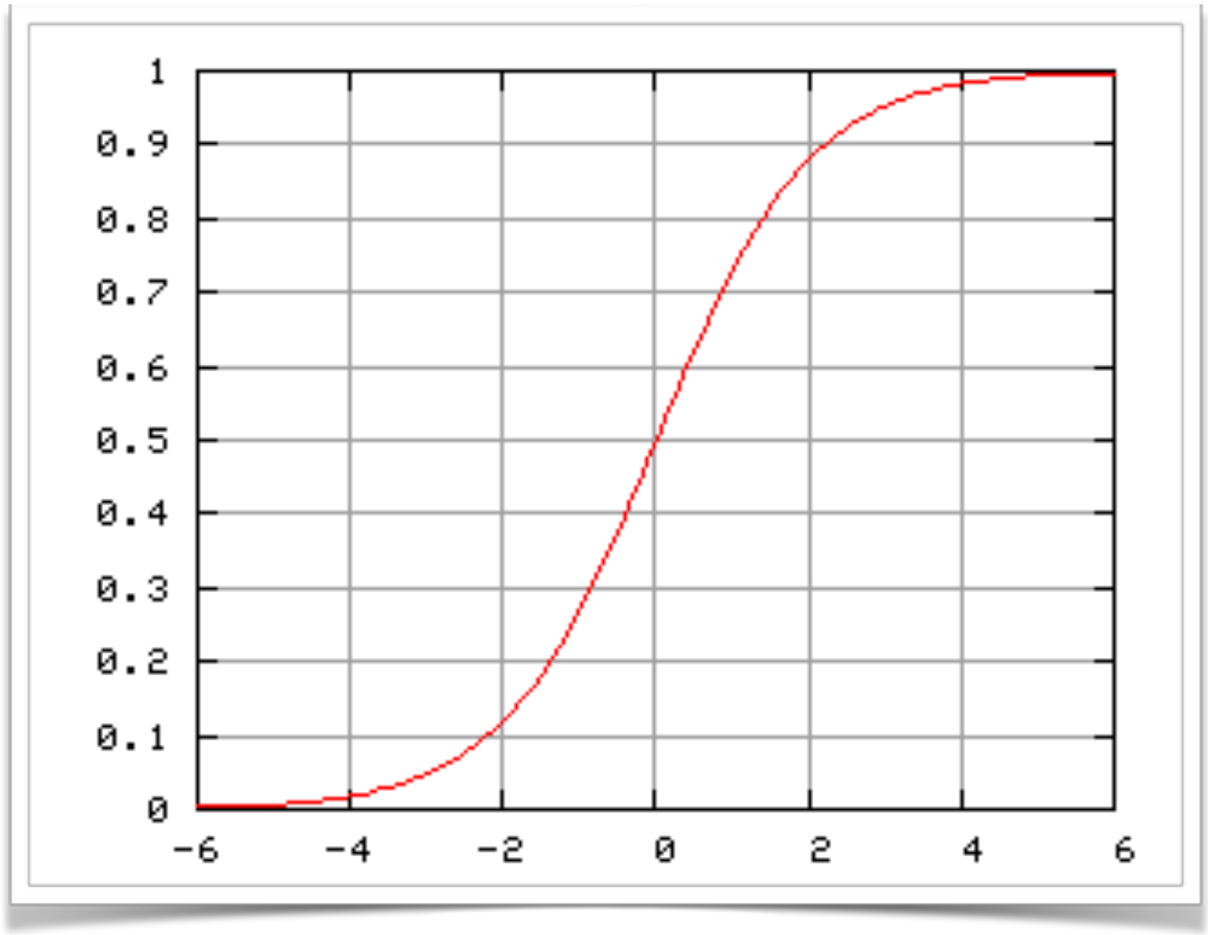
W — матрица весов

Математическое описание

$$\vec{y} = f(W^{(2)} g(W^{(1)} \vec{x}))$$

Композиция

Математическое описание



$$\sigma(a) = \frac{1}{1 + e^{-x}}$$

Обучение нейронной сети

Градиентный спуск: повторение

$$\vec{\omega}^{(\tau+1)} = \vec{\omega}^{(\tau)} - \eta \sum_{(\vec{x},\hat{y})\in\mathcal{D}} \nabla_{\vec{\omega}} E[y(\vec{x},\vec{\omega}^{(\tau)}),\hat{y}]$$

Градиентный спуск по батчам

Градиентный спуск: повторение

$$\vec{\omega}^{(\tau+1)} = \vec{\omega}^{(\tau)} - \eta \sum_{(\vec{x},\hat{y})\in\mathcal{D}'} \nabla_{\vec{\omega}} E[y(\vec{x},\vec{\omega}^{(\tau)}),\hat{y}]$$

Градиентный спуск по мини-батчам

Градиентный спуск: повторение

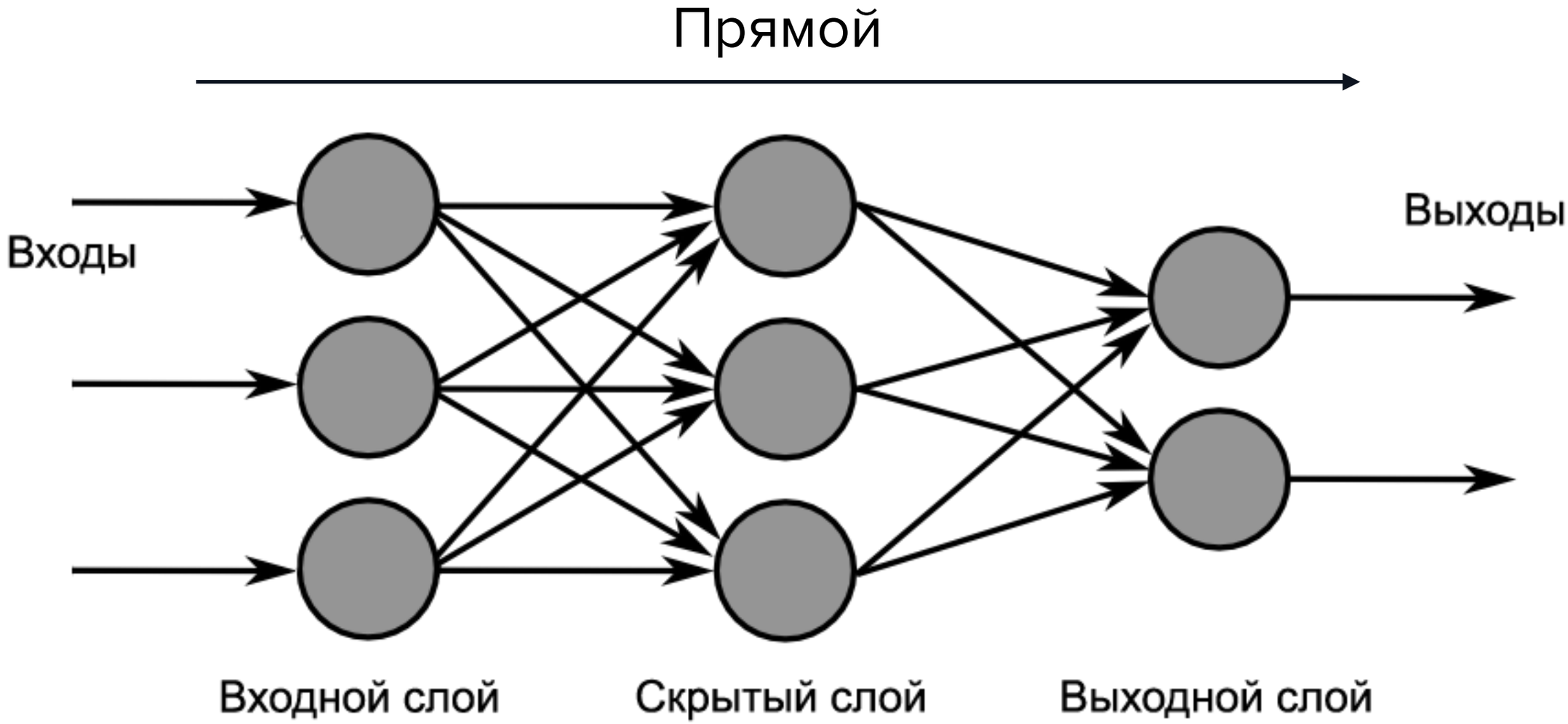
$$\vec{\omega}^{(\tau+1)} = \vec{\omega}^{(\tau)} - \eta \nabla_{\vec{\omega}} E[y(\vec{x}^{(\tau)}, \vec{\omega}^{(\tau)}), \hat{y}^{(\tau)}]$$

Стохастический градиентный спуск

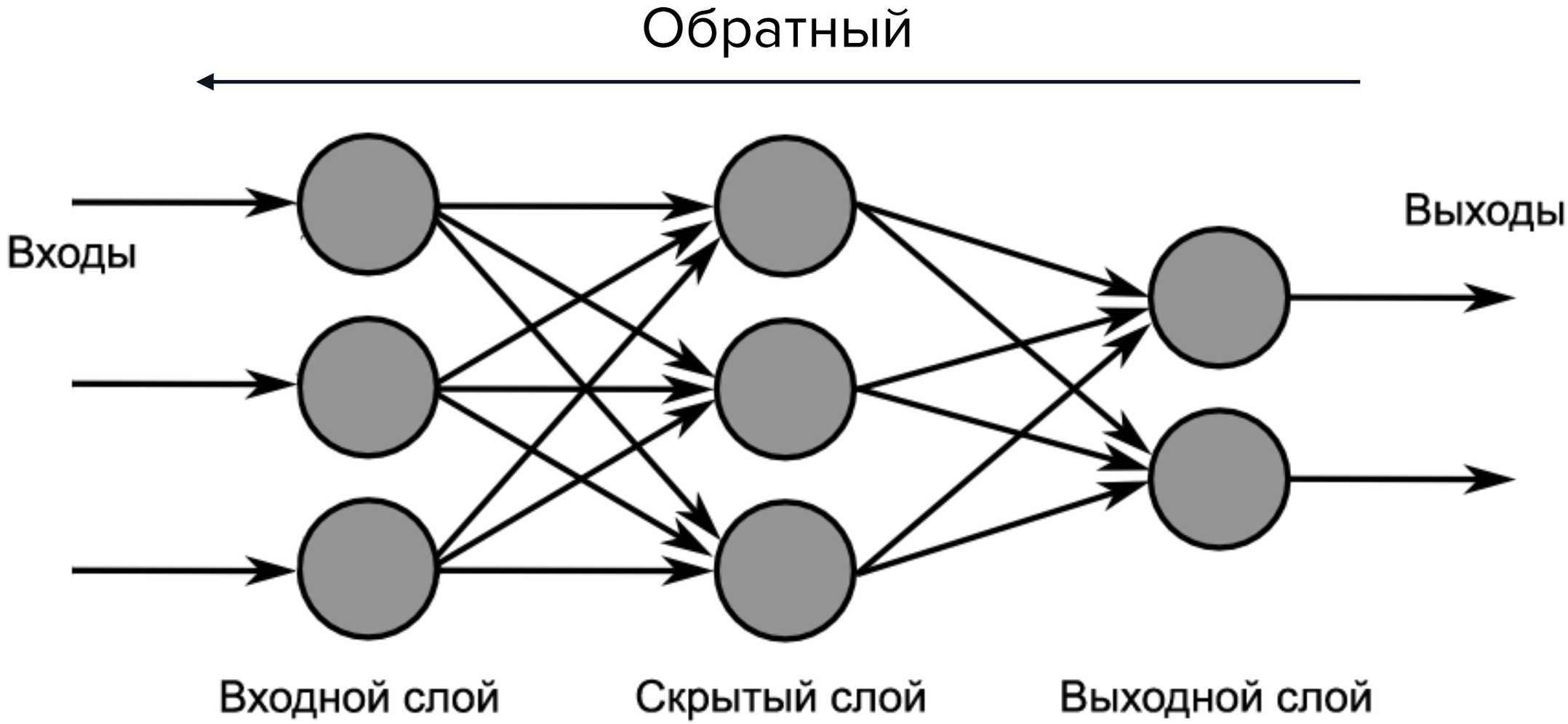
Функция ошибки

$$E = \frac{1}{2} \sum_i (y_i - \hat{y}_i)^2$$

Обратное распространение ошибки



Обратное распространение ошибки

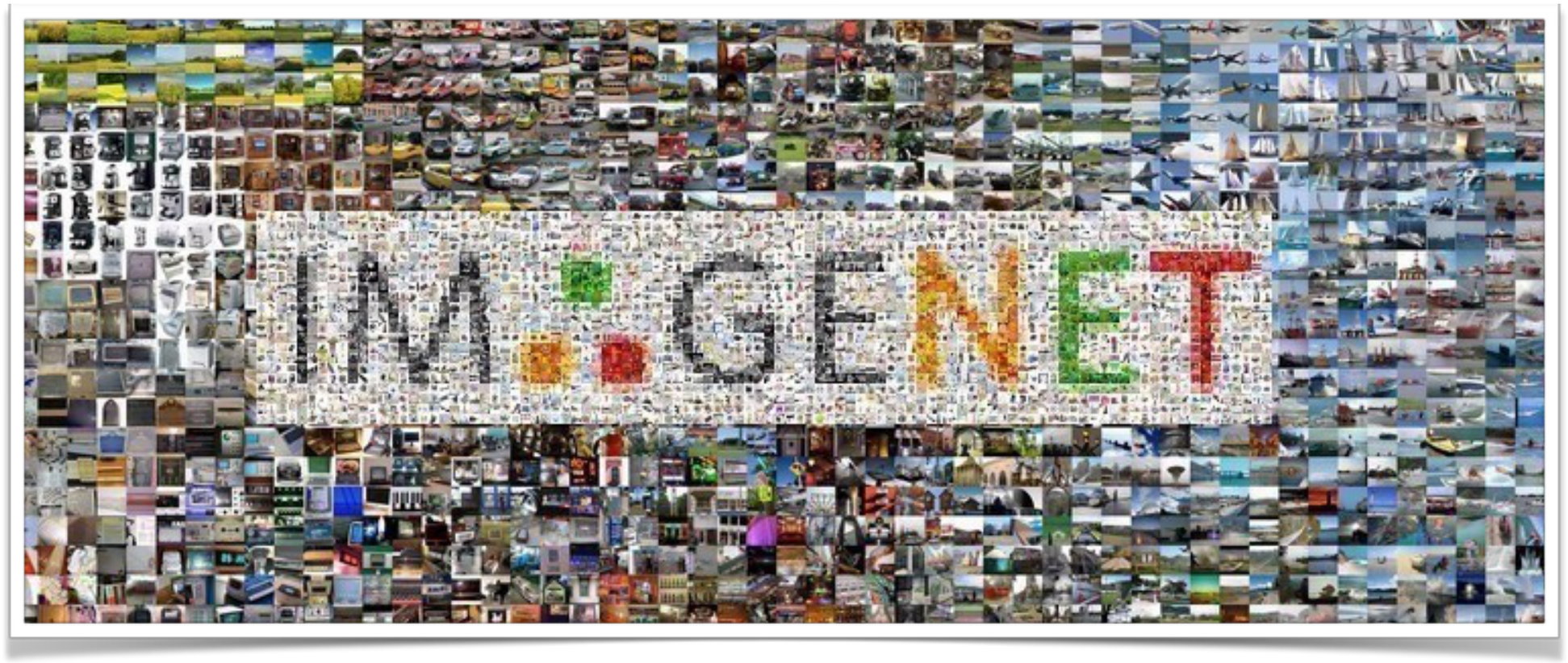


Нейронные сети с Python

Keras

Python-библиотека

Keras



Dataset

Keras

```python
# Необходимые библиотеки
from keras.applications.resnet50 import ResNet50, preprocess_input, decode_predictions
from keras.preprocessing import image
import urllib.request
import numpy as np

# Нейронная сеть с архитектурой ResNet50, обученная на imagenet dataset
model = ResNet50(weights="imagenet")

# Путь до изображения
img_url = "%путь до изображения%"

# Загружаем файл из интернета с картинкой и выполняем предобработку
tf,*_ =urllib.request.urlretrieve(img_url, "pic.jpg")
img = image.load_img(tf, target_size=(224, 224))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
x = preprocess_input(x)

# Обучение модели, вывод классов
preds = model.predict(x)
decode_predictions(preds, top=5)
```

WeChat