ИТМО

Практика. Состязательные атаки на системы ИИ

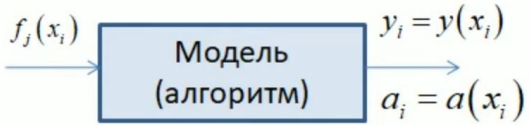
Воробьева А.А. – к.т.н., доцент ФБИТ
Сафиуллин Р.И. – магистрант ФБИТ

Базовые знания

$X^l = \{(x_i, y_i)_{i=1}^l\}$ - обучающая выборка

$F = \|f_j(x_i)\|_{l \times n}$ - признаковое пространство

$f_j(x_i) = x_{ij}, \quad j = 1, 2, \ldots$



$a_i = a(x_i)$ - модель, решающая функция (decision function)

$a(x) = g(x, \theta)$

$\theta = [\theta_1, \theta_2, \ldots, \theta_n]^T$ - подбираемые параметры

Базовые знания

Критерий качества модели

$a(x) = g(x, \theta)$ $\quad X \times Y$

$a(x) - y(x), \quad \forall x$

$L(a,x) = |a(x) - y(x)|$ - абсолютная ошибка;

$L(a,x) = (a(x) - y(x))^2$ - квадратичная ошибка.

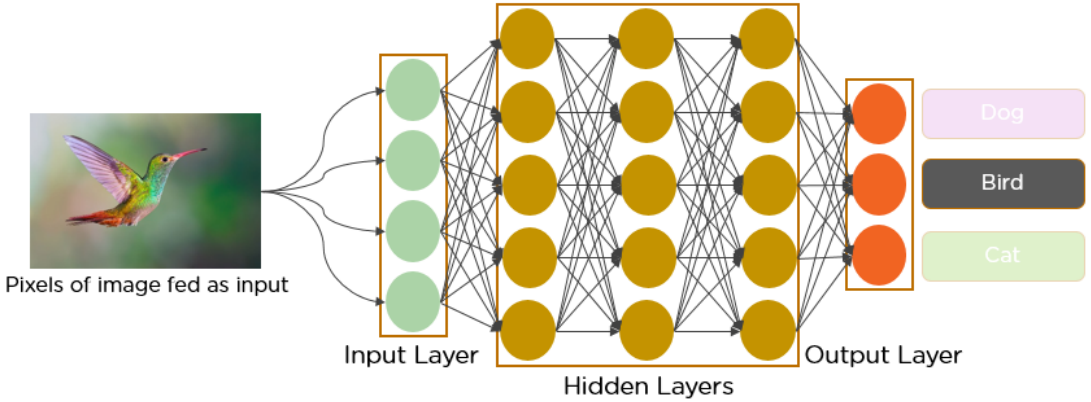
функции потерь (loss function)

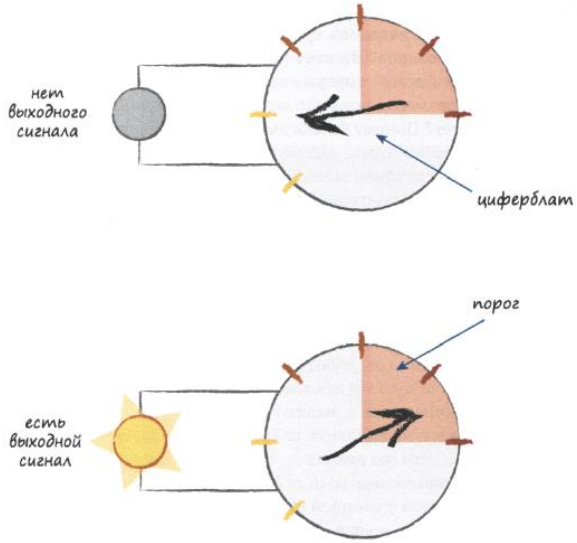
$$Q(a, X^l) = \frac{1}{l} \sum_{i=1}^{l} L(a, x_i)$$

средний эмпирический риск

$X^l = \{(x_i, y_i)_{i=1}^l\}$ - обучающее множество

Нейронные сети. Активация нейрона

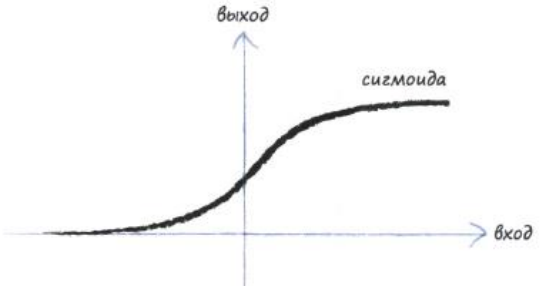




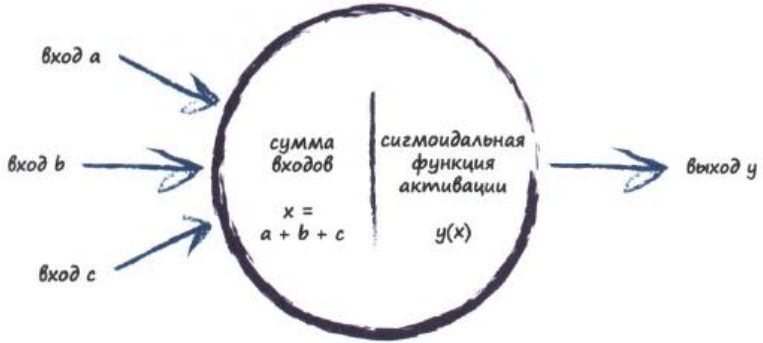
Входной сигнал -> нейрон -> Выходной сигнал (возникает, если превышен некоторый порог)

Нейронные сети

Функция активации - функция, которая получает входной сигнал, но генерирует выходной сигнал с учетом порогового значения (чаще используется сигмоидная функция)**.**



$$y = \frac{1}{1 + e^{-x}}$$



Если комбинированный сигнал недостаточно сильный, то сигмоида подавляет выходной сигнал.

Если же сумма **x** достаточно велика, то функция возбуждает нейрон

Нейронные сети

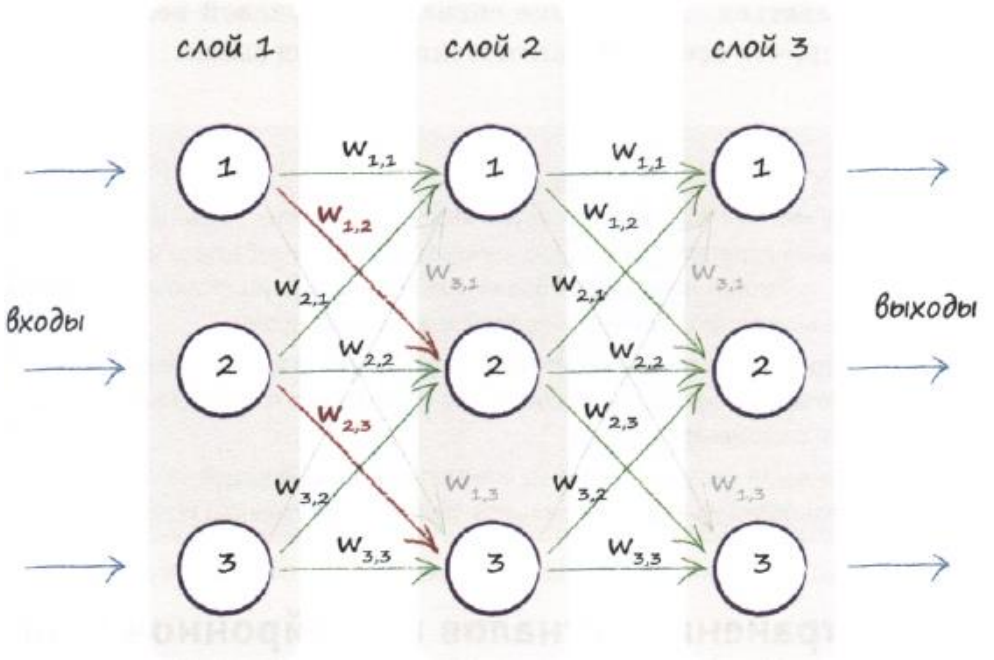
Где же обучение?

Выходы из каждого нейрона предыдущего слоя связаны со всеми нейронами следующего слоя.

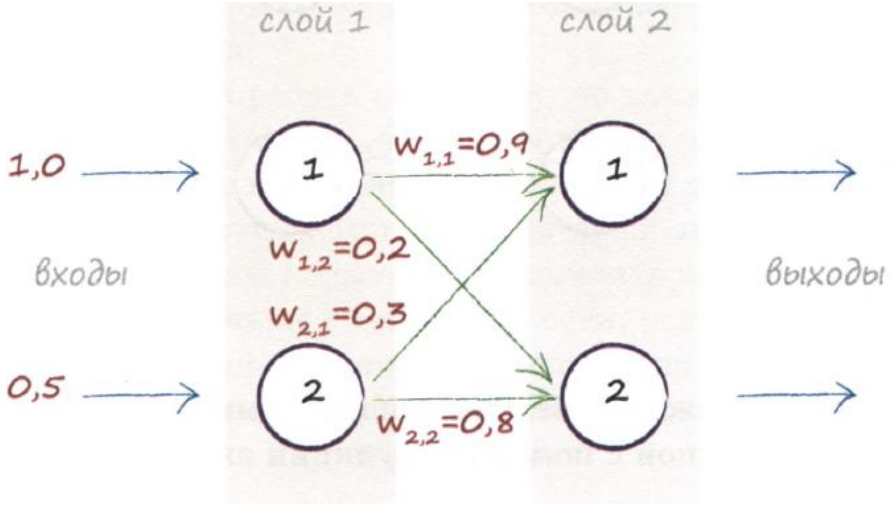
Но! Плохие связи не должны оказывать влияния на сеть!

Обучение = уточнение весовых коэффициентов

В процессе обучения плохие связи «обнуляются»



Нейронные сети

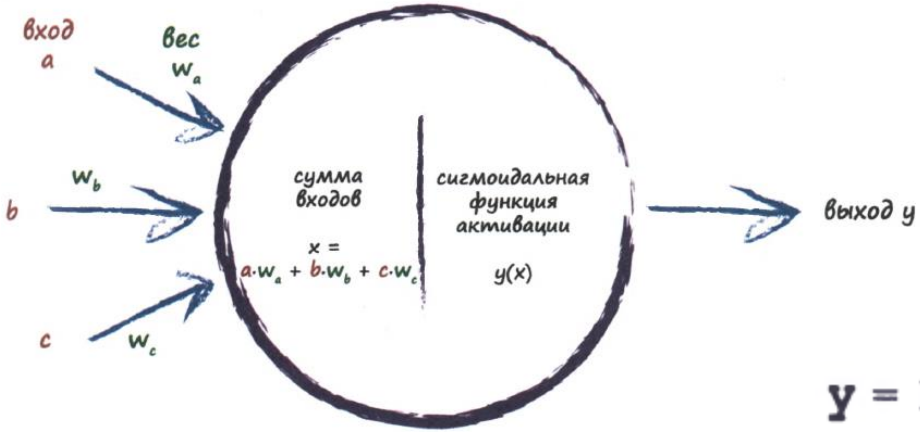


Изначальные значения весов могут быть рандомны.

Нейронные сети

$$y = \frac{1}{1 + e^{-x}}$$

где x — это сумма сигналов, поступающих в нейрон,
y — выходной сигнал этого нейрона



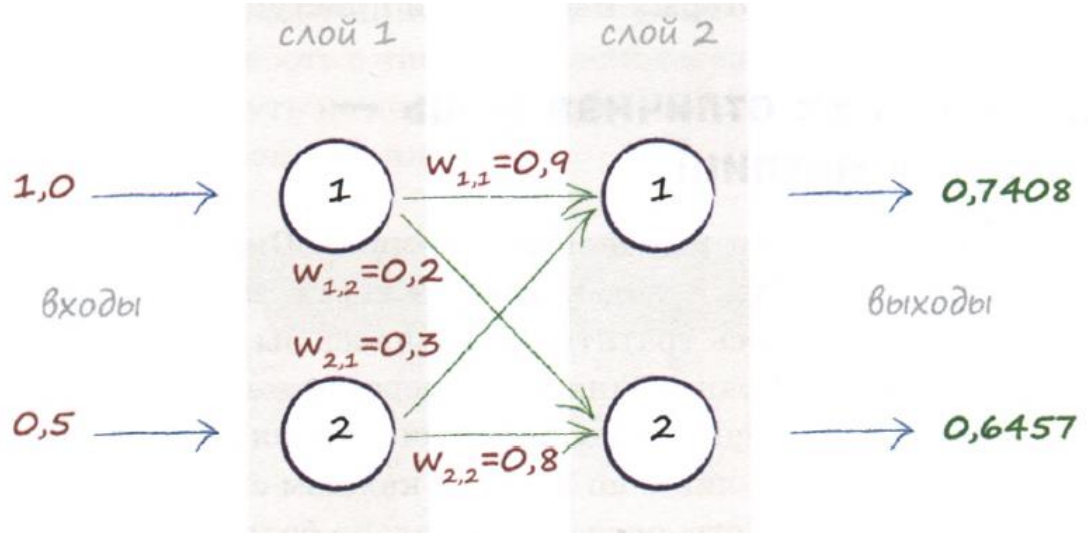
Для первого узла второго слоя

x = (*выход первого узла* * *вес связи*) +
+ (*выход второго узла* * *вес связи*)
x = (1,0 * 0,9) + (0,5 * 0,3)
x = 0,9 + 0,15
x = 1,05

y = 1 / (1 + 0,5488) = 1 / 1,5488

y=0,6457

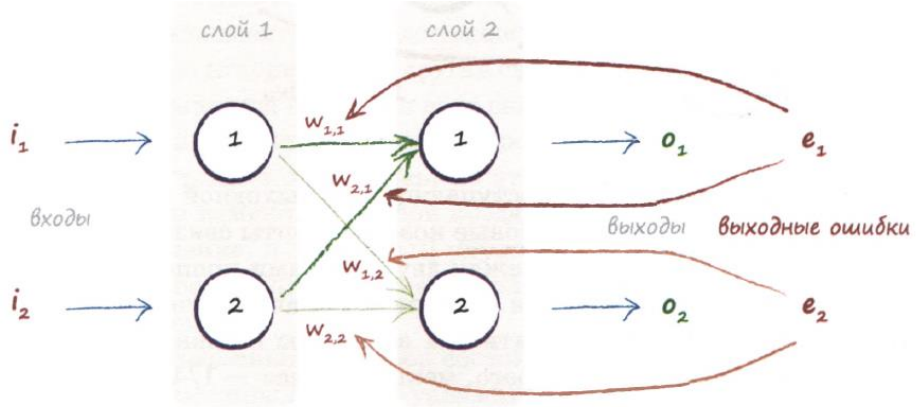
Нейронные сети



Нейронные сети. Как обновлять веса?

Предназначение весов:

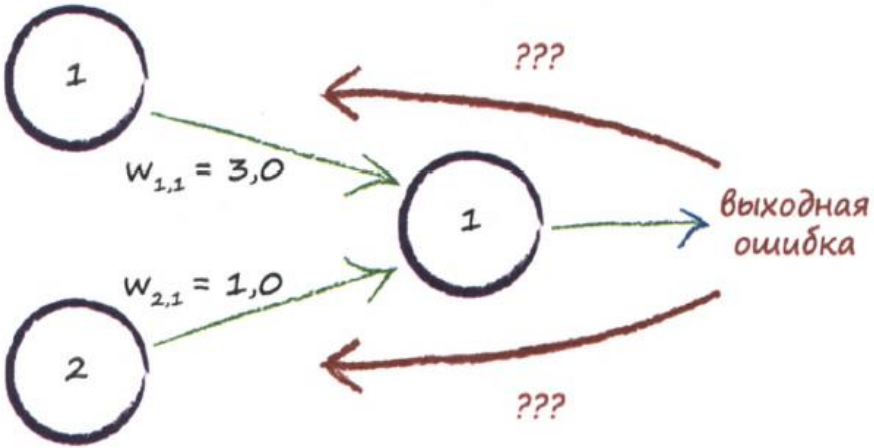
- учитываются при расчете распространения сигналов по нейронной сети от входного слоя до выходного.
- используются для распространения ошибки в обратном направлении (**Обратное распространение ошибки**).



ошибка E - разность между желаемым корректным значением **e (из тренировочных данных)** и фактическим выходным значением **o**, полученным для текущего пробного значения веса.

Нейронные сети. Как обновлять веса?

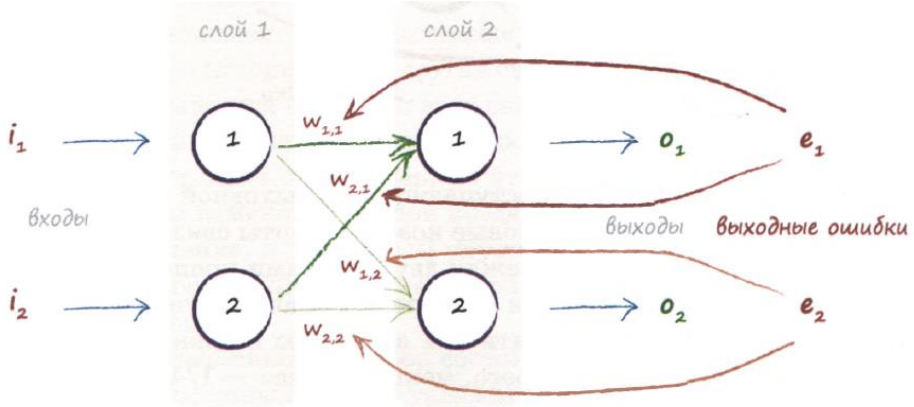
Как обновлять весовые коэффициенты связей в случае, если выходной сигнал и его ошибка формируются за счет вкладов более чем одного узла?



Большая доля ошибки приписывается вкладам тех связей, которые имеют больший вес.

Нейронные сети. Как обновлять веса?

Например, **для первого узла e1=t1-o1** (информирует о величинах поправок для весов w11 и w21).



При распределении ошибки между узлами доля **e1,** информация о которой используется для обновления w11, определяется выражением:

$$\frac{w_{11}}{w_{11} + w_{21}}$$

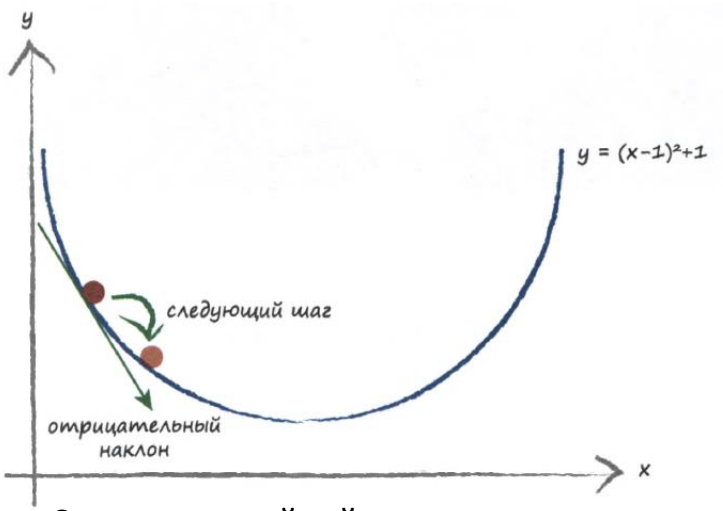
$$\frac{w_{21}}{w_{11} + w_{21}}$$ (для w21)

Нейронные сети. Как обучается сеть?

- Нейронные сети обучаются посредством уточнения весовых коэффициентов своих связей. Этот процесс управляется ошибкой — разностью между правильным ответом, предоставляемым тренировочными данными, и фактическим выходным значением.
- Ошибка на выходных узлах определяется простой разностью между желаемым и фактическим выходными значениями.
- В то же время величина ошибки, связанной с внутренними узлами, не столь очевидна. Одним из способов решения этой проблемы является распределение ошибок выходного слоя между соответствующими связями пропорционально весу каждой связи с последующим объединением соответствующих разрозненных частей ошибки на каждом внутреннем узле

Нейронные сети. Как обучается сеть?

Задача – минимизация ошибки сети.
Как выглядит функция ошибки? Как найти ее минимум?
Методом градиентного спуска! **(если интересно, почитайте сами. Это выходит за рамки этой практики)**



Старт со случайной точки

Наклон кривой (положительный или отрицательный) соответствует знаку градиента. Сдвигаем x в направлении обратном знаку градиента.

Нейронные сети. Как обучается сеть?

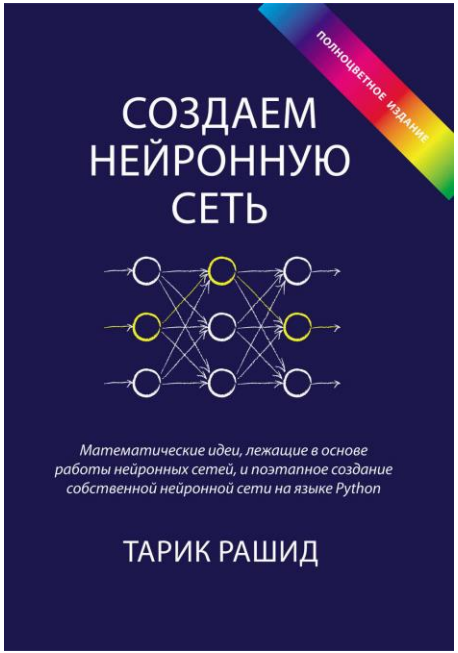
- Ошибка нейронной сети является функцией весов внутренних связей.
- Улучшение нейронной сети означает уменьшение этой ошибки посредством изменения указанных весов.
- Непосредственный подбор подходящих весов наталкивается на значительные трудности. Альтернативный подход заключается в итеративном улучшении весовых коэффициентов путем уменьшения функции ошибки небольшими шагами. Каждый шаг совершается в направлении скорейшего спуска из текущей позиции. Этот подход называется градиентным спуском.
- Градиент ошибки можно без особых трудностей рассчитать, используя дифференциальное исчисление ☺.

Хорошая книга по NN

СОЗДАЕМ НЕЙРОННУЮ СЕТЬ

Тарик Рашид

(иллюстрации и часть текстов оттуда)



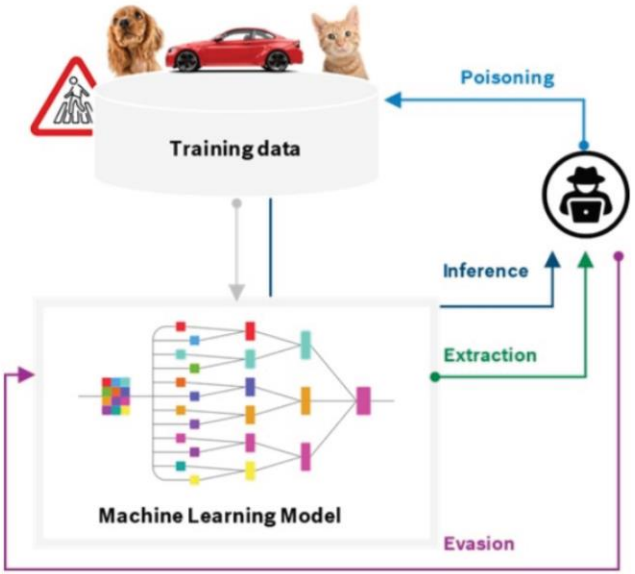
CNN (Свёрточные нейронные сети)

YouTube плейлист, CNN на русском

Атаки на системы ИИ

Основные типы атак на системы ИИ делятся на следующие 4 класса:

- Отравление
- Уклонение
- Атака инверсии модели
- Атака установления принадлежности



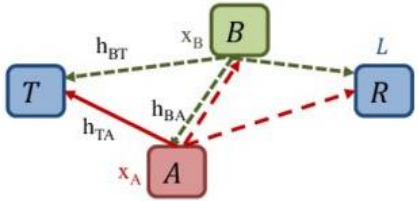
Атаки уклонения (evasion)

- Злоумышленнику известны «ответы» системы. У него есть возможность использовать сгенерированные состязательные примеры.
- Злоумышленник пытается обойти систему путем корректировки состязательных данных во время фазы тестирования.
- Атака не предполагает какого-либо влияния на обучающие данные.

It’s a Black box

Evasion attack

- T runs test data $D = h_{BT}\, x_B + n_T$ from B with the trained classifier to make a transmission decision.



- A transmits and *changes* test data of T from $h_{BT}\, x_B + n_T$ to $h_{BT}\, x_B + h_{AT}\, x_A + n_T$ over the air and T makes wrong decision.

Атаки уклонения (evasion)

Поиск вредоносных примеров, которые содержат незначительные искажения и практически не отличаются от обычных примеров.

- Основанные на алгоритмах градиентного поиска
- Безградиентные

Не изменяют обучающий набор данных.

В **атаках уклонения** состязательные примеры генерируются для того, чтобы уклониться от корректной классификации.

Атаки уклонения (evasion)

Атаки с алгоритмом Broyden-Fletcher-Goldfarb-Shanno с ограниченной памятью (L-BFGS)

Intriguing properties of neural networks

Атаки методом быстрого градиента (FGSM)

Explaining and Harnessing Adversarial Examples

Атака Carlini & Wagner (C&W)

Towards Evaluating the Robustness of Neural Networks

Атаки методом карт значимости (JSMA)

The Limitations of Deep Learning in Adversarial Settings

FGSM (Fast Gradient Sign Method)

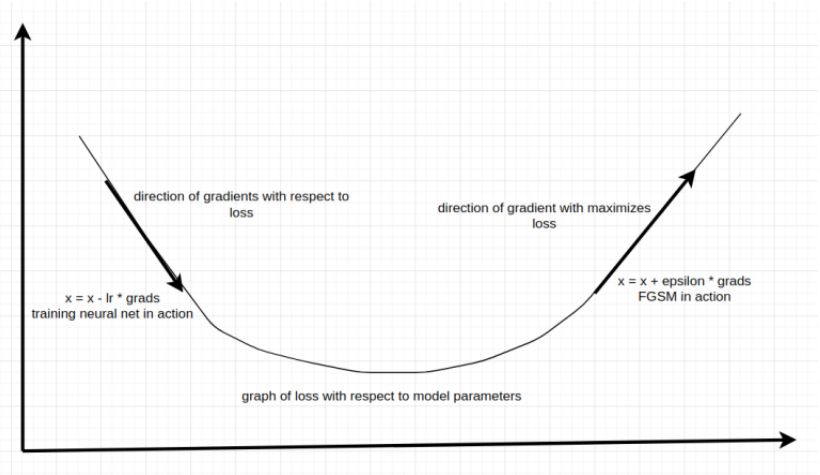
- Получение входного изображения;
- Составление предсказания с использованием CNN (convolutional neural network);
- Вычисление значения loss (среднеквадратичная ошибка) модели;
- Вычисление градиентов loss по отношению к пикселям входного изображения (в какие моменты значение увеличивается);
- Использование градиентов для построения выходного состязательного изображения (изменяем пиксели в направлении градиентов, чтобы максимизировать loss).

Простыми словами: злоумышленник, имеющий доступ к данным на которых обучалась нейронная сеть, может сгенерировать новые данные таким образом, чтобы максимизировать уровень ошибок модели.

FGSM (Fast Gradient Sign Method)

Вычисление градиентов - это метод с помощью которого вы определяете направление, в котором нужно изменить веса модели, чтобы уменьшить значение потерь (в направлении противоположном градиенту).

Вместо этого, злоумышленник корректирует пиксели входного изображения в направлении градиентов, чтобы максимизировать значение потерь.



FGSM (Fast Gradient Sign Method)

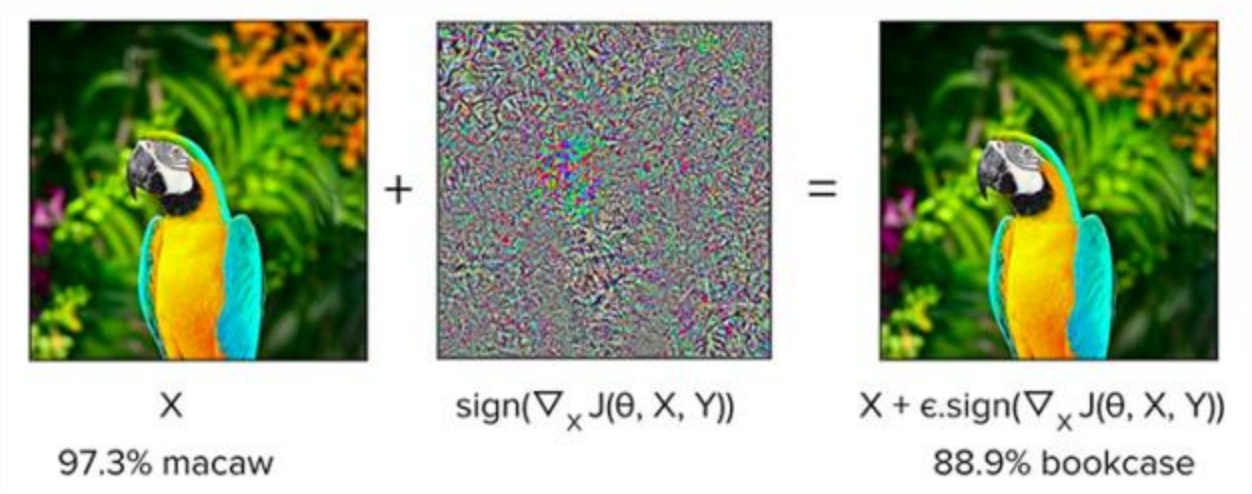
1) Вычисляет градиенты функции потерь (среднеквадратичную ошибку) по отношению к входному изображению.
2) Использует знак градиентов для создания нового, вредоносного, изображения, которое максимизирует потери.

$$adv\_x = x + \epsilon * \text{sign}(\nabla_x J(\theta, x, y))$$

adv_x - выходное состязательное изображение
x - входное изображение
y - метка класса true
ε - коэффициент на который умножаются градиенты, чтобы сделать изменения в изображении достаточно малыми, чтобы человеческий глаз не мог их обнаружить, но достаточно большими, чтобы они могли обмануть нейронную сеть
θ - модель нейронной сети
J - функция потерь (loss function)

FGSM (Fast Gradient Sign Method)

Результатом является выходное изображение, которое, для человеческого глаза, выглядит идентично оригиналу, но заставляет нейронную сеть давать неверный прогноз.



Задание

https://docs.google.com/document/d/1-2soxLICXUel-sbNBW8tsW9g_hlpbttlY4gPTpZMUlw/edit?usp=sharing

А что дальше?

Adversarial Robustness Toolbox (ART) — это библиотека Python для безопасности машинного обучения. ART предоставляет инструменты, которые позволяют разработчикам и исследователям защищать и оценивать модели и приложения машинного обучения от враждебных угроз уклонения, отравления, извлечения и логического вывода. ART поддерживает все популярные фреймворки машинного обучения (TensorFlow, Keras, PyTorch, MXNet, scikit-learn, XGBoost, LightGBM, CatBoost, GPy и т. д.), все типы данных (изображения, таблицы, аудио, видео и т. д.) и машинное обучение. задачи (классификация, обнаружение объектов, распознавание речи, генерация, сертификация и др.).

А что дальше?

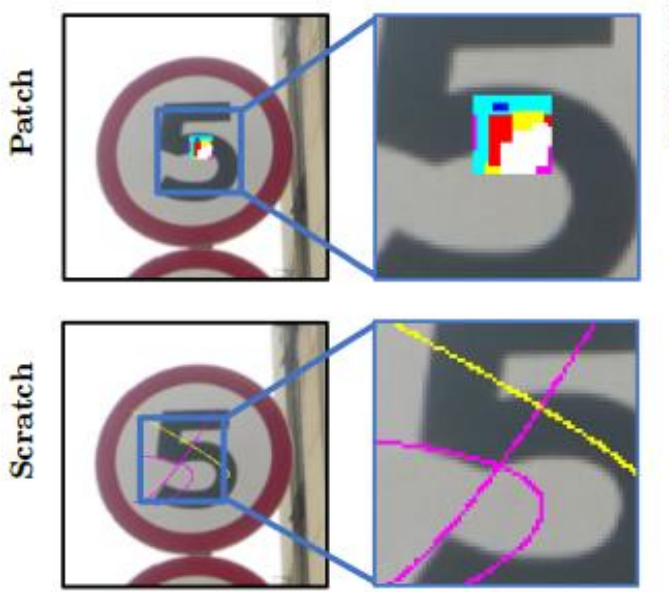
ИТМО

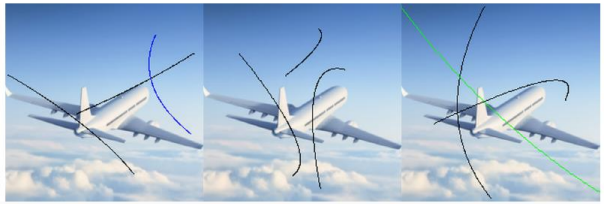
FoolBox – простая и удобная библиотека для применения большого количества атак на системы искусственного интеллекта. Работает со следующими фреймворками и библиотеками: TensorFlow, Keras, Theano, PyTorch, Lasagne, MXNet.

Спасибо
за внимание!

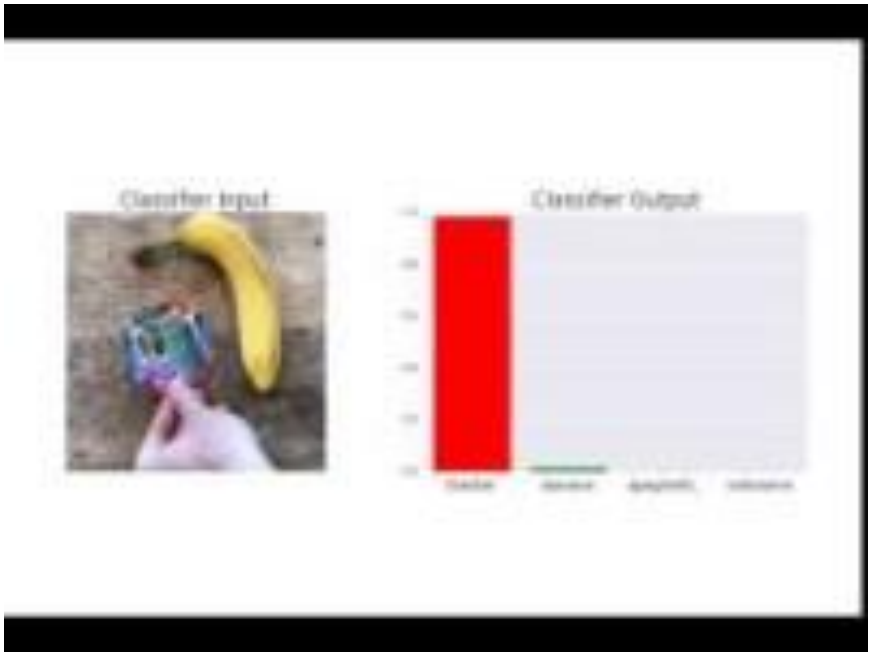
ITMO more than a UNIVERSITY

Атаки патчем и/или штрихом



	Original		
Input image			
Predicted caption	A plane flying in the sky	A green and white rocket	A wind turbine with a blue sky
Confidence	0.512	0.219	0.375
Iterations	0	6	25
L_0	0	341	435
Input image			
Predicted caption	A group of thin thin thin thin thin thin thin thin thin thin thin thin thin thin thin thin thin thin	A close-up of a barbed wire fence	A close-up of a bug
Confidence	0.169	0.493	0.539
Iterations	57	65	217
L_0	481	419	683

Атаки патчем и/или штрихом



Атаки патчем, доля неверных прогнозов

Class name	Patch size 32x32	Patch size 48x48	Patch size 64x64
toaster	72.02%	98.12%	99.93%
goldfish	86.31%	99.07%	99.95%
school bus	91.64%	99.15%	99.89%
lipstick	70.10%	96.86%	99.73%
pineapple	92.23%	99.26%	99.96%