ВНИМАНИЕ!

ДАННЫЙ КУРС СОДЕРЖИТ БОЛЬШОЕ КОЛИЧЕСТВО РАЗНООБРАЗНОГО КОДА И ЗАДАНИЙ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ.

НА ПЕРВЫЙ ВЗГЛЯД ОН МОЖЕТ ПОКАЗАТЬСЯ СЛОЖНЫМ И ТРАВМИРОВАТЬ НЕПОДГОТОВЛЕННУЮ ПСИХИКУ. ТАКЖЕ ОН СОДЕРЖИТ БОЛЬШОЕ КОЛИЧЕСТВО НЕУДАЧНЫХ ШУТОК И НЕУМЕСТНЫХ ОТСЫЛОК.

В СВЯЗИ С ЭТИМ КУРС НЕ РЕКОМЕНДУЕТСЯ ПРОСЛУШИВАТЬ … НИКОМУ.

Глубокое обучение и вообще

Соловей Влад и Шигапова Фирюза

14 октября 2020 г.

Посиделка 1: вводная

Risk
modelling.research

Agenda

- О том каким будет курс + что почитать/посмотреть
- Фреймворки для нейроночек
- Немного истории и важные тренды
- От регрессии к нейросетке
- Нейросетки - конструктор Lego
- Учим свою первую нейросетку

Правила игры и почиташки

Про пары

- что-то неясно $\Rightarrow$ ПЕРЕБЕЙ И СПРОСИ
- на парах смотрим презы, пишем код, решаем задачи, много говорим
- будет математика
- все материалы можно найти на страничке курса

Risk modelling.research

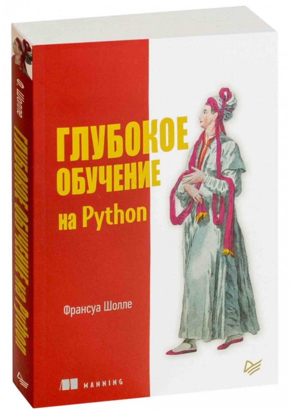
Что почитать про нейронки в первую очередь

Баланс математики и практики,
код устарел (tensorflow 1.14)

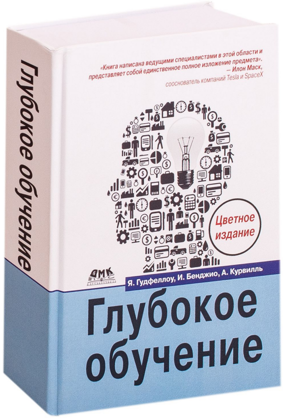
Философия и история ML

Risk
modelling.research

Что почитать про нейронки в первую очередь



Простая практика на Keras



Хардкорная математика,
библия глубокого обучения

Risk
modelling.research

Что посмотреть про нейронки

- Если ничего не знаете про машинное обучение, смотрите вводный курс от Яндекса и МФТИ. Для тех, кто мало знает про ML.

- Advanced ML от Яндекса. Там есть очень разные специфические курсы. Первый из них про нейронки. Код на Tensorflow. Версия библиотеки там пока что старая. Для тех, кто хочет развиваться дальше.

- Нейронные сети от Andrew Ng. Когда ML ещё не был таким модным, все смотрели его лекции. Для тех, кто хочет всё делать медленно и непринуждённо.

- Курс нейронок, который читают в ШАД и Сколтехе. Есть варианты кода на разных фреймворках. Есть видео лекций на русском и английском. Для тех, кто хочет посмотреть как читают курс по DL в ШАД.

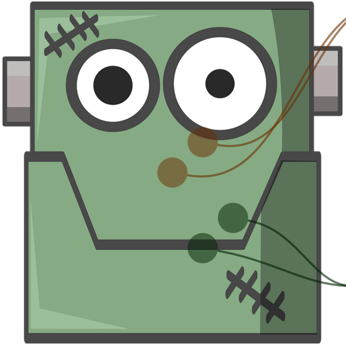
Что посмотреть про нейронки

- Бесплатный курс по tf от Google. Короткий. Покрывает весь базовый Keras. Все тетрадки выложены в colab. Есть странные интервью. Для тех, кто хочет быстро зашарить keras.

- Бесплатный курс по pytorch от Samsung. Вводный курс в нейросетки на pytorch. Для тех, кто хочет писать на pytorch.

- Deep learning на пальцах. Лекции из кремниевой долины. Задания на pytorch для самостоятельного решения. Для тех, кто хочет посмотреть няшные простые стримчики про нейронки.

Структура курса

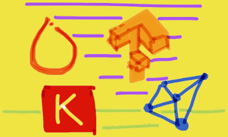


- Базовая часть курса:
- От регрессии к нейросети
- 50 оттенков градиентного спуска, Backpropagation
- Введение в Pytorch
- Эвристики и приёмы для обучения сеток
- Пайплайн обучения сетки

- Разные архитектуры:
- Сетки для табличных данных
- Свёрточные сети, локализация, сегментация, перенос стиля
- Автокодировщики, генеративные модели
- Рекурентные нейронные сетки
- Введение в NLP, эмбединги, автопереводы
- Обучение с подкреплением

Risk modelling.research

Фреймворки для Deep Learning



Фрэймворки

Монреальский университет (2007)

Static Computational Graph

Google (2011, открыта с 2015, с 2019 tf 2.0)

~~Static~~ Dynamic Computational Graph

Facebook (2016)

Dynamic Computational Graph

Обёртки

Сначала обёртка для Theano, потом для Tensorflow, сейчас фактически часть Tensorflow

K Keras

Обёртка для Theano

Лекция про фрэймворки на русском (из далёкого 2017 года): https://www.youtube.com/watch?v=ghZyptkanBo

Подходы к вычислениям

Императивный подход

```
a = np.ones(10)
b = np.ones(10) * 2
c = b * a
d = c + 1
```

Сразу вычислили

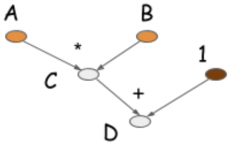
Символьный подход

```
A = Variable('A')
B = Variable('B')
C = B * A
D = C + Constant(1)

# компиляция функции
f = compile(D)

# исполнение
d = f(A=np.ones(10), B=np.ones(10)*2)
```

Сначала задали граф вычислений,
а потом уже считали



Risk
modelling.research

Символьный подход

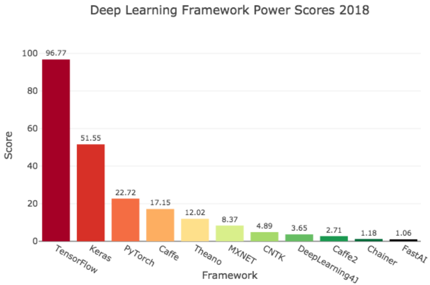
+ Легко строить сеть из вычислений и автоматически искать по ней производные (быстрая и простая оптимизация)

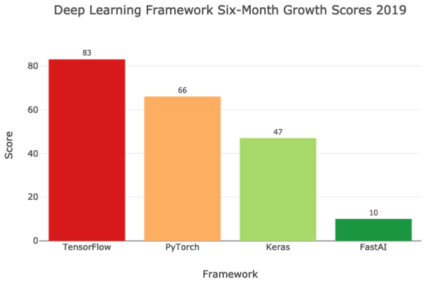
+ Более эффективные вычисления, как по памяти, так и по скорости (на этапе компиляции можно выявить неиспользуемые переменные, найти места для переиспользования и тп)

- Довольно сложно искать ошибки из-за того, что сначала задаётся граф вычислений

Risk
modelling.research

Мощь библиотек





https://towardsdatascience.com/deep-learning-framework-power-scores-2018-23607ddf297a
https://keras.io/why-use-keras/

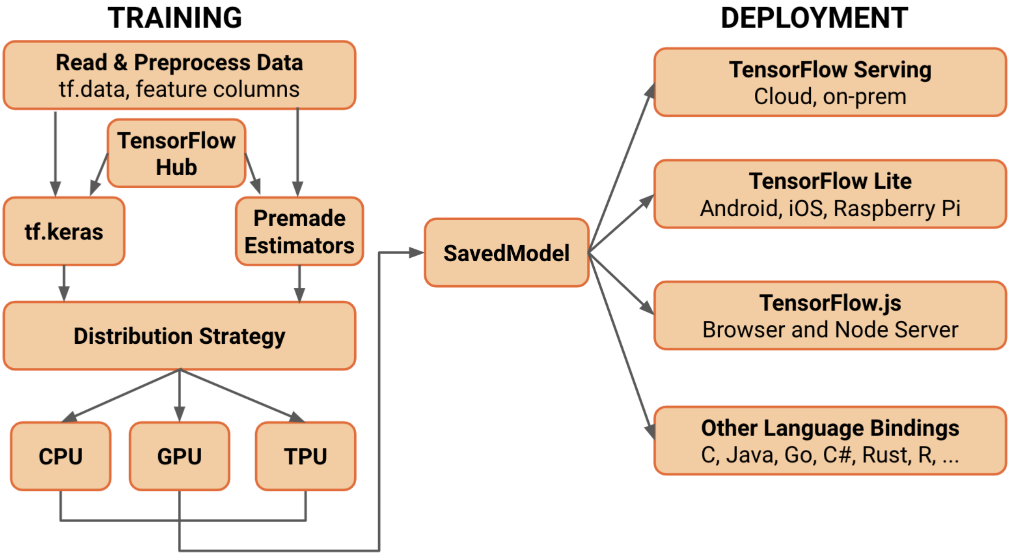
Risk
modelling.research

Tensorflow 2.0

https://www.tensorflow.org/beta/guide/effective_tf2

Risk modelling.research

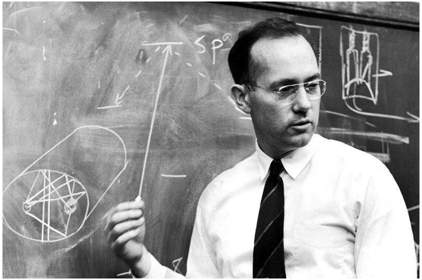
Tensorflow 2.0



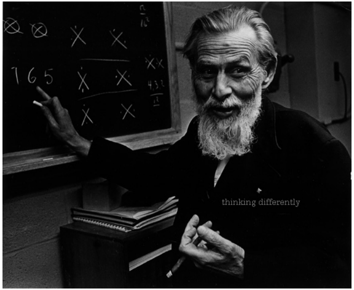
Risk
modelling.research

Немного истории

Первый формальный нейрон (1943)



Уолтер Питтс

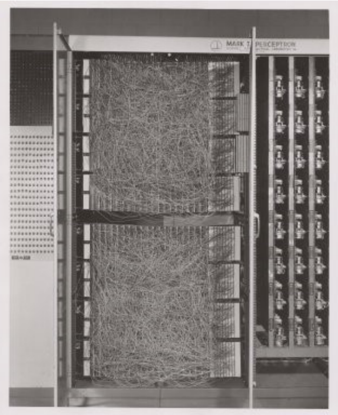


Уоррен Маккалок

Risk modelling.research

Первый формальный нейрон (1958)

Фрэнк Розенблатт

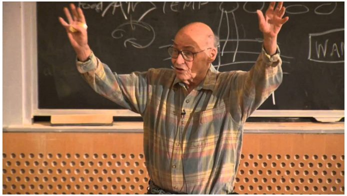


Mark I Perceptron
(Компуктер Розенблата)

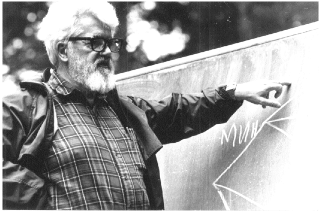
Гарольд
(Мышь Розенблатта)

Дартмундский семинар (1956)

Марвин Минский

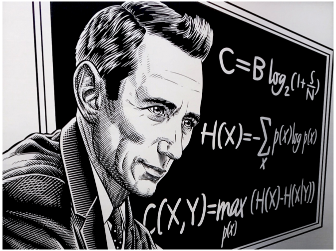


Джон Маккарти



Натаниэль Рочестер

Клод Шеннон



Risk
modelling.research

La Verendrye
Wildlife
Reserve
Québec
Алгонкинский
провинциальный
парк
Algonquin
Provincial
Park
Оттава
Ottawa
Монреаль
Montréal
Шербрук
Sherbrooke
Монктон
Moncton
Фредериктон
Fredericton
Сент-Джон
Saint John
МЭН
ВЕРМОНТ
Колледж Дартмут
Университет
Лиги плюща, открыт...
НЬЮ-ХЭМПШИР
Кингстон
Kingston
Торонто
Toronto
Миссисауга
Mississauga
Рочестер
Rochester
НЬЮ-ЙОРК
Олбани
Albany
Буффало
Buffalo
Бостон
Boston
МАССАЧУСЕТС
Провиденс
Providence
РОД-АЙЛЕНД
Нью-Йорк
New York
Питтсбург
Pittsburgh
ПЕНСИЛЬВАНИЯ
Филадельфия

Зима близко

- 1956 — Дартмунтский семинар, море оптимизма
- 1958 — Персептрон Розенблатта
- середина 1960-х — провал крупного проекта по машинному переводу с русского на английский и наоборот
- 1969 — Марвин Минский и Сеймур Пейперт опубликовали книгу «Персептроны» с критикой

Risk modelling.research

Зима наступила

- Зима искуственного интеллекта — период в истории исследований искусственного интеллекта, связанный с сокращением финансирования и снижением интереса

- Две длительные «зимы» относят к периодам 1974—1980 годов и 1987—1993 годов

- Несмотря на спад финансирования, исследования продолжались

Оттепель

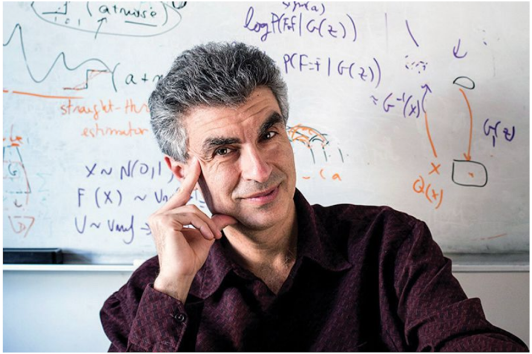
- 1970-е — Расцвет экспертных систем, принимающих решения на основе большого числа правил и знаний о предметной области
- MYCIN накопила около 600 правил для идентификации вирусных бактерий и выдачи подходящего метода лечения (угадывала в 69% случаев, лучше любого начинающего врача)
- 1980-е — появилось много разных архитектур
- 1980-е — алгоритм обратного распространения ошибки (backpropagation) позволил обучать сети за линейное время
- Ренессанс нейронных сетей

Зима близко

- Новая волна оптимизма
- 1986 — один из первых AI-отделов экономил компании DEC около 10 миллионов долларов в год
- Завышенные ожидания снова лопнули
- 1990-е — ударными темпами развивается классическое машинное обучение

Революция (2005-2006)

Джеффри Хинтон
(университет Торонто)

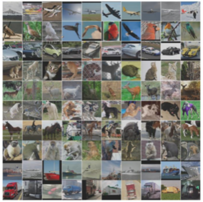


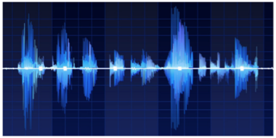
Йошуа Бенджи
(университет Монреаля)

Революция

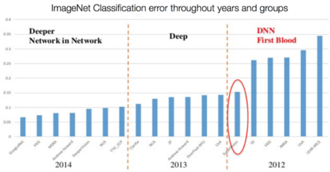
- 2005-2006 — группы Хинтона и Бенджи научились обучать глубокие нейросетки
- Накопилось больше данных! Огромные данные!
- Компьютеры стали на порядки мощнее! Появились крутые GPU!
- На больших данных и мощностях заработали старые архитектуры
- Появились новые алгоритмы, эвристики и подходы
- Ящик Пандоры открыт!

ImageNet





AlphaGo 4:1



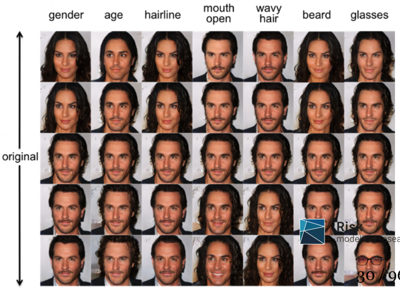
утро

вечер

Пабло Пикассо

Винсент Ван Гог

Василий Кандинский



Новая зима близко — ???

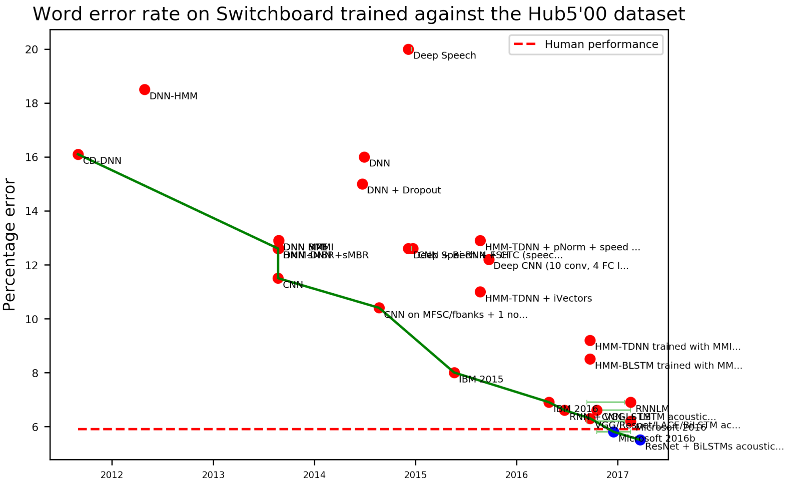
Важные тренды

Risk
modelling.research

1. Точность сетей растёт

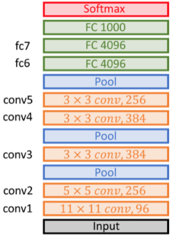
https://www.eff.org/ai/metrics

1. Точность сетей растёт



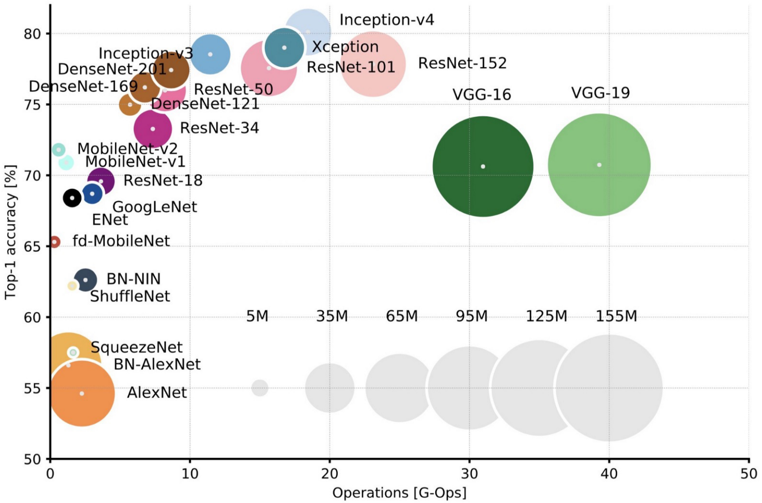
https://www.eff.org/ai/metrics

2. Сложность сетей растёт



AlexNet

2. Сложность сетей растёт



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

Risk modelling.research

3. Объёмы данных растут

4,267,149,297
Internet Users in the world

1,696,839,824
Total number of Websites

186,645,375,538
Emails sent today

29.06.2019

4,816,061,902
Google searches today

4,579,120
Blog posts written today

545,680,148
Tweets sent today

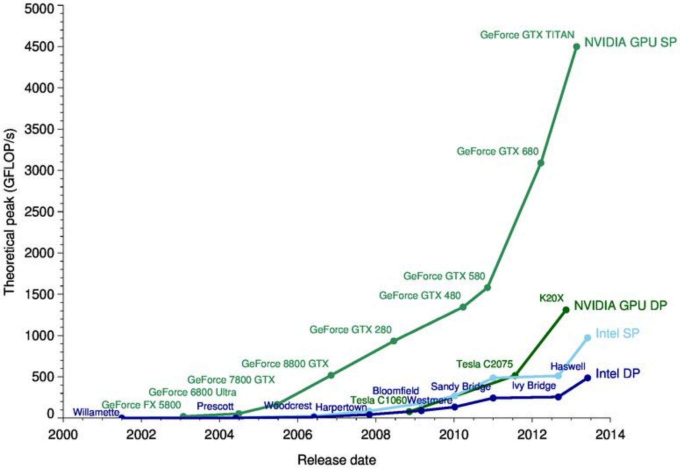
5,067,684,629
Videos viewed today on YouTube

58,968,701
Photos uploaded today on Instagram

98,899,252
Tumblr posts today

https://www.internetlivestats.com

4. Вычислительные мощности растут



Risk
modelling.research

4. Почему это возможно?

Assassin's Creed Unity

Grand Theft Auto Five

The Witcher Wild Hunt

Project Cars

Final Fantasy XV

ファイナルファンタジーXV

Call of Duty Advanced Warfare

Quantum Break

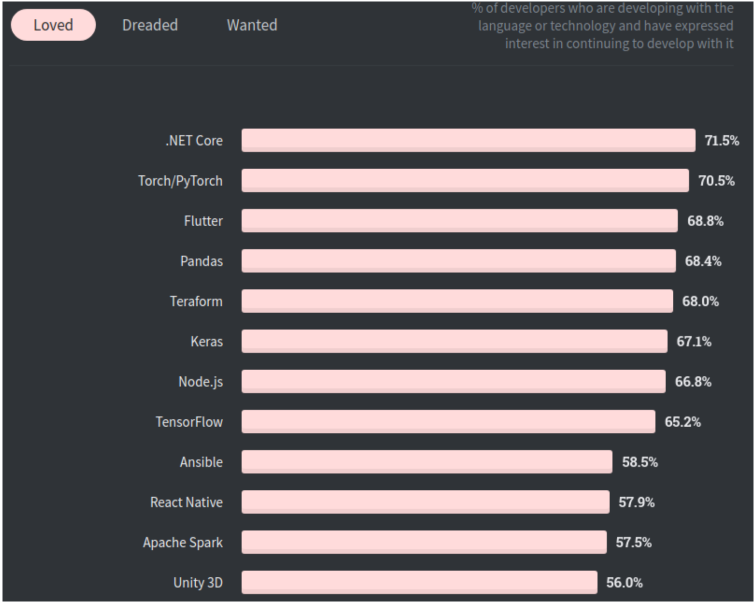
https://tproger.ru/articles/cpu-and-gpu/

Опросы программистов

Other Frameworks, Libraries, and Tools

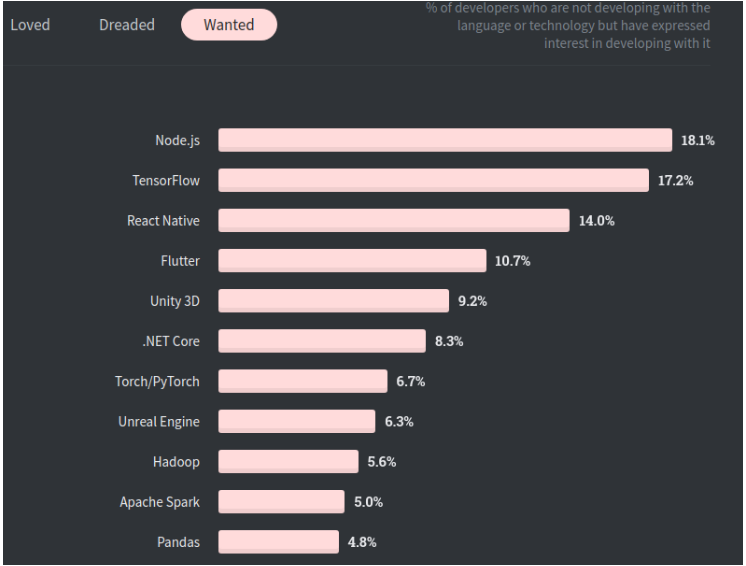
Similar to last year, we asked about many of the other miscellaneous technologies that developers are using. For the second year in a row, Node.js takes the top spot, as it is used by half of the respondents. We also see growth across the board in the popularity of data analysis and machine learning technologies such as Pandas, TensorFlow, and Torch/PyTorch.

Любимые технологии



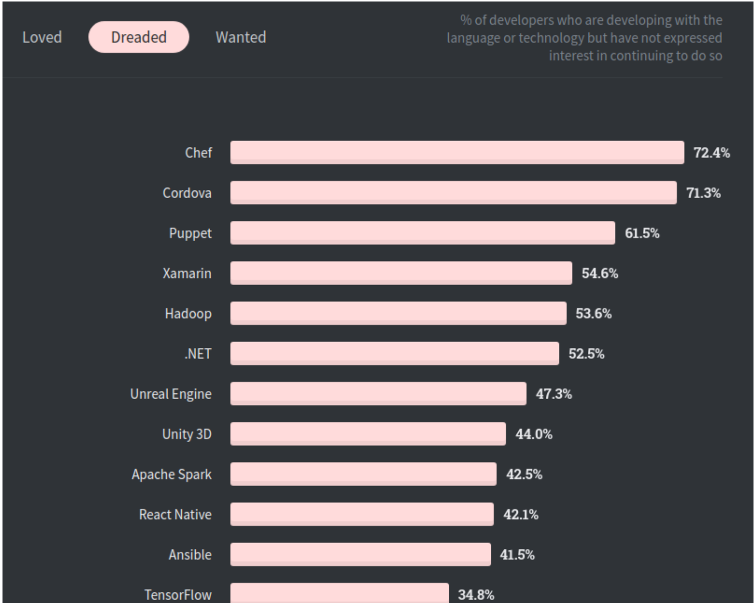
Risk
modelling.research

Желаемые технологии



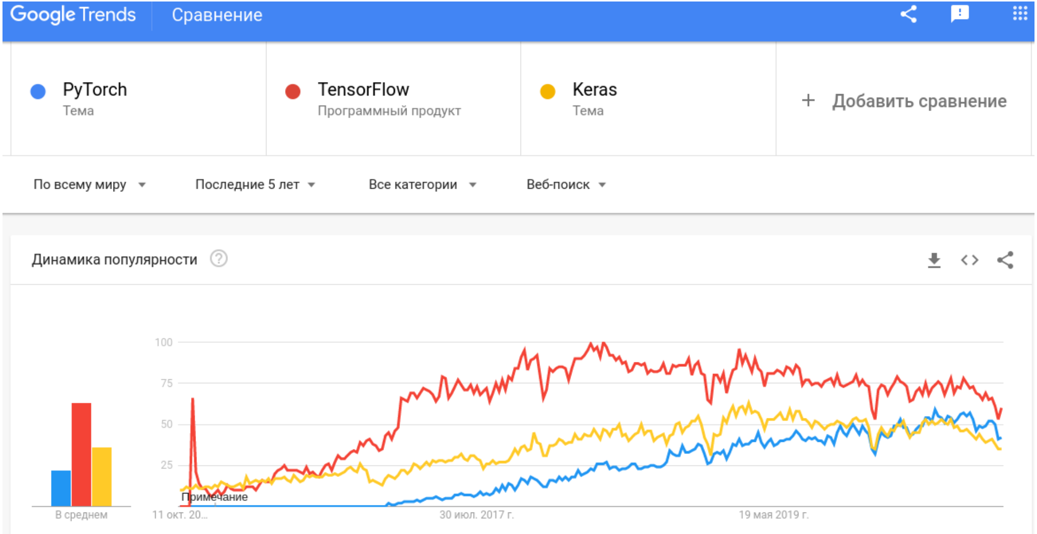
Risk
modelling.research

Пугающие технологии



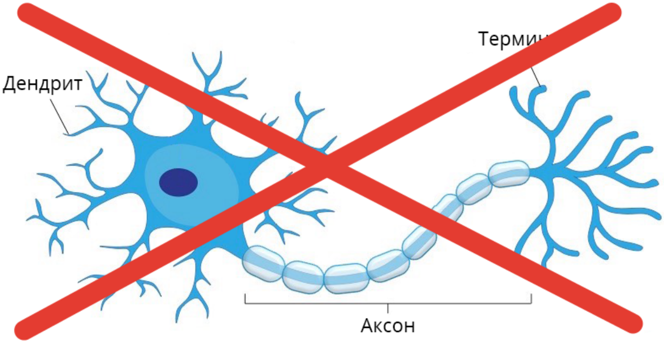
Risk modelling.research

Google Trends



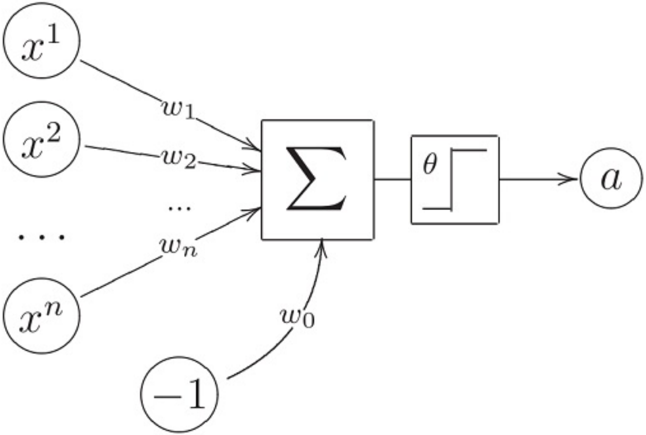
Персептрон

Нейрон



Хватит нам врать!

Формальная модель нейрона



Модель нейрона МакКаллока-Питтса.

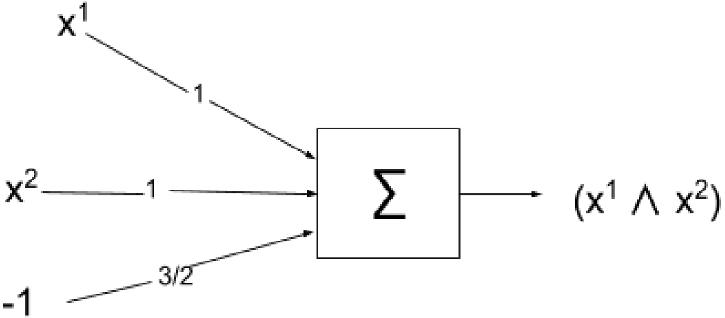
Нейрона вычисляет n-булеву функцию

$$a(x, w) = \phi(\sum_{j=1}^{n} w_j x_j - w_0)$$

Risk modelling.research

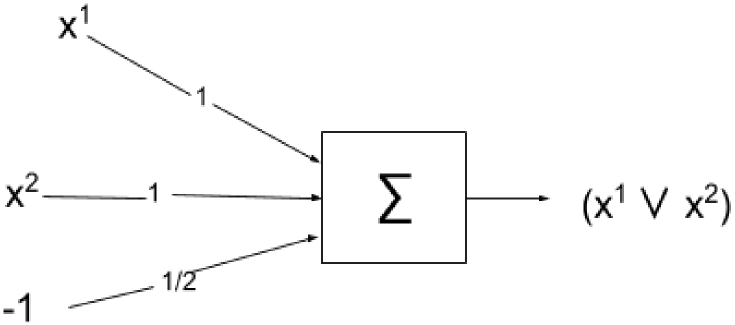
Нейронная реализация логических функций. И

$$x^1 \wedge x^2 = [x^1 + x^2 - \frac{3}{2} > 0]$$



Нейронная реализация логических функций. ИЛИ

$$x^1 \vee x^2 = [x^1 + x^2 - \frac{1}{2} > 0]$$



Risk
modelling.research

Как можно записать НЕ ?

Нейронная реализация логических функций. НЕ

$$\neg x^1 = [-x^1 + \frac{1}{2} > 0]$$

Risk
modelling.research

Нейронная реализация логических функций. XOR

$$x^1 \oplus x^2 = [x^1 \neq x^2]$$

Функция не реализуется одним нейроном

Risk
modelling.research

Нейронная реализация логических функций. XOR

Есть два способа:

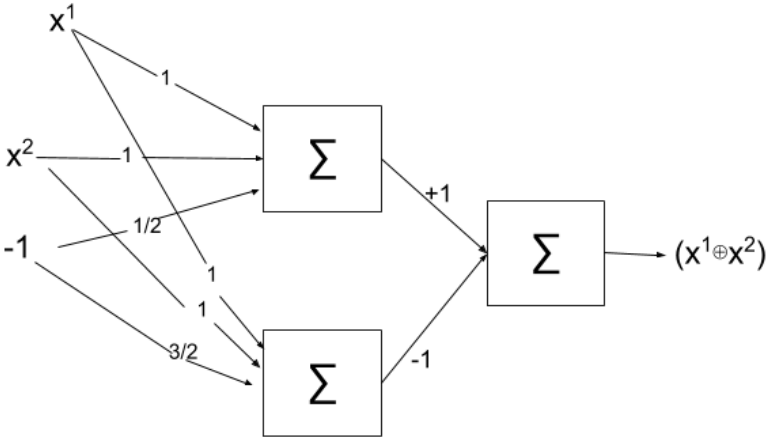
1. Добавления нелинейного признака

$$x^1 \oplus x^2 = [x^1 + x^2 - 2x^1x^2 - \frac{1}{2} > 0]$$

2. Двухслойной сетью (двухслойная суперпозиция функций)

$$x^1 \oplus x^2 = [(x^1 \vee x^2) - (x^1 \wedge x^2) - \frac{1}{2} > 0]$$

Нейронная реализация логических функций. XOR



Мышь Розенблата

- Термин нейронная сеть пришёл из биологии, его придумали 70 лет назад.

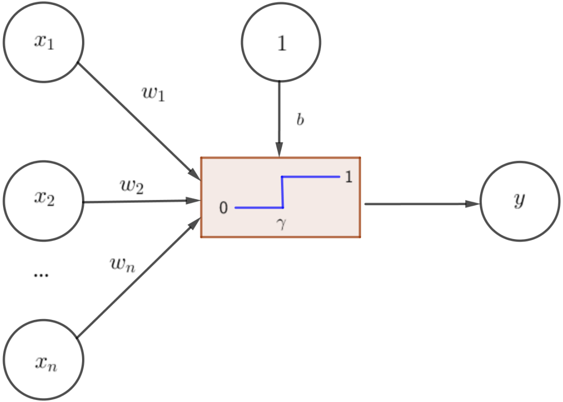
- Оказалось, что мозг устроен гораздо сложнее.

- Продуктивнее думать про нейросетки, как про вычислительные графы.

Не забыли его?

https://habr.com/ru/company/mailru/blog/405615/

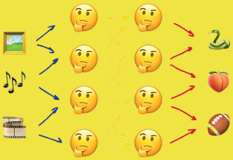
Персептрон Розенблатта (1950)



$$y = \begin{cases} 1, \text{ если } \sum w_i x_i \geq \gamma \\ 0, \text{ если } \sum w_i x_i < \gamma \end{cases}$$

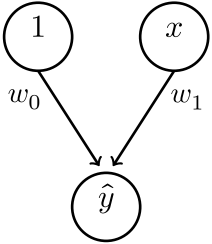
Risk modelling.research

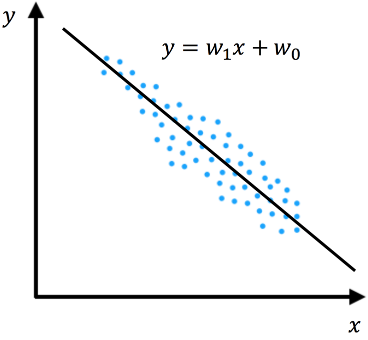
От регрессии к нейросетке



Risk modelling.research

Линейная регрессия



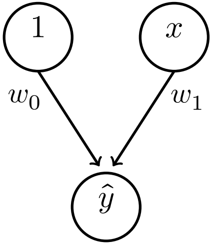


$$y_i = w_0 + w_1 \cdot x_i$$

$$y_i = \begin{pmatrix} 1 & x_i \end{pmatrix} \cdot \begin{pmatrix} w_0 \\ w_1 \end{pmatrix}$$

$$y_i = (x_i, w)$$

Линейная регрессия



$$
\begin{aligned}
y_1 &= w_0 + w_1 \cdot x_1 \\
y_2 &= w_0 + w_1 \cdot x_2 \\
y_3 &= w_0 + w_1 \cdot x_3
\end{aligned}
$$

$$
\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \end{pmatrix} \cdot \begin{pmatrix} w_0 \\ w_1 \end{pmatrix}
$$

$$y_i = w_0 + w_1 \cdot x_i$$

$$y_i = \begin{pmatrix} 1 & x_i \end{pmatrix} \cdot \begin{pmatrix} w_0 \\ w_1 \end{pmatrix}$$

$$y_i = (x_i, w)$$

Линейная регрессия (векторная форма)

$$y = \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix} \qquad X = \begin{pmatrix} 1 & x_{11} & \dots & x_{1k} \\ 1 & x_{21} & \dots & x_{2k} \\ \dots & \dots & \dots & \dots \\ 1 & x_{n1} & \dots & x_{nk} \end{pmatrix} \qquad w = \begin{pmatrix} w_0 \\ w_1 \\ \dots \\ w_k \end{pmatrix}$$

Модель:

$$y = Xw$$

Оценка:

$$\hat{w} = (X^T X)^{-1} X^T y$$

Прогноз:

$$\hat{y} = X\hat{w}$$

Risk modelling.research

Как обучить линейную регрессию?

- Нужно ввести штраф за ошибку:

$$MSE(w) = \frac{1}{n}\sum_{i=1}^{n}(w^T x_i - y_i)^2 \to \min_w$$

- Не для всех функция потерь бывает аналитическое решение, например для MAE из-за модуля его нет:

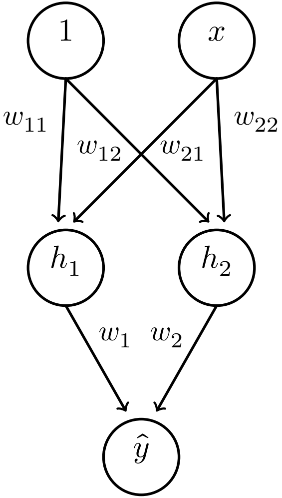
$$MAE(w) = \frac{1}{n}\sum_{i=1}^{n}|w^T x_i - y_i|$$

- Обычно модель обучают методом градиентного спуска

Про метрики для регрессии:
https://alexanderdyakonov.files.wordpress.com/2018/10/book_08_metrics_12_blog1.pdf

Risk
modelling.research

А что если...



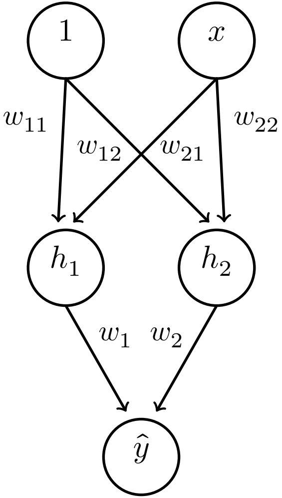
$$h_{1i} = w_{11} + w_{21} \cdot x_i$$
$$h_{2i} = w_{12} + w_{22} \cdot x_i$$

$$y_i = w_1 \cdot h_{1i} + w_2 \cdot h_{2i}$$

$$h = X \cdot W_1$$
$$y = h \cdot W_2$$

Норм идея?

A что если...



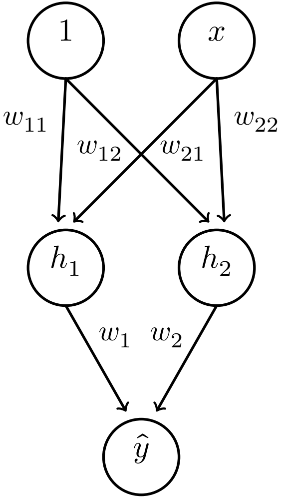
$$
\begin{aligned}
y = w_1 \cdot h_1 + w_2 \cdot h_2 = \\
= w_1 \cdot (w_{11} + w_{21} \cdot x) + w_2 \cdot (w_{12} + w_{22} \cdot x) = \\
= \underbrace{(w_1 w_{11} + w_2 w_{12})}_{\gamma_1} + \underbrace{(w_1 w_{21} + w_2 w_{22})}_{\gamma_2} x
\end{aligned}
$$

Чёрт возьми! Опять линейность...

$$y = h \cdot W_2 = X \cdot W_1 \cdot W_2 = X \cdot A$$

Risk modelling.research

А что если...



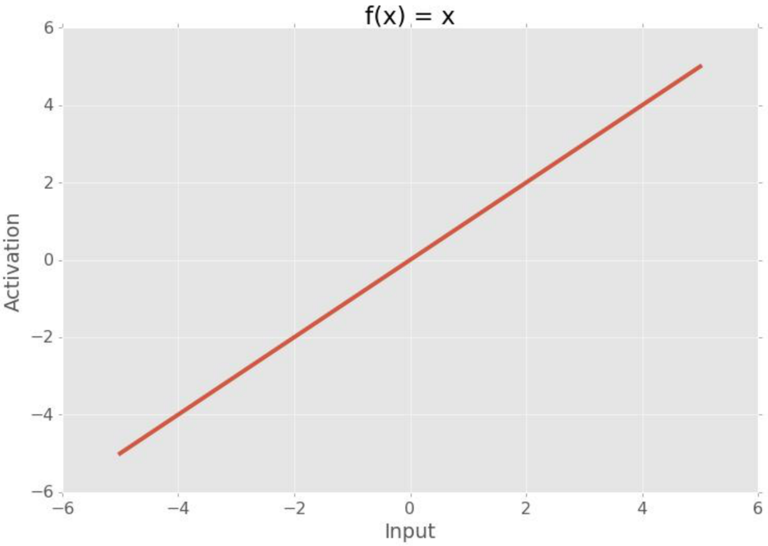
- Давайте добавим к скрытому состоянию какую-нибудь нелинейность:

$$h_{1i} = w_{11} + w_{21} \cdot x_i$$
$$h_{2i} = w_{12} + w_{22} \cdot x_i$$
$$y_i = w_1 \cdot f(h_{1i}) + w_2 \cdot f(h_{2i})$$

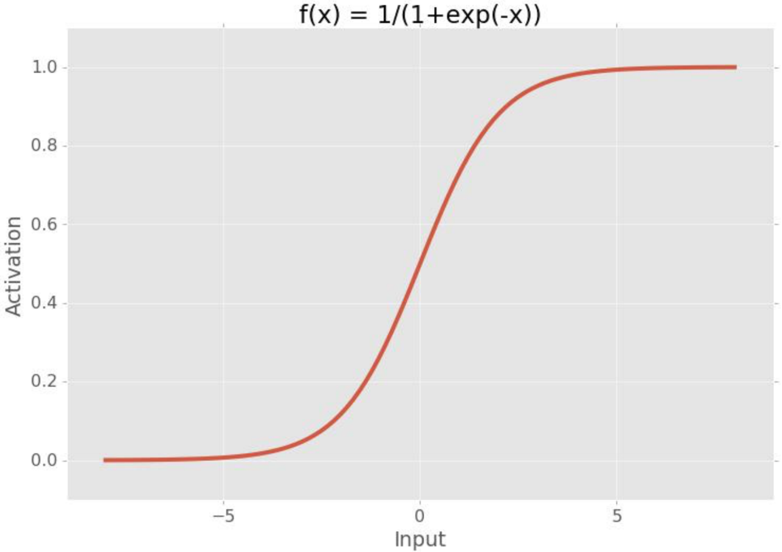
$$y = h \cdot W_2 = f(X \cdot W_1) \cdot W_2 \neq X \cdot A$$

Risk
modelling.research

Почему нельзя взять такую функцию?

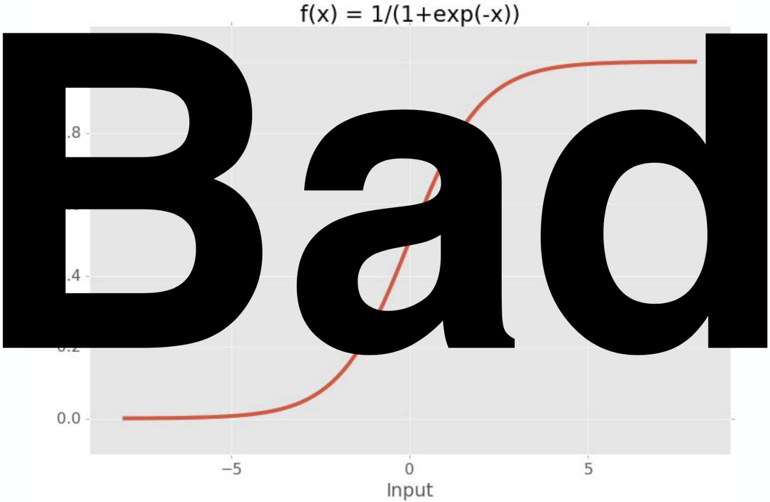


Сигмоида



Risk
modelling.research

Сигмоида

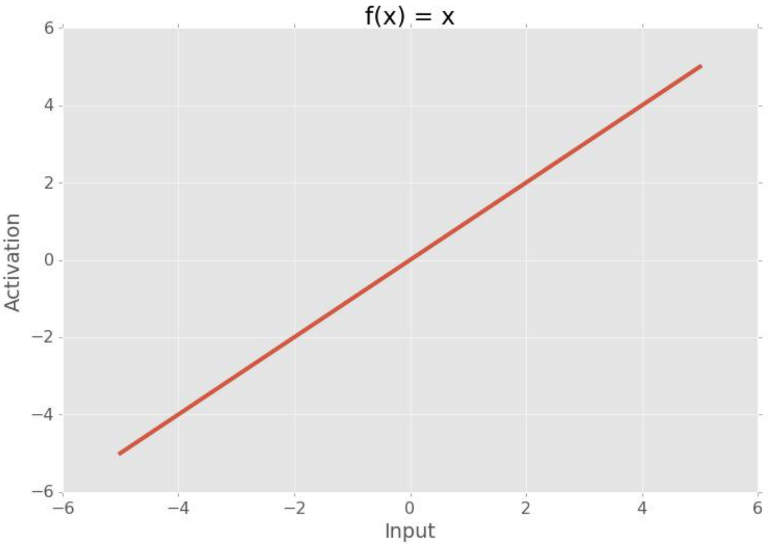


Risk
modelling.research

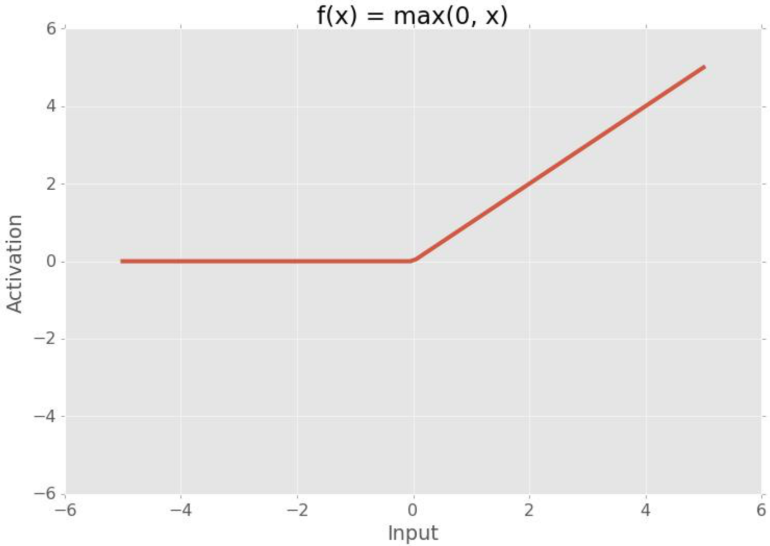
Сигмоида

- В маленьких сетках сигмоиду можно смело использовать
- В глубоких сетях из-за сигмоиды возникает паралич сети
- Про него подробнее мы поговорим чуть позже
- Нужна другая нелинейная функция активации

От линейной активации ...

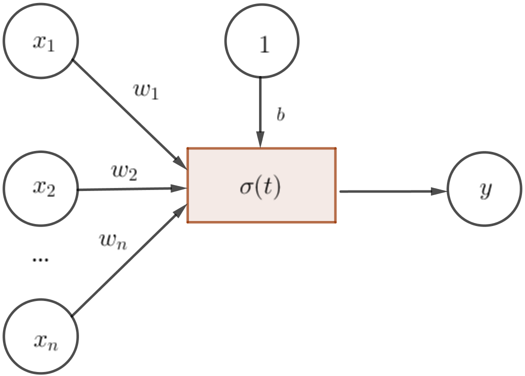


… к нелинейной



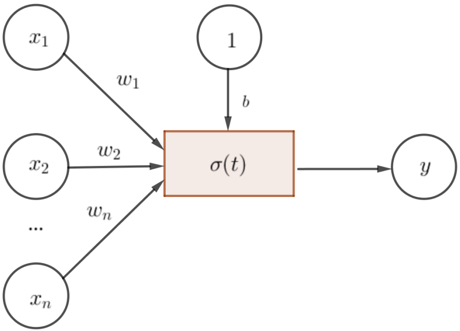
Risk modelling.research

Функция активации



- Функция активации $\sigma(t)$ вносит нелинейность, она может быть любой

Линейная регрессия



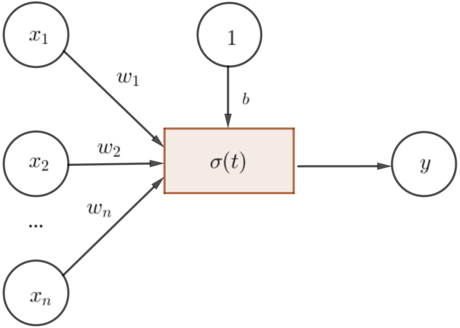
Нейрон с линейной функции активации — это линейная регрессия...

$$\sigma(t) = t$$

$$y = w_0 + w_1 \cdot x_1 + \ldots + w_n \cdot x_n$$

Risk modelling.research

Логистическая регрессия

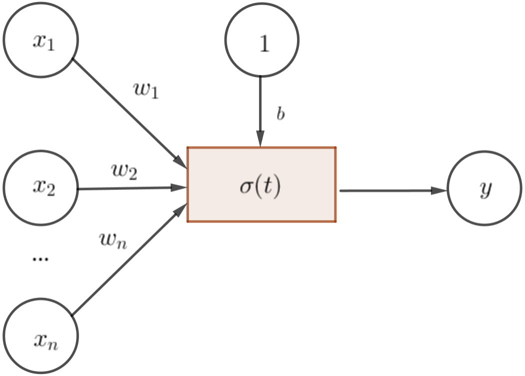


Нейрон с сигмоидом в качестве функции активации — это логистическая регрессия...

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

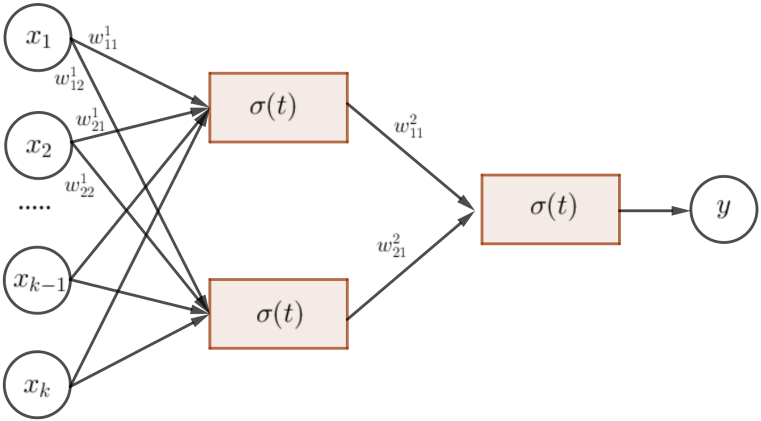
$$P(y = 1 \mid x) = \sigma(w_0 + w_1 \cdot x_1 + \ldots + w_n \cdot x_n)$$

Функция активации



$$y = \sigma(X \cdot W)$$

Две регрессии скрепили третьей

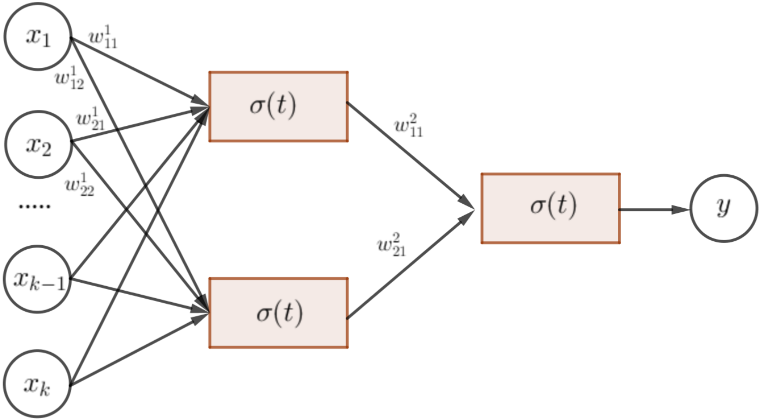


$$h_j = \sigma(w_0 + w^1_{j1} \cdot x_1 + \ldots + w^1_{jk} \cdot x_k)$$

$$y = \sigma(w^2_{11} \cdot h_1 + w^2_{21} \cdot h_2)$$

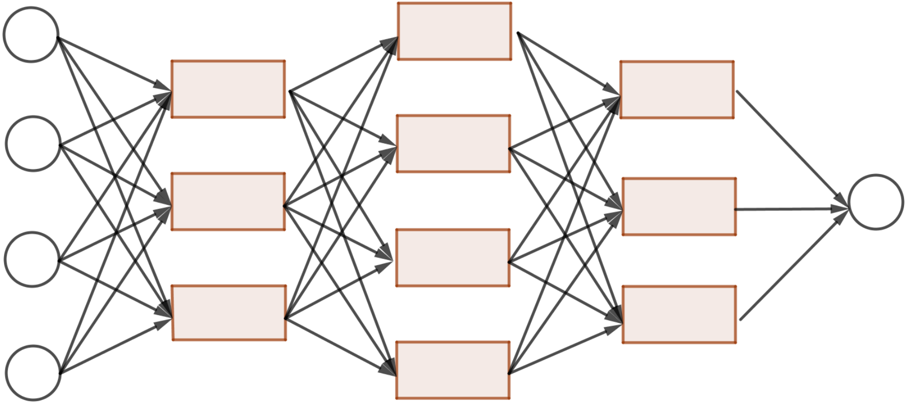
Risk
modelling.research

MLP (multi-layer perceptron)



$$h = \sigma(X \cdot W)$$
$$y = \sigma(h \cdot W)$$

Армия из регрессий



Risk
modelling.research

The Perceptron Convergence Theorem (Rosenblat, 1965)

- Любая непрерывная и ограниченная функция может быть сколь угодно точно аппроксимирована нейронной сетью с одним скрытым слоем с нелинейной функцией активации нейрона.

- Любая функция может быть сколь угодно точно аппроксимирована нейронной сетью с двумя скрытыми слоями с нелинейной функцией активации нейрона.

- Что ещё можно пожелать?

Графическое доказательство теоремы:
http://neuralnetworksanddeeplearning.com/chap4.html

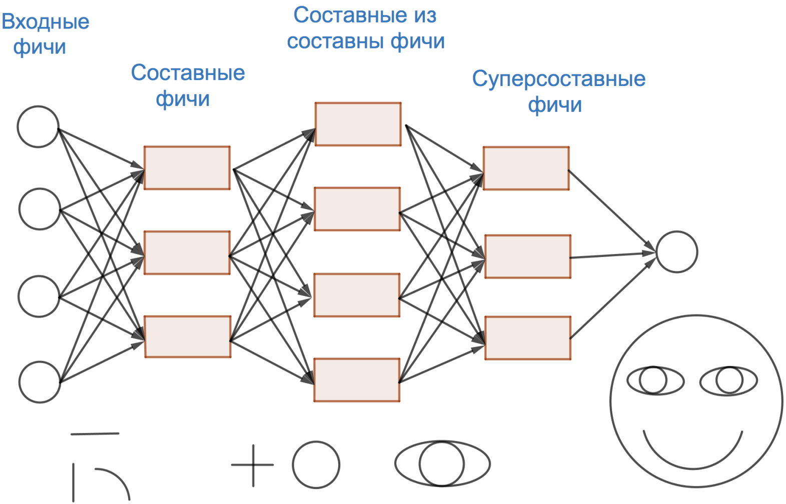
Risk
modelling.research

Going Deeper

Мотивация

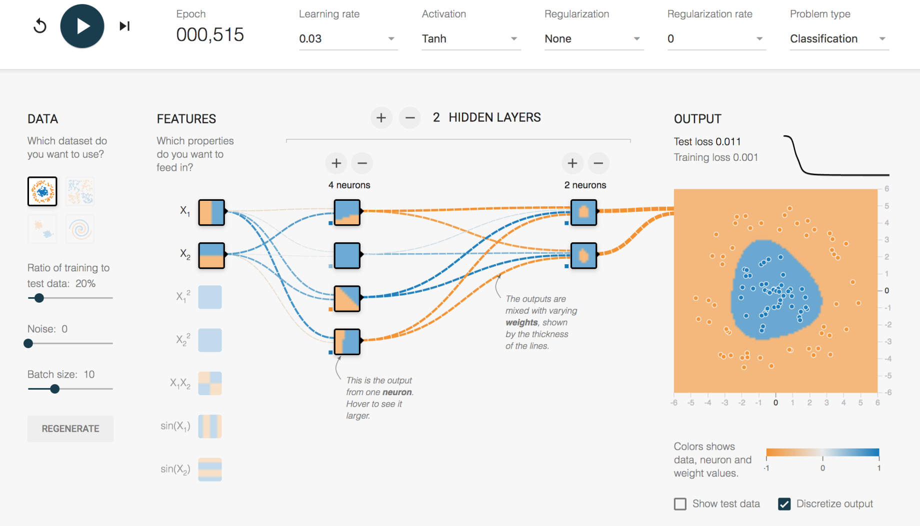
- Персептрон может решить любую проблему, но это дорого
- Глубокие архитектуры часто позволяют выразить то же самое, приблизить те же функции гораздо более эффективно, чем неглубокие
- Каждый новый слой сетки будет работать всё с более сложными фичами

Армия из регрессий



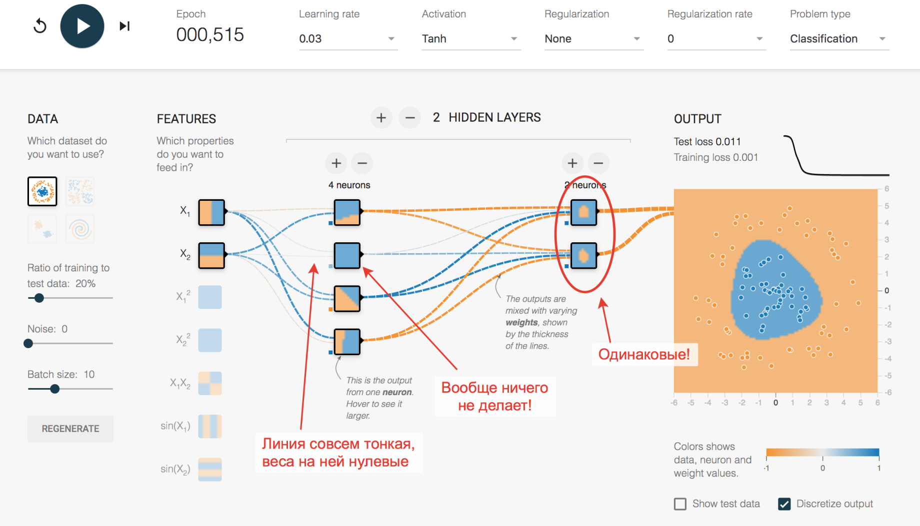
Risk
modelling.research

MLP не отходя от браузера



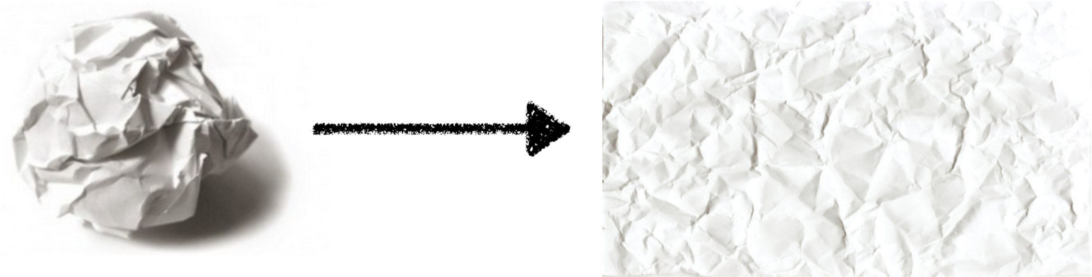
http://playground.tensorflow.org

MLP не отходя от браузера



http://playground.tensorflow.org

Ещё одна аналогия



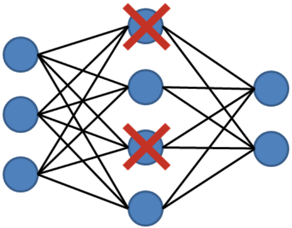
Нейросети — конструктор LEGO

Risk modelling.research

Слои бывают разными

- Слой, который просто взвешивает входы называется полносвязным.

- Слои бывают очень разными. Например, Dropout: с вероятностью p отключаем нейрон. Такой слой препятствует переобучению и делает нейроны более устойчивыми к случайным возмущениям.

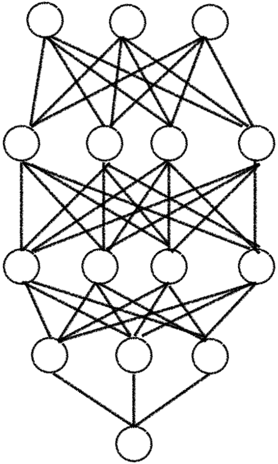


Функции активации бывают разными

Название функции	**Формула** $f(x)$	**Производная** $f'(x)$
Логистический сигмоид σ	$\frac{1}{1+e^{-x}}$	$f(x)\,(1-f(x))$
Гиперболический тангенс tanh	$\frac{e^x-e^{-x}}{e^x+e^{-x}}$	$1-f^2(x)$
SoftSign	$\frac{x}{1+\lvert x\rvert}$	$\frac{1}{(1+\lvert x\rvert)^2}$
Ступенька (функция Хевисайда)	$\begin{cases} 0, & x<0 \\ 1, & x\ge 0 \end{cases}$	0
SoftPlus	$\log(1+e^x)$	$\frac{1}{1+e^{-x}}$
ReLU	$\begin{cases} 0, & x<0 \\ x, & x\ge 0 \end{cases}$	$\begin{cases} 0, & x<0 \\ 1, & x\ge 0 \end{cases}$
Leaky ReLU, Parameterized ReLU	$\begin{cases} ax, & x<0 \\ x, & x\ge 0 \end{cases}$	$\begin{cases} a, & x<0 \\ 1, & x\ge 0 \end{cases}$
ELU	$\begin{cases} \alpha\,(e^x-1), & x<0 \\ x, & x\ge 0 \end{cases}$	$\begin{cases} f(x)+\alpha, & x<0 \\ 1, & x\ge 0 \end{cases}$

Risk modelling.research

Архитектуры бывают разными



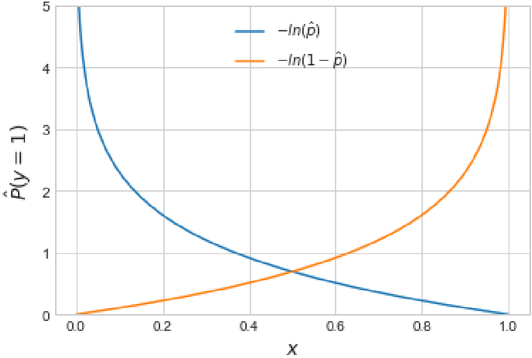
Input	
Fully connected layer (FC)	$XW + b$
ReLU	$max(0, x)$
Dropout	$Bern(p)$
FC	$XW + b$
ReLU	$max(0, x)$
FC	$XW + b$
Output	

Каждый слой — просто функция, каждая сетка — конструктор LEGO

Функция потерь

- Наши y принимают значения 0 и 1
- Если $y = 1$, хотим большое $\hat{p} = \hat{P}(y = 1)$, но чем ближе $\hat{p}$ к 1, тем меньше хотим его увеличить
- Если $y = 0$, хотим большое $(1 - \hat{p})$, получается функция потерь:

$$\text{logloss} = -\frac{1}{n}\sum_{i=1}^{n} y_i \cdot \ln \hat{p} + (1 - y_i) \cdot \ln(1 - \hat{p})$$



Risk modelling.research

Функция потерь

$$
\begin{aligned}
L(w) &= P(y_1, \ldots, y_n \mid X, w) = \\
&= P(y_1 \mid X, w) \cdot \ldots \cdot P(y_n \mid X, w) = \\
&= p^{\sum y_i} \cdot (1-p)^{n-\sum y_i} \to \max_w
\end{aligned}
$$

Risk
modelling.research

Функция потерь

$$
\begin{aligned}
L(w) &= P(y_1, \dots, y_n \mid X, w) = \\
&= P(y_1 \mid X, w) \cdot \dots \cdot P(y_n \mid X, w) = \\
&= p^{\sum y_i} \cdot (1-p)^{n-\sum y_i} \to \max_w
\end{aligned}
$$

$$
\begin{aligned}
\ln L(w) &= \sum y_i \cdot \ln p + \sum (1-y_i) \cdot \ln(1-p) = \\
&= \sum [y_i \cdot \ln p + (1-y_i) \cdot \ln(1-p)] \to \max_w
\end{aligned}
$$

Функция потерь

$$
\begin{aligned}
L(w) &= P(y_1, \dots, y_n \mid X, w) = \\
&= P(y_1 \mid X, w) \cdot \ldots \cdot P(y_n \mid X, w) = \\
&= p^{\sum y_i} \cdot (1-p)^{n-\sum y_i} \to \max_w
\end{aligned}
$$

$$
\begin{aligned}
\ln L(w) &= \sum y_i \cdot \ln p + \sum (1-y_i) \cdot \ln(1-p) = \\
&= \sum [y_i \cdot \ln p + (1-y_i) \cdot \ln(1-p)] \to \max_w
\end{aligned}
$$

$$
\operatorname{logloss}(w) = -\ln L(w) = -\sum [y_i \cdot \ln p + (1-y_i) \cdot \ln(1-p)] \to \min_w
$$

Функция потерь

$$
\begin{aligned}
L(w) &= P(y_1, \dots, y_n \mid X, w) = \\
&= P(y_1 \mid X, w) \cdot \ldots \cdot P(y_n \mid X, w) = \\
&= p^{\sum y_i} \cdot (1-p)^{n - \sum y_i} \to \max_w
\end{aligned}
$$

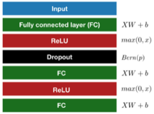
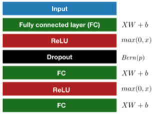
$$
\begin{aligned}
\ln L(w) &= \sum y_i \cdot \ln p + \sum (1 - y_i) \cdot \ln(1-p) = \\
&= \sum [y_i \cdot \ln p + (1 - y_i) \cdot \ln(1-p)] \to \max_w
\end{aligned}
$$

$$
\text{logloss}(w) = -\ln L(w) = -\sum [y_i \cdot \ln p + (1 - y_i) \cdot \ln(1-p)] \to \min_w
$$

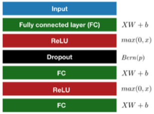
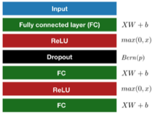
$$
p = P(y_i = 1 \mid X, w) = \sigma(x_i^T w) = \frac{1}{1 + e^{-x_i^T w}}
$$

Risk
modelling.research

Пример: два класса

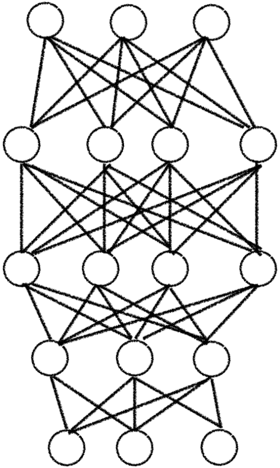
		t	$\sigma(t)$	logloss
 $\Rightarrow$	 $\Rightarrow$	$10 \Rightarrow$	$0.99 \Rightarrow$	$-(1 \cdot \ln 0.99 + (1-1) \cdot \ln(1-0.99))$
 $\Rightarrow$	 $\Rightarrow$	$-1 \Rightarrow$	$0.26 \Rightarrow$	$-(0 \cdot \ln 0.26 + (1-0) \cdot \ln(1-0.26))$

Пример: два класса

			t		$\sigma(t)$		logloss
	$\Rightarrow$	 $\Rightarrow$	10	$\Rightarrow$	$[0.99, 0.01]$	$\Rightarrow$	$-\ln 0.99$
	$\Rightarrow$	 $\Rightarrow$	-1	$\Rightarrow$	$[0.26, 0.74]$	$\Rightarrow$	$-\ln 0.74$

Risk modelling.research

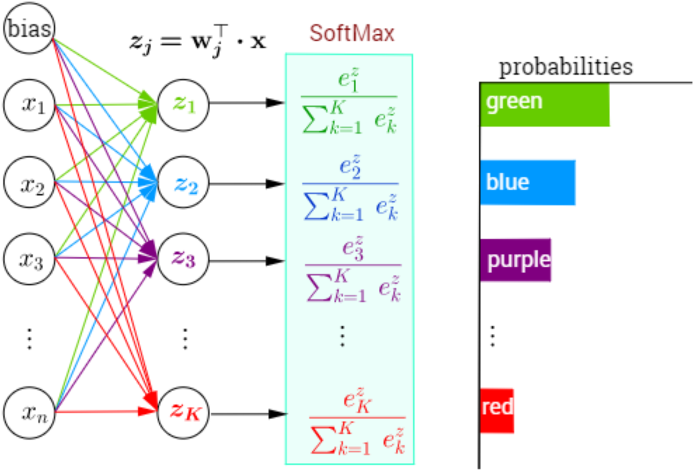
Мультиклассификация



Input	
Fully connected layer (FC)	$XW + b$
ReLU	$max(0, x)$
Dropout	$Bern(p)$
FC	$XW + b$
ReLU	$max(0, x)$
FC	$XW + b$
Softmax	$softmax(x)$
Output	

Мультиклассификация

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_K \end{bmatrix} = \begin{bmatrix} \mathbf{w}_1^\top \\ \mathbf{w}_2^\top \\ \mathbf{w}_3^\top \\ \vdots \\ \mathbf{w}_K^\top \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}$$



Softmax (мягкий максимум)

Много классов, $y \in 1, \dots, K$

$$(w_1^T x, \dots, w_K^T x)$$

$$\Downarrow$$

$$\left(e^{z_1}, \dots, e^{z_K}\right)$$

$$\Downarrow$$

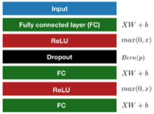
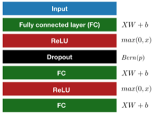
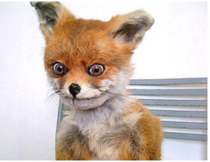
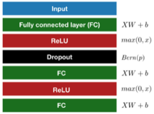
$$\left(\frac{e^{z_1}}{\sum_{k=1}^{K} e^{z_k}}, \dots, \frac{e^{z_K}}{\sum_{k=1}^{K} e^{z_k}}\right)$$

Потери (кросс-энтропия):

$$\text{logloss} = -\sum_{i=1}^{n} \sum_{k=1}^{K} [y_i = k] \cdot \ln \frac{e^{w_k^T x_i}}{\sum_{j=1}^{K} e^{w_j^T x_i}}$$

Risk
modelling.research

Пример: три класса

		t		Softmax		logloss
	$\Rightarrow$  $\Rightarrow$	$[5, 4, 2]$	$\Rightarrow$	$[0.71, 0.26, 0.04]$	$\Rightarrow$	$-\ln 0.71$
	$\Rightarrow$ Input Fully connected layer (FC) $XW+b$ ReLU $max(0,x)$ Dropout $Bern(p)$ FC $XW+b$ ReLU $max(0,x)$ FC $XW+b$ $\Rightarrow$	$[4, 2, 8]$	$\Rightarrow$	$[0.02, 0.00, 0.98]$	$\Rightarrow$	$-\ln 0.98$
	$\Rightarrow$  $\Rightarrow$	$[4, 4, 1]$	$\Rightarrow$	$[0.49, 0.49, 0.02]$	$\Rightarrow$	$-\ln 0.49$

Risk modelling.research

Нейросети - конструктор LEGO

FC

ReLU

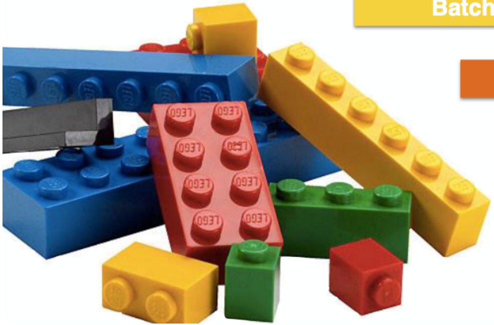
Dropout

Batch Norm

MaxPool

Convolution

1x1 Convolution



Учим свою первую нейросеть в Pytorch!