



Lesson 1 12.09.2022

Что можно сделать:

- классифицировать объекты
- распознавать объект - через сравнение
- бинарные признаки
- перспектива
- аддитивная информация
- генерации и синтезации - по объектам подобия, генерации
- генерации аномалий
- генерации

Все отработанное нужно применить практике

Lesson 2 26.09.2022

torch Tensor → tensor.to("cuda.0")

анализ np.array tensor.backward - для градиента

Нормализация

Пересечение

• MAE $|a-y|$

• MSE $(a-y)^2$

• RMSE $\sqrt{\text{MSE}}$

$$\text{Хулип} \quad \begin{cases} \frac{1}{2}(y-a)^2 & |y-a| < \delta \\ \delta(|y-a| - \frac{1}{2}\delta), & |y-a| \geq \delta \end{cases}$$

$y_i=1$ $y_i=-1$

Классификация

• same классификации всех остальных

• same проблемах - accuracy

• recall precision $\frac{TP}{TP+FP}$

• non-zero recall

$$\sum x_i + y_i \quad a(x_i) = 1 \quad \text{TP} \quad \text{FP}$$

$$\sum x_i = y_i \quad a(x_i) = -1 \quad \text{FN} \quad \text{TN}$$

• Точность + наименование F-measure

2. precision · recall

precision + recall

F-мера - комбинация точности и вспомогательного показателя β для оценки качества предсказания

Рассмотрение

PR кривая - точность / наименование

ROC кривая - TP и FP

Возможные MST классификации

Лекция 3

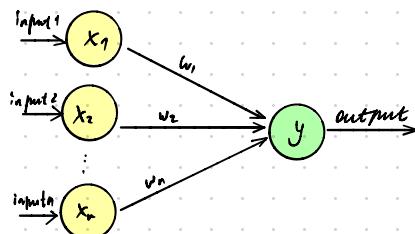
Backpropagation

Продукты для классификации:

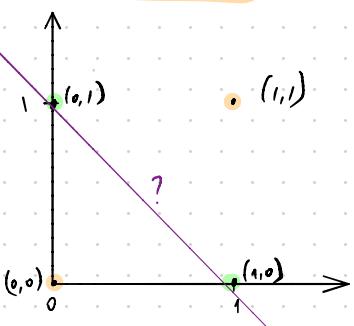
- гиперболическая синусоидальная
- полиномиальная
- логистическая
- гиперболический тангенс
- косинусоидальная

• биогиперболическое гиперболическое

Нейрон $n =$ одиничный классификатор



XOR problem:



Рекуррентные нейроны моделируют живой организм мышь, 2 часа, биологический элемент

Universal approximation theorem

- модель нейрона может аппроксимировать с любой степенью точности любые непрерывные функции с единицей единицой и непрерывными непрерывными непрерывными непрерывными непрерывными
- функция аппроксимации называется функцией, которая выполняет любую заданную функцию

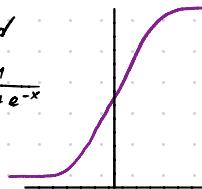
- even neurons **gradually**, as x approaches $+\infty$, move $\rightarrow 0$, converging towards 0
 - gradient of **single** neuron. If x goes away from zero increases exponentially towards infinity, but gradient of all neurons
- Generalization vs Extrapolation**
- ↑ something the approximators do well something they do not do well
- * neural networks aren't fit to perform outside of the scope they were taught in

Adding layers: higher layers watch for patterns within patterns

Pyramidal architecture

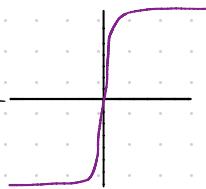
sigmoid

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



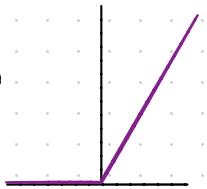
tanh

$$\tanh(x) = \frac{e^{2x}-1}{e^{2x}+1}$$



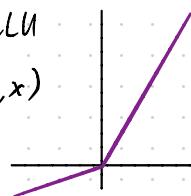
ReLU

$$\max(0, x)$$



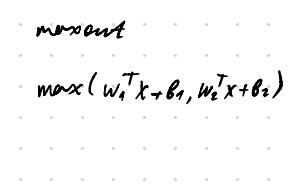
leaky ReLU

$$\max(0.1x, x)$$



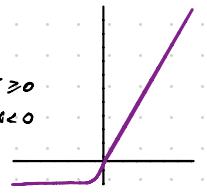
maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$



ELU

$$\begin{cases} x & x \geq 0 \\ d(e^x - 1) & x < 0 \end{cases}$$



Backpropagation through time

- many components required for backpropagation

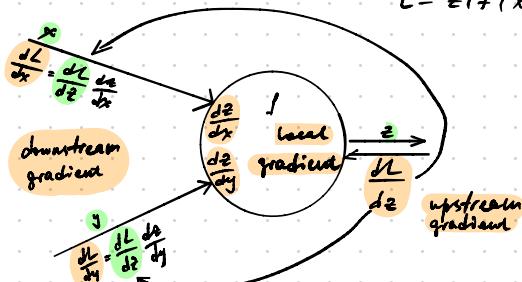
$$L = z(f(x, y))$$

Example

$$f(x, y, z) = (x+y)z \quad g = x+y \quad \frac{dg}{dx} = 1 \quad \frac{dg}{dy} = 1$$

$$\text{Want } \frac{df}{dx}, \frac{df}{dy}, \frac{df}{dz} \quad f = gz \quad \frac{df}{dx} = 2 \quad \frac{df}{dz} = g$$

$$\text{chain rule } \frac{df}{dz} = \frac{df}{dt} \frac{dt}{dz} \quad \text{upstream} \nearrow \text{local}$$

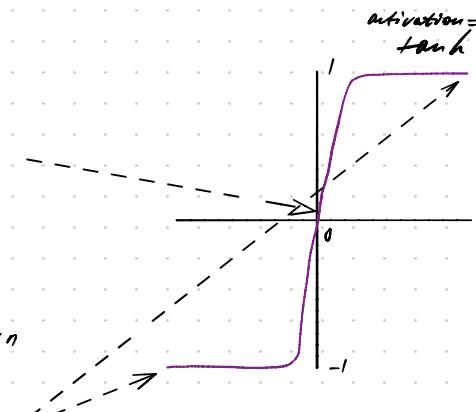
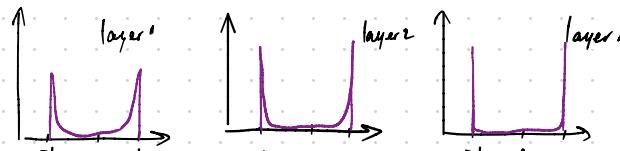
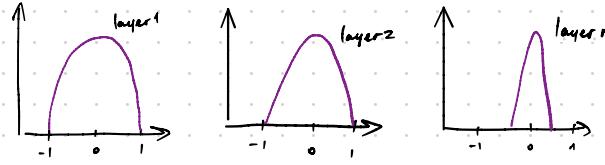


always known

normalisierung & unnormalisierung

- unnormalisierte - normale reihenfolge nach vorne
normalisierte - rückwärts

synchrone normalisierung



normalisierung mit einer Schicht

$$W = \text{np.random.rand}(D_{in}, D_{out}) / \text{np.sqrt}(D_{in})$$

normalisierung mit ReLU

$$x = \text{np.maximum}(0, x \cdot \text{dot}(W))$$

normalisierung mit Kaiming (ICMLR)

$$W = \text{np.random.rand}(D_{in}, D_{out}) * \text{np.sqrt}(2/D_{in})$$

Batch normalization

fixe synchrone parameterneinstellung bei mehreren Schichten

benutzt wahrsch. normalisierung aber bei: normalisierung, aufeinander folgende Schichten und
im Volumen aus

normalisierung:

- adaptive synchrone
- meistens aufeinanderfolgende Schichten
- fiktiver Wert für unnormalisierte
- wir benötigen den korrekten Wert für die Schicht
- fragmentiert in Gruppen
- Synchronisiert mit adaptivem

für convolutional layers:

fully connected

$$x : N \times D$$

$$\mu, \sigma^2 : 1 \times D$$

$$\gamma, \beta : 1 \times D$$

$$y = \gamma(x - \mu) / \sigma + \beta$$

für convolutional networks

$$x : N \times C \times H \times W$$

$$\mu, \sigma^2 : 1 \times C \times 1 \times 1$$

$$\gamma, \beta : 1 \times C \times 1 \times 1$$

$$y = \gamma(x - \mu) / \sigma + \beta$$

Optimizing a loss

Problem with SGD: vanishing/exploding gradients

SGD

$$x_{t+1} = x_t - \alpha \nabla f(x_t)$$

SGD + Momentum

$$v_{t+1} = \beta v_t + \nabla f(x_t)$$

build up velocity
as a running mean of
gradients

AdaGrad (ageneralization of momentum)

gradient sparsity

$$\delta x = \text{compute_grad}(x)$$

$$\text{grad_squared} += \delta x * \delta x$$

$$x = l_r * \delta x / (\text{np.sqrt}(\text{grad_squared}) + 1e-7)$$

numerop

numerop

- element-wise scaling of the grad based on the historical sum of squares in each dim
- adaptive learning rates

RMSProp

$$\text{grad_squared} = \text{decay_rate} * \text{grad_squared} + (1 - \text{decay_rate}) * \delta x * \delta x$$

Adam (adaptive moment estimation)

- momentum
- bias correction
- AdaGrad / RMSProp

NeurIPS 31.10.2022

hyperparameter choice strategy: cross-validation on validation set

Hyperparameter search

- optimization information

- numerop

numerop regularization & sparse gradients

Akkumulacija CNN

		accuracy
KNN		0.37
linear classifier		0.91
2-layer nn		0.53
3-layer nn	32/64/96	0.67 - 0.72

komponentos CNN

- nurov obrazinio suai FC
- autokorrelas
- ceptrus
- pooling
- batch normalization
- backprop
- nograminė gamtos
- pyramidinės
- nurov, užduotis
- optimizatorius

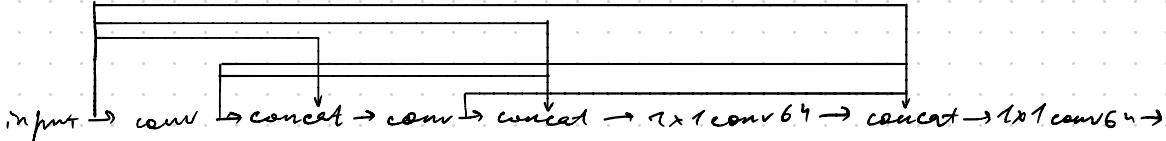
input \rightarrow conv + relu \rightarrow pooling \rightarrow conv + relu \rightarrow pooling \rightarrow flatten \rightarrow FC \rightarrow softmax,
 feature learning classification

Feature extraction

image \rightarrow conv + pooling \rightarrow final conv \rightarrow fully connected layers \rightarrow class scores \rightarrow softmax loss
 feature map classifier

DensNet (Densely connected convolutional networks)

- rečiausios išskirtis
- nonameinamo funkcs symmetrija
- gauskai suplokus
- cenuotinės
- nepriklausomai beveikab - nusijekas



Groups Convolution

- направление обратима
- уменьшение числа параметров
- подходит для архитектур

Lesson 6 29.11.2022

Transfer learning

Быстро обучение сети, имеющей ранее обученное ядро
→ для этого требуется настройка параметров

- * базовая архитектура обучена и имеет параметры, которые можно использовать

Асимметрическое

использование обучения с гамильтоновыми и противогамильтоновыми функциями, симметрии, нечеткости

不平衡 dataset

- коррелирует с различными классами
- некоторые классы в примерах → learning transfer
- одинаковый размер
- асимметрические
- коррелируют с предсказанием / асимметрически
- unbalanced