**Neural Networks**

Main Type : Convolutional Neural Network (CNN), Recurrent Neural Network (RNN)

Image Classification

Test image; they are all labeled with a single category.

Predicting these categories for a novel set of test images and measure the accuracy of the predictions.

1. Nearest Neighbor Classifier

The nearest neighbor classifier will take a test image, compare it to every single one of the training images, and predict the label of the closest training image.

* + Pros : very simple to implement and understand, he classifier takes no time to train, since all that is required is to store and possibly index the training data.
  + we often care about the test time efficiency much more than the efficiency at training time.

1. k - Nearest Neighbor Classifier

In the idea, instead of finding the single closest image in the training set, we will find the top k closest images, and have them vote on the label of the test image.

→How to find the “K”→ hyperparameters

1. Linear Classification

KNN problem

* The classifier must remember all of the training data and store it for future comparisons with the test data. This is space inefficient because datasets may easily be gigabytes in size.
* Classifying a test image is expensive since it requires a comparison to all training images.

So we can use following functions

f(xi,W,b)=Wxi+b

xi is vector of shape [D x 1], the matrix W (of size [K x D]), the vector b (of size [K x 1])

changing W depend on picture

* Score function (raw dataを分類クラスごとにスコア付け)

maps the raw data to class scores

a linear function that depends on weights W and biases b)

Unlike kNN classifier, the advantage of this parametric approach is that once we learn the parameters we can discard the training data. Additionally, the prediction for a new test image is fast since it requires a single matrix multiplication with W, not an exhaustive comparison to every single training example.

* Loss function (予測したスコアとラベルの持つスコアの差を計算)

We do have control over these weights and we want to set them so that the predicted class scores are consistent with the ground truth labels in the training data.

Making good predictions on the training data is equivalent to having a small loss

* + Multiclass Support Vector Machine loss(SVM)

SVM “wants” the correct class for each image to a have a score higher than the incorrect classes by some fixed margin

* + Softmax classifier

Softmax classifier is its generalization to multiple classes

scores for each class, the Softmax classifier gives a slightly more intuitive output (normalized class probabilities) and also has a probabilistic interpretation that we will describe shortly.

1. Optimization : to find W that minimizes the loss function.

Strategy 1: Random search, bad idea, accuracy :15.5%

Strategy 2: Random Local Search: random searching a direction at first → take a next step only when it leads downhill, accuracy 21.4%

Strategy 3: computing the best direction along by changing the weight vector that is mathematically guaranteed to be the direction of the steepest descend

Computing gradient descent

numerical gradient → easy way

analytic gradient → accurate

* + Vanilla
  + Mini-batch gradient descent : for big training data set,
  + Stochastic Gradient Descent (SGD)

1. Backpropagation, Intuitions

way of computing gradients of expressions through recursive application of chain rule.

Forward pass = computing values from inputs to output (start to end)

The backward pass = starting at the end and recursively applies the chain rule to compute the gradients all the way to the inputs of the circuit (end to start) = backpropagation

Intuitive understanding of backpropagation

1. Neural Networks Part 1: Setting up the Architecture

Input - Dendrites - Neuron – axon - synapses – output

f\*sum(ω\*x+b)

input signal x

synaptic strength ω

bias b

activation function f

activation functions type

* Sigmoid
* Tanh
* ReLU
* Maxout
* TLDR

Neural Networks = neurons graph

fully-connected layer : eurons between two adjacent layers are fully pairwise connected

there are 3 layer type : input, hidden, output

N-layer neural network, do not count the input layer.

More neurons layer;

Pro

* express more complicated functions

Con

* Overfitting

1. Setting up the data and the model

Data Preprocessing

* Mean subtraction : set a average to center

X -= np.mean(X, axis = 0)

* Normalization : set a average to center, and the min and max along the dimension is -1 and 1 respectively

X /= np.std(X, axis = 0)

Convolutional Neural Networks (CNNs / ConvNets)

Goof at images, allows us to encode certain properties into the architecture.

These then make the forward function more efficient to implement and vastly reduce the amount of parameters in the network.

1. Architecture, Convolutional Layer, Pooling Layer, Fully-Connected Layer

* Architecture

CNN architecture has 3-D, width, height, depth

three main types of layers to build CNN architectures, Convolutional Layer, Pooling Layer, Fully-Connected Layer

Order :

INPUT - Convolutional Layer - RELU - Pooling Layer - Fully-Connected Layer

Continue to process between Convolutional Layer and Pooling Layer and reduce the inputs size.

* Convolutional Layer

The CONV layer’s parameters consist of a set of learnable filters.

extend through the full depth, small height and width

Local Connectivity

× high-dimensional inputs – connect all neurons

connect each neuron to only a local region of the input volume

* Pooling Layer

insert a Pooling layer in-between successive Conv layers

reduce the spatial size of the representation to reduce the amount of parameters and computation in the network

* Example of CNN
* LeNet
* AlexNet
* ZF Net
* GoogLeNet
* VGGNet
* ResNet

Convolution Neural Network (CNN)

3 layers : convolution, pooling, full connection

Convolution layer: filter(weight, biasをもつ)を使用しデータを抽出する→算出されたものを特徴マップと呼ぶ

Pooling layer: 特徴マップをさらに圧縮する→微小な位置変化に対して頑健, 過学習を抑制, 計算コストを下げる

Full connection: 各層のユニットと次の層のユニットを全結合。ユニットとの接続は重みを持っている

Residual Network（ResNet）

CNNの層を深くした版

層を深くすることで特徴の認識レベルを高める

層の入力を参照した残差関数を学習することで層を深める