SHANMUGA PERUMAL HARIKUMAR
6418 3191 11

Dec 15 '16

Deep Learning Final Exam

1) Backpropagation:

a) No, the algorithm may get stuck at a local minima depending upon initialization. Since we are using entire dataset to update weights, the algorithm has no way of escaping local minima

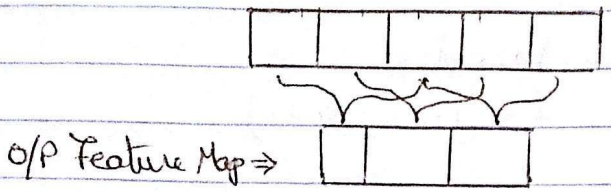
b) Depending upon the learning rate, the loss might get better or worse. If the learning rate is small enough, the loss must decrease.

c) No, we might get stuck at a local minimum with no way of escaping from it

d) No, we are not trying to optimize that directly, hence during training, there is no guarantee the classification error never increases.

ConvNets : Basics:

2) a) Let us consider a feature of size(n) 5 and a kernel of size(K) 3



The Size of the output feature map m is,

$$m = n - K + 1$$

⇒ The Size of f output feature maps is

$$f * (n - K + 1)$$

b) Not Counting the biases, the total number of parameters is equal to the number of weights in the layer.

Given kernel size K and f such kernels, total number of parameters is $K * f$

c) The process of convoluting the input with kernels consists of multiplying and accumulations of the values in the input with kernel values.

For input dimension n and kernel dimension K, Each layer would have $n-K+1$ multiply-accumulate operations. Considering f such kernels, the total number of multiply accumulate operations as $f * (n - K + 1)$

d) With a stride of 's', the size of each feature map is

$$m = (n-k)/s + 1$$

For f such kernels

$$m = f * ((n-k)/s + 1)$$

3) Convnet Object Detection:

a) Since the minimum height of pedestrians is 30, width cannot always be less than 15 we will be using a sliding window at different scales to find candidate regions. The minimum size of window will be ~~30x60~~ 60x30, in order to detect even the 15 Pixel wide pedestrians

b) Since a sliding windows at different scales is being used to detect candidate regions, scale normalization and cropping is taken care of.

To select negative examples, the following procedure can be followed

1) Starting from training dataset, randomly sample N regions without pedestrians

2) Compute color histogram of regions and normalize it

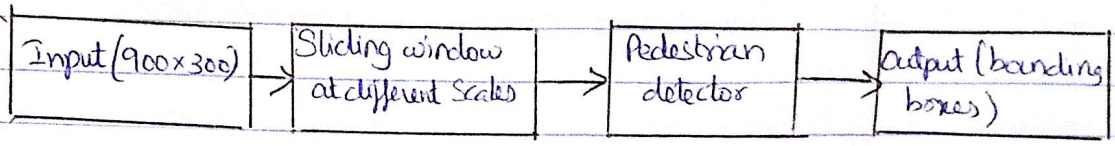
3) Compute euclodian distance of Cumulative color probability of each pair of negative regions

4) Select negative region with highest distance and remove from pool

5) Repeat 4 till required number of regions selected

c) In order to reduce false negatives the sliding window is very accurate as it passes through all regions of image at different scales. Also falsely negative classifications of images in validation set can be added to training set.

d)



The Sliding window slices the image into ~~to~~ smaller patches at different scales. This is sent to the pedestrian detector, since a sliding window is used the size range of pedestrians depends on the limit on the scale at which the window searches. Also each patch is independently being processed, so multiple pedestrians will not be a problem.

The size of the bounding box would depend on the size of the patch pedestrian was found at.

4) ConvNets: Weak Supervision

a) The architecture consists of Consecutive convolution and pooling layers followed by deconvolution layer consisting of deconvolution and unpooling layers. The reverse operations depend on the masks, weights in the forward operation. Based on the featuremap of highest abstraction level details of features are stored using deconvolutional layers. The entire feature map is then expanded and concatenated. This is followed by a convolution to generate class-specific activation map. These are finally softmaxed and aggregated into single vector, which is compared with image level vector label.

5) Word, Text and Image Embedding:

a) Distance Function, $D_W(\vec{X}_1, \vec{X}_2) = ||G_W(\vec{X}_1) - G_W(\vec{X}_2)||_2$

where $\vec{X}_1, \vec{X}_2$ are pair of input vectors

Then Loss function is

$$L(W) = \sum_{i=1}^{P} L(W, (Y, \vec{X}_1, \vec{X}_2)^i)$$

where, $L(W, (Y, \vec{X}_1, \vec{X}_2)^i) = (1-Y) L_S(D^i_W) + Y L_D(D^i_W)$

$L_S \rightarrow$ Partial loss function for similar pair of points

$L_D \rightarrow$ Partial loss function for dissimilar pair of points

The exact loss function beans

$$L(W, Y, \vec{X}_1, \vec{X}_2) = (1-Y)\frac{1}{2}(D_W)^2 + (Y)\frac{1}{2}\{\max(0, m - D_W)\}^2$$

b) A weighted least square regression model defined below can be used as an objective function in the given case of word embedding by GloVe

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2$$

where V → Size of vocabulary

The weighting functions $f(x)$ should obey following properties

1) $f(0)=0$, if f is seen as continuous, it must vanish as $x \to 0$ fast enough that $\lim_{x \to 0} f(x) \log^2 x$ is finite

2) $f(x)$ should be non decreasing, so that rare co occurances are not overweighted.

3) $f(x)$ should be relatively small for large values of x, so that frequent co-occurances are not over-weighted

c) Let ranking error function

$$err(f(x), y) = L(rank_y(f(x)))$$

where $rank_y(f(x))$ is the rank of true label y given by $f(x)$

$$rank_y(f(x)) = \sum_{i \neq y} I(f_i(x) \geq f_y(x))$$

where I is indicator function and $L(\cdot)$ transforms this into loss

$$L(k) = \sum_{j=1}^{k} \alpha_j \text{ , with } \alpha_1 \geq \alpha_2 \geq \ldots \geq 0$$

This lets us use different $L(\cdot)$ with different minimizers.

6) Recurrent Nets:

a) The gradients for Recurrent Neural Networks (RNNs) are calculated by an algorithm called Back Propagation Throughtime (BPTT) This is similar to normal Back Propagation but the recurrent network is expanded to how many ever time steps are required and then backpropagations is done.

The gradient Jacobian Matrix elements are pointwise derivatives. Both tanh and Sigmoid functions have 0 derivatives at one point. This zero gradient drives gradients in previous layers towards zero. As those small valued matrices get multiplied, they shrink and vanish in few time steps. Gradients from far away time steps become zero and do not contribute in learning. This is vanishing gradient problem.

For other activation functions the matrix values can become large leading to exploding gradient problems

b) Initializing the hidden weight matrices to identity, so when the network BPTT is carried out without inputs the matrices stay the same, which is not the case in non-identity initializations

c) LSTM [Long Short Term Memory] Networks are designed to remember information for a long time. Unlike normal RNNs LSTMs have a different recurring module. Instead of single neural network layer, there are four interacting in special way The cell state in LSTMs can pass information along, uninterrupted. The LSTM modules have the power to change cell states, regulated by structures called gates

Gates are a way to optionally let information through. LSTM has three such gates

Gated Recurrent Units (GRU) are similar to LSTMs but have fewer parameters and hence generally preferred. GRUs have two gates

7) Energy-Based learning: Weakly Supervised Object Localization

a) The entire image may be separated into 32×32 or 50×50 slices. Each slice is given as input to a deep neural network of convolutional and subsampling layers which produce a final vector output.

This network is replicated for all slices, producing a vector for each slice. This process is repeated across multiple scales.

~~The network is~~

b) The loss function

$$L_0(W,0,x^i) = k \exp\left[-E(1,\bar{z},x^i)\right]$$

where k is positive constant

$$L_1(W,1,z,x) = E_W(1,z,x)^2$$

The loss function combine is

$$L(W,S) = \frac{1}{|S_1|} \sum_{i \in S_1} L_1(W,z^i,x^i) + \frac{1}{|S_0|} \sum_{i \in S_0} L_0(W,x^i)$$

S_1 - Subset of training sample with pedestrian

S_2 - Subset of training sample without pedestrian.

The network can be trained by optimizing a loss function of three labels, pedestrian presence, location and image when three variables match the energy function is trained to have a smaller value and larger value if they don't match.

8) Unsupervised Learning and Auto-Encoders:

a) The shape of the energy function $I(y,w)$ in the space of y should be a surface with low points along the data points. The energy should increase as one moves away from the function that ~~the~~ the learning algorith is trying to approximate.

b)

objective function

$$J_{sparse}(w,b) = J(w,b) + \beta \sum_{j=1}^{S_2} KL(\rho \| \hat{\rho}_j)$$

where $w \rightarrow$ weights
$b \rightarrow$ bias
$\beta \rightarrow$ weighting of penalty term
$\rho \rightarrow$ sparsity parameter

$KL(\rho \| \hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}_j}$ is the penalty term

Average activation of hidden layer a_j is given by

$$\hat{\rho}_j = \frac{1}{m} \sum_{i=1}^{m} \left[a_j^{(2)}(x^{(i)}) \right]$$

Actual loss without penalty is

$$J(w,b) = \left[\frac{1}{m} \sum_{i=1}^{m} J(w,b;x^{(i)},y^{(i)}) \right] + \frac{\lambda}{2} \sum_{l=1}^{n_l-1} \sum_{i=1}^{s_l} \sum_{j=1}^{s_{l+1}} \left(w_{ji}^{(l)} \right)^2$$

where

~~$J(W,b)=$~~

$J(W,b;x,y) = \frac{1}{2} \| h_{W,b}(x) - y \|^2$ is individual loss term

Sparsity Constraint:

Enforcing a constraint that

$$\hat{\rho}_j = \rho$$

where $\hat{\rho}_j$ → average activation of hidden units j
ρ → sparsity parameter

sparsity parameter is generally a small value
To satisfy this constraint, hidden units activation must be near 0.

c) A denoising autoencoder is similar to a normal autoencoder, but the input is corrupted. A denoising autoencoder is trained to retrieve original input data from the corrupted data.

9) Optimization:

a) The standard deviation of inputs is inversely proportional to the learning rates

$$\sigma \propto 1/\nu \qquad \nu \propto 1/\sigma$$

$$\frac{\nu_1}{\nu_2} = \frac{1/\sigma_1}{1/\sigma_2} = \frac{1/1}{1/3} = 3:1$$

$$\nu_1 : \nu_2 = 3:1$$

where, σ - Standard deviations ν - learning rates

b)

$$\text{Hinge loss} = l(y) = \max_{t \neq y}(0, 1 + \max w_t x - w_y x)$$

the degree is 1

c) This suggests that there exists numerous landscapes that have their own local minima, which the algorithm will reach. This means there are multiple approximations of the same function that the neural network will reach. This symmetry also suggests that the gradient descent algorithm will do similar actions in all such landscapes. Additionally since in a fully connected network every neuron affects every following neuron, the symmetry is understandable.

10) General Questions:

a) Eventhough it is possible to approximate any well behaved function with a two layered network, the number of parameters required to do so is not feasible. So we need deep neural networks to better approximate. Also having more layers makes it easier to detect and learn hierarchical features.

b) Since most real world functions are non-linear, deep networks use non-linear layers to better approximate such functions. Linear layers could also be replaced by the sum of such non-linearities as a single layer, which in turn loses the freedom to better approximate functions

c) The commonly used modules include
→ convolutional layers
→ max pooling layers
→ fully connected layers
→ flatten module
→ dropout layers
→ Non-linear activation module.

d) Siamese networks contain two or more sub-networks that are similar in every sense, the weights, parameters. even parameter updating is mirrored in both networks. They are used to find similarity or connections between comparable things. Ex) Signature verification