Homework 1

CSE 291 I00: Machine Learning for 3D Data (Winter 2018)

PLEASE READ. Submit paper copies of your solutions on January 25th 2018. For codes, print them out. You are encouraged to discuss in groups (two people for each group), but please complete your programming and writeup individually. Indicate the name of your collaborator in your writeup.

Problem 1. Consider the following neural network:

$$\begin{aligned} h_1 &= W_1 X + b_1 \\ a_1 &= sigmoid(h_1) \\ h_2 &= W_2 a_1 + b_2 \\ a_2 &= tanh(h_2) \\ o &= W_3 a_2 + b_3 \\ p &= softmax(o) \\ softmax(o_i) &= \frac{e^{o_i}}{\sum_{j=0}^{K} e^{o_j}} \quad for\ i\ in\ \{0, 1, ..., K\} \end{aligned}$$

where, h_n denote the hidden layers, a_n denotes the activation layers, W_n are the weights, X being the input to the neural network, o denotes the output layer and p denotes the predicted probabilities. The cross-entropy loss is used as the loss function and is given by:

$$L = - \sum_{for\ all\ class\ c} y_c\ log(p_c)$$

where y is the target labels (one-hot vector, i.e., the $y_c = 1$ if the label of the instance is c).

Compute the derivative of the cross-entropy loss L w.r.t o, i.e compute $\frac{\partial L}{\partial o}$. (25 marks)

Problem 2. Draw the computational graph of the network described in Problem 1. Using the derivative calculated in the previous question, perform Backpropagation to compute the gradients of loss L w.r.t all the weights and biases, i.e compute $\frac{\partial L}{\partial W_1}$, $\frac{\partial L}{\partial W_2}$, $\frac{\partial L}{\partial W_3}$, $\frac{\partial L}{\partial b_1}$, $\frac{\partial L}{\partial b_2}$, $\frac{\partial L}{\partial b_3}$. (25 marks)

Problem 3. Batch Normalization is a technique that forces the input to any layer to be zero mean and unit standard deviation. Below is the algorithm from the paper (Refer:

https://arxiv.org/pdf/1502.03167.pdf), Using the algorithm, draw the computational graph of the batchnorm layer. Consider a function $F(y_i)$ and compute the derivatives of $F(y_i)$ w.r.t x, γ and β, i.e compute $\frac{\partial F}{\partial \gamma}, \frac{\partial F}{\partial \beta}, \frac{\partial F}{\partial x}$. Assume $\frac{\partial F}{\partial y_i}$ is given. (25 marks)

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Problem 4. Consider the MNIST dataset. It consists of 10 class labels (0-9) and has 60,000 training images and 10,000 test images.

1. Construct a model using fully connected layers (at least 3 layers or more!) and ReLu layers to solve this classification problem using Tensorflow. Report the accuracy obtained on the test set. Plot a graph demonstrating how the loss function decreases over the number of iterations. (5 marks)

2. Add batch normalization layers in the model. Report the accuracy obtained and plot a graph showing how loss decreases. Elaborate briefly on how and why batch normalization helped. (5 marks)

3. For the same dataset, train a Convolutional Neural Network (with and without batchnorm). Try experimenting with different architectures (different optimizers, number of convolutional layers, etc) and report the accuracy that you obtained with both using and without using batchnorm. Plot the loss vs iterations graph and explain why and how batch normalization helped. (15 marks)