DS-GA 1008 Deep Learning

Homework 1

Xing Cui(xc918), Sida Ye(sy1743), Junchao Zheng(jz2327)

March 01, 2017

I. Backprop

I. Nonlinear Activation Functions

Solution:

1. sigmoid:

$$\begin{aligned}
\frac{\partial E}{\partial x_{in}} &= \frac{\partial E}{\partial x_{out}} \cdot \frac{\partial x_{out}}{\partial x_{in}} && (1)\\
&= \frac{\partial E}{\partial x_{out}} \cdot \partial((1+exp(-x_{in}))^{-1}) && (2)\\
&= \frac{\partial E}{\partial x_{out}} \cdot \frac{exp(-x)}{(1+exp(-x))^2} && (3)\\
&= \frac{\partial E}{\partial x_{out}} \cdot (x_{out} - x_{out}^2) && (4)
\end{aligned}$$

2. tanh:

$$\begin{aligned}
\frac{\partial E}{\partial x_{in}} &= \frac{\partial E}{\partial x_{out}} \cdot \frac{\partial x_{out}}{\partial x_{in}}\\
&= \frac{\partial E}{\partial x_{out}} \cdot \partial(\frac{exp(2x_{in})-1}{exp(2x_{in})+1})\\
&= \frac{\partial E}{\partial x_{out}} \cdot \frac{4exp(2x)}{(1+exp(2x))^2}\\
&= \frac{\partial E}{\partial x_{out}} \cdot (1-x_{out}^2))
\end{aligned} \quad (5)$$

3. ReLU:

When $x_{in} > 0$, $\frac{\partial x_{out}}{\partial x_{in}} = 1$. When $x_{in} \leq 0$, $\frac{\partial x_{out}}{\partial x_{in}} = 0$. So,

$$\frac{\partial E}{\partial x_{in}} = \frac{\partial E}{\partial x_{out}} \cdot \frac{\partial x_{out}}{\partial x_{in}} \tag{6}$$

$$= \begin{cases} \frac{\partial E}{\partial x_{out}} & \text{if } x_{in} > 0 \\ 0 & \text{if } x_{in} \leq 0 \end{cases} \tag{7}$$

II. Softmax

Solution:

When $i = j$,

$$\frac{\partial (x_{out})_i}{\partial (x_{in})_j} = \partial(\frac{exp(-\beta(x_{in})_i)}{exp(-\beta(x_{in})_i) + \sum_{k\neq i} exp(-\beta(x_{in})_k})\tag{8}$$

$$= \partial(\frac{1}{1 + (\sum_{k\neq i} exp(-\beta(x_{in})_k)) \cdot exp(\beta(x_{in})_i})\tag{9}$$

$$= \frac{-\beta \sum_{k\neq i} exp(-\beta(x_{in})_k) exp(-\beta x_i)}{(1 + (\sum_{k\neq i} exp(-\beta(x_{in})_k)) \cdot exp(\beta(x_{in})_i)^2}\tag{10}$$

$$= \frac{-\beta \sum_{k\neq i} exp(-\beta(x_{in})_k) exp(\beta x_i)}{\sum_k exp(-\beta(x_{in})_k)}\tag{11}$$

$$= -\beta(x_{out})_i(1 - (x_{out})_i)\tag{12}$$

When $i \neq j$,

$$\frac{\partial (x_{out})_i}{\partial (x_{in})_j} = exp(-\beta(x_{in})_i)\partial(\frac{1}{exp(-\beta(x_{in})_j) + \sum_{k\neq j} exp(-\beta(x_{in})_k)})\tag{13}$$

$$= exp(-\beta(x_{in})_i)\frac{\beta exp(-\beta(x_{in})_j)}{\sum_k exp(-\beta(x_{in})_k)^2}\tag{14}$$

$$= \beta(x_{out})_i(x_{out})_j\tag{15}$$

II. Techniques

I. Optimization

Solution:

Classical Momentum(CM) method:

$$v_{t+1} = \mu v_t - \epsilon \nabla f(\theta_t)\tag{16}$$

$$\theta_{t+1} = \theta_t + v_{t+1}\tag{17}$$

Nesterov's Accelerated Gradient(NAG) method:

$$v_{t+1} = \mu v_t - \epsilon \nabla f(\theta_t + \mu v_t) \tag{18}$$
$$\theta_{t+1} = \theta_t + v_{t+1} \tag{19}$$

These two methods are quite similiar expecially by taking a look at formulas. However, the main differences are noticable. Both CM and NAG compute the new velocity by applying a gradient-based correction to the previous velocity vector, which is decayed, and then add the velocity to θ_t. In the CM method, we take the gradient on θ_t, then we use momentum v_{t+1} to corrct it. In NAG method, we first performs a partial update to θ_t by computing $\theta_t + \mu v_t$, then we correct it. This small change lets NAG change v in a quicker and more responsive way and simoutaneously make it behave stably especially with larger value of μ.

II. Reducing Overfitting

II.1 Dropout

Solution:

First, ensembling is defined as a general term for combining many classifiers by averaging or voting. Adaboost and GBM are examples of ensembling models. In this part, if dropping a neuron could treat as voting 0, and keeping could treat a voting 1. We can say each neuron is a model, and we combine many kept models that were voted 1. For each training case, a new network is sampled and trained. Therefore, training a neural network with dropout can be visualized as training a collection of sampled network with extensive weight sharing, which is an ensembling model.

II.2

Solution:

During the training time, the output of each layer, $y^{(l)}$, is multiplied element-wise with a dropout vector, $r^{(l)}$, which is a vector of independent Bernoulli random variables with probability p of being 1, to create a final output $\hat{y}$, which is the input for the next layer. In the test time, the weights are scaled as $W_{test}^{(l)} = pW^l$. In this case, the neural network is without dropout. In our experiment, the dropout probability is 0.5 as keep probability.

II.3 Data Augmentation

Solution:

We have implemented three method to enlarge labeled training dataset to avoid data scarcity problem. They are rotation, translating, and adding white noise. These methods are optical processes on images so that are fitting the MNIST dataset. The data augmentation is aiming at expanding the amount of data and proventing overfitting.

III. Initialization

III.1

Solution:

The main idea of the initialization method in He's findings is to investigate the variance of responses in each layer. In He's work, a zero-mean Gaussian distribution with $std = \sqrt{\frac{2}{n_l}}$. $n_l = (k^2c)l$ is the number of connections of a response of layer L in forward propagation, and similarly, a zero-mean Gaussian distribution with $std = \sqrt{\frac{2}{\hat{n}_l}}$ in back propagation. In Xavier's paper, a uniform distribution, $W_{ij} \sim U[-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}]$ is used to do weight initialization. The first difference is He's method takes ReLU and PReLU as the rectifier nonlinearities. Moreover, He allows extremely deep model with up to 27 ConvNet and 3 fully connected layers. These two do not appear in Xavier's paper.

III.2

Solution:

The pre-processing that VGG team used is sustracting the mean of RGB value and computing on the training dataset from each pixel. In Coate's paper, the method has 3 steps: step 1, feature learning: extract random patches from unlabelled training samples. Step 2, pre-processing: normalizing and whitening. Step 3, Unsupervised learning: learn a feature-mapping using an unsupervised learning. Our approach is using Pseudo-label for semi-supervised learning. We use labelled data to train a primary model and label the unlabelled data by the class with highest probability. Then we train the model on both labelled and unlabelled data. After each epoch, we update the labels of unlabelled data. This method is called Pseudo-label.

III. PyTorch(MNIST Handwritten Digit Recognition)

I. Model Architecture and Configuration

We used the neural network with 3 hidden layers. For each layer, we have 5,3,2 as kernel size and 0.5 as dropout probability. ReLU is used for hidden unit. We used a softmax for output unit. For optimization, we used mini-batch SGD with dropout as 0.5. The initial learning rate is 0.01. After each epoch, it multiples with 0.998. The batch size for labeled data is 128 and for unlabeled data is 64. These parameters were determined using validation dataset.

II. Data Augmentation

We used data augmentation, which is introduced in section 2, to improve our accuracy. We adopted three methods. First is rotation. We randomly rotate an image from -45 to 45 degrees. Second is vertical movement. We randomly move an image either up or down from 1 to 4 units. The last one is adding noise. We added some noise to an image, which followed a Gaussian distribution with mean as 0 and std as 1. We only add or minus value to non-zero pixels. By implementing all methods, we expend 3000 labeled data into 90000. We applied each method to the original data and expend it into 30000. Hence, in total, we have 90000 labeled training data after adopting data augmentation.

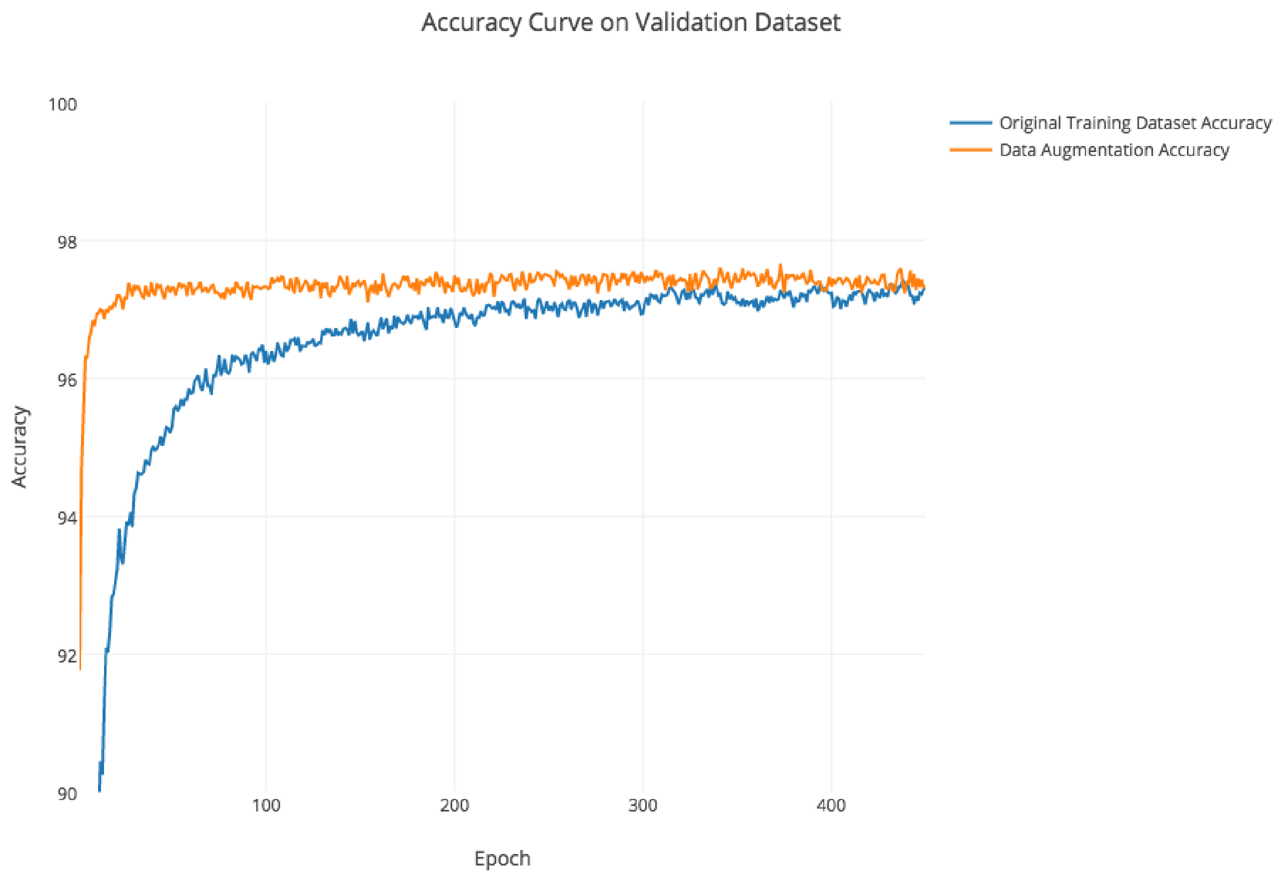

Figure 1: *Validation Accuracy Curve*

By looking at figure 1, we can see that the data augmentation slightly improve the accuracy.

III. Pseudo-Label

We applied Pseudo-Label method as our semi-supervised learning method. We have 90,000 labeled data and 47,000 unlabeled data. We train them simultaneously. We have a total loss which is the summation of labeled loss and unlabeled loss with a multiplier alpha.

$$loss = loss\_label + \alpha * loss\_unlabel$$

The alpha is calculated as following according to Dong-Hyun Lee's paper. For each epoch, we first predict unlable data by using current model. Then, we train both data simultaneously. We also choose an exponentially decaying learning rate, which is multiplied by 0.998 after each epoch.

$$\alpha(t) = \begin{cases} 0, & t < T_1 \\ \frac{t-T_1}{T_2-T_1}\alpha_f, & T_1 \leq t < T_2 \\ \alpha_f, & T_2 \leq t \end{cases} \tag{20}$$

With $\alpha_f = 3$, $T_1 = 100$, $T_2 = 600$.

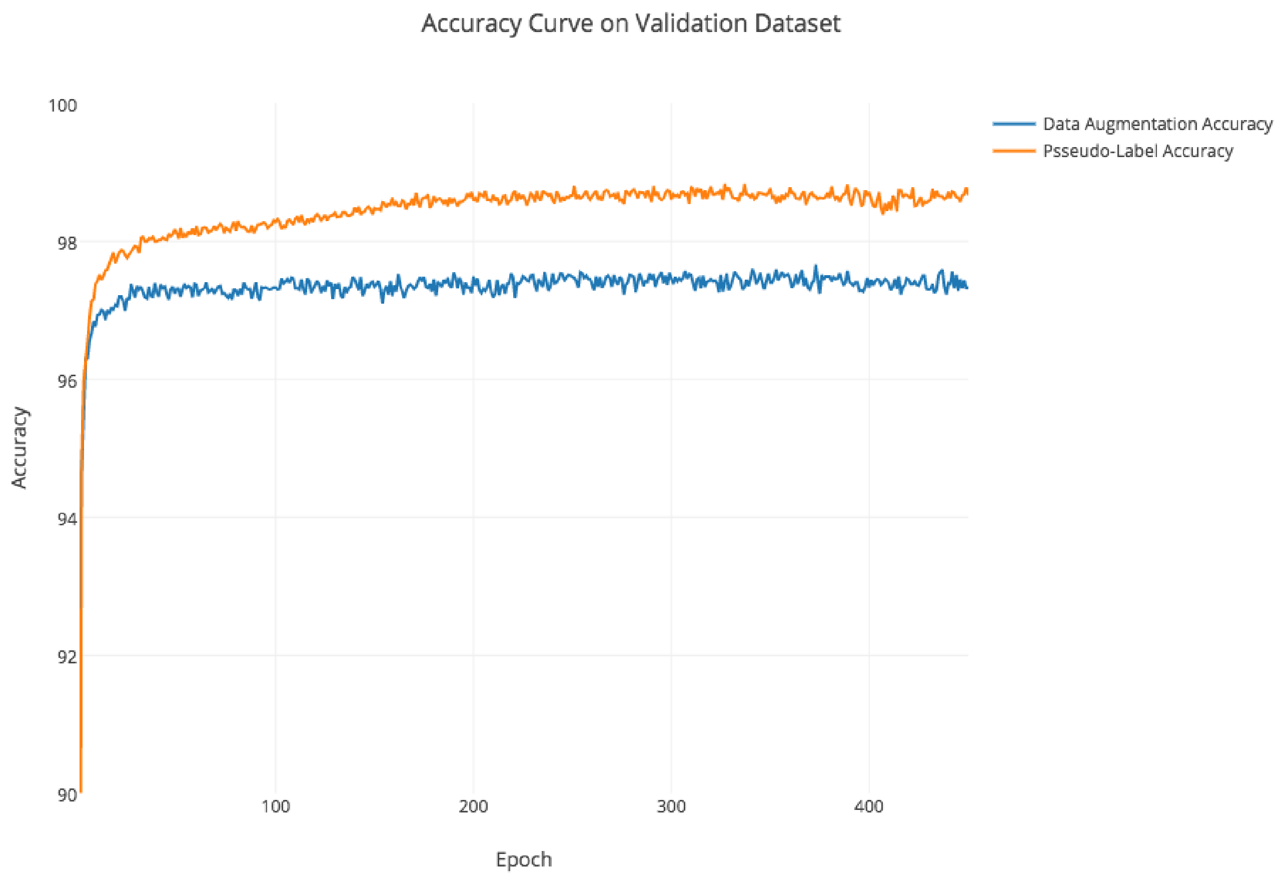

Figure 2: *Validation Accuracy Curve*

By checking figure 2, we can see that pseudo-label method indeed improve the model accuracy on validation dataset.

IV. Reference

Sutskever, I., Martens, J., Dahl, G. and Hinton, G: *On the importance of initialization and momentum in deep learning*, ICML (3) 28 (2013): 1139-1147.

Srivastava, Nitish, *Dropout: a simple way to prevent neural networks from overfitting.*, Journal of Machine Learning Research 15.1 (2014): 1929-1958.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, *ImageNet Classification with Deep Convolutional Neural Networks, Advances in Neural Information Processing Systems* 25: 26th Annual Conference on Neural Information

Processing Systems 2012. Pro- ceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.

Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun, *Delving Deep into Rec- tifiers: Surpassing Human-Level Performance on ImageNet Classification*, CoRR, http://arxiv.org/abs/1502.01852, 2015.

Xavier Glorot and Yoshua Bengio, *Understanding the difficulty of training deep feedfor- ward neural networks, Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010.

Karen Simonyan and Andrew Zisserman, *Very Deep Convolutional Networks for Large- Scale Image Recognition*, In ICLR, 2015.

Adam Coates, Andrew Y. Ng and Honglak Lee, *An Analysis of Single-Layer Networks in Unsupervised Feature Learning, Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics,* AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011.

Lee, Dong-Hyun. *"Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks."* Workshop on Challenges in Representation Learning, ICML. Vol. 3. 2013.APA