Report

Anonymous Author(s)
Affiliation
Address
email

1 Softmax Regression via Gradient Descent

1.1 Problem definition

In this problem, we need to classify MNIST datasets using softmax regression. In the experiments, we only use the first 20,000 training images and the last 2,000 test images.

1.2 Methods

Derive the gradient for Softmax Regression:
The cross-entropy cost function can be expressed as,

$$E = -\sum_{n}\sum_{k=1}^{c} t_k^n \ln y_k^n \tag{1}$$

Where,

$$y_k^n = \frac{\exp\left(a_k^n\right)}{\sum_{k'} \exp\left(a_{k'}^n\right)} \tag{2}$$

And,

$$a_k^n = w_k^T x^n \tag{3}$$

We can calculate the gradient for softmax regression as follows,

$$\begin{aligned} -\frac{\partial E^n(w)}{\partial w_{jk}} &= -\frac{\partial E^n(w)}{\partial a_k^n}\frac{\partial a_k^n}{\partial w_{jk}} \\ &= -\sum_{k'}\frac{\partial E^n(w)}{\partial y_{k'}^n}\frac{\partial y_{k'}^n}{\partial a_k^n}\frac{\partial a_k^n}{\partial w_{jk}} \end{aligned} \tag{4}$$

And

$$\frac{\partial y_{k'}^n}{\partial a_k^n} = y_{k'}^n \delta_{kk'} - y_{k'} y_k \tag{5}$$

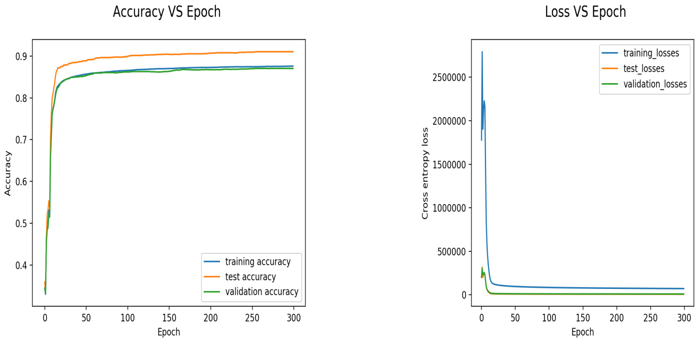
Where $\delta_{kk} = 1$ if $k = k'$, otherwise $\delta_{kk} = 0$. And

$$\frac{\partial E^n(w)}{\partial y_{k'}^n} = -\frac{t_{k'}}{y_{k'}} \tag{6}$$

Substitute Equation (5) and Equation (6) into Equation (4) we get,

$$-\frac{\partial E^n(w)}{\partial w_{jk}} = (t_k - y_k)x_j^n \tag{7}$$

Preprossing: First, we extract the first 20,000 training images and the last 2,000 test images. Then normailize the images to make sure the pixel values are in the range of [0,1]. Convert the labels to



(a) The percent correct over the number of training iterations for the training, hold-out and test set.

(b) The value of the loss function over the number of training iterations for the training, hold-out, and test set.

Figure 1: The performance of the algorithm over the number of training iterations.

one-hot vectors. Divide the training images into two parts, the first 10% are used for as a hold-out set and the rest 90% are used for training.
Experiments settings: We use an initial learning rate $\eta(0)$ = 0.004 and use equation $\eta(t) = \eta(0)/(1 + t/T)$ to anneal the learning rate by reducing it over time. t is used to index the epoch number and T is a metaparameter which is set to be 2. In the experiments, we find that above learning settings work best.
To determine the best type of regurization and the best λ, we try L_2 regularization and L_1 regularization seperately. For the L_2 regularizartion, we search the best λ in the set $\{0.01, 0.001, 0.0001\}$. If the accuracy on the hold-out set decreases for 3 epochs, we stop the algorithm and use the weights with the highest accuracy on the hold-out set as the final answer. For the L_1 regularization, we follow the same steps. We run the 1000 epochs for each setting. Then we compare the results got from these two regularization methods and use the best one as the final result.

1.3 Results

(a) In the experiments, we find that using L_2 regularization with $\lambda = 0.01$ obtain the best result on the validation set with an accuracy of 0.9045% on the validation set. The accuracy is 0.927% on the test set under such settings.

To further examine the performance of the algorithm, we try other λs. We choose λ in the set $\{0.05, 0.005, 0.0005\}$ and use L_2 regularization. The highest accuracy on the validation set is 0.8965% with $\lambda = 0.0005$. And the test accuracy is 0.933% which is slightly higher than when $\lambda = 0.01$. So in following experiments, we fix λ to be 0.0005.

(b) In this experiement, we use L_2 regularization with $\lambda = 0.005$. The figure is shown in Fig 1a.

(c) In this experiement, we use L_2 regularization with $\lambda = 0.005$. The figure is shown in Fig 1b.

(d) We plot the results in Fig 2. We can see that the image of the weight and the corresponding image of the average digit is almost the same. The reason is that we classify the images based on the inner product of the pixesls with the weights. And the inner product is maximized when the angle between the weight and the image is zero. So we see that the image of the weight and the corresponding image of the average digit is similar.

1.4 Discussion

Acknowledgments

.

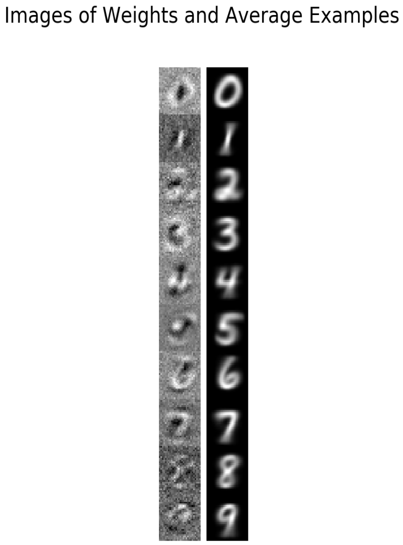


Figure 2

References