CS776 Computer Vision Assignment 1

1. Building model

1.1 Architecture of the models

1. MLP Model has a single hidden layer with 64 neurons. It takes a 512 size feature vector. On the hidden layer, The ReLU activation function is used.
2. Output layer has ten neurons, and softmax activation is used to get the probability of classified class.
3. This model used the categorical cross-entropy loss function to calculate the loss of the model.

1.2 Structure of the MLP code

1. Model function is main function to train model it takes all hyperparameter, data and labels. It initialize weights with initialize parameter function and call optimizer function to optimize this weights. It return traned weights in form of dictionary.
2. Optimizer function is called by model function. Mini batch gradient decent algorithm is used in optimizer. It update weights based on gradients return by propagation function.
3. propagation function perform forward and backward propagation. And return gradients and loss of model.

1.3 Training - learning rate, epochs used for training

1. Cifar-10 data set is used to train the Unaugmented and Augmented models. Four different transformations are used on the dataset to create augmented data used to train the augmented model.
2. Learning rate - Model is trained with 0.001 learning rate.
3. Epoch - 200 epochs for model trained on original dataset and 300 epoch for model trained on augmented dataset.
4. Batch Size - 128 batch size is used for model trained on original dataset and 64 for model trained on augmented dataset.

1.4 Evaluation metric

1. Categorical cross-entropy loss function to calculate the loss of the model.

2. How to run model

2.1 Folder structure

1. Project folder contain a) Jupiter notebook b) data folder c) feature extractory.py file.
2. The data folder contains original dataset, And augmented dataset and extracted feature are saved in data folder during running the code.

2.2 Instruction to run the model

Code can be run through python notebook. Packages need to run code are os, pickle, NumPy, random, OpenCV, matplotlib, torch, torchvision.

1. In section 1 of notebook function for image transformation is implemented. In 1.1 this method are demonstrated on image, change image var to try on diffrent images.
2. In section 2 augmented dataset is created and it is saved in data folder, next cell loaded the saved dataset from data folder.
3. In section 3 after feature extraction, extracted features are saved in data folder.
4. In section 5 model are trained on both datatset. For training model data is taken from file saved in feature extraction step.
5. In section 6 both mdel is evaluated on testset. Test data is from file saved after feature extraction. 6.0 have code to load test data.
6. To load weigths load_weights(file_name) method can be used. Weights file are named as "Weights_original", "Weights_augmented". Weights are saved at project folder.
7. To make prediction predict(W1,b1,W2,b2,X) function takes weights and batch of images after feature extraction in shape (batch_size, 512). And give predicted class labes in (batch_size,) shape.
8. In starting of notebook there is path var that have path for data folder.
9. Code can be directly run from middle taking data from saved file.

3. Derivation for back propagation

3.1 Equation for forward propagation

1. $Z^{[1]} = XW^{[1]} + b^{[1]}$
2. $A^{[1]} = ReLU(Z^{[1]})$
3. $Z^{[2]} = A^{[1]}W^{[2]} + b^{[2]}$
4. $A^{[2]} = Softmax(Z^{[2]})$
5. $L = Loss(A^{[2]}, Y) = \sum_j a_j^{[2]} log(y_j)$

3.2 Backpropagation

1. $\frac{\partial L}{\partial z_2} = \frac{\partial L}{\partial a_2}\frac{\partial a_2}{\partial z_2}$
2. $\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial z_2}\frac{\partial z_2}{\partial w_2}$
3. $\frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial z_2}\frac{\partial z_2}{\partial b_2}$
4. $\frac{\partial L}{\partial z_1} = \frac{\partial L}{\partial z_2}\frac{\partial z_2}{\partial a_1}\frac{\partial a_1}{\partial z_1}$
5. $\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial z_1}\frac{\partial z_2}{\partial w_1}$
6. $\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial z_1}\frac{\partial z_2}{\partial b_1}$

3.2.1 Derivation of loss function

$$\frac{\partial L(A^{[2]}, Y)}{\partial a_2} = \frac{\partial}{\partial a^{[2]}}\left[-\sum_j y_j log(a_j^{[2]})\right] = -\sum_j y_j \frac{\partial}{\partial a^{[2]}} log(a_j^{[2]})$$

$$\frac{\partial L}{\partial a_2} = -\sum_j \frac{y_j}{a_j^{[2]}}$$

3.2.2 Derivation of Softmax function

$$\frac{\partial a_j^{[2]}}{\partial z_k} = \frac{\partial}{\partial z_k}\left[\frac{e^{z_j}}{\sum_i e^{z_i}}\right] = \frac{\frac{\partial}{\partial z_k}e^{z_j}\sum_i e^{z_i} - e^{z_j}\frac{\partial}{\partial z_k}\sum_i e^{z_i}}{\left[\sum_i e^{z_i}\right]^2}$$

Now for j = k

$$= \frac{e^{z_j}\sum_i e^{z_i} - e^{z_j}e^{z_k}}{[\sum_i e^{z_i}]^2} = \left(\frac{e^{z_j}}{\sum_i e^{z_i}}\right)\left(\frac{\sum_i e^{z_i} - e^{z_k}}{\sum_i e^{z_i}}\right) = a_j^{[2]}.(1 - a_k^{[2]})$$

Now for j != k

$$= \frac{0.\sum_i e^{z_{i,j}} - e^{z_{i,j}}e^{z_{i,k}}}{[\sum_i e^{z_{i,j}}]^2} = \frac{-e^{z_{i,j}}e^{z_{i,k}}}{[\sum_i e^{z_{i,j}}]^2} = \left[\frac{e^{z_{i,j}}}{\sum_i e^{z_{i,j}}}\right]\left[\frac{e^{z_{i,k}}}{\sum_i e^{z_{i,j}}}\right] = -a_{i,j}^{[2]}.a_{i,k}^{[2]}$$

3.2.3 Derivation of loss with Z2

$$\frac{\partial L}{\partial z^{[2]}} = \frac{\partial L}{\partial a^{[2]}}\frac{\partial a^{[2]}}{\partial z^{[2]}}$$

$$= \frac{y_k}{a_k^{[2]}}.\frac{\partial a_k^{[2]}}{\partial z_k^{[2]}} - \sum_{j!=k}\frac{y_j}{a_j^{[2]}}.\frac{\partial a_j^{[2]}}{\partial z_k^{[2]}} = -y_k(1 - a_k^{[2]}) + \sum_{j!=k} = -y_k + y_k a_k^{[2]} + \sum_{j!=k} y_j a_k^{[2]} = a_k^{[2]} - y_k$$

3.2.4 Derivation of W2

$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial z_2}\frac{\partial z_2}{\partial w_2}$$

$$= dz^{[2]}\frac{\partial}{\partial w_2}\left(a^{[1]}w^{[2]} + b^{[2]}\right) = dz^{[2]}a^{[1]}$$

for matrix form

$$dW^{[2]} = \frac{A^{[1]T}dZ^{[2]}}{n}$$

3.2.5 Derivation of b2

$$\frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial z_2}\frac{\partial z_2}{\partial b_2}$$

$$= dz^{[2]}\frac{\partial}{\partial b_2}\left(a^{[1]}w^{[2]} + b^{[2]}\right) = dz^{[2]}$$

For Matrix form

$$db^{[2]} = \frac{\sum dZ^{[2]}}{n}$$

where n is number of examples

3.2.6 Derivation of Z1

$$\frac{\partial L}{\partial z_1} = \frac{\partial L}{\partial z_2}\frac{\partial z_2}{\partial a_1}\frac{\partial a_1}{\partial z_1}$$

$$= dz^{[2]}\frac{\partial}{\partial a_1}\left(a^{[1]}w^{[2]} + b^{[2]}\right)\frac{\partial}{\partial z_1}\left(ReLU(z^{[1]})\right) = dz^{[2]}w^{[2]}\frac{\partial}{\partial z_1}\left(ReLU(z^{[1]})\right)$$

For Matrix form

$$dZ^{[1]} = dZ^{[2]}.W^{[2]T}\frac{\partial}{\partial z_1}\left(ReLU(z^{[1]})\right)$$

3.2.7 Derivation of W1

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial z_1}\frac{\partial z_2}{\partial w_1}$$

$$= dz^{[1]}\frac{\partial}{\partial w_1}\left(xw^{[1]} + b^{[1]}\right) = dz^{[1]}x$$

for matrix form

$$dW^{[1]} = \frac{X^T dZ^{[1]}}{n}$$

where n is number of examples

3.2.8 Derivation of b1

$$\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial z_1}\frac{\partial z_2}{\partial b_1}$$

$$= dz^{[1]}\frac{\partial}{\partial b_2}\left(xw^{[1]} + b^{[1]}\right) = dz^{[1]}$$

For Matrix form

$$db^{[1]} = \frac{\sum dZ^{[1]}}{n}$$

where n is number of examples

3.3 Update weights and biases

1. $W^{[1]} = W^{[1]} - \alpha * dW^{[1]}$
2. $b^{[1]} = b^{[1]} - \alpha * db^{[1]}$
3. $W^{[2]} = W^{[2]} - \alpha * dW^{[2]}$
4. $b^{[2]} = b^{[2]} - \alpha * db^{[2]}$

4. Observation

1. Model trained on augmeted dataset gives little less accuracy as compared to model trained on original dataset when all hyperparameter are same.
2. When batch size is small there is high varience in accuracy of the model but model train faster. With large batch size model train slowly but there is less varience in accuracy while training.
3. Keeping very small learning rate train model slowly.

> Note : Due to upload file size limit in helloiitk, In data folder I have uploded original data file but was able to upload other files (augmented data file, feature extracted file) genrated by code. I am given drive link for these files, as data augmentation and feature extraction takes long time can use these file to run code from middle - https://drive.google.com/file/d/18ha9vGD8YbstImqWWctddlglWdwEM-xf/view