CSE616 : Neural Networks and Applications

Assignment 2

Name: Robeir Samir George

ID: 1901823

1. Consider an input image of shape 500x500x3. The image is flattened and a fully connected layer with 100 hidden units is used. What is the shape of the weight matrix of this layer (without the bias)? What is the shape of the bias.

Solution:

Flattening 500x500x3 in 1D, so dim of (x) is 750,000x1.

Shape of Weight Matrix is **100x750,000**

Shape of the bias is **100x1**

---

2. You run this image in a convolutional layer with 10 filters, of kernel size 5x5. How many parameters does this layer have?

Solution:

10 filters of size (5x5)

Input size = 500x500x3

So, number of parameters in this layer = 10 * (5*5*3) "without biases" = **750**

Number of parameters in this layer = 10 * (5*5*3) + 10 "with biases" = **760**

---

3. The top grey image has run through different types of filters and the results are shown in the following images. What type of convolutional filter was used to get each of the resulting images. Explain briefly and include the values of these filters. The filters have a shape of (3,3).

Solution:

Edge Detection Filter is used (known as: Prewitt Filter).

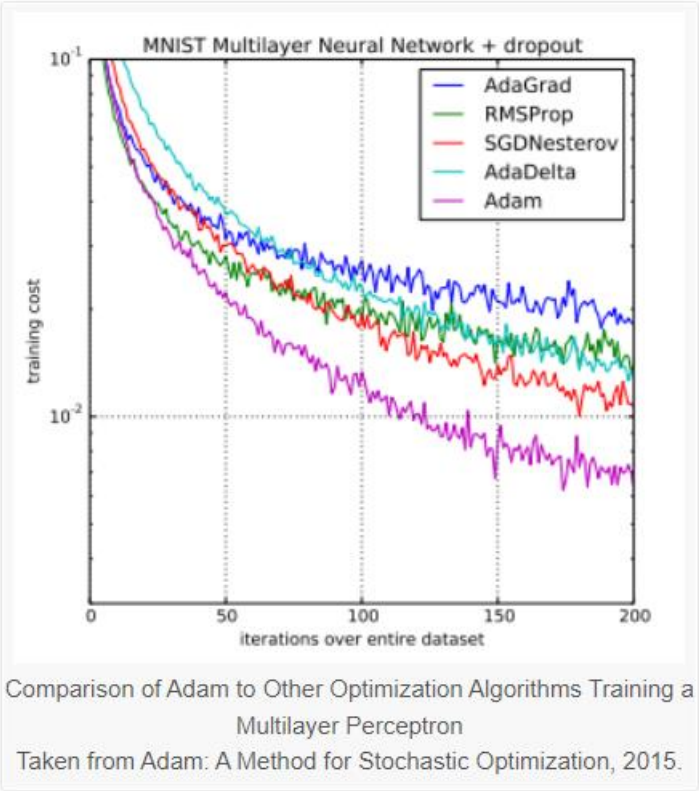
To get the image on the left we use vertical edge detection convolutional filter with values:

-1	**0**	**+1**
-1	**0**	**+1**
-1	**0**	**+1**

To get the image on the right we use horizontal edge detection convolutional filter with values:

-1	**-1**	**-1**
0	**0**	**0**
+1	**+1**	**+1**

4. In the Adam optimizer. Show what will happen the numbers of steps to compute the exponential moving averages gets large.



Comparison of Adam to Other Optimization Algorithms Training a Multilayer Perceptron
Taken from Adam: A Method for Stochastic Optimization, 2015.

It is quite clear from the previous graph that as the number of steps increases, the error of training decreases. This means that the learning rate is quickly adapting to the training instead of using certain value as in the classical gradient descent. Also, Adam shows significantly better results than other optimization algorithms “shown in the previous graph”.

Adam realizes the benefits of both AdaGrad and RMSProp.

Instead of adapting the parameter learning rates based on the average first moment (the mean) as in RMSProp, Adam also makes use of the average of the second moments of the gradients (the uncentered variance).

Specifically, the algorithm calculates an exponential moving average of the gradient and the squared gradient, and the parameters beta1 and beta2 control the decay rates of these moving averages.

5. Given a batch of size m, and assume that a batch normalization layer takes an input z = (z (1) , … , z (m)) as an input. Write down the output equation(s) of this layer. Give two reasons for using the batch normalization layer.

We take the average value out of each layer called **μB**. This is called calculated as the sum of all values of layer **x[i]** divided by average on all **m** values.

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i$$

We then calculate the variance σ²B as follows:
1. Subtracting the μB from every value which is the deviation of every value and the square for squared deviation
2. Sum up the results of doing that for each of the values, then divide by the number of values **m**, to get the average, or mean squared deviation.

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{1}(x_i - \mu_B)^2$$

We then find the standard deviation as the sum of mean squared deviation and epsilon under root. The epsilon is a constant value as small as 0.001. This is added to avoid cases of division by zero and also to increase variance.

Now that we calculated mean and standard deviation, we can normalize as follows.

$$\hat{x_i} = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

Batch normalization scales layers outputs to have mean 0 and variance 1. The outputs are scaled such a way to train the network faster. It also reduces problems due to poor parameter initialization.

Source: What is batch normalization?. How does it help? | by NVS Yashwanth | Towards Data Science

6. Suppose you have a convolutional network with the following architecture: The input is an RGB image of size 256 x 256. The first layer is a convolution layer with 32 feature maps and filters of size 3x3. It uses a stride of 1, so it has the same width and height as the original image. _ The next layer is a pooling layer with a stride of 2 (so it reduces the size of each dimension by a factor of 2) and pooling groups of size 3 x 3. Determine the size of the receptive field for a single unit in the pooling layer. (i.e., determine the size of the region of the input image which influences the activation of that unit.) You may assume the receptive field lies entirely within the image.

Solution:

Since input is 256x256x3, passing it through conv layer of stride = 1, produces an image of the same dimensions 256x256x3.

Passing it through pooling of stride s = 2, so, it reduces the dimensions to the half. It becomes of 128x128x3 dimensions.

So, each single unit in the pooling layer gets its activation from a 2x2 pixels in the original image.

Size of receptive field is 2x2 for a single unit in the pooling layer.

---

7. If an input data block in a convolutional network has dimension C×H×W = 96 ×128 ×128 , (96 channels, spatial dim 128x128) and we apply a convolutional filter to it of dimension D × C × HF × WF = 128×96 ×7×7 , (i.e. a block of D=128 filters) with stride 2 and pad 3,what is the dimension of the output data block?

Solution:

Image Height = 128, Image Width = 128, number of channels = 96.

Filter Height = 7, Filter Width = 7, number of channels = 96.

Number of filters = 128.

Stride S = 2

Padding p = 3.

After padding size of Image is (128+2p) x (128+2p) = (134x134) x 96

It outputs: (134-f+1) x (134-f+1) = (128)x(128)x96 image

We have stride S = 2, so the final output is equal to (128/2)x(128/2)x(96 channels)

So, output is CxHxW = 96x64x64

---

8. What is inverted dropout and what is its advantage?

Solution:

Inverted dropout is a variant of the original dropout technique developed by Hinton et al.

Just like traditional dropout, inverted dropout randomly keeps some weights and sets others to zero.

The one difference is that, during the training of a neural network, inverted dropout scales the activations by the inverse of the keep probability $q=1-p$.

This prevents network's activations from getting too large and does not require any changes to the network during evaluation.

In contrast, traditional dropout requires scaling to be implemented during the test phase.

---

9. Explain briefly why fully connected neural networks do not work well for image classification.

Solution:

First let's look at the similarities. Both convolution neural networks and neural networks have learnable weights and biases. In both networks the neurons receive some input, perform a dot product and follows it up with a non-linear function like ReLU and tanh.

Main problem with fully connected layer:

When it comes to classifying images — let's say with size 64x64x3 — fully connected layers need 12288 weights in the first hidden layer only! Which is a very huge number. The number of weights will be even bigger for images with size 225x225x3 = 151875. Networks having large number of parameters face several problems, for example: slower training time, chances of overfitting, …

Advantages of CNNs over the fully connected networks can be discussed in the solution of question 12.

---

10. Compute the convolution of the following two arrays: (4 1 -1 3) * (-2 1). Your answer should be an array of length 5. Show your detailed work.

Solution:

Assuming the first array is called A, the second array is called B. The resultant array C.

$C[i + j] = \sum_{for\ every\ i\ and\ j} A[i] * B[j]$

So,

C[0] = A[0]*B[0] = 4*-2 = -8

C[1] = A[0]*B[1] + A[1]*B[0] = 4*1 + 1*-2 = 2

C[2] = A[1]*B[1] + A[2]*B[0] = 1*-2 + -1*-2 = 0

C[3] = A[2]*B[1] + A[3]*B[0] = -1*1 + 3*-2 = -7

C[4] = A[3]*B[1] = 3*1 = 3

So, the convolution result is (-8 2 0 -7 3)

---

11. Describe what setting change in epochs 25 and 60 could have produced this training curve. Be brief.

Solution:

I think, changing the learning rate could have produced this change. Because Sudden changes occurred which can be done through changing the learning rate of the training process.

---

12. Why are convolutional layers more commonly used than fully-connected layers for image processing?

Solution:

The main reason is the number of parameters. The number of parameters in a neural network grows rapidly with the increase in the number of layers. This can make training for a model computationally heavy (and sometimes not feasible). Tuning so many of parameters can be a very huge task. The time taken for tuning these parameters is diminished by CNNs. CNNs are very effective in reducing the number of parameters without losing on the quality of models. Images have high dimensionality (as each pixel is considered as a feature).

Dimensionality reduction is achieved using a sliding window with a size less than that of the input matrix. This square patch is the window which keeps shifting left to right and top to bottom to cover the complete image.

All the layers of a CNN have multiple convolutional filters working and scanning the complete feature matrix and carry out the dimensionality reduction. This enables CNN to be a very apt and fit network for image classifications and processing.

---

13. Dropout layers implement different forward functions at train and test time. Explain what they do. Let p be the probability that node value is retained.

Solution:

Dropout is a regularization method that approximates training a large number of neural networks with different architectures in parallel.

During training, some number of layer outputs are randomly ignored or "*dropped out*." This has the effect of making the layer look-like and be treated-like a layer with a different number of nodes and connectivity to the prior layer. In effect, each update to a layer during training is performed with a different "*view*" of the configured layer.
Dropout simulates a sparse activation from a given layer, which interestingly, in turn, encourages the network to actually learn a sparse representation as a side-effect. As such, it may be used as an alternative to activity regularization for encouraging sparse representations in autoencoder models.

Dropout is not used after training when making a prediction with the fit network.

The weights of the network will be larger than normal because of dropout. Therefore, before finalizing the network, the weights are first scaled by the chosen dropout rate. The network can then be used as per normal to make predictions.

If a unit is retained with probability p during training, the outgoing weights of that unit are multiplied by p at test time

Source: A Gentle Introduction to Dropout for Regularizing Deep Neural Networks (machinelearningmastery.com)

---

14. Explain how standard momentum and Nesterov accelerated gradient differ. How does their performance (convergence as a function of time) differ on convex optimization problems? Use sketches as an aid.

Solution:

Nesterov Momentum is an extension to the gradient descent optimization algorithm.
Nesterov Momentum is just like more traditional momentum except the update is performed using the partial derivative of the projected update rather than the derivative current variable value.
Traditional momentum involves maintaining an additional variable that represents the last update performed to the variable, an exponentially decaying moving average of past gradients.

Momentum can be interpreted as a ball rolling down a nearly horizontal incline. The ball naturally gathers momentum as gravity causes it to accelerate, just as the gradient causes momentum to accumulate in this descent method.

— Page 75, Algorithms for Optimization, 2019.

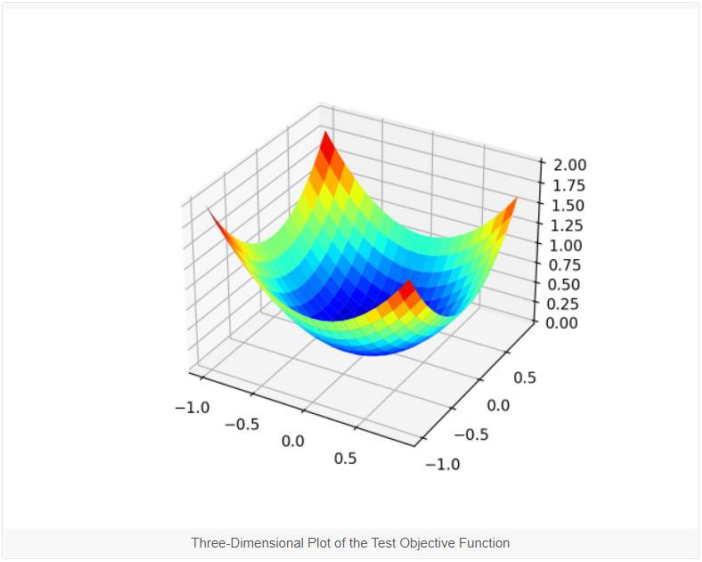
A problem with momentum is that acceleration can sometimes cause the search to overshoot the minima at the bottom of a basin or valley floor.

Nesterov Momentum can be thought of as a modification to momentum to overcome this problem of overshooting the minima.

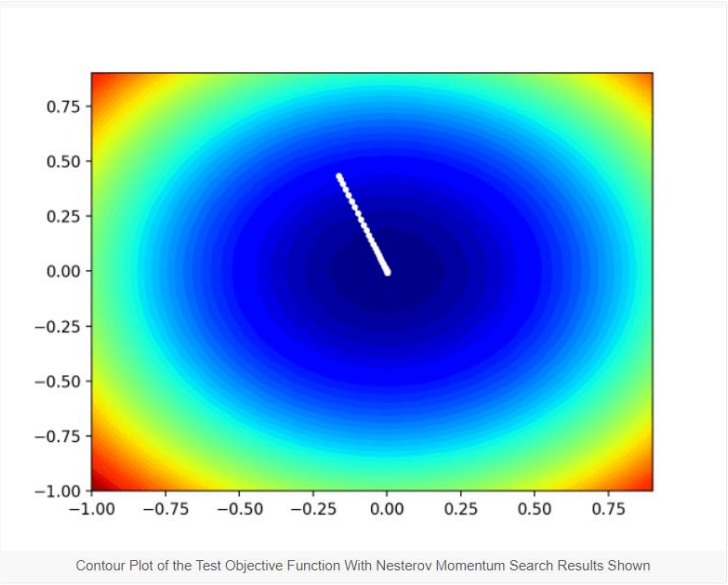
It involves first calculating the projected position of the variable using the change from the last iteration and using the derivative of the projected position in the calculation of the new position for the variable.

Calculating the gradient of the projected position acts like a correction factor for the acceleration that has been accumulated.

With Nesterov momentum the gradient is evaluated after the current velocity is applied. Thus one can interpret Nesterov momentum as attempting to add a correction factor to the standard method of momentum.



Three-Dimensional Plot of the Test Objective Function



Contour Plot of the Test Objective Function With Nesterov Momentum Search Results Shown

15. How does the actual learning rate of ADAGRAD change with the number of steps?

Solution:

$$\Delta w(t) = -\frac{\eta}{\sqrt{v(t) + \epsilon}} * \delta(t)$$

The above equation is the weight update equation for **AdaGrad**

The new learning rate for **AdaGrad** decays by a factor of the squared sum of the past gradients after each iteration. Although it solves our problem of updating the sparse features, it at the same time introduces a new one. For dense features, the past gradients will be non zero and after some iterations, the learning rate shrinks too rapidly due to the accumulation of all past squared gradient in its denominator.

End Of Assignment 2

Thank You

May 6, 2022