EECS 182 Deep Neural Networks
Fall 2022 Anant Sahai

Homework 3

This homework is due on Saturday, September 24, 2022, at 10:59PM.

1. Accelerating Gradient Descent with momentum

Consider the problem finding the minimizer of the following objective:

$$\mathcal{L}(w) = ||y - Xw||_2^2 \tag{1}$$

In the previous homework, we proved that gradient descent (GD) algorithm can converge and derive the convergence rate. In this homework, we will add the momentum term and how it affects to the convergence rate. The optimization procedure of gradient descent+momentum is given below:

$$\begin{aligned} w_{t+1} &= w_t - \eta z_{t+1} \\ z_{t+1} &= (1-\beta) z_t + \beta g_t, \end{aligned} \tag{2}$$

where $g_t = \nabla \mathcal{L}(w_t)$, η is learning rate and β defines how much averaging we want for the gradient. Note that when $\beta = 1$, the above procedure is just original gradient descent.

Let's investigate the effect of this change. We'll see that this modification can actually 'accelerate' the convergence by allowing larger learning rates.

(a) Recall that the gradient descent update of 1 is

$$w_{t+1} = \left(I - 2\eta(X^T X)\right) w_t + 2\eta X^T y \tag{3}$$

and the minimizer is

$$w^* = (X^T X)^{-1} X^T y \tag{4}$$

The geometric convergence rate (in the sense of what base is there for convergence as rate^t) of this procedure is

$$\text{rate} = \max_i |1 - 2\eta\sigma_i^2| \tag{5}$$

You saw on the last homework that if we choose the learning rate that maximizes Eq. 5, the optimal learning rate, η^* is

$$\eta^* = \frac{1}{\sigma_{\text{min}}^2 + \sigma_{\text{max}}^2}, \tag{6}$$

where σ_{max} and σ_{min} are the maximum and minimum singular value of the matrix X. The correspond-

ing optimal convergence rate is

$$\text{optimal rate} = \frac{(\sigma_{\max}/\sigma_{\min})^2 - 1}{(\sigma_{\max}/\sigma_{\min})^2 + 1} \tag{7}$$

Therefore, how fast ordinary gradient descent converges is determined by the ratio between the maximum singular value and the minimum singular value as above.

Now, let's consider using momentum to smooth the gradients before taking a step in Eq.2.

$$\begin{aligned} w_{t+1} &= w_t - \eta z_{t+1} \\ z_{t+1} &= (1-\beta) z_t + \beta(2X^T X w_t - 2X^T y) \end{aligned} \tag{8}$$

We can use the SVD of the matrix $X = U\Sigma V^T$, where $\Sigma = \text{diag}(\sigma_{\max}, \sigma_2, \ldots, \sigma_{\min})$ with the same (potentially rectangular) shape as X. This allows us to reparameterize the parameters w_t and averaged gradients z_t as below:

$$\begin{aligned} x_t &= V^T(w_t - w^*) \\ a_t &= V^T z_t. \end{aligned} \tag{9}$$

Please rewrite Eq. 8 with the reparameterized variables, $x_t[i]$ and $a_t[i]$. ($x_t[i]$ and $a_t[i]$ are i-th components of x_t and a_t respectively.)

Solution: Let's multiply V^T both sides in Eq.(8). Then,

$$\begin{aligned} V^T w_{t+1}[i] &= V^T w_t[i] - \eta V^T z_{t+1}[i] \\ x_{t+1}[i] + V^T w^*[i] &= x_t[i] + V^T w^*[i] - \eta a_{t+1}[i] \\ x_{t+1}[i] &= x_t[i] - \eta a_{t+1}[i] \end{aligned}$$

$$\begin{aligned} V^T z_{t+1} &= (1-\beta) V^T z_t + \beta V^T (2X^T X w_t - 2X^T y) \\ a_{t+1} &= (1-\beta) a_t + \beta V^T (2V\Sigma^2 V^T (V x_t + w^*) - 2V^T X^T y) \\ a_{t+1} &= (1-\beta) a_t + \beta (2\Sigma^2 V^T V x_t + \Sigma^2 V^T w^* - 2V^T X^T y) \end{aligned} \tag{10}$$

Note that,

$$w^* = (X^T X)^{-1} X^T y \tag{11}$$

$$= V\Sigma^{-2} V^T X^T y \tag{12}$$

Let's plug the above result into Eq.(10).

$$\begin{aligned} a_{t+1} &= (1-\beta) a_t + \beta (2\Sigma^2 V^T V x_t + 2\Sigma^2 V^T w^* - 2V^T X^T y) \\ &= (1-\beta) a_t + \beta (2\Sigma^2 V^T V x_t + 2\Sigma^2 V^T V \Sigma^{-2} V^T X^T y - 2V^T X^T y) \\ &= (1-\beta) a_t + \beta 2\Sigma^2 V^T V x_t \\ &= (1-\beta) a_t + \beta 2\Sigma^2 x_t \end{aligned}$$

If we express the above result element-wise,

$$a_{t+1}[i] = (1-\beta) a_t[i] + 2\beta\sigma_i^2 x_t[i]$$

$$x_{t+1}[i] = x_t[i] - \eta a_{t+1}[i]$$

(b) Notice that the above 2×2 vector/matrix recurrence has no external input. We can derive the 2×2 system matrix R_i from above such that

$$\begin{bmatrix} a_{t+1}[i] \\ x_{t+1}[i] \end{bmatrix} = R_i \begin{bmatrix} a_t[i] \\ x_t[i] \end{bmatrix} \tag{13}$$

Derive R_i.

Solution:

Since x_{t+1} is expressed with x_t and a_{t+1}, let's reformulate with x_t and a_t

$$\begin{aligned} x_{t+1}[i] &= x_t[i] - \eta a_{t+1}[i] \\ &= x_t[i] - \eta(1-\beta)a_t[i] - 2\eta\beta\sigma_i^2 x_t[i] \\ &= (1 - 2\eta\beta\sigma_i^2)x_t[i] - \eta(1-\beta)a_t[i] \end{aligned}$$

Therefore, the matrix R_i can be represented as below:

$$R_i = \begin{bmatrix} (1-\beta) & 2\beta\sigma_i^2 \\ -\eta(1-\beta) & 1 - 2\eta\beta\sigma_i^2 \end{bmatrix}$$

(c) **Use the computer to symbolically find the eigenvalues of the matrix R_i.**
When are they purely real? When are they repeated and purely real? When are they complex?

Solution: Let's derive the characteristic equation of R_i to derive eigenvalues:

$$\begin{aligned} f(\lambda) &= |R_i - \lambda I| && (14) \\ &= \left| \begin{bmatrix} (1-\beta) - \lambda & 2\beta\sigma_i^2 \\ -\eta(1-\beta) & 1 - 2\eta\beta\sigma_i^2 - \lambda \end{bmatrix} \right| \\ &= \lambda^2 - \left(2 - \beta - 2\eta\beta\sigma_i^2\right)\lambda + 1 - \beta && (15) \end{aligned}$$

Setting this to zero and solving, we get that the eigenvalues are:

$$\lambda_{1,i} = 1 - \beta\eta\sigma_i^2 - \frac{\beta}{2} - \frac{\sqrt{\beta\left(4\beta\eta^2\sigma_i^4 + 4\beta\eta\sigma_i^2 + \beta - 8\eta\sigma_i^2\right)}}{2}$$

$$\lambda_{2,i} = 1 - \beta\eta\sigma_i^2 - \frac{\beta}{2} + \frac{\sqrt{\beta\left(4\beta\eta^2\sigma_i^4 + 4\beta\eta\sigma_i^2 + \beta - 8\eta\sigma_i^2\right)}}{2}$$

The discriminant of Eq.14 is:

$$\begin{aligned} D &= \left(2 - \beta - 2\eta\beta\sigma_i^2\right)^2 - 4(1-\beta) && (16) \\ &= \beta\left(4\beta\eta^2\sigma_i^4 + 4\beta\eta\sigma_i^2 + \beta - 8\eta\sigma_i^2\right). && (17) \end{aligned}$$

To have distinctive and real eigenvalues, $D > 0$
To have repeated real eigenvalues, $D = 0$
To have complex eigenvalues, $D < 0$

(d) **For the case when they are repeated, what is the condition on η, β, σ_i that keeps them stable (strictly inside the unit circle)? What is the highest learning rate η as a function of β and σ_i that results in repeated eigenvalues?**

Solution: If eigenvalues are repeated or complex, this system is always stable as $0 < \beta < 1$. This result is very surprising in that the convergence does not depend neither on the learning rate nor on the singular values of the covariate matrix.
To find the maximum value, let's put the $D = 0$. Then, we can find two solutions:

$$\frac{2 - \beta - 2\sqrt{1-\beta}}{2\beta\sigma_i^2} \text{ or } \frac{2 - \beta + 2\sqrt{1-\beta}}{2\beta\sigma_i^2}$$

Therefore, the highest learning rate is $\frac{2-\beta+2\sqrt{1-\beta}}{2\beta\sigma_i^2}$

(e) **For the case when the eigenvalues are real, what is the condition on η, β, σ_i that keeps them stable (strictly inside the unit circle)? What is the upper bound of learning rate? Express with β, σ_i**

Solution: At first, to have the different real roots, the discriminant Eq.(16), D should be strictly positive:

$$D = \beta\left(4\beta\eta^2\sigma_i^4 + 4\beta\eta\sigma_i^2 + \beta - 8\eta\sigma_i^2\right) > 0 \tag{18}$$

If we solve Eq.(18) with respect to η,

$$\eta < \frac{2 - \beta - 2\sqrt{1-\beta}}{2\beta\sigma_i^2} \text{ or } \eta > \frac{2 - \beta + 2\sqrt{1-\beta}}{2\beta\sigma_i^2} \tag{19}$$

Also, to guarantee that those two solutions are in the open iterval (0, 1), the characteristic equation in Eq.(14) should satisfy $f(1) > 0$ and $f(-1) > 0$. We can check this by drawing the graph of $f(\lambda)$. Let's first compute $f(1)$

$$\begin{aligned} f(1) &= 1 - \left(2 - \beta - 2\eta\beta\sigma_i^2\right) + 1 - \beta \\ &= 2\eta\beta\sigma_i^2 > 0 \end{aligned}$$

Therefore, $f(1)$ is always positive.
Now, let's look at $f(-1)$.

$$\begin{aligned} f(1) &= 1 + \left(2 - \beta - 2\eta\beta\sigma_i^2\right) + 1 - \beta \\ &= 4 - 2\beta - 2\eta\beta\sigma_i^2 > 0 \end{aligned}$$

Thus,

$$0 < \eta < \frac{4 - 2\beta}{2\beta\sigma_i^2} \tag{20}$$

Let's compare Eq.(19) and Eq.(20). Since

$$4 - 2\beta \geq 2 - \beta + 2\sqrt{1-\beta}, \text{for all} \beta \in [0, 1]$$

the condition that Eq.(14) has two different real roots is

$$0 < \eta < \frac{2 - \beta - 2\sqrt{1-\beta}}{2\beta\sigma_i^2}$$

(f) **For the case when the eigenvalues are complex, what is the condition on η, β, σ_i that keeps them stable (strictly inside the unit circle)? What is the highest learning rate η as a function of β and σ_i that results in complex eigenvalues?**

Solution: If eigenvalues are repeated or complex, this system is always stable as $0 < \beta < 1$. This result is very surprising in that the convergence does not depend neither on the learning rate nor on the singular values of the covariate matrix

To find the maximum value, let's put $D < 0$. Then, we can find two solutions:

$$\frac{2 - \beta - 2\sqrt{1-\beta}}{2\beta\sigma_i^2} < \eta < \frac{2 - \beta + 2\sqrt{1-\beta}}{2\beta\sigma_i^2}$$

Therefore, the highest learning rate is $\frac{2-\beta+2\sqrt{1-\beta}}{2\beta\sigma_i^2}$. From the questions (d) (f), the takeaway is that we have to choose η such that every eigenvalues of R_i are either the same real roots or complex roots. Because η can be much higher for those cases.

(g) Now, apply what you have learned to the following problem. Assume that $\beta = 0.1$ and we have a problem with two singular values $\sigma^2_{\text{max}} = 5$ and $\sigma^2_{\text{min}} = 0.05$. **What learning rate η should we choose to get the fastest convergence for gradient descent with momentum? Compare how many iterations it will take to get within 99.9% of the optimal solution (starting at 0) using this learning rate and momentum with what it would take using ordinary gradient descent.**

Solution: For OGD, referring Eq.(7) the optimal convergence rate is $R_1 = (5 - 0.05)/(5 + 0.05) = 0.98$. Therefore, the minimum number of iterations (T) to get within 99.9% of the optimal solution is:

$$||w_{T_1} - w^*||_2 = R_1^{T_1}||w_0 - w^*||_2 = (\frac{5 - 0.05}{5 + 0.05})^{T_1}||w^*|| \leq \epsilon||w^*||$$
$$T_1 = \lfloor \frac{\log 1000}{\log 505/495} \rfloor = 346$$

For GD+momentum, let's derive the optimal learning rate first. The optimal learning rate η^* and corresponding convergence rate R_2 are:

$$\eta^* = \underset{\eta}{\text{argmin}} \max\left\{ \left\| \begin{bmatrix} 1-\beta & 2\beta\sigma^2_{\text{max}} \\ -\eta(1-\beta) & 1 - 2\eta\beta\sigma^2_{\text{max}} \end{bmatrix} \right\|, \left\| \begin{bmatrix} 1-\beta & 2\beta\sigma^2_{\text{min}} \\ -\eta(1-\beta) & 1 - 2\eta\beta\sigma^2_{\text{min}} \end{bmatrix} \right\| \right\}$$
$$= \underset{\eta}{\text{argmin}} \max\{|\lambda_{1,\text{max}}|, |\lambda_{2,\text{max}}|, |\lambda_{1,\text{min}}|, |\lambda_{2,\text{min}}|\}$$
$$R_2 = \min_{\eta} \max\{|\lambda_{1,\text{max}}|, |\lambda_{2,\text{max}}|, |\lambda_{1,\text{min}}|, |\lambda_{2,\text{min}}|\}$$

, where $\|\cdot\|$ is the maximum of absolute value of eigenvalues and $\lambda_{i,\text{max}}, \lambda_{i,\text{min}}$. are eigenvalues of matrices.

As you can see the above objective is seemingly very difficult to solve. But we can get some hints from the previous parts. First, Let's look at Eq.(14). The constant term is $1-\beta$, which is the product of two solutions. If one of solutions is λ, the other one is $\frac{1-\beta}{\lambda}$. In other words, if $\lambda < \sqrt{1-\beta}$, the other solution is larger than $\sqrt{1-\beta}$. Then R_2 cannot be smaller than $\sqrt{1-\beta}$ Therefore R_2 has the lower bound, $R_2 \geq \sqrt{1-\beta}$.

Then, let's hypothesize that we can achieve that lower bound. Remind that if two solutions are repeated real or complex, their absolute values are always $\sqrt{1-\beta}$. So to prove the hypothesis, we need to show that eigenvalues of each matrix are either repeated real or complex. In that case, Eq.(16) should be non-negative for both matrices.

$$\frac{2-\beta-2\sqrt{1-\beta}}{2\beta\sigma_{\max}^2} \leq \eta \leq \frac{2-\beta+2\sqrt{1-\beta}}{2\beta\sigma_{\max}^2}$$
$$\frac{2-\beta-2\sqrt{1-\beta}}{2\beta\sigma_{\min}^2} \leq \eta \leq \frac{2-\beta+2\sqrt{1-\beta}}{2\beta\sigma_{\min}^2}$$

Inserting β, σ_i^2, the above inequalities, we can get the following result:

$$0.002633 \leq \eta \leq 3.797$$
$$0.2633 \leq \eta \leq 379.7$$

The common interval that η satisfies both inequalities is

$$0.2633 \leq \eta \leq 3.797$$

If η is in that interval, the convergence rate is $\sqrt{1-\beta} = 0.949$

$$||w_{T_2} - w^*||_2 \leq R_2^{T_2}||w_0 - w^*||_2 = (0.949)^{T_2}||w^*|| \leq \epsilon||w^*||$$
$$T_2 \geq \lfloor \frac{\log 1/\epsilon}{\log 1/R_2} \rfloor = 132$$

We need at least 132 iterations to guarantee 0.1% error.

Note: if you find the convergence rate correctly $\sqrt{1-\beta}$ and derive any η in the interval [0.263, 0.380], then you can get the full credit for this part.

2. Understanding Convolution as Finite Impulse Response Filter

For the discrete time signal, the output of linear time invariant system is defined as:

$$y[n] = x[n] * h[n] = \sum_{i=-\infty}^{\infty} x[n-i] \cdot h[i] = \sum_{i=-\infty}^{\infty} x[i] \cdot h[n-i] \tag{21}$$

, where x is the input signal, h is impulse response (also referred to as the filter). Please note that the convolution operations is to 'flip and drag'. But for neural networks, we simply implement the convolutional layer without flipping and such operation is called correlation. Interestingly, in CNN those two operations are equivalent because filter weights are initialized and updated. Even though you implement 'true' convolution, you just ended up with getting the flipped kernel. In this question, we will follow the definition.

Now let's consider rectangular signal with the length of L (sometimes also called the "rect" for short, or, alternatively, the "boxcar" signal). This signal is defined as:

$$x(n) = \begin{cases} 1 & n = 0, 1, 2, ..., L-1 \\ 0 & \text{otherwise} \end{cases}$$

Here's an example plot for $L = 7$, with time indices shown from -2 to 8 (so some implicit zeros are shown):

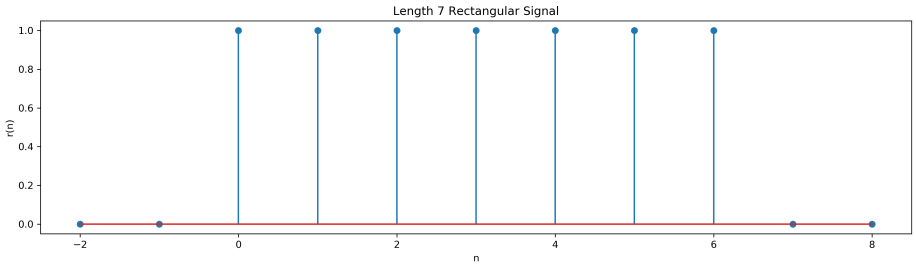

Figure 1: The rectangular signal with the length of 7

(a) The impulse response is define as:

$$h(n) = (\frac{1}{2})^n u(n) = \begin{cases} (\frac{1}{2})^n & n = 0, 1, 2, ... \\ 0 & \text{otherwise} \end{cases}$$

. **Compute and plot the convolution of $x(n)$ and $h(n)$.**

Solution: Let's divide this problem into three cases: 1) $n < 0$, 2) $0 \leq n < L-1$ and $n \geq L-1$. For the first case, $x(i) = 0$ and $h[n-i] = 0$ if $i < 0$, we only need to consider $i \geq 0$.

$$\begin{aligned} y(n) &= \sum_{i=-\infty}^{\infty} x[i] \cdot h[n-i] \\ &= \sum_{i=0}^{L-1} h[n-i] \end{aligned}$$

Here, $n-i$ is negative, $h(n-i)=0$,

$$y(n)=\sum_{i=0}^{L-1}h(n-i)=0$$

Consider the second case: $0\le n<L-1$. Since for $n-i<0$ $h(n)=0$, we only need to consider $i\le n$

$$\begin{aligned}y(n)&=\sum_{i=-\infty}^{\infty}x[i]\cdot h[n-i]\\&=\sum_{i=0}^{n}h[n-i]\\&=\sum_{i=0}^{n}\frac{1}{2^i}\\&=2-\frac{1}{2^n}\end{aligned}$$

For the last case, $x(n)=0$ when $n\ge L$. Therefore,

$$\begin{aligned}y(n)&=\sum_{i=-\infty}^{\infty}x[i]\cdot h[n-i]\\&=\sum_{i=0}^{L-1}h[n-i]\\&=\sum_{i=0}^{L-1}\frac{1}{2^i}\\&=\frac{2^L-1}{2^n}\end{aligned}$$

So $y(n)$ is

$$y(n)=\begin{cases}0 & n<0\\ 2-\frac{1}{2^n} & 0\le n<L-1\\ \frac{2^L-1}{2^n} & \text{otherwise}\end{cases}\tag{22}$$

(b) Now let's shift $x(n)$ by N, i.e. $x_2(n)=x(n-N)$. Let's put $N=5$ **Then, compute** $y_2(n)=h(n)*x_2(n)$**. Which property of the convolution can you find?**

Solution: The output is just shift the result of Eq. (22) by N. Therefore the output of this convolution is $y[n-N]=y[n-5]$. We can observe translational invariance (shift invariance / equivariance) property of the convolution operation.

Now, let's extend 1D to 2D. The example of 2D signal is the image. The operation of 2D convolution

is defined as follows:

$$y[m,n] = x[m,n] * h[m,n] = \sum_{i,j=-\infty}^{\infty} x[m-i, n-j] \cdot h[i,j] = \sum_{i,j=-\infty}^{\infty} x[i,j] \cdot h[m-i, n-j] \tag{23}$$

, where x is input signal, h is FIR filter and y is the output signal.

(c) 2D matrices, x and h are given like below:

$$x = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \end{bmatrix} \tag{24}$$

$$h = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{25}$$

Then, evaluate y. Assume that there is no pad and stride is 1.

Solution:

$$\begin{bmatrix} -40 & -40 & -40 \\ -40 & -40 & -40 \\ -40 & -40 & -40 \end{bmatrix}$$

(d) Now let's consider striding and padding. Evaluate y for following cases:

i. stride, pad = 1, 1

ii. stride, pad = 2, 1

Solution:
$$\begin{bmatrix} -19. & -28. & -32. & -36. & -29. \\ -30. & -40. & -40. & -40. & -30. \\ -30. & -40. & -40. & -40. & -30. \\ -30. & -40. & -40. & -40. & -30. \\ 49. & 68. & 72. & 76. & 59. \end{bmatrix}$$

$$\begin{bmatrix} -19. & -32. & -29. \\ -30. & -40. & -30. \\ 49. & 72. & 59. \end{bmatrix}$$

3. Feature Dimensions of Convolutional Neural Network In this problem, we compute output feature shape of convolutional layers and pooling layers, which are building blocks of CNN. Let's assume that input feature shape is $W \times H \times C$, where W is the width, H is the height and C is the number of channels of input feature.

(a) A convolutional layer has 4 hyperparameters: the filter size(K), the padding size (P), the stride step size (S) and the number of filters (F). How many weights and biases in this convolutional layer? And what is the shape of output feature that this convolutional layer produces?

Solution:

The number of weights $= K^2CF$
The number of biases $= F$
$W' = \lfloor (W - K + 2P)/S \rfloor + 1$
$H' = \lfloor (H - K + 2P)/S \rfloor + 1$
$C' = F$

(b) A pooling layer has 2 hyperparameters: the stride step size(S) and the filter size (K). What is the output feature shape that this pooling layer produces?

Solution:

$W' = (W - K)/S + 1$
$H' = (H - K)/S + 1$
$C' = C$

(c) Let's assume that we have the CNN model which consists of L successive convolutional layers and the filter size is K and the stride step size is 1 for every convolutional layer. Then what is the receptive field size?

Solution:

$L * (K - 1) + 1 = O(LK)$
Note that, the receptive field size increases linearly as we add more layers
We also accept $(L - 1) * (K - 1) + 1 = O(LK)$ as the answer.

(d) Consider a downsampling layer (e.g. pooling layer and strided convolution layer). In this problem, we investigate pros and cons of downsampling layer. This layer reduces the output feature resolution and this implies that the output features loose the certain amount of spatial information. Therefore when we design CNN, we usually increase the channel length to compensate this loss. For example, if we apply the max pooling layer with kernel size of 2 and stride size of 2, we increase the output feature size by a factor of 2. **If we apply this max pooling layer, how much the receptive field increases? Explain the advantage of decreasing the output feature resolution with the perspective of reducing the amount of computation.**

Solution: The receptive field size increases by a factor of 2. Since the spatial resolution decrease by 1/4 and the channel size increases by 2, the feature size decreases by a factor of 2. Therefore, the number of MAC operations after the pooling layer decreases.

4. Coding Question: BatchNorm

Look at the BatchNormalization.ipynb. In this notebook, you'll implement batch normalization layer. For this question, please submit a .zip file your completed work to the Gradescope assignment titled "HW 3 (Code)". No written portion.

(a) Implement forward operation of batch normalization layer.

(b) Implement backward operation of batch normalization layer.

(c) Implement Fully Connected Nets with Batch Normalization

5. Coding Question: Desigining 2D Filter

Look at the HandDesignFilters.ipynb. In this notebook, you'll design 2D image filter by hand. For this question, please submit a .zip file your completed work to the Gradescope assignment titled "HW 3 (Code)". No written portion.

(a) Design averaging filter.

(b) Design edge detection filter.

6. Coding Question: CNN

Look at the ConvolutionalNetworks.ipynb. In this notebook, you'll implement Convolutional Neural Networks. For this question, please submit a .zip file your completed work to the Gradescope assignment titled "HW 3 (Code)". No written portion.

(a) Implement forward operation of convolutional layer and max pooling layer.

(b) Implement Three-layer ConvNet

(c) Implement forward operation of spatial batch normalization layer.

7. Homework Process and Study Group

Citing sources and collaborators are an important part of life, including being a student!
We also want to understand what resources you find helpful and how much time homework is taking, so we can change things in the future if possible.

(a) **What sources (if any) did you use as you worked through the homework?**

(b) **If you worked with someone on this homework, who did you work with?**
List names and student ID's. (In case of homework party, you can also just describe the group.)

(c) **Roughly how many total hours did you work on this homework? Write it down here where you'll need to remember it for the self-grade form.**

Contributors:

- Suhong Moon.
- Gabriel Goh.
- Anant Sahai.

- Dominic Carrano.
- Babak Ayazifar.
- Sukrit Arora.
- Fei-Fei Li.
- Sheng Shen.
- Jake Austin.
- Kevin Li.

