

1.1.1 Minimum Norm Solution [1pt]

Recall Question 3.3 from Homework 1. For an overparameterized linear model, gradient descent starting from zero initialization finds the unique minimum norm solution \mathbf{w}^* such that $X\mathbf{w}^* = \mathbf{t}$. Let $\mathbf{w}_0 = \mathbf{0}$, $d > n$. Assume SGD also converges to a solution $\hat{\mathbf{w}}$ such that $X\hat{\mathbf{w}} = \mathbf{t}$. Show that SGD solution is identical to the minimum norm solution \mathbf{w}^* obtained by gradient descent, i.e., $\hat{\mathbf{w}} = \mathbf{w}^*$.

Hint: Is \mathbf{x}_i contained in span of X ? Do the update steps of SGD ever leave the span of X ?

$$\text{1.1.1 } \nabla_{w_t} L_i = \nabla_{w_t} \left(\frac{1}{n} \| \mathbf{x}_i^\top \mathbf{w}_t - t_i \|^2 \right) = \mathbf{x}_i \frac{1}{n} \cdot 2 (\mathbf{x}_i^\top \mathbf{w}_t - t_i) \\ = \frac{2}{n} \mathbf{x}_i (\mathbf{x}_i^\top \mathbf{w}_t - t_i) \quad \begin{matrix} \text{defined as row} \\ \uparrow \quad \text{space} \end{matrix}$$

As we know \mathbf{x}_i is the i^{th} row of \mathbf{X} , thus $\mathbf{x}_i \in \text{span}\{\mathbf{X}\}$.

Then $\nabla_{w_t} L_i = \frac{2}{n} \mathbf{x}_i (\underbrace{\mathbf{x}_i^\top \mathbf{w}_t - t_i}_{\in \mathbb{R}})$ is a constant multiple of \mathbf{x}_i , thus also in $\text{span}\{\mathbf{X}\}$.

Then, (1) $\mathbf{w}_0 = \mathbf{0} \in \text{span}\{\mathbf{X}^\top\}$

(2) if $\mathbf{w}_k \in \text{span}\{\mathbf{X}^\top\}$, then $\mathbf{w}_{k+1} = \mathbf{w}_k - \nabla_{w_t} L_i \in \text{span}\{\mathbf{X}^\top\}$

so if there is a convergent solution $\hat{\mathbf{w}}$, it must also be in the span of \mathbf{X}^\top .

Let $\hat{\mathbf{w}} = \mathbf{X}^\top \hat{\mathbf{c}}$, then $\mathbf{X}\hat{\mathbf{w}} = \mathbf{t}$ gives: $\mathbf{X}\mathbf{X}^\top \hat{\mathbf{c}} = \mathbf{t}$

$$\Rightarrow \hat{\mathbf{c}} = (\mathbf{X}\mathbf{X}^\top)^{-1} \mathbf{t} \Rightarrow \hat{\mathbf{w}} = \mathbf{X}^\top \hat{\mathbf{c}} = \mathbf{X}^\top (\mathbf{X}\mathbf{X}^\top)^{-1} \mathbf{t}$$

we then need to prove that $\hat{\mathbf{w}}$ is the minimum norm solution.

Consider any feasible vector \mathbf{w} s.t. $\mathbf{X}\mathbf{w} = \mathbf{t}$.

$$\text{We have: } \hat{\mathbf{w}}^\top (\hat{\mathbf{w}} - \mathbf{w}) = (\mathbf{X}^\top \hat{\mathbf{c}})^\top (\hat{\mathbf{w}} - \mathbf{w}) = \hat{\mathbf{c}}^\top (\mathbf{X}\hat{\mathbf{w}} - \mathbf{X}\mathbf{w}) = \hat{\mathbf{c}}^\top (\mathbf{t} - \mathbf{t}) = 0$$

This means that $\hat{\mathbf{w}}^\top$ is orthogonal to $\hat{\mathbf{w}} - \mathbf{w}$.

$$\text{Thus, } \|\hat{\mathbf{w}}\|^2 + \|\hat{\mathbf{w}} - \mathbf{w}\|^2 = \|\mathbf{w}\|^2$$

$$\text{so } \|\hat{\mathbf{w}}\|^2 \leq \|\mathbf{w}\|^2 \text{ meaning } \hat{\mathbf{w}} \text{ is the min norm sol}$$

1.1.2 SGD with Momentum [1pt]

As a corollary to Question 1.1.1 consider SGD with momentum. The update step at time t with scalars α and η is described as:

$$\delta_{t+1} = -\eta \nabla \mathcal{L}_i(\mathbf{x}_i) + \alpha \delta_t \quad (3)$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \delta_{t+1} \quad (4)$$

$\delta_0 = 0$

Under the assumptions in Question 1.1.1, provide an intuitive argument regarding whether stochastic gradient descent with momentum obtains the minimum norm solution on convergence.

Hint: Consider the effects of the momentum term on the linearity of gradient update.

$$(1) \delta_0 = 0 \in \text{span}\{\mathbf{x}^T\}$$

$$(2) \text{ If } \delta_k \in \text{span}\{\mathbf{x}^T\}, \text{ since } \nabla \mathcal{L}_i(\mathbf{x}_i) \in \text{span}\{\mathbf{x}^T\}, \text{ then}$$

$$\delta_{k+1} = -\eta \nabla \mathcal{L}_i(\mathbf{x}_i) + \alpha \delta_k \in \text{span}\{\mathbf{x}^T\}.$$

$$\text{Thus, by induction } \delta_t \in \text{span}\{\mathbf{x}^T\} \forall t.$$

We then use the same argument as in 1.1.1 to show that

$\mathbf{w}_t \in \text{span}\{\mathbf{x}^T\} \forall t$ and so \mathbf{w} converges to the minimum norm solution.

Intuitively, the momentum term doesn't affect linearity of gradient update.

1.2 Adaptive Methods

We will next consider the behavior of adaptive gradient descent methods. In particular, we will investigate the Adagrad method. Let w_i denote the i -th parameter. A scalar learning rate η is used. At time t for parameter i , the update step for Adagrad is shown by:

$$w_{i,t+1} = w_{i,t} - \frac{\eta}{\sqrt{G_{i,t} + \epsilon}} \nabla_{w_{i,t}} \mathcal{L}(w_{i,t}) \quad Gspan\{x^T\} \quad (5)$$

$$G_{i,t} = G_{i,t-1} + (\nabla_{w_{i,t}} \mathcal{L}(w_{i,t}))^2 \quad (6)$$

The term ϵ is a fixed small scalar used for numerical stability. Intuitively, Adagrad can be thought of as adapting the learning rate in each dimension to efficiently move through badly formed curvatures (see lecture slides/notes).

1.2.1 Minimum Norm Solution [1pt]

Consider the overparameterized linear model ($d > n$) for the loss function defined in Section I. Assume the Adagrad optimizer converges to a solution. Provide a proof or counterexample for whether Adagrad always obtains the minimum norm solution.

Hint: Compute the 2D case from HW1. Let $\mathbf{x}_1 = [1, 2]$, $w_0 = [0, 0]$, $t = [3]$.

1.2.1 Consider $x_1 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$, $w_0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, $t = [3]$. Then $\eta=1$, $d=2$. Let $\epsilon=0.1$. $\eta=0.01$.

$$L_i(x_i) = \nabla_{w_i} \left(\frac{1}{n} \|x_i^T w_i - t_i\|^2 \right)$$

$$\nabla L_i(w_i) = \frac{2}{\eta} x_i (x_i^T w_i - t_i) = 2 \begin{pmatrix} 1 \\ 2 \end{pmatrix} (0 - 3) = \begin{pmatrix} -6 \\ -12 \end{pmatrix}.$$

$$G_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 36 \\ 144 \end{pmatrix} = \begin{pmatrix} 36 \\ 144 \end{pmatrix}.$$

$$w_1 = w_0 - 0.01 \times \begin{pmatrix} \frac{1}{6+0.1} \cdot (-6) \\ \frac{1}{12+0.1} \cdot (-12) \end{pmatrix} = \begin{pmatrix} 0.009836 \\ 0.009917 \end{pmatrix}$$

However, this solution is not in the span of $\left\{ \begin{pmatrix} 1 \\ 2 \end{pmatrix} \right\}$, thus it's not a minimum norm sol.

2 Gradient-based Hyper-parameter Optimization

In this problem, we will implement a simple toy example of *gradient-based hyper-parameter optimization*, introduced in Lecture 3 (slides 14).

Often in practice, hyper-parameters are chosen by trial-and-error based on a model evaluation criterion. Instead, *gradient-based hyper-parameter optimization* computes gradient of the evaluation criterion w.r.t. the hyper-parameters and uses this gradient to directly optimize for the best set of hyper-parameters. For this problem, we will optimize for the learning rate of gradient descent in a linear regression problem, like in homework 1.

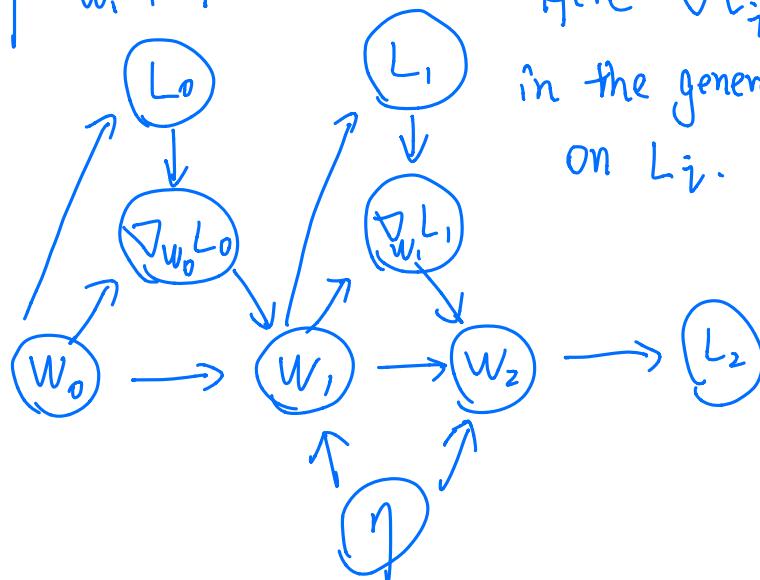
Similar to homework 1, a linear model will be used for this problem. Specifically, given n pairs of input data with d features and scalar label $(\mathbf{x}_i, t_i) \in \mathbb{R}^d \times \mathbb{R}$, we wish to find a linear model

2.1.1 [0.5pt]

Consider a case of 2 GD iterations. Draw the computation graph to obtain the final loss \mathcal{L}_2 in terms of $\mathbf{w}_0, \mathcal{L}_0, \nabla_{\mathbf{w}_0} \mathcal{L}_0, \mathbf{w}_1, \mathcal{L}_1, \nabla_{\mathbf{w}_1} \mathcal{L}_1, \mathbf{w}_2$, and η .

$$\left. \begin{aligned} \mathbf{w}_1 &= \mathbf{w}_0 - \eta \cdot \nabla_{\mathbf{w}_0} \mathcal{L}_0 \\ \mathbf{w}_2 &= \mathbf{w}_1 - \eta \cdot \nabla_{\mathbf{w}_1} \mathcal{L}_1 \end{aligned} \right\} \Rightarrow \mathcal{L}_2 = \frac{1}{n} \| \mathbf{x} \mathbf{w}_2 - t \|_2^2$$

Here $\nabla_{\mathbf{w}_i} \mathcal{L}_i$ depends only on \mathbf{w}_i , but in the general case, it also depends on \mathcal{L}_i .



2.1.2 [1pt]

Then, consider a case of t iterations of GD. What is the memory complexity for the forward-propagation in terms of t ? What is the memory complexity for using the standard back-propagation to compute the gradient w.r.t. the learning rate, $\nabla_\eta \mathcal{L}_t$ in terms of t ?

In forward propagation process, we only need to store one weight (w_i) at a time, so the complexity is $O(1)$. In back-propagation process, since w_1, w_2, \dots, w_t all depend on η , then we need to store $\nabla_\eta w_1, \nabla_\eta w_2, \dots, \nabla_\eta w_t$ at the same time, thus $O(t)$ memory is required.

2.2.1 [0pt]

Write down the expression of \mathbf{w}_1 in terms of \mathbf{w}_0 , η , \mathbf{t} and X . Then use the expression to derive the loss \mathcal{L}_1 in terms of η .

Hint: if the expression gets too messy, introduce a constant vector $\mathbf{a} = X\mathbf{w}_0 - \mathbf{t}$

$$\begin{aligned}\mathbf{w}_1 &= \mathbf{w}_0 - \eta \cdot \nabla_{\mathbf{w}_0} L_0(\mathbf{w}_0) \\ &= \mathbf{w}_0 - \eta \cdot \underbrace{\frac{2}{n}}_{\alpha} \cdot \mathbf{x}^T (\underbrace{\mathbf{x}\mathbf{w}_0 - \mathbf{t}}_{=\mathbf{a}}) \\ &= \overrightarrow{\mathbf{w}_0} - \alpha \cdot \mathbf{x}^T \overrightarrow{\mathbf{a}}\end{aligned}$$

$$\begin{aligned}\eta \cdot \mathcal{L}_1 &= \| \mathbf{x}\mathbf{w}_1 - \mathbf{t} \|^2 = \| \mathbf{x}(\overrightarrow{\mathbf{w}_0} - \alpha \mathbf{x}^T \overrightarrow{\mathbf{a}}) - \mathbf{t} \|^2 \\ &= \| \overrightarrow{\mathbf{w}_0} - \mathbf{t} - \alpha \mathbf{x} \mathbf{x}^T \overrightarrow{\mathbf{a}} \|^2 \\ &= \| \overrightarrow{\mathbf{a}} - \alpha \cdot \mathbf{x} \mathbf{x}^T \overrightarrow{\mathbf{a}} \|^2 \\ &= \| (\mathbf{I} - \alpha \mathbf{x} \mathbf{x}^T) \overrightarrow{\mathbf{a}} \|^2 \\ &= \| (\mathbf{I} - \eta \cdot \frac{2}{n} \mathbf{x} \mathbf{x}^T) \overrightarrow{\mathbf{a}} \|^2\end{aligned}$$

2.2.2 [1pt]

Determine if \mathcal{L}_1 is convex w.r.t. the learning rate η

Hint: a function is convex if its second order derivative is positive

$$\text{Let } M = -\frac{2}{n} \mathbf{x} \mathbf{x}^T.$$

$$\begin{aligned}\text{Then } \mathcal{L}_1 &= \| (\mathbf{I} + \eta M) \overrightarrow{\mathbf{a}} \|^2 = \| \overrightarrow{\mathbf{a}} + \eta M \overrightarrow{\mathbf{a}} \|^2 \\ &= (\| \overrightarrow{\mathbf{a}} \|^2 + \eta^2 \| M \overrightarrow{\mathbf{a}} \|^2 + 2\eta \overrightarrow{\mathbf{a}}^T M \overrightarrow{\mathbf{a}}) \cdot \frac{1}{n}\end{aligned}$$

$$\frac{\partial \mathcal{L}_1}{\partial \eta} = (2\eta \| M \overrightarrow{\mathbf{a}} \|^2 + 2\overrightarrow{\mathbf{a}}^T M \overrightarrow{\mathbf{a}}) \cdot \frac{1}{n}$$

$$\frac{\partial^2 \mathcal{L}_1}{\partial \eta^2} = \frac{2}{n} \| M \overrightarrow{\mathbf{a}} \|^2 > 0$$

2.2.3 [1pt]

Write down the derivative of \mathcal{L}_1 w.r.t. η and use it to find the optimal learning rate η^* that minimizes the loss after one GD iteration. Show your work.

$$\frac{\partial \mathcal{L}_1}{\partial \eta} = (2\eta \|Ma\|^2 + 2a^T Ma) \cdot \frac{1}{n} = 0$$

$$\Rightarrow \eta^* = -\frac{a^T Ma}{\|Ma\|^2} \quad \text{where } M = -\frac{2}{n} X^T, a = Xw_0 - b$$

This indeed is a minimum because \mathcal{L}_1 is convex.

2.3 Multiple Inner-loop Iterations

2.3.1 [0.5pt]

Derive the expression of the loss \mathcal{L}_t after t iterations of GD. Show your work.

Hint: proof by induction and binomial coefficients can be useful

Let $a_i = \mathbf{x}w_i - t$,



$$\begin{aligned}
 a_{k+1} &= \mathbf{x}w_{k+1} - t = \mathbf{x}(w_k - \eta \cdot \nabla_{w_k} L) - t \\
 &= \mathbf{x}\left(w_k - \frac{2\eta}{n} \mathbf{x}^T (\mathbf{x}w_k - t)\right) - t \\
 &= \mathbf{x}w_k - t - \frac{2\eta}{n} \mathbf{x} \mathbf{x}^T (\mathbf{x}w_k - t) \\
 &= \left(\mathbf{I} - \frac{2\eta}{n} \mathbf{x} \mathbf{x}^T\right) (\mathbf{x}w_k - t) \\
 &= \left(\mathbf{I} - \frac{2\eta}{n} \mathbf{x} \mathbf{x}^T\right) a_k
 \end{aligned}$$

$$\text{Thus, } a_m = \left(\mathbf{I} - \frac{2\eta}{n} \mathbf{x} \mathbf{x}^T\right)^m a_0 \quad \forall m \in \mathbb{N}$$

$$\text{So } L_m = \|\mathbf{x}w_m - t\|^2 = \|a_m\|^2$$

$$= \left\| \left(\mathbf{I} - \frac{2\eta}{n} \mathbf{x} \mathbf{x}^T\right)^m a_0 \right\|^2 \text{ after } m^{\text{th}} \text{ iteration}$$

2.3.2 [1pt]

Determine if in general \mathcal{L}_t is convex w.r.t. the learning rate η ? Show your work.

$$\begin{aligned} L_m &= a_0^T \left[\left(I - \frac{2\eta}{n} X X^T \right)^m \right]^T \left(I - \frac{2\eta}{n} X X^T \right)^m a_0 \\ &= a_0^T \left(I - \frac{2\eta}{n} X X^T \right)^{2m} a_0 \quad \text{since } \left(I - \frac{2\eta}{n} X X^T \right)^{2m} \\ &\qquad \text{is symmetric} \end{aligned}$$

Since $X X^T$ is symmetric, then it has a spectral decomposition: $X X^T = Q^T \Lambda Q$, where Q is orthogonal and Λ is diagonal. And entries of Λ are positive since $X X^T$ is positive definite. \downarrow denoted λ_i

$$\begin{aligned} \text{Then } I - \frac{2\eta}{n} X X^T &= Q^T I Q - \frac{2\eta}{n} Q^T \Lambda Q \\ &= Q^T \left(I - \frac{2\eta}{n} \Lambda \right) Q \end{aligned}$$

$$\text{and } \left(I - \frac{2\eta}{n} X X^T \right)^{2m} = Q^T \left(I - \frac{2\eta}{n} \Lambda \right)^{2m} Q$$

$$\begin{aligned} \text{So } L_m &= a_0^T Q^T \left(I - \frac{2\eta}{n} \Lambda \right)^{2m} Q a_0 \\ &= b^T \left(I - \frac{2\eta}{n} \Lambda \right)^{2m} b, \text{ where } b = Q a_0 \end{aligned}$$

and $I - \frac{2\eta}{n} \Lambda$ is diagonal, so entries of $\left(I - \frac{2\eta}{n} \Lambda \right)^{2m}$ are given by $\left(1 - \frac{2\eta}{n} \lambda_i \right)^{2m}$, where $\lambda_i > 0$.

$$\text{Thus, } L_m = \sum_{j=1}^n \left(1 - \frac{2\eta}{n}\lambda_j\right)^{2m} b_j^2$$

$$\frac{\partial^2 L_m}{\partial \eta^2} = \sum_{j=1}^n b_j^2 \cdot 2m \cdot (2m-1) \left(1 - \frac{2\eta}{n}\lambda_j\right)^{2m-2} \cdot \left(-\frac{2\lambda_j}{n}\right)^2 > 0$$

Thus L_m is convex w.r.t η .

3.1 Convolutional Filters [0.5pt]

Given the input matrix \mathbf{I} and filter \mathbf{J} shown below, 1) Write down the values of the resulting matrix $(\mathbf{I} * \mathbf{J})$ (the convolution operation as defined in the Lec 4 slides). Assume we have use zero padding around the input. 2) What feature does this convolutional filter detect?

$$\mathbf{I} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{J} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \mathbf{I} * \mathbf{J} = \begin{bmatrix} ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? \end{bmatrix}$$

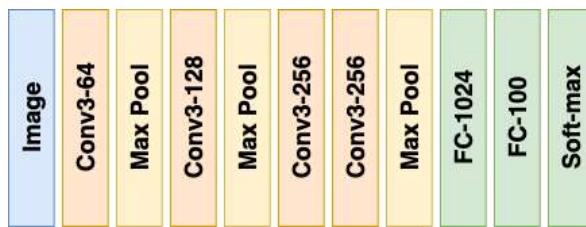
1)
$$\begin{pmatrix} -1 & 2 & 2 & -2 & 0 \\ -2 & 1 & 0 & 2 & -1 \\ 3 & 0 & 0 & 1 & -1 \\ -2 & 2 & 0 & 2 & -1 \\ 0 & -2 & 3 & -2 & 0 \end{pmatrix}$$

2) Edge detection.

3.2 Size of Conv Nets [1pt]

Consider a Conv Net with 4 conv layers like in the diagram below. All 4 conv layers have kernel size of 3×3 . The number after the hyphen specifies the number of output channels or units of a layer (e.g. *Conv3-64* layer has 64 output channels and *FC-1024* has 1024 output units). All the *Max Pool* in the diagram has size of 2×2 . Assume zero padding for conv layers and stride 2 for *Max Pool*.

$$112 \times 112 \times 3 \quad 3 \times 3 \times 3 \times 64$$



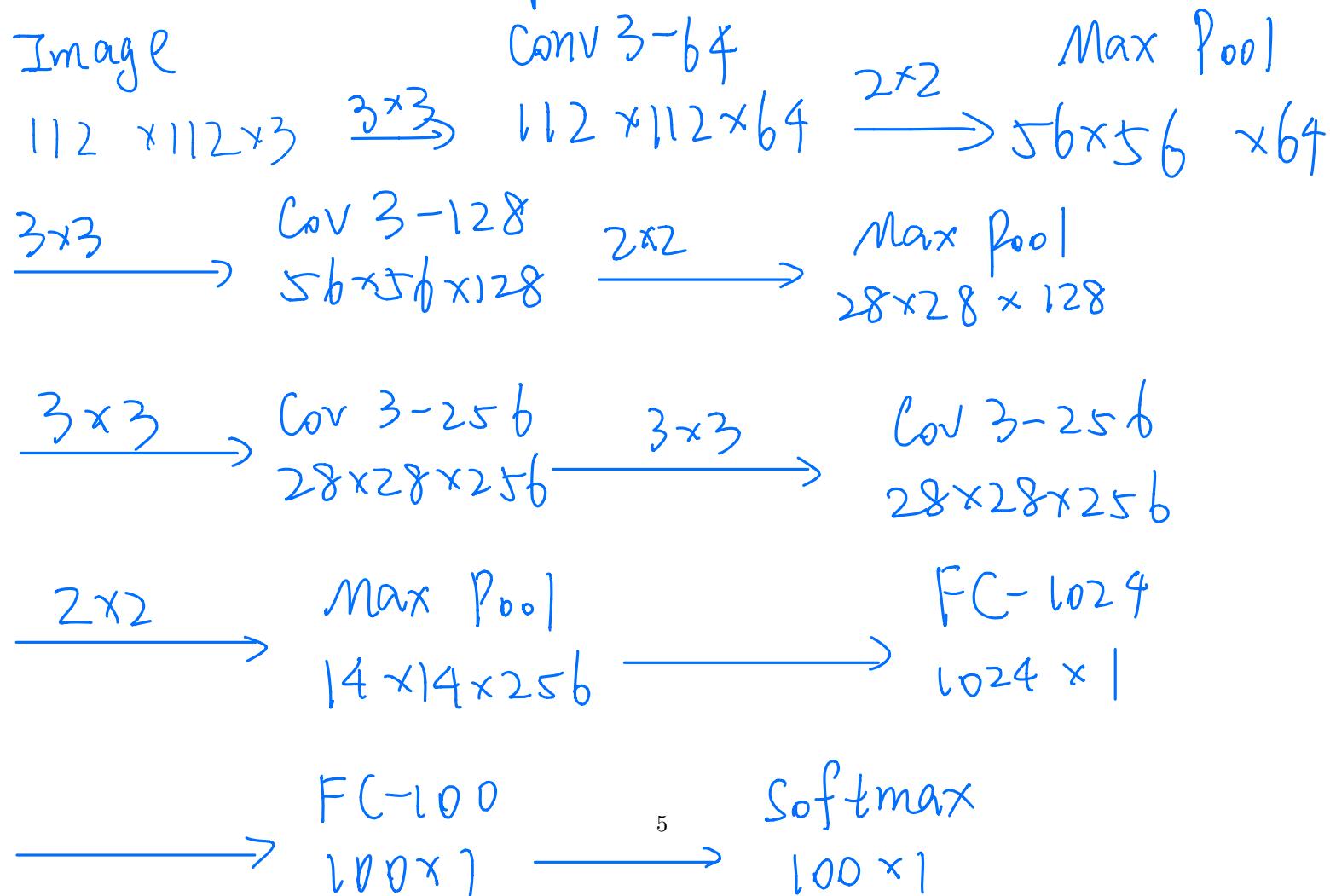
Size of the RGB input image is 112×112 (3 channels).

Calculate the following: 1) the number of parameters for this Conv Net including the bias units, 2) the total number of neurons and 3) number of input connections for neurons in each layer.

3.3 Receptive field [0.5pt]

Receptive field of a neuron in a Conv Net is the area of the input that can affect the neuron (i.e. the area a neuron can 'see'). For example, a neuron in a 3×3 conv layer is computed from an input area of 3×3 of the input, so its receptive field is 3×3 . A neuron in the next 3×3 conv layer is computed from an input area of 5×5 , so its receptive field is 5×5 . List 3 things that can affect the size of the receptive field of a neuron and briefly explain your answers.

The size of each output layer are as follows:



$$\begin{aligned}
 & 1) \# \text{ of parameters} = (\text{Conv } 3-64 + \text{Conv } 3-128 + \text{Conv } 3-256) \\
 & + (\text{Conv } 3-256 \times \text{Conv } 3-128 + \text{Conv } 3-128 \times \text{Conv } 3-256 + \text{Conv } 3-256 \times \text{Conv } 3-256) \\
 & + (\text{Conv } 3-256 \times \text{Conv } 3-256 + \text{Conv } 3-256 \times \text{FC-1024} + \text{Conv } 3-256 \times \text{FC-1024}) \\
 & + (\text{FC-1024} \times \text{FC-1024} + \text{FC-1024} \times 100) = 52444644 \\
 & 2) \# \text{ of neurons} = (\text{Conv } 3-64 + \text{Max Pool} + \text{Conv } 3-128) \\
 & + (\text{Max Pool} + \text{Conv } 3-256 + \text{Conv } 3-256) \\
 & + (\text{Max Pool} + \text{FC-1024} + \text{FC-100} + \text{softmax}) \\
 & = 1958088
 \end{aligned}$$

3) # of input connections for each layer:

$$\text{Conv 3-64: } 3 \times 3 \times 3 \times 64 \times 112^2 = 21676032$$

$$\text{Max Pool: } 4 \times 56 \times 56 \times 64 = 802816$$

$$\text{Conv 3-128: } 64 \times 3 \times 3 \times 56^2 \times 128 = 231211008$$

$$\text{Max Pool: } 4 \times 28 \times 28 \times 128 = 401408$$

$$\text{Conv 3-256: } 3 \times 3 \times 128 \times 28^2 \times 256 = 231211008$$

$$\text{Conv 3-256: } 3 \times 3 \times 256 \times 28^2 \times 256 = 462422016$$

$$\text{Max Pool: } 4 \times 14 \times 14 \times 256 = 200704$$

$$\text{FC-1024: } 14 \times 14 \times 256 \times 1024 = 51380224$$

$$\text{FC-100: } 1024 \times 100 = 102400$$

$$\text{Softmax: } 100 \times 100 = 10000$$

3.3 Receptive field [0.5pt]

Receptive field of a neuron in a Conv Net is the area of the input that can affect the neuron (i.e. the area a neuron can 'see'). For example, a neuron in a 3×3 conv layer is computed from an input area of 3×3 of the input, so its receptive field is 3×3 . A neuron in the next 3×3 conv layer is computed from an input area of 5×5 , so its receptive field is 5×5 . List 3 things that can affect the size of the receptive field of a neuron and briefly explain your answers.

- 1) Number of layers. Each output neuron contains information from all input units inside the $k \times k$ kernel. As a result, higher the # of layers, wider the range of units that can affect an output neuron.
- 2) The kernel size. This directly reflects how many input values can affect the output neuron in the next layer.
- 3) The stride value: A larger stride value can make far apart input units "close" in the output layer, thereby increasing the receptive field.