Introduction to Machine Learning

Fall 2024

University of Science and Technology of China

Lecturer: Jie Wang — Homework 6

Posted: Dec. 12, 2024 — Due: Dec. 24, 2024

Notice, to get the full credits, please show your solutions step by step.

Exercise 1: SVM for Linearly Separable Cases

Given the training set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$. Let

$$\mathcal{D}^+ = \{(\mathbf{x}_i, y_i) \in \mathcal{D} : y_i = 1\}, \quad \mathcal{D}^- = \{(\mathbf{x}_i, y_i) \in \mathcal{D} : y_i = -1\}.$$

Assume that $\mathcal{D}^+$ and $\mathcal{D}^-$ are nonempty and the training set $\mathcal{D}$ is linearly separable. We have shown in Lecture 13 that SVM can be written as

$$\begin{aligned} \min_{\mathbf{w},b} \ & \frac{1}{2}\|\mathbf{w}\|^2, \\ \text{s.t.} \ & \min_i y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) = 1. \end{aligned} \tag{1}$$

Moreover, we further transform Problem (1) to

$$\begin{aligned} \min_{\mathbf{w},b} \ & \frac{1}{2}\|\mathbf{w}\|^2, \\ \text{s.t.} \ & y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1, i = 1, \ldots, n. \end{aligned} \tag{2}$$

We denote the feasible set of Problem (2) by

$$\mathcal{F} = \{(\mathbf{w}, b) : y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1, i = 1, \ldots, n\}.$$

1. The Euclidean distance between a linear classifier $f(\mathbf{x}; \mathbf{w}, b) = \langle \mathbf{w}, \mathbf{x} \rangle + b$ and a point $\mathbf{z}$ is

   $$d(\mathbf{z}, f) = \min_{\mathbf{x}}\{\|\mathbf{z} - \mathbf{x}\| : f(\mathbf{x}; \mathbf{w}, b) = 0\}.$$

   Please find the closed form of $d(\mathbf{z}, f)$.

2. Show that $\mathcal{F}$ is nonempty.

3. Please show that Problem (2) admits an optimal solution.
   (**Hint:** recall the lecture of Convex Optimization Problems)

4. Please show that Problems (1) and (2) share the same set of optimal solutions.

5. Let $(\mathbf{w}^*, b^*)$ be the optimal solution to Problem (2). Please show that

   (a) If the training set $\mathcal{D}$ is linearly separable, we have $\mathbf{w}^* \neq 0$;

   (b)If all samples in training set $\mathcal{D}$ are positive or negative, then $\mathbf{w}^*$ can be 0.

6. Let $(\mathbf{w}^*, b^*)$ be the optimal solution to Problem (2). Show that there exist at least one positive sample and one negative sample, respectively, such that the corresponding equality holds. In other words, there exist $i, j \in \{1, 2, \ldots, n\}$ such that

$$\begin{aligned} 1 = y_i =&\langle \mathbf{w}^*, \mathbf{x}_i \rangle + b^*, \\ -1 = y_j =&\langle \mathbf{w}^*, \mathbf{x}_j \rangle + b^*. \end{aligned}$$

7. Show that the optimal solution to Problem (2) is unique and there is at least one of the constraints holds as an equality at the optimum.

8. Can we remove the inequalities that hold strictly at the optimum to Problem (2) without affecting the solution? Please justify your claim rigorously.

Solution: ■

Exercise 2: Discussions on Geometric Multiplier and Duality Gap

Consider the primal problem

$$\begin{aligned} \min_{\mathbf{x}} \ & f(\mathbf{x}) \\ \text{s.t. } & g_i(\mathbf{x}) \leq 0, \ i = 1, \cdots, m, \\ & h_i(\mathbf{x}) = 0, i = 1, \cdots, p, \\ & \mathbf{x} \in X. \end{aligned} \tag{3}$$

Recall the discussion on Geometric Multiplier and Duality Gap in Lecture 13.

1. For each of the following problems:

   (a)

   $$\begin{aligned} \min \ & f(x) = x \\ \text{s.t. } & g(x) = x^2 \leq 0, \\ & x \in X = \mathbb{R}. \end{aligned}$$

   (b)

   $$\begin{aligned} \min f(x) = & \begin{cases} e^x, & x \leq 0, \\ 1 - x, & x \in [0, 1], \\ 0, & x > 1. \end{cases} \\ \text{s.t. } & g(x) = x \leq 0. \end{aligned}$$

   (c)

   $$\begin{aligned} \min f(x) = & \begin{cases} -x, & x \leq 0, \\ 0, & x > 0. \end{cases} \\ \text{s.t. } & g(x) = x \leq 0. \end{aligned}$$

   finish the following three tasks:

   (a) Plot the graph of $f(x)$ relative to $g(x)$, that means the horizontal axis represents $g(x)$ and the vertical axis represents $f(x)$. Refer to the Figure 1 in Lecture 13.

   (b) Check whether a geometric multiplier exists and whether there is duality gap;

   (c) Find the dual problem of the primal problem and try to solve the dual problem.

2. Based on the above discussions, decide whether the following claims on the geometric multiplier and the duality gap for the primal problem are correct? Justify the claims rigorously if they are correct. Otherwise please give a counterexample for each.

   (a) The geometric multiplier for the primal problem (3) always exists.

   (b) If the geometric multiplier exists, then it is unique.

(c) If the geometric multiplier exists, then the duality gap is zero.

(d) If the duality gap is zero, there exists at least one geometric multiplier.

(e) Let (λ^*, μ^*) be a geometric multiplier. Then, the problem $\mathbf{argmin}_{\mathbf{x} \in X} L(\mathbf{x}, \lambda^*, \mu^*)$ always admits at least one solution, where $L(\mathbf{x}, \lambda, \mu)$ is the Lagrangian for (3).

(f) If (λ^*, μ^*) is a geometric multiplier and the problem $\mathbf{argmin}_{\mathbf{x} \in X} L(\mathbf{x}, \lambda^*, \mu^*)$ admits a solution $\mathbf{x}^*$, then $\mathbf{x}^*$ is feasible.

(g) (optional) Let (λ^*, μ^*) be a geometric multiplier. Then, $\mathbf{x}^*$ is a global minimum of the primal problem if and only if $\mathbf{x}^*$ is feasible and $\mathbf{x}^* \in \mathbf{argmin}_{\mathbf{x} \in X}\ L(\mathbf{x}, \lambda^*, \mu^*)$.

Exercise 3: Exercises of Dual Problems

For each of the following given optimization problems with constraints, answer the following questions respectively:

1. Give the feasible set, the optimal value and the optimal solution $\mathbf{x}^*$.
2. Write the dual problem of the primal problem, and give the KKT conditions.
3. State whether strong duality holds for this problem and support it by precise calculation.

The problems:
(1)

$$\begin{aligned} \min_{x}\, & x^2 + 2x + 3 \\ \text{s.t.}\, & (x+2)(x-2)(x-4) \leq 0, \\ & (x+1)^3 \geq 1, \\ & x \in \mathbb{R}. \end{aligned}$$

(2)

$$\begin{aligned} \min_{x_1, x_2}\, & e^{x_1} + e^{2x_2} \\ \text{s.t.}\, & x_1 + x_2 = 1, \\ & x_1, x_2 \in \mathbb{R}. \end{aligned}$$

Solution: ■

Exercise 4: The Dual Problem of SVM

Suppose that the training set is $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$. Let

$$\mathcal{D}^+ = \{(\mathbf{x}_i, y_i) \in \mathcal{D} : y_i = 1\}, \quad \mathcal{D}^- = \{(\mathbf{x}_i, y_i) \in \mathcal{D} : y_i = -1\}.$$

Assume that $\mathcal{D}^+$ and $\mathcal{D}^-$ are nonempty. The soft margin SVM takes the form of

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \ & \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^n \xi_i, \\ \text{s.t. } & y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \geq 1 - \xi_i, i = 1, \ldots, n, \\ & \xi_i \geq 0,\ i = 1, \ldots, n, \end{aligned} \tag{4}$$

The corresponding dual problem is

$$\begin{aligned} \min_{\alpha} \ & \frac{1}{2}\sum_{i=1}^n\sum_{j=1}^n \alpha_i\alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j\rangle - \sum_{i=1}^n \alpha_i \\ \text{s.t. } & \sum_{i=1}^n \alpha_i y_i = 0, \\ & \alpha_i \in [0, C], i = 1, \ldots, n. \end{aligned} \tag{5}$$

1. Consider the relation between the primal problem and dual problem:
   (a) Show that the problems (4) and (5) always admit optimal solutions.
   (b) (Optional) For a linearly separable data sample, shall we arrive at the same separating hyperplane by solving the problems in (2) and (5), respectively?

2. The function of the slack variables used in the optimization problem for soft margin hyperplanes takes the form $\sum_{i=1}^n \xi_i$. We could also use $\sum_{i=1}^n \xi_i^p$, where $p > 1$. The soft margin SVM becomes

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \ & \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^n \xi_i^p, \\ \text{s.t. } & y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \geq 1 - \xi_i, i = 1, \ldots, n, \\ & \xi_i \geq 0,\ i = 1, \ldots, n. \end{aligned} \tag{6}$$

   Please find the dual problem of (6) and the corresponding optimal conditions.

3. Let $(\mathbf{w}^*, b^*, \xi^*)$ and (α^*, μ^*) be the optimal solutions to problems (4) and (5), $(\mathbf{x}_i, y_i)$ is a point from the training set. Please **give the complementary slackness conditions** of problem (4) and:
   (a) Give the conditions that point $(\mathbf{x}_i, y_i)$ satisfies if it is a support vector.
   (b) Give the conditions that point $(\mathbf{x}_i, y_i)$ satisfies if it is misclassified.
   (c) Give the conditions that point $(\mathbf{x}_i, y_i)$ satisfies if it is in the region between the marginal hyperplanes.

4. As shown in Figure 1, the training set is $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{13}$, where $\mathbf{x}_i \in \mathbb{R}^2$ and $y_i \in \{+1, -1\}$. Suppose that we use the soft margin SVM to classify the data points and get the optimal parameters $\mathbf{w}^*$, b^*, and $\boldsymbol{\xi}^*$ by solving the problem (6).

   (a) Please write down the equations of the separating hyperplane (H_0) and the marginal hyperplanes (H_1 and H_2) in terms of $\mathbf{w}^*$ and b^*.

   (b) Please find the support vectors and the non-support vectors.

   (c) (Optional) Please find the values (or ranges) of the optimal slack variables ξ_i^* for $i = 1, 2, \ldots, 13$. (*Hint: The possible answers are* $\xi_i^* = 0$, $0 < \xi_i^* < 1$, $\xi_i^* = 1$, *and* $\xi_i^* > 1$). How do the slack variables change when the parameter C increases and decreases?

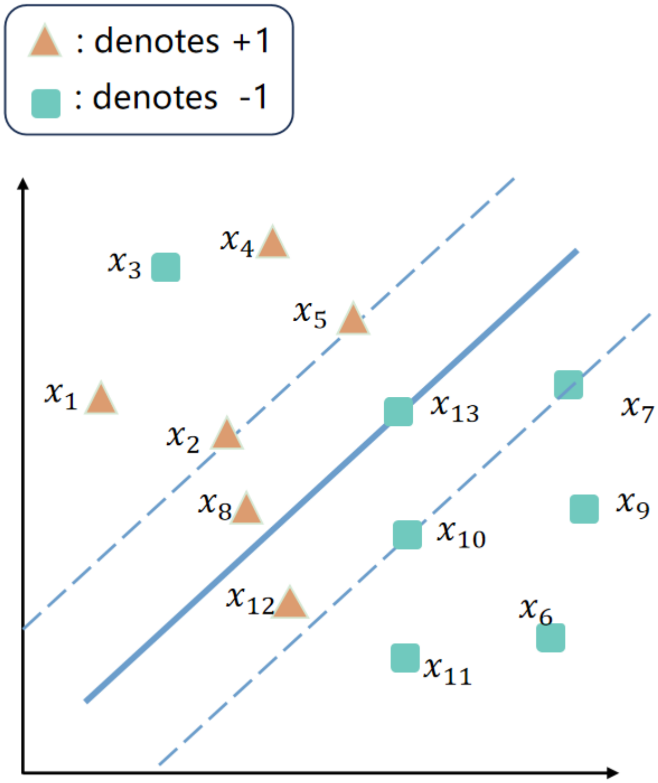

Figure 1: Classifying the data points using the soft margin SVM. H_0 is the separating hyperplane. H_1 and H_2 are the marginal hyperplanes.

Solution: ■

Exercise 5: Neural Networks

1. The softmax function $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ is defined by:

$$f_i(\mathbf{x}) = \frac{\exp(x_i)}{\sum_{k=1}^{n} \exp(x_k)}, i = 1, \ldots, n,$$

   where x_i is the i^{th} component of $\mathbf{x} \in \mathbb{R}^n$. The function $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x}))^\top$ converts each input $\mathbf{x}$ into a probability (stochastic) vector in which all entries are nonnegative and add up to one.

   (a) Please find the gradient and Jacobian matrix of $\mathbf{f}(\mathbf{x})$, i.e., $\nabla \mathbf{f}(\mathbf{x})$ and $J\mathbf{f}(\mathbf{x})$.

   (b) Show that $\mathbf{f}(\mathbf{x}) = \mathbf{f}(\mathbf{x} - c\mathbf{1})$, where $c = \max\{x_1, x_2, ..., x_n\}$ and $\mathbf{1}$ is a vector all of whose components are one. When do we need this transformation?

2. The log softmax function is one variant of softmax function, and is defined by:

$$f_i(\mathbf{x}) = \log \frac{\exp(x_i)}{\sum_{k=1}^{n} \exp(x_k)}, i = 1, \ldots, n,$$

   where x_i is the i^{th} component of $\mathbf{x} \in \mathbb{R}^n$.

   (a) Please find the gradient of $\mathbf{f}(\mathbf{x})$ (definition is the same as in 1) and show that f_i is concave.

   (b) For classification task, the **cross entropy** is usually used as loss function and its input is the output of softmax layer. But in practice, we combine the softmax layer and cross entropy as one layer, and use log softmax function in calculation instead of softmax function. Try to clarify the advantages of log softmax function in comparison to softmax function.

3. Consider the neural network with a single hidden layer in Figure 2. Let $\mathbf{x} \in \mathbb{R}^3$ be an input vector, and $\mathbf{y}$ be its corresponding output of the network. f implies that there exist four units in the hidden **fully connected** layer, each of which is followed by a **sigmoid activation function** σ, converting its input $\mathbf{z}$ to output $\mathbf{a}$. Then $\mathbf{a}$ will be transformed into the output $\mathbf{y}$ through another fully connected layer. Finally we use the **softmax layer** to get the probabilities of each class and use the **entropy loss** as the loss function. Suppose that an arbitrary input vector $\mathbf{x}_0 = (1, 1, 1)^\top$ and its ground truth label is $[0, 0, 1]^\top$.

   (a) If we initialize the weights $W_{ij}^1 = 0.40$ where $i \in \{1, 2, 3, 4\}$ and $j \in \{1, 2, 3\}$ and $W_{ij}^2 = 0.25$ where $i \in \{1, 2, 3\}$ and $j \in \{1, 2, 3, 4\}$. Compute the output $\mathbf{y}$ and loss L.

   (b) The goal of training of this neural networks is to get the weights $\mathbf{W}^1$ and $\mathbf{W}^2$, where $\mathbf{W}^1 \in \mathbb{R}^{4\times 3}$, $\mathbf{W}^2 \in \mathbb{R}^{3\times 4}$. We use the **gradient decent** to learn the weights and use the **back propagation algorithm** to compute the gradient. Write the update formula of W_{ij}^1 where $i \in \{1, 2, 3, 4\}$ and $j \in \{1, 2, 3\}$ and W_{ij}^2 where $i \in \{1, 2, 3\}$ and $j \in \{1, 2, 3, 4\}$ used in learning. The formula is expected to contain output $\mathbf{y}$ and ground truth $\mathbf{q}$.

(c) According to your results, try to represent them in the form of matrix. This will assist you in your project homework.

(d) Can we initialize all the parameters, i.e., weights and bias, of the neural network to zero? Please state you conclusion.
(**Hint:** You can search about the initialization of weights of neural networks if interested.)

(e) Thanks to the **back propagation**, the structure of neural networks can be very flexible. Rewrite the results in (b) if we replace the **sigmoid activation function** with **relu activation function** that is

$$\sigma(z) = \max(0, z).$$

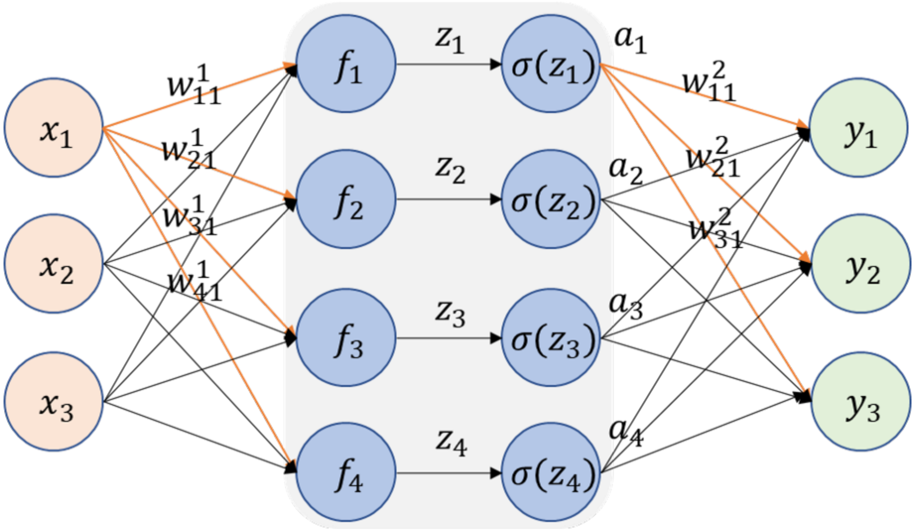

Figure 2: A neural network with a single hidden layer.

Solution:

■

Exercise 6: Convolutional Neural Networks and Some Advanced Networks Structure

conv3-32	conv5-32	max pool	conv3-64	conv5-64	max pool	FC-128	FC-10

1. Consider a convolutional neural network as shown in table above.

   (a) The convolutional layer parameters are denoted as "conv⟨filter size⟩-⟨number of filters⟩".

   (b) The fully connected layer parameters are denoted as "FC⟨number of neurons⟩".

   (c) The window size of pooling layers is 2.

   (d) The stride of convolutinal layers is 1.

   (e) The stride of pooling layers is 2.

   (f) You may want to use padding in both convolutional and pooling layers if necessary.

   (g) For convenience, we assume that there is no activation function and bias.

   Suppose that the input is a **210 × 160 RGB** image. Please derive the size of all feature maps and the number of parameters.

2. (optional) In previous discussion, the optimization method we take is actually **stochasitic**, that is we pick only one sample from the training set to compute the gradient and then update the weights. But in practice, we use **mini batch learning**, where a specific number of samples instead of only one sample or all of the samples are used to compute the loss and gradients. The loss becomes the mean of the losses of the specific number of samples. Take the cross entropy for example, the loss function becomes:

$$Loss = -\frac{1}{N}\sum_{n}\sum_{i} q_i \log(p_i),$$

   where N is the mini batch size.
   The gradient $\mathbf{g}$ also becomes the mean of the gradients of the mini-batch samples:

$$\mathbf{g} = \frac{1}{N}\sum_{n} \mathbf{g}_n,$$

   where $\mathbf{g}_n$ is the computed gradient by the nth sample in the mini batch samples.

   (a) From perspective of computational efficiency and convergence, clarify the advantages of mini-batch.
   (**Hint:** Recall the discussion of convergence of mini-SGD in HW 5.)

   (b) The initial values of weights and the distribution of the values will greatly affect the speed of the learning. The purpose of **Batch Normalization** is to adjust the distribution of activated values of each layer. Specifically, it is to adjust the distribution of the mini batch samples to a distribution with mean equal to **0** and variance equal to **1**.

For input mini batch samples $\{\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_m\}$, do the following tranformation:

$$\mu_B \leftarrow \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i$$

$$\sigma_B^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(\mathbf{x}_i - \mu_B)^2$$

$$\hat{\mathbf{x}}_i \leftarrow \frac{\mathbf{x}_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

where ϵ is a very small number to prevent division by zero.
Then output $\mathbf{y} = \{\mathbf{y}_1, \mathbf{y}_2, ...\mathbf{y}_m\}$ through an affine transformation:

$$\mathbf{y}_i \leftarrow \gamma\hat{\mathbf{x}}_i + \beta,$$

where γ and β are also to be adjusted by learning. γ is initialized with 1 and β is initialized with 0.
Compute the back propagation of **Batch Normalization Layer**.

Solution: ■

Exercise 7: Some Network Layers, Linear Transformation and Gradient (Optional)

In this exercise, we explore several kinds of network layers in the view of linear transformation.

1. **1-dimensional convolutional layer.** Suppose we have an input $\mathbf{x} \in \mathbb{R}^n$ and filter $\mathbf{w} \in \mathbb{R}^k$ $(n > k)$. We can compute the convolution of $\mathbf{x} * \mathbf{w}$ as follows:

   - Take the convolutional filter $\mathbf{w}$ and align it with the beginning of $\mathbf{x}$. Take the dot product of $\mathbf{w}$ and the $\mathbf{x}[0 : k-1]$ and assign that as the first entry of the output.
   - Suppose we have stride s. Shift the filter down by s indices, and now take the dot product of $\mathbf{w}$ and $\mathbf{x}[s : k-1+s]$ and assign to the next entry of your output.
   - Repeat until we run out of entries in $\mathbf{x}$.

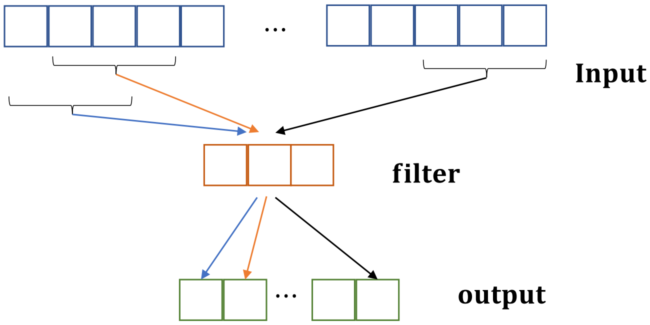

Figure 3: 1-dimensional convolutional layer.

   Now we set the stride s to be 1:

$$\mathbf{y} = \mathbf{x} * \mathbf{w} = \left(\sum_{i=1}^{k} w_i x_i, \sum_{i=1}^{k} w_i x_{i+1}, \cdots, \sum_{i=1}^{k} w_i x_{i+n-k} \right) \in \mathbb{R}^{n-k+1}.$$

   Is the 1-dimensional convolutional operation a linear transformation? If so, please find the transformation matrix, then write down the gradient with respective to $\mathbf{x}$.

2. 1×1 **convolutional layer.** Convolutional operations are linear transformations. We study a simple case, 1×1 convolutional operation, in this question. Suppose a convolutional layer takes as inputs the RGB $3 \times 28 \times 28$ images $\mathbf{X} = (x_{ijk}) \in \mathbb{R}^{3 \times 28 \times 28}$. Suppose that the convolutional layer has three $3 \times 1 \times 1$ filters where the i^{th} filter is denoted by $\mathbf{w}_i \in \mathbb{R}^3$. We set stride $= 1$ and padding $= 0$.

   Specifically, we denote the output by $\mathbf{Y} = (y_{ijk}) \in \mathbb{R}^{3 \times 28 \times 28}$, then

$$y_{ijk} = \sum_{t=1}^{3} w_{it} x_{tjk}, \quad i \in \{1, 2, 3\}, j, k \in \{1, \cdots, 28\}.$$

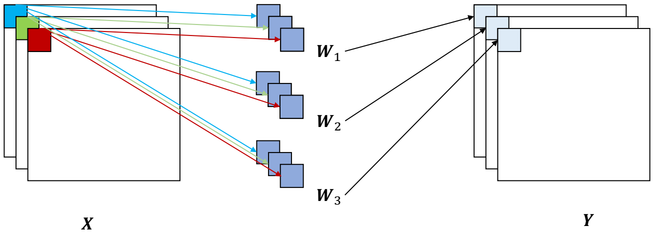

Figure 4: 1 × 1 convolutional layer.

Now we flatten the output $\mathbf{Y}$ to attain a $3 \times 28 \times 28$-dimensional vector,

$$\mathbf{y} = (y_{1,1,1}, y_{1,1,2}, \cdots, y_{1,1,28}, y_{1,2,1}, y_{1,2,2}, \cdots, y_{1,28,28}, y_{2,1,1}, y_{2,1,2}, \cdots, y_{3,28,28}).$$

We can also flatten $\mathbf{X}$ to attain a $3 \times 28 \times 28$-dimensional vector $\mathbf{x}$.

(a) Is the 1×1 convolutional operation a linear transformation? If so, Please find the transformation matrix.

(b) Please show that the 1×1 convolutional operation is invertible if and only if the matrix $(\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3)$ is invertible.

Hint: let $A = (a_{ij})_{m \times m}$, $B \in \mathbb{R}^{n \times n}$, then the $mn \times mn$ matrix

$$\begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1m}B \\ a_{21}B & a_{22}B & \cdots & a_{2m}B \\ \vdots & \vdots & \cdots & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mm}B \end{pmatrix}$$

is called the Kronecker product of A and B, denoted by $A \otimes B$. Furthermore, $\det(A \otimes B) = (\det(A))^n (\det(B))^m$.

(c) Suppose $\mathbf{x}$ is sampled from a standard Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{I})$, please find the density function of $\mathbf{y}$ if the 1×1 convolutional operation is invertible.

3. **Pooling layer.** We know that average pooling and overlapping pooling are linear transformations, but not the max pooling.

(a) Suppose an average pooling layer has window size 2×2 and stride 2, as illustrated in the following picture. The pooling layer takes as inputs the 4×4 matrices. Please find the transformation matrix of the average pooling layer.

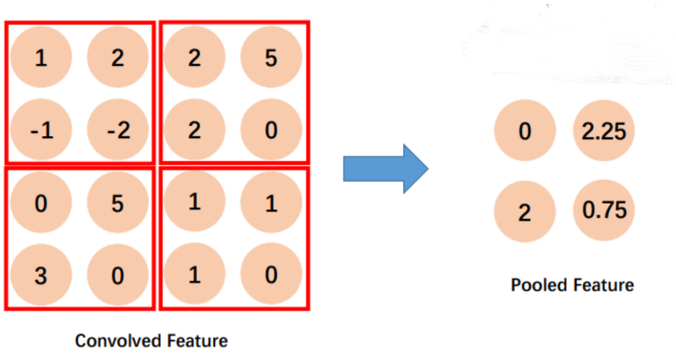

Figure 5: average pooling.

(b) Max pooling is generally not linear transformation. Consider the following example we studied in this course. The max pooling layer has window size 2×2 and stride 2, as illustrated in the following picture. The pooling layer takes as inputs the 4×4 matrices. Please find the subgradient of the max pooling operation. Then give an explanation of the "gradient" we studied in our course.

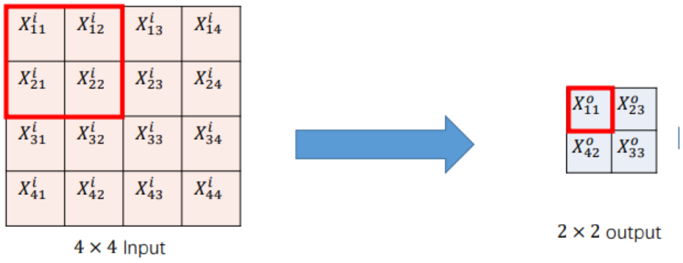

Figure 6: max pooling.

Solution: ■