CSCI567 Theory Assignment 2

Yifan Wang

3038184983

wang608@usc.edu

Problem 1 Convolutional Neural Networks (20 points)

Consider the following CNN. An $8 \times 8 \times 3$ image input, followed by a convolution layer with 2 filters of size 2×2 (stride 1, no zero padding), then another convolution layer with 4 filters of size 3×3 (stride 2, no zero padding), and finally a max pooling layer with a 2×2 filter (stride 1, no zero padding).

1.1 How many parameters are there in this network? Give two answer, with a bias neuron and without. **(10 points)**

Number of parameters without bias:

$$3 * 2 * 2 * 2 + 2 * 3 * 3 * 4 = 96$$

Number of parameters with bias:

$$96 + 2 + 4 = 102$$

1.2 What is the final dimension of the output of this network? **(10 points)**

$$8 \times 8 \times 3 -> 7 \times 7 \times 2 -> 3 \times 3 \times 4 -> 2 \times 2 \times 4$$

Problem 2 Kernel methods (15 points)

$k :^d \times^d \rightarrow$ is a valid kernel if and only if its Gram matrix, also known as Kernel Matrix, is positive semi-definite (PSD). So, to **disprove** that k is a valid kernel it is sufficient to find a set of $\mathbf{x}$ that breaks the condition of positive semi-definiteness of the Gram matrix. However, to prove that k **is** a valid kernel, it is more convenient to show that one of the following properties is true:

- it can be expressed as dot product in some transformed feature space i.e. $k(\mathbf{x}_1, \mathbf{x}_2) = \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_2)$ and $\phi :^d \rightarrow^{d'}$ is some transformation, or
- it is a linear combination of some kernels with positive coefficients k_1, k_2 i.e. $k(\mathbf{x}_1, \mathbf{x}_2) = ak_1(\mathbf{x}_1, \mathbf{x}_2) + bk_2(\mathbf{x}_1, \mathbf{x}_2)$, $a, b > 0$, or
- it is a product of two kernels k_1, k_2 i.e. $k(\mathbf{x}_1, \mathbf{x}_2) = k_1(\mathbf{x}_1, \mathbf{x}_2)k_2(\mathbf{x}_1, \mathbf{x}_2)$

2.1 Prove or disprove that $k(\mathbf{x}_1, \mathbf{x}_2) = (f(\mathbf{x}_1) + f(\mathbf{x}_2))^2$ is a valid kernel. **(10 points)**

$$\begin{aligned} & f(x_1)^2 + f(x_2)^2 + 2 * f(x_1)f(x_2) \\ & \begin{pmatrix} a & b \end{pmatrix} \begin{pmatrix} 4f(x_1)^2 & (f(x_1) + f(x_2))^2 \\ (f(x_1) + f(x_2))^2 & 4f(x_2)^2 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} \\ & = 4a^2 f(x_1)^2 + 2ab(f(x_1) + f(x_2))^2 + 4b^2 f(x_2)^2 \\ & = 3(a^2 f(x_1)^2 + b^2 f(x_2)^2) + (af(x_1) + bf(x_2))^2 \geq 0 \end{aligned} \tag{1}$$

so that it is a valid kernel

2.2 Show that if $k(\mathbf{x}_1, \mathbf{x}_2)$ is some valid kernel, then $f(k(\mathbf{x}_1, \mathbf{x}_2)) = \sum_{i=0}^{p} c_i k^i(\mathbf{x}_1, \mathbf{x}_2)$; $c_i \geq 0$, i.e. f is some polynomial of degree p with positive coefficients is also a kernel. **(5 points)**

For one component in the formular:

$$k^i(x_1, x_2)$$

based on the last point that produce of two kernels is a kernel, this part is a kernel as well;

Then, the positive linear combination of the kernels are kernels as well.

Problem 3 Gradient descent (15 points)

3.1 A logistic regression model is defined as

$$\hat{p}(y = 1|\mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x} + b)$$

where

$$\sigma = \frac{1}{1 + e^{-z}}$$

If we want to minimize the cross entropy loss function

$$L(\mathbf{w}, b, \mathbf{x}, y) = -y \log \hat{p}(y = 1|\mathbf{x}) - (1 - y) \log[1 - \hat{p}(y = 1|\mathbf{x})]$$

over an entire training set (i.e. $\min_{\mathbf{w},b} \sum_i L(\mathbf{w}, b, \mathbf{x}_i, y_i)$), we may use *stochastic* gradient descent (SGD). Write down the update rule for $\mathbf{w}$ for SGD with learning rate η (SGD updates the weights one example at a time, at each iteration). **(10 points)**

let $z = w^T x + b$

$$\begin{aligned} \frac{\partial z}{\partial w} &= x \\ \frac{\partial \sigma(z)}{\partial z} &= \frac{e^{-z}}{(1 + e^{-z})^2} \\ \frac{\partial \sigma(z)}{\partial x} &= \sigma(z)(1 - \sigma(z)) \\ L(w, b, x, y) &= -y \log \sigma(z) - (1 - y) \log(1 - \sigma(z)) \\ \frac{\partial L(w, b, x, y)}{\partial \sigma} &= -\frac{y}{\sigma(z)} - \frac{1 - y}{1 - \sigma(z)} \\ &= \frac{\sigma(z) - y}{\sigma(z)(1 - \sigma(z))} \\ \frac{\partial L(w, b, x, y)}{\partial w} &= x(z - y) \\ w &= w - \eta \frac{\partial L(w, b, x, y)}{\partial w} \end{aligned}$$

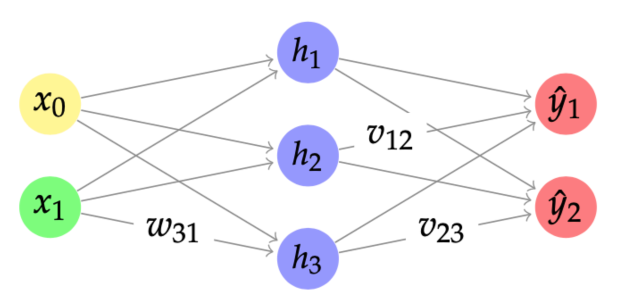
3.2 Assume all examples are normalized i.e. $\forall i : ||\mathbf{x}_i|| = 1$. Refer to your update rule. Describe, in words, when SGD makes a large update to the weights. **(5 points)**

After normalization, when SGD makes a larger update means the loss on many training sample is large. Without normaliztion, it can be resulted from that one training sample with large value has large loss. Normalization helps to stable the training process.

Problem 4 Backpropagation (20 points)

For a binary classification task, Let input $\mathbf{x} \in \mathbb{R}^2$ have a one-hot label $\mathbf{y} \in \mathbb{R}^2$. A simple neural network with weights $V \in \mathbb{R}^{2 \times 3}$ and $W \in \mathbb{R}^{3 \times 2}$ is illustrated below:

Input layer	Hidden layer	Output layer



and its forward propagation is described as:

$$h_i = \tanh\left(\sum_{j=0}^{1} w_{ij} x_j\right)$$

$$y_i = \text{softmax}(o_i), \quad o_i = \sum_{j=0}^{3} v_{ij}$$

where

$$\tanh(\alpha) = \frac{e^\alpha - e^{-\alpha}}{e^\alpha + e^{-\alpha}}$$

$$\text{softmax}(o) = \frac{\exp(o)}{\sum_{i=0}^{3} \exp(o_i)}$$

and our goal is to minimize the cross-entropy loss

$$L(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{i=0}^{2} y_i \log \hat{y}_i$$

using SGD on the gradient derived from backpropagation.

4.1 Write down $\frac{\partial L}{\partial v_{12}}$, only in terms of other partial derivatives by applying the chain rule (the first step of backpropagation). **(5 points)**

L can be written as since it is binary cross entropy loss

$$L = -y_1 \log(\hat{y}_1) - y_2 \log(1 - \hat{y}_1)$$

$$\frac{\partial L}{\partial v_{12}} = \frac{\partial L}{\partial \hat{y}_1} \frac{\partial \hat{y}_1}{\partial o_1} \frac{\partial o_1}{\partial v_{12}}$$

4.2 For the equation in part one, write down $\frac{\partial L}{\partial \hat{y}_i}$. **(5 points)**

$$\frac{\partial L}{\partial \hat{y}_1} = -(\frac{y_1}{\hat{y}_1} - \frac{y_2}{1 - \hat{y}_1})$$

$$\frac{\partial L}{\partial \hat{y}_2} = -(\frac{y_1}{1 - \hat{y}_2} - \frac{y_2}{\hat{y}_2})$$

4.3 For classification, gradients are usually calculated against a one-hot vector ($\sum y_i = 1$, where $y_j = 1$ for some class j). For the rest of the questions, consider backpropagating on an example $\mathbf{x}$ where $y_1 = 1$ and $y_2 = 0$. What is $\frac{\partial L}{\partial \hat{y}_2}$ and how does this simplify the equation in part one? **(5 points)**

with $y_2 = 0$, the loss function becomes

$$L = -y_1 \log(\hat{y}_1)$$

4.4 Write down $\frac{\partial \hat{y}_1}{\partial o_1}$ and $\frac{\partial o_1}{\partial v_{12}}$ with $y_1 = 1$ and $y_2 = 0$. **(5 points)**

$$\frac{\partial \hat{y}_1}{\partial \hat{o}_1} = y_1(1 - \hat{y}_1)$$

$$\frac{\partial \hat{o}_1}{\partial \hat{v}_{12}} = 1$$

4.5 Using the derivations above, write down $\frac{\partial L}{\partial v_{12}}$ and simplify. Explain what the SGD update rule does to v_{12}, w.r.t. to input z_k. **(5 points)**

$$\frac{\partial L}{\partial v_{12}} = -y_i(1 - \hat{y}_1)$$

4.6 Now consider v_{23}, a weight that is *not* connected to the node of the true prediction. Write down $\frac{\partial L}{\partial v_{23}}$ in terms of partial derivatives only, similar to part one. **(2 points)**

$$L = -(1 - y_2) \log(1 - \hat{y}_2) - y_2 \log(\hat{y}_2)$$

$$\frac{\partial L}{\partial v_{23}} = \frac{\partial L}{\partial \hat{y}_2} \frac{\partial \hat{y}_2}{\partial o_2} \frac{\partial o_2}{\partial v_{23}}$$

4.7 Simplify your expression above for $\frac{\partial L}{\partial v_{23}}$ using the result from part two. **(2 points)**

$$L = -\log(1 - \hat{y}_2)$$

$$\frac{\partial L}{\partial \hat{y}_2} = \frac{1 - y_2}{1 - \hat{y}_2}, \quad y_2 = 0$$

4.8 Write down $\frac{\partial \hat{y}_2}{\partial o_2}$ and $\frac{\partial o_2}{\partial v_{23}}$ with $y_1 = 1$ and $y_2 = 0$. **(2 points)**

$$\hat{y}_2 = 1 - \hat{y}_1$$

$$y_2 = 1 - y_1$$

$$\frac{\partial \hat{y}_1}{\partial \hat{y}_2} = -1$$

$$\frac{\partial \hat{y}_2}{\partial o_2} = y_2(1 - \hat{y}_2)$$

$$\frac{\partial \hat{o}_2}{\partial \hat{v}_{23}} = 1$$

4.9 Using the derivations above, write down $\frac{\partial L}{\partial v_{23}}$ and simplify. **(2 points)**

$$\frac{\partial L}{\partial v_{22}} = (1 - y_2)\hat{y}_2$$

4.10 Explain how the SGD update rule on this training example affects v_{12} and v_{23} differently. **(5 points)**

If predict is correct, it will try to make it predict more accurate (higher confidence), increase $\hat{y}_i$ by keep a negative gradient, otherwise the gradient will be positive to correct the wrong prediction.