BME/ECE 695 Deep Learning
Midterm II Solution
April 22, Spring 2021

Q1.
2 Points

Rules: I understand that this is an open book exam that shall be done within the allotted time of 120 minutes. I can use my notes, and web resources. However, I will not communicate with any other person other than the official exam proctors during the exam, and I will not seek or accept help from any other persons other than the official proctors.

Upload a scan of your signature here:

Name: _____

Q2 Back propagation of CNNs (36 Points)

As shown below, let

$$f_w(z) = z * w$$

be the forward model for a 2D single layer linear convolutional network with a single input channel and single output channel, and let

$$A = \nabla_z f_w(z)$$

$$B = \nabla_w f_w(z)$$

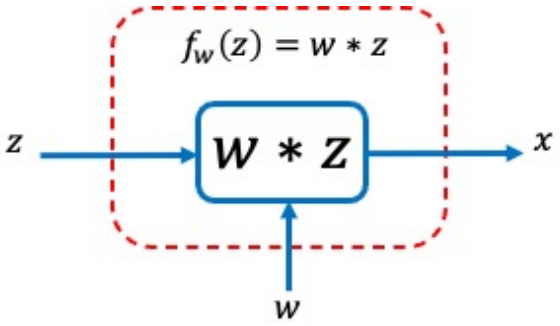
denote the gradient of the forward model with respect to the input z and its parameters w, respectively. Furthermore assume that w represents a 3×3 convolution kernel with "valid" boundary conditions, a stride of 1, and z represents a 128×128 single channel image. Also assume the use of conventional convolution (i.e., not correlation).

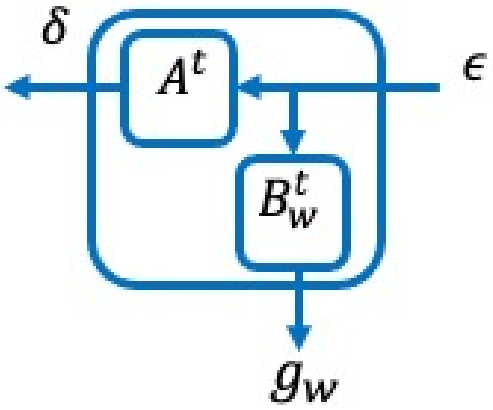
The following two functions must be implemented in order to implement both the forward inference and back propagation for this single layer.

$$x = F(z, w)$$

$$(\delta, g_w) = [G_\delta(\epsilon, w), G_{g_w}(\epsilon, z)] = G(\epsilon, z, w)$$

The figures below illustrate the functions graphically.





Q2.1 (6 Points)

Explain what the two functions $F(z, w)$ and $G(\epsilon, z, w)$ do.
For each function, explain why it is needed.

Q2.2 (6 Points)

What are the shapes of each of the following: x, ϵ, δ, and g_w?

Q2.3 (6 Points)

Write out a detailed expression for the function $x = F(z, w)$.

Q2.4 (6 Points)

Write out a detailed expression for the function $\delta = G_\delta(\epsilon, w)$.

Q2.5 (6 Points)

Write out a detailed expression for the function $g_w = G_{g_w}(\epsilon, z)$.

Q2.6 (6 Points)

Why doesn't the function $G_\delta(\epsilon, w)$ depend on z? Is this always the case?

Solution:

Q2.1
$F(z, w)$ is the forward inference function. It takes the input z and calculates the 2-D convolution between input z and kernel w. The forward function is needed to build inference models for machine learning problems and other applications.

$G(\epsilon, z, w)$ is the back propagation function. It multiplies the error ϵ by the adjoint gradient. This function is required to perform back propagation, which is required to compute the gradient of the overall inference function. The gradient of the inference function is required to perform stochastic gradient descent optimization that is in turn required to learn the parameters of a neural network.

Q2.2
The shape of x: $126 \times 126 \times 1$.
The shape of ϵ: $126 \times 126 \times 1$.
The shape of δ: $128 \times 128 \times 1$.
The shape of g_w: $3 \times 3 \times 1 \times 1$.

Q2.3

The detailed expression for x is

$$x^{j_1,j_2,j_3} = w_{(j_1,j_2),i}{}^{j_3} * z^{(j_1,j_2),i}$$

where j_1, j_2 denote the location of the element, i is the index of input channel and j_3 is the index of output channel.
Considering the single layer network has single input channel and single output channel and $z(j_1, j_2)$ is the input for $0 \leq j_1, j_2 < 128$, then

$$x^{j_1,j_2} = \sum_{k_1=0}^{2} \sum_{k_2=0}^{2} w(k_1, k_2)\, z(j_1 + 2 - k_1, j_2 + 2 - k_2)$$

where $0 \leq j_1, j_2 < 126$.

Q2.4 We have that

$$\delta_{j_1,j_2,i} = w_{(-j_1,-j_2),i}{}^{j_3} * \epsilon_{(j_1,j_2),j_3}$$

where $0 \leq j_1, j_2 < 128$.
So we will assume that ϵ is padded with zeros. More specifically padding with zeros means that if $j_1 < 0$ or $j_1 \geq 126$, then $\epsilon(j_1, j_2) = 0$; and if $j_2 < 0$ or $j_2 \geq 126$, then $\epsilon(j_1, j_2) = 0$.

Then we have that

$$\begin{aligned}\delta_{j_1,j_2} &= \sum_{k_1=0}^{2}\sum_{k_2=0}^{2} w(2-k_1, 2-k_2)\,\epsilon(j_1+2-k_1, j_2+2-k_2) \\ &= \sum_{k_1=0}^{2}\sum_{k_2=0}^{2} w(k_1,k_2)\,\epsilon(j_1+k_1, j_2+k_2)\end{aligned}$$

where $0 \le j_1, j_2 < 128$.

Q2.5 We have that

$$[g_w]_{j_1,j_2,i,j_3} = z^{(-j_1,-j_2),i} * \epsilon_{(j_1,j_2),j_3}$$

where $0 \le j_1, j_2 < 3$.
So considering the input $\epsilon(j_1, j_2)$ for $0 \le j_1, j_2 < 126$, then

$$\begin{aligned}[g_w]_{j_1,j_2} &= \sum_{k_1=0}^{125}\sum_{k_2=0}^{125} \epsilon(125-k_1, 125-k_2)\, z(j_1+125-k_1, j_2+125-k_2) \\ &= \sum_{k_1=0}^{125}\sum_{k_2=0}^{125} \epsilon(k_1,k_2)\, z(j_1+k_1, j_2+k_2)\end{aligned}$$

where $0 \le j_1, j_2 < 3$.
Q2.6

Since the forward model is a linear function of z, the adjoint gradient with respect to x doesn't depend on the value of z.
In the general case of a nonlinear operator, this gradient would depend on the value of z. So "no", this is not always the case.

Q3 ML Classification and Training (35 Points)

Let (X_k, m_k^*, Y_k) be a random triplet where

$$X_{k,m} = \delta(m - m_k^*) \ ,$$

$m = 0, \cdots, M-1$ represents the class index and m_k^* is the random variable taking on the correct class. Furthermore, assume that the triplets (X_k, m_k^*, Y_k) for $k = 0, \cdots, K-1$ are i.i.d. with

$$P\{X_{k,m} = 1 | Y_k = y_k\} = f_\theta(m|y_k) \ .$$

where θ is a parameter vector for a family of distributions.

Q3.1 (5 Points)

What type of encoding is this for $X_{k,m}$?

Q3.2 (5 Points)

What constraints must hold for the value of the function $f_\theta(m|y_k)$?

Q3.3 (5 Points)

Prove that the following equation holds.

$$P\{x_{k,m_k^*} = 1 | y_k\} = \prod_{m=0}^{M-1} [f_\theta(m|y_k)]^{x_{k,m}}$$

Q3.4 (5 Points)

Calculate an expression for $\log p_\theta(x_k|y_k)$, the conditional probability that $X_k = x_k$ given that $Y_k = y_k$.

Q3.5 (5 Points)

Calculate an expression for $\log p_\theta(x|y)$, the conditional probability that $X = x$ given that $Y = y$.

Q3.6 (5 Points)

Find an expression for the maximum likelihood estimate,

$$\hat{\theta} = \arg\min_\theta \{-\log p_\theta(x|y)\}$$

Q3.7 (5 Points)

Explain how one can implement this estimation procedure in a neural network framework.

Solution:

Q3.1

One-hot encoding.

Q3.2

Since $f_\theta(m|y_k) = P\{X_{k,m} = 1|Y_k = y_k\}$, we have that

$$f_\theta(m|y_k) \geq 0$$
$$\sum_{m=0}^{M-1} f_\theta(m|y_k) = 1 \ .$$

Q3.3

$$\begin{aligned} P\{x_{k,m^*} = 1|y_k\} &= f_\theta(m^*|y_k) \\ &= \prod_{m=0}^{M-1} f_\theta(m|y_k)^{\delta(m-m^*)} \\ &= \prod_{m=0}^{M-1} f_\theta(m|y_k)^{x_{k,m}} \end{aligned}$$

Q3.4

$$\begin{aligned} \log p_\theta(x_k|y_k) &= \log f_\theta(m^*|y_k) \\ &= \log \left\{ \prod_{m=0}^{M-1} f_\theta(m|y_k)^{x_{k,m}} \right\} \\ &= \sum_{m=0}^{M-1} x_{k,m} \log f_\theta(m|y_k) \end{aligned}$$

Q3.5

Given the triplets are i.i.d, we have

$$\begin{aligned} \log p_\theta(x|y) &= \log \left\{ \prod_{k=0}^{K-1} p_\theta(x_k|y_k) \right\} \\ &= \sum_{k=0}^{K-1} \log p_\theta(x_k|y_k) \\ &= \sum_{k=0}^{K-1} \sum_{m=0}^{M-1} x_{k,m} \log f_\theta(m|y_k) \end{aligned}$$

Q3.6

$$
\begin{aligned}
\hat{\theta} &= \arg\min_{\theta} \{-\log p_{\theta}(x|y)\} \\
&= \arg\min_{\theta} \left\{ -\sum_{k=0}^{K-1} \sum_{m=0}^{M-1} x_{k,m} \log f_{\theta}(m|y_k) \right\} \\
&= \arg\min_{\theta} \left\{ \sum_{k=0}^{K-1} \rho_{CE} \left(x_k, f_{\theta}(m|y_k) \right) \right\} \\
&= \arg\min_{\theta} \left\{ L_{CE} \left(\theta; x, y \right) \right\}
\end{aligned}
$$

Therefore, the maximum likelihood estimate of $\hat{\theta}$ is given by minimizing the Cross Entropy loss function.

Q3.7

For classification tasks, we first apply one-hot encoding to class labels of the ground truth data. Then we train the neural network to minimizes the cross entropy loss between output and ground truth. Typically, the SGD algorithm is used for training the neural network, and the gradient of the parameters is computed by back propagating of the error.

Q4 Stochastic Gradient Descent (20 Points)

While training a deep neural network, you use a training set of size K with batches of size K_b.

Q4.1 (5 Points)

What name is used for one full pass through the training data?

Q4.2 (5 Points)

If the batch size is doubled, how much reduction in gradient "noise" would one expect? Specify your answer in terms of the standard deviation of the noise.

Q4.3 (5 Points)

What are the advantages and disadvantages of using a small batch size?

Q4.4 (5 Points)

When is it a good idea to use a large batch size?

Solution:

Q4.1

One full pass throught the training data is called an "epoch".

Q4.2

Let ω denote the gradient noise is whole dataset, where

$$E[\omega] = 0, Var[w] = \sigma^2$$

Given the batch size K_b, the gradient noise of the batch is $\frac{\omega}{\sqrt{K_b}}$. The standard deviation of the noise is given by

$$\sigma_1 = \frac{\sigma}{\sqrt{K_b}}$$

If the batch size is doubled, then the gradient noise becomes $\frac{1}{\sqrt{2K_b}}$. The standard deviation of the noise is given by

$$\sigma_2 = \frac{\sigma}{\sqrt{2K_b}}$$

Therefore, the standard deviation of the noise is reduced by a factor of $\sqrt{2}$.

Q4.3

The advantages of small batch size include:

- It provides faster gradient updates.
- It provides better exploration.
- It has less memory requirements.

The disadvantages of small batch size include:

- The gradient estimate will be noisy.
- The gradient noise may cause the algorithms to hunt around the local minimum.

Q4.4

When we have a smooth optimization function and better local convergence is desired, the large batch size is preferred.

Q5 Momentum (10 Points)

Write out the pseudo-code for gradient descent with momentum.

Solution:

Q5

The pseudo-code for gradient descent with momentum:

Init $v \leftarrow 0, \gamma, \alpha$
Repeat until converged{
$d \leftarrow -\nabla L(\theta)$
$v \leftarrow \gamma v + \alpha d$
$\theta \leftarrow \theta + \alpha v^t$
}

where

- L denotes the loss function.
- $\gamma \in [0, 1]$ denotes the momentum scalar.
- α denotes the step size.
- θ denotes the learnable parameters.