EECS 545 Machine Learning
Homework #4

1 [20 points] Neural Network Layer Implementation

(a) **[15 points] Fully-Connected Layer**
Consider the inference equation given in the question:

$$Y = XW + B$$

Furthermore, using the fact that we have the following dimensions of the given variables, $X \in \mathbb{R}^{N \times D_{in}}$, $W \in \mathbb{R}^{D_{in} \times D_{out}}$, and $B \in \mathbb{R}^{N \times D_{out}}$, we can make the sample-wise inference equation as:

$$y^{(n)} = x^{(n)}W + b$$

In this case, $y^{(n)} \in \mathbb{R}^{1 \times D_{out}}$, $b \in \mathbb{R}^{1 \times D_{out}}$ and $x^{(n)} \in \mathbb{R}^{1 \times D_{in}}$ for $1 \leq n \leq N$. Now, in order to change the notation slightly, we can consider $y^{(n)} = Y^{(n)}$ to be the n-th output label while we can also consider $x^{(n)} = X^{(n)}$ to be the n-th training example. Please note that the inference equation can now be seen in the following matrix multiplication format for an arbitrary value of n where $1 \leq n \leq N$:

$$\begin{bmatrix} Y_1^{(n)} & Y_2^{(n)} & \cdots & Y_{D_{out}}^{(n)} \end{bmatrix} = \begin{bmatrix} X_1^{(n)} & X_2^{(n)} & \cdots & X_{D_{in}}^{(n)} \end{bmatrix} \begin{bmatrix} W_{1,1} & \cdots & W_{1,D_{out}} \\ W_{2,1} & \cdots & W_{2,D_{out}} \\ \vdots & \ddots & \vdots \\ W_{D_{in},1} & \cdots & W_{D_{in},D_{out}} \end{bmatrix} + \begin{bmatrix} b_1 & b_2 & \cdots & b_{D_{out}} \end{bmatrix}$$

Please note here that the notation $X_m^{(n)}$ represents the m-th dimension of the n-th training example. This can be further simplified to the following summation expression for all m where $1 \leq m \leq D_{out}$:

$$Y_m^{(n)} = \sum_{i=1}^{D_{in}} X_i^{(n)} W_{i,m} + b_m$$

[5 points] Gradient $\frac{\partial L}{\partial W}$:

Using the equation given in the hint, we have:

$$\begin{aligned} \frac{\partial L}{\partial W_{i,j}} &= \sum_{n=1}^{N} \sum_{m=1}^{D_{out}} \frac{\partial L}{\partial Y_m^{(n)}} \frac{\partial Y_m^{(n)}}{\partial W_{i,j}} \\ &= \sum_{n=1}^{N} \frac{\partial L}{\partial Y_j^{(n)}} \frac{\partial Y_j^{(n)}}{\partial W_{i,j}} \\ &= \sum_{n=1}^{N} \frac{\partial L}{\partial Y_j^{(n)}} X_i^{(n)} \end{aligned}$$

With this, consider the following matrix multiplication to understand the vectorized solution:

$$\frac{\partial L}{\partial W} = \begin{bmatrix} \frac{\partial L}{\partial W_{1,1}} & \frac{\partial L}{\partial W_{1,2}} & \cdots & \frac{\partial L}{\partial W_{1,D_{out}}} \\ \frac{\partial L}{\partial W_{2,1}} & \frac{\partial L}{\partial W_{2,2}} & \cdots & \frac{\partial L}{\partial W_{2,D_{out}}} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial L}{\partial W_{D_{in},1}} & \frac{\partial L}{\partial W_{D_{in},2}} & \cdots & \frac{\partial L}{\partial W_{D_{in},D_{out}}} \end{bmatrix}$$

$$= \begin{bmatrix} X_1^{(1)} & X_1^{(2)} & \cdots & X_1^{(N)} \\ X_2^{(1)} & X_2^{(2)} & \cdots & X_2^{(N)} \\ \cdots & \cdots & \cdots & \cdots \\ X_{D_{in}}^{(1)} & X_{D_{in}}^{(2)} & \cdots & X_{D_{in}}^{(N)} \end{bmatrix} \begin{bmatrix} \frac{\partial L}{\partial Y_1^{(1)}} & \frac{\partial L}{\partial Y_2^{(1)}} & \cdots & \frac{\partial L}{\partial Y_{D_{out}}^{(1)}} \\ \frac{\partial L}{\partial Y_1^{(2)}} & \frac{\partial L}{\partial Y_2^{(2)}} & \cdots & \frac{\partial L}{\partial Y_{D_{out}}^{(2)}} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial L}{\partial Y_1^{(N)}} & \frac{\partial L}{\partial Y_2^{(N)}} & \cdots & \frac{\partial L}{\partial Y_{D_{out}}^{(N)}} \end{bmatrix}$$

$$= \boxed{X^T \frac{\partial L}{\partial Y}}$$

[5 points] Gradient $\frac{\partial L}{\partial b}$:

Using the equation given in the hint, we have:

$$\frac{\partial L}{\partial b_j} = \sum_{n=1}^{N} \sum_{m=1}^{D_{out}} \frac{\partial L}{\partial Y_m^{(n)}} \frac{\partial Y_m^{(n)}}{\partial b_j}$$

$$= \sum_{n=1}^{N} \frac{\partial L}{\partial Y_j^{(n)}} \cdot 1$$

$$= \boxed{\begin{bmatrix} \sum_{n=1}^{N} \frac{\partial L}{\partial Y_1^{(n)}} & \sum_{n=1}^{N} \frac{\partial L}{\partial Y_2^{(n)}} & \cdots & \sum_{n=1}^{N} \frac{\partial L}{\partial Y_{D_{out}}^{(n)}} \end{bmatrix}}$$

[5 points] Gradient $\frac{\partial L}{\partial X}$:

Using the equation given in the hint, we have:

$$\frac{\partial L}{\partial X_i^{(n)}} = \sum_{m=1}^{D_{out}} \frac{\partial L}{\partial Y_m^{(n)}} \frac{\partial Y_m^{(n)}}{\partial X_i^{(n)}}$$

$$= \sum_{m=1}^{D_{out}} \frac{\partial L}{\partial Y_m^{(n)}} W_{i,m}$$

With this, consider the following matrix multiplication to understand the vectorized solution:

$$\frac{\partial L}{\partial X} = \begin{bmatrix} \frac{\partial L}{\partial X_1^{(1)}} & \frac{\partial L}{\partial X_2^{(1)}} & \cdots & \frac{\partial L}{\partial X_{D_{in}}^{(1)}} \\ \frac{\partial L}{\partial X_1^{(2)}} & \frac{\partial L}{\partial X_2^{(2)}} & \cdots & \frac{\partial L}{\partial X_{D_{in}}^{(2)}} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial L}{\partial X_1^{(N)}} & \frac{\partial L}{\partial X_2^{(N)}} & \cdots & \frac{\partial L}{\partial X_{D_{in}}^{(N)}} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\partial L}{\partial Y_1^{(1)}} & \frac{\partial L}{\partial Y_2^{(1)}} & \cdots & \frac{\partial L}{\partial Y_{D_{out}}^{(1)}} \\ \frac{\partial L}{\partial Y_1^{(2)}} & \frac{\partial L}{\partial Y_2^{(2)}} & \cdots & \frac{\partial L}{\partial Y_{D_{out}}^{(2)}} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial L}{\partial Y_1^{(N)}} & \frac{\partial L}{\partial Y_2^{(N)}} & \cdots & \frac{\partial L}{\partial Y_{D_{out}}^{(N)}} \end{bmatrix} \begin{bmatrix} W_{1,1} & W_{2,1} & \cdots & W_{D_{in},1} \\ W_{1,2} & W_{2,2} & \cdots & W_{D_{in},2} \\ \cdots & \cdots & \cdots & \cdots \\ W_{1,D_{out}} & W_{2,D_{out}} & \cdots & W_{D_{in},D_{out}} \end{bmatrix}$$

$$= \boxed{\frac{\partial L}{\partial Y} W^T}$$

(b) **[5 points] Gradient of ReLU**

Let X be a tensor and $Y = \text{ReLU}(X)$. Express $\frac{\partial L}{\partial X}$ in terms of $\frac{\partial L}{\partial Y}$.

For a scalar x,

$$y = \text{ReLU}(x) = \begin{cases} 0 \text{ if } x < 0 \\ x \text{ if } x \geq 0 \end{cases}$$

$$\Rightarrow \frac{\partial y}{\partial x} = \begin{cases} 0 \text{ if } x < 0 \\ 1 \text{ if } x > 0 \end{cases} = \delta[x > 0]$$

$$\Rightarrow \frac{\partial L}{\partial x} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial x} = \frac{\partial L}{\partial y}\delta[x > 0]$$

Since the ReLU operation is element-wise, we can generalize this to tensors X and Y,

$$\frac{\partial L}{\partial X} = \frac{\partial L}{\partial Y} \odot I[X > 0]$$

Where $\odot$ represents the Hadamard or element-wise product and $I[.]$ represents the indicator function.

2 [20 points] Multi-class classification with Softmax

With 200 hidden units, we achieved a test accuracy of approximately 97.70%.

3 [20 points] Convolutional Neural Network for multi-class classification

With the pre-specified hyperparameters, we achieved a test accuracy of approximately 97.61%.

4 [20 points] Application to Image Captioning

The learning curve can be seen in Figure 1. The training samples are shown in Figure 2 and the validation samples are shown in Figure 3. Because we train the RNN model on a small dataset, it is easy for the model to overfit the training dataset. Therefore, the model could not perform well on validation dataset and the captions aren't very close to the ground truth.

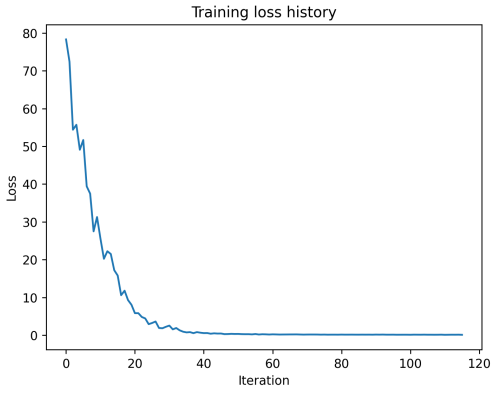

Figure 1: Loss versus Iteration Number

Figure 2: Training Samples

Figure 3: Validation Samples

5 [20 points] Transfer learning

In the scenario where we are working with the 'fine-tuned' model, we obtained the best validation accuracy as **93.46%** and the sample results can be seen in Figure 4.

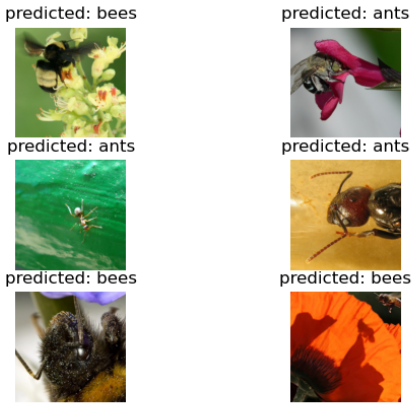

Figure 4: Sample results from the 'fine-tuned' model

In the scenario where we are working with the 'freeze' model, we obtained the best validation accuracy as **95.42%** and the sample results can be seen in Figure 5.

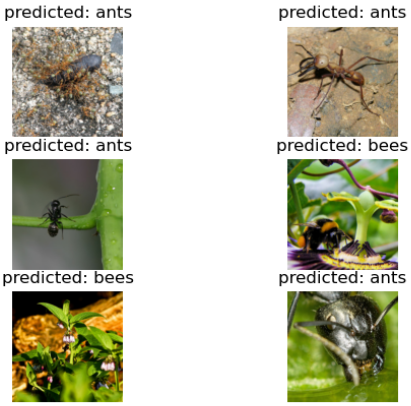

Figure 5: Sample results from the 'freeze' model

6 [6 points (extra credit)] Convolutional Backward Pass

[3 points] Gradient $\frac{\partial L}{\partial X_{n,c}}$

Consider now the output tensor generated by a convolutional layer Y. In order to compute Y, we use the following formula:

$$Y_{n,f} = \sum_{c'=1}^{C} X_{n,c'} *_{\text{filt}} K_{f,c'}$$

Now, we can expand this for every location in $Y_{n,f}$ and write out the filter operation in summation format to get the following expression:

$$Y_{n,f,p,q} = \sum_{c'=1}^{C} \sum_{m=p}^{p+H'-1} \sum_{n'=q}^{q+W'-1} X_{n,c',m,n'} K_{f,c',m-p+1,n'-q+1}$$

Now, consider the gradient of $Y_{n,f,p,q}$ with respect to $X_{n,c,i,j}$, which can be written as follows:

$$\frac{\partial Y_{n,f,p,q}}{\partial X_{n,c,i,j}} = \frac{\partial}{\partial X_{n,c,i,j}} \left(\sum_{c'=1}^{C} \sum_{m=p}^{p+H'-1} \sum_{n'=q}^{q+W'-1} X_{n,c',m,n'} K_{f,c',m-p+1,n'-q+1} \right)$$

Clearly, we can see that the partial derivative on the right hand side of the equation will only be non-zero when $c' = c$, $m = i$, and $n' = j$, while for all other indices c', m, n', we get a zero partial derivative (since we're taking the partial with respect to a single scalar $X_{n,c,i,j}$). This allows us to simplify the expression as:

$$\frac{\partial Y_{n,f,p,q}}{\partial X_{n,c,i,j}} = K_{f,c,i-p+1,j-q+1}$$

Now, we can consider the gradient of the loss with respect to $X_{n,c,i,j}$ by using the chain rule to get the following summation format:

$$\frac{\partial L}{\partial X_{n,c,i,j}} = \sum_{f=1}^{F} \sum_{p=1}^{H''} \sum_{q=1}^{W''} \frac{\partial Y_{n,f,p,q}}{\partial X_{n,c,i,j}} \frac{\partial L}{\partial Y_{n,f,p,q}}$$

Using the expression derived for partial of $Y_{n,f,p,q}$ with respect to $X_{n,c,i,j}$, we get:

$$\frac{\partial L}{\partial X_{n,c,i,j}} = \sum_{f=1}^{F} \sum_{p=1}^{H''} \sum_{q=1}^{W''} K_{f,c,i-p+1,j-q+1} \left(\frac{\partial L}{\partial Y_{n,f,p,q}} \right)$$

Finally, we can do a change of variable as follows:

$$m = i - p + 1 \Longrightarrow p = i - m + 1$$

Now, we can consider the limits for the summation $p = 1$ and $p = H''$

$$\begin{aligned} \text{For p} = 1 \implies m &= i - 1 + 1 = i \\ \text{For p} = H'' \implies m &= i - H'' + 1 \end{aligned}$$

Therefore, we can say that after a change in variable, we get $p = i - m + 1$ and m varies from $i - H'' + 1$ to i (reversed because $i - H'' + 1 < i$).

Similarly, we have:

$$t = j - q + 1 \Longrightarrow q = j - t + 1$$

Note here that t will vary from $j - W'' + 1$ to j. Finally, we can substitute these changed variables to get:

$$\frac{\partial L}{\partial X_{n,c,i,j}} = \sum_{f=1}^{F} \sum_{m=i-H''+1}^{i} \sum_{t=j-W''+1}^{j} K_{f,c,m,t} \left(\frac{\partial L}{\partial Y_{n,f,i-m+1,j-t+1}} \right)$$

One important caveat here is that the convolutional kernel $K_{f,c}$ is of size $(H' \times W')$ which means that whenever $m < 1$ or $m > H'$ or $t < 1$ or $t > W'$, we should treat $K_{f,c,m,t} = 0$. This actually corresponds to zero padding the kernel for full convolution. Finally, we can see that the inner two summations exactly match the definition of full convolution, so:

$$\sum_{m=i-H''+1}^{i} \sum_{t=j-W''+1}^{j} K_{f,c,m,t} \left(\frac{\partial L}{\partial Y_{n,f,i-m+1,j-t+1}} \right) = K_{f,c} *_{\text{full}} \left(\frac{\partial L}{\partial Y_{n,f}} \right)$$

And so,

$$\boxed{\frac{\partial L}{\partial X_{n,c}} = \sum_{f=1}^{F} K_{f,c} *_{\text{full}} \left(\frac{\partial L}{\partial Y_{n,f}} \right)}$$

[3 points] Gradient $\frac{\partial L}{\partial K_{f,c}}$

As we did before, we can consider the output tensor generated by a convolutional layer Y. In order to compute Y, we use the following formula:

$$Y_{n,f} = \sum_{c'=1}^{C} X_{n,c'} *_{\text{filt}} K_{f,c'}$$

Now, we can expand this for every location in $Y_{n,f}$ and write out the filter operation in summation format to get the following expression:

$$Y_{n,f,p,q} = \sum_{c'=1}^{C} \sum_{m=p}^{p+H'-1} \sum_{n'=q}^{q+W'-1} X_{n,c',m,n'} K_{f,c',m-p+1,n'-q+1}$$

Now, consider the gradient of $Y_{n,f,p,q}$ with respect to $K_{f,c,i,j}$, which can be written as follows:

$$\frac{\partial Y_{n,f,p,q}}{\partial K_{f,c,i,j}} = \frac{\partial}{\partial K_{f,c,i,j}} \left(\sum_{c'=1}^{C} \sum_{m=p}^{p+H'-1} \sum_{n'=q}^{q+W'-1} X_{n,c',m,n'} K_{f,c',m-p+1,n'-q+1} \right)$$

Clearly, we can see that the partial derivative on the right hand side of the equation will only be non-zero when $c' = c$, $m - p + 1 = i$, and $n' - q + 1 = j$, while for all other indices c', m, n', we get a zero partial derivative (since we're taking the partial with respect to a single scalar $K_{f,c,i,j}$). Note here that since $m - p + 1 = i$, we have $m = i + p - 1$. Similarly, since $n' - q + 1 = j$, we have $n' = j + q - 1$. This allows us to simplify the expression as:

$$\frac{\partial Y_{n,f,p,q}}{\partial K_{f,c,i,j}} = X_{n,c,i+p-1,j+q-1}$$

Now, we can consider the gradient of the loss with respect to $K_{f,c,i,j}$ by using the chain rule to get the following summation format:

$$\frac{\partial L}{\partial K_{f,c,i,j}} = \sum_{n=1}^{N} \sum_{p=1}^{H''} \sum_{q=1}^{W''} \frac{\partial Y_{n,f,p,q}}{\partial K_{f,c,i,j}} \frac{\partial L}{\partial Y_{n,f,p,q}}$$

Substituting the gradient of $Y_{n,f,p,q}$ with respect to $K_{f,c,i,j}$, to get:

$$\frac{\partial L}{\partial K_{f,c,i,j}} = \sum_{n=1}^{N} \sum_{p=1}^{H''} \sum_{q=1}^{W''} X_{n,c,i+p-1,j+q-1} \frac{\partial L}{\partial Y_{n,f,p,q}}$$

Now, it is easy to see that the last two summations represent a filtering operation (equivalent to the second summation term in the definition provided at the start of the problem). Mathematically, this means that:

$$\sum_{p=1}^{H''} \sum_{q=1}^{W''} X_{n,c,i+p-1,j+q-1} \frac{\partial L}{\partial Y_{n,f,p,q}} = X_{n,c} *_{\text{filt}} \left(\frac{\partial L}{\partial Y_{n,f}} \right)$$

And finally, this yields:

$$\boxed{\frac{\partial L}{\partial K_{f,c}} = \sum_{n=1}^{N} X_{n,c} *_{\text{filt}} \left(\frac{\partial L}{\partial Y_{n,f}} \right)}$$