Jingquan Li 119010148 DDA3020 Homework 3

Problem 1 (12pts) Decision Tree Construction

You want to figure out what makes people click on online advertisements, so you try a variety of different advertisements, and record user behavior in the following table, which is our training set.

Advertisement	Animated?	Popup?	Colorful?	Clicked
1	Yes	Yes	No	No
2	No	Yes	Yes	No
3	No	Yes	No	No
4	No	No	Yes	Yes
5	Yes	No	Yes	Yes
6	Yes	No	No	Yes
7	No	No	Yes	Yes
8	No	No	No	No

The middle three columns ("Animated", "Popup", "Colorful") are the features of each advertisement. The last column, "Clicked", is the label, specifying whether or not a user clicked on the advertisement. You want to predict the "Clicked" label for other advertisements not in this training set.

(a) Which feature should you split on at the root of the decision tree to maximize the information gain? Write an expression for the information gain of the best split. (Your expression can contain logarithms and fractions).

Solution: (a) $H(\text{class}) = H(\frac{4}{8}, \frac{4}{8}) = -\frac{4}{8}\log_2(\frac{4}{8}) - \frac{4}{8}\log_2(\frac{4}{8}) = 1$

$H(\text{class} \mid \text{Animated}) = \frac{3}{8}H(\frac{2}{3}, \frac{1}{3}) + \frac{5}{8}H(\frac{2}{5}, \frac{3}{5}) = \frac{3}{8}(-\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3}) + \frac{5}{8}(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}) \approx 0.951$

$H(\text{class} \mid \text{Popup}) = \frac{3}{8}H(\frac{0}{3}, \frac{3}{3}) + \frac{5}{8}H(\frac{4}{5}, \frac{1}{5}) = \frac{3}{8}(-\frac{0}{3}\log_2\frac{0}{3} - \frac{3}{3}\log_2\frac{3}{3}) + \frac{5}{8}(-\frac{4}{5}\log_2\frac{4}{5} - \frac{1}{5}\log_2\frac{1}{5}) \approx 0.451$

$H(\text{class} \mid \text{colorful}) = \frac{4}{8}H(\frac{3}{4}, \frac{1}{4}) + \frac{4}{8}H(\frac{1}{4}, \frac{3}{4}) = \frac{4}{8}(-\frac{3}{4}\log_2\frac{3}{4} - \frac{1}{4}\log_2\frac{1}{4}) + \frac{4}{8}(-\frac{1}{4}\log_2\frac{1}{4} - \frac{3}{4}\log_2\frac{3}{4}) \approx 0.811$

$\therefore I(\text{class}; \text{Animated}) = H(\text{class}) - H(\text{class} \mid \text{Animated}) = 1 - 0.951 = 0.049$ bits

$I(\text{class}; \text{Popup}) = H(\text{class}) - H(\text{class} \mid \text{Popup}) = 1 - 0.451 = 0.549$ bits

$I(\text{class}; \text{colorful}) = H(\text{class}) - H(\text{class} \mid \text{colorful}) = 1 - 0.811 = 0.189$ bits

$\because I(\text{class}; \text{Popup}) > I(\text{class}; \text{colorful}) > I(\text{class}; \text{animated})$

$\therefore$ I should split on the Popup feature at the root of the decision tree.

(b) Draw the decision tree that maximizes information gain at each split. Your drawing should include the leaves and the training points they store. (Do not include entropies or information gains). Stop splitting at pure nodes.

Advertisement	Popup	Colorful	Animated	Clicked
4	No	Yes	No	Yes
5	No	Yes	Yes	Yes
6	No	No	Yes	Yes
7	No	Yes	No	Yes
8	No	No	No	No

$H(\text{class}) = H(\frac{4}{5}, \frac{1}{5}) = -\frac{4}{5}\log_2\frac{4}{5} - \frac{1}{5}\log_2\frac{1}{5} \approx 0.722$

$H(\text{class} \mid \text{colorful}) = \frac{3}{5}H(\frac{3}{3}, \frac{0}{3}) + \frac{2}{5}H(\frac{1}{2}, \frac{1}{2}) = \frac{3}{5}(-\frac{3}{3}\log_2\frac{3}{3} - \frac{0}{3}\log_2\frac{0}{3}) + \frac{2}{5}(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}) = 0.4$

$H(\text{class} \mid \text{Animated}) = \frac{2}{5}H(\frac{2}{2}, \frac{0}{2}) + \frac{3}{5}(\frac{2}{3}, \frac{1}{3}) = \frac{2}{5}(-\frac{2}{2}\log_2\frac{2}{2} - \frac{0}{2}\log_2\frac{0}{2}) + \frac{3}{5}(-\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3}) \approx 0.551$

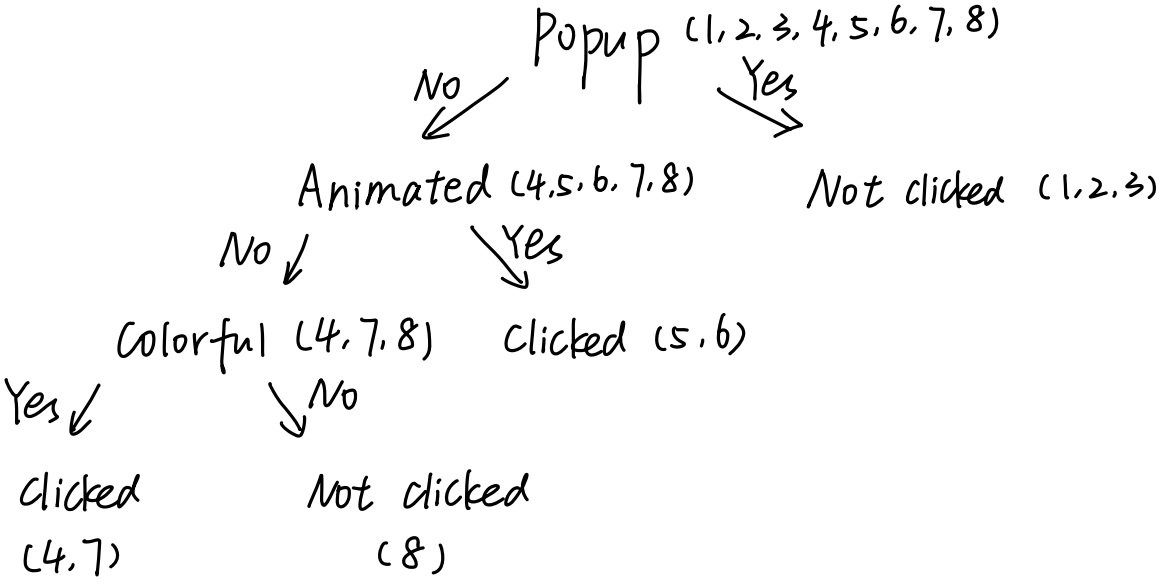
$\therefore I(\text{Class; colorful}) = H(\text{class}) - H(\text{class} \mid \text{colorful}) = 0.722 - 0.4 = 0.322$

$I(\text{Class; Animated}) = H(\text{Class}) - H(\text{class} \mid \text{Animated}) = 0.722 - 0.551 = 0.171$

$\because I(\text{class; colorful}) > I(\text{class; Animated})$

$\therefore$ We should split on the Animated feature on the second split

$\therefore$ The decision tree should be like:



Problem 2 (15pts) Computational Graph(CG) and Backpropagation

Consider the following classification MLP with one hidden layer:

$$
\begin{aligned}
\boldsymbol{x} &= \text{ input } \in \mathbb{R}^D \\
\boldsymbol{z} &= \mathbf{W}\boldsymbol{x} + \boldsymbol{b}_1 \in \mathbb{R}^K \\
\boldsymbol{h} &= \text{ReLU}(\boldsymbol{z}) \in \mathbb{R}^K \\
\boldsymbol{a} &= \mathbf{V}\boldsymbol{h} + \boldsymbol{b}_2 \in \mathbb{R}^C \\
\mathcal{L} &= \text{CrossEntropy}(\boldsymbol{y}, \text{softmax}(\boldsymbol{a})) \in \mathbb{R}
\end{aligned}
$$

where $\boldsymbol{x} \in \mathbb{R}^D, \boldsymbol{b}_1 \in \mathbb{R}^K, \mathbf{W} \in \mathbb{R}^{K \times D}, \boldsymbol{b}_2 \in \mathbb{R}^C, \mathbf{V} \in \mathbb{R}^{C \times K}$, where D is the size of the input, $\boldsymbol{y}$ is the groundtruth, K is the number of hidden units, and C is the number of classes.
hint : $\text{ReLU}(a) = \max(a, 0) = a\mathbb{I}(a > 0)$, where $\mathbb{I}(e)$ is the indicator function.

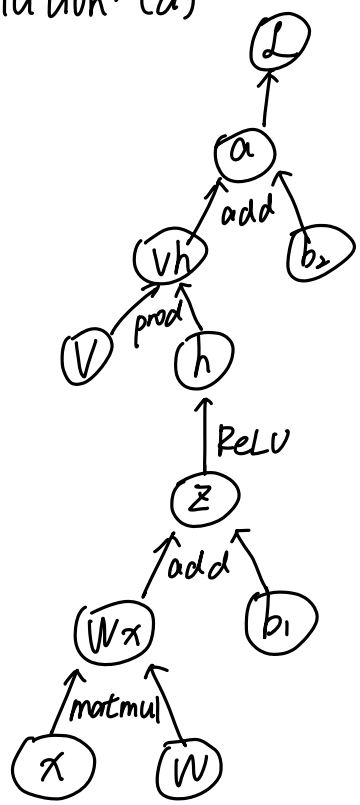
$$
\mathbb{I}(e) = \begin{cases} 1 & \text{if } e \text{ is true} \\ 0 & \text{if } e \text{ is false} \end{cases}
$$

You can use the fact that $\text{ReLU}'(a) = H(a)$, where $H(a)$ is the Heaviside step function $H(a) = \mathbb{I}(a > 0)$.

(a) Please plot the computational graph for the loss function $\mathcal{L}$, i.e. draw the computational graph of forward pass.

(b) Derive the backward pass, i.e. calculate parameter derivative of each layer, including $\nabla_{\mathbf{V}}\mathcal{L}, \nabla_{b_2}\mathcal{L}, \nabla_{\mathbf{W}}\mathcal{L}, \nabla_{b_1}\mathcal{L}$.

Solution: (a)



(b) Backward:

$$\nabla_a L = \frac{\partial L}{\partial a} = \frac{\partial}{\partial a} \text{CrossEntropy}(y, \text{softmax}(a))$$

$$\nabla_V L = \frac{\partial L}{\partial V} = \frac{\partial L}{\partial a} \cdot \frac{\partial a}{\partial V} = \nabla_a L \cdot h^T$$

$$\nabla_{b_2} L = \frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial a} \cdot \frac{\partial a}{\partial b_2} = \nabla_a L$$

$$\nabla_h L = \frac{\partial L}{\partial h} = \frac{\partial L}{\partial a} \cdot \frac{\partial a}{\partial h} = \nabla_a L \cdot V$$

$$\nabla_z L = \frac{\partial L}{\partial z} = \frac{\partial L}{\partial h} \cdot \frac{\partial h}{\partial z} = \nabla_h L \odot \text{ReLU}'(z)$$

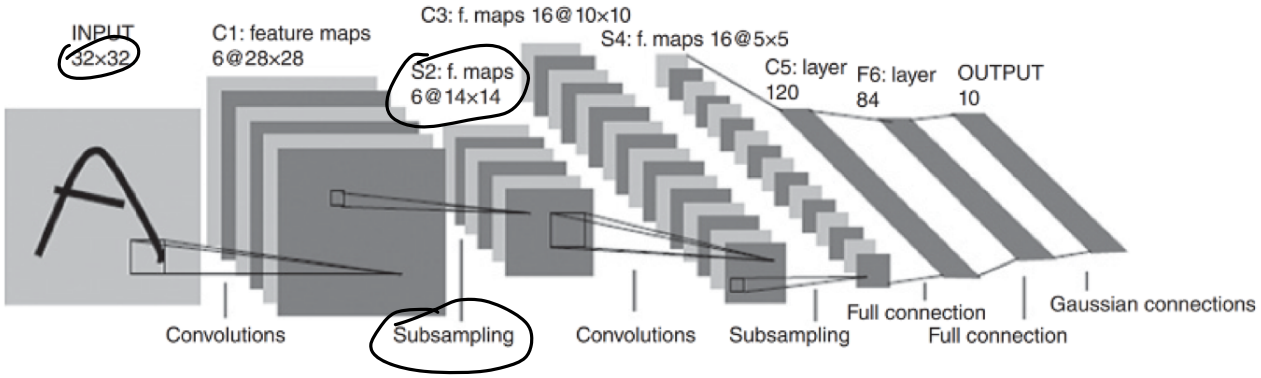
$$\nabla_W L = \frac{\partial L}{\partial W} = \frac{\partial L}{\partial z} \cdot \frac{\partial z}{\partial W} = \nabla_z L \cdot x^T$$

$$\nabla_{b_1} L = \frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial z} \cdot \frac{\partial z}{\partial b_1} = \nabla_z L$$

We can use these gradients in the backward propagation algorithm to update the model parameters.

Problem 3 (12pts) CNN

Consider the CNN network shown below with two convolution layers (C1 and C3) and two subsampling (max-polling) layers (S2 and S4). During convolution, we assume stride = 1 with no padding. The filter size at each layer is given, where $6@28 \times 28$ means six filter used to generate feature maps with 28×28 each. The full connection layers (C5 and F6) are specified with neuron counts. The input layer takes a 32×32 image filter to each neuron in the C1 layer. The output layer has 10 neurons.



(a) Determine the size of the convolutional filter used in the first layer C1, which equals to the number of input signals that each neuron in C1 receives. Justify your answer.

(b) How many neurons are needed in the convolutional layer C1?

(c) Given the 2×2 filter in the subsampling layer S2, what is the stride distance required for this filter? How many neurons are required in the S2 layer?

(d) What are the purposes in using the hidden layers for convolution and pooling? What are the purposes in using the fully connected layers at the output end?

Solution: (a) The receptive field size of each neuron in the C1 layer is 5×5. Because to cover a 32×32 input image with a 5×5 filter without padding, the filter has to move (32-5+1)=28 steps in both width and height directions. Therefore, each neuron in the C1 layer receives input from a 5×5 region in the input image.

(b) $6 \times 28 \times 28 = 4704$

∴ 4704 neurons are needed in the convolutional layer C1.

(c) ∵ output size = (N-F) / stride +1 ,

and ∵ output size = 14, N=28, F=2, ∴ 14 = 26 / stride +1

∴ stride = 2

∴ The stride distance required for this filter is 2.

$6 \times 14 \times 14 = 1176$

∴ 1176 neurons are required in the S2 layer.

(d) Purposes in using hidden layers for convolution and pooling:

① Enable the network to automatically learn hierarchical representations of the input data. These representations capture meaningful patterns and features at different levels of abstraction, facilitating effective

learning and generalization to new, unseen data.

② Allow the network to gradually build increasingly complex and discriminative features, leading to improved performance on various tasks.

Problem 4 (11pts) More CNN

(a) List two reasons we typically prefer convolutional layers instead of fully connected layers when working with image data.

(b) Consider the following 1D signal: $[1, 4, 0, -2, 3]$. After convolution with a length-3 filter, no padding, stride $= 1$, we get the following sequence: $[-2, 2, 11]$. What was the filter?

(c) Transpose convolution is an operation to help us upsample a signal (increase the resolution). For example, if our original signal were $[a, b, c]$ and we perform transpose convolution with pad $= 0$ and stride $= 2$, with the filter $[x, y, z]$, the output would be $[ax, ay, az+bx, by, bz+cx, cy, cz]$. Notice that the entries of the input are multiplied by each of the entries of the filter. Overlaps are summed. Also notice how for a fixed filtersize and stride, the dimensions of the input and output are swapped compared to standard convolution. (For example, if we did standard convolution on a length-7 sequence with filtersize of 3 and stride $= 2$, we would output a length-3 sequence).
If our 2D input is $\begin{bmatrix} -1 & 2 \\ 3 & 1 \end{bmatrix}$ and the 2D filter is $\begin{bmatrix} +1 & -1 \\ 0 & +1 \end{bmatrix}$ What is the output of transpose convolution with pad $= 0$ and stride $= 1$?

(a) ① Local connectivity: convolutional layers exploit the spatial locality of the data. Unlike fully connected layers, which connect each neuron to every neuron in the previous layer, convolutional layers have local receptive fields. This means that each neuron in a convolutional layer is only connected to a small region of the input data, allowing the network to capture local patterns and features. This local connectivity significantly reduces the number of parameters in the network, making it more efficient and scalable.

② Parameter sharing: In convolutional layers, the same set of weights (filters) is applied across the entire input data. This greatly reduces the number of parameters that need to be learned, making the model more robust and less prone to overfitting.

(b) Assume the filter is $h = [h_1, h_2, h_3]$

$\therefore h_1 \times 1 + h_2 \times 4 + h_3 \times 0 = -2,\ h_1 \times 4 + h_2 \times 0 + h_3 \times (-2) = 2,\ h_1 \times 0 + h_2 \times (-2) + h_3 \times 3 = 11$

$\Rightarrow h_1 + 4h_2 = -2,\ 4h_1 - 2h_3 = 2,\ -2h_2 + 3h_3 = 11$

$\Rightarrow h_1 = 2,\ h_2 = -1,\ h_3 = 3$

$\therefore$ The filter used in the convolution operation is $h = [2, -1, 3]$

(c) The output will be a 3x3 matrix.

$$-1\begin{bmatrix} +1 & -1 & 0 \\ 0 & +1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + 2\begin{bmatrix} 0 & +1 & -1 \\ 0 & 0 & +1 \\ 0 & 0 & 0 \end{bmatrix} + 3\begin{bmatrix} 0 & 0 & 0 \\ +1 & -1 & 0 \\ 0 & +1 & 0 \end{bmatrix} + 1\begin{bmatrix} 0 & 0 & 0 \\ 0 & +1 & -1 \\ 0 & 0 & +1 \end{bmatrix}$$

$$= \begin{bmatrix} -1 & +1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & +2 & -2 \\ 0 & 0 & +2 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ +3 & -3 & 0 \\ 0 & +3 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & +1 & -1 \\ 0 & 0 & +1 \end{bmatrix} = \begin{bmatrix} -1 & 3 & -2 \\ 3 & -3 & 1 \\ 0 & 3 & 1 \end{bmatrix}$$

$\therefore$ the output of transpose convolution is $\begin{bmatrix} -1 & 3 & -2 \\ 3 & -3 & 1 \\ 0 & 3 & 1 \end{bmatrix}$