non-linear dataset
XOR func^n

Quiz Revision Session

single-layer perceptron implements the linear decision boundary.

1. A single-layer perceptron with a hard step activation is trained on a dataset that is not linearly separable. Which of the following is the most accurate statement about its representational limitation?

- A. It will converge to a solution that minimizes mean squared error on the training set.
- B. It can represent any Boolean function if trained long enough.
- C. It cannot represent the XOR function due to linear separability limits. ✓
- D. It can represent XOR only if the learning rate is sufficiently small.

XOR func^n is not linearly separable. Even after long-time of training and scheduling the learning rate won't help.

2. Consider a 3-layer MLP with ReLU activations trained with cross-entropy loss. Which of the following choices increase the risk of vanishing/exploding gradients during training?

- A. Using very deep networks without skip connections. ✓
- B. Using sigmoid activations in hidden layers with poor weight initialization. ✓
- C. Using ReLU activations with He (Kaiming) initialization. → helps
- D. Using Batch Normalization between linear and activation layers.

3 layer MLP
→ ReLU Activation
→ CE loss

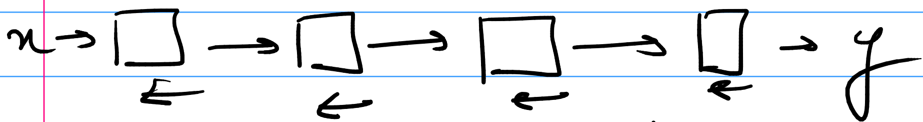
D ↪ helps to stabilize the layer stats ⇒ stabilizing gradient

Sigmoids

tend to saturate
⇓
small derivatives ⇒ vanishing gradients

A → if the networks depth ↑, while the skip connec^ns are not present ⇒ that lead to van./explo. gradient likelihood

AlexNet — 8 layers
ResNet — 152 layers



You train a classifier with softmax + cross-entropy. For a single sample, the model logits are [2,0,−1] for classes [0,1,2], and the true class is 0.

3. Compute the cross-entropy loss (natural log) to 4 decimal places.

$[2, 0, -1] \longrightarrow [0, 1, 2]$

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum e^{z_i}}$$

$e^2 = 7.389$

$e^0 = 1$ $\qquad$ $\text{sum} = \sum e^{z_i} = 8.7569$

$e^{-1} = 0.3679$

$$P(\text{true class} = 0) = \frac{7.389}{8.7569} \approx 0.8438$$

cross entropy loss →

$$L = -\ln(0.8438) = 0.1691$$

4. Which statement best captures the role of an activation function in an MLP?

- A. It guarantees convexity of the loss landscape.
- ✓ B. It introduces nonlinearity, enabling composition of linear transforms to model complex functions.
- C. It primarily serves as a learning rate schedule.
- D. It regularizes the model by dropping activations.

→ sigmoid, Relu, softmax, Leaky Relu, tanh

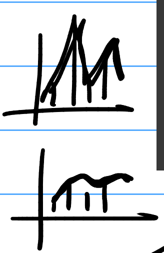
→ if activation function is absent, MLP ⇒ just becomes a chain of linear transformation.

→ activation function ⇒ learning complex, and non-linear patterns in the data.

5. You apply L2 weight decay, Dropout(p=0.5), and early stopping. Which statements are true about generalization? → regularization

- ✓ A. L2 penalizes large weights, typically smoothing decision boundaries.
- ✓ B. Dropout behaves like model averaging over sub-networks at train time. → dropout never used in eval
- ✓ C. Early stopping often selects a point with higher training loss but lower test loss than later epochs.
- D. Dropout increases the effective width at test time.

Train ⇓ empirical test ⇓ generalization



L2 weight → shrink large wts. (penalizing), it helps to prevent overfitting ⇒ improves generalization

Dropout → randomly deactivating some neurons during training time only ⇒ ensemble approach, reduces the training bias ⇒ improves generalization

inactive during test time, it won't add or remove any neurons

ES → to minimize test loss.

↳ Stable stats in btw. the layers

6. Batch Normalization primarily helps by:

- A. Eliminating the need for nonlinear activations. ✗
- B. Reducing internal covariate shift and stabilizing gradients across layers. ✓
- C. Guaranteeing faster test-time inference. ✗
- D. Replacing the optimizer's momentum term. ✗

BN → helps to have stable mean & var. within each mini-batch
→ make training stable
→ it will work on range of η (higher)
→ reduce the risk of vanishing/exploding gradients

7. You initialize a fully-connected layer with He normal initialization for ReLU: weights W ~ N(0, 2/fan_in). If fan_in = 50, what is the standard deviation (upto 2 decimal places).

$N \sim (\mu, \sigma^2)$ $\qquad \sigma^2 = \frac{2}{50} = 0.04$

$\sigma = 0.2$

→

8. Comparing SGD with momentum vs Adam, which statements are correct in typical deep learning practice?

- A. Adam adapts learning rates per-parameter using first and second moment estimates. ✓
- B. Momentum SGD cannot converge without weight decay. ✗
- C. Adam is often more robust to poorly scaled gradients at initialization.
- D. Adam always outperforms SGD on final generalization.

$\underbrace{\text{Adam} = \text{Adagrad} \geq \text{Adadelta} > \text{Momentum based SGDs} / \text{NAG} > \text{SGD}}_{}$

$E[X] \to 1^{st}$ moment
$E[X^2] \to 2^{nd}$ moment

general hierarchy
↳ how these optimizers perform.

→ Adam → MA of gradients (1^{st} moment) & squared gradients (2^{nd} moment)
↳ it adjust parameters accordingly

SGD with momentum → it can converge w/o wt. decay.

wt decay → "another regularizer" technique ⇒ ~~not~~ essential to guarantee convergence.

Adam outperforms other optimizer ~~init~~ techniques because it is robust to init. ⇒ It automatically scales based on the current gradient magnitudes.

In cases of CNNs, SGD with momentum is better than ADAM.

Adam mostly converges faster than other techniques ; not always outperforms them.

10. A two-layer MLP (no bias) is $f(x) = W_2\,\mathrm{ReLU}(W_1 x)$. With

$W_1 = \begin{bmatrix} 1 & -2 \\ -3 & 4 \end{bmatrix}$, $W_2 = [2 \;\; -1]$, and $x = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$.

Compute $f(x)$.

$$W_1 = \begin{bmatrix} 1 & -2 \\ -3 & 4 \end{bmatrix} \qquad W_2 = [2 \;\; -1]$$

$$x = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

$$W_1 x = \underset{2\times 2}{\begin{bmatrix} 1 & -2 \\ -3 & 4 \end{bmatrix}} \underset{2\times 1}{\begin{bmatrix} 2 \\ 1 \end{bmatrix}} = \begin{bmatrix} 0 \\ -2 \end{bmatrix}$$

$$\mathrm{Relu} = [0, -2] = [0, 0]$$

$$f(x) = 0$$

Naive

9. For a binary classifier with logits z and sigmoid outputs $\sigma(z)$, which loss formulation is numerically stable?

- A. $-y\log\sigma(z) - (1-y)\log(1-\sigma(z))$ computed directly from $\sigma(z)$. → Naive BCE loss $\sigma(z)$ → unstable, not generally used
- B. $\max(z,0) - z\cdot y + \log(1+e^{-|z|})$. → BCE loss (Relu − logits · output + log(σ))
- C. Mean squared error between y and $\sigma(z)$.
- D. Hinge loss on $\sigma(z)$. 1^{st}

$-y\log\sigma(z) - [(1-y)\log(1-\sigma(z))]$

↓ overflow of loss or cancellation.

Regressions → MSE / MAE → Hinge loss

Classification → Sigmoid → BCE loss (Binary cross entropy loss)

1. With what padding does a 1×1 filter with stride 1 does not change the output width and height?

$$O = \frac{W - F + 2P}{S} + 1$$

W → input

F → filter

P → padding

S → stride

F = 1

S = 1

$$O = \frac{W - 1 + 2P}{1} + 1 = W$$

$$W - \cancel{1} + 2P + \cancel{1} = W$$

$$P = 0$$

2. Which of the following would you consider to be a valid activation functions (to apply element wise non-linearity) to train a neural network?

- $-\min(2,x)$ → Clips the values & flips the sign
- $0.9x+1$ → linear x
- ✓ $f(x) = \begin{cases}\min(x,\ 0.1x) & x \geqslant 0 \\ \min(x,\ 0.1x) & x < 0\end{cases}$
- $f(x) = \begin{cases}\max(x,\ 0.1x) & x \geqslant 0 \\ \min(x,\ 0.1x) & x < 0\end{cases}$

min (x, 0.1x) x ≥ 0

min (x, 0.1x) x < 0

C = 0.1

max(0, x)

c. ✓ x = 5 min(5, 0.5) = 0.5

✓ x = −5 min(−5, −0.5) = −5

d. max(5, 0.5) = 5

min(−5, −0.5) = −5

a does not introduce non-linearity

Q3 - Q6

Consider the convolutional neural network defined by the layers in the left column below. Fill in the shape of the output volume and the number of parameters at each layer. You can write the activation shapes in the format (H, W, C), where H, W, C are the height, width and channel dimensions, respectively. Unless specified, assume padding 1, stride 1 where appropriate. Do not ignore biases.

Notation:

- CONV $F - K$ denotes a Convolution layer with K filter each of height and width equal to F.
- POOL N denotes a $N \times N$ max-pooling layer with stride N and padding 0
- FLATTEN denotes the task of flattening the input. Works same as `torch.nn.flatten`.
- FC N denotes the fully connected layer with N neurons

Layer	Activation Volume Dimensions	Number of parameters
Input	32 × 32 × 3	0
CONV3-8	3×3, 8, 1, 1	
Leaky ReLU	No Δ	
POOL-2	2×2, S=2, P=0	
CONV3-16	3×3, 16, 1, 1	
Leaky ReLU	No Δ	
POOL-2	2×2, S=2	
FLATTEN		
FC-10	FC 10 Neurons	

3. How many parameters are there in CONV 3 — 8 layer?

$$\boxed{\text{Parameters} = (F \times F \times \text{Input chan.}) \times \text{no. of filter} + \text{bias per filter}}$$

no. of params for each layer

$F = 3 \times 3$

$\text{Input}_c = 3$

no. $= 8$

$$\begin{array}{r} 27 \\ \times 8 \\ \hline 216 \end{array}$$

$= (3 \times 3 \times 3) \times 8 + 8$

$= 27 \times 8 = 216 + 8 = \underline{\underline{224}}$

4. What is the size of the output after applying **first POOL** 2 layer? Note that we are asking for the size after applying this layer that is the size that will be feed as an input to CONV $3 - 16$ layer.

- ☑ $16 \times 16 \times 8$
- $14 \times 14 \times 8$
- $16 \times 16 \times 16$
- $15 \times 15 \times 18$

① Conv 3 - 8

Input $= 32 \times 32 \times 3$

Filters $= 3 \times 3$

$P = 1$

$S = 1$

$O = \dfrac{W - F + 2P}{S} + 1$

$= \dfrac{32 - 3 + 2(1)}{1} + 1$

$= 31 + 1 = 32$

$\Rightarrow \boxed{32 \times 32 \times 8}$

② Pool 2

Filter $= 2\times2$

$S = 2$

$P = 0$

$$O = \frac{32-2}{2} + 1$$

$$= 15 + 1 = 16$$

$$\Rightarrow \boxed{16\times16\times8}$$

5. How many parameters are there for the CONV $3-16$ layer?

Input $= 16\times16\times8$

Filter size $= 3\times3$

Input C. $= 8$

Filters $= 16$

72
16

432
72×

1152

$$(3\times3\times8)\cdot16 + 16 = 72\times16 + 16$$

$$= 1152 + 16$$

$$= \boxed{1168} \rightarrow$$ no. of params

6. How many parameters are there in the second POOL 2 layer?

↳ there no learnable params in pooling layers ⇒ 0

7. You are solving the binary classification task of classifying images as cat (labelled as 1) vs. non-cat (labelled as 0). You design a CNN with a single output neuron. Let the output of this neuron be z. The final output of your network, $\hat{y}$ is given by:

$$\hat{y} = \sigma(ReLU(z))$$

$\hat{y} \geq 0.5$

You classify all inputs with a final value $\hat{y} \geq 0.5$ as cat images. If your training dataset contains a total of 100 images in which 60 images are of cat then what is the accuracy of your model? Write your answer in percentage.

Rule → $\hat{y} \geq 0.5$ ⇒ predicted cat.

size $= 100$

actual cat $= 60$

100 ⟹ all are cats

$$\text{accuracy} = \frac{\text{correct predict}}{\text{total}} = \frac{60}{100} = \underline{60}$$

$\sigma(0) = 0.5$ $\quad$ + ve value > 0

$\sigma(\text{+ve value}) \geq 0.5$

8. Which of the following statements are trur about Mcculloch Pitts neurons? Here representation of function means we are able to classify all data points correctly for that function.

- [x] It only accepts boolean inputs ✓
- [ ] It gives different importance to each features ✗ the wts. were not present
- [ ] It can represent any boolean function ✗ XOR (first example)
- [ ] It can represent ~~all linearly~~ seperable boolean function ✗ w/o thresholding can't do that.

For a 2 layer network with ReLU activation

$$a = Wx + b, \quad h = ReLU(a), \quad y = v^T h$$

If $x \in \mathbb{R}^2$, $W \in \mathbb{R}^{3\times 2}$ then answer the following questions

[NAT]

10. If $v \in \mathbb{R}^n$ then what is the value of n?

$a = Wx + b, \quad h = Relu(a), \quad y = v^T h$ → single value

$x \in R^2, \quad W \in R^{3\times 2}$ $\qquad$ $Wx + b_{3\times 1}$

$v \in R^n, \quad n?$ $\qquad$ $\underbrace{W_{3\times 2} \; x_{2\times 1}}_{3\times 1}$

$a \in R^3$

$\overset{v}{[\;]_{3\times 1}} [\;]_{3\times 1}$ $\quad h = Relu(a) \in R^3$

$v^T h \Rightarrow$ dot product $\Rightarrow v \in R^3$

$n = 3$

$[\;]_{1\times 3} [\;]_{3\times 1} = {}_{1\times 1}[\;]$

11. Which of the below expression is for $\frac{\partial y}{\partial x}$? $\odot$ denotes the element wise multiplication

- $W\left(v \odot \frac{\partial h}{\partial a}\right)$
- $(W^T v) \odot \frac{\partial h}{\partial a}$
- $W^T\left(v^T \frac{\partial h}{\partial a}\right)$
- ✓ $W^T\left(v \odot \frac{\partial h}{\partial a}\right)$

$a = Wx + b$

$\downarrow$

$h = \text{Relu}(a)$

$\downarrow$

$y = v^T h$

$\frac{\partial y}{\partial x} = ?$

$\frac{\partial y}{\partial h}$ $\quad y = v^T h \Rightarrow \frac{\partial y}{\partial h} = v$

$\frac{\partial h}{\partial a}$ $\quad h = \text{Relu}(a)$

$\frac{\partial h}{\partial a} = (\text{Relu}'(a))$

$\frac{\partial h}{\partial a} \Rightarrow v \odot \frac{\partial h}{\partial a}$

$\frac{\partial a}{\partial x}$ $\quad a = Wx + b \Rightarrow \frac{\partial a}{\partial x} = W$

$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial h} \frac{\partial h}{\partial a} \frac{\partial a}{\partial x}$$

$$= W^T\left(v \odot \frac{\partial h}{\partial a}\right)$$