Torch - Pytorch - Deep Learing (ANN,CNN,RNN) - Interview-Questions-Answers

1. What is PyTorch?

PyTorch is an open-source deep learning framework developed by Facebook for building and training neural networks.

2. What is a Tensor in PyTorch?

A Tensor is a multidimensional array similar to NumPy arrays but optimized for GPU acceleration.

3. How to import PyTorch?

```
import torch
```

4. Difference between NumPy arrays and PyTorch tensors?

Tensors can run on GPU for faster computation, whereas NumPy arrays run only on CPU.

5. How to create a tensor in PyTorch?

Using `torch.tensor()` , `torch.zeros()` , `torch.ones()` , or `torch.rand()` .

6. How to check tensor device and datatype?

Use `.device` and `.dtype` attributes.

7. How to move a tensor to GPU?

```
tensor = tensor.to('cuda')
```

8. How to get the shape of a tensor?

Use `tensor.shape` or `tensor.size()` .

9. How to create a tensor with random numbers?

`torch.rand(size)` or `torch.randn(size)` .

10. How to create identity matrix in PyTorch?

`torch.eye(n)` creates an `n × n` identity matrix.

11. What is autograd in PyTorch?

It's the automatic differentiation engine that computes gradients for backpropagation.

12. How to enable gradient tracking?

Set `requires_grad=True` while creating the tensor.

13. How to stop gradient tracking temporarily?

Use `with torch.no_grad():` block.

14. What is the difference between `torch.Tensor()` and `torch.tensor()` ?

`torch.tensor()` copies data; `torch.Tensor()` can create uninitialized tensors (not recommended).

15. How to get the number of elements in a tensor?

Use `tensor.numel()` .

16. How to reshape a tensor?

Use `tensor.view()` or `tensor.reshape()` .

17. How to concatenate tensors?

Use `torch.cat((t1, t2), dim=axis)` or `torch.stack()` .

18. What is broadcasting in PyTorch?

It allows automatic expansion of tensors with different shapes for element-wise operations.

19. What is the difference between `.view()` and `.reshape()` ?

`view()` requires contiguous memory; `reshape()` can handle non-contiguous tensors.

20. How to compute dot product or matrix multiplication?

Use `torch.dot()` for vectors or `torch.mm()` / `@` for matrices.

21. How to compute transpose of a tensor?

Use `tensor.T` or `tensor.transpose(0,1)` .

22. What are common activation functions in PyTorch?

ReLU, Sigmoid, Tanh, Softmax — available in `torch.nn.functional` .

23. What is a computational graph?

A dynamic structure in PyTorch that records tensor operations for gradient calculation.

24. What is the purpose of `zero_grad()` ?

It resets gradients to zero before backpropagation in each iteration.

25. What's the difference between `torch.save()` and `torch.load()` ?

`torch.save()` serializes and saves tensors/models; `torch.load()` restores them.

26. Create a 3×3 tensor of random numbers.

```
import torch
a = torch.rand(3,3)
print(a)
```

27. Create a tensor filled with zeros and ones.

```
zeros = torch.zeros(2,3)
ones = torch.ones(2,3)
print(zeros, ones)
```

28. Convert a Python list to a PyTorch tensor.

```
lst = [1,2,3,4]
t = torch.tensor(lst)
print(t)
```

29. Create a tensor with specific dtype (float32).

```
t = torch.tensor([1,2,3], dtype=torch.float32)
print(t.dtype)
```

30. Get tensor shape, size, and number of elements.

```
a = torch.rand(2,3,4)
print(a.shape, a.size(), a.numel())
```

31. Perform element-wise addition and multiplication.

```
a = torch.tensor([1,2,3])
b = torch.tensor([4,5,6])
print(a + b)
print(a * b)
```

32. Matrix multiplication (3×3 matrices).

```
A = torch.randint(1,10,(3,3))
B = torch.randint(1,10,(3,3))
print(A @ B)  # or torch.mm(A, B)
```

33. Find transpose of a tensor.

```
A = torch.arange(9).reshape(3,3)
print(A.T)
```

34. Compute mean, max, min, sum.

```
a = torch.arange(1,6)
print(a.mean(), a.max(), a.min(), a.sum())
```

35. Reshape and flatten a tensor.

```
a = torch.arange(9)
print(a.view(3,3))
print(a.flatten())
```

36. Stack tensors vertically and horizontally.

```
x = torch.ones(2,3)
y = torch.zeros(2,3)
print(torch.cat((x,y), dim=0))  # vertical
print(torch.cat((x,y), dim=1))  # horizontal
```

37. Find indices of non-zero elements.

```
a = torch.tensor([0, 2, 0, 5, 0, 7])
print(torch.nonzero(a))
```

38. Create identity matrix and diagonal matrix.

```
I = torch.eye(4)
D = torch.diag(torch.tensor([1,2,3,4]))
print(I, D)
```

39. Compute dot product between two vectors.

```
a = torch.tensor([2,3,4])
b = torch.tensor([1,5,2])
print(torch.dot(a, b))
```

40. Compute matrix inverse and determinant.

```
A = torch.rand(3,3)
print(torch.inverse(A))
print(torch.det(A))
```

41. Apply element-wise exponential, sqrt, log.

```
x = torch.tensor([1., 4., 9.])
print(torch.sqrt(x), torch.exp(x), torch.log(x))
```

42. Normalize a tensor.

```
a = torch.rand(5)
a_norm = (a - a.min()) / (a.max() - a.min())
print(a_norm)
```

43. Create a random tensor and move to GPU if available.

```
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
a = torch.rand(3,3).to(device)
print(a.device)
```

44. Enable gradient computation and perform backward pass.

```
x = torch.tensor([2.0, 3.0], requires_grad=True)
y = (x**2).sum()
y.backward()
print(x.grad)
```

45. Use `torch.no_grad()` to disable gradient tracking.

```
x = torch.tensor([3.0], requires_grad=True)
with torch.no_grad():
    y = x * 2
print(y.requires_grad)  # False
```

46. Compute cosine similarity between two tensors.

```
a = torch.rand(3)
b = torch.rand(3)
cos_sim = torch.nn.functional.cosine_similarity(a, b, dim=0)
print(cos_sim)
```

47. Save and load a tensor.

```
t = torch.rand(3,3)
torch.save(t, 'tensor.pt')
loaded = torch.load('tensor.pt')
print(loaded)
```

48. Compute row-wise and column-wise sum.

```
A = torch.arange(6).reshape(2,3)
print(A.sum(dim=0))  # column
print(A.sum(dim=1))  # row
```

49. Compare two tensors element-wise.

```
a = torch.tensor([1,2,3])
b = torch.tensor([1,1,3])
print(a == b)
print(torch.equal(a,b))
```

50. Convert between PyTorch tensor and NumPy array.

```
a = torch.tensor([1,2,3])
np_arr = a.numpy()
```