1. How does unsqueeze help us to solve certain broadcasting problems?

**unsqueeze adds a new dimension to a tensor. It helps with broadcasting by increasing the tensor's dimensionality, allowing it to match the shape of another tensor for elementwise operations. For example, it can be used to turn a 1D tensor into a 2D tensor for broadcasting with a higher-dimensional tensor.**

2. How can we use indexing to do the same operation as unsqueeze?

**We can achieve the same operation as unsqueeze using indexing by adding a new axis with None or np.newaxis. For example:**

**import numpy as np**

**a = np.array([1, 2, 3])**

**b = a[:, np.newaxis] # Adds a new axis to 'a', making it a 2D array**

3. How do we show the actual contents of the memory used for a tensor?

**To view the actual contents of memory used by a tensor, you can use the .data\_ptr() method in PyTorch or the .data attribute in TensorFlow. However, accessing the raw memory contents should be done with caution.**

4. When adding a vector of size 3 to a matrix of size 3×3, are the elements of the vector added

to each row or each column of the matrix? (Be sure to check your answer by running this

code in a notebook.)

**When adding a vector of size 3 to a matrix of size 3x3, the elements of the vector are added to each row of the matrix. The vector is effectively broadcasted along the rows of the matrix.**

5. Do broadcasting and expand\_as result in increased memory use? Why or why not?

**Broadcasting and expand\_as themselves do not result in increased memory use. They operate on the existing data without creating additional memory copies. However, if new tensors are created as a result of these operations, memory may be allocated for the new tensors.**

6. Implement matmul using Einstein summation.

**Here's an example of implementing matrix multiplication using Einstein summation notation in NumPy:**

**import numpy as np**

**A = np.random.rand(2, 3) # A 2x3 matrix**

**B = np.random.rand(3, 4) # A 3x4 matrix**

**C = np.einsum("ij,jk->ik", A, B) # Matrix multiplication using Einstein summation**

7. What does a repeated index letter represent on the lefthand side of einsum?

**A repeated index letter on the left-hand side of einsum represents summation over that index. It indicates that the values along that axis should be summed or contracted when performing the operation specified in the right-hand side.**

8. What are the three rules of Einstein summation notation? Why?

**Repeated Indices: Repeated indices are summed over.**

**Index Labels: Indices must have the same label in both the left-hand and right-hand sides of einsum.**

**Output Indices: All indices in the right-hand side must be present in the output.**

9. What are the forward pass and backward pass of a neural network?

**The forward pass is the process of feeding input data through the neural network to compute predictions or activations.**

**The backward pass (backpropagation) is the process of computing gradients of the loss with respect to the network's parameters. It is used for training the network through gradient descent.**

10. Why do we need to store some of the activations calculated for intermediate layers in the

forward pass?

**Intermediate layer activations are stored in the forward pass for later use in the backward pass during backpropagation. These activations are needed to compute gradients efficiently and update the network's weights.**

11. What is the downside of having activations with a standard deviation too far away from 1?

**Activations with a standard deviation too far from 1 can lead to issues in training neural networks. If the standard deviation is too high, it can cause exploding gradients, making optimization difficult. If it's too low, it can lead to vanishing gradients, hindering learning.**

12. How can weight initialization help avoid this problem?

**Weight initialization techniques like He initialization and Xavier initialization are used to set the initial weights of neural networks. These techniques help control the standard deviation of activations, bringing them closer to 1 and mitigating the problems associated with extreme standard deviations during training.**