

AD Project Dry Questions

Question 1

- a) The Universal Approximation Theorem (UAT) states that a neural network with a single hidden layer can approximate any continuous function on a compact subset of R^n , given enough neurons and a suitable activation function. This implies that a single-layer MLP (with sufficient neurons) theoretically has the capacity to achieve the best possible accuracy on any dataset by approximating the optimal function.
- b) While a single hidden layer can approximate any continuous function, achieving this for complex functions or high-dimensional inputs may require an impractically large number of neurons. Deeper networks allow for efficient approximations by using fewer neurons and capturing hierarchical features. Multiple layers also facilitate better generalization by breaking down complex functions into simpler, stacked components. Furthermore, a wide single-layer network is often more resource-intensive and may require more data and computational power, making deeper networks a more practical choice in most cases.

Question 2

- a) The primary advantage of using a CNN for image classification over an MLP is that CNNs are structured to recognize spatial hierarchies and local patterns through convolutional layers. These layers use small filters that capture local features, such as edges and shapes, enabling CNNs to recognize patterns independent of their specific location in an image. Additionally, CNNs benefit from weight sharing and local receptive fields, making them efficient in parameter use and effective at capturing translational invariance. In contrast, an MLP treats each pixel as an independent feature, resulting in the loss of spatial structure and potentially poorer learning of image patterns. If Alice uses an MLP, she will face challenges in preserving the spatial structure of the image data. Flattening the image for input into an MLP discards the spatial relationships between pixels, making it difficult for the model to effectively learn from image data.
- b) Although the convolution operation is linear, there is a significant difference between using a convolutional layer and a fully connected linear layer in an MLP. Convolutional layers operate on local patches of the input, allowing the model to learn spatially localized features. Additionally, convolutional layers use weight sharing across different patches, capturing spatial dependencies and enabling efficient learning of hierarchical features. In contrast, MLPs treat every pixel independently, missing spatial relationships crucial for image recognition tasks.

Question 3

- a) Momentum is expected to assist in the optimization process, even with a convex loss surface. By incorporating a portion of the previous gradient, momentum accelerates convergence and stabilizes the optimization path, reducing oscillations in directions with varying gradients. This can be particularly helpful in cases where the gradient changes direction frequently. However, it's important to tune momentum carefully, as excessive momentum can lead to overshooting the minimum, causing oscillations around the optimal point instead of smooth convergence. Thus, while momentum can speed up optimization, careful tuning is essential to prevent overshooting.

Question 4

- a) PyTorch requires the loss tensor to be a scalar to perform backpropagation effectively. The primary reason is that backpropagation computes gradients by applying the chain rule, which requires a single scalar value to indicate the model's performance. A scalar loss provides a singular path for calculating gradients, ensuring clear direction for parameter updates. If the loss were a vector, it would introduce ambiguity as to which component to use for each parameter's gradient calculation. This scalar target unifies parameter adjustments, allowing the model to optimize effectively.
-

Question 5

- a) From the form of the network we can see that there is no bias parameter. Moreover from the Conv2d function, we can observe that there is one kernel of sized 32X32. This means that there are 1024 parameters to learn. We will denote the kernel as W , and the gradient of each parameter is calculated as follows:

$$\begin{aligned} L &= \sum_{i,j} (\hat{y}[i, j] - y[i, j])^2 \\ \frac{\partial L}{\partial W} &= \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial W} \\ \frac{\partial L}{\partial \hat{y}} &= 2 \cdot (\hat{y} - y) \\ \frac{\partial \hat{y}}{\partial W} &= x \\ \frac{\partial L}{\partial W} &= 2 \cdot (\hat{y} - y) \cdot x \\ \frac{\partial L}{\partial W} &= 2 \sum_{i,j} (\hat{y}[i, j] - y[i, j]) \cdot x[i, j], \text{ where } \hat{y}[i, j] \text{ is a scalar, but we are using broadcasting} \end{aligned}$$

Question 6

- a) Yes, it is possible to add positional embeddings to an RNN model. Positional embeddings encode the order of tokens in a sequence and can be added to the token embeddings before inputting them to the RNN layer. This can be done by either concatenating the positional embedding with the input embedding or adding them element-wise.
- b) While adding positional embeddings to an RNN is feasible, it is usually unnecessary. RNNs process data sequentially and maintain a hidden state that inherently captures the order of inputs. Unlike Transformers, which lack inherent sequence order, RNNs retain this information through sequential processing, making explicit positional encoding redundant in most cases.

Question 7

- a) Each pixel in the attention map represents the attention weight assigned by the model to a specific English (input) word when generating a French (output) word. Brighter pixels indicate higher attention, meaning the model focuses more on these word pairs during translation.

- b) Rows with only one non-zero pixel imply that the model relies on a single English word to translate a specific French word, indicating a clear one-to-one translation mapping.
- c) Rows with several non-zero pixels indicate that the model distributes attention across multiple English words for generating a single French word, suggesting that the translation requires contextual information from several words in the source sentence.
- d) Rows with only one white pixel have a high attention weight for a single source word, resulting in a focused, high-value attention signal. In contrast, rows with gray pixels represent lower, distributed attention across multiple words, indicating a need for context rather than a direct one-to-one translation.

Question 8

- a) The fundamental difference between a basic GAN and a Wasserstein GAN (WGAN) lies in their loss functions and training strategies. Basic GANs use a binary cross-entropy loss based on Jensen-Shannon divergence to distinguish real from generated data. This approach has limitations, including vanishing gradients, where the generator struggles to receive useful updates when the discriminator becomes highly effective. Basic GANs also suffer from mode collapse, where the generator produces limited variations in outputs, failing to capture the diversity of the real data distribution. WGAN addresses these issues by replacing the Jensen-Shannon divergence with the Wasserstein distance (Earth Mover's Distance). The Wasserstein distance provides smoother, non-vanishing gradients even when the real and generated distributions have little overlap. This results in more stable training and allows the generator to receive informative gradient feedback, encouraging it to explore a wider range of outputs and reducing mode collapse. In WGAN, the discriminator is replaced with a critic, which assigns a continuous scalar score to real and fake samples rather than classifying them as real or fake. This scalar scoring aligns with the Wasserstein distance and provides a more meaningful loss signal for the generator. To compute the Wasserstein distance, WGAN enforces a Lipschitz constraint on the critic. The original WGAN achieves this through weight clipping, but this approach has limitations, such as restricting the model's capacity. An improved version, WGAN-GP (Wasserstein GAN with Gradient Penalty), replaces weight clipping with a gradient penalty, ensuring a more stable and efficient enforcement of the Lipschitz condition. Overall, WGAN provides significant advantages, including smoother gradients, better training stability, and improved diversity in generated outputs. While it does not completely eliminate mode collapse, it reduces its likelihood and makes training less sensitive to hyperparameter tuning compared to basic GANs.

Question 9

- a) The decision to ignore the KL-divergence term is based on its non-negativity. Optimizing the ELBO, which is a lower bound on the log-likelihood, remains a valid training approach even without this term. Since the KL-divergence is often small or constant, excluding it minimally impacts the outcome.
- b) The KL-divergence term is generally intractable because it requires the true posterior $p(x_{0:T})$, which involves the exact joint distribution over latent and observed variables. Since this computation is infeasible, we approximate the posterior using a variational distribution $q(x_{1:T}|x_0)$, allowing us to focus on optimizing the ELBO directly.
- c) During training, we ignore the term $-D_{KL}(q(x_T|x_0) \parallel p_\theta(x_T))$ because x_T typically comes from a simple prior distribution. The KL-divergence between $q(x_T|x_0)$ and $p_\theta(x_T)$ is usually

constant or negligible, meaning it has little effect on gradients, simplifying the optimization process without compromising model performance.

Question 10

- a) Vanishing and exploding gradients are challenges in training deep neural networks.
Vanishing gradients occur when gradients become very small during backpropagation, often with narrow activation ranges like the sigmoid function. This slows or stops learning.
Exploding gradients happen when gradients grow excessively during backpropagation, causing instability and divergence.
- b) Vanishing gradients: With the sigmoid $f(x) = \frac{1}{1+e^{-x}}$, initialized with small weights (e.g., 0.5), a network's gradients can shrink to almost zero across layers. If each layer multiplies by 0.25, after five layers the gradient shrinks to 0.000976, causing slow updates.
Exploding gradients: For a network with 100 layers, initialized with weights of 5, gradients can exponentially grow. With each layer multiplication, the gradient reaches 5^{100} , leading to instability and potential divergence.
- c)
 - **MLP: ReLU activation** avoids vanishing gradients by maintaining gradients close to 1 for positive values, preventing them from diminishing through layers.
 - **CNN: Skip connections** allow gradients to bypass some layers, preserving strong signals in deep networks and preventing gradient decay.
 - **RNN: Gradient clipping** controls exploding gradients by setting a maximum gradient value, scaling down excessively large gradients to stabilize training.