**DL theory : Assingments-3**

1. It is generally not recommended to initialize all weights to the same value, even if that value is selected randomly using He initialization. This is because initializing all weights to the same value can lead to symmetry issues and make it difficult for the network to learn.
2. Initializing the bias terms to 0 is generally not a problem, but it can depend on the specific task and architecture of the network. In some cases, initializing the bias terms to small random values may be beneficial.
3. Three advantages of the SELU activation function over ReLU are:

* SELU can help to mitigate the vanishing and exploding gradients problem, as it is self-normalizing and can maintain the mean and variance of the activations.
* SELU is less likely to produce dead neurons, as it can maintain positive activations even when the input is negative.
* SELU can help to improve the performance of deep neural networks by enabling the network to perform well even when it has many layers.

1. SELU: it is useful when you want to maintain the mean and variance of the activations and prevent the vanishing and exploding gradients problem. Leaky ReLU: it can be useful when you want to avoid dead neurons, as it allows a small positive gradient when the input is negative. ReLU: it is useful when you want to introduce non-linearity in the network, but can lead to dead neurons if the input is negative. tanh: it can be useful when you want to introduce non-linearity in the network, but can saturate when the input is large. logistic: it can be useful when you want to predict a probability and the output is binary. softmax: it can be useful when you want to predict a probability and the output is multi-class.
2. If the momentum hyperparameter is set too close to 1, the optimizer may overshoot the optimal solution and fail to converge.
3. Three ways to produce a sparse model are:

* L1 regularization, which encourages the model to have sparse weights by adding a penalty term to the loss function for the absolute value of the weights.
* L2 regularization, which encourages the model to have sparse weights by adding a penalty term to the loss function for the square of the weights.
* Pruning, which involves removing the weights that have the smallest absolute values.

1. Dropout can slow down training, as it randomly drops out neurons during each iteration, reducing the effective number of neurons in the network. However, it does not slow down inference, as dropout is not applied during inference. MC Dropout can slow down inference, as it involves making multiple forward passes through the network with different dropout masks.
2. a. To build a DNN with 20 hidden layers of 100 neurons each, using He initialization and the ELU activation function, you can use the following code snippet in Keras:

from keras import layers model = keras.Sequential() for \_ in range(20): model.add(layers.Dense(100, kernel\_initializer='he\_normal', activation='elu'))

b. To train the network on the CIFAR10 dataset using Nadam optimization and early stopping, you can use the following code snippet:

from keras.datasets import cifar10 (X\_train, y\_train), (X\_test, y\_test) = cifar10.load\_data() model.add(layers.Dense(10, activation='

**Top of Form**