**Practice Questions for Mid-Semester Test**

**COMP3010 Machine Learning**

**Question 1 – Machine learning methods and problem formulation**

**You are developing a machine learning system to classify the images into two different classes: red images and green images.**

* **Name a suitable method for this task and justify your answer.**

Since we are developing an algorithm to classify images into just one of two classes, ‘red’ and ‘green’, we are dealing with a fairly simple image classification problem. Therefore, a suitable method is softmax regression as with softmax regression, we can train models for simple multiclass classification.

* **Use this example with the named method to describe the four important key components of a machine learning algorithm.**

The four key features of a machine learning algorithm are:

1. The *data* which we seek to learn from. In this case, the data is images.
2. A *model* to transform the data. In this case, the model is a convolutional neural network.
3. An *objective function* which quantifies the performance of the model. In this case, the objective function is cross-entropy loss.
4. An *algorithm* to adjust the model’s parameters to optimise the objective function. In this case, the algorithm is stochastic gradient descent (SGD).

**Question 2: Multilayer Perceptrons (MLPs)**

**Suppose you are implementing an MLP network with two hidden layers which share both their weights and biases, and a linear output layer, for image classification with 5 classes, and the input image size is 20x20.**

**2.1. Define the MLP model with** nn.Sequential()**.**

Since we are defining the model with nn.Sequential(), the softmax operation will be

defined within nn.Sequential().

shared = nn.Linear(400, 400)

net = nn.Sequential(nn.Flatten(),

shared, nn.ReLU(),

shared, nn.ReLU(),

nn.Linear(400, 5),

nn.Softmax())

**2.2. Write the encoding of the class labels.**

Using one-hot encoding, the encoding of the class labels is given by y ∈ C = {(1, 0, 0, 0, 0),

(1, 0, 0, 0, 0), (1, 0, 0, 0, 0), (1, 0, 0, 0, 0), (1, 0, 0, 0, 0)} where Li corresponds to class i.

**2.3. Name the loss function.**

The loss function is cross-entropy loss.

**2.4. Draw the computational graph of the forward propagation path.**

The computational graph is given by:

A screenshot of a computer

Description automatically generated with low confidence

**Question 3. Regularisation**

**3.1. Explain why regularization methods are required in training neural networks.**

They are required as, often, neural networks are prone to *overfitting*­–the phenomenon of fitting our training data more closely than fitting the underlying distribution. When we train neural networks, due to resource constraints we must learn from only a fraction of data examples. However, when working with limited data sets, we are in danger of discovering apparent associations that eventually turn out to not hold up when we collect more data.

**3.2. Name two regularisation methods and briefly explain how to use them in training of neural networks.**

One regularisation method is *weight decay* (commonly called *L*2 regularisation). With this method, we compute the norm of the weight vector and add it as a penalty term in the loss function. Therefore, rather than minimising the prediction loss on the training labels, we opt to minimise the sum of the prediction loss and the penalty term. Now, if the norm of the weight vector is too large, our learning algorithm might focus on minimising the weight norm rather than the training error, reducing the likelihood of overfitting.

Another such method is *dropout*. With this method, we inject noise while computing each hidden layer during training by literally dropping out some neurons. Throughout training, on each iteration, standard dropout consists of zeroing out some fraction of the nodes in each layer before calculating the subsequent layer. Moreover, we ensure to dropout neurons in an unbiased manner so that the expected value of each layer–while fixing the others–equals the value it would have taken without dropout.

**Question 4**. **Hyper-parameters**

**With an example to explain what hyper-parameters are in training of neural networks and how they should be selected.**

*Hyperparameters* are parameters which are tuneable but tuned outside of the training loop. These parameters should be selected through the process of *hyperparameter tuning*, which typically requires that we adjust them based on the results of the training of the training loop as assessed on a separate validation dataset. Specifically, we tune these hyperparameters to maximise testing accuracy and minimise training loss. Examples of hyperparameters include the number of epochs, learning rate, batch size, regularisation rate and dropout rate.

**Question 5 – Convolutional Neural networks**

**5.1. Suppose 3x3 convolution kernels are used in a convolutional neural network with three convolutional layers, what is the receptive field of the element in the last convolutional layer? How many layers do we need if a receptive field of 21x21 is required?**

The receptive field *RF* is given by:

where:

* *kh* and *kw* denote the height and width dimensions, respectively, of kernel *k*
* *n* denotes the number of convolutional layers

Therefore, the receptive field *RF* of the elements in the last convolutional layer is 7 × 7.

The number of layers *n* needed is given by:

where:

* *kh* and *kw* denote the height and width dimensions, respectively, of kernel *k*
* *n* denotes the number of convolutional layers

Therefore, the number of layers *n* needed if a receptive field of 21 × 21 is required is 10.

**5.2. Describe how the neurons of convolutional neural networks and multilayer network are designed differently and why convolutional neural networks are more suitable for image classification tasks.**

In multi-layer networks, at each subsequent layer, neurons are designed to receive, as inputs, the output of every neuron in the previous layer where each input has its own unique weight. In convolutional neural networks, however, neurons are designed to receive, as inputs, only a window of contiguous outputs from the previous layer where weights are shared among the same inputs.

Unlike multilayer networks, which are invariant to the order of the features, convolutional neural networks enable us to leverage the knowledge that nearby pixels are typically related to each other, which can allow us to construct more efficient models for learning from images.

**5.3. Consider the following image**

A picture containing text, crossword puzzle

Description automatically generated

**and a convolution kernel below with stride 2 and zero padding (2,2).**

Calendar

Description automatically generated

**What is the output of the convolution?**

The output O of the convolution is given by:

A picture containing text

Description automatically generated

**END OF TEST**