AI 410 Prerequisite Test *

Beaver-Edge AI Institute

If you are not sure whether you are ready to take AI 410, please complete this prerequisites test. There is no time limit on solving problems below.

If you can solve 70% of the problems, you are ready to take AI 410.

1. Create a tensor with dimension 4 and only 1 element. Set `requires_grad` as `True` and `device` as `"cuda"`.

   Extract the value from this tensor. Is it a NumPy array or not?

2. Let `X` be an image tensor with shape `(num_samples,num_channels,height,width)`. So `X[n,c,i,j]` is the pixel value on position `[i,j]` in channel `c` in sample `n`.

   Let `Y` be a new image tensor with shape `(num_channels,height,width)`.

   Please insert image `Y` to the end of `X`. After this operation, the tensor shall be with shape `(num_samples+1,num_channels,height,width)`.

3. Consider the following module structure and forward propagation. Let the input be `x0` with shape `num_samples,20`.

   (a) `y1 = f0(x0)`, where `f0` consists of the following layers that shall be put as a sequence:
      i. Linear layer with 20 input features and 128 output features.
      ii. 0-dim batch normalization layer.
      iii. `ReLU` layer.
      iv. Linear layer with 128 input features and 128 output features.
      v. 0-dim batch normalization layer.

   (b) Residual/skip connection: `y2 = x0 + y1`

   (c) `ReLU` layer. Denote the output as `x1`.

   We pack the above layers into a list.

   After getting the output `x1`, we do the following operations:

   (a) Concatenate with another input `z` with shape `num_samples,256`.

   (b) A dropout layer with dropout probability 0.3.

   (c) A linear layer with 10 output features.

   (d) The dropout and linear layers shall be packed into a dictionary.

Define this module.

4. Build a next sentence prediction (NSP) dataset.

   Create a synthetic raw dataset.

   (a) Each sample consists of two sentences that follow a logical relation.

   (b) Each sample's data format: A list with two items. Each item is a 1-dim tensor with each token represented by its ID.

   (c) The end of each sentence is a special token with ID 2.

   (d) While generating tokens, avoid using special token IDs 0, 1, 2.

   Build a dataset object.

5. In this problem, we use the augmented next sentence prediction (NSP) dataset that you construct in the assignment for the dataset chapter.

   (a) Create your collation function.

   (b) Build a DataLoader object.

   (c) Iterate over your DataLoader object.

6. This is a coding task.

   Consider a training dataset with a classification task that has `K` distinct labels in total.

   (a) `y`: The shape is `(num_samples,)`. `y[n]` is the ground-truth label of sample `n`.

   (b) `y_pred_logit`: The shape is `(num_samples,num_labels)`. `y_pred_logit[n,k]` is the predicted logit for label `k` in sample `n`. The predicted probability of each label is obtained from softmaxing those predicted logits.

   Write code to compute the cross entropy.

   Loop is not allowed.

7. We know that digit 8 is too similar to 0, 3, 6. So we particularly care about the following things:

   (a) If the ground-truth digit is 8, we want it to be correctly predicted.

   (b) If the ground-truth digit is 0, 3, or 6, we want to particularly avoid the prediction to be 8.

   Do the following tasks:

   (a) Modify your cross-entropy loss function.
   Explain the intuition behind your model.

   (b) Retrain and retest your model with the new loss function.

   (c) In test dataset, compute the probability that you corrected predict digit 8.

(d) In test dataset, compute the probability that you incorrectly predict digit 8 when the ground-truth is 0, 3, or 6.

8. Consider the following binary classification model:

$$\hat{y} = \begin{cases} 1 & \text{with probability } \sigma\left(\theta^\top \bar{\mathbf{x}}\right) \\ 0 & \text{with probability } 1 - \sigma\left(\theta^\top \bar{\mathbf{x}}\right) \end{cases} .$$

The ground-truth output values are 1 and 0.

Define a class called `MyBinaryClassLogisticRegression`.

(a) The attribute is θ.

(b) Methods include `fit`, `predict`, `score`. All these mimic those Sklearn models, except `score` has an input that allows you to choose between outputting an average score or an f1 score.

9. Do linear regression on the Medical Cost Personal Dataset to predict insurance charges.

Source: `https://www.kaggle.com/datasets/mirichoi0218/insurance/data`

Hints:

(a) Check whether there is any missing or duplicate data.

(b) Input features have both numeric and categorical data. One-hot encoding is needed for categorical data.