National University of Singapore
College of Design & Engineering – ECE

EE4400 Data Engineering and Deep Learning
Tutorial 4 - CNN

Q1. What is the purpose of having the convolution layer in a CNN?

Q2. Why does the convolution filter slide across the input image?

Q3. (a) A single channel input image with size 9x9 pixels is convolved with a 5x5 filter, applied with stride 2. What is the size of the output activation map?
(b) A slightly bigger output activation map is required. What simple modification can be made in order to achieve this?
(c) Determine the number of weights in each case.

Q4. See next page.

© CK Tham ECE NUS

Q4. This question is on handwritten digit image classification using a convolutional neural network (CNN).

We shall use the Tensorflow package which contains the Keras sub-package to implement the CNN. The dataset is the MNIST dataset found within Keras. Import the necessary packages.

The following functions are provided in the skeleton notebook file:

- function `load_dataset()` to load the training and test datasets, reshape the inputs to be in a single channel, and convert the outputs to one-hot-encoded format.
- function `prep_pixels()` to prepare (convert integers to floats) and scale pixels (so that range is [0,1]) in the input images.
- function `summarize_performance(scores)` to summarize the model performance using a box plot of the accuracy values, using `scores` as input.

(a) Write a function `define_model()` to define the CNN model. Set up a CNN network with 16 filters each of size (3,3) followed by a 2D max pooling layer of pooling size (2,2). Add a dense layer of 50 nodes before the output with 10 nodes (i.e., 1 node per digit). Use the SGD optimizer and the categorical cross entropy loss function.

(b) Print out the model summary which shows the number of parameters at each layer. Explain the numbers of parameters that you observe.

(c) Write a function `evaluate_model(dataX, dataY, n_folds=5)` to evaluate a model through *n*-fold cross-validation using the sklearn.model_selection function `KFold()`. The `define_model()`, `model.fit()` and `model.evaluate()` functions are used. The `evaluate_model()` function returns the `scores` and `histories`, which are lists of accuracy values and model statistics, respectively.

(d) Write a function `summarize_diagnostics(histories)` to plot the learning curves of cross entropy loss and classification accuracy, using `histories` as input.

The main processing steps that calls the functions defined above to do the following steps: (i) load the dataset, (ii) prepare pixel data, (iii) evaluate model, (iv) plot learning curves, and (v) summarize estimated performance, are shown in the skeleton file.

Note: the `evaluate_model()` function may take a few minutes (even with GPU acceleration; longer without GPU acceleration) to run since the network and data are fairly large, and a lot of processing is required.

© CK Tham ECE NUS