Classifying Handwritten Digits with TF.learn

Outlines

(i) Installation (ii) Download dataset (iii) Visualize image (iv) train → classifier (v) Evaluate (vi) visualize weights

Docker-setup: How to install TF using Docker
Starting by ensuring Docker itself is installed.

Installation

- Docker setup:
  - Install Docker, open terminal, note the Docker
  - Run Tensorflow Docker image from Docker Hub
  - Containers start Jupyter Notebook on port 8888 - access via browser.

- Pre-installed Packages include:
  - Tensorflow with tf.contrib.learn
  - matplotlib for plotting
  - TF.learn high-level Estimators.

Download dataset

- Load MNIST via → one-linear
  mnist = learn.datasets.load_dataset('mnist')
  - Comes with 55k training, 10k test 28×28 grayscale images.
  - Accs via: .mnist.train.images, .labels
    .mnist.test.images, .labels
  - Can you subset for faster iteration

Visualize images

- Plot Sample digits with a helper function.
  image.reshape([28,28])
- images are 784-length 1D-arrays - reshape to 2D for plot
- Displays a variety of clear to ambiguous handwriting samples.

Train Linear classifier

- problem: multi class classification (digits 0-9)
- features; raw pixel values, 784 per image.
- flatten image → Linear Classifier input.

feature_columns = [real_valued_column("", dimension
Classifier = LinearClassifier(n_classes=10, feature
_columns = feature_columns)

- Conceptually:
  - 784 input nodes, 10 output nodes, fully connected via weights.
  - each pixel contributes evidence (weight x pixel value) to each class.

- Train using
  - 784 inputs nodes, 10 output nodes, fully connected via weights.
  - Each pixel contributes evidence (weight x

Classifier.fit(data, labels, batch_size=100
steps=1000)

Evaluate the models

— Evaluate Performance:
classifier.evaluate(test_data, test_labels) [accuracy]
— Approximate Accuracy : ~ 90%.
— Use Predict() to see Correct and incorrect example Predictions.

Visualize Learned Weights

- Access learned Weights for each class [784 dims]
— Reshape to 28 x 28 and Plot With a red/blue (seismic) Color map.
  - Red = Positive Weight (evidence for class)
  - Blue = -ve ——— (evidence against)

784]]

— Interpretation:
  - Middle Pixel Strong +ve for '1'.
  - Same Pixel Strong -ve for '0'.