Using AI-powered Analytics in Oracle DB 12c for Automatic Image Recognition

By

Lakshman Bulusu

Independent Consultant

Greater NY Metro Area

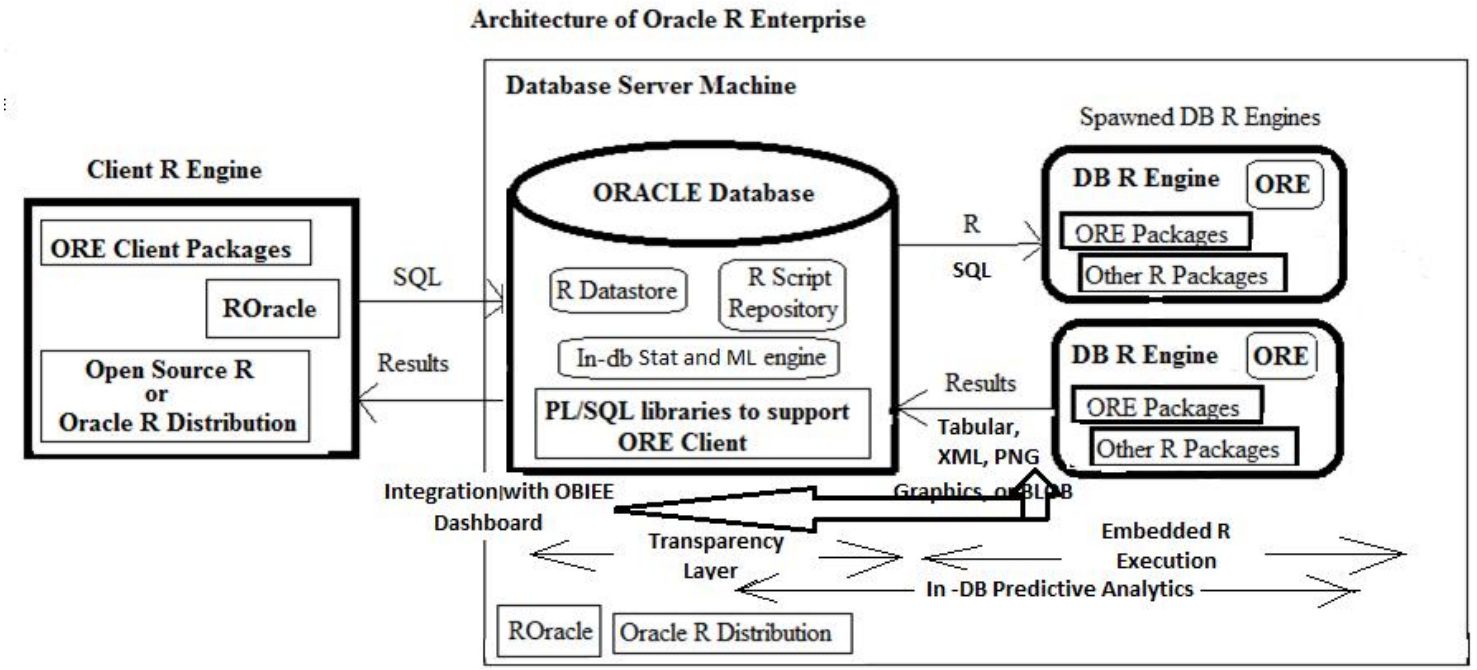
Using AI-based CNN model

- Using AI-based Machine Learning (ML) via Oracle R Technologies in oracle DB 12c
- Implementing an AI-based Convolutional Neural Network (CNN) algorithm in R and Oracle DB 12c
- Using the keras fashion mnist dataset and keras-based CNN model in R
- Output of CNN compared visually to test automatic image recognition
- Key takeaways:
  - An outline of Oracle R technologies relevant to CNN implementation
  - An outline of CNNs
  - Using the keras fashion mnist dataset and keras-based CNN model in R
  - Building and scoring a model using *CNN in R* algorithm for in-database execution in Oracle12c based on keras dataset and keras and TensorFlow.
  - End-to-End implementation steps of the above and its accuracy and loss determination

Outline of Oracle R Technologies revelant to CNN implementation – Oracle R Enterprise (ORE)

- Oracle R Enterprise (ORE) – the main component of Oracle R technologies available with the Oracle Advanced Analytics option
- Architecture
  - Database Server Machine with Oracle DB installed – contains libraries and PL/SQL programs for ORE client
  - R Engine installed on Oracle DB for embedded R execution - in-database execution of statistics and machine learning functions and models. Each DB R Engine has ORE Server and ORE Client packages.
  - Multiple R Engines spawned for data parallelism.
  - Native Oracle DB features for SQL and PL/SQL
  - Oracle R Distribution
  - ROracle for database connectivity
  - Client R Engine with client ORE packages, open source R (or Oracle R Distribution) and ROracle
  - In-DB R Script Repository that stores R scripts – callable by name directly from SQL
  - In-DB R datastore that stores R objects

Oracle R Enterprise (ORE) contd….



ORE contd....

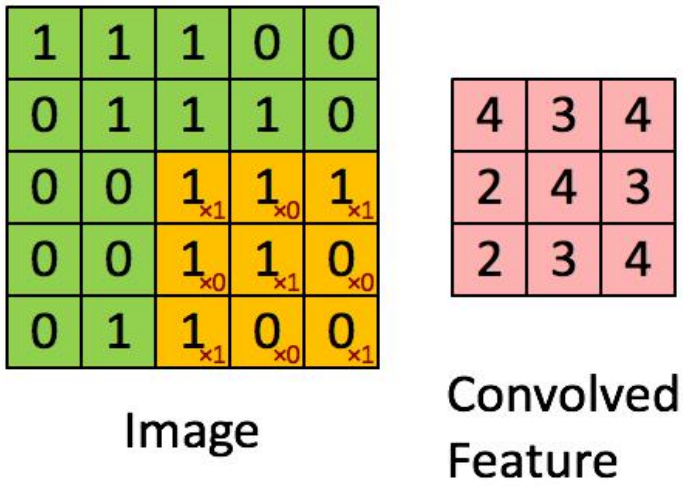
- Advantages –
  - In-database execution on stored data directly
  - Minimal latency, high-performance, multi-threaded and parallel capabilities for data and models,
  - Scalability
  - Minimal memory usage
- ORE extends open source R in the following manner:
  - ORE Transparency Layer
  - Embedded R Execution (via the Embedded R Engine) - both R Interface and SQL Interface
  - Predictive Analytics
  - As of this writing, the latest version of Oracle R Enterprise is 1.5.0.
- Further details and additional enhancements to ORE can be found at *http://www.oracle.com7technetwork/database/database-technologies/r/r-enterprise/overview/index.html*.

An outline of Convolutional Neural Networks (CNNs)

Source: https://yashk2810.github.io/Applying-Convolutional-Neural-Network-on-the-MNIST-dataset/

What is a Convolutional Neural Network?

A convolution in CNN is nothing but a element wise multiplication i.e. dot product of the image matrix and the filter.



In the above example, the image is a 5 x 5 matrix and the filter going over it is a 3 x 3 matrix. A convolution operation takes place between the image and the filter and the convolved feature is generated. Each filter in a CNN, learns different characteristic of an image.

Using the keras fashion mnist dataset and keras-based CNN model in R

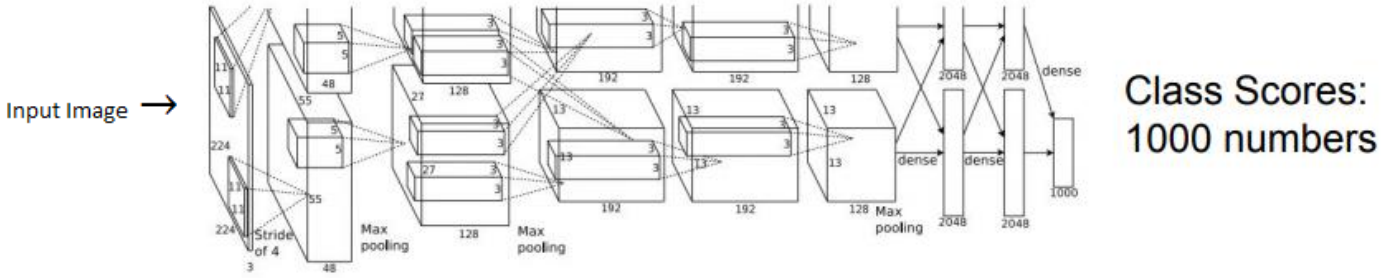
- Keras is a high-level API originally written in Python and now available in R too. Both the APIs use TensorFlow as back-end platform to run on.
- Fashion MNIST is a dataset of 60000 images in grayscale each 28X28 pixels size along with a test set of 10000 images. The class labels are encoded as 10 integers ranging from 0 to 9 representing T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, and Ankle Foot.
- The dataset structure is such that it returns lists of training and test data: the x-part is an array of images with dimension (60000,28,28) and y-part is an array of corresponding labels in the range 0-9 with dimension (10000).
- The CNN model first trains with training data and then classifies the images in test data. This way it recognizes the images in test data by applying the AI-based model to it. The output of the classification (thereby recognition) is correctly determining the image label for any given image and checking the raster display of the same image in the test set.

Using the keras fashion mnist dataset and keras-based CNN model in R

- The first step in proceeding further is preparing the image data so that it is in acceptable format of the CNN model. This consists of:
  - Reshaping the x-array into dimension of rank 4, i.e, (num_samples, width in pixels, height in pixels, channels). This we reshape the -part as follows:
  - train_images <- array_reshape(train x-part, c(60000,28,28,1))
  - test_images <- array_reshape(test x-part, c(10000,28,28,1))
- The second step is to normalize the x-data so that each image pixel lies in the interval [0,1] as opposed to original [0,255]. This is done by diving the train_images and test_images by 255
- The third step is to one-hot encode the train and test labels (the y-part). One-hot encoding means converting an integer or label value is transformed into an array that has only one ‘1’ value and the rest ‘0’ values.
  - train_labels <- to_categorical(train y-part)
  - test_labels <- to_categorical(test y-part)

Visualizing an example CNN

(source: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture12.pdf)



Input image →ConvLayer →ActivationLayer(Relu) → MaxPooling →DropoutLayer → ConvLayer →ActivationLayer(Relu)→ MaxPooling →DropoutLayer → Flatten → DenseLayer → ActivationLayer(Relu) → DropoutLayer → DenseLayer → Softmax (activation)→output layer

- Input image →ConvLayer →ActivationLayer(Relu) → MaxPooling →DropoutLayer → ConvLayer →ActivationLayer(Relu)→ MaxPooling →DropoutLayer → Flatten → DenseLayer → ActivationLayer(Relu) → DropoutLayer → DenseLayer → Softmax (activation)→output layer

Building and scoring a model using *CNN in R* algorithm for in-database execution in Oracle12c based on keras dataset and keras and TensorFlow

The following are the steps to builds a CNN model (code shown in Demo):

1. Build CNN architecture (Adding linear stack of layers)
    1. Add hidden layers
        1. Add Convolution(-al layer) specifying number of filters and size of each filter

        and the shape (dimension) of the input in (width, height, channels) format. This runs the filter over the whole image thus helping in recognition more precisely.
        1. Add Activation (layer) (This layer is to reduce training time and uses the most famous “relu” activation that all negative values in the matrix to 0 and keeps all other values the same)
        2. Add pooling (layer) (to reduce the dimensionality of the features map and make the model less complex to compute)
        3. Add dropout (layer) (to avoid over-fitting)
    2. Add layer to flatten input (this is needed to input the layers so far added to the dense layers)
    3. Add dense layers (using this the CNN classifies the inputs)
    4. Add output layer (using softmax activation to calculate categorical cross-entropy – since we have 10 classes which correspond to 10 images)

Building and scoring a model using *CNN in R* algorithm for in-database execution in Oracle12c based on keras dataset and keras and TensorFlow

1. Compile the CNN
2. Train the CNN by way of fitting it using the features (train x-part), targets (train y-part), number of epochs, batch size and validation split.
3. Plot the fitted output for loss and accuracy curves. (Shown in Figure 1)
4. Evaluate the CNN on test data set (test x-part, test y-part)
5. Predict the classes using the model and the test images (test x-part)
6. Test the validity of the model using a table of predicted and actual values (output of the prediction above and the test y-part).
7. It outputs a table of 10 rows and 10 columns.
8. Check the values of the labels (Predicted vs. actual) for any given label, i.e., pred_test[1] and test y-part [1]. Also plot the raster image of original test x-part [1, ,] and see that it's label matches with the predicted label number (class). (Raster image shown in Figure 2)

Summary of built CNN model

- > str(model)

Model

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 28, 28, 32)	320
activation_1 (Activation)	(None, 28, 28, 32)	0
conv2d_2 (Conv2D)	(None, 26, 26, 32)	9248
activation_2 (Activation)	(None, 26, 26, 32)	0
max_pooling2d_1 (MaxPooling2D)	(None, 13, 13, 32)	0
dropout_1 (Dropout)	(None, 13, 13, 32)	0
conv2d_3 (Conv2D)	(None, 13, 13, 32)	9248
activation_3 (Activation)	(None, 13, 13, 32)	0

Summary of built CNN model

conv2d_4 (Conv2D) (None, 11, 11, 32) 9248

---

activation_4 (Activation) (None, 11, 11, 32) 0

---

max_pooling2d_2 (MaxPooling2D) (None, 5, 5, 32) 0

---

dropout_2 (Dropout) (None, 5, 5, 32) 0

---

flatten_1 (Flatten) (None, 800) 0

---

dense_1 (Dense) (None, 512) 410112

---

activation_5 (Activation) (None, 512) 0

---

dropout_3 (Dropout) (None, 512) 0

---

dense_2 (Dense) (None, 10) 5130

---

activation_6 (Activation) (None, 10) 0

Summary of built CNN model

==========

Total params: 443,306

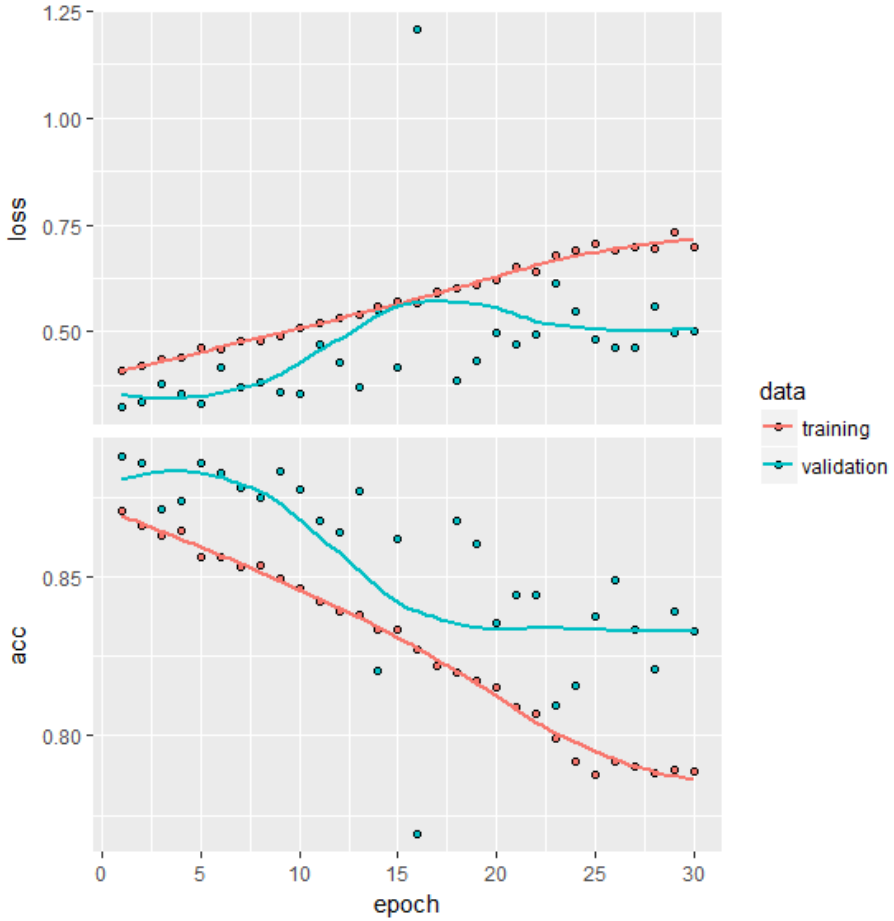
Trainable params: 443,306

Non-trainable params: 0

_____

Plot of fitted CNN using keras and TensorFlow

Figures 1 and 2 showing loss and accuracy curves of the model on training images and test images; and the raster plot of recognized image from CNN - belongs to class 9 – Ankle Foot



Inetgrating the CNN with Oracle DB 12c

- 1. Train the CNN using R as stated in earlier slides.
- 2. Transferred the trained model into Oracle DB 121c for scoring it. This gives the ability to apply CNNs (and NNs in particular) to relational databases.
- We need two tables TENSOR_ARRAY and TESTDATA_ARRAY. The former contains numerical values corresponding to the tensors from the CNN. The tensors are defined using the data type UTL_NLA_ARRAY_FLT
- The TESTDATA_ARRAY contains the test images encoded using the UTL_NLA_ARRAY_FLT.
- Next using a Oracle built-in PL/SQL package UTL_NLA (Linear Algebra).
- Finally we write a custom package called FASHION_MNIST with the INIT procedure and function. INIT must load the structure of the CNN from the table TENSORS_ARRAY into PL/SQL variables and the SCORE function that takes an image as input and returns a number which is the predicted value of the image class (the label value).
- Next we call these two sub-programs in order as follows:

SQL> exec fashion_mnist

SQL> select fashion_mnist.score(test_images), label from testdata_array where rownum <2;

This returns the number 9 as both the column values of the select above indicating the Ankle Foot image.

Inetgrating the CNN with Oracle DB 12c

- Examples of the table definitions of testdata_array and tensors_array is shown below:
  - The initial tensor definition and test data is defined as per the following tables:

```
create table tensors (name varchar2(20), val_id number, val binary_float, primary key(name, val_id));
create table testdata (image_id number, label number, val_id number, val binary_float, primary key(image_id, val_id));
```

  - Next we define the tensors and test images in VARRAYS of type UTL_NLA_ARRAY_FLT

```
create table testdata_array as
select a.image_id, a.label,
cast(multiset(select val from testdata where image_id=a.image_id order by val_id) as utl_nla_array_flt) image_array
from (select distinct image_id, label from testdata) a order by image_id;

create table tensors_array as
select a.name, cast(multiset(select val from tensors where name=a.name order by val_id) as utl_nla_array_flt) tensor_vals
from (select distinct name from tensors) a;
```

A similar example is available in the Oracle Docs as well as in the blog:
https://db-blog.web.cern.ch/blog/luca-canali/2016-07-neural-network-scoring-engine-plsql-recognizing-handwritten-digits

Demo

Demo

Thanks

Thanks for coming

Further Reading

- Foster Provost & Tom Fawcett, Data Science for Business, O'Reilly Media, Inc., 2013
- Oracle R Enterprise Documentation Media Library, Release 1.5, R Enterprise Users Guide, https://docs.oracle.com/cd/E67822_01/OREUG/toc.htm
- http://www.oracle.com/technetwork/database/database-technologies/r/r-technologies/documentation/documentation-2166653.html
- http://analyticsvidhya.com/
- https://datascienceplus.com/
- https://blogs.oracle.com/r/

Q and A

Q and A