AI-ENABLED ANALYSIS-READY DATACUBES: TOWARDS A ROADMAP FOR MORE HUMAN-CENTRIC SERVICES

Peter Baumann, Otoniel Campos, Dimitar Misev

Jacobs University | rasdaman GmbH, Germany, p.baumann@jacobs-university.de

ABSTRACT

We present an integration of AI – specifically: Neural Networks – into a datacube query language. The concept is implemented in the query language of an Array Database System, rasdaman. The expected advantages are a combination of the ML intelligence with datacube scalability, based on the ARD-oriented, more human-centric paradigm of spatio-temporal datacubes as compared to sets of scenes and products. The principles are outlined, in particular the User Defined Function (UDF) approach used, as well as application to spatio-temporal remote sensing datacubes.

Index Terms— Datacubes, Machine Learning, ML, AI-Cube

1. INTRODUCTION

Earth-observing sensors, such as satellites and drones, play a critical role in today's intelligence information mix. However, the huge volume of data obtained is not commensurate with evaluation capabilities; in fact, manifold opportunities for operative insight are missed due to the Big Data problem: too big, too fast, too diverse to analyse.

This prompts a demand for Analysis-Ready Data (ARD). A cornerstone of ARD is the datacube paradigm where the zillions of single images, such as from the Landsat and Sentinel satellite families, are homogenized and lined up in space and time so as to appear as one single object per sensor. As experience shows, this dramatically eases and potentially speeds up data assessment even by non-experts.

Analysis today often means use of Machine Learning (ML) through Neural Networks, Transformers, and other AI methods. Therefore, it is an interesting question how datacube services have to be shaped so as to become analysis-ready for AI. In our research we investigate this question following a database approach where a declarative datacube query language gets enhanced with AI functionality. Datacubes served through AI-enabled services we call *AI-Cubes*. In this contribution we present status and results of our research which is based on the Array Database System rasdaman.

2. DATACUBES

Datacubes are a natural concept for spatio-temporal Earth data where data points sit at the points of some (regular or irregular) grid, with coordinates given by space, time, or even abstract axes (like spectrum band). Typically, such data today are served on disk as a usually huge number of files with cumbersome conventions for encoding data and metadata (such as locations). The effect is that only experts simultaneously versed in programming, data management, and Earth data wrangling can work on these Big Data, and even for them it requires substantial time to achieve the desired results. Datacubes, conversely, offer intuitive functionality like extraction of time slices, aggregation in space and time, and combination (fusion) of different datacubes, even with divergent dimension, size, and resolution.

3. ML ON DATACUBES

Our starting point is the observation that both ML and array query languages like rasql, SQL/MDA, and WCPS share the same mathematical basis, Tensor Algebra. Also the recent support of processor architectures for tensor operations boosts ML, and array query processing can be expected to benefit likewise.

All this suggests adding ML support to array languages. One approach consists of using the code injection capabilities of Array DBMSs, such as rasdaman, in the database community called User-Defined Functions (UDFs). With this technique, external code can be invoked from within database queries, resulting in dynamically linking that code into the database server during query execution. For users, such code is offered as a function that appears like the query syntax has been extended.

This name is slightly misleading – not database users provide code, but the database administrator

We use the syntax ISO SQL array extension, MDA (Multi-Dimensional Arrays), and rasdaman to illustrate UDFs through an example. It is based on a table like

```
CREATE TABLE A(
id: integer,
name: string,
image integer mdarray(0:100, -1000:+1000, 50:100)
)
```

Where the image attribute is a 3-D *mdarray* (multi-dimensional array) over integers with respective dimension extents of 0 to 100, -1000 to +1000, and 50 to 100.

The first example takes, for every tuple in the A table, the *fib*() function from the *math* namespace to compute, for every pixel in the image attribute, the Fibonacci number. From all those values, the maximum is determined and returned to the caller, resulting in a column of one maximum value per image:

```
SELECT max_cells( math.fib( A.image ) )
FROM A
```

The second example takes the *avg*() function from the *stat* package and applies it to the pixel-wise logarithm of a sub-array:

```
SELECT stat.avg( log( A.image [0,100:200,*:*] ) )
FROM A
```

As the examples show, the UDF can be invoked anywhere in a query expression, and is fully subject to the standard query orchestration, including parallelization.

Similarly, in our work using rasdaman a pre-trained forest-fire detection model can be invoked via

```
SELECT ml.eval( A.image, M.model ) )
FROM A, M
WHERE M.name='forest fire'
```

In a similar fashion, model training can be incorporated via some UDF like *ml.train*().

UDFs can be registered by the administrator through statements of the kind

```
CREATE FUNCTION stat.avg( array a )
RETURNS double
LANGUAGE cpp
EXTERN "stat/average.so"
```

Via the native C++ interface code in C, C++, and most other languages can be coupled. In our work on ML we link in the torch ML library.

6. CONCLUSIONS

With the UDF approach ML methods can be added to array database queries in a straightforward, seamless manner, easy to handle for users and administrators likewise. The added value consists of the combination of ML intelligence with datacube scalability, plus the flexibility of the resulting "any query, any model, any time" paradigm.

Currently, evaluation in several remote sensing use cases is ongoing. Next steps include further investigation of the new opportunities offered as well as benchmarking to determine the scalability curve.

7. ACKNOWLEDGEMENT

This research is supported by EU CENTURION and German AI-Cube.