Advances in multimodal image data management in OMERO Plus:

connecting tissues, cells, sequences, and features

Erin E Diel, David R Stirling, Emil Rozbicki, Chris Allan
Glencoe Software Inc., Seattle, WA, USA

Background

- **OMERO Plus** is an image data management platform designed for storage and analysis of bioimaging data and metadata.
- Backed by the **Bio-Formats** library, OMERO Plus can natively store and retrieve image formats arising from a wide variety of microscopy hardware, from high content screening (HCS) systems to spatial transcriptomics platforms.
- Metadata can range from simple annotation tags to rich tables describing per-object (cells, nuclei, spots, and other structures) features.

Together, this allows for large and complex image datasets and associated sequencing, segmentation, or other analytical results to be both stored and retrieved remotely.

The utility of this system is enhanced by closer integration with existing data science tools for processing stored data and optimized performance of remote data retrieval.

- To provide a more convenient method for working with tabular data in OMERO Plus (**OMERO.tables**) from Python environments, Glencoe Software released the open source **omero2pandas** package. This provides access to the full suite of Python data science and machine learning packages when working with OMERO data.

omero2pandas is another step towards making OMERO Plus the data engine of choice for data analytics and AI in bioimaging.

Aims

We evaluated the performance and user experience of the above data retrieval operations for three use cases: cell segmentation of highly multiplexed fluorescence whole slide images, cell segmentation of Cell Painting HCS datasets, and spatial transcriptomics datasets.

- Omero2pandas and OMERO.tables enable remote retrieval of tabular data, including support for querying for selective data retrieval.
- New technologies for the backend data storage for OMERO.tables were evaluated for improvements in this query performance. TileDB was selected for comparison with the existing backend of PyTables.

Finally, we demonstrate a practical example for dimensionality reduction on a table detailing millions of objects with hundreds of features, including tools for the visualization and mining of these results.

omero2pandas

Omero2pandas is an open-source Python library designed to streamline data retrieval and storage by converting OMERO.tables to Pandas DataFrames and vice versa. This will assist in analyzing data stored on OMERO with the scientific Python stack.

Key features:

- Load OMERO.tables to DataFrame remotely

```
df = omero2pandas.read_table(file_id=402)
```

- Download the table for local use

```
omero2pandas.download_table(
    "/path/to/output.csv", file_id=2, chunk_size=1000)
```

- Upload a results table to OMERO

```
ann_id = omero2pandas.upload_table(
    my_data_frame, "Table Name", 142, "Image")
```

- Retrieve a list of columns from a remote table.

```
columns = omero2pandas.get_table_columns(
    annotation_id=142)
```

- Read specific rows and/or columns

```
my_dataframe = omero2pandas.read_table(
    file_id=10, column_names=['object', 'intensity'],
    rows=range(0, 100, 10))
```

- Support for custom queries **NEW!**

```
omero2pandas.read_table(file_id=10, query='(area>20)')
```

Use Case: Nuclear Segmentation

A multiplexed fluorescence image (19 biomarkers, 13 x 9 mm, 20x magnification) of tonsil tissue was segmented using Glencoe's **OMERO Segmentation Connector**. The process identified 1.2 million objects and calculated more than 300 features describing each object. The image was acquired on RareCyte's Orion Platform and stored on OMERO Plus.

Segmentation label image viewed in Glencoe's **PathViewer**. OMERO.tables data visualized in Glencoe's **Pageant**.

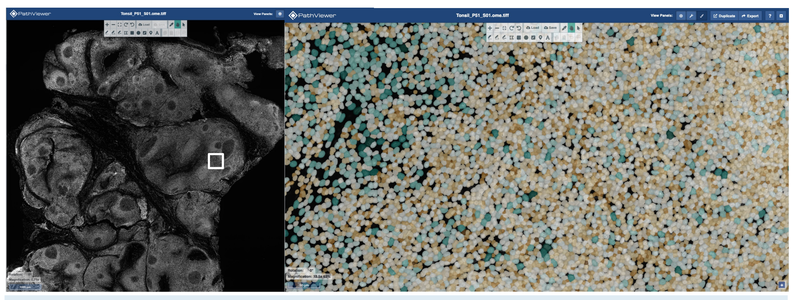

Figure 1: Left: Original image showing nuclear stain (Hoechst). Right: White rectangle region in the left panel showing segmentation results from the StarDist segmenter executed with Glencoe's OMERO Segmentation Connector. Objects are colored by their area.

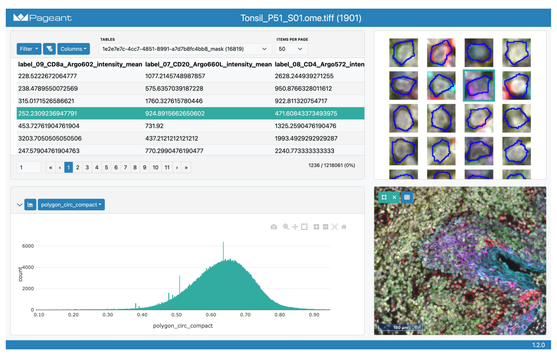

Figure 2: Table displayed in Pageant showing the features calculated by the OMERO Segmentation Connector for the objects segmented with a StarDist model. Each row represents one nucleus.

Use Case: Spatial Transcriptomics

Whole transcriptome sequencing results were populated in OMERO.tables. Reference data was obtained from public datasets from 10x Genomics' Visium platform.

Rows of the table represent individual 55 µm spots across the tissue (~14k), while columns represent genes or clustering results (hundreds to ~20k).

Spotted heat maps can be visualized overlaid with the image in Glencoe's **PathViewer**. OMERO.tables data is visualized in Glencoe's **Pageant**.

Figure 3: An H&E reference image overlaid with 55 µm spots color coded by cluster ID in PathViewer

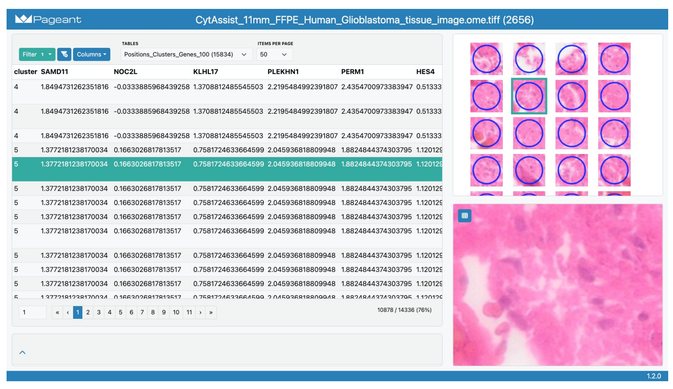

Figure 4: Table displayed in Pageant showing the features output from the Visium platform. Each row represents one spot, and each column represents a gene.

Use Case: Cell Painting

A JUMP pilot dataset (cpg0000) representing perturbation conditions and cell types was imported into OMERO Plus along with experimental metadata and used as the input to a CellProfiler pipeline. 384 rows, thousands of columns

Aggregation, normalization and feature selection were performed by PyCytominer to generate well-averaged single cell profiles, which serve as the input for perturbation profiles.

OMERO.tables data is visualized as a plate map in Glencoe's **Parade Plus**.

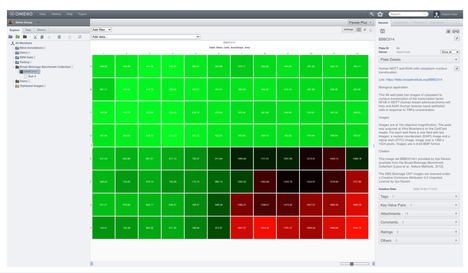

Figure 5: A plate visualized in OMERO Parade Plus with feature heatmap overlaid.

Query support in omero2pandas

The addition of support for custom queries in OMERO.tables data retrieval with omero2pandas enables significantly faster selective data access.

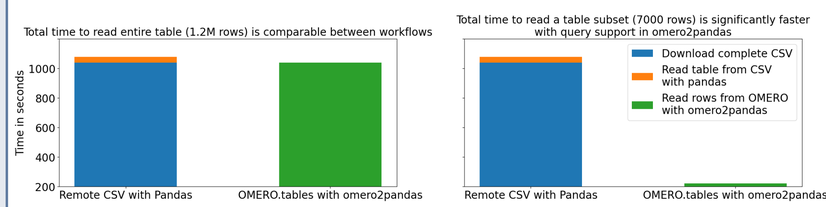

Figure 6: Examples of the timings to read the entire OMERO.table or select row(s) compared to remote CSV with pandas.read_csv().

Alternative OMERO.tables Backend

OMERO.tables is typically backed by PyTables, which uses a row-major order format. This tabular backend is therefore fundamentally limited in its support for feature-based (column) queries and results.

In contrast, TileDB supports column-major order and chunked data storage, which improves data query and read speeds.

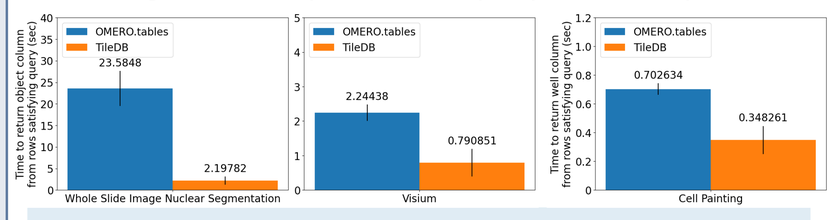

Figure 7: Column-major order in TileDB significantly improves the speed of data retrieval from remote tabular datasets when compared with OMERO.tables/PyTables. Visium data in OMERO.tables was smaller due to current column count limitations, so relative improvements are likely even greater.

Use Case: Nuclear Classification

Unsupervised clustering of segmented nuclei in multiplexed fluorescence images was performed using the features stored within OMERO.tables with **omero2pandas**.

Strongly correlated features from the **OMERO Segmentation Connector** were removed using Spearman correlation, and dimensionality reduction was performed using UMAP. An unsupervised clustering model (Birch) was built to classify the cells.

Clustering results saved back to OMERO.tables can be visualized using **PathViewer** and **Pageant** (not shown).

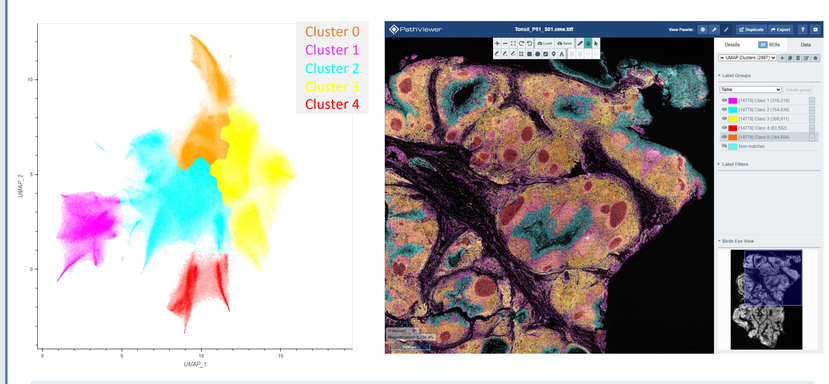

Figure 8: UMAP clustering of objects (left), identifying 5 primary clusters. (Right) original image with objects colored according to cluster identity, revealing relative localization of objects from different clusters.

Jupyter Notebook available at:

www.github.com/glencoesoftware/webinar-notebooks

Conclusions

- OMERO Plus, together with OMERO.tables, supports images and analytics from across diverse domains.
- The omero2pandas package seamlessly integrates the OMERO Plus data management platform with popular Python AI tools.
- Added support for OMERO.tables queries enables researchers to perform more advanced data filtering when requesting data with omero2pandas, easing and speeding up downstream analysis.
- Integrated applications, such as PathViewer and Pageant, enable remote visualization of complex data, supporting diverse research and clinical workflows.
- Dedicated data management for analytical results in a multiuser collaborative environment fosters improved sharing and accessibility of data among researchers.

Next steps:

- Integrate the improved backend performance of TileDB into OMERO.tables to improve scalability and efficiency, especially when handling large datasets.