Implementing Dataset Enhancements on the THREDDS Data Server

Jessica Souza*, Tara Drwenski, Hailey Johnson, Thomas Martin
*jessicacssouza@gmail.com
Unidata, University Corporation for Atmospheric Research

SUMMARY

➔ Unidata has ongoing efforts to provide new datasets and open source workflows for machine learning.
➔ This project aims to perform dataset preprocessing before user access targeting machine learning applications.
➔ As part of the preprocessing step, two common types of rescaled data are provided: standardized and normalized.
➔ Enable original + rescaled datasets of interest on THREDDS-test server.

1. Background
THREDDS Data Server (TDS)¹ is an open access web server that provides catalog, metadata and data access for real-time and archived datasets of environmental data sources at a number of distributed server sites, using a variety of remote data access protocols.

2. Motivation
There is an extensive use of machine learning (ML) models in earth sciences research.
➔ Artificial Intelligence for the Earth Systems (AIES)² is an AMS journal launched in 2022.
Data preprocessing in ML generally involves cleaning, rescaling and splitting the data.
➔ The goal of rescaling is to transform features to be on a similar range and improve the performance and training stability of the model.
Two common types of rescaling are Standardization and Normalization.
➔ In 7 AIES issues, 13 papers used Standardizer or Normalizer to preprocess their dataset, including forecast (GFS), satellite (GOES), and radar (NEXRAD) data.

Goal: Add dataset preprocessing on TDS targeting ML applications.

3. Preprocessing Data
Based on Python's Scikit-learn³ machine learning library that has an extensive list of scaling types, we implement StandardScaler and MinMaxScaler.

4. Code
External library implementations (Apache Commons Mathematics Library ⁴):
➔ Computes summary statistics for very large data streams (values are not stored in memory).

Integrating with netcdf-java codebase:
➔ Create constants / attributes in the Common Data Model class.
➔ Include Standardizer / Normalizer in the set of data enhancements.
➔ Apply enhancements to the data if “standardizer”, “normalizer” is a variable attribute and data is floating point type.

Using it in the TDS:
➔ Preprocess the THREDDS catalog.xml using the NetCDF Markup Language (NcML)⁵.
➔ NcML creates a virtual dataset without changing the original data.

5. Testing
Automated tests to evaluate if behaviour is as expected.
➔ Calculate mean, standard deviation and standardized values.
➔ Calculate minimum, maximum and normalized values.
➔ Standardizer and Normalizer act on floating points data types.
➔ Return the same input for data that contain integer data type and / or equal values.

Performance test using Apache HTTP server benchmarking tool⁶:
GFS 0.25 Degree - Forecast Model Data +80 variables
GOES 18 Product - Full Conus 1 variable

6. Results
Visualization (Jupyter notebook and Godiva³):

7. Next Steps
➔ Improvements on performance (caching for example).
➔ Provide more datasets relevant to the users.

References:
1. https://www.unidata.ucar.edu/software/tds/
3. https://scikit-learn.org/
5. https://docs.unidata.ucar.edu/netcdf-java/current/userguide/ncml_overview.html
6. https://httpd.apache.org/docs/2.4/programs/ab.html
7. https://docs.unidata.ucar.edu/netcdf-java/current/userguide/ncml_overview.html