Scalable Machine Learning in Python with Dask

10/06/2021

DoD High Performance Computing and Modernization Program (HPCMP)
User Productivity Enhancement and Training (PET)

Gedion Teklemariam, HPCMP PET

GENERAL DYNAMICS
Information Technology

Light Review of terms covered during intro course

- **Dask.array**
  - cuts up large arrays into many small ones.
- **Dask.dataframe**
  - Cuts up large dataframes into many pieces.
- **Dask.Delay**
  - Puts off bringing data into memory, until after task graph is developed.
- **Dask.bag**
  - Used to parallelize computation on unstructured or semi-structured data, like text data.
- **Trask Graphs**

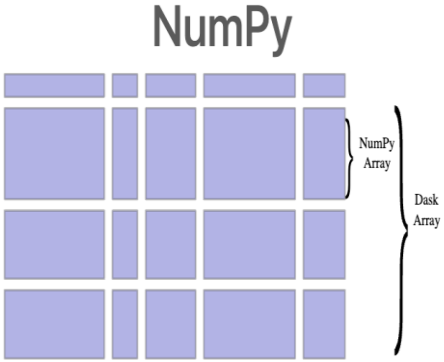
Intro Course on Dask available

USER PRODUCTIVITY ENHANCEMENT AND TRAINING

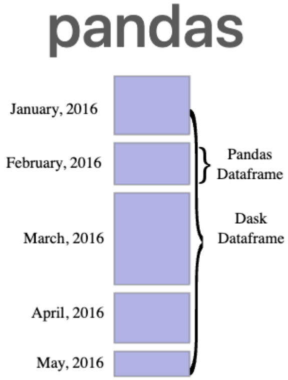
GENERAL DYNAMICS
Information Technology

The Dask Benefits

Scale Numpy Workflows



Scale Pandas workflows

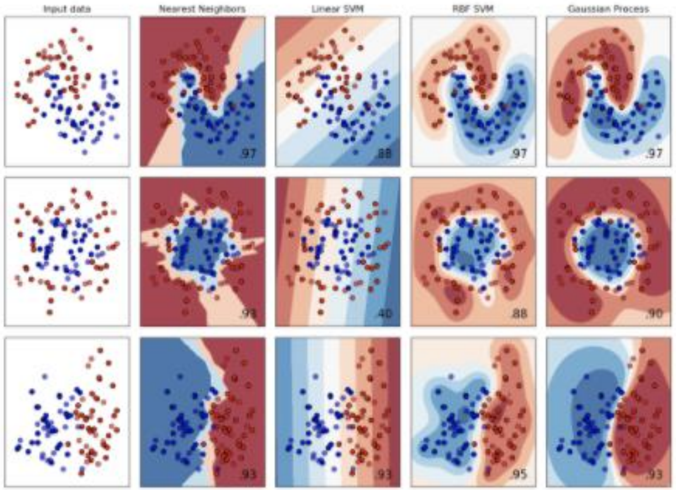


- **Parallel computing / multiprocessing :**
  - Use all cores on system (even on laptops).
- **Larger-than-memory problems:**
  - Not limited by large datasets, if using Dask.
  - work on datasets that are larger than your available memory can handle.
  - Chunk/break up your array into many small pieces, and compute on the chunks in parallel.
  - Only stream chunk data from disk when needed for computation.
- **Blocked Algorithms:**
  - Fracture large computations to many smaller computations, and execute on the smaller computations.

Intro Course on Dask available

Parallelize Scikit-learn / Joblib

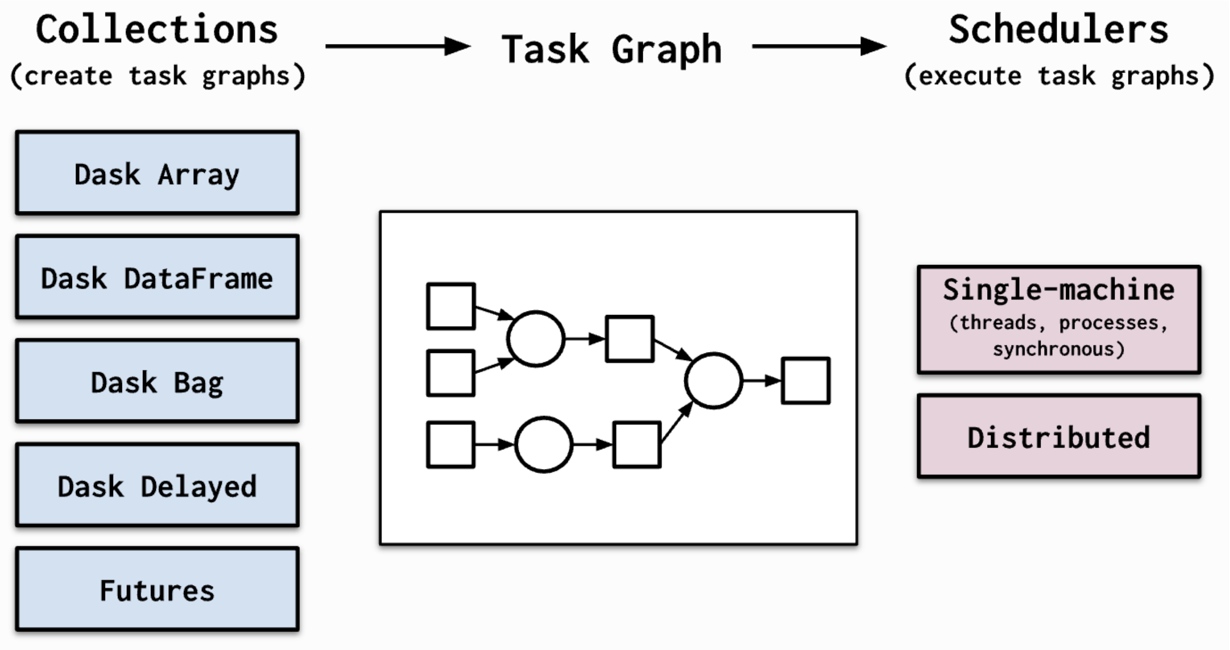
scikit-learn



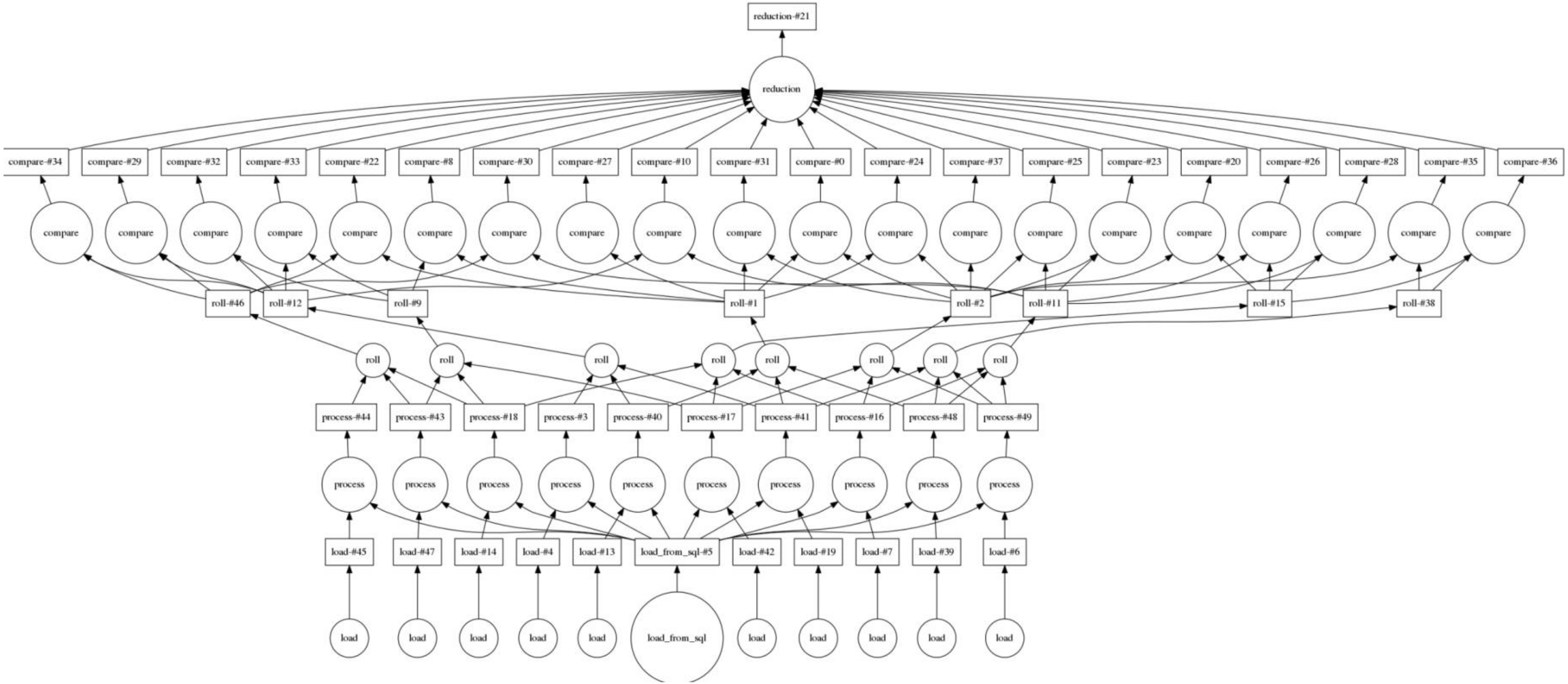
- **Scale Machine Learning APIs**
  - Scikit-learn
  - XGBoost
- **Enable scalable training and predictions on large models & large datasets.**
- **With use of wrappers for Tensorflow & Pytorch Models, integrate Dask with Neural Network models.**
  - We will be proving the concepts with light models.

Use of Task Graphs

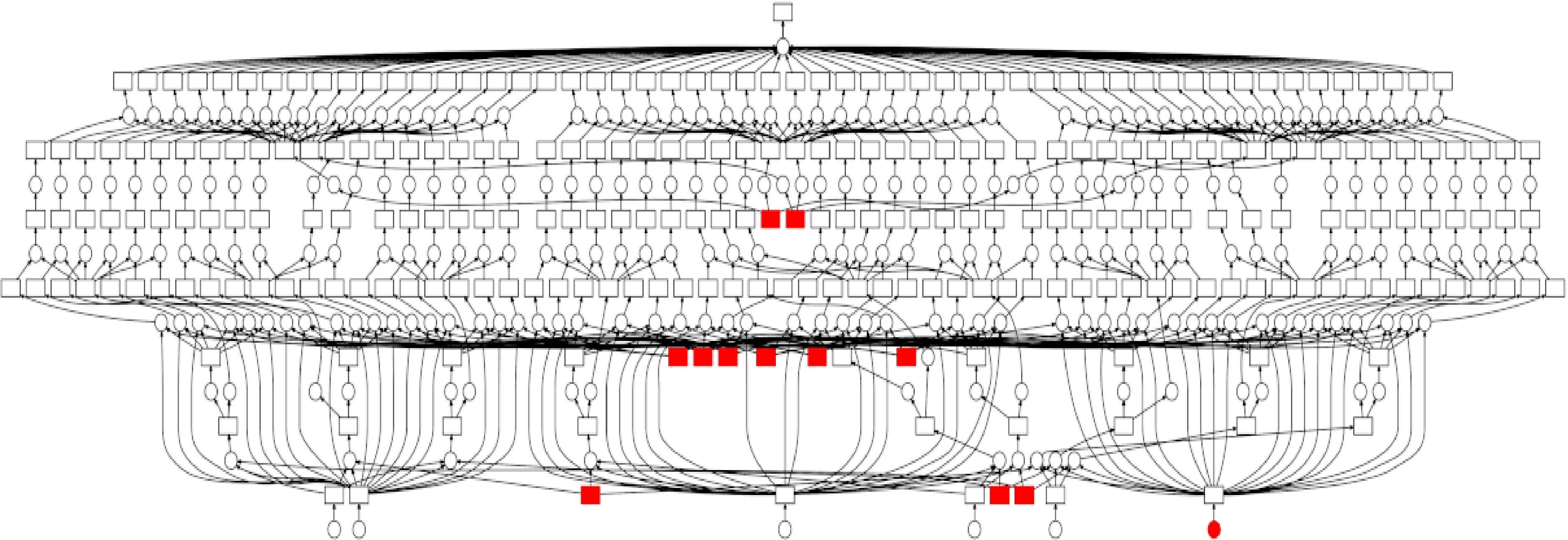
- **First Generates Task Graph.**
- **Execute them on parallel nodes.**



Dask Task Graph - Example



Parallel Computing on Chunks



Familiar to Python Users

- Designed for Python Ecosystem
- Familiar APIs for Python Users
- Scales upto 1000 node clusters

```
# Arrays implement the NumPy API
import dask.array as da
x = da.random.random(size=(10000, 10000),
                     chunks=(1000, 1000))
x + x.T - x.mean(axis=0)
```

```
# Dataframes implement the pandas API
import dask.dataframe as dd
df = dd.read_csv('s3://.../2018-*-*.csv')
df.groupby(df.account_id).balance.sum()
```

```
# Dask-ML implements the scikit-learn API
from dask_ml.linear_model \
  import LogisticRegression
lr = LogisticRegression()
lr.fit(train, test)
```

PET
USER PRODUCTIVITY
ENHANCEMENT AND TRAINING

GENERAL DYNAMICS
Information Technology

Easily Parallelize existing codebases

- **Example of using existing codes:**
  - Easily apply lazy function dask.delayed() to existing functions to improve computational speeds.
  - Computation would not occur until the dask.compute() function is executed.
- **Dask is flexible**
  - Supports current scikit-learn workflows
  - You may also develop your own new scalable algorithms.

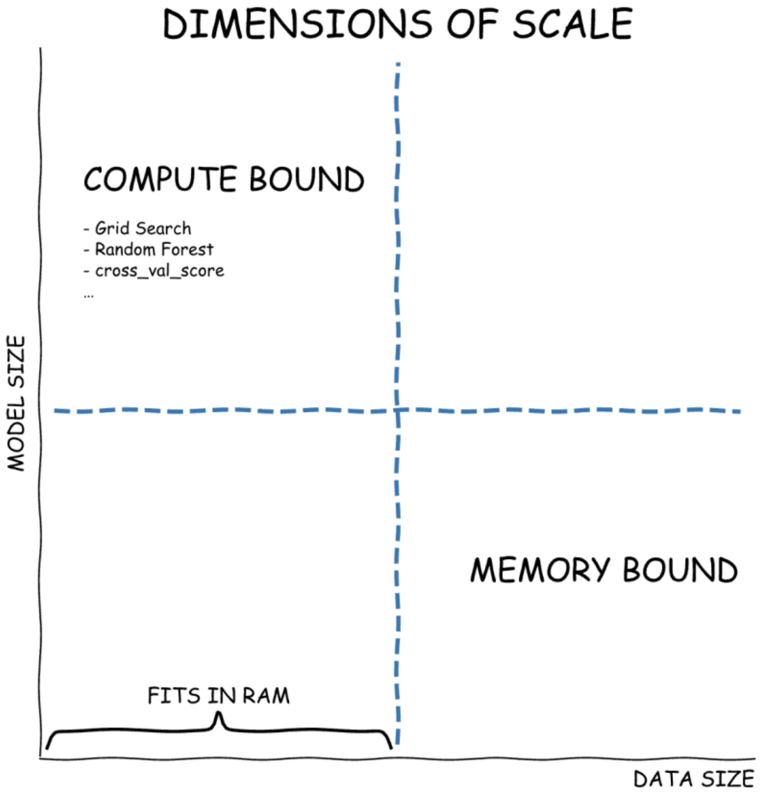
```
func1 = dask.delayed(func1)
func2 = dask.delayed(func2)

for i in X_list:
    for j in Y_list:
        if i > j:
            c = func1(i,j)
        else:
            c = func2(i,j)

        results.append(c)

final = dask.compute(results)
```

```
grid_search.fit(data.data, data.target)
```

```
with joblib.parallel_backend('dask'):
    grid_search.fit(data.data, data.target)
```

Challenges with Large ML models

- **Large Memory Problem:**
  - Dask chunks the dataset into several pieces, so that only a fraction at a time comes into memory.
- **Large Model Problem:**
  - Dask distributes and trains sub-models onto available nodes.
  - Dask distributes batches onto multiple nodes.



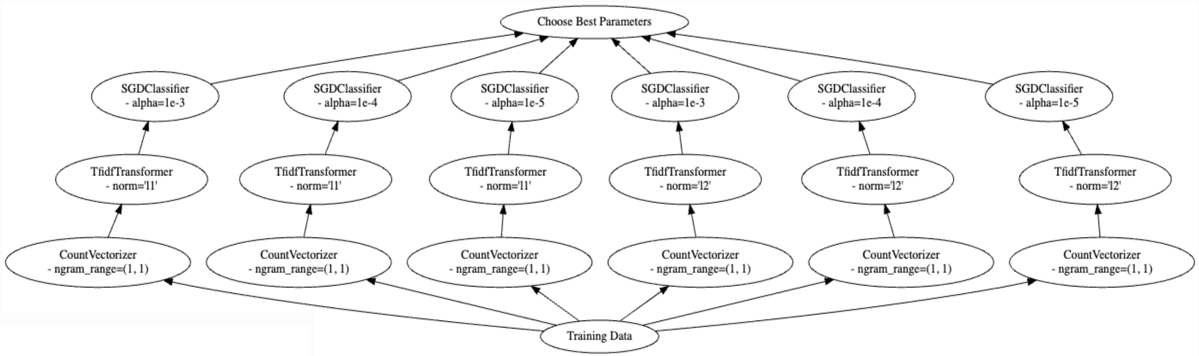
ML jobs where Dask can help

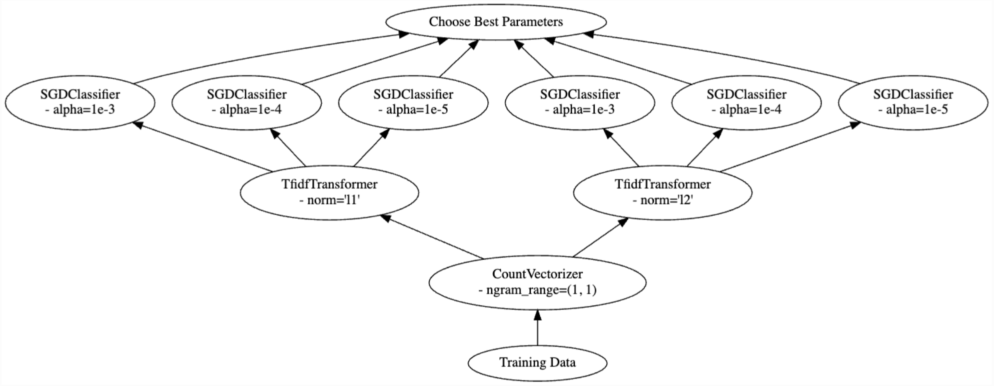
- HyperParameter Search
- Generalized Linear Models
- Parallel Meta-estimators
- Incremental Learning
- Text-Vectorization
- Automated Machine Learning (tpot)

- Use Dask for Batch Prediction with CNN models trained in PyTorch.
- Perform Hyperparameter search on tensorflow based Keras models Generalized Linear Models

Hyperparameter search – Impact of Dask

Dask (task-graph below) eliminates need to repeat redundant computational steps.





dask_ml.model_selection.IncrementalSearchCV(…)

Incrementally search for hyper-parameters on models that support partial_fit

dask_ml.model_selection.HyperbandSearchCV(…)

Find the best parameters for a particular model with an adaptive cross-validation algorithm.

dask_ml.model_selection.SuccessiveHalvingSearchCV(…)

Perform the successive halving algorithm [R424ea1a907b1-1].

dask_ml.model_selection.InverseDecaySearchCV(…)

Incrementally search for hyper-parameters on models that support partial_fit

Notes

- **During the next session, we will go through implementation examples.**
  - Demonstration will focus on utilization of Dask primarily.
  - Understanding and Developing ML models is not the primary objective of the training.
- **For additional information, links are embedded in the Jupyter Notebooks.**
- **If there are interests in some of the models discussed, a separate training will be prepared on those subject matters.**

Contact

Questions and Information

https://training.hpc.mil/

pet@hpc.mil

Gedion.R.Teklemariam.ctr@mail.nrl.navy.mil

USER PRODUCTIVITY ENHANCEMENT AND TRAINING

GENERAL DYNAMICS Information Technology