Solution Brief

Analytics
Databricks Runtime for Machine Learning

Accelerating Databricks Runtime for Machine Learning

Intel®-optimized ML libraries improve performance for data scientists

Databricks, a unified data-analytics platform for data engineering, machine learning (ML), and collaborative data science, runs on Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform. Data scientists interested in improving the performance of Databricks while reducing costs can use Intel libraries with optimized versions of scikit-learn and TensorFlow.

Getting the best possible performance out of an application always presents challenges. This fact is especially true when developing machine learning (ML) and artificial intelligence (AI) applications. Over the years, Intel has worked closely with the ecosystem to optimize a broad range of frameworks and libraries for better performance.

This brief discusses the performance benefits derived from incorporating Intel-optimized ML libraries into Databricks Runtime for Machine Learning on 2nd Generation Intel® Xeon® Platinum processors. The paper focuses on two of the most popular frameworks used in ML and deep learning (DL): scikit-learn and TensorFlow.

Intel-optimized ML libraries

The Intel oneAPI AI Analytics Toolkit gives data scientists, AI developers, and researchers familiar Python tools and frameworks to accelerate end-to-end data science and analytics pipelines on Intel architecture. The components are built using oneAPI libraries for low-level compute optimizations. This toolkit improves performance from preprocessing through ML, and it provides interoperability for efficient model development. Two popular ML and DL frameworks are scikit-learn and TensorFlow.

Scikit-learn

Scikit-learn is a popular open source ML library for the Python programming language. It features various classification, regression, and clustering algorithms, including support for:

- Vector machines
- Random forests
- Gradient boosting
- k-means
- DBSCAN

This ML library is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

The Intel Extension for Scikit-learn, available through the Intel oneAPI AI Analytics Toolkit, can help boost ML performance and give data scientists more time to focus on their models. Intel has invested in optimizing the performance of Python itself with the Intel Distribution for Python, and it has optimized key data science libraries used with scikit-learn, such as XGBoost, NumPy, and SciPy. For more information on using these extensions, read the following article: https://medium.com/intel-analytics-software/save-time-and-money-with-intel-extension-for-scikit-learn-33627425ae4.

TensorFlow

TensorFlow is another popular open source framework for developing end-to-end ML and DL applications. It has a comprehensive, flexible ecosystem of tools and libraries that lets researchers easily build and deploy applications.

To take full advantage of the performance available in Intel processors, TensorFlow has been optimized using Intel oneAPI Deep Neural Network Library (oneDNN) primitives. For more information on these optimizations, in addition to performance data, refer to the article, "TensorFlow Optimizations on Modern Intel Architecture," available here: https://software.intel.com/content/www/us/en/develop/articles/tensorflow-optimizations-on-modern-intel-architecture.html.

Databricks Runtime for Machine Learning

Databricks is a unified data-analytics platform for data engineering, ML, and collaborative data science. It offers comprehensive environments for developing data-intensive applications. Databricks Runtime for Machine Learning is an integrated end-to-end environment that incorporates:

- Managed services for experiment tracking
- Model training
- Feature development and management
- Feature and model serving.

It includes the most popular ML/DL libraries, such as TensorFlow, PyTorch, Keras, and XGBoost, and it also includes libraries required for distributed training, such as Horovod. For more information, visit the Databricks web page at https://docs.databricks.com/runtime/mlruntime.html.

Databricks has been integrated with Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform. These cloud service providers (CSPs) bring great convenience to managing production infrastructure and running production workloads. Though cloud services aren't free, there are opportunities to reduce the cost of ownership by using optimized libraries. This article uses Databricks on Azure to demonstrate the solution and the performance results achieved in Intel testing.

Intel-optimized ML libraries on Azure Databricks

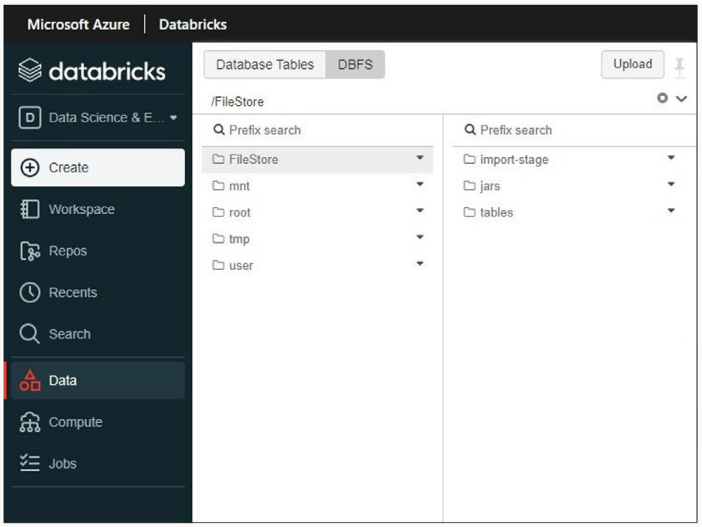
Databricks Runtime for Machine Learning includes the stock versions of scikit-learn and TensorFlow. To boost performance, however, Intel engineers replaced these versions with Intel-optimized versions and tested the results. Databricks provides initialization scripts to facilitate customization at https://docs.databricks.com/clusters/init-scripts.html.

These scripts run during the startup of each cluster node. Intel also developed two initialization scripts to incorporate the Intel-optimized versions of scikit-learn and TensorFlow. The right script for your needs depends on whether you want the statically patched version or not:

- init_intel_optimized_ml.sh installs statically patched scikit-learn and TensorFlow in the runtime environment.
- init_intel_optimized_ml_ex.sh installs the Intel Extension for Scikit-learn and TensorFlow in the runtime environment.

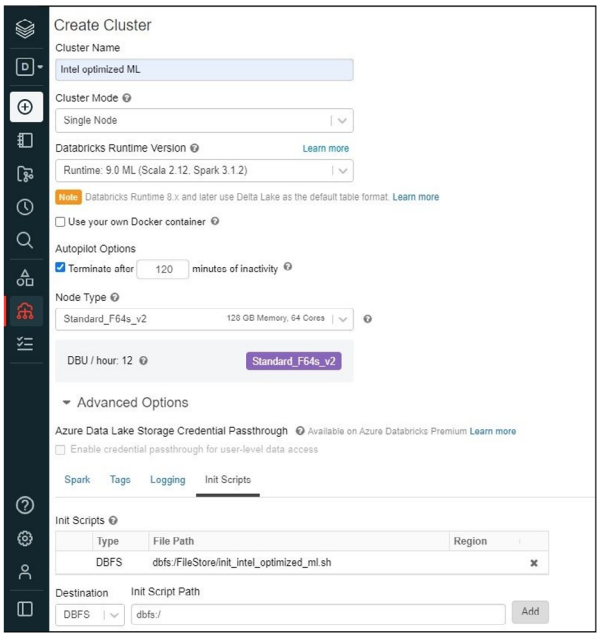
The following instructions describe how to create a cluster using either script. First, copy the initialization script to Databricks File System (DBFS) by completing the following steps:

1. Download either **init_intel_optimized_ml.sh** or **init_intel_optimized_ml_ex.sh** to a local folder.
2. In the left sidebar, click the **Data** icon.
3. Click the **DBFS** button, and then click **Upload** at the top. If the DBFS button isn't shown, follow the guidance provided in "Manage the DBFS file browser" to enable it.
4. In the **Upload Data to DBFS** dialog, select a target directory (for example, *FileStore*).
5. Browse to the local file that you downloaded to the local folder, and then upload it in the **Files** box.



Next, launch the Databricks cluster using the uploaded initialization script:

1. On the **Cluster Configuration** page, click the **Advanced Options** toggle.
2. At the bottom-right, click the **Init Scripts** tab.
3. In the **Destination** drop-down menu, select the **DBFS** destination type.
4. Specify the path to the previously uploaded initialization script (that is, **dbfs:/FileStore/init_intel_optimized_ml.sh** or **dbfs:/FileStore/init_intel_optimized_ml_ex.sh**).
5. Click **Add**.



Refer to the Intel optimized ML for Databricks guidance at https://github.com/oap-project/oap-tools/tree/master/integrations/ml/databricks for more detailed information.

Performance measurements

The Intel engineers compared scikit-learn and TensorFlow performance for both training and prediction between two Databricks clusters. The baseline cluster was created using default libraries without the initialization script. The other cluster was created using Intel-optimized ML libraries through specifying the initialization script discussed previously.

Scikit-learn training and prediction performance

Intel used Databricks Runtime 9.0 for Machine Learning with the following benchmarks for this testing. The Intel engineers used scikit-learn_bench (https://github.com/IntelPython/scikit-learn_bench) to compare the performance of common scikit-learn algorithms with and without the Intel optimizations. The benchmark_sklearn.ipynb notebook (https://github.com/oap-project/oap-tools/blob/master/integrations/ml/databricks/benchmark/benchmark_sklearn.ipynb) is provided for convenience to run scikit-learn_bench on a Databricks cluster.

Intel compared training and prediction performance for the libraries by creating one single-node Databricks cluster with the stock library and another using the Intel-optimized version. Both clusters used the Standard_F16s_v2 Azure instance type.

The benchmark notebook was run on both clusters. The Intel engineers set multiple configurations for each algorithm to get accurate training and prediction performance data (for details, see https://github.com/oap-project/oap-tools/blob/master/integrations/ml/databricks/benchmark/skl_config_databricks.json). Intel tested multiple configurations for each algorithm. Table 1 shows the performance data of one configuration for each algorithm.

Table 1. Comparing the training and prediction performance of stock libraries and Intel-optimized libraries (all times in seconds)

Algorithm	Input configuration	Training time (seconds)		Prediction time (seconds)	
		Stock scikit-learn (baseline)	Intel Extension for Scikit-learn	Stock scikit-learn (baseline)	Intel Extension for Scikit-learn
kmeans	config1	17.91	17.87	3.76	0.39
ridge_regression	config1	1.47	0.10	0.07	0.06
linear_regression	config1	5.03	0.10	0.07	0.06
logistic_regression	config3	74.82	6.63	0.62	0.08
svm	config2	173.81	10.61	49.90	0.46

The Intel-optimized version of scikit-learn greatly improved training and prediction performance for each algorithm. For some algorithms, like svm and brute_knn, the Intel-optimized version of the scikit-learn library achieved an order of magnitude leap in performance. See Figures 1 and 2 for the training and prediction performance results, respectively.

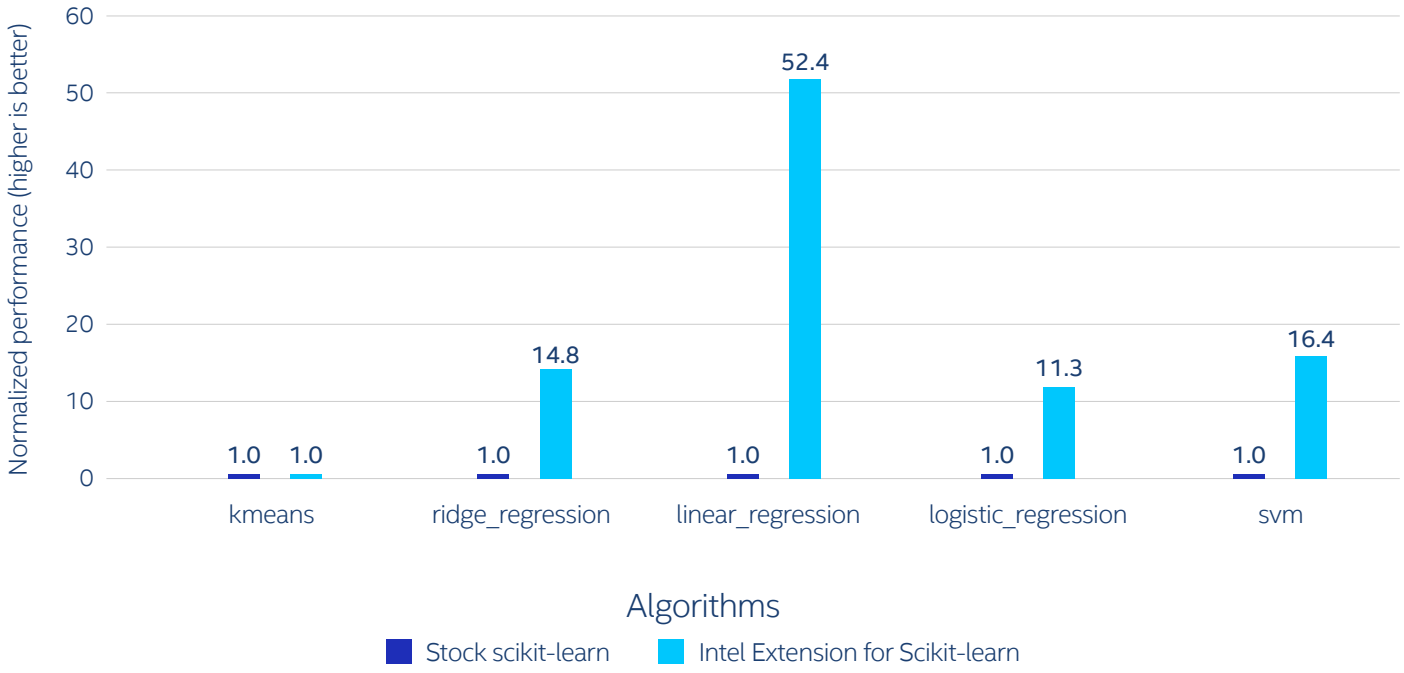


Figure 1. Training performance of the Intel-optimized scikit-learn library over the stock version

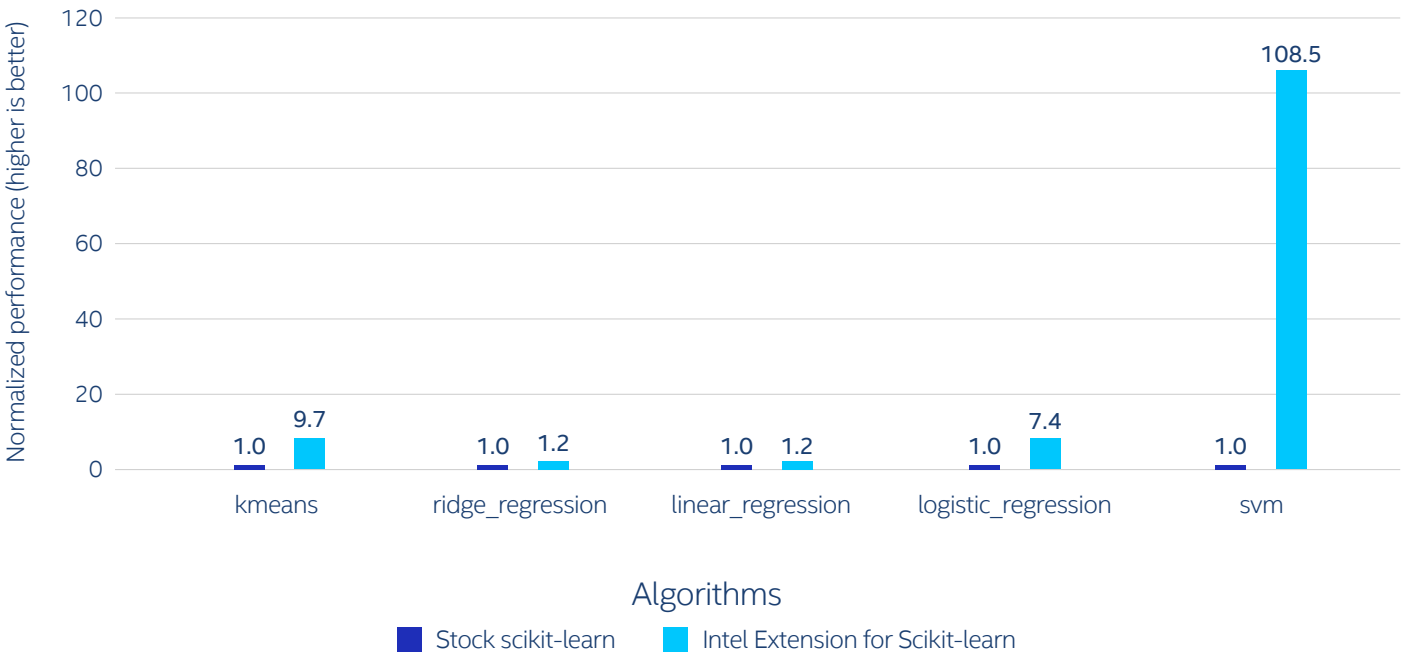


Figure 2. Prediction performance of the Intel-optimized scikit-learn library over the stock version

TensorFlow training and prediction performance

Bidirectional Encoder Representations from Transformers, or BERT (https://github.com/google-research/bert), is a new method of pre-training language representations. This method obtains state-of-the-art results on a wide range of natural language processing (NLP) tasks. Model Zoo (https://github.com/IntelAI/models) contains links to pretrained models, sample scripts, and step-by-step tutorials for many popular open-source ML models optimized to run on Intel Xeon Scalable processors.

The Intel engineers used Model Zoo to run the BERT Large (https://github.com/IntelAI/models/tree/v1.8.1/benchmarks/language_modeling/tensorflow/bert_large/README.md) model on SQuADv1.1 datasets to compare the performance of TensorFlow with and without Intel's optimizations. Once again, the team provides a notebook (benchmark_tensorflow_bertlarge.ipynb) to run the benchmark on the Databricks cluster. For more details, refer to "Run Performance Comparison Benchmarks" at https://github.com/oap-project/oap-tools/tree/master/integrations/ml/databricks/benchmark.

The Intel engineers used a single-node Databricks cluster with Standard_F32s_v2, Standard_F64s_v2, and Standard_F72s_v2 instance types for the TensorFlow performance evaluation. For each instance type, the Intel engineers compared the inference and training performance between the stock TensorFlow and Intel-optimized TensorFlow libraries.

The testing found that the latter delivers 1.92x, 2.12x, and 2.24x inference performance on Databricks Runtime for Machine Learning with Standard_F32s_v2, Standard_F64s_v2, and Standard_F72s_v2 instances, respectively (see Figure 3). For training, the Intel-optimized TensorFlow library delivers 1.93x, 1.76x, and 1.84x training performance on Standard_F32s_v2, Standard_F64s_v2, and Standard_F72s_v2 instances, respectively (see Figure 4).

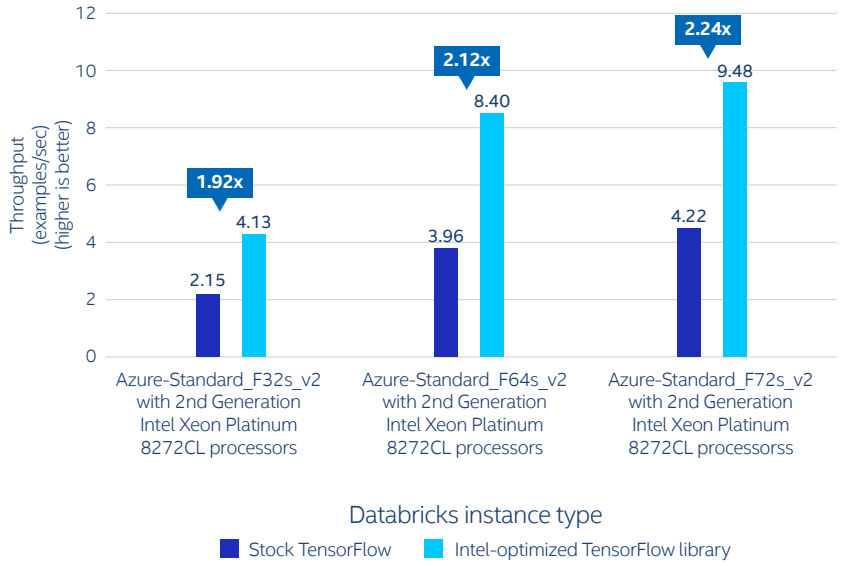


Figure 3. Inference speedup of the Intel-optimized TensorFlow library over the stock version; performance varies by use, configurations, and other factors[1]

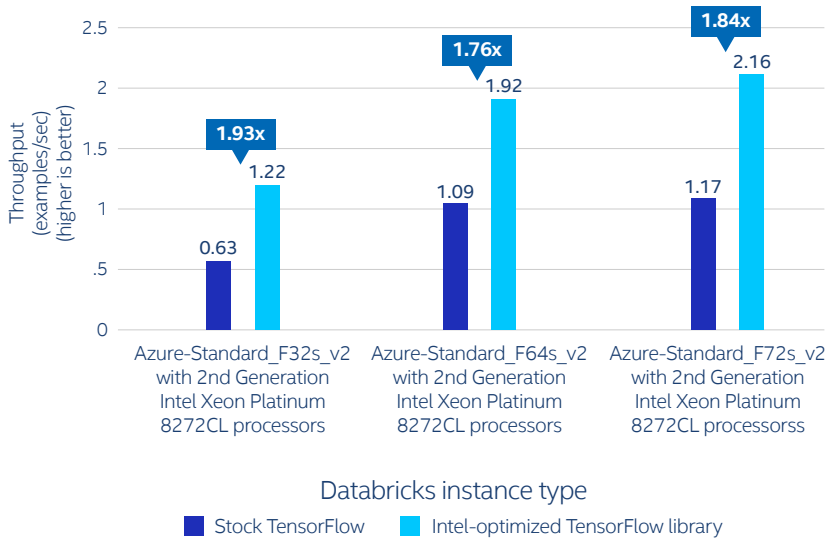


Figure 4. Training speedup for the Intel-optimized TensorFlow library over the stock version; performance varies by use, configurations, and other factors[1]

Concluding remarks

The Intel-optimized versions of the scikit-learn and TensorFlow libraries deliver significant improvements in training and inference performance on Intel XPUs. Intel has demonstrated that organizations can improve performance and reduce costs by replacing the stock scikit-learn and TensorFlow libraries included in Databricks Runtime for Machine Learning with the Intel-optimized versions.

When Databricks adds support for Databricks Container Services (https://docs.databricks.com/clusters/custom-containers.html) to the Databricks Runtime for Machine Learning, Intel will explore incorporating these optimized libraries through Docker images. This approach could make it easier to integrate with your continuous integration/continuous deployment (CI/CD) pipelines as part of your MLOps environment.

For more information, visit: https://github.com/oap-project/oap-tools/tree/master/integrations/ml/databricks

---

Haifeng Chen is a software engineering manager at Intel, based in China. He and his team work on exploring AI/ML optimizations on Intel architecture to deliver superior performance and lower cost for customers.

He can be reached at haifeng.chen@intel.com.

Lakshman Chari is a cloud ISV partner manager at Intel, based in the US. He works with ISVs in the data analytics and AI domain to ensure that they're taking advantage of all the optimizations and technologies available with the latest Intel processors on the cloud. Doing so helps ensure that their end customers can benefit from higher performance and lower cost.

He can be reached at lakshman.chari@intel.com.

Printed in USA 1021/SMR/PRW/PDF Please Recycle 348450-001US

[1] Based on Intel testing as of September 23, 2021. **Baseline:** Processing times speedup: Intel-optimized TensorFlow/BERT-large: Azure-US-West, Standard_F32s_V2, 32 vCPUs, 1 instance, Intel Xeon Platinum 8168 processor at 2.70 GHz/Intel Xeon Platinum 8272CL processor at 2.60 GHz, 64 GB memory capacity/instance, direct-attached storage, Ubuntu 18.04.5 LTS, 5.4.0-1051-azure, Databricks Runtime 9.0 for ML, stock TensorFlow 2.3.1 vs. Intel TensorFlow 2.3.0. **New:** Processing times speedup: Intel-optimized TensorFlow/BERT-large: Azure-US-West, Standard_F64s_V2, 64 vCPUs, 1 instance, Intel Xeon Platinum 8168 processor at 2.70 GHz/Intel Xeon Platinum 8272CL processor at 2.60 GHz, 128 GB memory capacity/instance, direct-attached storage, Ubuntu 18.04.5 LTS, 5.4.0-1051-azure, Databricks Runtime 9.0 for ML, stock TensorFlow 2.3.1 vs. Intel TensorFlow 2.3.0. **New**: Processing times speedup: Intel-optimized TensorFlow/BERT-large: Azure-US-West, Standard_F72s_V2, 72 vCPUs, 1 instance, Intel Xeon Platinum 8168 processor at 2.70 GHz/Intel Xeon Platinum 8272CL processor at 2.60 GHz, 144 GB memory capacity/instance, direct-attached storage, Ubuntu 18.04.5 LTS, 5.4.0-1051-azure, Databricks Runtime 9.0 for ML, stock TensorFlow 2.3.1 vs. Intel TensorFlow 2.3.0.