Prediction model training data generation guide (Paper 62)

Artifact Evaluation

1. Overview

The *SpTRSV prediction framework* is a tool for automated prediction of the fastest sparse triangular solve (SpTRSV) algorithm for a given input sparse matrix on a CPU-GPU platform. It is based on a machine learning-based prediction model that is trained using features and SpTRSV performance of six algorithms mentioned in the paper for a pre-selected set of real, square matrices from SuiteSparse matrix collection. The matrices have sizes ranging from 1K to over 16M rows. To facilitate the collection of training data on a given platform, we have automated the feature extraction and SpTRSV performance data collection using a tool that can be used to generate the training data with a few simple commands on a given CPU-GPU platform.
In this guide, we outline the prerequisites and the instructions for using our automation tool to generate the training data for our SpTRSV prediction framework.

2. Pre-requisites

In this section, we explain the procedures for installing necessary tools for building and running the training data generation tool. This section assumes that instructions laid out in the Section 2 (Getting started Guide) of the overview document have been successfully completed.

2.1. Installing SSGet

SSGet is a command line tool for downloading and working with matrices from SuiteSparse matrix collection. The tool is available at https://github.com/ginkgo-project/ssget. To download the SSGet repository code and install the tool locally on your Linux system, use the following commands:

```
> git clone https://github.com/ginkgo-project/ssget.git
> cd ssget
> sudo make install
```

2.2. The libUFget library installation

The SpTRSV framework uses the libUFget library to automatically download sparse matrices from Suite Sparse matrix collection. The library is freely available from https://zenodo.org/record/897632#.Xra9VCkzbmE and is also included in *thirdparty* folder. On a Ubuntu system, the following sequence of commands should be sufficient for installation of libUFget library:

The following commands will unzip the zipped file and change the working directory to the libUFget main directory

```
> tar -xvzf libufget-1.0.2.tar.gz
> cd libufget-1.0.2
```

The following command will install all the dependencies for libUFget (Ubuntu)

```
> sudo apt-get install cmake libmatio-dev libblas-dev libsqlite3-dev \
                       libcurl4-openssl-dev libarchive-dev liblzma-dev
```

Finally, the library can be configured, built and installed by the following commands

```
> mkdir build && cd build
> cmake ../ -DCMAKE_INSTAL_PREFIX=<YourInstallPath>
> make
> make install
```

Installation instructions for other Linux versions are listed in the INSTALL.md file provided with the libufget library.

2.3. Installing Anaconda

Our framework depends on a number of Python libraries like Scikit-learn, Pandas, matplotlib, numpy, scipy etc. that can be installed individually on Linux. However, it is more convenient to install Anaconda data science platform that includes all of these platforms by default. The installation package for the 64-bit Individual version of Anaconda for Python3.7 can be downloaded with the following command line:

```
> wget https://repo.anaconda.com/archive/Anaconda3-2020.02-Linux-x86_64.sh
```

Alternatively, the installation package for the latest Individual version of Anaconda can be downloaded from https://www.anaconda.com/products/individual.

Verify data integrity of the installation package and install Anaconda with the following commands:

```
> sha256sum Anaconda3-2020.02-Linux-x86_64.sh
> bash Anaconda3-2020.02-Linux-x86_64.sh
```

Follow the prompts during the installation to complete the installation. For more comprehensive details, refer to the link: https://docs.anaconda.com/anaconda/install/linux/

2.4. Installing CUDA toolkit

For our research paper, experiments are performed using NVIDIA CUDA version 10.1 on Ubuntu 18.03. Instructions for the installation of the same are listed here in this section. For other operating system, instructions can be found at https://docs.nvidia.com/cuda/cuda-quick-start-guide/index.html.

- Check existence of NVIDIA PCI V100 GPU device. Verify that output of the following command shows presence of NVIDIA V100 GPU

```
> lspci | grep -i nvidia
```

- Check that gcc compiler is installed with `gcc --version` command. If not found, it can be installed with the following command:

```
> sudo apt install gcc
```

- Install linux headers using the following commands:

```
> HDR="linux-headers-"
> KRNL_VER=$(uname -r)
> HEADERS="$HDR$KRNL_VER"
> sudo apt-get install $HEADERS
```

- Download and install CUDA toolkit 10.1 as follows:

```
> sudo curl -O
http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_6
4/cuda-repo-ubuntu1804_10.1.243-1_amd64.deb
> sudo dpkg -i ./cuda-repo-ubuntu1804_10.1.243-1_amd64.deb
> sudo apt-key adv --fetch-keys
http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_6
4/7fa2af80.pub
> sudo apt-get update
> sudo apt-get install cuda-10-1 -y
```

Verify installation of CUDA toolkit with the following command:

```
> nvidia-smi -pm 1
```

2.5. Installing Intel Parallel Studio XE

Our framework requires Intel Parallel Studio for intel compiler and Intel MKL library. The full-featured student or academic version or 30-day free trial version can be downloaded and installed from
https://software.intel.com/content/www/us/en/develop/tools/parallel-studio-xe/choose-download.html
To download the appropriate version, choose Student, Classroom Educators or directly download for Linux for the link provided under “Try It Now”. The students and classroom educators are required to accept a few statements after which they are asked to provide their university email address to continue with the download and installation. Once registered, you will be provided a serial number and a download link for a customizable package (online installation) and Full Package (online installation). Use full package if you do not have storage issues on your computer as the full package download size is around 2.8 GB. In online installation package, only the selected components are downloaded and installed. Both installers provide graphical user interface as well as command line installation interface. Once the package is downloaded and unzipped, the installer command line interface or the GUI interface can be activated with `sudo ./install.sh` or `sudo ./install_GUI.sh` respectively from the main directory of the installation package. Complete step-by-step installation guide for both command line and GUI installation methods can be found in section 3 of the Installation Guide which can be downloaded from https://software.intel.com/content/www/us/en/develop/download/parallel-studio-xe-2020-install-guide-linux.html. The installer requires around 12 GB of hard disk space to install all components. For the purpose of this artifact, main packages required are Intel MKL library and Intel C/C++ compiler packages. Rest of the packages like Intel VTune amplifier, Intel Advisor, DAAL, GDB can be skipped for installation in the customize installation section of the installer to save disk space. Once installation is completed, follow steps in section 5 of the install guide to complete the installation.

3. Step-by-Step Instructions on how to generate training data for the prediction model

To facilitate generation of training data for the prediction model on a CPU-GPU platform, we have developed a Python script **datasetgenerator.py** which is stored in ./datasets/datasetGenerator/ path in the main directory. This script utilizes executable files of the each of the six algorithms in SpTRSV algorithm repository (stored in ./src folder) and our feature extraction tool. The following subsections describe step-by-step procedure on how to use the script and generate the training data.

3.1. Building the Feature Extraction tool/SpTRSV algorithm repository

Before beginning the training data generation process, it is required to build binaries for the feature extraction tool and each SpTRSV algorithm in SpTRSV algorithm repository. For this purpose, we have provided **install.sh** Bash script in the main directory. To build the binaries on a Linux system, use the following command line:

```
> ./build.sh
or
> bash ./build.sh
```

If the process is successfully completed, the output contains build SUCCESS messages for the tools. Any FAILURE message indicates a problem and should be resolved before proceeding with the following sections.

Once steps in section 3.1 are completed, **datasetgenerator.py** file in ./datasets/datasetGenerator/ folder accessible from the main directory can be used for both feature extraction and SpTRSV algorithm performance data collection.

3.2. Feature Extraction

For extraction of the selected features (listed in Table 2 in the accepted paper), **datasetgenerator.py** uses the feature extraction tool (./src/matrix_feature_extractor) and the SuiteSparse matrix IDs stored in file ‘./datasets/datasetGenerator/UF Sparse data set - Matrix list.csv’ file accessible from the main directory. The tool automatically downloads the matrices from the SuiteSparse collection, extract features and stores the features of each matrix, line-by-line, in ./datasets/datasetGenerator/data/Features.csv’ file accessible from the main directory. This ‘Features.csv’ file can be used instead of ./datasets/Features.csv’ file to validate the results in the paper as discussed in the overview document. To start the feature extraction process, use the following command from the main directory:

```
> python3 ./datasets/datasetGenerator/datasetgenerator.py
featureextraction
```

The feature extraction process may take several hours to complete.

3.3. SpTRSV Algorithm Performance Data

In addition to the matrix features, SpTRSV performance data for each of the matrices is required for training the prediction model. The **datasetgenerator.py** script uses the SpTRSV algorithms stored in ./src/ directory accessible from the main directory. If not already downloaded, the tool automatically downloads each matrix from the SuiteSparse

collection, runs each of the six SpTRSV algorithm, determines the ranking of each of the algorithm and stores the result in ‘./datasets/datasetGenerator/data/perfdata.csv’. The performance data collection can be started by using the following command from the main directory:

```
> python3 ./datasets/datasetGenerator/datasetgenerator.py perfdata
```

The algorithm performance data collection process may take several hours to complete.

3.4. Training Data Generation

To generate the training data file, use the following command:

```
> python3 ./datasets/datasetGenerator/datasetgenerator.py trainingfile
```

This command generates the training data file stored as ‘./datasets/datasetGenerator/data/Training_data.csv’. This ‘Training_data.csv’ file can be used instead of ./datasets/‘Training_data.csv’ file to validate the results in the paper as discussed in the overview document.