GSoC'19

Cythonizing Parts of Dask's Core Algorithms

By - Harmohit Singh

Organization - Numfocus

About the Developer

I, **Harmohit Singh** am pursuing Bachelor of Computer Engineering from **Netaji Subhas University of Technology, Delhi**. Currently, I am attending 6th semester and have been involved in a bunch of projects as well as Open Source Programs. So far, **Algorithms and Data Structures**, Software Engineering, **High Performance Computing**, **Compiler Design** and Artificial Intelligence are some subjects i had opted in my University.I have also worked on **Xception model in Keras** as my **primary project** in **Smart India Hackathon'19**. I have been involved in High Performance Computing, Web Development and a bit of Machine and Deep Learning.

Email : harmohitsingh05@gmail.com

University - Netaji Subhas University of Technology

Github : www.github.com/HSR05

Gitter : https://gitter.im/HSR05

Phone : +91-7888763776

Experience with programming:

- I have good experience with programming. I am in my 6th semester of Computer Engineering program. I have been coding in C++ , Python and Javascript. I am also familiar with **High Performance Computing**.

  I have created along with a team of 4 other students an **Automated Visual Inspection Tool for Oily components** and implemented **Xception model** of Deep learning.

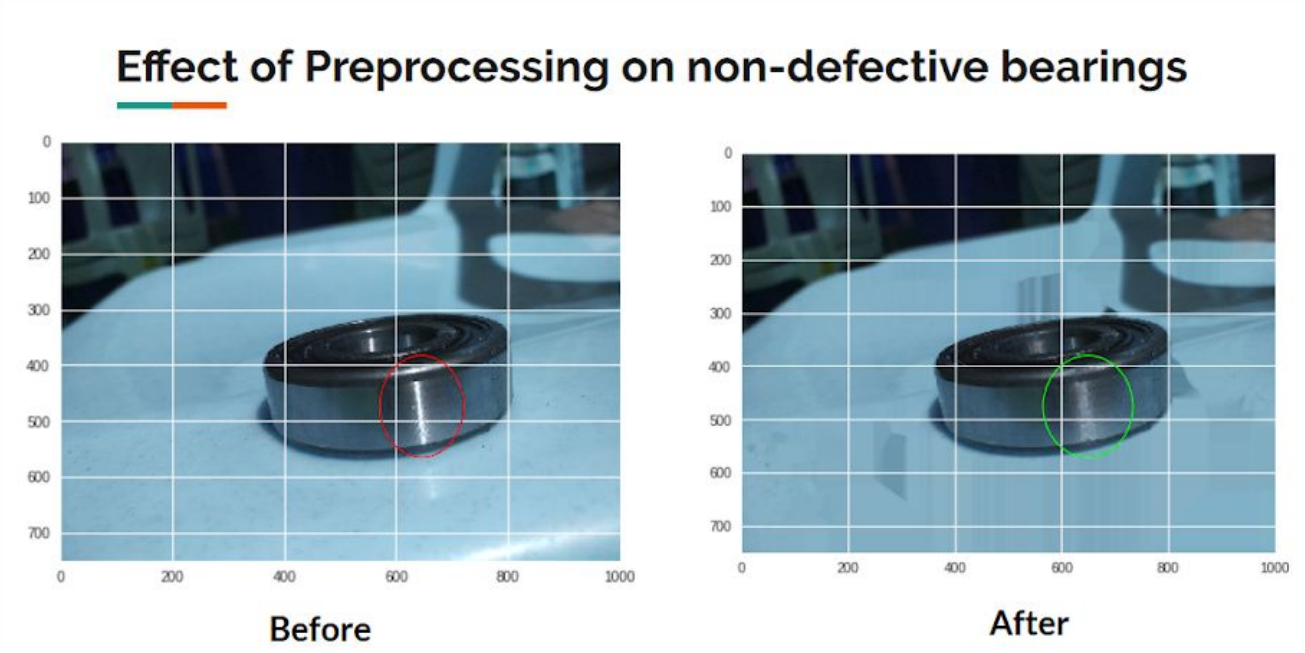
  Our solution comprised of a deep learning model comprising of multiple Convolutional Neural Networks (CNNs) which acts as an ensemble of predictors. Each CNN saw a different angle of the Bearing from the input images taken from a setup of cameras and make a prediction.

We worked and researched on these algorithms to find the most efficient model for the automated visual inspection of the oily components.

Model / Characteristics	Alex Net	VGG-16	ResNet-18	Xception	Inception-V4
Layers	8	16	18	36	**96**
Evaluation Time	2	3	4	**1**	5
Model size	62M	125M	**10M**	15M	35M
Performance/ Accuracy	54%	72%	69%	79%	**81%**
Complexity	**Low**	**Low**	Medium	High	Very High
Parallel Stages	1	1	2	**3-7**	**3-7**
Total Score =	17	20	18	**14**	20
Final Rank =	2	5	3	**1**	4

We analyzed and ranked each model on basis of the given characteristics And decided to implement the **Xception** model using Keras.

We preprocessed the image to remove the glare due to the oil. It was implemented using OpenCV's impainting algorithm.



Experience with python

- I have been coding in python for ~2 years. I worked as a research intern at Embit plus where the research was focused on **Latest Developments in the field of Web Development.**
  I was enrolled in a course of Python and Django with the same institute which was finished by submitting a fully functional website using Python and Django.
- I have mostly been a self taught programmer and i have also done a few side projects in Python:
  Web Scraper using Beautifulsoup.

- I also have experience with parallel computing using threads, OpenMP and MPI (Here is a list of programs I tried out with our team of three people including me to learn more about parallel computing)

Experience with Git and Github

- I am very much familiar with Git. I have used Github for few projects and I am familiar with the working of the version control systems.

Experience with Open Source:

I have been very impressed with Open Source Softwares. It has not been very long since I have started contributing to Open Source community via various organisations. The journey till now has been amazing. The community is full of helping developers who guide through difficult cases and teach how to tackle the bugs. I have learnt a lot from the Open Source Community and I look forward to become helping developers for other newcomers.

Some Open Source Projects I've been associated with:

- Mozilla bugbug
  Mozilla bugbug is a Platform for Bugzilla Machine learning projects. I started contributing in this by cleaning the data which is still ongoing.

- Mozilla foundation.mozilla.org
  It is the Mozilla Foundation website. The Mozilla Foundation believes the Internet is a global public resource that must remain open and accessible to all. I solved few issues and also contributed in code clean-up. You can see the merged PR - #2779

- Dask dask
  Dask is a flexible parallel computing library for analytics. Dask nativey scales Python. Dask provides advanced parallelism for analytics, enabling performance at scale for the tools that people love. I have solved few issues and created a PR and have started using dask for the computation work.

Project Description

Google Summer of Code 2019 is around 12 weeks long. My main **Aim** will be to maximize my efforts in this timeframe, in order to speed up the core parts of Dask using Cython.

Other tasks to be carried out during the same time-frame:
- Profiling the core parts and algorithms of Dask
- Analyzing algorithms and discussing about potential speedup
- Using Cython for speed

Why Dask

Dask is a flexible library for parallel computing in Python which makes it unique and also the flexibilty in scaling, speed and also familiarity with NumPy arrays and Pandas Dataframe objects. I really appreciate the community and the work it has been doing to create the positive environment for the newcomers. I have been using dask for quite a while now and I have become very much interested in the working of dask and I have been reading the documentation, development guidelines and Scheduling policies after

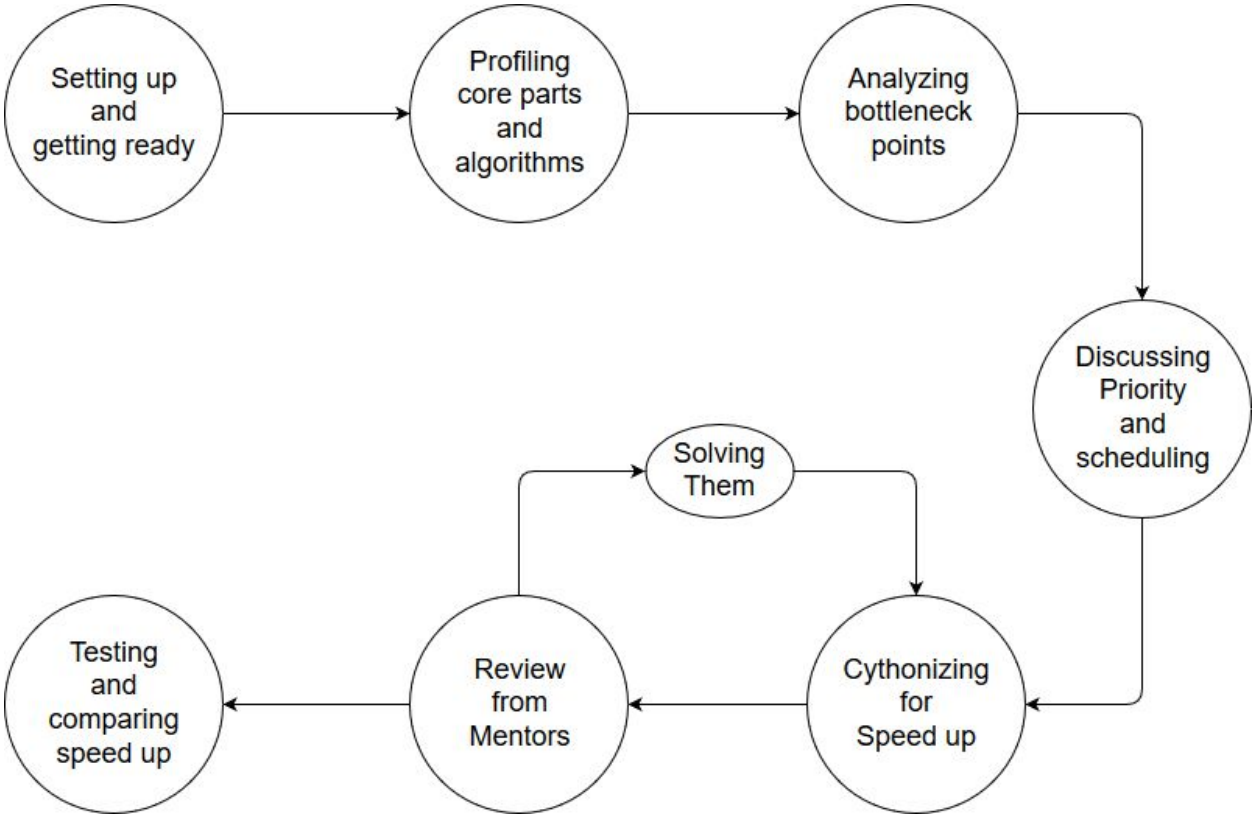
reading about the idea of Cythonizing the core parts of Dask because optimization is one of the most desirable goal in the making of a program and if one can optimize a program then the person is able to think in a particular way of implementing algorithms and data structures in an efficient manner which increases the quality of programming of a developer. I really look forward to contribute to a tool that is so widely used by the community of developers and to be able to say that I have worked on Dask is something that I would love.

Why me

I am a self-taught programmer who learns quickly and is very eager to learn new things and try to handle things on my own with a little guidance. I am good at working in a team of professionals and I also take my work seriously. I have the required programming skills and I am also familiar with the developer guidelines. To understand the working and usage of Dask I already started to work upon some issues like removing warning for deprecated pandas(#4467), adding updated arguments in the functions(#4579), currently adding the support for bytes chunksize in dd.from_pandas (#4555). I have also started working on good first issues in Cython to get familiar with it. I can relate optimising and speeding with my course content in High performance computing. I am familiar with optimization techniques, profiling and the penalties that we deal with if the program is not optimized. I also have experience with algorithms and Data structures which will be useful in implementing the core parts of Dask in an efficient manner. I will not mind working in late night if time zones come up to be an issue.

Implementation:

We will discuss about how the project will be implemented, the lifecycle of the project using all the available resources and how the project will be carried out in GSoc '19.



The GSoC '19 will follow this pattern:

- **Setting up and getting ready**

  Setting up the Dask environment onto the local machine prior the GSoC '19 period to get

  comfortable with the working. Also setting up environment for Cython and get familiar

  with it and work with mentors to begin nicely.

- **Profiling core parts like scheduler, algorithms and datastructures**

  Profiling is one of the most important step towards high performance. It is used to find out where exactly does a program uses most of its time. There are many tools that can be used to quantify the run time of the program. Our main focus for profiling will be on the core parts like **scheduler, algorithms and data structures**. Which can be efficiently written in Cython to improve the speed.

- **Analyzing performance bottlenecks**

In this phase,we will analyze the core parts to find where the performance bottlenecks are to efficiently to increase the speed of the computation speed. This will help in keeping the native code to the minimum, while providing high speed up.

- **Review from Mentors**
  Mentors will provide their feedback on the bottleneck points that are going to be prioritized for increasing the efficiency. The mentors' input will be respected and worked upon.

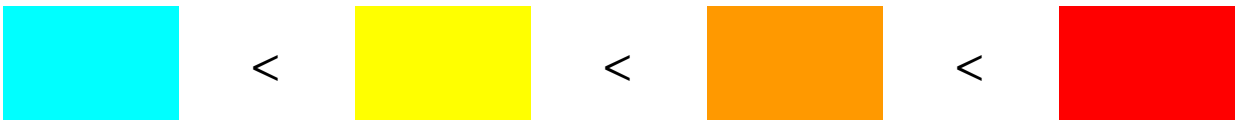
- **Cythonizing the Core parts**
  In this phase, the modules will be written for the core parts of the Dask in Cython and will
  be sent over to the mentors regularly for reviews.

- **Review from Mentors**
  Mentors will provide their respective input to the work done in a particular part. Possibilities will be discussed about what can be further changed and done in a better and efficient way and finally correct them to produce desired results.

- **Testing**
  Once a part of the Dask is Cythonized, then we can carry on with the testing phase of the application.

In addition to this, I will be writing blog posts and submitting scrums for the same.

The order of applied **efforts and devoted time** is given below:



Timeline prepared for GSoC 2019 is presented below

Time	Work	Efforts & Time
6 May 2019 - 26 May 2019	**Community Bonding Period** ● Discussions with Mentors regarding potential speed up in core parts. ● Discuss about issue #854 in dask/dask. ● Talk about possible speed up using threads if we achieve the desired speed up by implementing algorithms and data structures efficiently in Cython.	
27 May 2019- 10 June 2019	**Coding Period** ● Understanding and solving previous bugs and issues. ● Profiling the Scheduler under various problems to find slow points ● Profiling other core parts like Graph algorithms and other possible bottleneck points. ● Make a priority list from the cumulative time obtained by profiling. ● Work with 90/10 rule that is to optimize 10% of the code to obtain at least 90% speedup. ● Finalize parts to be cythonized. ● Compare various optimization techniques. ● Start Cythonizing the core.	
11 June 2019- 23 June 2019		
Extras	● Write blog posts about my GSoC '19 journey. ● Daily scrums will be documented.	
24 June 2019- 28 June 2019	**1st Evaluation** ● Get the work evaluated by Mentors	

	• Work on their reviews and make changes in the priority list, if required.	
29 June 2019- 11 July 2019	**Coding period** • Start Cythonizing the bottleneck points in scheduler. • Discuss the algorithms that can be used to speed up the computations. • Implement efficient Data Structure like Hash tables which give good performance. • Make a PR in to use the efficient algorithms and data structures. As more code is made efficient PRs can be sent to avoid 'code dump' in the of the project. • Modify the priority list if needed. • Write algorithms and data structures in efficient manner to get maximum speed up. • Perform tests for the Cythonized parts and compare the increased efficiency with previous benchmarks.	
12 July 2019- 21 July 2019		
Extras	• Write blog posts about my GSoC '19 journey. • Daily scrums will be documented.	
22 July 2019- 26 July 2019	**2nd Evaluation** • Get the work reviews by Mentors • Work on their reviews and implement the changes requested.	

27 July 2019- 10 August 2019	**Coding Period** ● Cythonizing other bottleneck points and writing graph algorithms efficiently to maximize the performance. ● Analyze the entire flow of algorithms and see if all standard algorithmic speed up techniques are used such as Hash tables(considering it from data types point of view) ● Comparing the speed up obtained in the core parts with the previous benchmarks and measure the percentage of increased performance.	
11 August 2019- 18 August 2019		
Extras	● Write blog posts about my GSoC '19 journey. ● Daily scrums will be documented.	
19 August 2019- 26 August 2019	**Final Evaluation** ● Submit the work ● Discuss the future of Dask with mentors and further potential of speed up using threads.	
Post GSoC 2019	● Once GSoC 2019 is over, I will continue to be a part of this organization by contributing to their projects. I have in-depth knowledge about this project and would love to get new comers get started with this organization. ● It would really be an honor to be an asset to this organization.	

Pull Request Requirement

I have been using Dask since beginning of this year and started contributing in this project when I became comfortable using it and understood its working.

Merged:

- PR 4530
  This was a fix to issue #4467, which dealt with the pandas deprecation.

Unmerged:

- PR 4581
  This was a fix to issue #4568, which was to fix Pandas warnings that showed up when we run the dask.dataframe test suite with a new version of Pandas.

- PR 4607
  This was a patch to issue #4579, where NumPy 1.11 added an optional dtype argument to RandomState.randint and dask-ml would use that. So a `dtype` argument was added and passed through `self._wrap`.