# Individual reflection

Junfan Huang

June 21, 2019

## 1 Summary

This time, we had an exploration of deal with pyspark and parallelsation. To begin with, we assumed that we have a system where contains "lots of" messages. In order to make user have a clear world, we have to make sure the system is able to detect spam, ham and some fraud messages. After that, the system should be able to encrypt the message parallelly. During this exploration, we learned how to use pyspark to do machine learning by using series classification algorithms(like Logistic Regression, Random Forests) and had a experience of doing vectorized computation. In the end, we measure the time of en/decrpyting some artificial unbalanced datasets to compare the linear speedup curves.

The dataset is not easy to build when we decided to do this, so we took some efforts to build the dataset by using the previous project code and other online resources (can be find in **documentation/preprocess/**). The dataset consists of books, spam emails, ham emails. Our first step was to distinguish this out.

## 2 Usage of pyspark

Learning pyspark is a non-trivial task. There are a number of troubles, for example, setup: how to make it executable on our own laptop; dealing with data format of pyspark; the csv reader or dataframe creator for pyspark sometimes didn't do the right job; a bad data-point will influence the whole processing. After learning some basic operation of the pyspark, we were doing word counts for the dataset by using pyspark and got some cool graphs though the frequencies of the words.

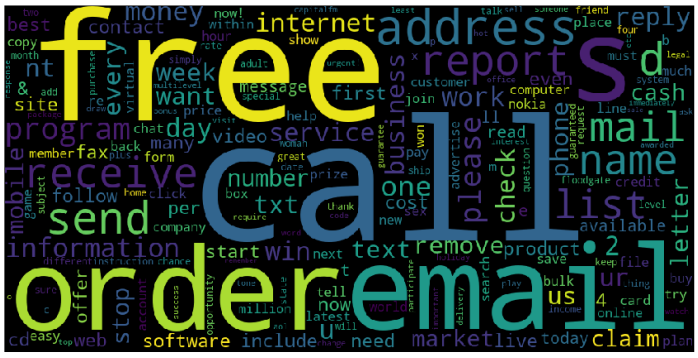


Figure 1: Word cloud graph for spam email

It is interesting to see that, for spam email, things like ”free”, ”call”, ”order” appear frequently.

After that, we applied a couple of machine learning algorithms such as Logistic Regression, Decision tree, Random forests for our dataset for a small subset. Random forests especially is an ideal algorithm for parallelisation since the training process of every tree can be treat as independent. By playing around with this, we learned the format of pyspark machine learning toolbox as well as how to do prediction and evaluation. However, it didn't work well for big dataset so we failed to evaluate the whole dataset. Part of the reason is there are some bad data points containing. Also, sometimes the memory is not enough to support the whole processing.

## 3 Encryption and decryption

We create different length of messages for encryption and decryption part as we want to visualise the time evolution of data with unbalanced length. In this part, we used Fernet, a symmetric algorithm, to encrypt every single txt message, however, because the Fernet is to decode the binary sequence of the txt message, reading messages and doing encrytion line by line is not worth to try. But in order to speed up the whole process, we used vectorized computation which is unsurprisingly fast. As the time consumption are basically the same for both en/decryption part, we are more curious about the whole time cost and the speed of unbalanced data processing. (Unbalanced data means the datasets are skewed. eg: 10% of the data points have 1000 words whereas 90% of them contains 1000000 words. It is fun to play with the percentage of the unbalanced level and see if map function in python can handle this situation to avoid some process finished faster and need to wait to continue.) It terms out that python vectorized computation can handle unbalanced data well(as the graphs shows in **report/project.ipynb**). In the project.ipynb file, it shows the linear speed up. But with the data size increasing, the slop seems to be a bit unstable curves.

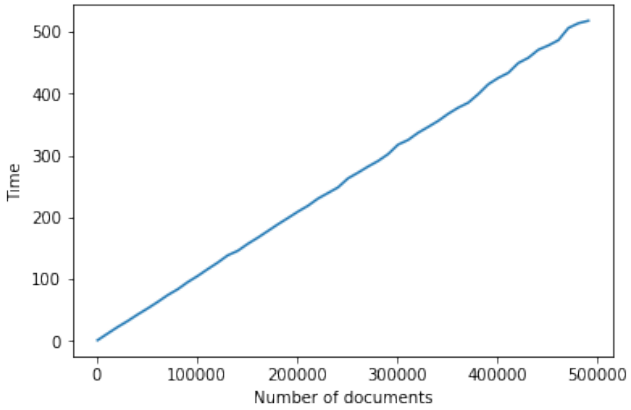


Figure 2: Linear speed up

# 4 Parallelisablility and speedup

During the exploration of this assignment, we understood that some processes are not parallelisable which limited the max speedup. For example the work dependent on the previous process is not parallelisable. I also implement the previous **Jaccard distance matrix** example by using vectorized computation which is 17 times faster than before.

In addition, we found that the communication speed is a serious issue for dealing with big data when we doing our summer project. Reading data from the hard disk is much slower (possibly hundreds of times) when compared to reading the same data from RAM directly. It is necessary to keep in mind when we deal with big data in our future work.

# 5 Unfinished things

Since pyspark contains a lot of machine learning tools, it could be interesting to play around with kdd data and previous project and doing a comparison of the speed, especially for the algorithms can be parallelised.
Another thing is because of the out of memory issue, I cannot train the whole dataset(got bad accuracy) and limit to the time I am unable to figure out the reason of this issue and got a whole better prediction.
Although we got some graphs, we don't have enough time to write a beautiful report to explain it and setup a good environment to run. Moreover, we didn't play around with the number of cores as we planned.

**Thanks for data science toolbox and your detailed feedbacks! And thanks to everyone who helped me improving this year! Hope this course will go better in the future!**