Large Datasets for Scientific Applications
A2
Mei Wu

Task 1.1: Word count example in local (standalone) mode

1. Look at the contents of the folder “output” - what are the files placed in there? What do they mean?

```
ubuntu@a2-mei:~/wordcount$ l output/
total 192
-rw-r--r-- 1 ubuntu ubuntu 196183 Apr  2 10:39 part-r-00000
-rw-r--r-- 1 ubuntu ubuntu      0 Apr  2 10:39 _SUCCESS
```

   a. The file part-r-*0 contains a list of word occurrences from the input file.
2. How many times did the word ‘Discovery’ (case-sensitive) appear in the text you analyzed?
   a. It appears 5 times.
3. In this example we used Hadoop in “Local (Standalone) Mode”. What is the difference between this mode and the Pseudo-distributed mode?
   a. In a standalone mode, Hadoop is configured to run on a non-distributed mode and thus only one java process can be run at a time. In pseudo-distributed mode, different java processes can run in parallel.

Task 1.2: Setup pseudo-distributed mode

1. What are the roles of the files core-site.xml and hdfs-site.xml?
   a. The core*.xml file is to generate a backbone for a pseudo-distributed system and informs Hadoop daemon where the core is located. The hdfs*.xml is the configuration file for the core and its constituents; datanode, secondary namenode, and namenode. It’s possible to replicate the data blocks in specified directories here.
2. Describe briefly the roles of the different services listed when executing ‘jps’.

```
ubuntu@a2-mei:~$ jps
8705 DataNode
8934 SecondaryNameNode
8502 NameNode
9208 Jps
```

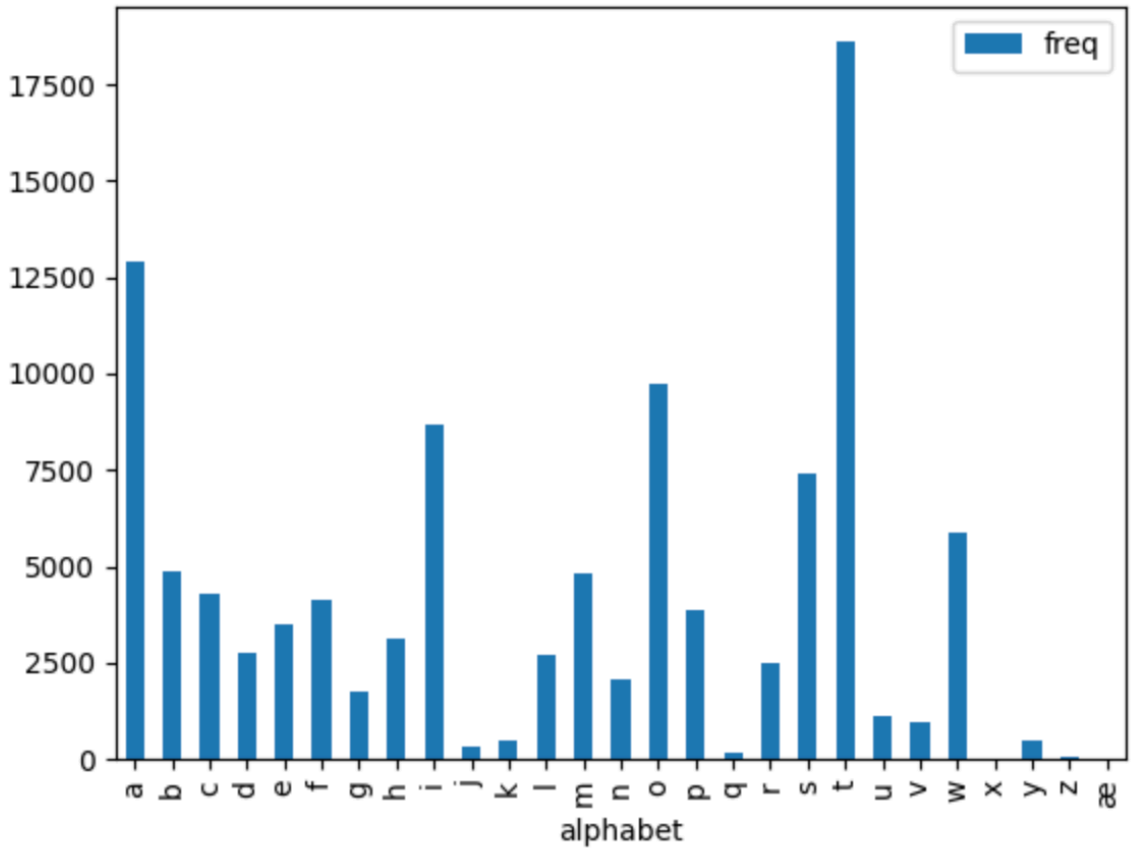
   a. The DataNode contains the actual HDFS data blocks and communicates with the NameNode as frequently as possible when there are any sort of updates, NameNode contains the metadata (information about the location, size of files/blocks), and the Secondary NameNode comes in handy when the NameNode fails as a housekeeper.

Task 1.3: Word count in pseudo-distributed mode

1. Explain the roles of the different classes in the file WordCount.java.

a. The WordCount class catches all the words that's flanked by a space and the IntSumReducer counts the occurrences of a word.

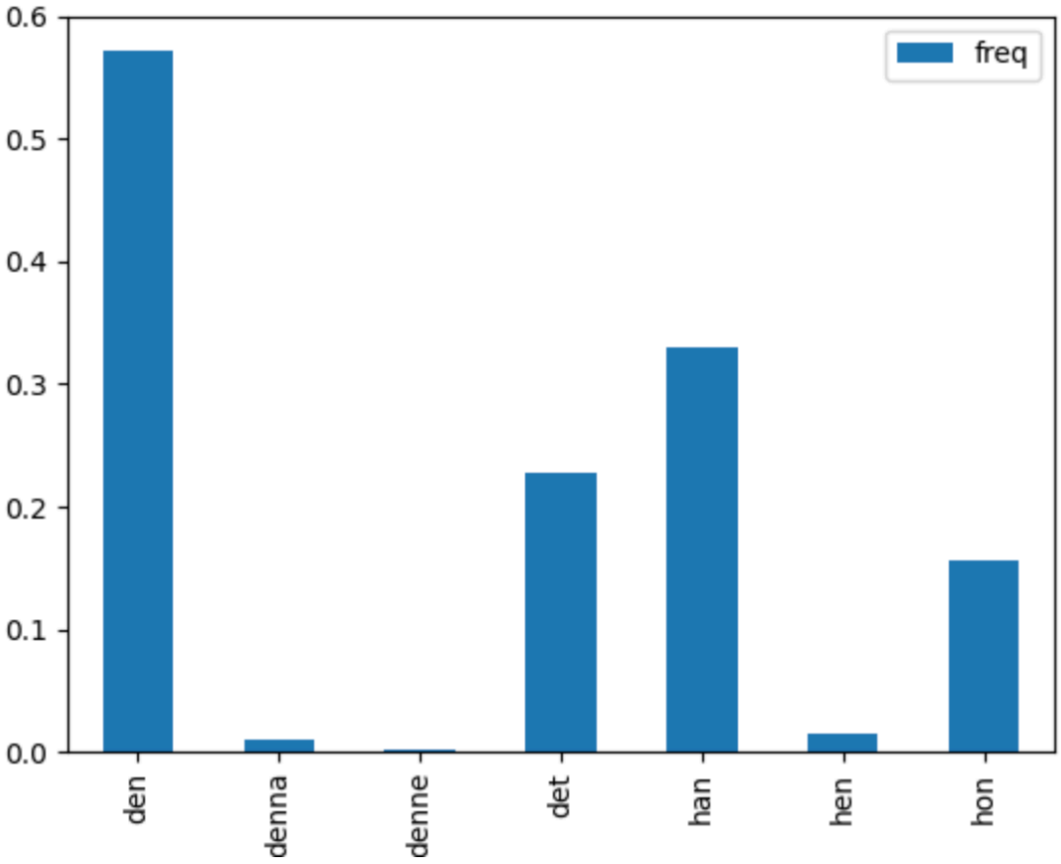
2. What is HDFS, and how is it different from the local filesystem on your virtual machine?
    a. HDFS, Hadoop Distributed File System, is a distributed file system designed to handle large streams of data working on a master/slave architecture. Its master is the NameNode and the DataNodes are the slaves providing the commodities. In the scenario where NameNodes fail, the Secondary Name Node takes precedence and resume the job. This system works to reduce bottlenecks and provide security by using racks of blocks with duplicated datanodes. This is different from our local filesystem because in HDFS a node can have multiple local storage in which all can be synchronized to work efficiently in a scalable manner whereas in a local filesystem, files are physically mounted and works linearly.



Task 2.1

1. Based on the twitter documentation in the above link, how would you classify the JSON-formatted tweets - structured, semi-structured or unstructured data?
    a. Semi-structured because it's formatted like XML and has child/parent relationships.

2. What could be the challenges of using traditional row-based RDBMs to store and analyze this dataset (apart from the possibility of very large datasets)?
    a. A need for file format-based parser, cannot have indexing, difficult to maintain reliability and consistency, difficult to maintain concurrent access, and users are responsible for memory management.
3.



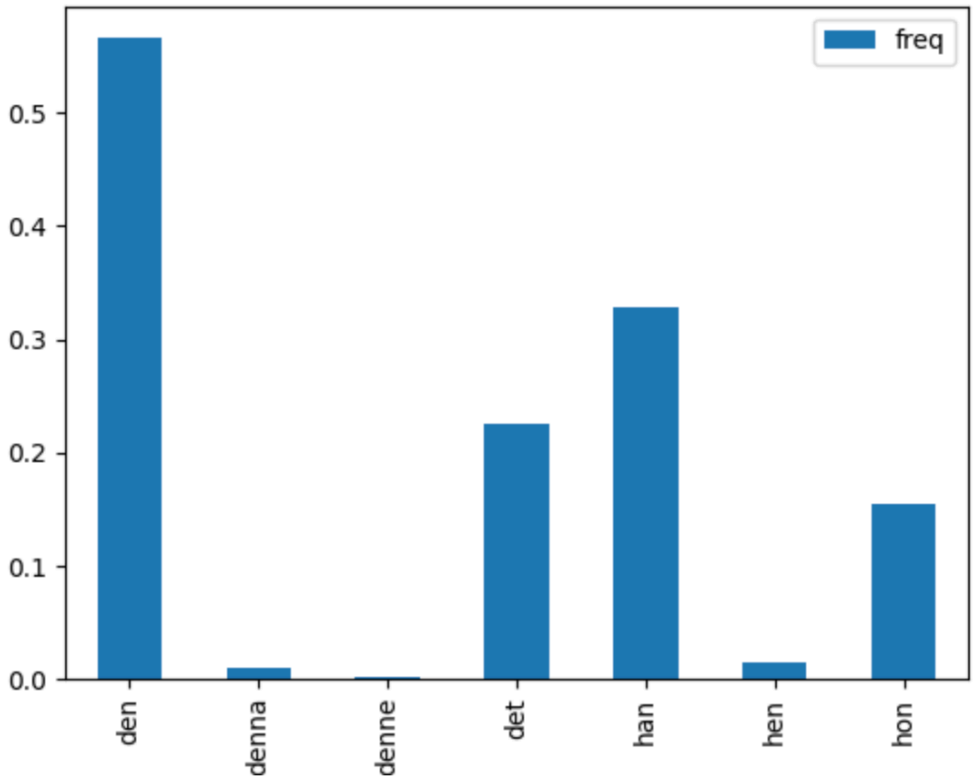
Part 2.

1. Answer the following question (expected answer length: 0.5 A4 pages):
   State the pros and cons for SQL and NoSQL solutions, respectively. Give some examples of particular data sets/scenarios that might be suitable for these types of databases.
    a. SQL provides a powerful way to manipulate data because it's structured and organized in a way that even the most complex of queries can manipulate and fetch a desired output. Databases can either be relational or object-oriented and both representing a transaction-oriented structure. It follows the ACID compliance and increases the integrity of the data from being disrupted or mutated due to external or unrelated failures. Additionally, SQL databases are very stable and secure as it can be designed to defend against threats and interception when populating new data fields. However, this is at the cost of pre-defined structure for data inserted into the database and it is no trivial task

especially when dealing with foreseeable large streams of data. Expertise is required to design a functional and scalable architecture and therefore, this kind of database could be costly. If there is ever the need to make changes to the database architecture, it would be difficult to relate this to other objects or tables- possible but difficult. SQL databases are most suitable for users storing transactional data and knowing they won't change so much over the course of their use with the database.

NoSQL, on the other hand, does not require pre-defined structure and provides dynamic solutions to different kinds of data: column-oriented, document-oriented, graph-based, or key-value based. It is easy to make changes without introducing major design modifications and relatively simple to deploy. As a result of this flexibility and design simplicity, NoSQL is cost efficient. A major improvement of NoSQL from SQL is that it runs horizontally, meaning each independent object can be run on multiple servers. Despite this, the caveats with this data storing method is it's relatively new in comparison to the well-established world of relational databases. It lacks standardization thus making it difficult to query and it lacks ACID properties found in SQL databases which could potentially result in data corruption. NoSQL databases are suitable for texts, log files, click streams, blogs, tweets, and audio + video data.

2.



3. Motivate your chosen implementation and how you did it. What are some pros and cons of the MongoDB solution compared to the implementation you did in Task 2.1?

a. I chose to use PyMongo as I am more familiar with the syntax. A drawback with this is that it is built on the shell and therefore new features might not be well documented or exists in the python version. Despite this, I think it offers more flexibility to users coming in with a different programming language and don't know much Javascript. It's better than having to work on Hadoop's Java framework that isn't so friendly and flexible towards non-Java users.

   As such, when I used MongoDB to replicate the results I produced in Hadoop made much more sense and was a very fluid process. The steps I took to count the pronouns included, loading in pre-processed tweet data into the database, used native functions like MapReduce to query for each pronoun found in each unique tweet text and count them using regular expression. I had to use regular expression to ensure I caught all the queries in each tweet because some pronouns could be flanked by special characters. This could be found in db-mongo.py and db-access.py. In db-mongo.py, I loaded the tweets in the database and the second script contains the MapReduce method.

   To conclude my experience with both data management systems, MongoDB was the better choice in handling the small dataset I had. It was especially friendly towards my amateur experience with these systems. Thus, I think the learning curve with MongoDB is much smoother than Hadoop because of how it is used, its interface, and its comprehensible documentation. Since Mongo provided a tool for python users, it makes it that I don't have to work around its framework and potentially writing work-around code that might bottleneck the database in real-time. The native methods I used to manipulate the stored data produced was comparatively faster results than in Hadoop. I would use Mongo over Hadoop any day but if I was dealing with massively big data I might consider Hadoop because it was primarily designed to handle just that. As good as Mongo is, I did experience a con. It was that I could not stream in a document larger than 16MB and therefore I had to make each tweet its own document in a collection. All in all, implementing a solution in Mongo was much more pleasant than in Hadoop.