Processing the Data into Footfall

Data Toolkit

Big data and its analytics promises huge benefits in terms of value realisation, cost reduction, insights but it also introduces a numerous pitfalls [1]. With developments in information technology, mobile communications and internet of things, large assemblages of data are readily available leading to immense possibilities in research. But when we analyse these data at such scale, we also encounter a large amount of added complexity and cost. Hence it is important to be careful in choosing the methods and tools in dealing with big data where we should look to devise right methods and tools for the right problems. Moreover in several disciplines, such as statistics and geography etc., the existing methods and tools are already developed for dealing with large scale data. These methods along with improvements in hardware has made the processing big data in these disciplines possible without a major changes in workflow. In the current environment of constant change and growth of sources of data, we cannot afford to lose the opportunity to extract information from them while trying to create a perfect, future proof approach in dealing with them. We need to move fast with a pragmatic approach where we look at other disciplines and adopt best practices and solutions in them and develop consistent approach for our needs rather than reinventing the wheel.

[1] Amir Gandomi and Murtaza Haider. Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, 35 (2):137–144, 2015

In the previous chapters we looked at various methods we devised to collect and process data from Wi-Fi probe requests emitted by phones. Though we discussed the methods conceptually, we left out the rationale behind choosing the toolkit employed to implement those methods. In this section we elaborate the thought process and rationale behind these decisions. We start by discussing the concept of 'Big Data' in general and look at previous literature to understand its definition, nature and the challenges they pose. Then we look at the data-sets we collected through the pilot studies and the 'Smart Street Sensor' project and evaluate them in terms of the dimensions of the big data. We also discuss the challenges faced in dealing with our dataset in detail and try to understand the requirements for devising a toolkit for it. Finally we put together a toolkit to suit our datasets built from simple small UNIX tools. [2]

[2] "Write programs that do one thing and do it well. Write programs to work together. Write programs to handle text streams, because that is a universal interface.", Doug McIlroy on UNIX philosophy.

What is 'Big Data'?

With the proliferation of internet enabled personal devices, we have quickly moved from data sparse environment to a data rich one. We can even

confidently say that we are in an age of data deluge where the amount of data which are collected and stored are increasing exponentially in a very short period of time [3]. As we saw in the previous chapters collecting large amount of data is quick and easy. Technological advancements have enabled us to be able to think about utilising such large assemblages of data which would have been impossible even in the recent past. By providing unprecedented coverage, these large assemblages of data - 'Big data', provide us with insights which were not possible before. They often change our approach and methods employed in entire disciplines. For example, In computer science, fuelled by the explosion of collected user data, there is a paradigm shift in Artificial Intelligence with the use of data mining, machine learning and deep learning. It is only time before this approach pervades social sciences research as well. In addition to the above advantages, Big data because of their nature also introduce numerous challenges in their collection, storage, analysis and visualisation. This is not including the enormous additional overhead and complexity introduced when we try to employ big data methods and tools. If we are not careful, using big data tools and methods for solving 'normal' problem can be counter productive where the advantages realised don't justified the overheads introduced. Hence it is important to understand the 'Big data' nature of the datasets we are dealing with at a granular level and choose the tools and methods without any presumptions.

[3] Rob Kitchin. Big data, new epistemologies and paradigm shifts. *Big Data & Society*, 1(1): 2053951714528481, 2014

The first and foremost challenge we face while discussing big data is its definition. It is hard to clearly and objectively define 'Big data' as it can vary widely based on the discipline and perspective. What may be 'big' in one discipline may not be in another. The nature of data can also be evaluated in various dimensions and can exhibit different properties in those dimensions. 'Big data' is generally defined within the context of disciplines, as data which cannot be managed with traditional methods and tools in those disciplines and requires substantial change in the approach of the practitioners. This definition is too subjective and falls short of giving us more understanding of 'Big data'. One of the most subscribed definition is to define the scale of the data in the dimension of volume - size of the data, velocity - speed of the data and variety - the complexity of the data [4]. This has also been extended to include more dimensions such as, veracity - the reliability or truthfulness of the data, visualisation - the complexity in visual interpretation and presentation of the data, and others such as visibility validity, variability, volatility and value. There have also been other alternative dimensions proposed such as Cardinality, continuity and complexity [5]. However we can consider the core dimensions of data - volume, velocity, variety, veracity and visualisation for evaluating our datasets. Since not all data is 'Big' in all these dimensions, we need to evaluate the 'bigness' of the data in each dimension and consider the associated challenges and solutions.

[4] Doug Laney. 3d data management: Controlling data volume, velocity and variety. *META group research note*, 6(70):1, 2001

[5] Shan Suthaharan. Big data classification: Problems and challenges in network intrusion prediction with machine learning. *ACM SIGMETRICS Performance Evaluation Review*, 41(4): 70–73, 2014

The second set of challenges arise while we process the big data, its acquisition, storage, extraction, cleaning, annotation, integration, aggregation, modelling, analysis, visualisation and interpretation. Challenges in each one of these processing activity arises due to the data being big in one or more dimensions. The data being big in volume, velocity and variety poses

challenges in data acquisition, aggregation, cleaning and analysis [6]. These challenges make traditional methods impractical and introduce the need for distributed, crowdsourced collection of data, heavily parallelised computing and application of functional programming concepts. The unstructured nature of the big data also introduces notable biases which mandate careful consideration, proper calibration and weighting during analysis so that we can understand and remove any uncertainties arising from them. The data being big in veracity dimension poses significant challenges in its analysis and modelling. Since simple methods such as linear regression fails in such scenarios, we require complex methods such as support vector machines, neural networks and hidden Markov models which compensate the lack of structure with the volume of data. With such computationally intensive methods, heavily parallelised high performance computing techniques such as GPU processing become indispensable. We also face significant challenge in visualising such complex features and methods which not only supports critical decision making but also is indispensable in exploratory analysis. The volume and velocity of big data makes them hard to visually simplify and digest. They are especially complex to interpret in the time dimension unless presented in small parts. Geographic information systems do a good job in visualising complex geographic data but struggle to maintain legibility and meaning when dealing with the temporal dimension. The visualisations of big data need to be highly processed, simplified and interactive to present meaning to the viewer. They have to balance between functionality, aesthetics and performance. Finally, because of the variety, big data creates need for consistent, well engineered standards so that multiple approaches and tools can be employed in tandem.

[6] Songnian Li, Suzana Dragicevic, Francesc Antón Castro, Monika Sester, Stephan Winter, Arzu Coltekin, Christopher Pettit, Bin Jiang, James Haworth, Alfred Stein, et al. Geospatial big data handling theory and methods: A review and research challenges. *ISPRS journal of Photogrammetry and Remote Sensing*, 115: 119–133, 2016

Apart from these processing challenges, we also have management challenges associated with big data such as privacy and security, data governance and ownership, data and information sharing, and cost[7]. Since these big datasets are usually comprehensive, securing them and protecting the privacy of the users becomes a central consideration in any project dealing with them. In many cases, though the data collected itself may not contain personal information but at these scales, in conjunction with other datasets, it can be used to infer them. The overall approach, methods, tools should comply with relevant legislation such as GDPR as well as the research ethics of all the stakeholders. This is especially challenging since these large unstructured datasets exhibit ambiguity of their ownership as well which calls for a clear, transparent and secure way to share them with other stakeholders along with publications of results in a timely, accessible manner. The associated project management and tracking tools need to be capable of handling these data ownership and sharing concerns as well.

[7] HV Jagadish, Johannes Gehrke, Alexandros Labrinidis, Yannis Papakonstantinou, Jignesh M Patel, Raghu Ramakrishnan, and Cyrus Shahabi. Big data and its technical challenges. *Communications of the ACM*, 57(7):86–94, 2014

Finally, the biggest challenge we face with big data is the cost in terms of money, resources and time. Though most of the big data tools are developed openly and distributed freely there can be lot of incidental, non-direct costs associated with collecting, processing and managing data with them. For example, there are the operational costs collecting data at such scale, network costs moving them, server costs storing and processing them, cost of procuring and supporting specialised tools and the human resource cost in hiring and training people who are capable for dealing with them.

Though there are economies of scale at larger scales, the overall resources required to manage big data effectively can be several folds of what is needed for a traditional dataset. This makes it important to look at the data in our hands closely and carefully so that we can make informed decisions on how 'big' it is and choose the methods which are the most suited for such dataset.

How big are the Wi-Fi probe request datasets?

In this section we take a detailed look at the three sets of Wi-Fi probe requests collected as described in chapter on data collection using the 5Vs big data framework. Our aim is to understand the nature of the data in each dimension and thus evaluate the challenges we face in that specific dimension leading to a bespoke solution. We look at each set of data in each dimension and try to answer the following questions,

1. How can this dimension be measure objectively?
2. How big is the data in terms of the defined measurement?
3. How does it data compare with datasets in other disciplines?
4. How can we describe the size of the data?

We then combine these isolated evaluations to form a combined description of the datasets. This is then used as the basis for developing a list of requirements for designing the data processing and management toolkit.

Volume

Probe requests data, being dynamic and continuous, cannot be quantified as an absolute static number in terms of volume. Hence we use a long term measurement - yearly rate, for each location instead. On shorter datasets such as the pilot study, we estimate the yearly volume linearly from the available data. We standardise this measure as the amount of disk space needed to store the collected data when encoded in text form. It is important to note that this can be reduced many folds by using compression or binary formats but we chose text since it the de-facto standard for exchanging data.

Study	Maximum*	Minimum*	Average*	Total**
Pilot Study	134	3	54	48.3
Main Study	6.1	2.4	4.42	4.1
Smart Street Sensor	5.4	0.001	0.8	0.8

Table 1: Comparison of volume or size of the datasets of Wi-Fi probe requests.

** Measured/ Estimated for each location in gigabytes per year. ** Measured/ Estimated for 920 locations in terabytes per year*

We can see that there is a lot of variability in the volume of probe requests generated at a given location. This mostly depends on how many mobile devices are present around the location. We observe that when we collect most of the information present in the probe requests in a busy area such as Oxford street in the Pilot studies, we generate around 50 terabytes of data in a year. But in a more real world setting such as the Smart Street Sensor project where the sensors fail at times and the amount of data collected is optimised, the volume is around a 1 gigabyte. The total volume of data we deal with in the case of a national scale project with around 920

sensors running for around 4 years is around 2 terabytes. A comparison of this to datasets from other disciplines is shown in Figure 1. It is key to note that the y-axis is scaled exponentially.

We can see that the probe requests data is not truly 'Big data' as experienced in other fields. It is only when we reach a complete coverage, i.e, putting a sensor at each retail establishment in UK, our estimated data volume reaches around 250 petabytes which is comparable to scales experienced in other fields such as particle physics and world wide social networks. At the same time, the scale of probe request data is not small either. The volume of 2 terabytes is more than the memory available in any desktop systems and is more than any of them can process in a timely manner. Summarising from the above, we can confidently say that the probe request datasets are 'Medium Data' - especially the dataset collected by the smart street sensor project. Though it has potential to scale into a truly big dataset, for the purposes of this research we can classify it as 'Medium data' in the volume dimension.

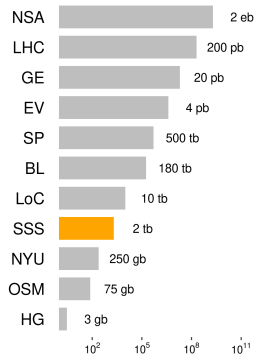

Figure 1: Comparison of volumes of datasets across various disciplines.

NSA - National Security Agency, LHC - Large Hadron Collider, GE - Google Earth, EV - Event Horizon project, SP - Spotify music, BL - British Library data store, LoC - Library of Congress, SSS - Smart Street Sensor, NYU - New york city Uber trips 2009-15, OSM - Open Street Map and HG - Human Genome Project

Velocity

Velocity is the rate at which the data is collected over time. It is significant when evaluating big data since some data which may not scale in terms of absolute volume but the speed at which they are received makes them challenging to deal with. A perfect example is the comparison between data generated by the Large Hadron Collider project by European Council for Nuclear Research and a world wide social network such as Facebook. Though their total volumes are comparable at 200 petabytes, the data from LHC is generated in concentrated experiments at a rate of 3 petabytes in 5 seconds while Facebook generates the same about in about a day or two. Since the size of an individual Wi-Fi probe request doesn't vary widely, we define the velocity of this dataset as the number of requests received at a given location at a given location within a given time interval. Though the precision of the time measured during data collection is in microseconds, the practical data transfer resolution in all the datasets is around 5 minutes. Hence we measure velocity of out datasets in terms of number of requests every 5 minutes. Table 2 compares the datasets we collected on Wi-Fi probe requests in terms of their volume.

Study	Maximum*	Minimum*	Average*	Total**
Pilot Study	8577	188	3469	3.20
Main Study	1362	534	782	0.72
Smart Street Sensor	5024	6	408	0.27

Table 2: Comparison of velocity or speed of the datasets of Wi-Fi probe requests.

** Measured/ Estimated for each location in number of requests per 5 minutes. ** Measured/ Estimated for 920 locations in Millions of requests per 5 minutes*

We observe that locations can receive up to 8500 requests in 5 minutes or can get no request at all depending on the time and how busy it is. But we can see that on average a national scale project with around 900 locations generates around a million requests every 15 mins. Compared to the LHC's 180 billion records or Google's 190 million searches per 5 minutes this seems to be not high speed data. However, this is much faster compared to traditional data sources such as census or geographical surveys which are updated anywhere between 6 months to 10 years.

To summarise, in terms of velocity, the Wi-Fi probes data can be described as 'Medium' at best. The methods dealing with the data should be time sensitive and be able to deal with a continuous stream of data but at the same time need not be real time or need sub-second latency. Since the Wi-Fi probe requests don't have actual location information the mobile devices, it does not have the similar value in real-time analytics as shown in comparable location or movement based datasets.

Variety

Variety is defined by the amount of variance in the type and characteristics of the data. Since variety is hard to quantify and compare across disciplines we evaluate the dataset subjectively for the variety present in it. The data transmitted in a Wi-Fi probe request is defined by the 802.11 Wi-Fi specification [8] and every probe request has to have a set of mandatory fields for Wi-Fi to work. This set of fields is the same everywhere across the world and the specification, especially the probe request part, has remained stable over years. Though there is some variability allowed within the specification, being part of a global standard, the data collected is heavily structured in general.

The first set of variety present in the Wi-Fi probes data set arises from the 'information elements' part of the probe request. The structure of a probe request is discussed in detail in the data collection chapter and is summarised in Figure ?. Essentially the information about the capabilities and type of the mobile device is encoded in the information elements part of the probe request and this information is optional and is implemented at the discretion of the manufacturers. As this information elements are demonstrated to be useful in successfully fingerprinting the mobile devices [9], mobile devices increasingly don't include any information in them. Emergence of manufacturers with large market share and narrow set of device models such as Apple and Samsung also reduce further variability in them. The second set of variety in the dataset arises from the rate at which these probe requests are generated by the mobile devices. Unlike devices which generate data on events or at regular intervals, mobile phones generate probe requests at a rate based on various factors. Though this leads to some challenges in counting footfall from these probe requests the variability exhibited here is neither so large nor so complex that traditional methods could not deal with them.

Comparing with some of the big data encountered in unstructured data collected over web such as social networks or other sensor based methods, the variability here can be considered trivial. Further when we convert these probe requests in to footfall counts, the variety in the dataset drops almost to zero as it becomes just an ordinal data point varying in geography and time. Summarising the above, we can confidently say that the Wi-Fi probe request data does not exhibit any 'big data' properties in the variety dimension.

Veracity

Veracity is defined as the amount of abnormality present in the data

[8] IEEE. IEEE standard for information technology-telecommunications and information exchange between systems local and metropolitan area networks-specific requirements - part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)*, pages 1–3534, Dec 2016. DOI: 10.1109/IEEESTD.2016.7786995

[9] Mathy Vanhoef, Célestin Matte, Mathieu Cunche, Leonardo S Cardoso, and Frank Piessens. Why mac address randomization is not enough: An analysis of Wi-Fi network discovery mechanisms. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, pages 413–424. ACM, 2016

in the form of inaccuracies, biases and noise. Similar to variety, veracity is hard to quantify hence required a subjective evaluation. Being sensor collected data, veracity is the dimension where the data exhibits most 'big data' properties.

First set of veracity in the dataset arise from the fact that it is collected through sensors located in multiple locations which communicate to the central server using 3G mobile data connectivity. We know from experience that the sensors are unreliable and fail to send back data regularly due to various reasons. More over the sensors are installed and uninstalled regularly as partners join and leave the project. This results in a data stream which is often erratic and incomplete with large gaps in them. In addition to this the sensors need to be rebooted regularly due to issues or updates leading to small gaps as well. Since the sensors are part of retail establishments they can be switched on and off regularly in some of them as well. Figure 2 demonstrates the veracity of the data in terms of missing data for a sample of locations in London. All the above pose immense challenges when we attempt to aggregate the data where we have to estimate and fill these gaps.

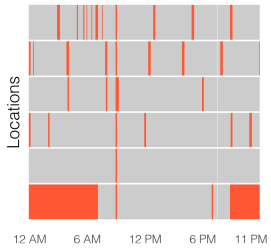

Figure 2: Missing data from five locations at Tottenham Court Road, London on 15 January 2018 demonstrating the veracity of the data.

There is also a lot of variability in the physical location of the sensors and the area of measurement. The sensors may report higher or lower count due to their configuration and the context of their location as discussed in chapters pertaining to data cleaning. This leads to a situation where the accuracy of the data collection varying quite widely across location and times [10]. It is often not clear if the change in the data is due to actual changes at the location or just the change in the configuration of the device. For example, Opening of a mobile shop next door to the sensor can increase the estimated footfall without any change in actual footfall at the location.

[10] Karlo Lugomer, Balamurugan Soundararaj, Roberto Murcio, James Cheshire, and Paul Longley. Understanding sources of measurement error in the wi-fi sensor data in the smart city. In *Proceedings of GISRUK 2017*. GIS Research UK (GISRUK), 2017

Finally we also have to work within the changing mobile landscape. Though the Wi-Fi probe requests are standardised by IEEE, the mobile manufacturers have started adopting obfuscation techniques to protect the privacy of the users. This started with randomisation of MAC addresses, removal of information elements and generally getting more sophisticated with new versions of operating system. There is also bias in terms of operating system adoption and change in market share between manufacturers. There is no inherent structure or information on what is changed and how often these changes occur which leads to questions on the continuity of the data over long periods of time.

Summarising from the above, we can confidently conclude that Wi-Fi probe requests dataset shows 'Big data' characteristics in terms of its veracity and requires appropriate tools and methods when aggregating, analysing and modelling it.

Visualisation

Visualisation is closely related to volume, velocity and variety of the data. The Wi-Fi data due to its non-trivial volume and velocity, exhibits similar characteristics and challenges in terms of visualisation. Since there is not much variety in the dataset, when we process the raw data into footfall counts we are left with just the time, location and footfall count for each

data point. Out of these, location and footfall counts are easy to visualise but time exhibits big data properties. This is primarily due to its granularity at 5 minute intervals and longitudinal nature of the data collection. The major challenge with Wi-Fi data is to simplify and visualise them in a legible way while showing change in term of time. The veracity of the data presents challenges in simplifying them and the volume poses challenges in maintaining legibility. We also have to take the 'near real time' aspect of the data into consideration while visualising them. There is a clear need for always on, interactive, real time dashboards with geographic capabilities in addition to the capabilities of traditional desktop GIS. There is also need for multiple linked dynamic visualisation platform for separating the scope of the visualisation into manageable units. Figure 3 demonstrates the illegibility of simple visualisations of the data due to granularity, variability and veracity. We can safely say that the Wi-Fi probe requests dataset is at best 'Medium' in the visualisation dimension.

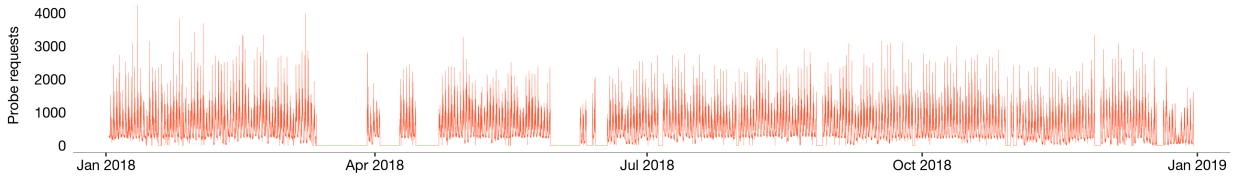

Figure 3: Number of probe requests collected for every five minute interval at Tottenham Court Road, London on the year 2018 showing the visual complexity of data in the time dimension.

Summarising the above discussion, we can conclude that the datasets collected from Wi-Fi probe requests are at best of 'medium'. They show the most big data characteristics in terms of their veracity. In rest of the dimensions the datasets are not truly big data and we need to look at tools and methods appropriate to their size. The toolkit we devise need to be able to deal with their mid-size volume, velocity and visualisation dimensions and at the same time need to able to deal with the large amount of veracity of in them. Figure 4 illustrates the summary our discussion. This leads us to devise a 'medium data toolkit' which can be used without incurring the extra cost and complexity introduced by big data tools while be able to handle the data at hand.

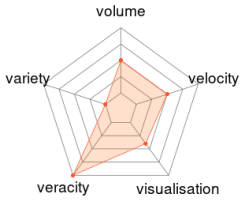

Figure 4: Big data characteristics of the Wi-Fi probe request datasets in their corresponding dimensions

A Survey of Methods and Tools

Having classified the Wi-Fi probes dataset as a 'Medium' sized data, in this section, we survey the tools and methods available at various stages of the data processing and management process - data collection, storage and retrieval, processing and analysis, visualisation. We first survey the tools available in each stage and specifically look at their suitability for Wi-Fi probe request datasets in terms of the following characteristics,

- *Performance* - How much data can be processed in a given time?
- *Flexibility* - How easy it is to change the scale and scope?
- *Complexity* - How many components or parts are involved?
- *Cost* - How much money or infrastructure do they require?

We then discuss the principles of UNIX philosophy and how it helps in solving similar sized problems in computer science. Finally we pick and connect the tools to devise our toolkit which is best suited for out Wi-Fi probe request dataset.

Collection

We discussed various technologies used in collecting passive data on ambient population and pedestrian movement in the literature search. In this section we look at tools and methods used to collect Wi-Fi based data passively. The primary considerations for evaluating data collection strategy are the scale of the infrastructure, expertise and effort required to implement it and cost involved. There have been numerous sensors, tools and associated software platforms made available for data collection under the umbrella of 'internet of things'. We start by looking at different approaches in the Wi-Fi data collection tools and try to reason the most appropriate solution for our research. On one end there are low level and low cost bespoke solutions which require lot of effort to implement and maintain. On the other end there are turn key solutions which doesn't require lesser effort but costs considerably more. The key is finding a balance between both while satisfying the requirements of the project. Since the Wi-Fi data is medium sized in terms of volume and velocity, we can deal with solutions with less than optimal scalability but since the data is 'big' in terms of veracity the toolkit has to give us most flexibility. Essentially, we are looking for a data collection methodology which prioritises flexibility and cost while performing moderately in terms of scalability and complexity as illustrated in Figure 5.

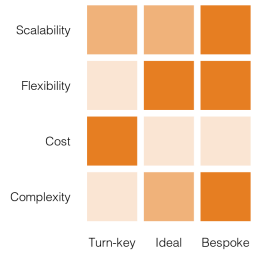

Figure 5: Characteristics of types of Wi-Fi data collection tools at each end of the spectrum compared to an ideal candidate

(Darker colors show higher score)

Type of solution	Examples
Bespoke	Micro-controllers with Wi-Fi modules e.g. Audrino + ESP8266
Turn-key	End to end commercial services e.g. Blix, Euclid, Pygmalios etc.
Ideal	**General purpose hardware e.g. Raspberry Pi, Repurposed mobile devices - Tablets, Phones etc.**

Table 3: Examples of different types of Wi-Fi based data collection solutions.

In terms of hardware, an example of a highly customised solution would be a micro-controller, such as Audrino, coupled with dedicated Wi-Fi module and programmed with custom software to collect the exact data needed. Designing and implementing of such system is time consuming, cumbersome and usually involves significant cost but it can also be highly flexible, efficient and cheap to deploy. On the other end of this spectrum, we have end-to-end solutions such as Blix, Walkbase, Ecuclid, Retail next, pygmalios etc. where the data is collected through multiple sensors and sources and syndicated into a clean footfall information by a third party service provider. These platforms for footfall data collection and analysis have the advantage of being quick and easy to develop and deploy while they can also be highly inflexible for changes and turn out to be costly when scaled up. A middle ground here is to use a general purpose hardware such as single board computers or repurposed mobile devices, augment them with addi-

tional hardware modules and use general purpose scripting languages to write software for them. This way we avoid low level hardware or software design and implementation while maintaining good amount of flexibility. Table 3 shows some examples of such systems while highlighting an ideal system.

The Smart Street Sensor project uses its own proprietary sensor system designed and instrumented by the data partner. The design and implementation decisions were made with the commercial application in mind and is not entirely relevant to our discussion in the context of our research. For the research conducted with the data, it is necessary to understand the data collection process and make sure it aligns and integrates with the rest of the toolkit. As discussed in the data collection chapter, the methodology used in the smart street sensor project satisfies our requirements. The toolkit we designed to collect other datasets are in-part inspired by this methodology or a modified version to include more flexibility. The toolkit consists of Raspberry Pi, Linux, tcpdump or tshark and nodejs. Raspberry Pi and the Linux OS it provides is the flexible and general purpose base system. On top of this we built our data collection system by assembling open source and free network analysis tools such as tcpdump and tshark along with other tools providing functions such as scheduling, personal data obfuscation and data transmission with scripting languages like nodejs and bash.

Storage

Data storage technology is one of the most diverse landscape in terms of both methods and tools available. It has been constantly in research and development since the beginning of computing and is one of the fastest changing landscapes with the advent of big data paradigm. A comprehensive review of storage solution warrants a chapter in itself so we restrict our survey to an outline of most significant approaches and corresponding systems and tools.

At one end of the spectrum is one of the most underappreciated for of data storage - File systems. Though they seem like a low level interface for storing data, file systems have their advantages as well. When the data is not complex or inter-related, flat text files in file systems could be the fastest way to store, search and retrieve data. Since operating systems are usually optimised to manage storage media through file systems, they involve no additional overhead and are extremely reliable. The hierarchical file systems use in most of the operating systems act as an index with hierarchical data. The major disadvantage of file systems is that they are not useful for managing data with any kind of complexity. This is the primary reason why database management systems are developed on top of file systems.

Database systems can be broadly divided into relational and document based. The relational databases are optimised to deal with relational data and usually enforce strict structure for the data In general they can handle large number of rows and are designed to scale vertically. Most relational database systems try to guarantee ACID [11] compliance and hence used in critical systems such as financial operations, sales etc [12]. The document

[11] Atomicity, Consistency, Isolation and Durability are properties which make sure that the data in the database is valid even during failures.

[12] Theo Haerder and Andreas Reuter. Principles of transaction-oriented database recovery. *ACM Comput. Surv.*, 15(4):287–317, December 1983. ISSN 0360-0300. DOI: 10.1145/289.291. URL http://doi.acm.org/10.1145/289.291

based databases are optimised to deal with unstructured data and can doesn't need a strictly defined scheme. In general they can handle large number of columns and are designed to be distributed and scaled horizontally. Being distributed, most document based databases try to pick a focus and compromise on others as specified in CAP theorem [13]. There a numerous databases systems which prioritise different things and the right solution depends on the properties of the data and the requirements of the project.

[13] Brewer's theorem or CAP theorem states that it is impossible to simultaneously guarantee consistency, availability and partition tolerance in a distributed data store.

Since the publication of the paper on 'Google file system' by google [14]. There have been significant effort in designing and building 'big data' file storage systems which can large data in the range of petabytes. These systems are designed to be distributed and optimised for high throughput for queries on them. Hadoop Distributed File System (HDFS) is one such file system which is also the most widely adopted. There are numerous cloud based, third-party solutions built with these file systems making them easy to use. There are also numerous tools, libraries and frameworks which emulate the features of database systems on these distributed file systems making them easier to use further. The primary advantage of these systems is the sheer scalability they provide when it comes to data volume. The primary disadvantage is the associated overheads in terms of cost and time incurred in learning, designing and implementing them. Unless the project is sufficiently large, the advantages gained usually do not justify the overheads introduced. Table 4 summarises the above discussion along with relevant examples.

[14] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The google file system. In *Proceedings of the 19th ACM Symposium on Operating Systems Principles*, pages 20–43, Bolton Landing, NY, 2003

Approach	Data size	Examples	Comments
File system	< 10 TB	ext, ntfs, zfs, btrfs	Simple and efficient. Best for hierarchical data. Cannot handle complex connected data.
Relational DB	< 5 TB	MySQL, PostgreSQL	Handles structured and relational data. Optimised for large amount of rows and tries to guarantee validity.
Document DB	<10 TB	MongoDB, Cassandra	Handles unstructured data. Optimised for large number of fields and distribution to multiple clusters. Tries to focus on any two guarantees of the Brewer's theorem.
Distributed FS	> 10 TB	HDFS, Ceph, GFS	Optimised for really large datasets which need to be distributed over multiple nodes.
Cloud Storage	> 10 TB	AWS, Azure, SWIFT	Implements distributed file systems on the cloud. Has more reliability and scalability than local implementations.
Data Warehouse	> 10 TB	Hive, Hbase, Impala, Presto	Interfaces built on top of distributed file systems to emulate capabilities of relational databases on them.

Table 4: Various data storage approaches and their characteristics.

We saw that the Wi-Fi probe request datasets are 'Medium' sized hence we can safely eliminate distributed file systems for storing them. Though

the smart street Sensor project uses Azure Blob Storage, when the data is downloaded to the local servers at the university, we can just store them in the file system because of their size (2TB approx.) and the hierarchical nature. The folder structure of - year/month/day/location/interval/ with individual text file, enable us to query the data for any given location at any interval nearly instantaneously without any further database operations. When this raw data is processed into 5 minute counts, we require more relational queries. For this purpose a relational database is sufficient as volume is quite small (25GB approx.). We chose PostgreSQL because of the PostGIS extension which gives us flexibility in handling geographic data.

Processing

We saw that the data is medium in terms of volume and velocity and shows big data properties in terms of veracity. Hence we require tools which are capable of dealing with the veracity of the data while being able to manage the volume and velocity. The traditional approach to deal with such dataset is to load it into a general purpose analysis tool such as R or a GIS packages and process it. The size of the dataset and the lack of meaningful complexity of geography element in the data eliminates the use of GIS packages. Scripting languages such as R and Python can deal with the dataset and its requirements but the time taken to do so increases exponentially with the size of the data as the size of objects in memory increases. Figure 6 illustrates the increase in processing time with respect to number of location for a simple exercise where a day's worth of raw data is parsed and aggregated into number of probe requests per 5 minute intervals (The code used to produce these benchmarks are detailed in the appendix at page 45). This becomes prohibitively expensive as the number of locations and complexity of the processing increases. Though this can be improved with more efficient coding practices, the margin of improvement is quite limited hence creating the need for better techniques. It is important to note that data processing is done in two stages - the first stage where the raw Wi-Fi probe requests are filtered, cleaned and aggregated into footfall counts and second stage where the footfall counts are in turn analysed to produce reports and visualisations. The traditional methods are sufficient for the second stage of the processing and the first stage is the one which requires a better solution.

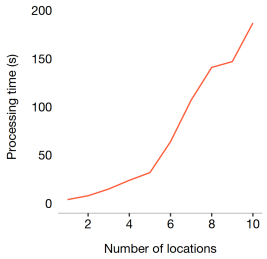

Figure 6: Exponential increase in the processing time when using traditional methods.

The processing involves parsing JSON data received for a single day at each location and aggregating them as number of probes requests received in every five minute intervals.

On the other end we have big data analysis tools which are built for dealing with extremely large amount of data. Since the publication of the paper on MapReduce, there have been immense developments in the Big data analysis landscape. There numerous distributed programming tools to use the data stored within a distributed storage system each focussing on specific type of data and analysis. A concise, non-comprehensive list of types of data or specialities and corresponding big data tools is shown in Table 5.

We can rule out the necessity of the above big data tools since our dataset is neither big enough nor fast enough. The dataset does not have any specialised structure such as graph or network but just a very minimal component of geography to it. Using any of specialised big data tools is

Tools	Speciality
General purpose	MapReduce, Spark
Real-time streams	Flink, Pulsar
Events or messages data	Storm, Kafka, Flume
Networked or graph data	Tinkerpop, Corona
Scheduling	Oozie, Falcon, Azkaban
Turn-key platforms	SpringXD, Cask Data

Table 5: Various types of big data processing tools and corresponding examples.

just going to introduce immense overheads without any added benefits. We need something in-between the above two approaches where we is sufficiently fast and flexible for our datasets.

This is where we come across the possibility of using standard Unix tools along with connecting them to create a processing pipeline. In some cases, a data processing pipeline made using command line Unix tools have been demonstrated to be 230 times faster than using big data toolkits [15]. The command line tools were developed as parts of Unix operating system for processing text. They are developed in line with the Unix philosophy which focuses on modular and minimal software development. The core tenants of the Unix philosophy has been summarised by Doug McIlroy as below,[16],

1. Make each program do one thing well. To do a new job, build afresh rather than complicate old programs by adding new "features".
2. Expect the output of every program to become the input to another, as yet unknown, program. Don't clutter output with extraneous information. Avoid stringently columnar or binary input formats. Don't insist on interactive input.
3. Design and build software, even operating systems, to be tried early, ideally within weeks. Don't hesitate to throw away the clumsy parts and rebuild them.
4. Use tools in preference to unskilled help to lighten a programming task, even if you have to detour to build the tools and expect to throw some of them out after you've finished using them.

These principles along with the 'pipe' operator gives us necessary tools to build more complex tools. We can replace most of the libraries we used in the R implementation of our processing with a corresponding command line tools and connect them together with a text interface to achieve similar pipeline. The first advantage of such design is that it is much more efficient than a monolith design. These tools being actively developed for since their invention are compiled as native binaries and are usually extremely optimised resulting in a much faster pipeline. Because of the design of the pipe operator, the individual parts of the pipeline are executed in parallel as chunks of data are passed through them thus avoiding the need to load entire datasets into memory which results in an exponential increase processing time with the size of the data. Being modular, we can even introduce process level parallelism to parts of the pipeline without any major change in the overall design. Finally the modular structure also gives us the advantage of using the best tool for any part of the pipeline. All of the gives us an extremely minimal and efficient toolkit to process the raw Wi-Fi probes data into counts in a scalable way. Figure 7 compares

[15] Command-line tools can be 235x faster than your hadoop cluster, Jan 2014. URL https://bit.ly/2s2XZYI

[16] Malcolm D McIlroy, Elliot N Pinson, and Berkley A Tague. Unix time-sharing system: Foreword. *Bell System Technical Journal*, 57(6):1899–1904, 1978

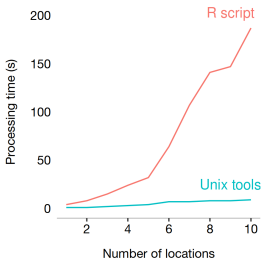

Figure 7: The increase in processing time with the Unix pipeline is linear thus improves the scalability compared to R based processing

the processing times of such Unix toolkit with the traditional R based toolkit as the data size increases. We can see that Unix toolkit performs extremely well and the performance gains are significant as the size of the data increases. For example, to process data for 25 locations, R based toolkit takes around 20 minutes while the Unix toolkit gets it done in 20 seconds (The code used to produce these benchmarks are detailed in the appendix at page 45). Table 6 shows the activities in our pipeline and corresponding libraries in the traditional R workflow along with the equivalent Unix tools. It is important to note that tools for doing specialised actions such as statistical analysis, machine learning and time-series analysis are built on top of scripting languages such as R and Python. These can be embedded into our Unix pipeline as scripts running in corresponding front-ends such as Rscript or python.

Tools	R Library	Unix tool(s)
Move data to and from Azure blog storage, SQL server and Postgres	AzureR, odbc , RPostgreSQL	azcopy, mssql, psql
Convert data from JSON format to CSV	jsonlite	jq
Encrypt raw data for secure storage	Rcrypt	gnupg
Anonymise personal data into cryptographic hash	digest	openssl
Transform and manipulate tabular data	dplyr	find, cat, cut, grep, sed, awk, sort, uniq, column, paste, join
Impute missing value using time series analysis	imputeTS	Rscript
Visualise the results into maps and charts	ggplot2	Rscript
Create and manipulate geographic data	sf, rgdal	postgis, gdal

Table 6: Tasks in the processing pipeline, corresponding R libraries and equivalent Unix tools

This toolkit can be further accelerated by parallelising parts of the pipeline using gnu-parallel [17]. For example, the previous example pipeline can be parallelised by spawning a pipeline for each location this reduces the processing time for a set of 25 locations from 18 seconds to 3 seconds. This done by utilising every processor cores available in the CPU. Figure 8 compares the processing times of the Unix toolkit with a parallelised implementation (The code used to produce these benchmarks are detailed in the appendix at page 45). Finally all the Unix tools discussed in this toolkit are open source and free software which has almost no cost in terms of resources. Since these tools are part of the POSIX specification [18] for operating systems, the expertise in their design and use are transferable to and from other disciplines thus reducing researcher time learning and using these tools.

[17] Ole Tange. *GNU Parallel 2018*. Ole Tange, March 2018. ISBN 9781387509881. DOI: 10.5281/zenodo.1146014. URL https://doi.org/10.5281/zenodo.1146014

[18] Stephen R. Walli. The posix family of standards. *StandardView*, 3(1):11–17, March 1995. ISSN 1067-9936. DOI: 10.1145/210308.210315. URL http://doi.acm.org/10.1145/210308.210315

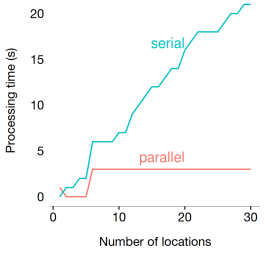

Figure 8: The scalability of the processing pipeline could be further improved with parallelising it.

Visualisation

In the last section we saw that the visualisation dimension of the data shows some level of complexity. The primary source of this complexity

arises from the longitudinal nature of the data and the noise due to granularity of the data. For the processed dataset, traditional visualisation and mapping libraries with R is sufficient while the visualisation of raw data across long time periods for either for exploratory analysis or for communication needs some form of interactivity or simplification to be able to legible. Data driven documents (D3) [19]and Dimensional charting (DC) provides us with both of these requirements. Both of these tools can accept text based input and can fit with other Unix tools discussed earlier. In case of binary file output such as images or documents, they could be directed to the file system and then read into other programs.

[19] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D3: Data-driven documents. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 2011. URL http://vis.stanford.edu/papers/d3

Conclusions

In this chapter we saw how the advent of internet and internet enabled devices has lead to significant increase in the amount of data generated and collected across disciplines. This data deluge and improvements in the capabilities of computing hardware has fuelled an explosion of research and development in tools and methods to deal with these 'Big data'. Though these big data tools promise huge improvements in processing capabilities, when used under wrong circumstances they can lead to unwanted overheads and costs. Thus we need a framework to examine and understand the scale of the data that is being used so that we use the right tools for the right purposes and the 5Vs of 'Big data' - Volume, Velocity, Veracity, Variety and Visualisation provides us with such frame work. Every dimension of big data poses unique set of challenges and we need make right decisions in choosing specialised tools and methods to overcome them.

We then closely examined the Wi-Fi probes data we collected with this framework and found that the data, though posed significant challenges with traditional data processing techniques, do not exhibit 'big data' properties in all its dimensions. Only veracity of the data was found to have any meaningful big data properties, while volume and velocity was found to be 'medium' at best. The datasets lacked any variety and posed minimal challenge in the visualisation dimension because of it high temporal granularity. Thus we arrived at the requirements for a bespoke 'medium data toolkit' which is able to deal with these challenges.

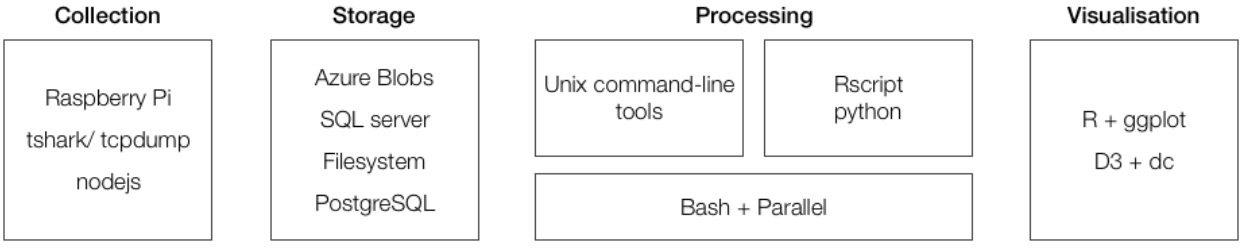

Figure 9: Outline of the 'Medium data toolkit' devised to collect, process, visualise and manage the Wi-Fi probe requests data

We undertook a brief survey of tools available for collecting, storing, processing and visualising the Wi-Fi probe request data and with the understanding of the data from the previous analysis chose the ones which are perfect for the datasets. For collecting Wi-Fi probes data in a scalable way, we chose a general purpose single board computers such as Raspberry-Pi

along with open source tools such as tcpdump and tshark in a Linux environment. For data storage we narrowed in on using just the file system for the raw data and relational database management system for the processed counts. To process the raw data we chose to devise a processing pipeline using an assortment of standard Unix command line tools linked together using a shell scripting language and parallelised at the process level with gnu-parallel. We also demonstrated that this processing pipeline can be 400 times faster than the using a monolithic pipeline even with a small sample of locations. For visualisation we chose D3 and DC as the solution for communicating time information legibly. Finally we arrive a 'medium data toolkit', illustrated in Figure 9, which is best suited for the Wi-Fi probes dataset which we employ to process and examine the data further.