International Journal of Research in Advanced Engineering and Technology
ISSN: 2455-0876; Impact Factor: RJIF 5.44
Received: 01-01-2020; Accepted: 20-01-2021; Published: 24-02-2021
www.newengineeringjournal.in
Volume 7; Issue 1; 2021; Page No. 10-16

Testing of map reduce application with black box approach using induction method

Ashish Kumar Rai, AK Malviya
Department of Computer Science and Engineering, Kamla Nehru Institute of Technology (KNIT), Sultanpur, Uttar Pradesh, India

Abstract
Big data is now no longer mere buzz word; industries are started realizing benefits with processing large volume of data known as big data. Earlier gold diamond mines are used represent prosperity but in modern word data resembles new ore for currency. Many technologies have evolved around processing heterogenous data to fetch business value. Hadoop Ecosystem and their components are playing vital role to provide accurate business insights, but this would not be possible without having proper and effective software testing for developed application. Testing big data application is certainly different from traditional approach specially applications which involved map reduce approach to split and combine. Current industrial practice mainly focuses on unit testing and component level tools-based automation for integration and regression. Functional testing with large volume data is still considered as a major challenge and business team also face similar issue for user acceptance testing. In this thesis, we researched on functional testing technique which can utilized large data based on mathematical induction principle to validate functionality of developed application. The proposed testing technique is equally effective on heterogenous data structured as well as unstructured data. The true potential of map reduce or big data application would be only realized when testing activity can be accomplished with accuracy and acceptance is provided accordance to business.

Keywords: bigdata functional testing, map reduce testing, end to end testing, induction testing, acceptance testing in bigdata, data defects

Introduction

Information has always played important role for decision making, it will not be wrong if it should be considered as fuel of organization, society or any endeavor. We know the Greek story of Marathon where a soldier ran 25 miles to convey message or information. In modern world with ease of access internet and other network technologies now messages can be conveyed in few seconds. Many organization's operations are now dependent on the capabilities of pushing away, extracting and using information. Data acts as an atom for the information and analogous to building blocks of information related application and services. With advancement in technologies data is now omnipresent, all modern internet application like Twitter, Facebook, Google, LinkedIn, YouTube, Chat Bots and many more are contributing in the increase of data volume along with sensor's data and traditional transactional business data. This large volume of data which comprises of structured as well as unstructured data is termed as Big Data. Processing these data having characteristics of velocity, volume and variety is tough and cannot be done in traditional way like RDBMS (Relational Database Management System). So new tools, applications are created to work on Big Data. Like processing of large volume, testing of Big Data application has its own challenges.

According to Wikibon, there will be CAGR of 10.48% worldwide bigdata market revenue in software and services which is projected to increase from USD 42 billion to USD 103 billion [1]. Gartner mentioned in their report top 10 strategic technology trends for 2020 that there could 20 times more smart devices generating data every second [2]. Some of the example for bigdata can be transaction of stock exchange NYSE generates more than one terabyte daily, In Facebook every minute 510 thousand comments get posted along with 293 thousand statuses get updated which turns up as more than 500 terabyte data daily. Similarly twitter gets 456000 tweets per minute while 4146600 videos get watched every minute. Now all the user activities are used for many strategic decisions, so applications are capturing all users action including searches, clicks even mouse or pointer movements. In aviation industry, more than 10 terabyte data are generated with 30 minutes flight time and contributes data in several petabyte daily.

Successful implementation of bigdata application software is dependent on both software development and testing. Software testing is the process of finding error or defect in program or finding deviation (if any) in expected behaviour or result. The purpose of this exercise is to improve the quality of software and reduce related cost of defect fix if encountered in live environment. To test bigdata application individual testing required in each stage from extraction of data, loading data in HDFS, transformation and utilization of data as per business requirement and further representing report or dashboard. To meet envisioned purpose of business application it is equally desirable to perform functional and non-functional testing. MapReduce should be considered as layer of bigdata application where key business rules get implemented. Further MapReduce also represents solution classification where program works in parallel in distributed environment with divide and solve approach. This makes testing of MapReduce as key factor for successful of the bigdata implementation [3].

Due to complexity of data and its huge volume, velocity and variety testing bigdata has its own challenges. Testing includes both aspects functional testing as well as non-functional testing. There are many tools available to help in non-functional testing which covers volume, stress testing, Response and capacity measurement. On the other hand, functional testing has challenge to deal with huge volume along with variety and provide conformance for functionality as per the requirements.

The remaining paper is organized as follows: First describe some of the related work for testing bigdata or Map Reduce application. While section 3 illustrate proposed techniques of functional testing based on Induction method. Section 4 presents case study which showcase proposed technique is effective to identify defects or provide conformance. Next section provides Results and discussion for the experiment carried out in section 4. Further next section provides conclusion and future scope for this paper.

Background and Related Work

The term Bigdata is getting used since 1990 and computer scientist John R. Mashey helped propagating the use of this words through different lectures and his articles. It is usually referred as large volume of data which cannot be stored, managed or processed with traditional way of handling data within permissible elapsed time [4]. Initially bigdata has been characterized by 3 Vs velocity, volume and variety but later with industrial influences others Vs got added like veracity, value.

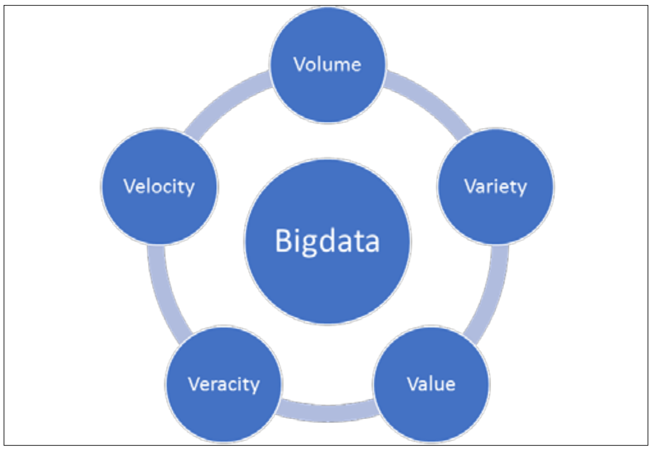


Fig 1: Bigdata Characteristics

With data in size of petabytes, exabytes for images, videos, audios, text managing, and processing has its own challenges, but the values provided by bigdata applications are winning bets with better ROI. It has been currently used in multiple domain like healthcare, government, retails, Scientific researches, Internet of things, Insurance and media.

In 2010, authors Pan, Tan, *et al* studied and presented paper on mapreduce testing to diagnose performance issue in Hadoop with an eye of black box testing approach. Authors name this approach "Ganesha" which analyze all underlying node and identify faulty node in the architecture assuming fault free node shows similar behaviour [5]. Authors carried out experiment in support of their approach with calculating true positive and false positive case on experimental data.

In 2011, author Kocakulak H. and Temizel T. T. presented paper describing use of Hadoop cluster for storage and processing large volume of multimedia data using map reduce for ballistic images with exploiting trait of fault tolerant feature provided by Hadoop framework with high computation and storage. This indicates the use of bigdata and map reduce in defense domain [6].

In 2011, the authors Chen, Ganapathi, *et al.* studied MapReduce and presented paper with their finding as performance evaluation for Map Reduce [7]. This paper suggested framework to synthesize map reduce job workload and setup benchmarking along with comparison of enough load two production trace. Authors also inclined one benchmark fits to all could not be a suitable way to evaluate performance for MapReduce jobs.

In same year 2011, authors Csallner, Fegaras and Li presented paper "New Ideas Track: Testing MapReduce-Style Programs" where they described on systematically searches for bugs in MapReduce program and generates test cases automatically [8]. They proposed the generation of test cases based on symbolic execution dynamically focusing mainly on maintenance of indexed tree and correctness condition for MapReduce program.

In 2013, researcher Camargo and Vergilio presented paper with their review of mapping study of different authors and papers for testing of map reduce style program and came with observation that majorly paper focused on performance, failover, availability, reliability and fault tolerance but very limited papers pay attention on actual map reduce program testing including functional testing [9]. They identified the gap as other than performance related, mostly papers touched only unit testing related technique or approach for map reduce program. This further leads to identify other area to be researched but convergence to techniques for map reduce program testing. Our work to identify testing technique for functional testing is an attempt in this area and motivated with authors conclusions.

In 2014, Morán, Riva, and Tuya in paper "MR Tree: Functional testing based on Map Reduce's execution behaviour", showcases the functional testing method for MapReduce program based on tree node navigations depth and breadth coverage to find out potential faults in Map Reduce program [10]. The approach was based on fault classification to find out functional defects in program developed using map reduce style and running on underlying infrastructure of Hadoop. The MR Tree technique was used for functional testing in example where average and maximum temperatures were captured for different years.

In year 2015 authors Morán, Riva, and Tuya in another paper "Testing data transformations in MapReduce programs" discussed approach to test MapReduce program based on data flow and proposed testing technique as MR Flow to analyze transformation in MapReduce program by depicting graph to cover different cases and to reveal defect [11]. For given Word Count program [19], authors presented MR Flow graph based on data flow.

In 2017, authors Morán, Riva, and Tuya paper presented another paper "Towards Ex Vivo testing of MapReduce applications" where authors suggested "Ex Vivo" context independent test approach to detect faults based live data and run on different environment [12]. Authors recommended to gather information with approach "*In Vivo*" and with using this information run program outside live environment as "Ex Vivo". Proposed approach also gave due weightage on data sampling and suggested to collect desirable sample as reservoir sampling technique using map reduce processing.

In year 2018, Authors Morán, Bertolino, *et al* worked on the detection of design fault in MapReduce where test data executed in parallel depends on test input data and test configurations. Authors proposed MRTest testing techniques and presented details in paper to automate detection of configuration and design fault [13]. They suggested techniques MRTest-Random and MR Test-t wise where first technique suggested to use random configuration for testing while second technique suggested to use partition and combination strategy for identifying configuration settings.

In same year 2019, authors Hsaini, Azzouzi, *et al*. presented another paper for testing distributed system similar like MapReduce with temporal approach. They described the intrinsic potential issues which could be arises when multiple testers are working parallelly, so they proposed to consider inter or intra port time constraints for calculating response time in map, shuffle or reduce phases [14].

As per book Concrete Mathematics in 1989, authors Ronald L. Graham, Donald E. Knuth, *et al*. narrated scientific acceptance of mathematical induction has already discussed in different articles and can be understood with example that we will climb as tall as we like on a stepping stool, by demonstrating that able to climb onto the foot rung (the premise) which from each rung we are able climb up to the following one (the step) [15]. This metaphor helps to utilize mathematical induction to solve by formal verification.

In 1996, author Hong Zhu in his paper discussed the way software testing resemble with inductive interference and tried formalized its interpretation through different Lemma and Theorem Induction The author showcased the properties of inductive inference for showing correctness of program and using this for software testing [16]. The research article also illustrated that software complexity must be considered for software testing even it is more vital for inductive interference.

In 2010, author Neil Walkinshaw, *et al*. presented their paper on inductive testing and uses of inductive testing to increase the functional coverage. They validated this approach on TCP stack behavior program and shown increased depth of test cases along with more acceptance and rejection transitions for inductive testing [17].

Proposed novel technique based on mathematical induction

From functional testing prospective testing specially for acceptance testing, considering the complexity of MapReduce program, it is hard to test and verify if program is running correctly and application is working as per business requirement. Most of the time functional testing is done as black box testing with minimal code structure knowledge. To support functional testing of applications based on MapReduce program, an approach can be adopted which is influenced by mathematical induction. It suggests that for given domain if it can be proved that application is working fine for base case, data set and incremental data set as expected, application or program is more likely correct and conform to business requirement.

Finding mathematical results based on mathematical principle to showcase its larger applicability: an assertion A (i) for natural number I can be proved if base or initial case A(1) is true and assuming it is also true for A(n) where n is any other natural number n but it can be proved true for next natural number n+1 implies that A(n+1) is also true. The proof of initial case A(1) is the first step while proof of A(n+1) is called the induction step and n is called the induction parameter. It is basis for inductive definition [18]. The proof can be represented as following steps:

- The base or initial case: proving statement holds for 0 or 1.
- The induction step: with assumption statement holds for n and proving statement holds for n+1.

So far mathematical induction is used to prove program correctness using formal method or logical inference. Other approach based on induction is inductive testing where based on given test data sets and corresponding program output functionality of given program is inferred[30]. But we recommend using the applied understanding of mathematical induction for functional testing MapReduce application in align with black box approach. Since acceptance testing is performed business user or mix of tester along with business user, this approach can be easily adopted due to its simplicity yet effectiveness.

Algorithm

```
Step 1.  Run Application for primitive value which is
         NULL
Step 2.  Validate that the application is giving correct
         output with NULL value
Step 3.  Run Application for primitive value which is Zero
Step 4.  Validate that the application is giving correct
         output with Zero value
Step 5.  Run Application for base value which is minimal
         data (or data set)
Step 6.  Validate that the application is giving correct
         output with minimal data set
Step 7.  Run the application for given data set X and record
         the output for further analysis
Step 8.  Add $\Delta X$ (delta) in given data set x
Step 9.  Run the application for $X + \Delta X$ data set
Step 10. Compare the output with step 7
Step 11. Validate if data is as per the acceptance criteria
Step 12. Output in Step 11 is as per the acceptance criteria
Step 13. Iterate the program from step 7 for other data sets
         (variety of data) and validate
Step 14. Validate output for other data sets to see
         correctness of the program
```

Depending on business requirement or logical inference base case can be identified which represent minimal data set on which program run. Step 1 and 3 validate program for NULL and Zero to provide a fair chance to check negative test condition if MapReduce program is built considering no input or blank data. Since we are doing acceptance testing, output for primitive cases for Zero or NULL along with base case can be validated based on business logic. For other input and output data business may have defined domain for input and corresponding range values for output. Step 7 recommends running application program for given test data set and record results considering it is inline as per business expectation. Now Step 8 suggests adding a known Δ (delta – small) value in input data set X and validate if output changes are corresponding input Δ changes in conjugation of output of step 7. Step 11 and 12 helps in validation of input and output matching with corresponding domain and range along with meeting business logic of application [3].

Since MapReduce program usually run on variety of volume data step 13 and 14 helps to iterate program for other variety of data.

Benefits of proposed testing technique

With existing approach, testing in big data applications are performed at various level such as Pre-Hadoop process validation, mapreduce process validation, ETL process validation and Reporting validation. The proposed technique is useful performing functional testing as system testing or E2E (End to End) testing which can be further prolonged to UAT (user acceptance testing) carried out by team involved actual business user. So, the proposed testing is useful in conjunction with other approaches to inspire customer confidence. The key benefits with proposed techniques can be listed as follows-

1. The proposed testing technique is simple yet effective to adopt easily for functional testing. It is recommended to use this technique along with existing before shipping developed code to business.
2. Since the proposed technique can be applied on system considering developed application as black box, it encapsulates complexity of architecture and different component and purely validate as per business requirement.
3. The proposed technique does not require any prerequisite of technical champion to start functional testing.
4. Test data provided by business or governance can be used as proposed approach is not restricted with size. It further analyzes the shift in outcomes with delta changes in input file.
5. The proposed technique can be adopted easily by business team for performing user acceptance testing as they can directly work on real world or live test data to validate outcomes matching with their expectation in induction steps.
6. Proposed technique exercised to run on variety of data with multiple iteration of program execution to extend coverages and expose defects related to data or configuration settings.
7. The environment requirement for testing would be similar to other approach either live like environment with lower scale or capability would work or need virtualizations to handle live like data as execution would be based on live like data.
8. The proposed technique is more generic in nature so can be utilized for testing other complex application other than mapreduce for example statistical analytics or business analytics where with addition of delta in input data file we can measure deviation in output of program.
9. It inspires more confidence of developer, tester and customer in product or services developed and rewards more profit with every successful satisfactory story. It also boosts more customer satisfaction and gain reputation as customer can easily understand steps and corelate with their business use case.

Case Study

In support of proposed technique, we carried out experiment on twitter data which is an example of structured (or semi structured) data. Major activity was to exhibit the proposed technique can be used, and able to show if given program is meeting business requirement conformance. This helps to gain confidence for the given program and providing consent for functional testing or acceptance testing.

This is an imaginary business case but closure to live issue. Multiplex wanted to screen Hindi movie 'Chhapaak' on multiple screens. Due to some trending news to boycott this movie, management decided to collect some inputs from internet and use guided decision for number of screens, frequency and timings to play this movie inline to their business benefit. Input can be taken from social media feedback. In this scenario, management collected a sample twitter data of 8th January 2020 while movie released was planned on 10th January 2020. In this experiment we have started our processing from collecting twitter data available on Kaggle website for public domain so extraction from twitter website using API can be skipped [19]. It was recommended to scrutinize the tweets of account with higher followers count as they would be source of high impact. Initial analysis can be proceeded with counting frequency of the words in these tweets.

A program written to count frequency of words in tweets. Block representation of the solution designed is given below. Our aim is to test the functionality of given solution which involved mapreduce processing on twitter data.

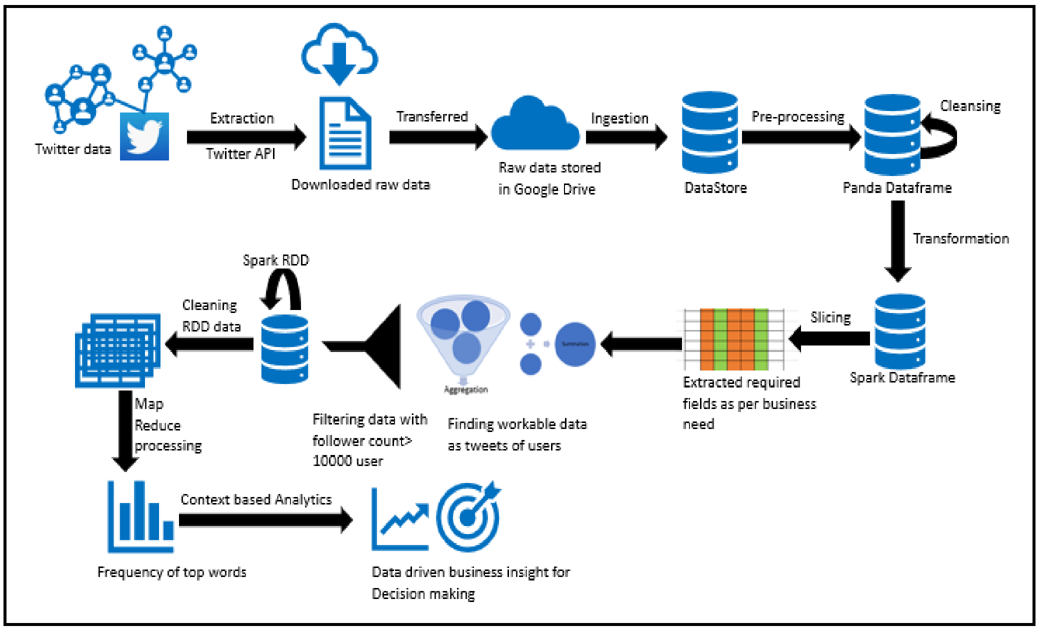


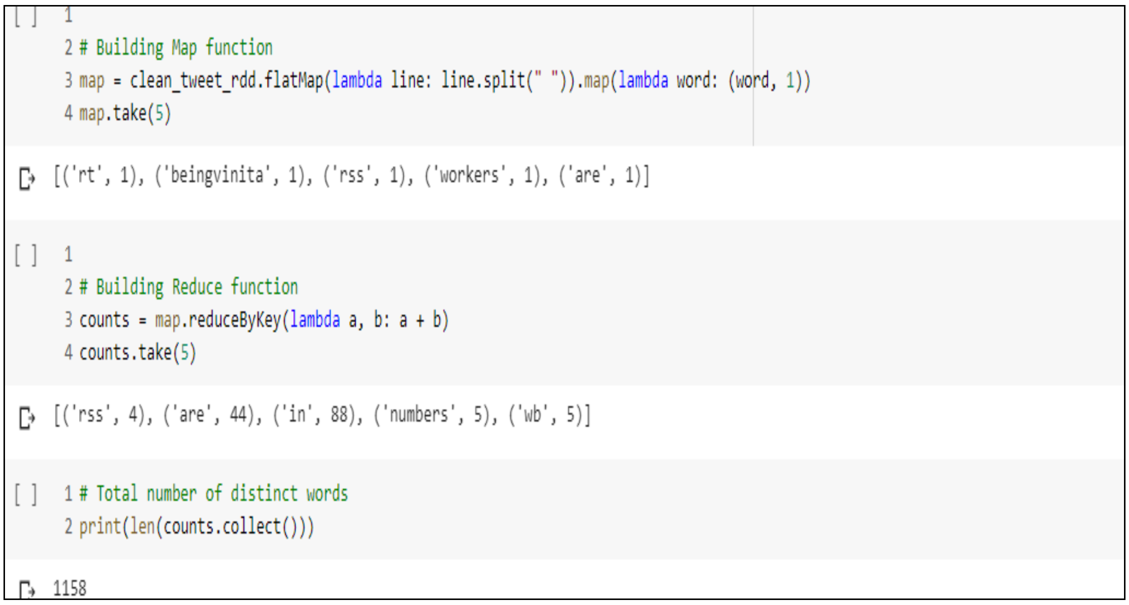
Fig 2: Block diagram to count frequency of word in given twitter data

This experiment uses different component and ran over publicly available distributed cloud environment Google Colab where program was written in python pyspark using Jupyter notebook. We consider the program implementation as black box and follow proposed testing technique. Our proposed algorithm suggested to execute following scenarios.

Table 1: Test Scenarios of Case Study

S.No.	Scenario	Description
TS#1	Run on primitive Case- NULL value	Run program without input file. This can be achieved by providing incorrect file name.
TS#2	Run on primitive Case- Zero value	Run program with input file containing no data. This can be achieved by providing file with only header information.
TS#3	Run for Base class - minimal data set	Run program with input file containing minimal data. This can be achieved by providing file with only header information and one data row.
TS#4	Run for Given Data Set X	Run program for given input file
TS#5	Run for X + ΔX data set	Change input file by adding or updating some delta text. Run program with changed input file.
TS#6	Repeat TS#4 and TS#5 with variation	Run test scenario TS#4 and TS#5 on given file X and updated file X + ΔX data set with variation (finding popular data set with followers_count greater than 5000)

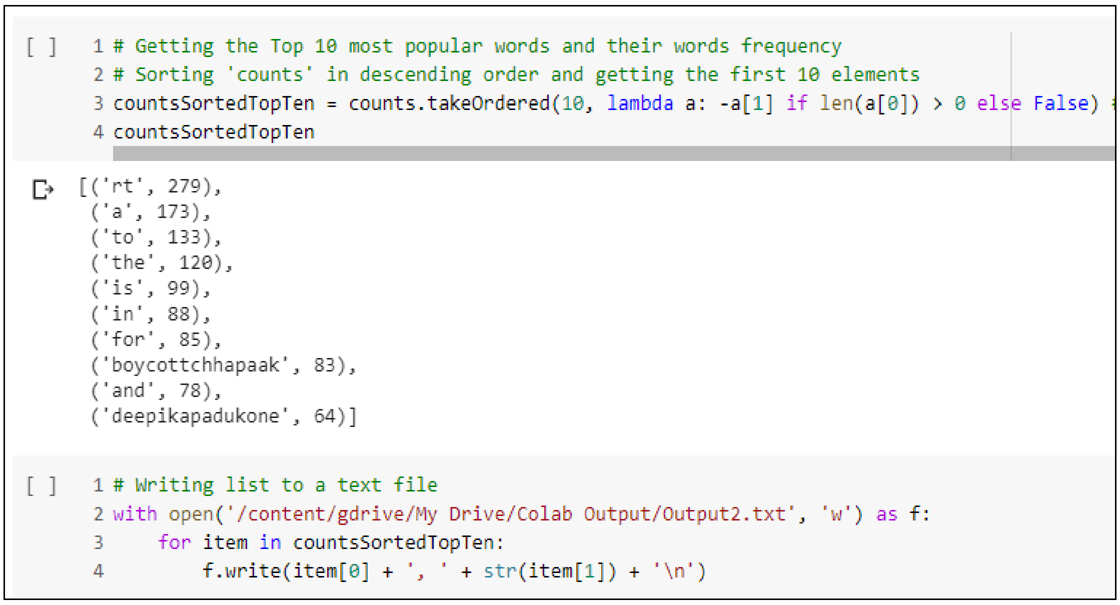
Following program snippet shows Map and Reduce functions are defined using RDD and Lamda



```
[ ]
    # Building Map function
    map = clean_tweet_rdd.flatMap(lambda line: line.split(" ")).map(lambda word: (word, 1))
    map.take(5)

[('rt', 1), ('beingvinita', 1), ('rss', 1), ('workers', 1), ('are', 1)]

[ ]
    # Building Reduce function
    counts = map.reduceByKey(lambda a, b: a + b)
    counts.take(5)

[('rss', 4), ('are', 44), ('in', 88), ('numbers', 5), ('wb', 5)]

[ ] # Total number of distinct words
    print(len(counts.collect()))

1158
```

Fig 3: Map and Reduce function

Fetching top 10 words with highest frequency and writing output in a file located in G-drive



```
[ ]  # Getting the Top 10 most popular words and their words frequency
     # Sorting 'counts' in descending order and getting the first 10 elements
     countsSortedTopTen = counts.takeOrdered(10, lambda a: -a[1] if len(a[0]) > 0 else False)
     countsSortedTopTen

[('rt', 279),
 ('a', 173),
 ('to', 133),
 ('the', 120),
 ('is', 99),
 ('in', 88),
 ('for', 85),
 ('boycottchhapaak', 83),
 ('and', 78),
 ('deepikapadukone', 64)]

[ ]  # Writing list to a text file
     with open('/content/gdrive/My Drive/Colab Output/Output2.txt', 'w') as f:
         for item in countsSortedTopTen:
             f.write(item[0] + ', ' + str(item[1]) + '\n')
```

Fig 4: Fetching frequency of words

On execution of program as per test scenarios, meaningful results are captured. These results are discussed in next section in detail.

Results and Discussion

We are going to analyse further on the applicability and effectiveness of our proposed technique on these example in context of finding answers to question related programs are meeting business requirements and as a testing team if we are in position of providing consent for functional testing. In this case study, program is executed for different test scenarios aligned with the proposed technique. Table-1 articulated details about these test scenarios for which program was executed with different input files as per proposed techniques.

The results were examined for following points

- Ability to count frequency of words for given twitter data.
- Ability to count frequency of words on variation of filter condition.
- Ability to fetch meaningful information from given twitter data.
- Ability of exposing defects in program with proposed technique.
- Ability to provide conformance with requirement while following proposed technique

TS#1, TS#2 representing primitive NULL and Zero test. With help of proposed technique corresponding error captured as File Not Found Error and Value Error empty dataset. These errors should be handled either with try-catch or some user-friendly message.

TS#3 test scenario resembles to base class of induction method showing program is working for unit level. We tested this file having one record, and for the given one record frequency is retuned 1 for word 'boycott chhapaak' and matching with expected count.

TS#4 test scenario considers running program for actual data captured from twitter. In our case it is chhapaak.csv data file. The output recorded and corresponding graph to show frequency of top 10 words for given file is chhapaak.csv. Here word 'rt' is not having any meaning so does not carry any significance for business analysis. It appears as acronym for retweeted tweets. While count for word 'boycott chhapaak' and 'deepika padukone' still holds our eyes to analyze further.

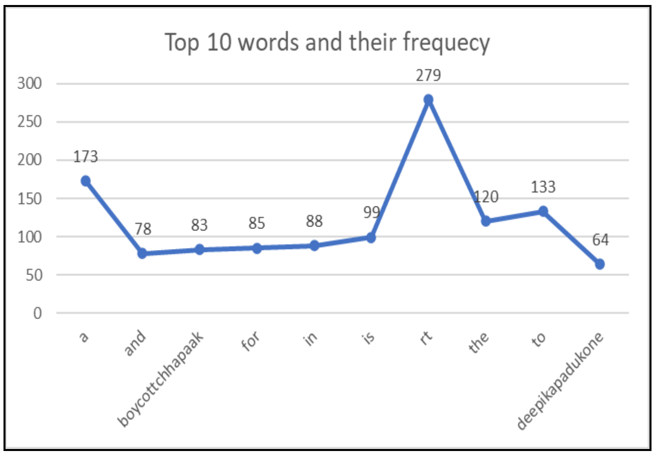
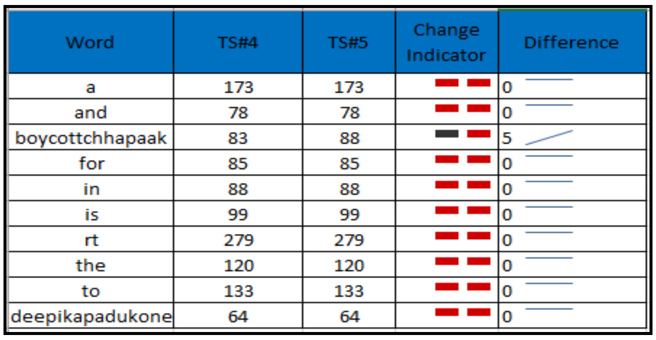


Fig 6: TS4 Top 10 words for given file

TS#5 scenario represents induction step where we add delta in given file and execute program for this updated file. In our case study, the given file was chhapaak.csv used in TS#4. So, we created a copy of this file and added delta text as 5 more words 'boycottchhapaak', the updated file was saved as chhapaakDelta.csv. On execution of the program with test data as chhapaakDelta.csv, Output was captured, and the frequencies of words is recorded as below

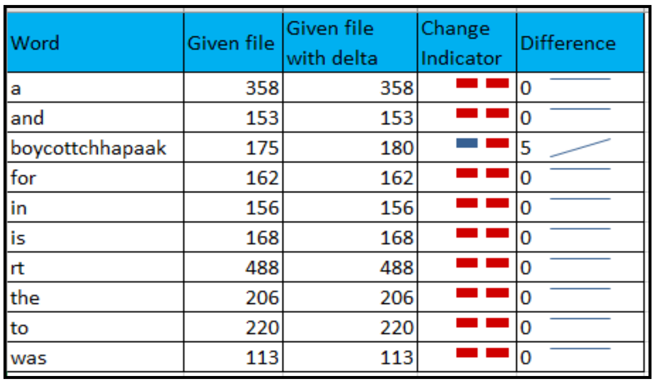


Word	TS#4	TS#5	Change Indicator	Difference
a	173	173	🟥🟥	0
and	78	78	🟥🟥	0
boycottchhapaak	83	88	⬛🟥	5
for	85	85	🟥🟥	0
in	88	88	🟥🟥	0
is	99	99	🟥🟥	0
rt	279	279	🟥🟥	0
the	120	120	🟥🟥	0
to	133	133	🟥🟥	0
deepikapadukone	64	64	🟥🟥	0

Fig 6: TS5 Change in frequency with delta file

Change indicator and difference values shown the change in frequency of words are as expected.

TS#6 scenario presents running same program with given input file and delta file with variation of filter condition as followers count was greater than 5000.The highest frequency relevant word still appears as word 'boycottchhapaak'.



Word	Given file	Given file with delta	Change Indicator	Difference
a	358	358	🟥🟥	0
and	153	153	🟥🟥	0
boycottchhapaak	175	180	🟦🟥	5
for	162	162	🟥🟥	0
in	156	156	🟥🟥	0
is	168	168	🟥🟥	0
rt	488	488	🟥🟥	0
the	206	206	🟥🟥	0
to	220	220	🟥🟥	0
was	113	113	🟥🟥	0

Fig 7: TS6 Change in frequency with delta file

The below figure displayed the comparison in word frequency when followers count was greater than 10000 or greater than 5000.

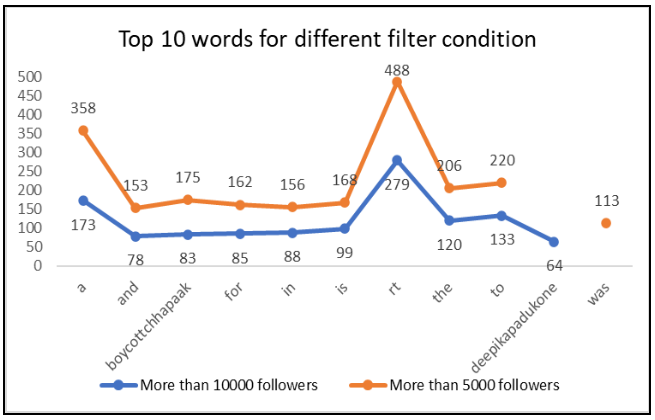


Fig 8: TS6 Comparison of highest frequency word in different configuration

We can ignore the “Stop words” or common English words like ‘a’, ‘to’, ‘the’, ‘is’, ‘in’, etc. word ‘rt’ is not having any meaning so does not carry any significance and represent retweet. While ‘deepikapadukone’ was not listed in top 10 with changed condition. The prominent words we noticed are ‘boycottchhapaak’ with frequency of 83. It clearly indicates that word ‘boycottchhapaak’ holds more contextual value. This experiment provides evidence of negative sentiments for the movie ‘Chhapaak’.

Conclusion

We have worked on novel functional testing technique based on mathematical principle. The presented technique is simple, and Bigdata including map reduce application can be tested easily using this technique. The case study on structured data was presented with experiment details to resolve business problem using map reduce framework, subsequently the functional testing was performed with proposed technique and was quite successful in getting defects related to boundary values or zero and null cases of proposed technique also reveals test for deviation of results with variety/volume of data. In this experiment we have validated against actual data file for the frequency of words and found word ‘boycottchhapaak’ was prominent and considerable contains due weightage for business problem with evident run for scenario where follower’s count greater than 10000 and 5000 and frequency returned as 83 and 175 respectively.

We also presented the benefits and utilizations of proposed testing technique and how proposed testing technique can be adopted in user acceptance testing of bigdata applications, for example testing at business level for mapreduce applications. Due to generic nature, it can also be applied for functional testing or business rule validation of other ETL or high extensive data usages application.

In future some more case studies can be carried out showing proposed testing technique also works on unstructured data. In term of enhancement, we recommend adopting automation framework which helps to create multiple scenarios based on different conditions primitive, base or induction steps.

References

1. Columbus L. “10 Charts That Will Change Your Perspective of Big Data’s Growth,” 2018. https://www.forbes.com/sites/louiscolumbus/2018/05/23/10-charts-that-will-change-your-perspective-of-big-datas-growth/#49784e8a2926.
2. Panetta K. “Gartner Top 10 Strategic Technology Trends for 2020,” Gartner, 2019. https://www.gartner.com/smarterwithgartner/gartner-top-10-strategic-technology-trends-for-2020/.
3. Rai AK, Malviya AK. "Testing MapReduce program using Induction Method," 2020 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS), Bhopal, India, 2020, 1-5. doi: 10.1109/SCEECS48394.2020.178
4. “Big Data,” Wikipedia, 2014. http://en.wikipedia.org/wiki/Big_data.
5. Pan X, Tan J, Kavulya S, Gandhi R, Narasimhan P. “Ganesha: Black-box diagnosis of map reduce systems,” Perform. Eval. Rev. 2010; 37(3):8-13. Doi: 10.1145/ 1710115.1710118.
6. Kocakulak H, Temizel TT. “A Hadoop solution for ballistic image analysis and recognition,” Proc. Int. Conf. High Perform. Comput. Simulation, HPCS, 2011, 836-842. Doi: 10.1109/HPCSim.2011.5999917.
7. Chen Y, Ganapathi A, Griffith R, Katz R. “The case for evaluating mapreduce performance using workload suites,” IEEE Int. Work. Model. Anal. Simul. Comput. Telecommun. Syst. - Proc, 2011, 390-399. Doi: 10. 1109/MASCOTS.2011.12.
8. Csallner C, Fegaras L, C Li. “New ideas track: Testing MapReduce-style programs,” SIGSOFT/FSE 2011 - Proc. 19th ACM SIGSOFT Symp. Found. Softw. Eng. 2011, 504-507. Doi: 10.1145/2025113.2025204.
9. Camargo LC, Vergilio SR. “Testing MapReduce Programs: A Mapping Study,” Proc. - Int. Conf. Chil. Comput. Sci. Soc. SCCC. 2013; 0:85-89. Doi: 10.1109/SCCC.2013.10.
10. Moran J, La Riva CD, Tuya J. “MR Tree: Functional testing based on map reduce’s execution behaviour,” Proc. - 2014 Int. Conf. Futur. Internet Things Cloud, Fi Cloud. 2014, 379-384. Doi: 10.1109/Fi Cloud. 2014. 67.
11. Morán J, De La Riva C, Tuya J. “Testing data transformations in MapReduce programs,” 6th Int. Work. Autom. Test Case Des. Sel. Eval. A-TEST. Proc, 2015, 20-25. 2015, doi: 10.1145/2804322.2804326.
12. Morán J, Bertolino A, De La Riva C, Tuya J. “Towards ex vivo testing of map reduce applications,” Proc. - 2017 IEEE Int. Conf. Softw. Qual. Reliab. Secur. QRS 2017, 73-80. 2017, doi: 10.1109/QRS.2017.17.
13. Moran J, Bertolino A, De La Riva C, Tuya J. “Automatic Testing of Design Faults in MapReduce Applications,” IEEE Trans. Reliab. 2018; 67(3):717-732. Doi: 10.1109/TR.2018.2802047.
14. Hsaini S, Azzouzi S, Charaf MEH. A Temporal Approach for Testing Distributed Systems-A MapReduce Case Study, 2019, 1085.
15. And O. P. Ronald L. Graham, Donald E. Knuth, Concrete Mathematics: A Foundation for Computer Science, 1989.
16. Zhu H. “A formal interpretation of software testing as inductive inference,” Softw. Test. Verif. Reliab. 1996; 6(1):3-31. Doi: 10.1002/(SICI)1099-1689(199603) 6:1<3::AID-STVR108>3.0.CO;2-D.
17. Walkinshaw N, Bogdanov K, Derrick J, Paris J. “Increasing functional coverage by inductive testing: A case study,” Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), LNCS. 2010; 6435:126-141. Doi: 10.1007/978-3-642-16573-3_10.
18. Hazewinkel and Michiel, “Mathematical induction,” Encyclopedia of Mathematics, Springer Science+Business Media B.V. / Kluwer Academic Publishers, 2001. https://www.encyclopediaofmath.org /index.php/Mathematical induction (accessed Apr. 12, 2020).
19. Souren, “Twitter Data - boycott-chhapaak,” 2020. https://www.kaggle.com/souren/boycottchhapaak.