Efficient Algorithms for Data Extraction in Big Data

Abiye-Suku, Ominini Monima & E. O. Bennett
Department of Computer Science,
Rivers State University,
Port Harcourt, Nigeria
Ominini.monima@ust.edu.ng, bennett.okoni@ust.edu.ng

Abstract
This paper is centered on the development of an information system for the extraction of e-mails addresses of staff/students in an organization with big data to assist in the dissemination of vital information in a very short time using two algorithms that are rule and machine learning based. The rule-based uses regular expression technique while the machine based is implemented with the decision tree classifier. The proposed tool can be used in the banking sector to extract customers email addresses for posting transaction details and goodwill messages to improve customers' relationship, in the educational sector the tool can be used to send students' progress report, registration and payment receipt and lastly in the health sector to send medical reports, globalization and monitoring the hospital quality. The database was generated online since most organizations are very discreet with staff details. Tokenization of the generated data takes place immediately where the domains are determined. The constructive research methodology and object-oriented design technique was used to analyze, design and implement the tool. A customized software application was developed in python programming language for its implementation ensuring the system evaluation met with system requirements and potential users' expectations. The research extensively carried out testing using different data sizes to execute email addresses extraction and showed the limitations of the tool and potential for further work on the software package.

Keywords: *Algorithm, Data Extraction, Rule Based, Machine Learning*

1. Introduction

One of the major advancements that have influenced Technology in the last fifteen years is the 'Data Balloon'. This Data Balloon has inflated gigantically giving rise to 'Big Data' that has also led to new data analysis, mining tools, and techniques to manage it (Big Data Analytics). Small data is data that is small enough for human comprehension; it is what we think of as data. Small Data is about 'People' Big Data is about machines. According to (Kumar et al 2014), Big Data is the gathering of informational collections so immense and complex that it ends up hard to process utilizing available database framework apparatuses or customary information handling applications. However, an important thing to note is the term 'Big' used in Big Data. Big data does not mean that Data is big or small in volume. 'Big' refers to the big decisions/value that it enables an organization to take time after time, which ultimately results in increased revenue, more customers and responsibility to nurture relationships. Big Data is one of the emerging technologies in Information Technology world. Information has turned into the most profitable asset on earth. In any case, it should be morally separated, refined, conveyed and adapted. Like the manner in which oil has driven development and delivered riches for amazing countries, the following rush of development will be driven by information. Big data is winding up significant in various regions of human undertakings today with numerous areas receiving operational procedures that line up with this pattern (Baesens, 2014). People around us do online banking, online bill payment, ticket booking etc. conveniently

because we are living in the world of internet. Large organizations and governments have embraced big data technology and they in fact constitute the biggest adopters (Kitchin, 2014). Big data has increased much consideration from the Academia and IT industry, over two million people will connect to the internet, and over five million own mobile phones, by 2020, 50 billion devices will connect to the internet (Khan, 2018). On a daily basis upload of data on social networking sites like Twitter, WhatsApp, Facebook, millions of people carry out Gmail etc. across the world that is now a pool of data. Data obtained from credit cards, phones, computers, televisions, sensor traffic, planes, GPS signals, bridges, factories, banks, hospitals, agriculture and other section of the economy are in terms of terabyte. This paper builds email address extraction tool using regular expression and machine learning. This tool will be used by organizations to extract email addresses of clients and staffs for information dissemination.

2. Related Literature

Data as a raw fact are the new oil of the digital age. This is because information (processed data) is used to power the transformative technology we use today- Artificial Intelligence, Predictive Analytic and Big Data Analytic. Data was less and easily managed by Relative Database Management System (RDBMS) twenty years back but recently (Vikram, et al 2013) demonstrated that it is not possible to handle massive data (big data) through RDBMS tools. In this regard, they concluded that big data differs from other data in five dimensions such as volume, velocity, variety, value and complexity. They illustrated the Hadoop architecture ecosystem to handle huge data collections; Hadoop Distributed File System (HDFS) to handle big data systems, scalable algorithm does log management application of big data. The authors concentrated on the limitations like data privacy, search analysis etc. that organizations encounter when managing big data. Big Data are streams of data traditional data management methods cannot handle, (Bernice, 2013). Big Data is different data sets making up the variety attributes. The data storage techniques used comprises multiple clustered Network Attached Storage (NAS) and object-based storage. Map Reduce of the Hadoop architecture helps to process unstructured and structured data by locating all useful data and selecting the exact data answering the query. Bernice concluded that the coming of Big Data has acted openings as well as difficulties to the business world. A study of big data definition was carried out by (Jonathan & Adam, 2013), without any scientific or statistical analysis the authors opined that big data is majorly associated with two ideas: data storage and data analysis. Despite the sudden interest in big data, these concepts are far from the new and have long lineages. This, therefore, raised the question as to how big data is notably different from conventional data processing techniques. “Big" implies significance, complexity and challenge. Unfortunately, the term “big" also invites quantification and therein lays the difficulty in furnishing a definition. The lack of a consistent definition introduces ambiguity and hampers discourse relating to big data. This paper attempts to collate the various definitions which have gained some degree of traction and to furnish a clear and concise definition of an otherwise ambiguous term. In their journal, “Using Regular Expressions to Abstract Blood Pressure and Treatment Intensification Information from the Text of Physician Notes” (Alexander, 2006) examined the utility of regular expressions to identify clinical data important to the epidemiology for the treatment of hypertension. They developed a framework that utilized regular expressions to mark and extract instances of documented blood pressure values and anti-hypertensive treatment intensification from the text of physician notes. They determined sensitivity, specificity and precision of identification of blood pressure values and anti-hypertensive treatment intensification using a gold standard of manual abstraction of 600 notes by two independent reviewers. The software processed 370 Mb of text per hour, and identified elevated blood pressure documented in free text physician notes with sensitivity and specificity of 98%, and precision of 93.2%. Anti-hypertensive treatment intensification was identified with sensitivity

83.8%, specificity of 95.0%, and precision of 85.9%. Regular expressions can be an effective method for focused information extraction tasks related to high-priority disease areas such as hypertension. In a paper (Albert, 2002) proposed taxonomy for characterizing web data extraction tools, they briefly survey major Web data extraction tools described in the literature, and provide a qualitative analysis of them. This work will stimulate other studies aimed at a more comprehensive analysis of data extraction approaches and tools for web data. The importance of this problem derives from the fact that, once extracted, the data can be handled in a way similar to instances of a traditional database. The approaches proposed in the literature to address the problem of web data extraction use techniques borrowed from areas such as natural language processing, languages and grammars, machine learning, information retrieval, databases, and ontology. Consequently, they present very distinct features and capabilities that make a direct comparison difficult. In their paper "Learning regular expression for clinical text classification' (Duy, et al 2014) developed a Regular Expression Discovery (RED) algorithm and brought about two text classifiers based on RED. The RED + ALIGN classifier combines RED with an alignment algorithm, and RED + SVM combines RED with a Support Vector Machine (SVM) classifier. They used two clinical datasets: SMOKE and PAIN datasets for testing and evaluation. The evaluations' results proved that the two classifiers used with RED patterns are slightly better than SVM in performance. Although generalization was one of their challenges, they concluded that using regular expressions to tap into the sequential relationships among salient words is a promising approach to improve text classification performance. In a journal "A Memory Efficient Regular Expression Matching by Compressing Deterministic Finite Automata" (Utkarsha, et al 2015) proposed that Deterministic Finite Automaton (DFA) is often used to represent regular expression. Since a good size of the memory is required to store the transition functions of the DFA they development a tool that could reduce the size of the DFA. This method has four major phases as Regular expression to Non-Deterministic Finite Automata (NFA) conversion, NFA to DFA conversion, DFA compression and matching on DFA. Since these four phases reduce DFA to the barest minimum they concluded that DFA is the easy way to express regular expressions. The DFA is stored into memory has compressed rules. The compressed DFA of regular expressions is used at the end in regular expression matching process.

3. Tokenization

The tokenization component scans web pages and breaks them into tokens by the process of lexical analysis. HTML tags are analyzed to get layout information: after this phase, they are removed. Table 1 represents the way to define a lexical analyzer.

Table 1: Definition of lexical analyzer

Token	Description
(\s+)	Whitespace
(//)[^\n]*	Comments
0[xX]([0-9A-Fa-f]+)	hexadecimal integer literals
(\d+)	integer literals
(<<|>>)	multi-char punctuation
([][(){}<>=,;:*+-/])	Punctuation
([A-Za-z_][A-Za-z0-9_]*)	Identifiers
"""(.*?)"""	multi-line string literal
"((?:[^"\n\\]|\\.)*)"	regular string literal
(.)	an error!

Table 1 shows a way to tokenize an input string; this approach scans the tokens in the input string. The caveats are:

i. Each token must explicitly match
ii. The regex matcher will silently skip past errors in the input string!

4. Regular Expression Notations

To extract email addresses from web pages, the regular expression approach requires address elements to be identified in order to find the boundaries of address text blocks as shown in Table 2. It applies pattern matching to find address elements:

Table 2: Tokenization using regular expressions

Pattern	Description
$	Matches the end of the line
\s	Matches whitespace
\S	Matches whitespace
*****	Repeats a character zero or more times
\S	Matches any non-whitespace character
***?**	Repeats a character zero or more times (non-greedy)
+	Repeats a character one or more times
+?	Repeats a character one or more times (non-greedy)
[aeiou]	Matches a single character in the listed set
[^XYZ]	Matches a single character not in the listed set
[a-z, 0-9]	The set of characters can include a range
(	Indicates where string extraction is to start
)	Indicates where string extraction is to end

Lots of patterns match the patterns to the text and extract the necessary facts. Patterns may be manually entered.

Rules

The following heuristic rules are used to match address elements and extract addresses.

i. non-textual in the web page are identified as 0-9
ii. textual are identified as a-z
iii. All textual that are immediately preceded by @ character are marked domain names
iv. All textual that are before @ character are marked user names.

5. Rule-Based Extraction Method

This method buttressed on the volume of email addresses shown on the web following the rule-based extraction method format by starting with word/number then followed by @ character

and domain name as shown in Figure 2. A number that occurs could be [0-9]. Using regular expression to build extraction system.

<word>|<word><number><@><domain name>

Figure 2: Token Pattern Lexeme

6. Algorithm1 (Tokenization with Regular Expression)

Address extraction with regular expressions is achieved using the following procedure:
1: Pre-Processing: The input web page is pre-processed to get clean text.
2: Matching "user.some": should match "some" or "user" or "user_some" or "user-some" or Matching "user.some": should match "some" or "user" or "user_some" or "usersome" or Matching "user.some": should match "some" or "user" or "user_some" or "someuser".
3: Matching domain "gmail.com": should match "yahoo.com" or "msn.com" or ust.edu.ng etc
4: Matching number "101": should match zero-nine digits.
5: Extracting address: Text blocks that begin with optional textual /non- textual followed by @ character, and end with domain name are extracted as email addresses.

Table 3: Matching Pattern

S|N	Symbol	Matching Pattern
1	user.some	<user>|<some>|<user.some>|<usersome>|<user_some>
2	Firstname.Lastname	<firstname>|<lastname>|<firstname.lastname>|<firstnamelastname>|<firstname_lastname>
3	string@literal	<string>|<literal>|<string@literal>|<stringliteral>|<string_literal>
4	Identifier_digit	<identifier>|<digit>|<identifier_digit>|<identifierdigit>|<identifier_digit>
5	Firstname-identifier	<firstname>|<identifier>|<firstname-identifier>|<firstnameidentifier>|<firstname_identifier>

7. Word N-Gram Model (Machine Learning)

One of the most widely used model for natural language is the N-Gram modeling. N-gram is a combination of n letters. Examples:
Unigram for 1 letter
Bigram for 2 letters combinations
Trigram for 3 letters combinations
A language model is a probability distribution over an entire sentence or texts. Language modeling can be viewed as classification. N-gram model predicts the occurrence of a word based on the occurrence of its N – 1 previous word. For instance, a bigram model (N = 2) predicts the occurrence of a word given only its previous word (as N – 1 = 1 in this case). Similarly, a trigram model (N = 3) predicts the occurrence of a word based on its previous two words (as N – 1 = 2 in this case). Word n-gram model is used to represent the surrounding context of a token to be classified. In this model, the token at position i is defined as t_i, the feature set of the t_i is defined as f(t_i). The n-gram of token t_i with order n is defined as:

$$\mathbf{ngram_i(n) = \{t_{i-n+1},\ldots,t_{i-1}, t_i, t_{i+1},\ldots,t_{i+n-1}\}} \quad \mathbf{(1)}$$

We define the feature set of a n-gram as the combined feature sets of all tokens contained in the n-gram:

$F(ngram_i(n)) = f(ti-n+1)$ U…U $f(ti-1)$U $f(ti)$U $f(ti+11)$ U …U $f(ti+n-1)$ (2)

For each n-$gram_i$ (n), the value of each feature in the feature set is calculated, and subsequently used along with class label of t_i for training and testing classifier. After the classifier is trained, it reads the feature values of each token and decides which class label should be assigned to token t_i according to its feature values along with information from its n − 1 pre-words and post-words. The number of n − 1 dummy words shall be added to both the beginning and ending of source text to make the first and last n−1 tokens of the text have the same feature set as other tokens.

Example: input text

For more information, email John Pretty j.pretty@prettyawesome.net

Example: Word n-gram output(n=2)

For more ---> {attribute, ..., attribute} ---> OTHER
For more information ---> {attribute, ..., attribute} ---> OTHER
More information email ---> {attribute, ..., attribute} ---> OTHER
Information email John ---> {attribute, ..., attribute} ---> OTHER
Email John Pretty ---> {attribute, ..., attribute} ---> OTHER

John Pretty ---> {attribute, ..., attribute} ---> Other
j.pretty ---> {attribute, ..., attribute} ---> START
j.pretty@ {attribute, ..., attribute} ---> MIDDLE
j.pretty@prettyawesome.net {attribute, ..., attribute} ---> END

8. Post-Processing

This is the last component of the extraction system: it outputs the final extracted addresses. The predicted results of the classifier are a sequence of tokens with predicted labels, which are one of START (textual or non-textual or mixture of both), MIDDLE (@ character), END (domain name). This component identifies the address block by applying rule matching to the predicted results. A block of tokens is identified as an address only if its first token is predicted as START by the classifier, and its last token is predicted as END.

Features

Textual Level Features: are widely used in Information Extraction systems, it provides capitalization and digitalization.

Non-Textual Level: Provides number ranging from 0-9.

Punctuation: Punctuation can also help the classifier capture the syntactic context of the token and improve the accuracy of address boundary detection. For example, addresses in web pages are typically segmented into fields.

- **i.** Comma (,)
- **ii.** Period (.)
- **iii.** Dash (-)
- **iv.** Underscore (_)
- **v.** At (@)

Table 4: Textual level features

Feature	Description	Example
ALLCAPS	All characters capitalized	TEST@DOMAIN.ORG
TEXTUAL	Contain words	test@domain.es
CONTAINDASH	Contains at least one dash	Test-user@domain.com or test_user@domain.net
PUNCTUATION	Punctuation	Test.user@domain.de
MIXTURE	Mixture of both Textual and Non-Textual	Test101@domain.co.uk

ALGORITHM 2 (Combination of the rule-based and classifier)
Input: Web pages with each token labeled as one of the four classes: START (textual or non-textual or mixture of both), MIDDLE (@ character), END (domain name).
1: for each input web page do
2: The web page is pre-processed to get textual document or web pages' information, and then tokenized.
3: Tokens are fed into two rule-based systems.
4: The tag assigned to each token by each system is noted, and used as a feature of the token.
5: Tokens are tagged textual, non-textual, other feature attributes are extracted as well.
6: n-grams are generated for each token; the feature attribute set of each n-gram is then constructed.
7: The classifier is trained with the feature attributes and corresponding class labels.
8: end for
The system begins by extracting and selecting tokens, if the token extracted are not email features it will return to feature selection, otherwise move to classifying the selected features by START (textual or non-textual or mixture of both), MIDDLE (@) and END (domain).
An address extraction system is a system that takes input from web pages, extracts and outputs email addresses contained in those web pages.

9. Results and Discussions

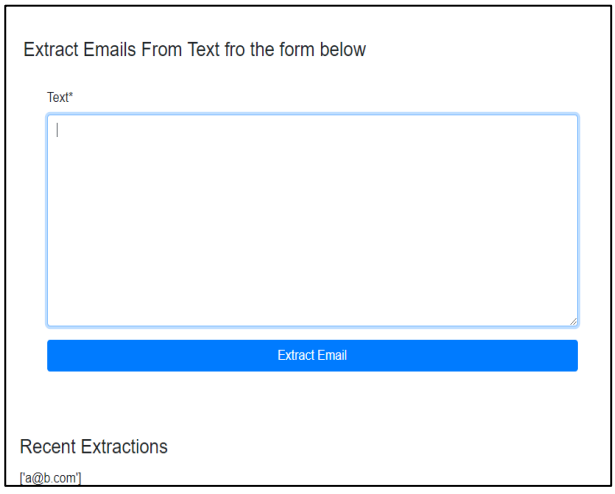

Figure 3: Email Extraction Tool

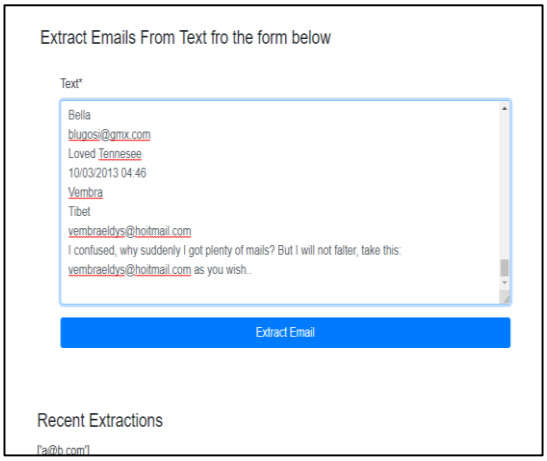

Figure 4. Text collected to be Extracted

S|N	TIME (Second)	EMAIL COUNT	SIZE OF DATA (Bytes)
1	00.03	16	14,557
2	00.17	1024	323,615
3	00.11	654	161,724
4	00.07	377	77,882
5	00.10	615	117,062

Table 5: Time Log for different data sizes

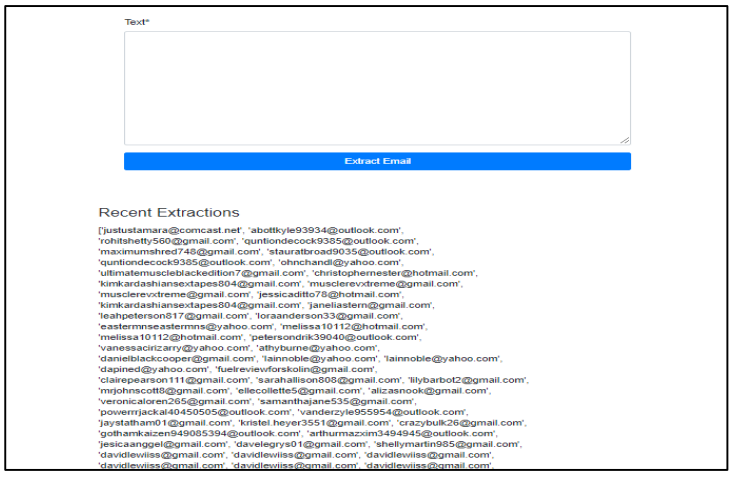

Figure 5. Extracted Email Addresses

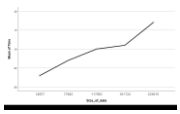

Figure 6: Graph showing the mean of time against the size of the data of Table 5

Table 6: Data size before and after Extraction

DATA SIZE(KB)	DATA BEORE EXTRACTION	DATA AFTER EXTRACTION
20	1587	517.5
40	3174	1035
60	6348	2070
80	12696	4140
1000	25392	8280
10000	50784	16560
20000	101568	33120
30000	203136	66240
40000	406272	132480
50000	812544	264960

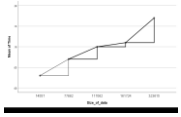

Figure 7: Graph showing the slope for the extraction model

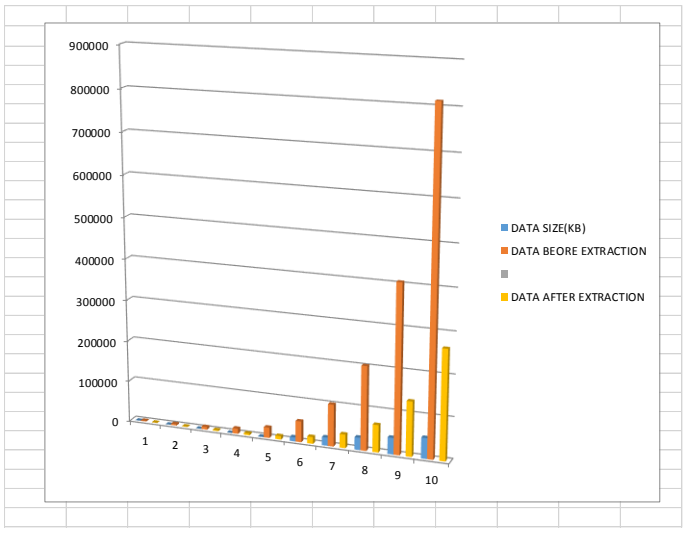

Figure 8: Graph showing data size before and after extraction of Table 6

The tool was built to extract email addresses from text document. The extraction tool is made up of text area that could contain large amount of data (textual and non-textual), "Extract Email Button", "Text Area" and "Text View" (displays extracted addresses) as shown in Figure 3. Text Area: This contain the textual and non-textual information to be extracted. The text document to be extracted were gotten from Google query containing textual and non-textual (number, alpha-numeric, date and time, special characters and email address) as shown in Figure 4, the document was inserted via copy and paste, the tool could accept large data at a time. Extract Email Button: This button when clicked filters all the information in text area, and discard other information that is not Email Address related. Text View: This displays the extracted Email Addresses in threes on a row, whereby each Email Address is separated with a comma as shown in Figure 5. The extraction tools discarded all textual and non-textual document based on the rules-based and classification specified by undergoing the processes of tokenization, feature extraction and selection (START, MIDDLE and END), classification, email processing. Table 5 shows the graph of time and the different sizes of data indicating that as the size of data increases inside the tool, the time taken to extract the emails also increases significantly. Figure 6 shows the mean of time against the size of data indicating that the change in time is not completely linear. There is break at regular interval where the slope of the graph and be calculated to represent the size of the data. Table 6 show the graph of data size before and after extraction. Figure 7 shows the graph of time and the size of data and how the slope at different time interval can be calculated. The data sets are all in kilobyte with a graph scale of 10:20. From the bar chart in Figure 8, it is obvious that there is significant difference between the sizes of data measured as against various parameters i.e data collected before extraction, and data collected after extraction.

10. Conclusion

This paper that developed a tool for data extraction in big data using two efficient algorithms (rule-based and machine learning based) have shown that extraction of email addresses from some tested data sizes was successful. The extraction has been hosted in Heroku server online making it a web based application. The related works gave an insight on the different methods and algorithms used for data extraction prior and after big data that has been carried out previously. All the shortcomings of the existing algorithms were observed and taken into consideration for the achievement of the required results. For optimal result and in an attempt to combat most of the challenges of other extraction tools used for big data, a reasonable amount of raw data should be pasted into the developed extraction tool to assist in information dissemination in the shortest possible time. This work is the beginning of a unique research in the area of data extraction. However, this research approached data extraction in a new dimension as regular expression was the rul- based pattern used and machine learning method. Data set were being analyzed and python programming language used to achieve the tool because of its compatibility with Hadoop. This research when utilized in organizations (governmental or non-governmental) can be used to get the email addresses of staff in a short time for information dissemination.

References

Albert, B. (2002). Mining big data in real time. *Informatica*, 37, 15–20

Alexander T. (2006). Using Regular Expression to Abstract Blood Pressure and Treatment Intensification Information from he Text of Physician Notes". *Journal of the American Medical Informatics Association*, Pages 691–695,.

Baesens, B. (2014). Analytics in a big data world: The Essential Guide to Data Science and its Applications. GOOGLE. pp.15-20

Bernice, P. (2013). The emergence of big data technology and analytics. *Journal of Technology Research.*

Duy D, An B, Qing Zeng-Treitler (2014). *Learning regular expression for clinical text classification.* Journal of the American Medical Information Association column 21, issue 5

Jonathan, S. W., & Adam, B. (2013). *Undefined by data: A survey of big data definitions* (Master's thesis), University of St. Andrews, UK.

Khan. N., Habib, S., Gran, B., Soulmaz, S., Mohammed, A. & Aftab, A. A. (2018) The 10 Vs, Issues and Challenges of Big Data. ResearchGata

Kitchin, R. (2014). Big Data, new epistemologies and paradigm shifts: SAGE Journal DOI: 10.1177/2053951714528481. pp. 1-12

Kumar, S., Kamesh, K. & Syed U. (2014). A study on Big Data and its Importance, *International Journal of Applied Engineering Research* ISSN 0973-4562 Volume 9, Number 20. pp. 7469-7479

Utkarsha P. Pisolkar, Shivaji R. Lahane (2015). A memory Efficient Regular Expression Matching by Compressing Deterministic Finite Automata. International journal of computer Application vol 122-no.20

Vikram P. S., Madhusudhana R. E (2013) *Big Data - Solutions for RDBMS Problems – A Survey.* International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 9. pp 3686 - 3693