© 2016 IJEDR | Volume 4, Issue 3 | ISSN: 2321-9939

# Effective Semantic Search over Huge RDF Data

[1]Dinesh A. Zende, [2] Chavan Ganesh Baban
[1] Assistant Professor, [2] Post Graduate Student
Vidya Pratisthan's Kamanayan Bajaj Institute of Engineering & Technology, Baramati
Pune, India.

*Abstract:* **Keyword search is a basic way of searching data on the web. It offers a straightforward and flexible way for retrieving relevant information. It is studied in a context of relational databases, XML schema, graph and RDF data. Previous techniques based on constructing distance matrix or building summaries from RDF graph to process the query, but not seems to be realistic and can't handle large RDF data. It leads to the incomplete or incorrect result. This work proposed an effective summarization algorithm to summaries RDF data. Summaries were built and incrementally updated to support content update and gave significant pruning power.**

*Index Terms:* **Keyword search, RDF graph, RDF data, scalability**.

## I. INTRODUCTION

Keyword search used to retrieve relevant data from database using keyword query. It can be used in the context of relational databases, XML documents, graphs and RDF data. RDF consists of triples in the form of subject, predicate and object. As day by day contents on the web are growing and changing so, RDF is preferred. Keyword search as a context of relational databases [1,2,3] requires structured content as it's widely used technique but difficult due to data transformation that eliminates redundancy. Keyword search as a context of XML documents [4,2] stores XML documents in a relational database and then optimize query processing using graph schema which executes on all possible expressions instead only most promising nodes. Keyword search as a context of the graph [5,8] unable to handle the millions of vertices as data sizes grown and as contents updated. Previous work as the context of RDF graph [6,7] also have some limitations as it returns incomplete or incorrect results for various data sets and inability to scale large RDF data.

Proposed approach gives the solution to the problems detects in previous techniques by designing scalable and exact solution which handles large RDF dataset. Backward search idea is based on baseline method used to address incomplete results generation. This work provides scalability which leads in significant pruning power and speeds up search by designing a summary which is light weighted and updatable. The semantic keyword search in this approach should provide scalable and correct result. The formatting will need to create Ease of Use for semantic keyword search.

## II. RELATED WORK

Various keyword search techniques are built to retrieve relative data. Keyword search techniques applied on unstructured, semi-structured and structured [14] data. To process a given keyword query in relational databases based on indexing [15] generates the response as answers and calculate a score value for each keyword answer. The value of scores arranged according to the highest ranked specific answers.

In schema-based approach [4] on databases, keyword query is processed by matching keywords to database elements. Using the valid schema sequences of joins are derived, then these schemas used to combine the elements of calculated keywords form networks which represents possible results for given query.

The technique proposed by H. Wang [5] based on a graph with bi-level indexes, which is used for accelerating search. In this, the graph is partitioned into blocks to avoid memory space for indexes. To guide for search among blocks, summary information at the block level is stored in bi-level indexes. It uses backward search strategy with balanced cost expansion, which provides the graph consist keyword from the query. The indexing supports bi-directional search i.e. backward as well as forward at the same time to make an effective search. This approach partitions a data graph into various blocks and bi-level index consist summary information at the block level to provide search among blocks.

M. Kargar [6] proposed an algorithm that finds r-cliques in graphs. The r-clique is a collection of content nodes which covers all keywords from the query. The distances between two nodes considered as 'r' or less than 'r'. To find all r cliques in the input graph algorithm is used based on branch and bound method. An approximation approach finds all r cliques with two approximations and makes an ascending order of their weights with polynomial delay. This approach explores only the content nodes rather than the whole graph.

P. Karras [7] Proposed an approach in which indexing is done on RDF data in six ways, for three RDF elements one for each possible order. The same priority should be given for subjects, properties, and objects. Each RDF element type contains index designed around it. The order of arrangement of these elements in an indexing scheme is developed. Every instance of an element of RDF is a concern with two vectors; each vector collect elements of other which emerge sextuple index. The result provides a sextuple index maintains six indices a known as hexacore. Each index belongs an RDF element and defines a priority of other two elements.

T. Tran [10] proposed a Schema based approach for graph structured data focuses on RDF data model. In given approach first queries are generated from keywords rather than finding answers. To generate queries, the keywords are mapped to elements of the data graph. These keyword elements in graph explored to find an element which is connected to all keyword. The paths between the connecting them are combined to design a matching sub-graph. Based on the structure of triple of a query and edges, vertices are mapped to variables or constants, and edges are mapped to predicates for computing queries. Exploration is perform using indexing on a summary data structure which derived from the data graph. Finally, users are allowed to choose specific query which is processed using database engine to retrieve the specific result.

## III. PROPOSED SYSTEM

RDF(resource description framework)approach makes use of triples, which is in the form of the sentence like structure. It is also considered as triple which is in the form of subject, predicate and object. It is standard for representation of data on the web. Information about resources is represented by RDF data model. Rather than display to the individual, RDF is intended for processing web resources required by applications. Fig. 1 shows an example of simple RDF graph with possible triples.

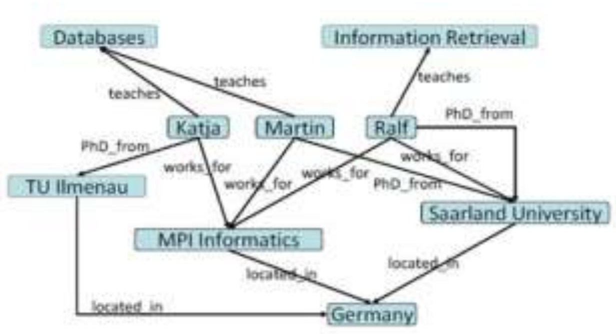


**Fig.1.** RDF graph.

Proposed keyword search technique aims to provide scalable, effective and fast search on big RDF data files. RDF data files may compose of millions of triplets from several databases. The fast growing RDF information files from different domains not successfully handled by existing technique. To address the incorrect or incomplete result generation and handle updates need a mechanism. This scheme uses backward search technique with correct termination condition. Partitioning use to split large RDF graph into smaller with type based summary speed up the search. Summarization generates a template which serves as a summary of partition associated with one partition. To support an addition to the RDF data, which is collected in an RDF store, the data of a partition is retrieve using SPARQL/SQL queries from the partition.

As shown in Fig. 2 RDF dataset is accessed, and RDF graph is built referring such Dataset. From such large RDF graph, it needed to be divided into smaller sub-graphs and partitioned into smaller and semantically similar structure. Information for partition which is considered as summary generates as a template by Summarization process. There is one to one association between template and partition. Using summary index, from the collection of partitions the templates identified. Auxiliary indexes use to concatenate two partitions and also map partition candidate to data in the summary. Using summary index keyword search by estimating the path length.

The proposed system composed of different techniques for searching keyword in RDF datasets such as partitioning, Summarization, Indexing and Keyword search with a summary as follows.

**Partitioning:**

Partitioning use to split large RDF graph into smaller graphs and those small graphs are semantically similar. This graph partitioning is based on the type based summary. Doing this speed up search achieved. Partition is performed on input RDF files and produces small graphs containing neighboring nodes with similar type information. It provides collection of partition with similar semantics

.

**Summarization:**

The partitions are processed with all its neighbors to identify core partition. All cores from data files are identified and add these cores as a summary structure. Summarization generates a template which serves as a summary information for partition. Each template associates with at least one partition which can be used for retrieval purpose.

**Indexing:**

Indexes are used for the fast search. Indexes were designed by giving the appropriate identifier to the search patterns which belong from same query patterns. The search patterns which are considered to be similar and retrieved by the same type of query are grouped together. When such type of query entered for the search, only indexes from that pattern should be searched and avoids unnecessary searching.

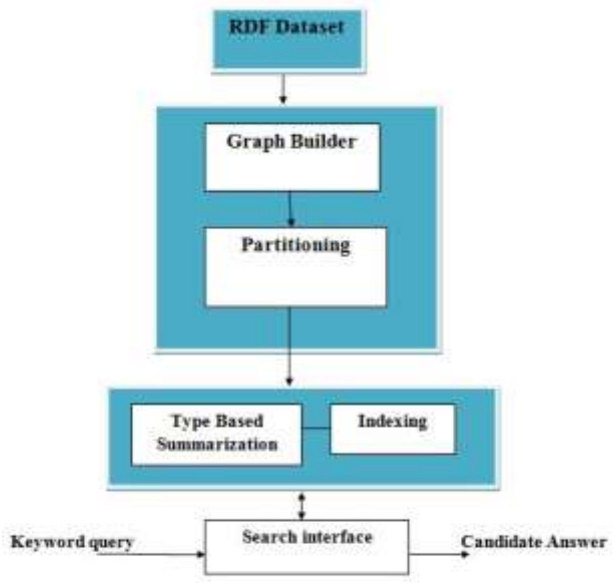


**Fig.2.** System Architecture.

**Keyword Search:**
Auxiliary indexes used to concatenate two partitions and for mapping purpose. Two level searches performed, first summary level and another data level search. It uses backward search which will retrieve top n results for entered query keyword. Using indexes keyword search checks all summaries and matching summary entry searches for data in the mapped partition without intact other partitions. The result is given in the form of candidate answer.

Natural language query interface is also provided for searching with which user can enter simple as well as join queries with natural language. Natural language query then converted into SPARQL query for searching.

## IV. IMPLEMENTATION DETAILS:

### 4.1. Algorithms:

**Partition the RDF graph into sub-graphs.**

**Input:** RDF graph with the set of vertices V and edges E With α hops.
**Output:** the set of partitions P.

1) Identify the distinct types T of V.
2) Initially partition P will be empty.
3) Find the edge/edges of high betweenness.
4) Remove these edges from the graph which separates it into multiple components as sub-graphs.
5) Repeat the step 4 as long as edges remain in the graph.

**Summarization and indexing the partitions in RDF graph.**

**Input:** Set of partitions with α hop neighborhood.
**Output: the** set of Summaries in S and indexes.

1) Initially set of summaries S will be null.
2) Process partitions in P with all neighbors.
3) Identify the core from all partitions.
4) Identify the homomorphic cores.
5) Add core as summary structure.
6) Return the set of summaries in S as a core.
7) Summaries are identified and associated with indexes.
8) Indexes built for specific search patterns for searching.

### 4.2. Mathematical Model:

**Input set:**

1. Let $G=\{V,E\}$ - is the RDF graph for any vertex $v \in V$.
   Where

$V=\{ v_m ; \quad 0<m<n \}$ -set of vertices containing keyword.

$v=\{ s_i , o_i ; 0<i<n\}$

where s, o represents set of subject and objects in triplet of RDF data.

$E=\{ p_i ; 0<i<n \}$ -set of edges represent predicates of triplets.

2 . $q=\{w_1,\ldots, w_n\}$ -set of keywords in query.

**Preprocessing Sets:**

1. $P=\{\{G_T\}, \alpha, h, P_o \}$ for $G_T \in G$
   where
   α is constant denotes hops to reach to the keyword.
   $G_T =\{V_i, E_i, T_i ;0<i<n\}$ -set of vertices with distinct types
   $P_o=\{h_i, \alpha ;0<i<n\}$ -set of partitions
   Where h represents α hop neighborhood.

2. $S=\{P_o\}$
   After finding core 'c'
   $S=\{P_o, c\}$ -set of partition with core 'c' where

   $f(c,s)= 1$ -c is homomorphic to s.
   $= 0$ -otherwise.

3. $I=\{L,\sum\}$ -is index to search the partition
   Where L is portal node joins different partitions.
   $\sum=\{\sigma(v_i); 0<i<n\}$ -is set of all one to many mapping in partition h.

4. $K=\{G, q, W, a, M, I\}$
   Where
   $W=\{w_1,w_2,\ldots\ldots,w_m\}$ –set of vertex containing keyword.
   $a=\{a_1,a_2,\ldots\ldots..,a_m\}$ - is m priority queue.
   $M=\{u,r,d_l,d_u\}$ -set of entries for visited partitions.
   where
   u represents first vertex; r denotes path to reach to the partition from u with lower bound $d_l$ and upper bound $d_u$.

**Output Sets:**

$R=\{A(q)\}$ - set of top k answers for query.

Let Y be the system, we mathematically represent Y using set theory as,

$Y= \{ G,q,P,S,I,K\}$.

## V. OUTPUT DESIGN AND RESULTS ANALYSIS:

In given approach the dataset used as a large synthetic and real RDF data generated by dblp which includes data about conferences. The second dataset includes Lehigh University Benchmark which includes data of students, departments, etc.. The given dataset is assumed to be in the form of condensed view of RDF graph which is used for further processing.

The system performs keyword search beginning from RDF graph, partitioning, summarization with indexing and natural language query interface. These operations are shown in fig.3, fig.4 and fig. 5 given bellow.

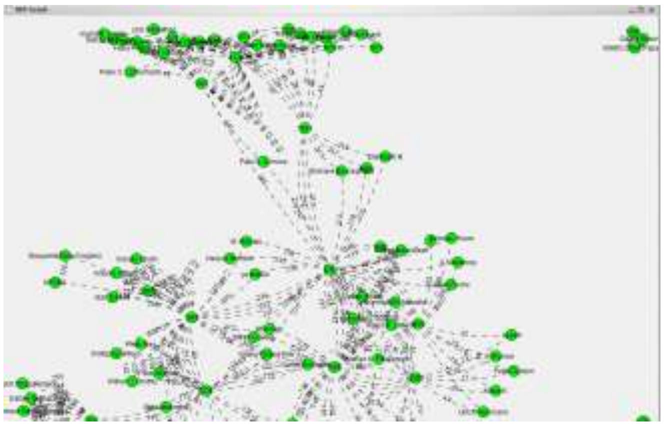
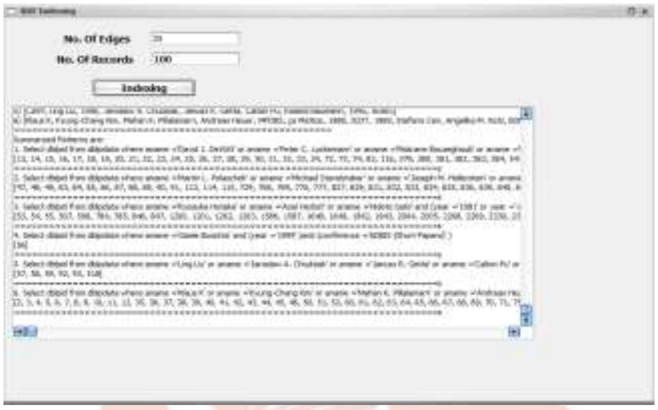


**Fig.3**. RDF Graph for dblp.

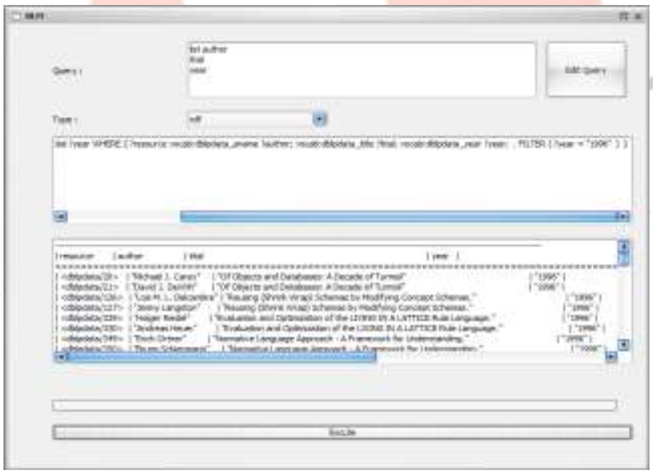


No. Of Edges 21
No. Of Records 100

Indexing

```
[illegible]
[illegible]
=====================================================
Summarized Patterns are:
1. Select dblpid from dblpdata where aname ='David J. DeWitt' or aname ='Peter C. Lockemann' or aname ='Philippe Bourgnhoud' or aname =
[13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 72, 77, 79, 83, [illegible]
=====================================================
2. Select dblpid from dblpdata where aname ='Martin L. Polaschek' or aname ='Michael [illegible]' or aname ='Joseph M. Hellerstein' or aname
[47, 48, 49, [illegible]
=====================================================
3. Select dblpid from dblpdata where aname ='[illegible]' or aname ='[illegible]' or aname ='[illegible]' and (year ='1997' or year ='
[53, 54, 55, [illegible]
=====================================================
4. Select dblpid from dblpdata where aname ='[illegible]' and (year ='1999' )and (conference ='SEBD' (Short Papers)' )
[36]
=====================================================
5. Select dblpid from dblpdata where aname ='Ling Liu' or aname ='[illegible]' or aname ='[illegible]' or aname ='Calton Pu' or
[57, 58, 89, 92, 93, 118]
=====================================================
6. Select dblpid from dblpdata where aname ='Maus K' or aname ='Kyung-Chang Kim' or aname ='Mehan K. Pillaiman' or aname ='Andreas Heu
[2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 50, 51, 52, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 7
=====================================================
```

**Fig.4**. Indexing of  RDF Graph



Query : [illegible] author
[illegible]
[illegible]

Edit Query

Type : rdf

```
[illegible] ?year WHERE { ?resource <[illegible]dblpdata_aname> ?author; <[illegible]dblpdata_title> ?title; <[illegible]dblpdata_year> ?year; . FILTER ( ?year = "1996" ) }
```

```
| resource        | author               | title                                                        | year   |
=================================================================================================
| <dblpdata/20>   | "Michael J. Carey"   | "Of Objects and Databases: A Decade of Turmoil"              | "1996" |
| <dblpdata/21>   | "David J. DeWitt"    | "Of Objects and Databases: A Decade of Turmoil"              | "1996" |
| <dblpdata/126>  | "Lois M. L. Delcambre" | "Reusing (Shrink Wrap) Schemas by Modifying Concept Schemas." | "1996" |
| <dblpdata/127>  | "Jimmy Langston"     | "Reusing (Shrink Wrap) Schemas by Modifying Concept Schemas." | "1996" |
| <dblpdata/229>  | "Holger Riedel"      | "Evaluation and Optimization of the LIVING IN A LATTICE Rule Language." | "1996" |
| <dblpdata/230>  | "Andreas Heuer"      | "Evaluation and Optimization of the LIVING IN A LATTICE Rule Language." | "1996" |
| <dblpdata/349>  | "[illegible]"        | "Normative Language Approach - A Framework for Understanding." | "1996" |
| <dblpdata/350>  | "[illegible]"        | "Normative Language Approach - A Framework for Understanding." | "1996" |
```

Exit

**Fig.5**. Searching Query on the system.

### 5.1 Result Discussion:

Results of searching are mainly dependent on the size of RDF dataset and the time required for partitioning, building the summaries, size of summary and indexes for searching. As the size of RDF data given as triples present in it. Previous approaches like schema method are also faster however the results of such system are not correct for all RDF dataset. The summaries built by Schema method might be useful in some setting, but it makes unnecessary search as not desired for keyword searching.

### 5.2 Analysis Graph:

In graph table, the existing system like Schema searching time for the query entered by the user is near about proposed approach, but the proposed system gives an exact result for given query and time factor is also less than existing approach. Graph shows in fig. 4 comparisons between Schema and Proposed system by considering the time required for partitioning, summarization, indexing and search on size of data in number of triplets present in RDF dataset.

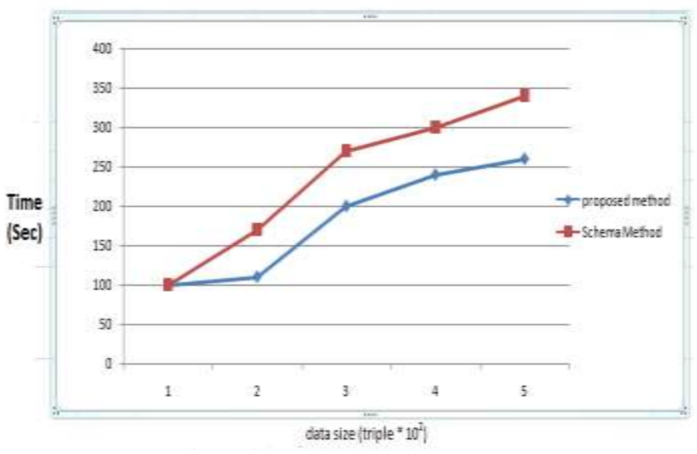


**Fig.4.** Analysis graph

## VI. CONCLUSION

Keyword search in the RDF data graph deals with the problems such as the size of RDF data and the specific result returned by the system. As the data size increased in RDF graph previous approaches unable to handle such situation efficiently and which returns an incorrect solution with the unnecessary search. Proposed approach prevents unnecessary search by including a specific index for search and retrieve specific result to the user. It uses splitting given large RDF graph into smaller sub -graphs as partitions and design specific summary according to types of such partitions. Which provides scalability to the system as data get updated or modified system provides support to update summaries so, the system handles changes in the content efficiently.

## VII. ACKNOWLEDGEMENT

This paper would not have been written without the valuable advice and encouragement of Prof. D. A. Zende guide of ME Dissertation work. We thank Prof. P. M. Patil and Prof. S. S. Nandgaonkar, Head of Department and Hon'ble Principal Prof. V. U. Deshmukh, for their valuable support and for giving an opportunity to work on this paper.

## REFERENCES

[1] S. Agrawal, S. Chaudhuri, and G. Das, “ DBXplorer: A system for keyword-based search over relational databases,” In ICDE, 2002.
[2] V. Hristidis and Y. Papakonstantinou, “Discover: Keyword Search in Relational Databases,” Proc. 28th Int’l Conf. Very Large Data Bases (VLDB).
[3] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan, "Keyword Searching and Browsing in Databases using BANKS, " In ICDE, 2002.
[4] Y. Luo, X. Lin, W. Wang, and X. Zhou, “Spark: Top-K Keyword Query in Relational Databases,” Proc. ACM SIGMOD Conf., pp. 115-126, 2007.
[5] H. He, H. Wang, J. Yang, and P.S. Yu, “Blinks: Ranked Keyword Searches on Graphs,” Proc. ACM SIGMOD Int’l Conf. Management of Data (SIGMOD ’07), 2007.
[6] M. Kargar and A. An, “Keyword Search in Graphs: Finding Rcliques,”Proc. VLDB Endowment, vol. 4, pp. 681-692, 2011.
[7] C. Weiss, P. Karras, and A. Bernstein, “Hexastore: Sextuple Indexing for Semantic Web Data Management,” Proc. VLDB Endowment, vol. 1, pp. 1008-1019, 2008.
[8] Wangchao Le, Feifei Li, Anastasios Kementsietsidis, and Songyun Duan, "Scalable Keyword Search on Large RDF Data"," Proc. 10th Int’l Conf. Semantic Web IEEE TRANSACTIONS ON KNOWLEDGEAND DATA ENGINEERING, VOL. 26, NO. 11, NOVEMBER 2014.
[9] Y. Guo, Z. Pan, and J. Hein, "LUBM: A benchmark for OWL Knowledge BaseSystems", J. Web Semantics, vol. 3, pp. 158-182, 2005.
[10] T. Tran, H. Wang, S. Rudolph, and P. Cimiano, "Top-k Exploration of Query Candidates for Efficient Keyword Search on Graph-Shaped (RDF) Data", Proc. IEEEIntl Conf. Data Eng. (ICDE), 2009.
[11] C. Bizer and A. Schultz, "The Berlin SPARQL Benchmark", Intl J. Semantic Web and Information Systems, vol. 5, pp. 1-24, 2009.
[12] Y. Chen, W. Wang, Z. Liu, and X. Lin, "Keyword Search on Structured and Semi-Structured Data", Proc. ACM Intl Conf. Management of Data (SIGMOD), 2009.
[13] G. Li, B.C. Ooi, J. Feng, J. Wang, and L. Zhou, “Ease: Efficient and Adaptive Keyword Search on Unstructured, Semi-structured and Structured Data,” Proc. ACM SIGMOD Int’l Conf. Management of Data, 2008.
[14] Q. Su and J. Widom, —Indexing Relational Database Content Offline for Efficient Keyword-Based Search, Proc. Int’l Database Eng. and Application Symp. (IDEAS), 2005.
[15] J. Broekstra et al., “Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema,” Proc. First Int’l Semantic Web Conf. The Semantic Web (ISWC), 2002.

**Author Profile**

**Prof. Dinesh A. Zende** is working as Assistant Professor in the department of Information Technology, VPKBIET Baramati. Dinesh has 15 years of teaching experience at undergraduate and postgraduate level. He has received a Summer Faculty Research Fellowship from IIT, Delhi in 2015. His research interests are Machine Learning and Social Media Data Mining.

**Mr. Chavan Ganesh B. :** Received B.E. Computer in 2007 from North Maharashtra University and currently the student of M.E. second year at VPCOE, Baramati. He also works as an Assistant Professor in Department of Information Technology Engineering at SVPM's C.O.E. Malegaon(Bk), Pune(India).