International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-9 Issue-3, January 2020

Document Clustering based on Phrase and Single Term Similarity using Neo4j

Preeti Kathiria, HarshalArolkar

Abstract:Document similarity generally rely on single term similarity such as cosine similarity. To achieve better document similarity, along with single term phrase- more informative feature can be used. To find out shared phrases across the corpus the Document Index graph (DIG) representation model is used. Document representation - DIG model incrementally construct the graph and simultaneously finds the shared phrase between current document and previously inserted documents from the graph. The similarity between documents is mainly depends on the number of shared phrases and single term similarity – known as hybrid similarity. The hybrid similarities are used with well-known density based clustering technique DBSCAN to assess their effect on quality of the clusters. Experimental results shows that hybrid similarity gives more accurate degree of document similarity and performs better cohesive clustering.

Keywords: DBSCAN clustering, Document Index model, Neo4j Graph database, Phrase based similarity

I. INTRODUCTION

Document similarity can be used to get the better search result, automated matching of CV to job description, Document clustering etc. The traditional way of finding similar text documents is based on single term similarity. The well- known approach for finding single term similarity is vector space model with cosine similarity. A mathematical model, Vector space requires the data in the numerical form. So, the text data must be converted into numerical data. Different techniques are there to represent text data into numerical form like bag- of- words(BoW), term frequency(tf), term frequency–inverse document frequency(tf-idf). The simple bag of words model represents the set of unique words in the document and the frequency of their appearance in the document. The tf model denotes the ratio which is given by frequency of their appearance in the document and the document size. The tf model indicates how particular term is important for the document. Furthermore, tf-idf model is represented using equation tf value * idf value, whereas idf is the ratio of total number of documents in dataset to the number of documents in which particular term resides. The tf-idf model indicates how particular term is important to the whole dataset.Vector space model contributes widely in research and industrial applications as well but, it has a limitation.

Revised Manuscript Received on January 30, 2020.
***** Correspondence Author
PreetiKathiria*, Assitant Professor, Department of Computer Science and Engineering, Nirma University, Ahemdabad, India. Email: Preeti.kathiria@nirmauni.ac.in
HarshalArolkar, Faculty Department of Computer Technology, GLS University, Ahemdabad, India. Email: harshal.arolkar@glsuniversity.ac.in

The similarity calculated in vector space model is based on single term of the documents. Single term analysis does not cover the proximity of the words- so phrase based similarity needs to be considered.

For getting phrase based similarity different document representation models like n-gram[9], suffix tree[3], document index graph(DIG)[7],[8] is used. The n-gram uses the matrix data structure. In n-gram[9] model the combination of n adjacent words called n-gram are used. The value of n defines the phrase size. In the model, matrix is generated as the n-grams as rows and documents identification as columns. Finally, the cosine similarity is applied to find the similarity between the documents. N-gram model has some limitations: As the number of documents increases it generates high dimension sparse matrix and also it finds fixed length phrase. The suffix tree model uses tree data structure [3]. The tree is constructed by moving word to word for each sentence, the entire sentence is inserted as first node, and now from next word the rest of the sentence is inserted as next node. Similarly, it keeps inserting the part of the sentences as nodes. If a similar part of the sentence is detected in the sentence, in this case the node is fragmented to form two nodes, parent node consider the common phrase and the child node consider the uncommon phrase. Each node of the suffix tree stores information like - the length of word sequence stored in the node, the document that the word sequence resides and the frequency of the word sequence resides in each document. The method finds out dynamic length of shared phrases but it stores redundancies in the form of suffixes. In the DIG model the documents are stored into single graph sentence by sentence. In graph model words are store as nodes and the proximity of the words are stored as edges. This way it keeps the inherent structure of the original document. The graph model can be stored in relational databases or in other databases like no-SQL database, graph databases. The DIG model is used to find common phrases among documents so the connectivity of the words are important and therefore neo4j graph database has been used.

The paper is structured as follows: section 2 talks about the related work. Section 3 explains the Document Index Graph model. Section 4 provides the database paradigm for DIG model implementation.Section 5 shows methodology of construction of DIG model in Neo4j graph database, calculation of hybrid similarity and clustering. Section 6 shows the experiment results and section 7 provides conclusion and future work.

II. RELATED WORK

The comparative analysis of the n-gram, Suffix tree and Document Index Graph models are given by the authors P. Kathiria and H. Arolkar[10].

In the paper, different parameters like time complexity for storing and retrieving data, space complexity, data structure to be used to store data, identification of dynamic length of phrase are considered for the analysis.

Based on the analysis, DIG model can be effectively used to find common phrases among all the documents. DIG model uses graph structure, to store the whole corpus and does complete phrase matching in almost linear time. It uses less space compared to other document representation models Suffix Tree and n-gram.

The DIG model is effectively used by T. A. Nguyen Hoang and K. Hoang[12] to develop Incremental text clustering model for Vietnamese documents. Set of given documents are initially pre-processed and then used for incremental construction of a graph. While constructing graph incrementally, the common phrases among the documents are identified which can be used for the calculating hybrid document similarity. The hybrid similarity is calculated by combining single term and phrase similarity, which improves the cluster accuracy. The incremental DBSCAN algorithm is used for the clustering Vietnamese documents. For incremental document clustering the TF-IDF scheme is not suitable, so the TF-IG (Term frequency –Information Gain) is used for incremental feature selection process. The results obtained by ICGVi – Incremental Clustering based on Graph forVietnamese documents is compared with SHC - Similarity Histogram clustering and have shown that ICGVi model is more appropriate for incremental databases.

The concept of DIG model is used by S. George et. al. to propose a Word Sequence Graph (WSG), which can efficiently search and retrieve lengthy queries quickly[2]. The queries includes stop words which gives effective results. For faster search and retrieval, Neo4j graph database is used. Relational databases are nowadays replaced by NoSQL databases as per the growing demand of big databases. The Author J. Pokorny[13] discussed about strength and limitations of graph databases in their paper.

The Authors S. Chandrababu and D. R. Bastola [2] have done the comparative analysis of relational and graph database with HerbMicrobeDB. The HerbMicrobeDB database has 29 categories of herbal data constituents. Additionally, the new knowledge on biochemical, biological, phytochemical and its association is constantly being found. So to include such multiple and dynamic integrated elements the relational and graph databases are explored. Here the relationships of the entities are as important as entities itself. The results of Quantitative and Qualitative analysis of relational (MySQL) and graph database (Neo4j) is given by them. The quantitative analysis includes storage cost and execution time. They have clearly shown Neo4j is better for storing large scale networks due to its schema free structure. Execution time is evaluated against data insertion, search and for some built in functions. Few experiments are carried out which shows that Neo4j outperforms MySQL for each aspect of execution time. Qualitative analysis includes Ease of programming, Maturity level of support, security, Flexibility and data visualization. The graph database depicts the result of query in the form of table and graph both which enables the direct view of the properties of the node and relationship in the graph [16]. Graph databases stores huge data with high performance through flexible schema have low level of maturity [16] and security. The traversing between nodes and edges is done in linear time as compared to the exponential time in relational database. Health care data is heavily marked with relationships so, graph databases is an effective tool used by healthcare professionals to learn and manage new relationships. So, the authors M. Singh and K. Kaur[16] proposed a better way (algorithm) to migrate relational database (MySQL) into graph (Neo4j) database. The conjunctive queries fired on relational database are mapped to graph database query effectively. The comparison of time taken by some sample queries in RDBMS and Neo4j is done and result shows that there is significant improvement in access time in Neo4j graph database. The authors Baser Preeti and Jatinderkumar Saini [1] have done a comparative analysis of clustering techniques used for very huge datasets based on parameters like shape of clusters, Time complexity, ability to handle outliers. They have included various Partition methods, Hierarchical methods, density based methods and grid based methods. Depending on the domain of the dataset, characteristics of the dataset the best technique can be chosen.

III. DOCUMENT INDEX GRAPH MODEL

The Document Index Graph model makes use of graph data structure to store the documents and to find the common phrases. The DIG is a directed graph G= (V,E) where V : set of nodes {v1,v2,….Vn} where each node v signifies unique word appearing from the document corpus and E : set of edges {e1,e2,….en}, where each edge e depicts ordered pair of nodes (vi,vj) of particular sentence. Edge (vi,vj) is directed edge from vi->vj. The number of unique words in the document set is represented by the number of nodes in the graph. Multiple instances of the single word is counted as the one unique node and count is stored as the property of the node. If the word w1 and w2 are adjacent then the directed connectivity is created between the nodes which shows the proximity of the words which is depicted in figure -1. Each edge between two consecutive words (w1-> w2) of the graph stores the table - the document in which the consecutive term appears, the sentence number in that document and word place in that sentence as per the figure-1. This proximity of the words helps to find the common phrases between documents. The main advantage of the graph-based models is that they store the intrinsic structural information of theeach sentence of the original document.

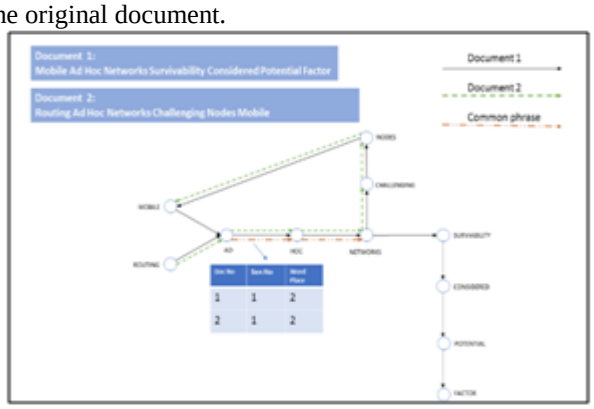

Figure 1 Document Index Grapph Model

IV. DATABASE PARADIGM

To store the Document index graph representation model it requires the help of any data store. DIG model can be stored in relational database management system or in graph database as well.

The relational model emphasizes on data as a pool of relations, relation is represented as table of values. Relational databases works with the structured data – fixed schema which can be easily fit into tables. Now a days, the size of the data is increasing, to extract the relationships from the data set, costly programming that is Join operation in terms of relational database systems is required more frequently.

The join operator does the Cartesian product – every possible combination, of any two table, this results in performance degradation [16],[2]. S. Chandrababunand D R Bastola have shown in their paper the relational algebra join operation performs at a Big-O exponential growth.[2]. The Graph database Stores all data as an "intelligent schema" that makes it easy to find and build any relationships as a graph with nodes and edges.

It is designed to proficiently store and work on highly connected data. Here, the relationship between the data (connectivity) is more valuable than the data itself. [16],[2],[6]

It has schema free environment, support for wide variety of languages through simple API and can handle massive amount of data speedily. It provides support for data safety and reliability through replication of data [11].

Data is stored efficiently by writing nodes and relationships close to each other that is called native graph storage. So, ultimately it requires less interaction with the hardware and provides faster performance to the queries. Every node is a pointer to its adjacent element provides index free adjacency which is efficient also [16]. Some of the graph databases get help of the other data storage such as a relational, columnar or other NoSQL database to store data that is known as Nonnative graph storage [11].

It includes one more storage layer which can lead to latent result. Global indexing concept is also used with graph databases to link nodes together, as the indexes add another layer to each traversal, which slows processing considerably.

So, where the relationships connecting the object are as important as the objects themselves then graph database is a perfect fit.

Other databases which use graph as a data model are FlockDB, Allegrograph, GraphDB, InfiniteGraph etc.

V. CLUSTERING BASED ON HYBRIDSIMILARITY ON DOCUMENT INDEX GRAPH MODEL

The Neo4j along with python 3.6 is used for implementation of DIG model and finding phrase similarity. Figure 2 shows the overall work flow.

For making document corpus a sample of 547 abstracts from the Scopus open access research papers are downloaded. The workflow of how the document similarity and clustering is done has been shown in figure-2.

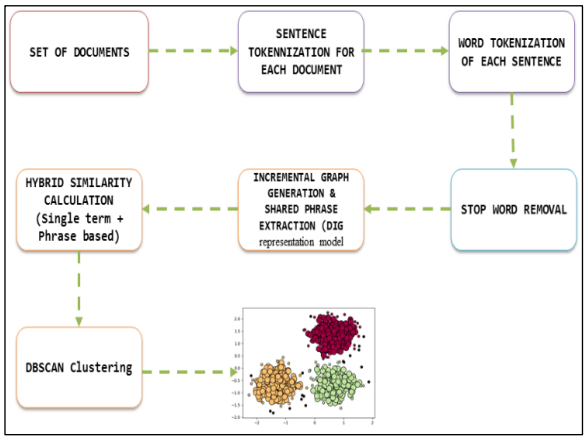

Figure 2 Work Flow of Document Clustering based on Phrase and Single term Similarity using Neo4j

The detailed working is as mentioned.

A. Pre-Processing

All documents passes through the pre-processing stage. The various tasks done by Pre-processing steps are as follows:

- Conversion of text into UTF-8 encoding
- Segment the text into sentences and each sentence into sequence of words
- Remove stop words and commkon words

B. Graph construction and identification of phrases

The Filtered text is passed for the graph construction. The DIG algorithm proposed in paper [7],[8] is used for incremental construction of the graph and finding common phrases among the given set of documents. The algorithm processes one preprocessed document at a time and builds the graph incrementally as shown in figure -3.

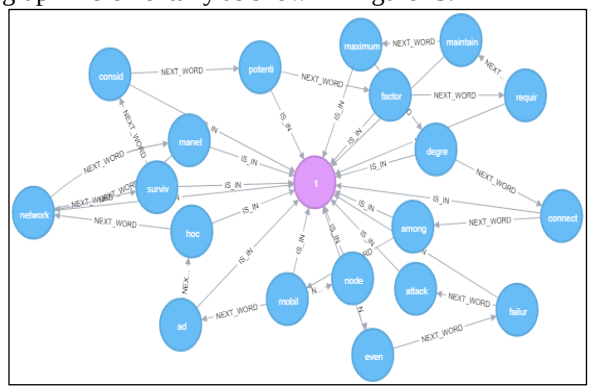

Figure 3 DIG Model In Neo4j

Three different types of vertices are created as the documents proceed.

i) words (w0,w1….wn) – for each unique word of the document corpus

ii) documents (d0,d1,…dn) – for each unique document of the corpus

iii) phrases (p0,p1,..pn) – for each unique phrase among all the documents The new words that appears in the document are inserted in the graph and linked with other nodes which reflects the adjacency of the words.

Three different types of edges are created as the documents proceed. The count of availability of particular word in the respective document is stored in the graph as an edge IS_IN between word (w0) and document(d0). Each edge NEXT_WORD between two consecutive words (w0-> w1) of the graph stores the information - the document in which the consecutive term appears, the sentence number in that document, word place in that sentence and the size of the sentence. This is displayed in figure 3. Each edge P_IS_IN between the phrase (p0) and respective document (d0) is recorded in the graph also stores the information – the frequency of the phrase in respective document and the total sentence size in which this phrase appears which is displayed in figure 4. This information is used for calculation of phrase similarity.

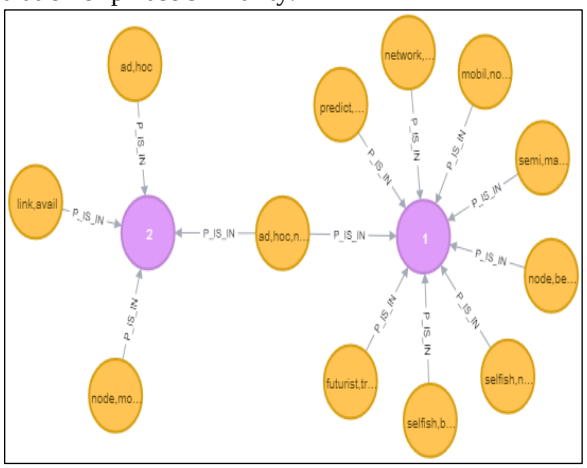

Figure 4 Common Phrases Between Document

C. Document similarity calculation

Based on identified common phrases across all the documents the phrase similarity calculation is done as per equation-1 using the four important factors:

- no. of common phrases (P)
- length of common phrase(l_i)
- frequency of phrase in each document(f_{1i}, f_{2i} - frequency of phrase i 1st and 2nd doc)
- Sentence length in which the phrase appears ($|s_{ij}|$: the length of the sentence j in the document d_i,

Now, the phrase similarity between document 1 and 2 is calculated using the equation1:

$$sim_{sp}(d_1, d_2) = \frac{\sqrt{\sum_{i=1}^{P} \left(\frac{l_i}{avg(|s_i|)}\right) \cdot (f_{1i} + f_{2i})^2}}{\sum_j |s_{1j}| + \sum_k |s_{2k}|} \quad (1)$$

Where, avg (|si|): the average length of sentences containing common phrase i. If the similarity between documents only depends on shared phrases then it may happen some documents do not share enough phrases so, only phrase based similarity is not sufficient. Therefore, along with phrase similarity, single term similarity is considered and hybrid

similarity is made.

The single term based similarity between document 1 and 2 is calculated as equation 2:

$$sim_{df}(d_1, d_2) = \frac{(d_1 \bullet d_2)}{\|d_1\| \cdot \|d_2\|} \quad (2)$$

The hybrid similarity is calculated as equation 3 for whole the document corpus.

$Sim(d1, d2) = \alpha * \text{sim}_{sp}(d_1,d_2) + (1-\alpha) * simdf(d_1,d_2)$ (3)

Where, α ∈[0 ,1] is a similarity blend factor. The value of α is selected as 0.3, as the phrase similarity is having more importance.

D. Clustering

To show the performance of the hybrid similarity measure, DBSCAN Clustering (Density based) is used.Clustering is the process of grouping a set of data points into groups of similar characteristics. The main objective of clustering is to increase intra-group similarity and inter-group dissimilarity. DBSCAN needs two parameters to create cluster ε (radius) and minpts (minimum number of points required to make cluster). A point is to be taken as core point if sufficient (minimum number of points defined) number of points comes in to radius are to be involved in cluster and rest are the outliers [15],[4]. For finding out the optimized value of ε the dataset is given to the algorithm prescribed in paper [5],[14]. As per the results from the implementation of the algorithm 0 .64 is chosen as the radius (ε).For clustering 7 is chosen as min pts.

VI. EXPERIMENT RESULTS AND DISCUSSIONS

To evaluate the performance of hybrid similarity, DBSCAN clustering model is used and experiment results are presented in this section. Experiment is carried out on 547 open access abstracts. The effectiveness of the Hybrid similarity versus single term similarity and phrase based similarity is tested using Silhouette score on the DBSCAN clustering. Silhouette score is used to measure the document is similar to its own cluster (cohesive) compare to other cluster (separation). It can also be used to measure the parting distance between the resulting clusters. The range of the silhouette value is between [-1,1]. If the Silhouette value is near to -1 it indicates that document is misclassified that is it is very close to adjacent cluster. If the Silhouette value is near to 0it indicates that document could be assign to another neighboring cluster and the document lies equally far away from both the cluster. If the Silhouette value is near to 1 it indicates that document is far away from adjacent cluster and very close to assigning cluster means the sample is well – clustered.The Silhouette Coefficient is calculated using equation 4.

$S(i) = (b(i) – a(i)) / \max \{a(i), b(i)\}$ (4)

Where,

a(i) = average distance of i^{th} document from all other documents in the same cluster

b(i) = smallest average distance of i^{th} document from all other documents in the nearest cluster.

The Silhouette score is mean of Silhouette Coefficient of all samples. For our dataset the results of Silhouette score are shown in the table-1.

Table 1 Silhouette Score

	Cosinesimilarity	Phrasebased	Hybrid
Silhouette Score	-0.04	0.178	0.243

The Silhouette score for Cosine similarity is close to -1 indicating sample is misclassified. If the Phrase similarity and Hybrid similarity is considered then the Silhouette score moves towards 1. As can be observed the Silhouette score of Hybrid similarity is better than Phrase similarity.

VII. CONCLUSION

The implementation of Document Index Graph model using neo4j finds out n-length phrase among all the documents in linear time with less time complexity. The shared Phrases are efficiently found with the help of Neo4j graph database.These shared phrases are considered for calculating Phrase wise similarity. Based on Phrase similarity and single term similarity, the Hybrid similarity is calculated. Hybrid similarity is used by DBSCAN density based calustering for grouping the documents. The Silhouette score of all the samples is calculated for Cosine, Phrase and Hybrid similarity on the decided clusters. The Silhouette score indicates that cluster made by Hybrid similarity have better cohesion as compared to Cosine similarity and Phrase similarity clusters which further states that it is better to use hybrid similarity instead of Phrase similarity and Cosine similarity.

REFERENCES

1. Baser, Preeti, and Jatinderkumar R. Saini. "A comparative analysis of various clustering techniques used for very large datasets." International Journal of Computer Science & Communication Networks 3.5 (2013): 271.
2. Chandrababu, Suganya, and Dhundy R. Bastola. “Comparative Analysis of Graph and Relational Databases Using HerbMicrobeDB.” Proceedings - 2018 IEEE International Conference on Healthcare Informatics Workshops, ICHI-W 2018, IEEE, 2018, pp. 19–28, doi:10.1109/ICHI-W.2018.00010.
3. Chim, Hung, and Xiaotie Deng. “Efficient Phrase-Based Document Similarity for Clustering.” IEEE Transactions on Knowledge and Data Engineering, vol. 20, no. 9, 2008, pp. 1217–29, doi:10.1109/TKDE.2008.50.
4. Ester, Martin, and RüdigerWittmann. Incremental Generalization for Mining in a Data Warehousing Environment. 2006, pp. 135–49, doi:10.1007/bfb0100982.
5. Gaonkar, Manisha Naik, and KedarSawant. "AutoEpsDBSCAN: DBSCAN with Eps automatic for large dataset." International Journal on Advanced Computer Theory and Engineering 2.2 (2013): 11-16.
6. George, Soumya, et al. A Novel Sequence Graph Representation for Searching and Retrieving Sequences of Long Text in the Domain of Information Retrieval. Vol. 2, no. 5, 2017, pp. 108–13.
7. Hammouda, K. M., and M. S. Kamel. “Phrase-Based Document Similarity Based on an Index Graph Model.” 2002 IEEE International Conference on Data Mining, 2002. Proceedings., 2002, pp. 203–10, doi:10.1109/ICDM.2002.1183904.
8. Hammouda, Khaled M., and Mohamed S. Kamel. “Document Similarity Using a Phrase Indexing Graph Model.” Knowledge and Information Systems, vol. 6, no. 6, 2004, pp. 710–27, doi:10.1007/s10115-003-0118-5.
9. Hussein, Ashraf S. “Visualizing Document Similarity Using N-Grams and Latent Semantic Analysis.” Proceedings of 2016 SAI Computing Conference, SAI 2016, 2016, pp. 269–79, doi:10.1109/SAI.2016.7555994.
10. Kathiria, Preeti, and HarshalArolkar. “Study of Different Document Representation Models for Finding Phrase-Based Similarity.” Smart Innovation, Systems and Technologies, vol. 106, 2019, pp. 455–64, doi:10.1007/978-981-13-1742-2_45.
11. Makris, Antonios, et al. “A Classification of NoSQL Data Stores Based on Key Design Characteristics.” Procedia Computer Science, vol. 97, The Author(s), 2016, pp. 94–103, doi:10.1016/j.procs.2016.08.284.
12. Nguyen Hoang, Tu Anh, and Kiem Hoang. “Efficient Approach for Incremental Vietnamese Document Clustering.” International Conference on Information and Knowledge Management, Proceedings, 2009, pp. 47–54, doi:10.1145/1651587.1651599.
13. Pokorný, Jaroslav, et al. Graph Databases : Their Power and Limitations Jaroslav Pokorný To Cite This Version : HAL Id : Hal-01444505 Graph Databases : Their Power and Limitations. 2017, pp. 58–69.
14. Rahmah, Nadia, and ImasSukaesihSitanggang. "Determination of optimal epsilon (eps) value on dbscan algorithm to clustering data on peatland hotspots in sumatra." IOP Conference Series: Earth and Environmental Science. Vol. 31. No. 1. IOP Publishing, 2016.
15. Sarmah, Sauravjoyti, and Dhruba K. Bhattacharyya. "An effective technique for clustering incremental gene expression data." IJCSI International Journal of Computer Science Issues 7.3 (2010): 31-41.
16. Singh, Manpreet, and Karamjit Kaur. “SQL2Neo: Moving Health-Care Data from Relational to Graph Databases.” Souvenir of the 2015 IEEE International Advance Computing Conference, IACC 2015, vol. 0, IEEE, 2015, pp. 721–25, doi:10.1109/IADCC.2015.7154801.
17. https://neo4j.com/

AUTHORS PROFILE

Prof. Preeti Kathiria is working as an Assistant Professor of Computer Science and Engineering Department since 2004. She has a professional experience of over 17 years which includes both Academia and Industry. She received her MCA from Gujarat University in 2003. Currently, she is pursuing her PhD from Gujarat Technological University in the area of Text Mining. She has published several research papers in international journals and conferences. She works in the area of Machine Learning, Data Analytics and Natural Language Processing. She has conducted ISTE approved STTP in the field of Web Services using PHP.

Dr. Harshal A. Arolkar is Professor and HoD, at GLS University. An ardent practitioner and teacher he possesses more than 20 years of teaching experience in Computer Science. He has published 3 books on computer science published by Wiley India and Dreamtech Press. He has also publishedand presented more than 30 research papers in international and national conferences and journals. His area of interests includes WSN, Big Data and NLP.