Detecting and Classifying Toxic Language in Twitter using Machine Learning

Nischal Lakhotia, Omprakash Harod, T. Manoranjitham

Abstract: ***Today international on-line content material has turned out to be a first-rate part due to growth in the use of net. Individuals of various societies and instructive foundation can speak through this platform. Therefore, for automatic detection of poisonous content, we need to distinguish between hate speech and offensive language. Here a way to robotically stumble on and classify tweets on Twitter into 3 commands: hateful, offensive and easy is proposed. We do not forget n-grams as functions and by way of passing their time period frequency-inverse document frequency (TFIDF) values to numerous system gaining knowledge of fashions using Twitter dataset, we perform comparative evaluation of the models. We work towards classification and comparison of different classifiers using the combination of best feature from each type of feature extraction and determining which model works best for the purpose of classification of tweets into hate-speech, offensive language or neither.***

Keywords: ***Toxic Language, hate speech, offensive language, n-gram, tf-idf, machine learning.***

I. INTRODUCTION

The increase in the development of social media for example, Twitter and other such networking platforms has changed correspondence and communication, but on the other hand is progressively abused for the proliferation of detest speech and the association of abhor based exercises. The term 'hate speech' was officially characterized as 'any correspondence that decries an individual or a group based on certain attributes (to be alluded to as kinds of abhor or loathe classes, for example, race, sex, color, ethnicity, religion, sexual orientation, nationality along with other characteristics). In the UK, there has been noteworthy increment of detest discourse towards the transient and Muslim people group following ongoing occasions including leaving the EU, the Manchester and the London assaults[1]. What's more, related wrongdoings dependent on strict convictions, ethnicity, sexual direction or sex, as 80% of respondents have experienced loathe discourse on the web and 40% felt assaulted or undermined. Measurements additionally show that in the US, abhor discourse and wrongdoing is on the ascent since the Trump's political election[2]. The desperation of this issue has been progressively perceived, as a scope of worldwide activities have been propelled towards the capability of the issues and the improvement of countermeasures. There has been a blast in this issue in the most recent decade and henceforth distinguishing or expelling such substance physically from the web is a tedious assignment. So conceiving a computerized model that can distinguish harmful substance on the web is required. In this report, a machine studying model which can differ among those two factors of harmful language is proposed. We come across hate speech and offensive textual content on Twitter platform. We educate our classifier version by the use of n-gram and Term Frequency-Inverse Document Frequency (TFIDF) and compare them for metric rankings, by way of the usage of publicly available Twitter datasets. We perform comparative analysis of the results obtained using linear regression (LR), Random forest (RF), Naive Bayes (NB) and Support vector machine (SVM) as classifier models.

Revised Manuscript Received on June 10, 2020.
* Correspondence Author

Nischal Lakhotia*, Department of Computer Science and Engineering, SRMIST, Kattankulathur, India. E-mail: nishlakhs@gmail.com

Omprakash Harod, Department of Computer Science and Engineering, SRMIST, Kattankulathur, India. E-mail: oh9438@srmist.edu.in

T. Manoranjitham, Department of Computer Science and Engineering, SRMIST, Kattankulathur, India. E-mail: manorant@srmist.edu.in

II. EXISTING WORK

Existing studies on hate speech detection have primarily reported their results using micro-average Precision, Recall and F1 measure. The issue with this is in a lopsided dataset where occasions of one class (to be known as the 'predominant class') essentially out-number others (to be called 'minority classes'), miniaturized scale averaging can cover the genuine exhibition on minority classes. In system architecture, we can see the steps involved in the processing in the system[3].

A. Disadvantage of existing system

Existing studies on hate speech detection have primarily reported their results using micro-average Precision, Recall and F1. The problem with this is that in an unbalanced dataset where instances of one class (to be called the 'dominant class') significantly out-number others (to be called 'minority classes'), micro-averaging can mask the real performance on minority classes.

III. PROPOSED SYSTEM

All datasets are significantly biased towards non-hate, as hate Tweets account between only 5.8% (DT) and 31.6% (WZ). When we inspect specific types of hate, some can be even scarcer, such as 'racism' and as mentioned before, the extreme case of 'both'.

This has two implications. Firstly, an evaluation measure such as the micro F1 that looks at a system's performance on the entire dataset regardless of class difference can be biased to the system's ability of detecting 'non-hate'[4]. In other words, a hypothetical system that achieves almost perfect F1 in identifying 'racism' tweets can still be overshadowed by its poor F1 in identifying 'non-hate', and vice versa.

Secondly, compared to non-hate, the training data for hate tweets are very scarce. This may not be an issue that is easy to address as it seems, since the datasets are collected from Twitter and reflect the real nature of data imbalance in this domain. Thus to annotate more training data for hateful content we will almost certainly have to spend significantly more effort annotating non-hate.

B. Advantages

The models that are trained after the extraction of N-gram highlights from content give better outcomes. This can be inferred from the survey on the related work done in this field. Additionally, the TFIDF approach on the bag-of-words includes likewise and show reassuring outcomes. We chose to separate n-grams and weight them according to their TFIDF esteems with the end goal that these highlights are then taken care of to a machine learning calculation with the end goal of classification. Our motive is to classify them into three groups: hateful, offensive and clean from the given set of tweets. Another motive is to compare the performance of different classifiers using different combinations of features and determining which features are best for the purpose of toxic language detection and which classifier performs the best in these situations. In proposed framework, a framework of the newly developed system is proposed.

IV. SYSTEM ARCHITECTURE

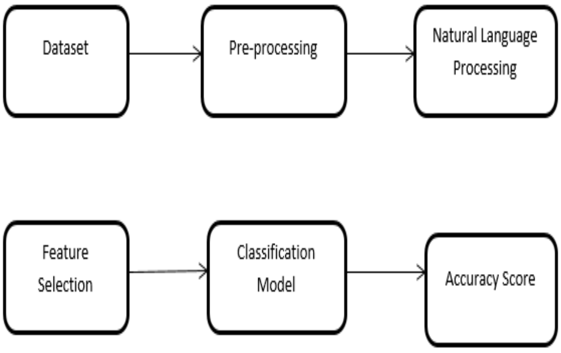


Fig. 1. System Architecture

V. IMPLEMENTATION

The two most important steps while implementation are data pre-processing and feature extraction. Fig. 2 To Fig. 7 show the testing results of various modules used.

A. Data Pre-processing

It is the step followed before extracting features from your data. It involves manipulating the text data in a way which is useful for text processing.

- **Lower-casing**:

This abstains from having different duplicates of similar words.

- **Remove punctuation:**

It doesn't include any additional data while treating content information. In this way expelling all examples of it will assist us with lessening the size of the preparation information. Remove Stop words: It's better to remove the commonly occurring words like they are quite useless for text processing.

- **Removal of common words:**

We can likewise expel generally happening words from our content information. Finding the most frequently occurring words and then take a call to either remove or retain them.

- **Rare word removal:**

We expel the once in a while happening words on the grounds that the relationship among them and different words is ruled by commotion.

- **Tokenization:**

Dividing the content into a grouping of words or sentences.

- **Stemming:**

Stemming alludes to the evacuation of does the trick, such as "ing", "ly", "s", and so on by a straightforward principle based methodology. Porter Stemmer from the NLTK library can be utilized for Stemming.

B. Feature extraction

It is the science and specialty of extricating more data from existing information. You are not including any new information here, yet you are really making the information you as of now have progressively helpful. While classifying twitter text data some features of tweets that can be used are:

- **TFIDF weights for n-grams:**

TF figures the great number of times the word shows up in the content. IDF processes the overall significance of this word which relies upon what number of writings the word can be found.

- **Sentiment Analysis:**

Using Vader we decide the notion score of every content. Vader returns four extremity scores – positive, negative, unbiased and compound. Vader not just tells about the Positivity and Negativity score yet in addition educates us concerning how positive or negative a notion is.

- **Doc2Vec Columns:**

Word2Vec essentially changes over a word into a vector. Doc2Vec not exclusively does that, yet in addition totals all the words in a sentence into a vector. To do that, it essentially treats a sentence mark as a unique word, and does some procedure on that uncommon word. Thus, that extraordinary word is a name for a sentence.

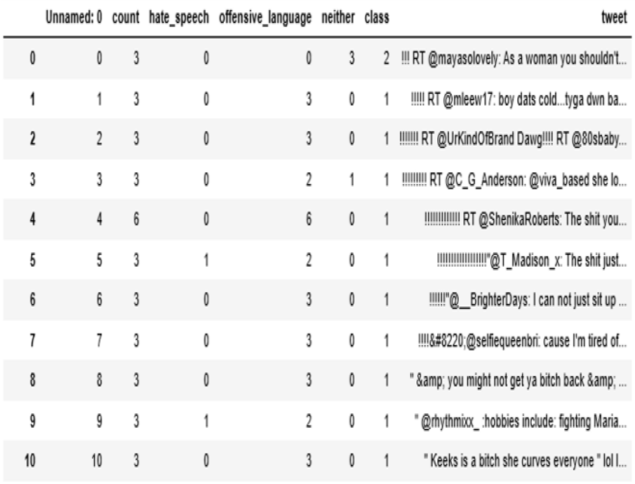
Each text can also be transformed into numerical vectors using the word vectors.

- **Other enhancement twitter specific features:**

a. Count of syllables present in the given tweet
b. Length of the tweet text
c. Count of words in a tweet
d. Count of unique words in a tweet
e. Average number of syllables in a tweet given figure we can understand the various feature engineering steps involved in the process.

Retrieval Number: E9714069520/2020©BEIESP
DOI: 10.35940/ijeat.E9714.069520
Journal Website: www.ijeat.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication



	Unnamed: 0	count	hate_speech	offensive_language	neither	class	tweet
0	0	3	0	0	3	2	!!! RT @mayasolovely: As a woman you shouldn't...
1	1	3	0	3	0	1	!!!!! RT @mleew17: boy dats cold...tyga dwn ba...
2	2	3	0	3	0	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...
3	3	3	0	2	1	1	!!!!!!!!! RT @C_G_Anderson: @viva_based she lo...
4	4	6	0	6	0	1	!!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...
5	5	3	1	2	0	1	!!!!!!!!!!!!!!!!!!"@T_Madison_x: The shit just...
6	6	3	0	3	0	1	!!!!!!"@__BrighterDays: I can not just sit up ...
7	7	3	0	3	0	1	!!!!&#8220;@selfiequeenbri: cause I'm tired of...
8	8	3	0	3	0	1	" &amp; you might not get ya bitch back &amp; ...
9	9	3	1	2	0	1	" @rhythmixx_ :hobbies include: fighting Maria...
10	10	3	0	3	0	1	" Keeks is a bitch she curves everyone " lol I...

Figure 2: Dataset overview

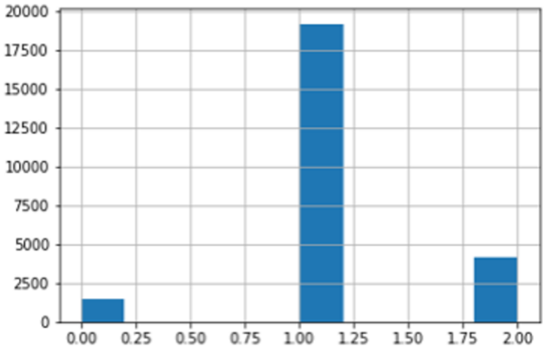
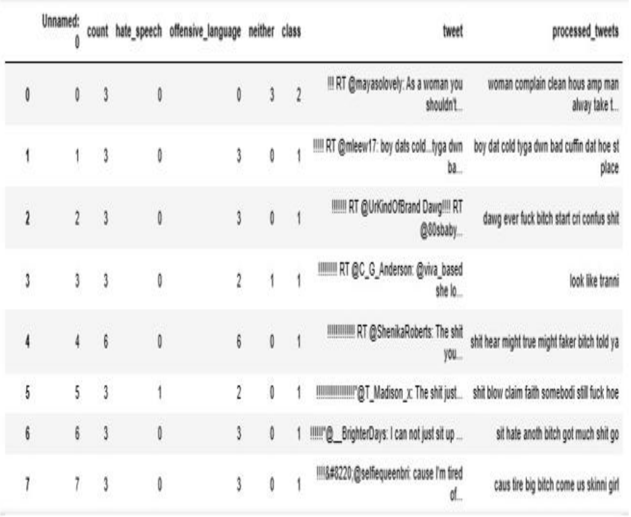


Fig. 3.Histogram representing count of each class



	Unnamed: 0	count	hate_speech	offensive_language	neither	class	tweet	processed_tweets
0	0	3	0	0	3	2	!!! RT @mayasolovely: As a woman you shouldn't...	woman complain clean hous amp man alway take t...
1	1	3	0	3	0	1	!!!!! RT @mleew17: boy dats cold...tyga dwn ba...	boy dat cold tyga dwn bad cuffin dat hoe st place
2	2	3	0	3	0	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...	dawg ever fuck bitch start cri confus shit
3	3	3	0	2	1	1	!!!!!!!!! RT @C_G_Anderson: @viva_based she lo...	look like tranni
4	4	6	0	6	0	1	!!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...	shit hear might true might faker bitch told ya
5	5	3	1	2	0	1	!!!!!!!!!!!!!!!!!!"@T_Madison_x: The shit just...	shit blow claim faith somebodi still fuck hoe
6	6	3	0	3	0	1	!!!!!!"@__BrighterDays: I can not just sit up ...	sit hate anoth bitch got much shit go
7	7	3	0	3	0	1	!!!!&#8220;@selfiequeenbri: cause I'm tired of...	caus tire big bitch come us skinni girl

Fig. 4.Processed tweets

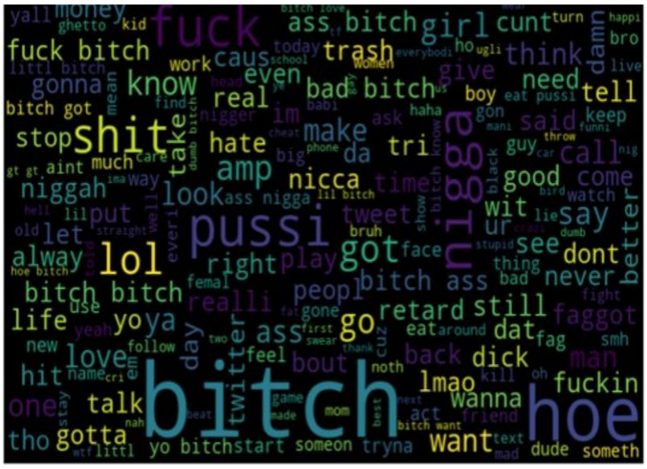
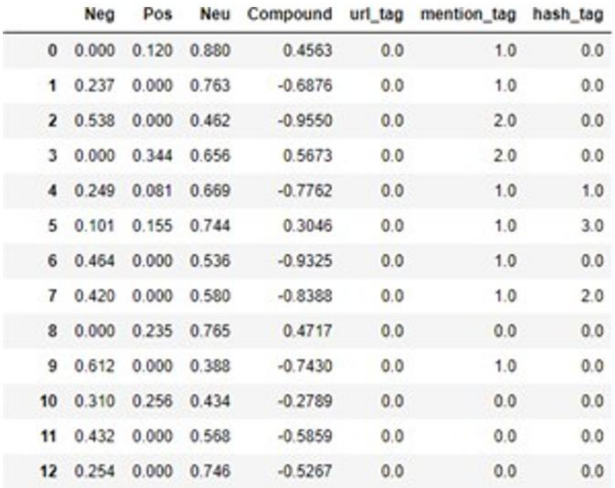


Figure 5: Pictorial representation of most important terms



	Neg	Pos	Neu	Compound	url_tag	mention_tag	hash_tag
0	0.000	0.120	0.880	0.4563	0.0	1.0	0.0
1	0.237	0.000	0.763	-0.6876	0.0	1.0	0.0
2	0.538	0.000	0.462	-0.9550	0.0	2.0	0.0
3	0.000	0.344	0.656	0.5673	0.0	2.0	0.0
4	0.249	0.081	0.669	-0.7762	0.0	1.0	1.0
5	0.101	0.155	0.744	0.3046	0.0	1.0	3.0
6	0.464	0.000	0.536	-0.9325	0.0	1.0	0.0
7	0.420	0.000	0.580	-0.8388	0.0	1.0	2.0
8	0.000	0.235	0.765	0.4717	0.0	0.0	0.0
9	0.612	0.000	0.388	-0.7430	0.0	1.0	0.0
10	0.310	0.256	0.434	-0.2789	0.0	0.0	0.0
11	0.432	0.000	0.568	-0.5859	0.0	0.0	0.0
12	0.254	0.000	0.746	-0.5267	0.0	0.0	0.0

Fig 6.Feature sentiment scores and and count of urls, has tags and mentions

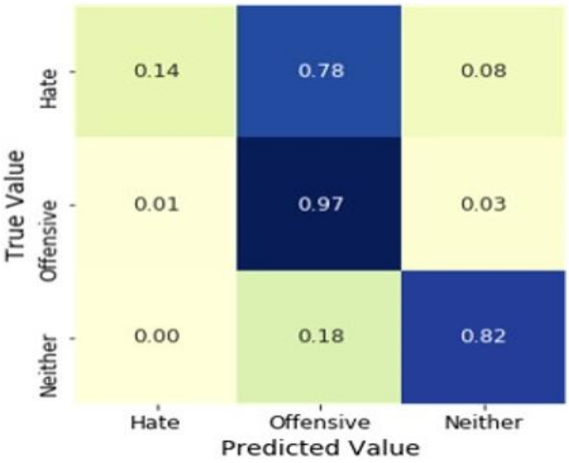
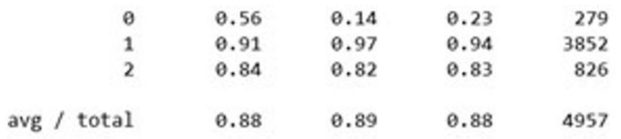


Fig 7.Confusion matrix

VI. RESULT

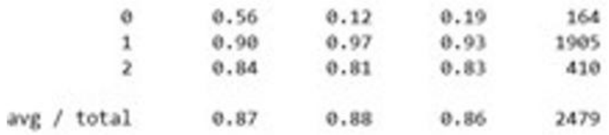
Here, Fig. 8 to Fig. 11 show the evaluation metrics of various classifier models used. Fig. 12 shows the final comparative analysis of the classifier models used.



```
           0       0.56      0.14      0.23       279
           1       0.91      0.97      0.94      3852
           2       0.84      0.82      0.83       826

avg / total       0.88      0.89      0.88      4957
```

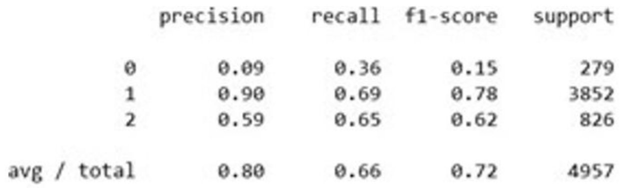
Fig. 8.Linear Regression Classification Model



```
           0       0.56      0.12      0.19       164
           1       0.90      0.97      0.93      1905
           2       0.84      0.81      0.83       410

avg / total       0.87      0.88      0.86      2479
```

Fig. 9.Support Vector Machine Classification Model Evaluation Metrics.

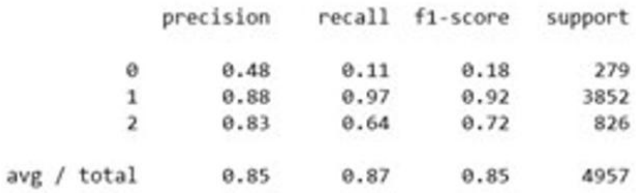


```
             precision    recall  f1-score   support

           0       0.09      0.36      0.15       279
           1       0.90      0.69      0.78      3852
           2       0.59      0.65      0.62       826

avg / total       0.80      0.66      0.72      4957
```

Fig. 10.Naive Bayes Classification Model Evaluation Metrics



```
             precision    recall  f1-score   support

           0       0.48      0.11      0.18       279
           1       0.88      0.97      0.92      3852
           2       0.83      0.64      0.72       826

avg / total       0.85      0.87      0.85      4957
```

Fig. 11.Random Forest Classification Model Evaluation Metrics

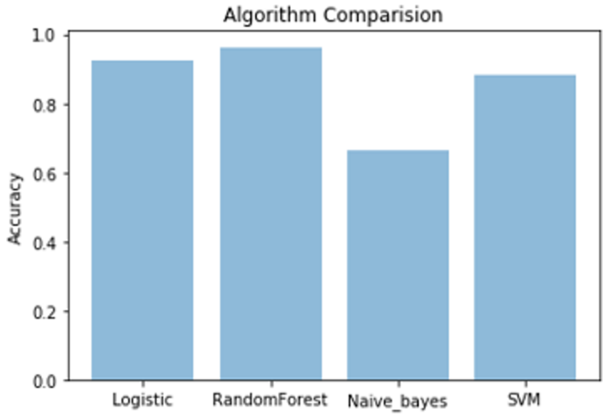


Fig. 12. Final Result: Comparative Analysis of Classifiers

VII. CONCLUSION AND FUTURE WORK

For the detection of foul language on Twitter, we have proposed a system which requires machine learning which uses n-gram features weighted with TFIDF values. A comparative analysis of Random Forest, Linear Regression Naive Bayes and Support Vector Machines on various sets of feature values is conducted. To measure the classification success, a set of statistical metrics is used. For their computation an amount of basic measures has to be taken into account. All measures are calculated per class, so that the present three-class classification problem is treated as three binary classification problems with positive and negative samples. For every class, true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN) are counted. Overall Logistic Regression and the Random Forest are the best performing classifiers. Our Experiments results are shown in the graph. The accuracy percentage of logistic regression 89% and Random Forest is given 96% and then SVM accuracy is 88% and the last Naive Bayes gives low accuracy rate that is 66%. The latter however results in a higher recall than the first one. Especially in hate speech detection applications in social media, a high recall is preferable over a high precision. If the recall is high and the precision is in turn low, further measures can be taken to extract the actual hate speech samples from the samples classified as hate speech. The outcomes indicated that Random Forest performs better with the ideal n-gram range 1 to 3 for the L2 normalization of TFIDF. The Random forest calculation will perform better with a larger number of training information, yet speed during testing and application will endure. Application of more pre-processing techniques would likewise help. All the insights be provided in a graph and table format. Upon evaluating the model on test information, we accomplished 96% precision. Practically 4.8% of the contemptuous tweets were misclassified and were named offensive. By obtaining more instances of offensive language which doesn’t contain scornful words, this issue can be solved by improving the review for the offensive class and precision for the hateful class.

REFERENCES:

1. Al-Hassan, A. and Al-Dossari, H. (2019). “Detection of hate speech in social networks: A survey on multilingual corpus. 83–100.
2. Al-makhadmeh, Z. and Tolba, A. (2019). “Automatic hate speech detection using killer natural language processing optimizing ensemble deep learning approach.” Computing, 102, 501 – 522.
3. Bansal, P. (2019). “Detection of offensive youtube comments, a performance comparison of deep learning approaches.
4. Greevy, E. and Smeaton, A. F. (2004). “Classifying racist texts using a support vector machine.” SIGIR ’04.
5. Hinduja, S. and Patchin, J. W. (2010). “Bullying, cyberbullying, and suicide.” Archives of Suicide Research, 14, 206 – 221.
6. Oriola, O. and Kotzé, E. (2020). “Evaluating machine learning techniques for detecting offensive and hate speech in south african tweets.” IEEE Access, 8, 21496–21509.
7. Raufi, B. and Xhaferri, I. (2018). “Application of machine learning techniques for hate speech detection in mobile applications. 1–4.
8. Sajjad, M., Zulifqar, F., Khan, M. U. G., and Azeem, M. (2019). “Hate speech detection using fusion approach.” 2019 International Conference on Applied and Engineering Mathematics (ICAEM). 251–255.
9. Santosh, T. and Aravind, K. (2019). “Hate speech detection in hindi-english code-mixed social media text. 310–313.
10. Warner, W. and Hirschberg, J. (2012). “Detecting hate speech on the world wide web.” Proceedings of the Second Workshop on Language in Social Media, Montréal, Canada. Association for Computational Linguistics, 19–26.

AUTHORS PROFILE

Nischal Lakhotia is pursuing his Bachelor of Technology in the department of Computer Science and Engineering at SRMIST (formerly known as SRM University).He has done his internship from National Informatics Centre in Computer Networking and Cyber Security.

He has also worked in Web Development. Currently, he is placed in a well-known IT company as associate programmer.

Omprakash Harod is pursuing his Bachelor of Technology in the department of Computer Science and Engineering at SRMIST (formerly known as SRM University). Currently, he is placed in a well-known Networking Company (Nokia Network) as an assistant programmer.

T. Manoranjitham , Assistant Professor (S.G) in the Department of Computer Science and Engineering at SRM Institute of Science and Technology. She has many years of teaching and research experience and has published more than 10 research papers. She has done B.E in Electronics and Communication Engineering in 1991 and done M.E in Computer Science and Engineering in 2002. Her research interest includes Wireless sensor networks, Network security, Software Defined Network, Internet of Things, Cloud Computing, and Mobile Ad Hoc Network. She always teaches the students in a very detailed and nice manner and she is very knowledgeable.