**Model Overview**

**1.Elements of a tweet & Used inputs (∑)**

* Normal text (may have a meaning or may not)
* # trend

where trend is a variable which can take values of trending incidents

* @ user\_name

where user\_name can take values of registered users with twitter

* url / video / image

These all can be included as url as twitter first stores them as url and post the link in the tweet.

* .

. is used when we address someone specifically.

* RT

Is used when retweet someone's tweet.

English: words with minimum value length

Normal English language + Unicode characters

* ENGLISH
* є

**2.Rules for valid tweet**

1. only above inputs are allowed

2.tweets comprising only text will be rejected to differentiate between normal text

3. tweets can n't be end with dependent inputs, @ and #

4.tweets ending with @U\_NAME are rejected (to cross check on tweets with only @U\_NAME)

hence the reversal property gets hampered in case of tweets starting with @U\_NAME

5.tweet with only urls(context derived from vid/img/audio) cann't be used through the model

6.Transition from q3 to q1 with input RT is discarded as

Q3\*RT-> q1

Won’t provide any meaning like

Eg-

@apple RT @aplusk: Sending love & light to everyone @Apple & the entire Jobs family.Today we lost a Giant who will be missed even by those who ...

Doesn’t give any meaning as it tries to retweet and converse at the same time .

Our model assumes tweets are not of this type.

**Closure Properties**

Qf = [ T + ENG.T +

(@+ENG.@+RT.@+ENG.RT.@ ). (U\_NAME.@) \*.U\_NAME. (ENG+T) +

[ # + ENG.# + (@+ENG.@+RT.@+ENG.RT.@ ). (U\_NAME.@)\*.U\_NAME.#].TREND ].

[ (RT.@+@)). (U\_NAME.@) \*.U\_NAME.(ENG+T) +

(ENG+T) +

[(RT.@+@)). (U\_NAME.@) \*.U\_NAME.# + # ].TREND ]\*

A language L< ∑\* is regular if there exists a DFA such that L=L(M).

To define this as a valid RE and it is getting accepted by the proposed DFA,

We take two languages

RE1=  @.U\_NAME.ENG.[ ENG + #. TREND + T ] \*

RE2= @.U\_NAME.ENG.[ ENG + #. TREND] \*

------------------------------------------------------------------------------------------------

both these language are defined by the proposed DFA

and proved the following properties.

1.concatenation

L1.L2

set of strings with beginning in L1 and continuation in L2

eg- L1={ **@sbharti1984** We must teach him a lesson <http://bit.ly/2ego> **,**......}belongs to RE1

L2={[~~@~~**SudheenKulkarni**](https://twitter.com/SudheenKulkarni) Is Pakistan condemned terrorist attack on SSB convoy in [~~#~~**Zakura**](https://twitter.com/hashtag/Zakura?src=hash),......}belongs to RE2

and in this case

L=L1.L2= {**@sbharti1984 We must teach him a lesson ttp://bit.ly/2ego** [**SudheenKulkarni**](https://twitter.com/SudheenKulkarni) Is Pakistan condemned terrorist attack on SSB convoy in [~~#~~**Zakura**](https://twitter.com/hashtag/Zakura?src=hash),......}

Now RE(L1. L2)= @.U\_NAME.ENG.[ ENG + #. TREND + T ] \*. @.U\_NAME.ENG.[ ENG + #. TREND] \*

Here this is a RE.

----------------------------------------------------------------------------------------------------------

2.Kleene closure:

L\*

set of repeated concatenation of a string in L1

RE1\* is f the form L1= [ @.U\_NAME.ENG.[ ENG + #. TREND + T ] \* ]\*

L1\* ={**@sbharti1984 We must teach him a lesson ttp://bit.ly/2ego ,**

**@sbharti1984 We must teach him a lesson ttp://bit.ly/2ego @sbharti1984 We must teach him a lesson ttp://bit.ly/2ego ,**...}

RE(L1\*) is also a regular expression

-----------------------------------------------------------------------------------------

3.union:

L1∪L2=L1+L2

set of strings in L1 or in L2 are individually satisfied by FSM, hence will L1+L2.

eg- L1={@sbharti1984 We must teach him a lesson ttp://bit.ly/2ego,......}belongs to RE1

L2={@SudheenKulkarni Is Pakistan condemned terrorist attack on SSB convoy in #Zakura,......}belongs to RE2

and in this case

L=L1+L2= {@sbharti1984 We must teach him a lesson ttp://bit.ly/2ego, @SudheenKulkarni Is Pakistan condemned terrorist attack on SSB convoy in #Zakura,......}

RE(L)=@.U\_NAME.ENG.[ ENG + #. TREND + T ] \*

which is also a RE

-----------------------------------------------------------------------------------------

4.complementation:

Σ∗−L

set of all possible strings that are not in L

let RE= RE(Qf )

~RE=the not valid tweets

This condition get satisfied as we change the final states to non final and non final states to final.

The complement RE will be satisfied by the complement FSM.hence this pproperty also satisfies.

------------------------------------------------------------------------------------------

5.difference:

L1−L2

taking

RE1= @.U\_NAME.ENG.[ ENG + #. TREND + T] \*

RE2= @.U\_NAME.ENG.[ ENG + #. TREND] \*

eg- L1={ @sbharti1984 We must teach him a lesson http://bit.ly/2ego,......}belongs to RE1

L2={@SudheenKulkarni Is Pakistan condemned terrorist attack on SSB convoy in #Zakura,......}belongs to RE2

and in this case

set of strings which are in L1 but not in L2

L1-L2={ @sbharti1984 We must teach him a lesson http://bit.ly/2ego ,......}

hence RE(L1-L2)= @.U\_NAME.ENG. [ ENG\*.( #. TREND)\*.T+ +

ENG\*.T+ .( #. TREND)\*+

T+.ENG\*.( #. TREND)\* ] \*

which is also a RE

----------------------------------------------------------------------------------------

6.intersection:

set of strings in both L1 and L2

L1={ @sbharti1984 We must teach him a lesson http://bit.ly/2ego,......}belongs to RE1

L2={@SudheenKulkarni Is Pakistan condemned terrorist attack on SSB convoy in #Zakura,......}belongs to RE2

and in this case

L=L1∩L2= {@SudheenKulkarni Is Pakistan condemned terrorist attack on SSB convoy in #Zakura , .....}

Now RE(L1∩L2) =@.U\_NAME.ENG. [ ENG + #. TREND]\*

which is also a RE.

------------------------------------------------------------------------------------------------

7.reversal:

(L)r

here the taken REs won't satisfy reversal rule due to the definatio for proposed language.

Hence we take another RE to prove this property which is in accordance with rules with which we define tweets

let RE(L)=ENG\*.(#.TREND)\*

L= {Ashwin has got Williamson's number in this series! [~~#~~**INDvNZ**](https://twitter.com/hashtag/INDvNZ?src=hash)**,.....}**

RE((L)r) =(#.TREND)\*.ENG\*

eg- [~~#~~**Dussehra**](https://twitter.com/hashtag/Dussehra?src=hash) gives us a message that GOOD will ALWAYS triumph over EVIL in the end. Happy Vijayadashami

hence reversal of the proporsed RE is also a RE

**Abstract**

In this era of internet, social media grows exponentially and has become the primary medium to interact with others on upcoming and ongoing events. Among the plethora of social media, Twitter has emerged as the favorite destination for web surfers in recent years. Also, the Twitter platform has acted as a boon for the researchers of Natural Language Processing (NLP) due to its huge sample space and unique platform features like "#trending" which provides a gateway to perform tasks like POS tagging, parsing, sentiment analysis, etc. on human behavioral aspects. Sarcasm is one of the most challenging components in analyzing sentiment in text. In recent times, many researchers have shown their interest towards sarcasm detection in textual data and proposed their sarcasm detection system. However, there are some shortcomings in the existing system among which the inclusion of noise sample space can be considered as a critical one. In this paper, we proposed a finite state machine (FSM) model for Twitter language which will recognize valid tweets over other social media text and hence will play a crucial role in choosing credible input data. It filters out null value tweets like one or two-word tweets; tweets that contain only handles without a text message; tweets that contain URL of a video, image, etc. before sarcasm detection. These tweets don't provide a proper value for sentiment analysis. Finally, we deployed a sarcasm detector using hashtag dictionary, having words other than “#sarcasm” and “#sarcastic” to detect sarcasm from filtered tweets and also shown that after elimination of null value tweets, the existing system has achieved better accuracy.

**Conclusion & future scope**

In this paper, we exploited the tweet property and designed a finite state machine model for tweet identification to distinguish tweet with other social media text. The proposed system attains around 94.67% accuracy for tweet identification. Further, we eliminated the null value tweet to make sentiment analyzer more efficiently work and proposed a sarcasm detection system for tweets using negation word and hashtag dictionary. Finally, we made a comparison with existing sarcasm detector result with the result after elimination of null value tweets in the test set and observed that after elimination accuracy increased significantly in the existing system. In future, we aim to extend this experiment to other platforms and to build a social media classifier.

**Introduction**

Online social media, where the exchange of ideas takes place over the internet and in virtual communities between groups or individuals. People from all regions, all religion, and all generations have come to embrace the changes that have been brought out by the social network. People are forming online communities that allow them to get support, education, and even promote and sell products. The outcomings of social media have been interesting to observe with the participation of millions of people around the globe over the past few years. Events have been reshaped by the proper use of these online media.With time these social media have gained astounding worldwide growth and popularity and produces an enormous amount of data which has led researchers globally to turn their faces towards this for sentiment analysis through openly available data. And here comes the problem of selecting the proper sample data for the experiment where existing systems fail. We proposed a model through with the aim of discarding these noises.

In this paper, we developed an FSM model, which will classify Twitter language over other social media language such as posts and comments on Facebook, blogs and posts on LinkedIn, chat messages on WhatsApp, etc. The proposed FSM model accepts a valid tweet which has some unique properties like \#trending, @User, re-tweet and 140 character length constraints. These properties of the tweet make it unique over other social media text. The FSM model used these unique features for tweet identification. Apart from 140 character length constraints, Twitter has not given any restriction to users for posting a tweet, but, the user makes a unique pattern of writing tweet and

used symbolic notations to accommodate more information within 140 character length tweet. People uses universal resource locator (URL) of videos, images, etc., frequently in their tweets to show more visual information which cannot be shown through text. Therefore, we made a rule-set of tweet acceptance and rejection and deployed a deterministic finite automata (DFA) machine for tweet identification.

Further, we extend this for the elimination of null value tweets. These null value tweets acted as a noisy data for applications like sentiment analysis and sarcasm detection. These tweets doesn't have proper sentiment value of analysis. In the presence of null value tweets in the data set, performance accuracy has degraded. Therefore, we eliminate null value tweets to detect sarcasm sentiment in Twitter data. Finally, deployed an algorithm based on negation word like 'not,' 'never' and hashtag dictionary like '\#kidding' ,'\#lying', '\#pretend', etc. for sarcasm detection in filtered tweets. For example: “Super easy to focus at work today #kidding”. In this example, tweet sentiment is looking positive but due to hashtag 'kidding', sentiment becomes negative, and tweet becomes sarcastic. Using this approach, we collect the possible hashtag word which may use to make a sentence sarcastic and made a dictionary for sarcasm detection. Similarly, another example, “ I am never late for office \#joking”. In this example, the sentiment of the sentence is positive even though negative word (late) is present due to negation word 'never'. But hashtag 'joking' again make it sarcastic as

sentiment flips positive to negative due to the presence of joking hashtag. These two ideas are implemented for sarcasm detection in filtered tweets.