In this world there are plethora of data. Data from your school records, data of your office attendance, data of your computer memory utilisation etc. In recent years’ methods had been found and new methods are being discovered to use this data in constructive manner where constructive means to use for the benefits of society. The idea of all these processes can be obtained from one of the popular and mind boggling trend “collecting openly available data and analysing data” followed by research community. Twitter has evolved as the first choice for researcher to collect data and use them in their activities. Not only research community but commercial community also using these data to build products. These products are changing lives of millions. Due this practice followed by societies it becomes tremendously important to have meaningful message on the platforms from which data is going to be collected.

By allowing users from all over the world these communities open space for bigger sample space which is an advantage. But with this comes a disadvantage of having messages with no meaning or having meaning not related to the field. Having those messages on the sample space increases time and space required for the intended result. Our work is to define messages on these platforms so that we can discard such messages from sample space and save time.

Model Overview

We started defining our model with Twitter.

When we communicate with others there is a structure which helps in understanding the content. Like in English we can divide the sentences as (just a model not proposed)

S=NP+VP

S=>sentence

NP=>noun phrase

VP=verb phrase

After which further division and simplification of each section can be done.

Similarly, Twitter language can be put in a predefined structure

TS=@U\_NAME + TIMESTAMP + TP

TS=>twitter sentence

@U\_NAME=>@(handle) + registered username

TIMESTAMP=>time of posting the tweet

TP=>twitter phrase

Similar to further simplification is done in our model as follows

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

English: words with minimum value length

Normal English language + Unicode characters

* є

Transition Table(change background to white)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **RT** | **@** | **U\_NAME** | **#** | **TREND** | **ENGLISH** | **T** | **Є-CLOSURE** |
| 0 | 1 | 2 |  | 6 |  | 4 | 5 | 0,1,3,4 |
| 1 |  | 2 |  |  |  |  |  |  |
| 2 |  |  | 3 |  |  |  |  |  |
| 3 |  | 2 |  | 6 |  | 4 | 5 |  |
| 4 | 1 | 2 |  | 6 |  | 4 | 5 |  |
| 5 | 1 | 2 |  | 6 |  | 4 | 5 |  |
| 6 |  |  |  |  | 4 |  |  |  |

Images of proposed NFE-lambda and NFE

**NFA-E**

**NFA**

**DFA**

Techniques followed for parser

->first approach

Using regex properties

Constraint:

->url, #, @ followed by space (because of unusual behaviour of & with #, $ with http etc,)

Some basic logic rules

1. First separate all words separated by spaces
2. @/#/url can’t be followed by @/#/url without any character (including white space) in between them
3. For @ check if it’s preceding character is ‘.’, then check for ‘.’s preceding character is
   1. Other than “” then ENGLISH.@(everyone)
   2. “” then handle(follower)
4. If @ is followed rt, Rt, rT, RT which are followed by “” then universal user

To prove the proposed model validates a proper regular expression(RE), we first found the RE using transition table of minimised dfa

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **RT** | **@** | **U\_NAME** | **#** | **TREND** | **ENGLISH** | **T** |
| 0 | 1 | 2 |  | 4 |  | f | f |
| 1 |  | 2 |  |  |  |  |  |
| 2 |  |  | 3 |  |  |  |  |
| 3 |  | 2 |  | 4 |  | f | f |
| f | 1 | 2 |  | 4 |  | f | f |
| 4 |  |  |  |  | f |  |  |

Q0=lambda

Q1=Q0.RT+Q4.RT

=RT+Q4.RT

Q2=Q0.@+Q1.@+Q3.@+Q4.@

Q3=Q2.U\_NAME

Q4=ENG+T+Q3.(ENG+T)+Q4.(ENG+T) +Q6.TREND

Q6=Q0.#+Q3.#+Q4. #

=# +Q3.# +Q4.#

To find the RE for state Q4 which is a final state we expressed every other term in terms of Q4, so that we can use Arden’s theorem

Q0 and Q1 need no more modification.

Q2=@+RT.@+Q4(RT.@+@) +Q2.U\_NAME.@

= (@+RT.@+Q4. (RT.@+@)). (U\_NAME.@) \* [using Arden’s theorem]

Q3=Q2.U\_NAME= (@+RT.@+Q4. (RT.@+@)). (U\_NAME.@) \*. U\_NAME

Q4 don’t need modification.

Q6=# +Q3.# +Q4.#

We get

Now putting values in terms of Q4 IN

Q4=ENG+T+Q3.(ENG+T)+Q4.(ENG+T) +Q6.TREND

We get

Q4= [ ENG +

T +

#. TREND+

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

] +

Q4. [ ENG +

T +

#. TREND +

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

Now using Arden’s theorem, we get RE for Q4 and hence for the FSM as follows

Q4= [ ENG + #. TREND+ (@+RT.@) (U\_NAME.@) \*.U\_NAME.(ENG + #. TREND+T)]. [ ENG + (@+RT.@) (U\_NAME. @) \*.U\_NAME.(ENG + #. TREND+T) + #. TREND] \*

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

-------------------------------------------10 th October---------------------------------------------------

Now we got the final RE

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

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

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

L1= [(@+RT.@) (U\_NAME.@) \*.U\_NAME.(ENG + #. TREND)] +

L2= [ 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=Swing with him from the chandelier [~~#~~MakeItJhakaas](https://twitter.com/hashtag/MakeItJhakaas?src=hash)

L2=[~~@~~TLCINOfficial](https://twitter.com/TLCINOfficial) [~~@~~arjunk26](https://twitter.com/arjunk26) [~~#~~MakeItJhakaas](https://twitter.com/hashtag/MakeItJhakaas?src=hash) high heel dance on his style

Now L1. L2 is of the form [@+RT.@) (U\_NAME.@) \*.U\_NAME.(ENG + #. TREND)] .[ ENG ] . [ ENG ]. [ #. TREND] \*

and in this case

L=L1.L2= Swing with him from the chandelier [~~#~~MakeItJhakaas](https://twitter.com/hashtag/MakeItJhakaas?src=hash) [~~@~~TLCINOfficial](https://twitter.com/TLCINOfficial) [~~@~~arjunk26](https://twitter.com/arjunk26) [~~#~~MakeItJhakaas](https://twitter.com/hashtag/MakeItJhakaas?src=hash) high heel dance on his style

Here the L satisfies the RE and this example passes through the proposed DFA

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

2.Kleene closure:

L1∗

set of repeated concatenation of a string in L1

L1\* is f the form L1= [(@+RT.@) (U\_NAME.@) \*.U\_NAME.(ENG + #. TREND)] +

L1\* =Swing with him from the chandelier [~~#~~MakeItJhakaas](https://twitter.com/hashtag/MakeItJhakaas?src=hash) Swing with him from the chandelier [~~#~~MakeItJhakaas](https://twitter.com/hashtag/MakeItJhakaas?src=hash)

Here the L1 satisfies the RE and this example passes through the proposed DFA

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

3.union:

L1∪L2=L1+L2

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

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

4.complementation:

Σ∗−L1

set of all possible strings that are not in L1

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

L1=  [ ENG + #. TREND+ (@+RT.@) (U\_NAME.@) \*.U\_NAME.(ENG + #. TREND)] +

L2= [ ENG + #. TREND]+

set of strings which are in L1 but not in L2

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

this passes through the FSM

eg- Mera Ishq is going to be a super hit. [~~@~~**SaanseinTheFilm**](https://twitter.com/SaanseinTheFilm)[~~#~~**MeraIshqByArijitSingh**](https://twitter.com/hashtag/MeraIshqByArijitSingh?src=hash) Beautiful voice ....

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

6.intersection:

L=L1∩L2 =[ ENG + #. TREND] +

set of strings in both L1 and L2

eg-Ashwin 4 - 0 Williamson! Ashwin has got Williamson's number in this series! [~~#~~**INDvNZ**](https://twitter.com/hashtag/INDvNZ?src=hash)

it passes through DFA

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

7.reversal:

(L)r

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

(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

set of the reversal of all strings in L