

Lab Based Project

New Entity Discovery



*Project under Dr Dhaval Patel,
Professor at Department of computer Science and Engineering, IIT Roorkee*

*Report Submission By:
Manu Chandel 13114013, Rahul Aggarwal 13114041, Rishabh Bansal 13114045
Department of computer Science and Engineering, IIT Roorkee*

Table of contents

Introduction	3
Abstract	4
Collecting Wiki Human Names	5
Using Stanford NER	7
Indian Names Data	10
Twitter Api	11
Refining Headlines	13
New Popular Entity Discovery	15
Flow Diagram	16
Testing System	18
Result Analysis and Ambiguities	19
Output	20
Files Attached	21

Introduction

Wikipedia is the ultimate source of information on internet. It provides information about everything that possibly exists. Right from the famous people to famous places, events, history, geography, science, maths, literature and much more can be found on wiki pages. Wikipedia is so popular that search queries fired on various search engines returns links to wiki pages on the first page of query results. Yet how do wikipedia stay updated? how are new articles added to wikipedia everyday ? There are tens of thousands of volunteers who keeps updating the information about the article that they find interesting. But how to these volunteers know that which article should be updated when. Well obvious answer to this question is when the new things comes up regarding an article it should be updated. But how do we know when the new things comes up?? Or what are the new articles that are to be added to wikipedia pages ??.

Finding new articles to create a wiki page and old articles to update them can be done by following news articles. Journalism and media is the boon of 21st century. At any part of the world whatever events takes place happens to appear in the news headlines of various news agencies. Our focus is on mainly human entities. We want to create a system that would analyze news headlines and try to dig out Human Entity Names that were not before in wikipedia databases. This would help volunteers to add new articles to wikipedia about new people rising in the limelight. Thus speeding up the process of updating about "NEW HUMAN ENTITIES". Hence the "NEW ENTITY DISCOVERY".

Abstract

We built a system that would analyze headlines of a given day and find out new entities. We build this system on python using mysql server. We made use of api's like twitter and standard tools like Stanford NER and NLTK .

New entities discovery is focused on human names only. Some of the results below are displayed from the output of the system from when it was first analyzed on headlines set data from 1st January 2016 to 31st March 2016.

33	visheshwar ojha	2016-02- 12	vice-president BJP Bihar
34	dipti sarna	2016-02- 12	Rukh 'Darr Shah Snapdeal
35	kanhaiya kumar	2016-02- 12	Afzal JNU Guru
36	umar khalid	2016-02- 19	JNU
37	tushar mahajan	2016-02- 22	Pampore Army Captain
38	anirban bhattacharya	2016-02- 23	Police JNU Delhi Khalid Umar
39	ashutosh kumar	2016-02- 27	JNU Shanker Friday Chakravarty

Collecting Wiki Human Names

Collection of names of people who have their wikipedia page is important so that whenever new entity is discovered it can be compared with already existing entities.

The first thing we did was collection of names of the people who have wikipedia page of their own. We wrote our very own scraper in python which scraped about 1.3million names from 4500 web pages. Scraper script was written in python.

The script makes use of BeautifulSoup library. The initial problem faced was how to keep track of web pages which were already scraped and the one which were left for scraping. This problem was solved by creating a text file named `url`. This text file would keep record of the link of the webpage which is to be scraped next. In case where the connection was reset by the host or some other error may occur, scraping can be resumed from the url which was present in `url` text file.

The `url` text file in a way added resuming capability for scraping. After the error has occurred, to resume scraping, simply copy the url from the text file paste it on line 10 of `scraper.py` and run the script. To keep track of progress made by script, a variable `i` was used and it would print a number when a page was successfully scraped. All the transactions into the database were committed only after one complete web page was scraped. This avoided incomplete transactions to take place due to error.

Information was scraped from






























































<https://www.wikidata.org/w/index.php?title=Special:WhatLinksHere/Q5&limit=500>

The script file is attached by the name **a.py**

All the script files written assumes that there exist a database "*Lab Based Project*" running on a local mysql server. To create database run script by the name **db.py**

Here are few results in the database which were scraped by the above script.

The image represents database entries on mysql server with phpmyadmin GUI.

		ID	NAME	DESCRIPTION
<input type="checkbox"/>	 Edit  Copy  Delete	864714	manouchehr eghbal	iranian prime minister
<input type="checkbox"/>	 Edit  Copy  Delete	865182	manuel azaa	prime minister of spain president of spain
<input type="checkbox"/>	 Edit  Copy  Delete	865940	manuel maria coelho	prime minister of portugal
<input type="checkbox"/>	 Edit  Copy  Delete	866148	manuel pardias	spanish anarchist who assassinated jos canalejas t...
<input type="checkbox"/>	 Edit  Copy  Delete	866336	manuel ruiz zorrilla	prime minister of spain
<input type="checkbox"/>	 Edit  Copy  Delete	866906	maphevu dlamini	prime minister of swaziland
<input type="checkbox"/>	 Edit  Copy  Delete	872939	marcolino moco	prime minister of angola
<input type="checkbox"/>	 Edit  Copy  Delete	873083	marcos gutierrez	argentine footballer that his last club was deport...
<input type="checkbox"/>	 Edit  Copy  Delete	875685	margaret thatcher	prime minister of the united kingdom
<input type="checkbox"/>	 Edit  Copy  Delete	875701	margaret trudeau	exwife of the late canadian prime minister pierre ...
<input type="checkbox"/>	 Edit  Copy  Delete	875754	margaret whitlam	wife of gough whitlam an australian prime minister
<input type="checkbox"/>	 Edit  Copy  Delete	876267	marers skukenieks	prime minister of latvia
<input type="checkbox"/>	 Edit  Copy  Delete	877039	mari alkatiri	former prime minister of east timor
<input type="checkbox"/>	 Edit  Copy  Delete	877764	maria das neves	santomese politician first female prime minister
<input type="checkbox"/>	 Edit  Copy  Delete	878673	maria liberia peters	prime minister of the netherlands antilles
<input type="checkbox"/>	 Edit  Copy  Delete	879848	maria lady walpole	wife of prime minister of the united kingdom
<input type="checkbox"/>	 Edit  Copy  Delete	884321	marina mahathir	daughter of malaysian prime minister
<input type="checkbox"/>	 Edit  Copy  Delete	885555	mario frick	prime minister of liechtenstein
<input type="checkbox"/>	 Edit  Copy  Delete	886333	mrio pires	prime minister of guineabissau
<input type="checkbox"/>	 Edit  Copy  Delete	886563	mario scelba	prime minister of italy
<input type="checkbox"/>	 Edit  Copy  Delete	887332	mrir gailis	prime minister of latvia

Using Stanford NER

Next step was to extract human names from the given headline data.

Stanford NER is pretty standard Named Entity tagger used widely. Stanford NER Tagger successfully tags person, location, organization in English sentence.

Stanford NER works well on a large set of data. Also it takes few seconds to tag a sentence. It takes input stream of tokens and tags them accordingly. We decided to create a table of tokens where we can keep count of different tags of the token. We used 3-class classifier hence there were only four possible tags namely PERSON, LOCATION, ORGANIZATION and O.

We faced two problems while using NER tool.

The first one was the time efficiency problem. It took around 3 seconds for NER tool to tag a single headline. We had to build our table of tokens from 8 Lakh Headlines. So it was not possible to tag each headline separately. To cope up with this problem we decided to tokenize all 8 Lakh headlines and give it as a input to the tool instead of one single headline.

The time efficiency problem solution gave rise to another problem namely buffer overflow error. NER tool could not handle 8L headlines and gave runtime error of buffer overflow. This was solved by tagging 50k headlines instead of 8 Lakhs. We made a set of tokens out of 8 Lakh headlines by splitting them into 16 files of 50k headlines each. This solved both the time efficiency and buffer overflow problem.

Number 50k was chosen by hit and trial.

b.py : creates 16 different textfiles from headlines. Each text file contains tokens of 50k headlines

After the text files were created, next step was to tag tokens from the text files as PERSON, ORGANIZATION, LOCATION and O.

NER tool was run on each text file and corresponding output of each text file was recorded in another text file named **tagged tokens**. This was performed by a script file c.py

The script file c.py.

c.py : script to tag tokens from each token text file created by b.py.

Each text file contained repeated tokens like **is,or much** etc.. After tagging of the tokens were complete we needed to unify these tokens into one single file named **TrainingData** so that there were no repeating tokens and token tagging frequencies were updates. This was performed by a script file d.py

d.py : script to unify all tagged tokens and add their frequencies.



























































Here is a image of some of the text in TrainingData

```
sevenfold 0 12
iamai ORGANIZATION 7
buscater 0 4
bhoopasandra LOCATION 1
nailah PERSON 7
beetle 0 19
has PERSON 61
canela PERSON 1
crow PERSON 6
tsonga PERSON 32
losse 0 1
sonowal 0 12
tcommulkvwx7x 0 1
tcokzhqwmngx 0 1
565000 0 4
illuminator 0 1
febmarh 0 2
slithering 0 4
younger 0 150
chamdimal 0 1
unblocked 0 3
gritty ORGANIZATION 4
pawning 0 5
srfti ORGANIZATION 1
railauto 0 3
madarsa 0 16
isclaime 0 28
announce ORGANIZATION 30
mate 0 41
smoking ORGANIZATION 6
waterboards 0 1
tcoiyszaxqdxv 0 1
cultivated 0 4
kat LOCATION 1
saral ORGANIZATION 1
expremiers 0 1
```


After getting unique set of tagged tokens and their frequencies we entered these tokens into database using this script

e.py : script to insert tagged tokens to database TrainingDataDictionary table

Below is screenshot of some table entries

<input type="checkbox"/>				safepassage	O	1
<input type="checkbox"/>				safend	ORGANIZATION	1
<input type="checkbox"/>				safely	ORGANIZATION	1
<input type="checkbox"/>				safely	O	68
<input type="checkbox"/>				safehouses	O	1
<input type="checkbox"/>				safehaven	ORGANIZATION	1
<input type="checkbox"/>				safehaven	O	138
<input type="checkbox"/>				safeguards	O	36
<input type="checkbox"/>				safeguarding	O	1
<input type="checkbox"/>				safeguarded	O	3
<input type="checkbox"/>				safeguard	ORGANIZATION	2
<input type="checkbox"/>				safeguard	O	37
<input type="checkbox"/>				safegaurds	O	1
<input type="checkbox"/>				safeerazeez	PERSON	2
<input type="checkbox"/>				safeer	PERSON	32
<input type="checkbox"/>				safeek	PERSON	6
<input type="checkbox"/>				safeed	PERSON	2
<input type="checkbox"/>				safecharge	ORGANIZATION	1
<input type="checkbox"/>				safe	ORGANIZATION	25
<input type="checkbox"/>				safe	O	837
<input type="checkbox"/>				safder	PERSON	6
<input type="checkbox"/>				safdarjung	ORGANIZATION	7
<input type="checkbox"/>				safdarjung	LOCATION	12

Indian Names Data

Stanford NER is not so good at tagging Indian names. As we can see in below two screenshots.

Stanford Named Entity Tagger	Stanford Named Entity Tagger
Classifier: <input type="text" value="english.all.3class.distsim.crf.ser.gz"/>	Classifier: <input type="text" value="english.all.3class.distsim.crf.ser.gz"/>
Output Format: <input type="text" value="highlighted"/>	Output Format: <input type="text" value="highlighted"/>
Preserve Spacing: <input type="text" value="yes"/>	Preserve Spacing: <input type="text" value="yes"/>
Please enter your text here:	Please enter your text here:
<input type="text" value="Rahul Aggarwal study at IIT Roorkee"/>	<input type="text" value="Donald Trump study at IIT Roorkee"/>
<input type="button" value="Submit"/> <input type="button" value="Clear"/>	<input type="button" value="Submit"/> <input type="button" value="Clear"/>
Rahul Aggarwal study at IIT Roorkee	Donald Trump study at IIT Roorkee
Potential tags: ORGANIZATION LOCATION PERSON	Potential tags: ORGANIZATION LOCATION PERSON

As in the above Image it tagged Donald Trump correctly but could not tag Rahul Aggarwal

Twitter Api

To deal with Indian Name recognition problem we wanted huge dataset of Indian names so that we fill our table entries.

Getting Indian names dataset was itself a problem. We decided to use twitter api to gather Indian names. Indian celebrity Amitabh Bachchan has 20 Million followers on twitter. If we could get a follower of list of Amitabh Bachchan we could get a handful of Indian names.

Twitter Api has a request limit restriction in a window of 15 mins. That is only 30 requests could be made for each 15 minutes. Each request could fetch about 200 followers. We ran the script for 24 hours fetched about 7.5 Lakhs followers.

f.py : Script to fetch twitter follower list of Amitabh Bachchan.

After fetching follower list we unified all the the follower names . Names were tagged as PERSON and repeated names were accounted for frequency. After unifying these names were inserted into table created previously.

g.py : Unifies all Twitter followers to one file

h.py : Enters Twitter followers with their tag into TrainingDataDictionary

Here is the screenshot of unified tokens of twitter followers

gavai PERSON 1	raichandani PERSON 4
woods PERSON 4	samirkhan PERSON 3
gaval PERSON 1	manishmessy PERSON 1
foluke PERSON 1	sharadjadhav PERSON 1
sonju PERSON 1	parvaz PERSON 3
karthee PERSON 1	saptashwa PERSON 1
gavad PERSON 1	parvat PERSON 6
haron PERSON 1	laxmansingh PERSON 1
polishetty PERSON 1	dewvasi PERSON 1
khosto PERSON 1	parvas PERSON 1
irisbrock PERSON 1	prisha PERSON 4
durugkar PERSON 1	ajesh PERSON 12
maneeshdhiman PERSON 1	raghveer PERSON 1
pragadish PERSON 1	mosfiqur PERSON 1

Here is the screenshot of followers list

ID	NAME	SCREEN_NAME
407660440	Shankar khupse	khupse_shankar
717084792580341760	ALEKH ROUT	Alekhrou09
717084619473031168	Toofa Aln	toofa_aln
717080449957253120	priyankapaul	priyank25278747
717084089874927616	Sandeep Chhikara	Sandeep28060689
717084271576539137	md saddam shahjada	saddam_shahjada
2354622403	Saleh Muhammed	salehmuhammed39
717083808839766016	Zoheb khan	Zohebkx71177417
264504704	Nafiz Rahman	taklagunda
717080766610583552	Ibrahim Merchant	IbrahimMerchan6
2491438189	Souvik Mallick	SouvikMallick9
717083682566082560	Mohit Kaushik	immkaushik
717083825633828864	Krishna upadhyay	Krishna75679007
2222241482	Karthik Bhanu	KarthikBhanu1
717083348414300160	MAHESH GUPTA	mahesh9473
1162765831	kiran	371Kiran
717083017563377664	Raju yadav	Rajuyad92067818
593341136	parth amin	i_parthamin
717083393385635841	sanjay kumar	sanjayk43501028
717083774752673793	Sonam sharma	Sonamsh60368805
717059721509752832	Sangharsh	DSangharsh
717083417662218240	naresh malviya	naresh92miles
4437102019	James Hof	JayCee_Hof
4517035097	LilJayTheThugginGee	Sexygoodietosh1
1658380741	Malik	UnisMalikk

Refining Headlines

After creating TrainingDataDictionary containing tagged tokens and Indian names next step was to refine headlines data. Original data of headlines contained url and many columns.

A screenshot representing original headlines data

id	newsHeadline	start_time_stamp	end_time_stamp	first_occurance	last_occurance	total_count	URL
3741143	Horoscope 2015 Predictions	2015-01-03 00:47:01	2016-02-04 12:45:01	15292	33633	18342	http://www.astrospeak.com/horoscope-predictions
3741430	Horoscope 2015 Predictions	2015-01-03 01:17:01	2016-02-04 14:15:01	15293	33636	18344	http://www.astrospeak.com/horoscope-predictions
3756949	CNN 10: Startups to watch	2015-01-04 09:17:01	2016-01-06 21:17:01	15357	32258	16902	http://www.cnn.com/interactive/2014/10/tech/cnn10-...
3778296	The hottest apps in tech	2015-01-06 10:47:01	2016-01-06 21:17:01	15456	32258	16803	/2014/11/24/tech/social-media/industry-insiders-ho...
3805953	CES 2015: Gadgets extravaganza begins	2015-01-08 03:47:01	2016-03-09 08:15:01	15538	35213	19676	http://timesofindia.indiatimes.com/tech/special/ce...
3820624	Germany beat Pak to win CT	2015-01-09 01:17:01	2016-03-09 08:15:01	15581	35213	19633	http://timesofindia.indiatimes.com/videos/sports/o...
3909406	Inside the Moments After Astronauts Evacuated ISS	2015-01-15 08:47:01	2016-03-09 08:15:01	15882	35213	19332	/Technology/international-space-station-inside-mom...
3923704	The weird things we sent to space	2015-01-16 06:47:01	2016-01-25 18:15:01	15926	33164	17239	/videos/tech/2015/01/15/iss-odd-items-crane-orig-m...
3927505	Elon Musk Wants to Rein in the Robots	2015-01-16 11:17:01	2016-03-09 08:15:01	15935	35213	19279	/Technology/elon-musk-donates-10-million-make-robo...
3939004	Long-Missing Beagle 2 Mars Lander Finally Found	2015-01-17 05:47:02	2016-03-09 08:15:01	15972	35213	19242	/Technology/long-missing-beagle-mars-lander-final...

Above table contained 9 columns. The relevant information needed was source, headline and its first occurrence. Also source of the headline was not provided.

Source of the headline could be found out by the url column. Url of each headline contained information about the domain of the news agency. Domain names generally go by the news agency names namely hindustantimes.com, bbc.co.uk etc..

We extracted source of the headlines from url and made a separate table for these refined headlines.

h.py : script to generate refined headlines with source

Here is the screenshot of refined headline entries

ID ▲	NEWS	DATE	SOURCE
1	Only Europe together can rise to terrorism challen...	2016-03-28	suntimes
2	The annihilation by caste	2016-02-02	thehindu
3	Need a movement against foeticide, dowry: Arvind K...	2016-03-08	ibnlive
4	Luke Cage Teaser, Daredevil Season 2 (Spoiler Aler...	2016-03-24	huffingtonpost
5	BoAML: Higher chance of Brexit	2016-03-23	cnbc
6	When Shraddha Kapoor beat Tiger Shroff in dance	2016-02-19	hindustantimes
7	Spooked by FANG: One more thing to watch out for i...	2016-01-16	cnbc
8	Commodity exchanges eye revival under Sebi after m...	2015-12-21	financialexpress
9	Neymar Suffers Another Blow Off The Pitch And This...	2016-02-16	ndtv
10	Einstein was right! 'Ripples' from black holes det...	2016-02-12	rediff
11	One injured in Tri-State crash near Bridgeview	2016-03-04	suntimes
12	Technical forex report for Jan 22	2016-01-22	thehindubusinessline
13	J&K government formation: Consensus has emerged be...	2016-04-01	india
14	Gurudwara Vandalised In US	2016-03-04	ndtv
15	January Exports Fall for 14th Straight Month	2016-02-15	ndtv
16	Virat Kohli pips Aaron Finch to top T20I batting r...	2016-03-29	cricbuzz
17	Rivals line up challenge to Alipay in China	2015-12-21	ft
18	Day in Pics: 19th January	2016-01-19	india
19	Update:- intraday 19.2.16 below stocks looking goo...	2016-02-20	moneycontrol
20	Veggies costlier, milk supply hit in NCR	2016-02-21	tribuneindia
21	Talks on terror, defence during French President F...	2016-01-22	indianexpress
22	Reliance Jio?s plan: Rs 200 sim card will give you...	2016-03-30	hindustantimes

New Popular Entity Discovery

New popular entity can be discovered by analyzing headlines of a particular date.

deciding_new_entity() function takes in a Date argument in YYYY-MM-DD format and returns map with a key as new entity and value as list of related nouns.

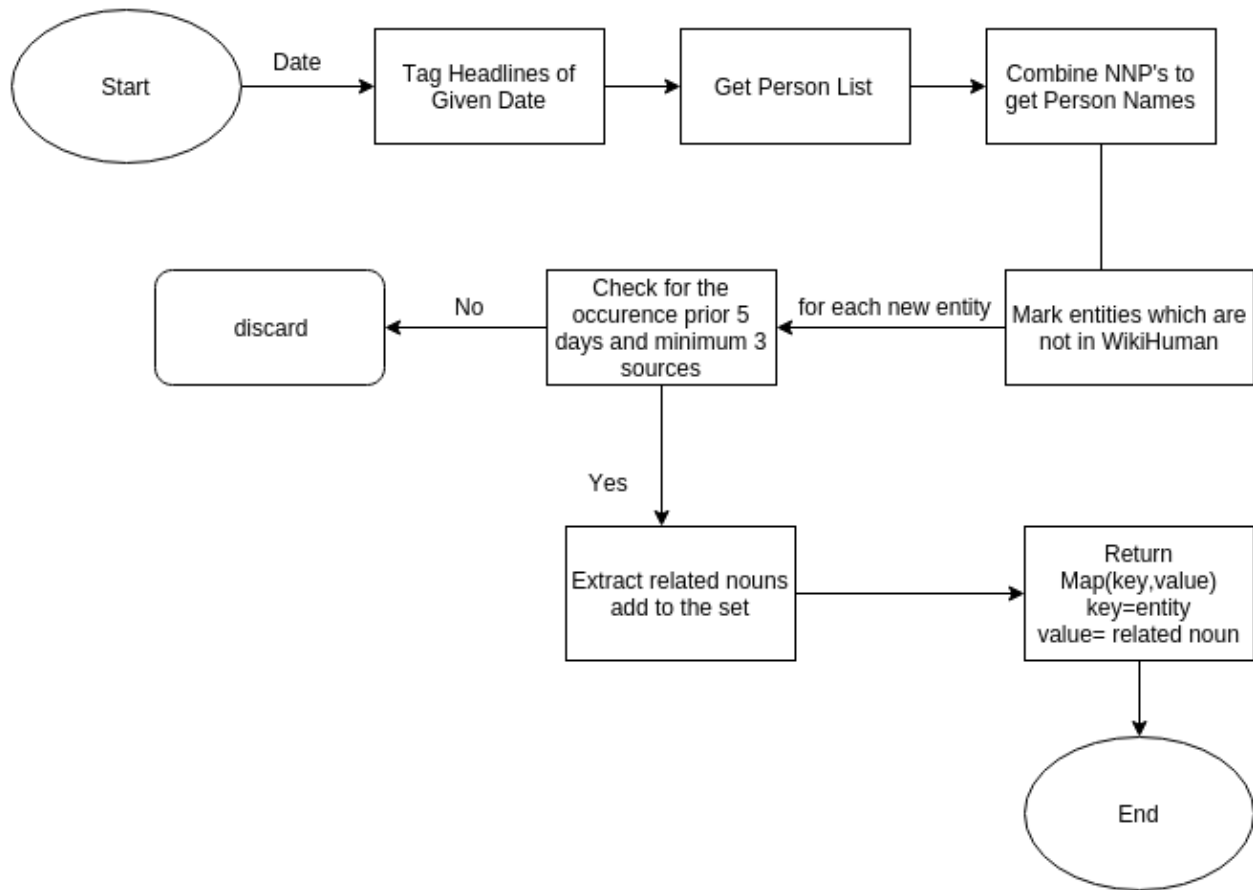
i.py : Script to find new popular entity

We developed following algorithm to discover new entity

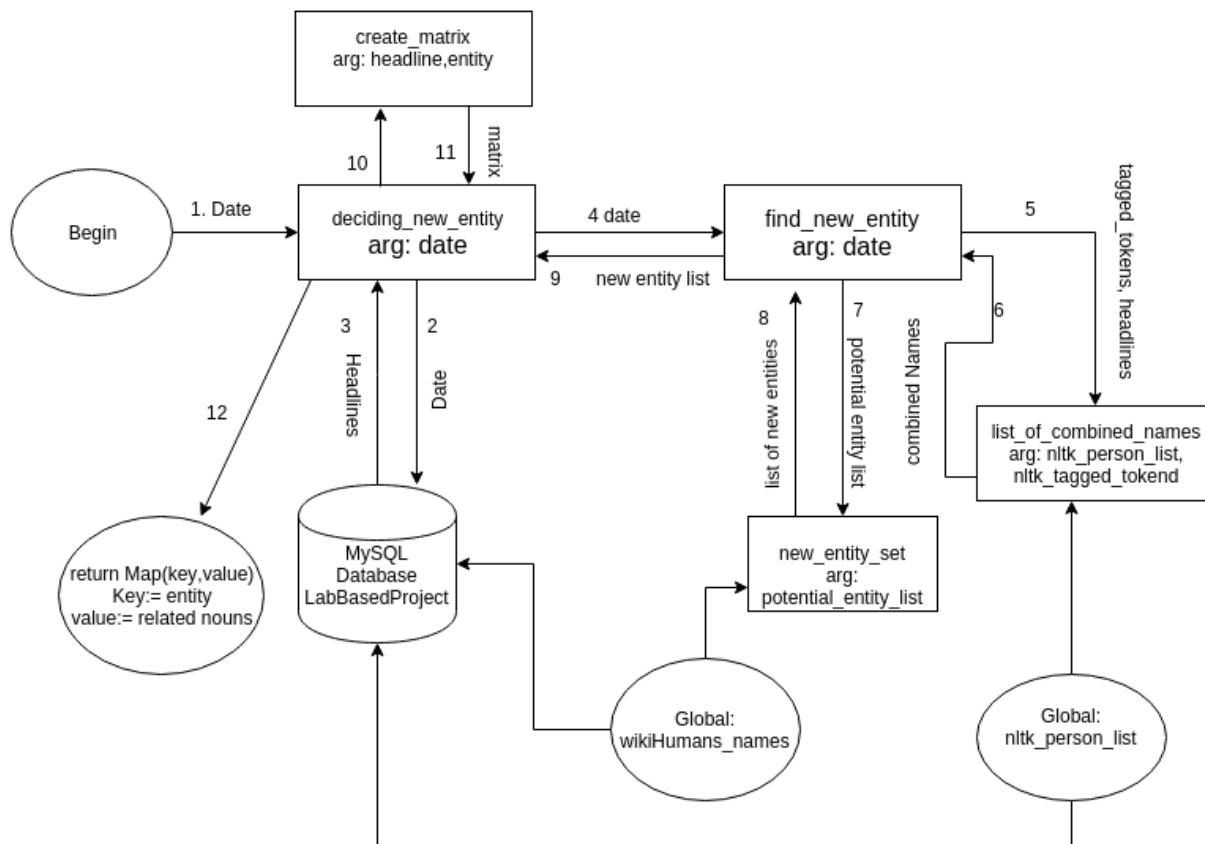
1. Do POS tagging of a headline using `nltk.pos_tag()`
2. For all NNP's search for ones who have PERSON tag occurrences greater than all other tags in TrainingDataDictionary. Mark them as person.
3. For NNP's who are tagged person in step 2 and occur side by side combine them to form full name. Mark these names as person names.
4. For all person names check whether they already exist in WikiHumans Data Base. If not mark them potential entities.
5. For each potential entities who have not occurred 5 days prior in any of the headlines, have occurred more than 5 times in present day headlines and have news coverage of at least 3 different sources then mark them as new popular entity.
6. For each new popular entity search proper nouns in headlines of given date. If a proper noun occurs more than half times of the number of headlines then that noun is related to that entity.

Flow Diagram

Here is the flow diagram of the system



Here is flow diagram of execution of **i.py**

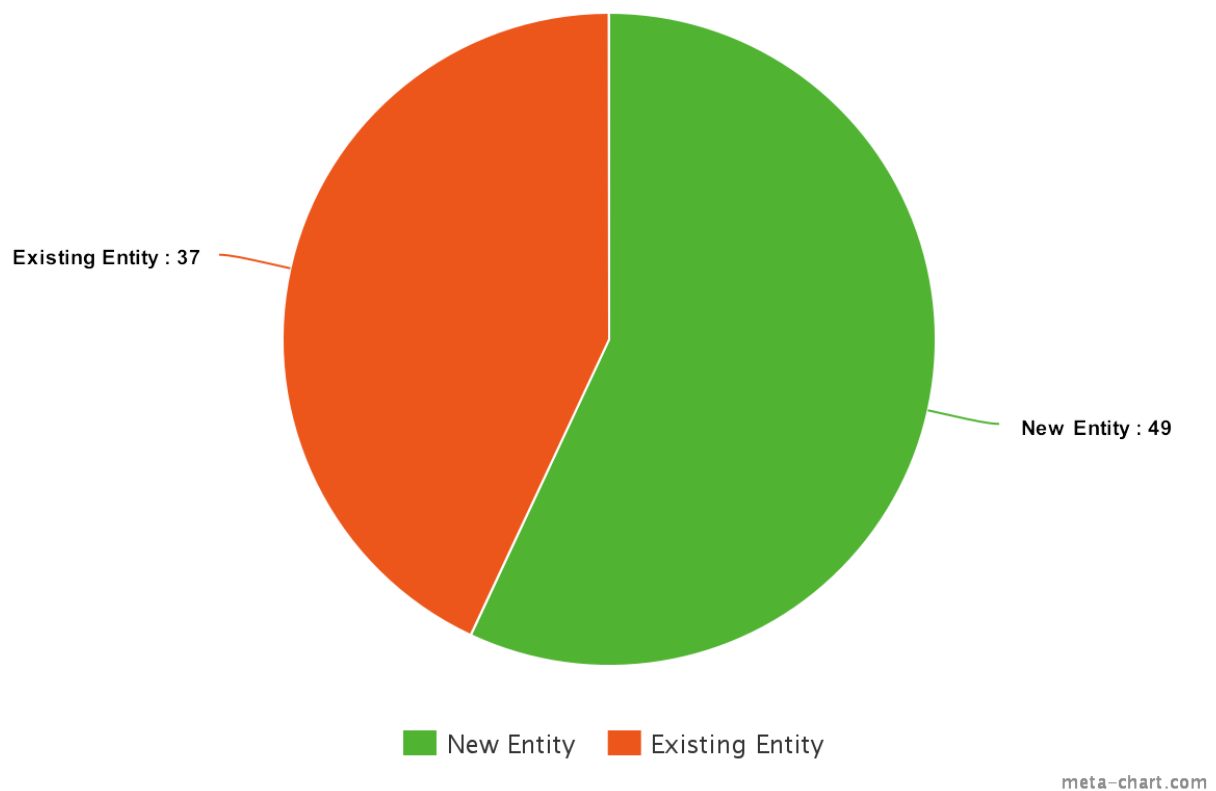


Testing System

System was tested by running from 1st Jan 2016 to 31st March 2016.

j.py: Script used to run system

k.py : script used to create html file of output



Result Analysis and Ambiguities

The above system do not deal with ambiguities in names. Also some times media uses nickname for famous personalities. Eg Sanju Baba for Sanjay Dutt. System do not know that Sanjay Dutt and Sanju Baba refers to same person.

In addition to above problems there is a problem of name abbreviations. Eg. MS Dhoni and Mahendra Singh Dhoni refers to same person. System treats both persons as different.

Due to above problems the efficiency of system reduces. It tends to discover entities whose wikipedia page is already there. To overcome this problem system can be modified to generalize more. In addition of searching new entities it can suggest which of the old entity pages are to be updated according to new events occurring in news media.

Output

Index.html file contains output of the testing code. Here are screenshots of results

New Entities Discovered

Sr.No	Entity Name	Date of Detection	Releated Nouns	Wikipedia Page Creation	Wikipedia Page Edit Near to Date of Detection	No of Wikipedia Edits Edit Date
1	gursewak singh	2016-01-04	Pathankot	None	None	None
2	niranjan kumar	2016-01-04	Pathankot Col Lt	None	None	None
3	pranav dhanawade	2016-01-05	Tendulkar Sachin	2016-01-05	2016-01-05	36
4	khushi kapoor	2016-01-13	Sridevi	None	None	None
5	severus snape	2016-01-14	Potter Alan Harry Rickman	None	None	None
6	navya naveli	2016-01-15	Amitabh Bachchan Nanda	None	None	None

Files Attached

1. **a.py** : WikiHuman Scraper
2. **b.py** : creates 16 different textfiles from headlines. Each text file contains tokens of 50k headlines
3. **c.py** : script to tag tokens from each token text file created by b.py.
4. **d.py** : script to unify all tagged tokens and add their frequencies.
5. **e.py** : script to insert tagged tokens to database TrainingDataDictionary table
6. **f.py** : Script to fetch twitter follower list of Amitabh Bachchan.
7. **g.py** : Unifies all Twitter followers to one file
8. **h.py** : Enters Twitter followers with their tag into TrainingDataDictionary
9. **i.py** : Script to find new popular entity
10. **j.py**: Script used to run system
11. **k.py** : script used to create html file of output
12. **Index.html** : output