WEB PLACE NAME DICTIONARY IMPLEMENTATION USING TWITTER AS SOURCE TO DEVELOP TRAFFIC INFORMATION SYSTEM

Rachmad Akbar, Ari Wibisono and Petrus Mursanto

Faculty of Computer Science, Universitas Indonesia,
Kampus Baru UI Depok, Jawa Barat, 16424, Indonesia

E-mail: rachmad.akbar@ui.ac.id

Abstract

Congestion is one of the severe problems that occurs in big cities like Jakarta, the center of the Indonesian government. One alternative solution to solve this problem is with a Traffic Information System which is easily accessible by the public. Twitter as a social media can be used as one of information source to develop the Traffic Information System. The most common problem in the processing of information on Twitter, especially in Indonesia, is the use of abbreviations and typographical error (typo). We propose an application of Web Place Name Dictionary in addressing abbreviations and typos in the name of the location contained on Twitter to develop a Traffic Information System. Three cases in find coordinate point in the street that derived from location name in Twitter can be handled through the proposed system. This system produces 90% accuration rate on 100 data set that obtained via @TMCPoldaMetro. This study only focus on the implementation of Web Place Name Dictionary method so we do not compare our system with the other methods.

Keywords: *Coordinate, Dictionary, Location, Traffic, Twitter*

Abstrak

Kemacetan merupakan salah satu masalah berat yang terjadi pada kota-kota besar di Indonesia, tidak terkecuali di DKI Jakarta yang menjadi pusat pemerintahan Indonesia. Salah satu alternatif solusi untuk mengatasi masalah tersebut adalah dengan adanya Traffic Information System yang mudah diakses oleh masyarakat. Media sosial Twitter dapat dimanfaatkan sebagai salah satu sumber informasi untuk mengembangkan Traffic Information System tersebut. Permasalahan yang sering dihadapi dalam pengolahan informasi pada Twitter, khususnya di Indonesia adalah penggunaan singkatan serta typo. Kami mengajukan penerapan *Web Place Name Dictionary* dalam mengatasi masalah singkatan serta kesalahan ketik (*typo*) pada informasi nama lokasi yang terdapat pada Twitter untuk mengembangkan Traffic Information System. Tiga kasus pencarian titik koordinat pada jalan berdasarkan nama lokasi yang diperoleh dari Twitter dapat ditangani melalui sistem yang diajukan. Sistem ini menghasilkan akurasi 90% pada 100 data uji nama jalan yang diperoleh melalui @TMCPoldaMetro. Studi ini hanya berfokus pada implementasi metode *Web Place Name Dictionary* sehingga perbandingan dengan metode lain tidak dilakukan.

Kata Kunci: *Lalu lintas, Lokasi, Kamus, Koordinat, Twitter*

1. Introduction

Congestion is the impact of the imbalance between capacity of the road and the number of passing vehicles. Currently the length of roads in Jakarta are only 7,208 kilometers [1], while the number of motor vehicles until April 2012 is 13,346,802 [2]. Based on that facts, we will produce a ratio of 1,851 vehicles per kilometer. The number is very far from the ideal ratio. The ideal ratio is less than 100 vehicles per kilometer [3].

The rapid development of technology makes price of mobile devices become more affordable. Hence, the use of social media Twitter increases. Various informations are distributed to the public through the media, including information of traffic conditions shared by @TMCPolda-Metro, @LewatMana and @SonoraFM92. This phenomenon can be used as an opportunity to develop a Traffic Information System to fulfill the needs of information regarding traffic conditions on the roads that are going to be passed.

The most common problem in the processing of information on Twitter, especially in Indonesia, is the use of abbreviations and typographical error (typo). There are some methods that can be used to solve this problem, such as utilization of geotag, utilization of tagging by tagger, and construction of a web place-name dictionary [4]. Web place-name dictionary is chosen to be a method because Indonesian people seldom use geotagging feature and usually use non-standard sentence that causes tagger tagging process to be difficult to implement.

Similar studies have been carried out by [5]. Endarnoto used Natural Language Processing to identify the name of the location being mentioned in a Tweet. However, the research conducted does not include how to obtain precise location coordinates of a location name that has been obtained through Twitter.

2. Methodology

OpenStreetMap

OpenStreetMap (OSM) is a free and editable map that allows user to access the entire dataset on the map (latitude and longitude coordinates) around the world [6]. Disclosure of this dataset can be used as a source of data and information about coordinates of path and name of street in Jakarta. The structure of the node in OSM can be seen in Figure 1.

A street is represented as a collection of nodes that have latitude and longitude coordinates, the nodes is ordered from starting point to the end point of the street. The structure of OSM street can be seen in Figure 2.

MapQuest is one of the leading providers of online mapping services that support OSM [7]. MapQuest provides several features to support developers to build applications, both desktop applications, web and mobile. JavaScript 6.0 SDK is available to develop applications on the client-side. While on the server-side developers can develop an application using C++, .NET and Java SDK [8]. In determining the coordinate of a point on the street that represents information from the Twitter, a mechanism is needed to calculate the distance between two coordinate points via existing roads. One of the services that can be used is a web-service which is provided by yournavigation.org.

Yournaviagation.org use Gosmore routing machine as routing engine so that the service is more lightweigh, scalable and fast [9]. There are some parameters is used in this web service:

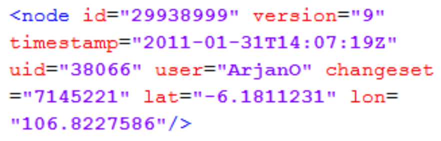
- flat = latitude from start location
- flon = longitude from start location
- tlat = latitude from destination location
- tlon = longitude from destination location
- v = kind of vehicle
- fast = 1 for fastest route, 0 for shortest route
- layer = the selection of Gosmore agencies that used to calculate the route

The yournavigation.org web service returns a kml file that has a structure shaped like xml, so the file is easy to understand. One of the data used in developing this system is the value of the distance from two point coordinates. The sample of yournavigation.org kml file can be seen in figure 4. In fact, there are 3 kinds of Tweet that will be informed by @TMCPoldaMetro. The kinds are Tweet with one location name, two location and three location names. The Tweet that has two or three location names will be processed by utilizing the web service.

Web Place-name Dictionary

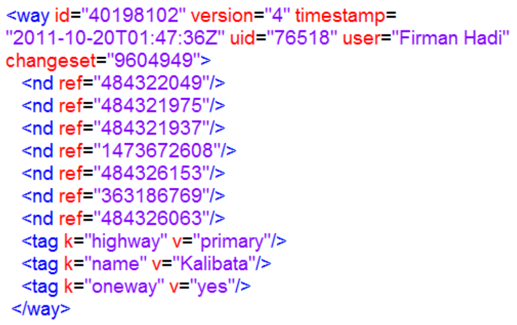
One of the solutions that can be proposed to overcome the problem of finding matching location name by raw location name obtained from Twitter is Web Place-name Dictionary [4]. Web place-name dictionary is made by collecting the sources of the location name, then using it as a database that will be accessible to search for the location names that are similar to a given word or query.

In developing this system, the name and street coordinates are obtained by utilizing the data sets held by OpenStreetMap data on a particular area of Jakarta. The data used can be obtained through



```
<node id="29938999" version="9"
timestamp="2011-01-31T14:07:19Z"
uid="38066" user="ArjanO" changeset
="7145221" lat="-6.1811231" lon=
"106.8227586"/>
```

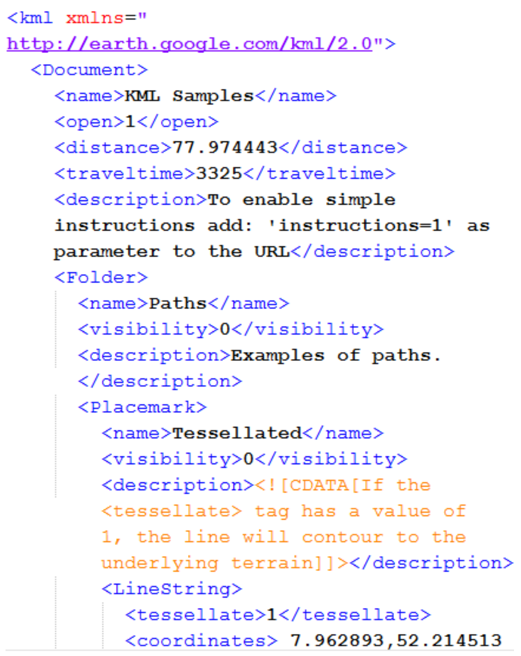
Figure 1. Data structure of node in OSM



```
<way id="40198102" version="4" timestamp=
"2011-10-20T01:47:36Z" uid="76518" user="Firman Hadi"
changeset="9604949">
  <nd ref="484322049"/>
  <nd ref="484321975"/>
  <nd ref="484321937"/>
  <nd ref="1473672608"/>
  <nd ref="484326153"/>
  <nd ref="363186769"/>
  <nd ref="484326063"/>
  <tag k="highway" v="primary"/>
  <tag k="name" v="Kalibata"/>
  <tag k="oneway" v="yes"/>
 </way>
```

Figure 2. Data structure of way in OSM

Figure 3. MapQuest display



```
<kml xmlns="
http://earth.google.com/kml/2.0">
  <Document>
    <name>KML Samples</name>
    <open>1</open>
    <distance>77.974443</distance>
    <traveltime>3325</traveltime>
    <description>To enable simple
    instructions add: 'instructions=1' as
    parameter to the URL</description>
    <Folder>
      <name>Paths</name>
      <visibility>0</visibility>
      <description>Examples of paths.
      </description>
      <Placemark>
        <name>Tessellated</name>
        <visibility>0</visibility>
        <description><![CDATA[If the
        <tessellate> tag has a value of
        1, the line will contour to the
        underlying terrain]]></description>
        <LineString>
          <tessellate>1</tessellate>
          <coordinates> 7.962893,52.214513
```

Figure 4. kml file example

http://downloads.cloudmade.com/asia/southeastern_asia/indonesia/jakarta_raya/jakarta_raya.highway.osm.bz2. Data that will be used in this study is only the primary and secondary data path of the street that is known by tags `<tag k=”highway v=”primary”/>` and `<tag k=”highway” v=”secondary” />`.

String Matching Algorithm

String Matching Algorithm is an algorithm for searching string that have the highest similarity to the given query string. This algorithm is used to solve the typo and abbreviation problem that often happens on Indonesian Tweets.

String Matching Algorithm consists of two main processes, search string algorithm to find candidate result and the calculation of similarity between each candidate string and query string. The string matching algorithm flow details can be seen in Figure 6.

Q-grams method is an algorithm that is often used in search string candidates. Q-grams chops the string into several substrings with each substring consists of q characters. Q-grams method is divided into two types, overlapping and consecutive. In the overlapping Q-Grams, a q-gram will begin at each position in the string. While for consecutive Q-Grams, a q-gram will begin at each position that is multiple of q [10]. On a String "juanda" overlapping 3 - grams will yield {(jua), (uan), (and), (nda)}, whereas if using consecutive 3 - grams will yield {(jua), (nda)}. The value of Q can be changed as needed, but in this study we used Q = 3. This is based on the characteristics of words in Indonesian and street naming in Indonesia.

There are three methods commonly used to calculate the similarity between two strings; Levenshtein Distance, Jaccard, and Cosine. Levenshtein Distance algorithm or often called Edit distance describes the similarity value between two strings based on minimal number of insert, delete, and substitution process to transform a string into another string [11] [12].

Edit distance $ed(x, y)$ between string $x = x_1, \dots, x_m$ and $y = y_1, \dots, y_n$ where $x, y \in \Sigma^*$ and Σ is alphabetic character. We can construct a matrix with size $M_{1\dots m+1, 1\dots n+1}$ to calculate $ed(x, y)$ where the value of edit distance is at $M_{m+1,n+1}$. The calculation of each cell at the position $M_{i,j}$ is calculated based on equation 1, where the value of

$$\delta(a, b) = 0 \; if \; a = b \; and \; 1 \; if \; a \neq b$$

$$M_{1,1} \leftarrow 0$$
$$M_{i,j} \leftarrow min \begin{cases} M_{i-1,j} + 1 \\ M_{i,j-1} + 1 \\ M_{i-1,j-1} + \delta(x_i, y_j) \end{cases} \qquad (1)$$

Even if we get the value of the edit distance, we have not been able to assess whether the two strings are similar or not because the value of the edit distance is relative to the length of the string. We need the equation to calculate how similar the two strings. One of the equations that can be used is

$$sim(s_1, s_2) = \frac{\max(|s_1|, |s_2|) - ed(s_1, s_2)}{\max(|s_1|, |s_2|)} \qquad (2)$$

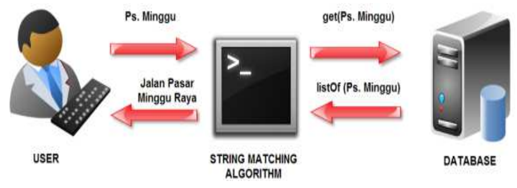


Figure 5. String Matching Algorithm Process

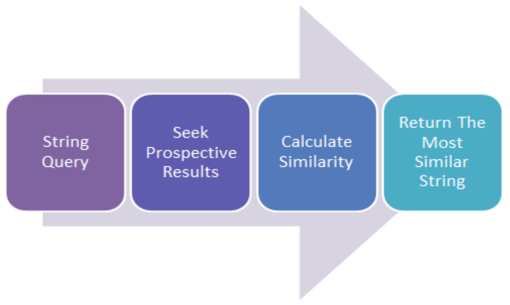


Figure 6. String Matching Algorithm Flow Details

Unlike the Levenshtein Distance algorithm, Algorithm Jaccard and Cosine do not count from letter to letter but the grams that are formed. Set X is the result of Q-grams method to first String, and Set Y is the result of Q-grams method to second String. Jaccard algorithm calculates the similarity value by the equation:

$$Jaccard(x, y) = \frac{|X \cap Y|}{|X \cup Y|} \quad (3)$$

Cosine algorithm calculates the similarity value by the equation:

$$Cosine(x, y) = \frac{|X \cap Y|}{\sqrt{|X||Y|}} \quad (4)$$

In this study, we only use Levenshtein Distance algorithm because the sequence of the Q-gram made by Q-gram method is an important point.

Efficient Approximate Candidate String

In developing this kind of system, efficiency is certainly an important factor. One method that can be applied is the Scan Count and Divide Skip, which is an algorithm proposed by Li in 2008 [13].

Scan Count Algorithm counts the number of occurrences of each string in the database that contains the elements of a given set of queries grams. Scan Count Algorithm is then developed into Divide Skip. Divide Skip is an algorithm where filters are added into Scan Count algorithm in advance to speed up the calculation. We applied some modifications to suit the environment of the experiments conducted. The pseudo code of Divide Skip algorithm can be seen in Figure 6.

```
Input :   lists and threshold T
Output :  List of string that appears at least T times
1         Initiate an empty list R
2         Create a L long list which is a collection of L
          longest list of the input list
3         Create a L short list which is a collection of the
          remaining list (short lists)
4         Use Scan Count Algorithm for L short to find a
          collection of records that appears at least T-L
          times
5         FOR (each record r that found) {
6             FOR (each lists in L long )
7                 Check whether r appears in the list
8             IF (r appears > = T times on the whole list)
9                 Add r to R;
10        }
11        RETURN R;
```

Figure 7. Pseudocode of Divide Skip Algorithm

In general, Divide Skip algorithm discards non potential data by dividing the entire list into two parts, list with a little number of members (L short) and list with the lot number of members (L long). The main problem in this algorithm is the selection of the value of L so that the algorithms can run properly and efficiently.

Li stated that the value of L can be found using the equation 5:

$$L = \frac{T}{\mu \log M + 1} \quad (5)$$

M is the length of the longest list, and μ is an independent coefficients. Based on the results of the experiment that have been carried out by Li, the value of μ is 0.0085.

3. Results and Analysis

Problem Identification

Problem identification is done by analyzing the pattern of the provision of information via Tweet from the account @TMCPoldaMetro and methods that have been applied to map the location of the street based on information from social media. The methods that have been applied in Indonesia is utilizing Google API with location or street name as an input.

The use of Google's API does not guarantee that the precise location can be found as a result of the presence of the typos and abbreviation of the name of the location/street.

There are three cases that must be solved in this system. The case with one location name, two locations name, and three locations name.

Tweet with one location name usually informs the overall road conditions (Figure 8). Tweet with two locations usually informs the certain path on the way (Figure 9). Tweet with three locations

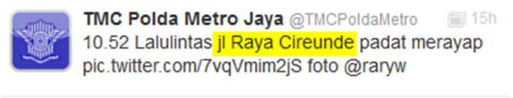

TMC Polda Metro Jaya @TMCPoldaMetro 15h
10.52 Lalulintas jl Raya Cireunde padat merayap pic.twitter.com/7vqVmim2jS foto @raryw

Figure 8. Tweet with one location

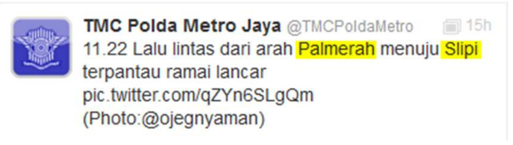

TMC Polda Metro Jaya @TMCPoldaMetro 15h
11.22 Lalu lintas dari arah Palmerah menuju Slipi terpantau ramai lancar pic.twitter.com/qZYn6SLgQm (Photo:@ojegnyaman)

Figure 9. Tweet with two location

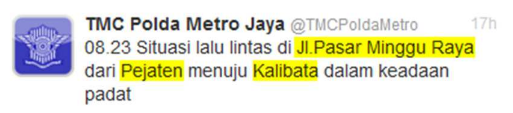

TMC Polda Metro Jaya @TMCPoldaMetro 17h
08.23 Situasi lalu lintas di Jl.Pasar Minggu Raya dari Pejaten menuju Kalibata dalam keadaan padat

Figure 10. Tweet with three location

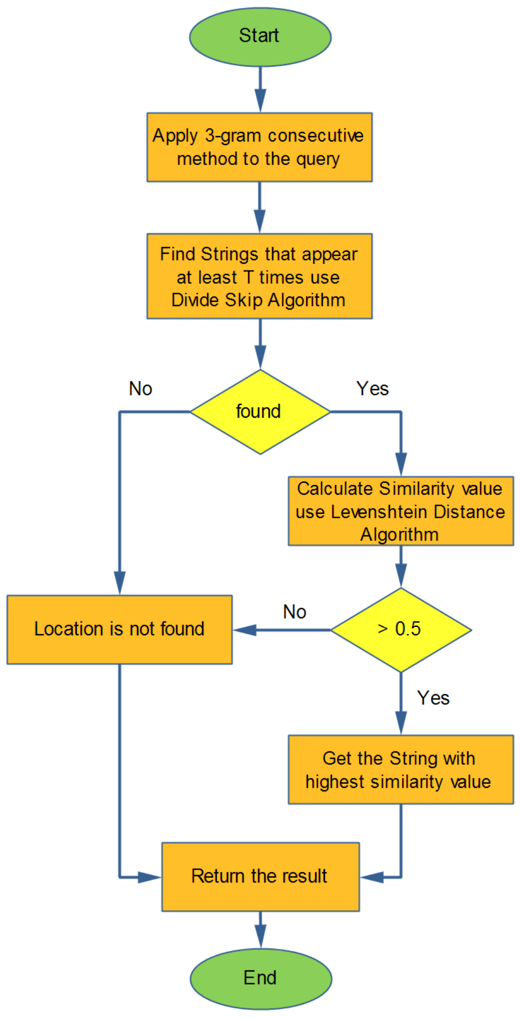

Start
Apply 3-gram consecutive method to the query
Find Strings that appear at least T times use Divide Skip Algorithm
found
No
Yes
Calculate Similarity value use Levenshtein Distance Algorithm
> 0.5
No
Yes
Location is not found
Get the String with highest similarity value
Return the result
End

Figure 11. Flow diagram for one input query

usually inform the way in certain path that intersects with some other way (Figure 10).

System Requirement

In developing a system that is able to provide accurate coordinate information, a fairly in-depth analysis of the pattern of information provided through Twitter is required. In general, the system is built to meet the following specifications:

- The system is able to search the street name accurately although there are some typos or abbreviation.
- The system is able to locate specific coordinates on the road with one input location na-me.
- The system is able to locate specific coordinates on the road with two input location na-me.
- The system is able to locate specific coordinates on the road with three input location name.

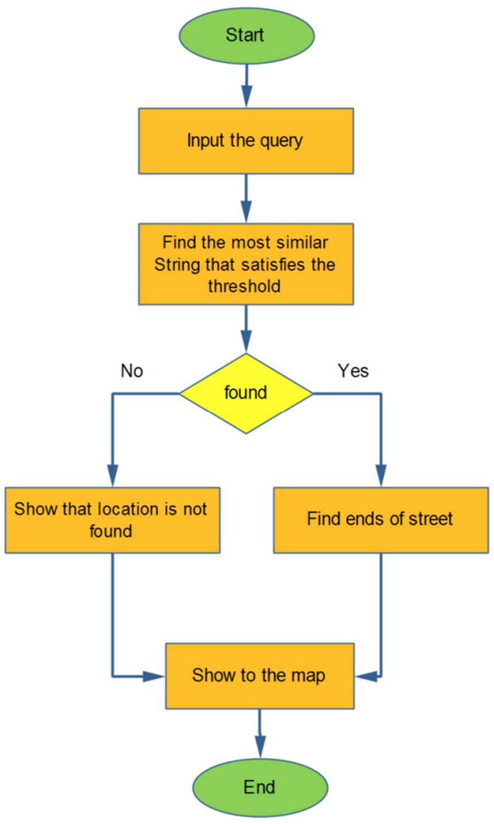

Start
Input the query
Find the most similar String that satisfies the threshold
found
No
Yes
Show that location is not found
Find ends of street
Show to the map
End

Figure 12. Flow diagram for one input query

- The system is able to visualize on a map the coordinates of the intended path.

Systems Design

In developing this system, the primary focus is the process of finding a location or street name that corresponds to what is meant by information providers. The flow diagram is shown in figure 9. The value of T that is used in the flow can be obtained by calculating the result of (query.length-2)/2. This means that there are at least a candidate string containing half of gram that are formed from a given query.

The sistem has four main flows; flow on the input processing, flow on finding coordinate by one location name, flow on finding coordinate by two locations name, and the last is flow on finding coordinate by three locations name. The system will return the coordinates that consist of latitude and longitude which considered as the location referred by a Tweet.

On the flow of processing one input query, the coordinates that will be taken are the start point and end point of a path. This decision is made because the location that is meant usually covers the entire road. If the road which is meant is a two-way street, then the system will do the search process

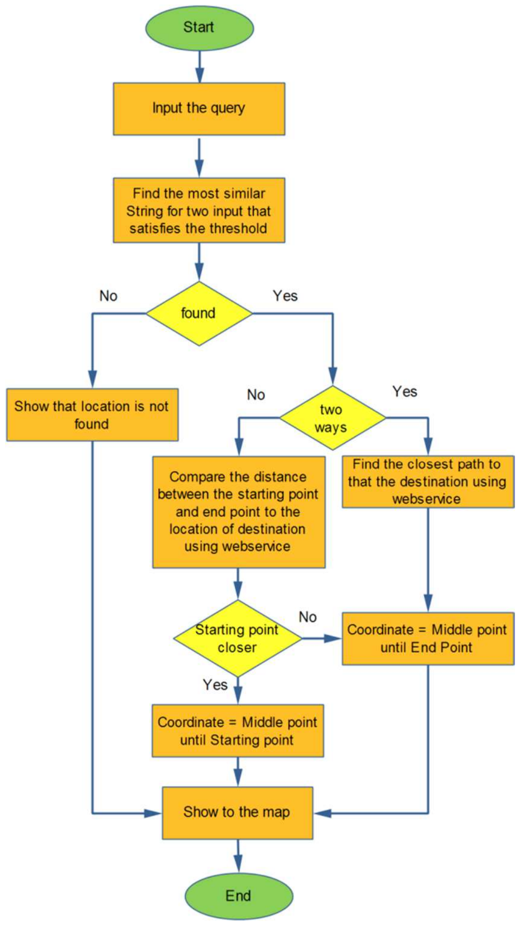

Start
Input the query
Find the most similar String for two input that satisfies the threshold
found
No
Yes
Show that location is not found
two ways
No
Yes
Compare the distance between the starting point and end point to the location of destination using webservice
Find the closest path to that the destination using webservice
Starting point closer
No
Yes
Coordinate = Middle point until End Point
Coordinate = Middle point until Starting point
Show to the map
End

Figure 13. Flow diagram for two input query

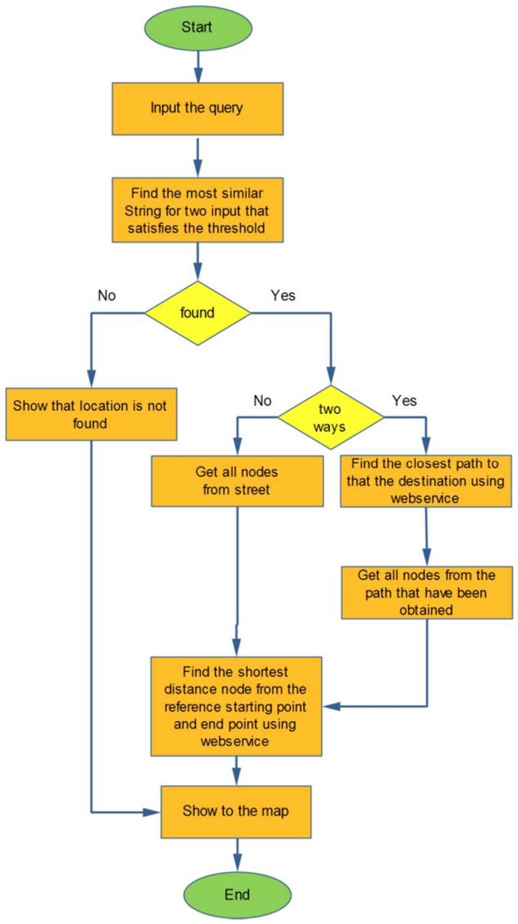

Start
Input the query
Find the most similar String for two input that satisfies the threshold
found
No
Yes
Show that location is not found
two ways
No
Yes
Get all nodes from street
Find the closest path to that the destination using webservice
Get all nodes from the path that have been obtained
Find the shortest distance node from the reference starting point and end point using webservice
Show to the map
End

Figure 14. Flow diagram for three input query

twice to obtain the coordinates of the starting point and end point on each track. Then the coordinates will be visualized into a map.

Contrast to one input query processing, two input query processing utilizes yournavigation.org web service to determine the most appropriate location that is provided by a Tweet. There are two possibilities in this case; first the case where the location is from middle point to starting point or the second case is from middle point to end point on the road.

The determination of the case is based on the reference point given by the second query. The first query is a street that will be searched for its coordinates. If the road on the first query is a road with two ways, we must search which lane that is closest to the reference point on the second query.

The case of two input queries is similar to the case of three input queries. We use the services of yournavigation.org to determine the specific coordinates on the way. In the case of three input query, the system will search the node that has shortest distance to the first reference (query 2) and the node with the shortest distance to the second reference point (query 3).

Implementation

There are some applications that are used in building this system such as Tomcat server, Xampp and MySQL. Implementation of the code is done by using the Java language. Figure 13 is an overall system view.

The prototype consists of two component; input menu for one input, two inputs, three inputs (figure 14) and the map to visualize the coordinate (figure 15).

The given input on the field will be sent to the server and processed to produce the coressponding location as shown in figure 15.

Testing

Testing of the prototype system is done by conducting a blackbox testing against the street name/location name as input. There are 100 street name /

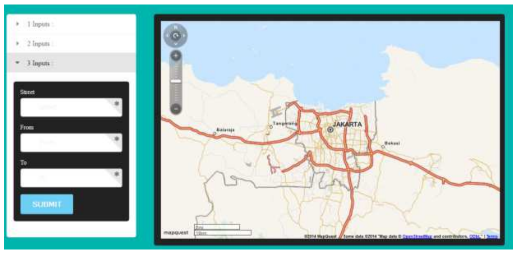


Figure 15. Prototype of system

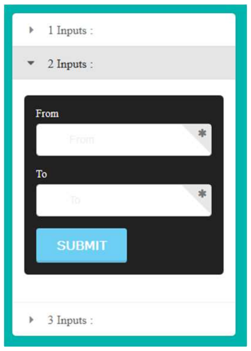


Figure 17. Input Query Field

location name that are taken randomly from the data that ever provided in Tweets by @TMCPolda-Metro from May to August 2013.

TABLE 1
EXPERIMENT RESULT SAMPLE

Input	Output
Jl H R Rsauna Said	data tidak ditemukan
Pos Tanung Priok	taman stasiun tanjung priok
JlThamrin	mh thamrin
jl Danau Sunter Utara	danau sunter utara
jalan Sunter Utara	danau sunter utara
Slipi Jaya	Slipi Jaya
Jl Matraman	matraman raya
Tomang	tomang raya
JlPos Pengumben	pos pengumben
JlGatot Subroto	gatot soebroto
Kalibata	kalibata
Cililitan	cililitan besar
Tomang	tomang raya
JlRasuna Said	rasuna said
JlJatibaru	jatibaru
Jl Gajah Mada	gajah mada
Slipi	Slipi Jaya
Jl Panglima Polim	panglima polim raya
Jl Sudirman	sudirman
Jl I Gusti Nghrah Rai	i gusti ngurah rai
Mangga Besar	mangga besar
jln raya bogor	raya bogor
Jl Margonda Raya	jalan raya margonda
Fatmawati	fatmawati
Pasar Jumat	pasar jumat
Pancoran	pancoran
JlPramuka Raya	pramuka
JlImam Bonjol	imam bonjol
jalan Raya Kalimalang	kalimalang
jalan Raya Pesanggrahan	pesanggrahan

Experiments show that the prototype system

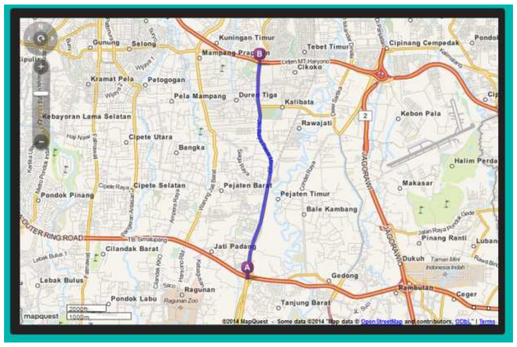


Figure 16. Location visualization

can recognize the input with an accuracy of 90% on 100 test cases.

4. Conclusion

Development of traffic information system using web place-name dictionary to determine the location mentioned in a tweet is able to handle the typos and abbrevations that are often done by Indonesian society. The system is able to reduce the inaccuracy that often occurs as a result of existing APIs such as Google Maps that could not solve the typos and abbreviations used by Indonesian people. In black box testing, the system is able to produce an accuracy of 90% with 100 data set that taken randomly.

Utilization of web place-name dictionary can be developed in a variety of other systems which utilize the name of the location mentioned through Twitter to be processed. In this study, the system still has weaknesses because there are no location name (non street name) on OpenStreetName so the problem must be resolved.

5. Acknowledgment

This research is partly supported by Master-plan Percepatan dan Perluasan Pembangunan Eko-nomi Indonesia (MP3EI) and Hibah Riset PUPT UI (Awal) 2014. The grants have been made possible by Indonesia's Ministry of Education and Culture No:3715/H2.R12/ HKP.05.00/2012 and Universitas Indonesia.

Reference

[1] PU, "Panjang Jalan Jakarta Baru Penuhi 60 Persen Kebutuhan," [Online]. Available: www.pu.go.id/main/view_pdf/7843. [Accessed 17 November 2013].

[2] BPS, "Perkembangan jumlah kendaraan bermotor menurut jenis tahun 1987-2011," [Online]. Available: http://www.bps.go.id/

tab_sub/view.php?tabel=1&id_subyek=17&notab=. [Accessed 17 November 2013].

[3] D. J. Widiantono, "Kebijakan dan Strategi Penanganan Kemacetan Lalulintas di Perkotaan," *tataruang,* vol. 2, no. 1, pp. 65--70, March 2008.

[4] T. a. M. Y. a. Y. T. a. C. N. a. N. K. Sakaki, "Real-time event extraction for driving information from social sensors," in *Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2012 IEEE International Conference on*, 2012.

[5] S. Endarnoto, S. Pradipta, A. Nugroho and J. Purnama, "Traffic Condition Information Extraction amp; Visualization from Social Media Twitter for Android Mobile Application," in *Electrical Engineering and Informatics (ICEEI), 2011 International Conference on*, 2011.

[6] OpenStreetMap, "About," [Online]. Available: http://wiki.openstreetmap.org/wiki/About. [Accessed 18 November 2013].

[7] OpenStreetMap, "MapQuest," [Online]. Available: http://wiki.openstreetmap.org/wiki/MapQuest. [Accessed 18 November 2013].

[8] MapQuest, "MapQuest," [Online]. Available: http://company.mapquest.com. [Accessed 18 November 2013].

[9] I. A. Hamid, "YOURS navigation routing system-Lambertus Ijsselstein," *Society of Cartographer,* no. November, pp. 20-22, 2009.

[10] C. Miao, G. Chang and X. Wang, "Filtering Based Multiple String Matching Algorithm Combining q-Grams and BNDM," in *Genetic and Evolutionary Computing (ICGEC), 2010 Fourth International Conference on,* 2010.

[11] V. I. Levenshtein, "{Binary codes capable of correcting deletions, insertions, and reversals}," *Soviet Physics Doklady,* vol. 10, no. 8, pp. 707--710, 1966.

[12] G. Navarro and M. Raffinot, Flexible Pattern Matching in Strings: Practical On-line Search Algorithms for Texts and Biological Sequences, New York, NY, USA: Cambridge University Press, 2002.

[13] Chen Li, Jiaheng Lu and Yiming Lu, "Efficient Merging and Filtering Algorithms for Approximate String Searches," in *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, 2008.

[14] Bazis DKI Jakarta, "Petunjuk Praktis Bagi Mustahik," [Online]. Available: http://www.bazisdki.go.id/panduan/zakat12/85-petunjuk-praktis-bagi-mustahik. [Accessed 27 Mei 2013].

[15] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. H. Witten, "The WEKA Data Mining Software: An Update," *SIGKDD Explorations*, vol. Volume 11, no. Issue 1, pp. 10-18, June 2009.

[16] S. Hardikar, A. Shrivastava and V. Choudhary, "Comparison between ID3 and C4.5 in Contrast to IDS," *VSRD International Journal of Comptuer Science & Information Technology,* vol. Vol. 2, no. 7, pp. 659-667, 2012.

[17] A. K. Sharma and S. Sahni, "A Comparative Study of Classification Algorithms for Spam Email Data Analysis," *International Journal on Computer Science and Engineering (IJCSE),* vol. Vol. 3 No. 5, pp. 1890-1895, May 2011.

[18] D. Upton, CodeIgniter for Rapid PHP Application Development, Birmingham: Packt Publishing, 2007.

[19] M. J. Berry and G. S. Linoff, Data Mining Techniques For Marketing, Sales, Customer Relationship Management, Second ed., Indianapolis, Indiana: Wiley Publishing, Inc., 2004.

[20] P.-N. Tan, M. Steinbach and V. Kumar, Introduction to Data Mining, 1st ed., Boston: Pearson Addison-Wesley, 2006.

[21] J. R. Quinlan, C4.5: Programs for Machine Learning, USA: Morgan Kaufmann, 1993.

[22] J. Han and M. Kamber, Data mining Concepts and Techniques, Second ed., San Fransisco: Morgan Kauffman, 2006.

[23] Random.org, "RANDOM.ORG," 2012. [Online]. Available: http://www.random.org/lists/. [Accessed 8 June 2013].

[24] I. H. Witten, E. Frank and M. A. Hall, Data Mining: Practical Machine Learning Tools and Techniques, 3rd ed., Burlington: Morgan Kaufman, 2011.

[25] E. Turban, J. E. Aronson and T. P. Liang, Decision Support Systems and Intelligent Systems, USA: Pearson Education Inc., 2005.

[26] C. Vercellis, Business Intelligence: Data Mining and Optimization for Decision Making, United Kingdom: John Wiley & Sons Ltd, 2009.

[27] M. Kantardzic, Data Mining: Concepts, Models, Methods, and Algorithms, 2nd ed., Hoboken, NJ, USA: John Wiley & Sons, Inc., 2011.

[28] A. Z. Zaenal Abidin, "Implementasi Algoritma C4.5 Untuk Menentukan Tingkat Bahaya Tsunami," *Seminar Nasional Informatika,* pp. ISSN: 1979-2328, 2011.

[29] N. G. A. P. H. Saptarini and R. Wardoyo, "Penggunaan Algoritma C4.5 Dan Logika Fuzzy Untuk Klasifikasi Talenta Karyawan (Studi Kasus: Politeknik Negeri Bali)," *Jurnal Matrix,* vol. 2, pp. 95-100, Juli 2012.

[30] M. N. Anyanwu and S. G. Shiva, "Comparative Analysis of Serial Decision Tree Classification Algorithms," *International Journal of Computer Science and Security (IJCSS),* vol. 3, no. 3, pp. 230-240, 2009.

[31] D. D. Patil, V. M. Wadhai and J. A. Gokhale, "Evaluation of Decision Tree Pruning Algorithms for Complexity and Classification Accuracy," *International Journal of Computer Applications,* Vols. Volume 11 - No.2, pp. 23-30, December 2010.

[32] EllisLab, Inc., "Application Flow Chart : CodeIgniter User Guide," 2012. [Online]. Available: http://ellislab.com/codeigniter/user-guide/overview/appflow.html. [Accessed 20 June 2013].

[33] D. T. Larose, Discovering Knowledge in Data: an Introduction to Data Mining, Hoboken, New Jersey: John Wiley and Sons, Inc., 2005.

[34] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand and D. Steinberg, "Top 10 Algorithms in Data Mining," *Knowledge and Information Systems,* pp. 1-37, 2007.

[35] Y. Yusuf W., "Perbandingan Performansi Algoritma Decision Tree C5.0, CART, dan CHAID: Kasus Prediksi Status Resiko Kredit di Bank X," *Seminar Nasional Aplikasi Teknologi Informasi (SNATI),* pp. 59-62, 2007.

[36] H. Shi, "Best-first Decision Tree Learning," Hamilton, New Zealand, 2007.

[37] A. Syamsuddin, "Algoritma Decision Tree C4.5," 23 September 2012. [Online]. Available: http://blogs.itb.ac.id/aiceware/2012/09/23/algoritma-decision-tree-c4-5/.

[38] J. R. Quinlan, "Simplifying Decision Trees," *International Journal of Human-Computer Studies,* vol. Volume 51, no. Issue 2, p. 497–510, August 1999.

[39] Machine Learning Group at the University of Waikato, "Weka 3: Data Mining Software in Java," [Online]. Available: http://www.cs.waikato.ac.nz/ml/weka/.