KLASIFIKASI *TWEET* KONDISI LALU LINTAS KOTA JAKARTA DENGAN PENERAPAN METODE *K-NEAREST NEIGHBOR*

Ziza Amira Syafini[1], Muhammad Nasrun[2], Casi Setianingsih[3]

[1, 2, 3]Fakultas Teknik Elektro, Universitas Telkom, Bandung, Indonesia
[1]amira.syafini@gmail.com,[2]nasrun@telkomuniversity.ac.id,
[3]setiacasie@telkomuniversity.ac.id

Abstrak

Setiap tahun, jumlah kendaraan bermotor di Kota Jakarta, Indonesia semakin meningkat. Namun, peningkatan jumlah kendaraan bermotor ini tidak sebanding dengan penambahan ruas jalan. Kondisi ini menyebabkan terganggunya kelancaran lalu lintas dan menimbulkan kemacetan. Untuk mengantipasi kemacetan tersebut, pengguna lalu lintas mencari dan saling bertukar informasi tentang kemacetan di media sosial. Salah satu media sosial yang sering digunakan masyarakat untuk menyebarkan informasi adalah Twitter. Penelitian ini dilakukan untuk memgklasifikasi kondisi lalu lintas berdasarkan data yang didapatkan dari Twitter. Data kemacetan diklasifikasikan menjadi 3 kondisi yaitu lancar, padat dan macet. Metode yang digunakan pada penelitian ini adalah k-*nearest neighbor*. Dari beberapa uji skenario yang dijalankan, didapatkan hasil rata-rata-rata akurasi klasifikasi kondisi lalu lintas di atas 70%. Nilai k yang optimal pada penelitian ini adalah 8.

Kata Kunci: Twitter, kemacetan, *data mining, k-nearest neighbor*

Abstract

Every year, the number of motor vehicles in Jakarta City, Indonesia is increasing. However, the increase in the number of motor vehicles in Jakarta City is not proportional to the area of road segments. This condition causes the disruption of the traffic smoothness and cause some congestion points. To anticipate being caught in traffic jam, traffic users search and exchange information about traffic condition through on social media. One of the social media that people often use to spread information is Twitter. The purpose of this research is to classify traffic condition based on data obtained from Twitter. The data are classified into 3 conditions of traffic, i.e. smooth, moving slower and jammed. The method used in this research is k-nearest neighbor. From several test scenarios that run, the average results of accuracy of traffic condition classification are above 70%. The optimal k value in this research is 8.

***Key Words*: Twitter, traffic, data mining, k-nearest neighbor**

1. Pendahuluan

Salah satu kota yang terjerat dalam masalah kemacetan adalah Kota Jakarta, Indonesia. Selain karena perkembangan mode transportasi, kemacetan juga disebabkan urbanisasi, yaitu perpindahan penduduk dari daerah-daerah kecil ke kota besar. Secara tidak langsung, perpindahan ini menjadikan angka penduduk di kota Jakarta meningkat. Hal ini sangat berpengaruh terhadap kondisi lalu lintas. Berdasarkan data dari Badan Pusat Statistik Jakarta, tingkat pertumbuhan kendaraan dari tahun 2010-2014 yaitu sebanyak 9.93% per tahun tidak sebanding dengan peningkatan luas jalan yang hanya bernilai 0.33% per tahun [1].

Artinya, ketidakseimbangan ini akan berakibat pada kepadatan lalu lintas yang berhujung kepada timbulnya titik-titik kemacetan. Ada tiga cara yang dapat digunakan untuk mengurangi kemacetan lalu lintas [2]; yang pertama adalah menambah baik infrastruktur transportasi dengan memperlebar luas jalan. Akan tetapi, cara ini memerlukan lahan yang banyak dan menelan biaya yang tidak sedikit. Cara kedua adalah mempromosikan kendaraan umum untuk mengurangi kemacetan tetapi cara ini dinilai tidak mudah. Sedangkan cara ketiga adalah dengan menentukan kondisi jalan pada suatu waktu. Hal ini dapat mendorong pengguna lalu lintas untuk

merancang waktu perjalanan mereka.

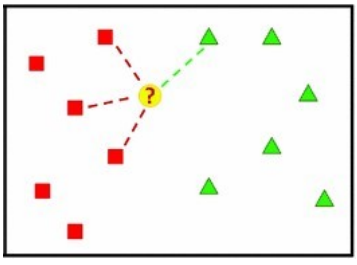
Untuk memprediksi kemacetan lalu lintas, maka diperlukan satu set data kondisi lalu lintas. Salah satu cara untuk mendapatkan data tersebut adalah melalui media sosial. Kondisi lalu lintas dapat diperoleh dengan mudah dari media sosial kerana manusia telah menjadikan media sosial sebagai tempat untuk saling berbagi ide dan informasi. Potensi tersebut dapat digunakan untuk membuat suatu sistem klasifikasi kondisi lalu lintas di Kota Jakarta dengan menggunakan metode k-*Nearest Neighbor* untuk memprediksi kondisi lalu lintas di beberapa ruas jalan.

2. Metode Penelitian

Sistem yang akan dibangun pada penelitian ini adalah sistem yang mampu mendeteksi kondisi lalu lintas di kota Jakarta dengan mengimplementasikan salah satu teknik *data mining*. Teknik yang digunakan adalah klasifikasi menggunakan algoritma k-*nearest neighbor*. Studi kasus yang akan dibahas dalam penelitian ini adalah kondisi lalu lintas yang diambil dari beberapa akun Twitter terpercaya. Input yang akan diberikan pada sistem ini adalah hari, waktu dan nama tempat.

2.1 Teorema k-*Nearest Neighbor*

Salah satu algoritma yang sering digunakan dalam *data mining* adalah k-*nearest neighbor* (k-NN). Algoritma k-NN adalah sebuah metode untuk melakukan klasifikasi terhadap suatu objek berdasarkan data pembelajaran yang jaraknya paling dekat dengan objek tersebut [3]. k-NN merupakan algoritma *supervised learning*, dimana hasil dari *query instance* yang baru diklasifikasikan berdasarkan kategori mayoritas pada algoritma k-NN. Kelas yang paling banyak muncul nantinya akan menjadi kelas hasil dari klasifikasi. Nilai k yang optimum adalah ketika nilai k tersebut dapat meminimumkan *error* klasifikasi.



Gambar 1. Sebaran data dalam algoritma k-NN

Gambar 1 menunjukkan sebaran data dalam algoritma k-NN yang terdiri dari 2 kelas, yaitu kelas segi tiga dan segi empat. Dalam hal ini, satu data uji akan diklasifikan berdasarkan data training. Sebagai contoh, k yang digunakan adalah 4. Maka, data uji akan mencari 4 *data training* yang berdekatan dengan data uji. Setelah dilakukan pencarian, ditemukan 1 titik segi tiga dan 3 titik persegi. Karena jumlah titik persegi lebih banyak dari titik segi tiga, maka data uji akan diklasifikasikan sebagai kelas titik persegi.

2.2 Pengambilan Data

Data yang akan digunakan pada penelitian ini adalah *tweet* dari beberapa akun *Twitter* yang dinilai mempunyai tingkat validasi yang tinggi seperti @TMCPoldaMetro dan @lewatmana. *Tweet* yang digunakan merupakan *tweet* dalam rentang waktu setahun berawal dari tanggal 01-09-2016 sampai tanggal 31-09-2017. Data-data tersebut didapat dengan cara mengekstraknya menggunakan API (*Application Program Interface*) resmi dari Twitter. Namun, pihak Twitter telah membatasi jumlah pengambilan data Twitter dengan jumlah 200 *tweet* per halaman.

Untuk mengatasi masalah ini, penulis menggunakan sebuah *open source project* yang dikembangkan oleh Jefferson Henrique [4]. *Open source project* ini memiliki kapabilitas untuk mengambil *tweet* melebihi jumlah yang ditetapkan. Proses ini memberikan atribut *tweet* berupa *permalink, username, text, date, retweets, favorites, mentions, hashtags, id tweet* dan *geo*.

2.3 Seleksi Data

Data yang diambil dari beberapa akun yang dinilai memiliki tingkat validitas tinggi, yang berlangsung selama 12 bulan tersebut tidak semua *tweet* mengandung informasi kondisi lalu lintas, maka dilakukan seleksi data berdasarkan *keyword* yang menggambarkan kondisi lalu lintas, agar data yang mengandung informasi kondisi lalu lintas saja. *Keyword* terpilih yang akan menjadi parameter *tweet* adalah lancar, lengang, padat, padat merayap, ramai, ramai lancar, macet dan tersendat. Selain itu, hanya beberapa lokasi jalan di Kota Jakarta yang dipilih berdasarkan frekuensi penyampaian informasi lalu lintas. Jumlah total *data set* yang digunakan pada penelitian ini adalah sebanyak 1481 yang terdiri dari 17 ruas jalan di Kota Jakarta.

2.4 *Pre-Processing Data*

Pre-processing data perlu dilakukan karena data yang didapatkan bersifat tidak konsisten dan tidak memiliki pola yang sama, sehingga akan menyebabkan *data mining* tidak efektif. Tahap *pre-processing* yang digunakan pada *data mining* adalah sebagai berikut [5]:

1. *Case folding*: mengubah semua masukan huruf menjadi huruf kecil (*lower case*). Selain mengubah semua huruf menjadi huruf kecil, karakter selain huruf dan angka dihilangkan

kecuali tanda titik dan titik dua, karena kedua tanda tersebut akan digunakan untuk informasi waktu.

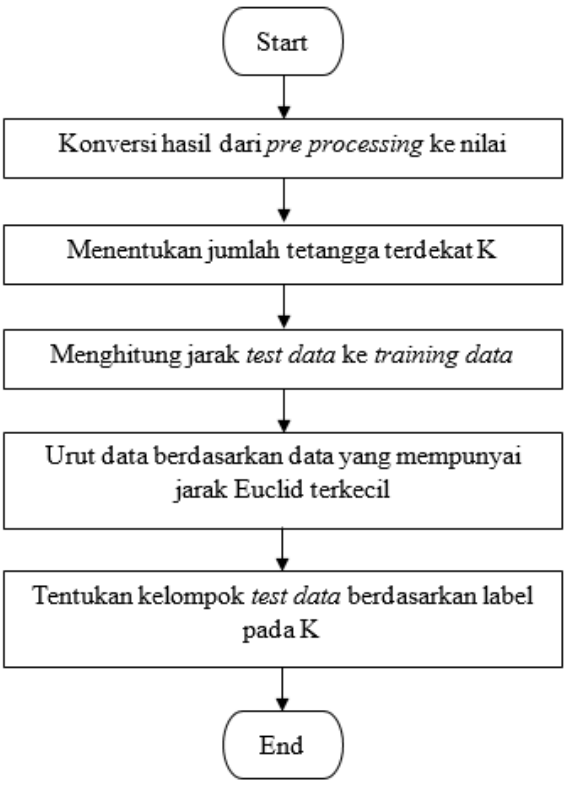
2. *Cleaning*: kata-kata yang tidak diperlukan untuk proses selanjutnya akan dihilangkan. Kata-kata yang dimaksud adalah kata-kata yang tidak mengandung informasi waktu, lokasi dan kondisi lalu lintas.
3. *Data Transformation*: mengganti beberapa kata ke dalam suatu representasi yang sama untuk mempermudah proses selanjutnya. Kata yang akan ditransformasi adalah kata yang sinonim dengan kata 'arah' antaranya 'menuju', 'arah ke' dan 'mengarah'. Kata-kata tersebut akan ditransformasi menjadi karakter '-'. Selain itu, akan dilakukan konversi waktu ke dalam bentuk rentang waktu dan konversi tanggal ke hari.
4. *Tokenizing*: pemotongan string berdasarkan setiap kata yang menyusunnya menjadi bentuk *array*. *List array* tersebut akan dimasukkan ke dalam tabel basis data berdasarkan atributnya. Ilustrasi *pre-processing* data dapat dilihat pada Tabel 1.

Setelah proses *tokenizing*, data akan disimpan ke database di mana *array* (1) akan dimasukkan ke atribut waktu, *array* (2) akan dimasukkan ke dalam atribut lokasi, *array* (3) akan dimasukkan ke dalam atribut kondisi dan *array* (4) akan dimasukkan ke dalam atribut hari. Sebelum dilakukan klasifikasi menggunakan algoritma k-*nearest neighbor*, kondisi lalu lintas akan dikelompokkan menjadi tiga kelas, seperti dijelaskan pada Tabel 2 dan Tabel 3 secara berturut-turut.

2.5 Algoritma k-*Nearest Neighbor*

Metode yang akan digunakan dalam penelitian ini adalah k-*Nearest Neighbor*. Gambar 2. adalah *flowchart* dari algoritma k-*Nearest Neighbor*:

1. Hasil dari *pre-processing* akan menjadi *training data* yang bakal menjadi acuan terhadap test data. Hasil tersebut adalah hari, waktu, lokasi dan kondisi lalu lintas. Hari dan waktu selanjutnya akan dikonversi ke dalam bentuk menjadi nilai tertentu. Angka tersebut akan menjadi variabel pada perhitungan *euclidean distance*. Masing-masing atribut hari dan waktu diberi nilai yang dapat dilihat pada Tabel 4. dan Tabel 5. secara berturut-turut.
2. Penentuan nilai k (k adalah jumlah tetangga terdekat) yang terbaik tergantung pada data. Nilai k yang tinggi akan mengurangi efek *noise* pada klasifikasi, tetapi membuat batasan antara setiap klasifikasi menjadi lebih kabur.
3. Jarak antara *data test* dengan *data training* dihitung dengan cara mengukur jarak antara titik



Gambar 2. *Flowchart* algoritma k-*nearest neighbor*

yang merepresentasikan *data testing* dengan semua titik yang merepresentasikan *data training* dengan rumus *uclidean distance*. *Distance d*, seperti diperlihatkan pada persamaan (1), dimana x dan y adalah titik pada ruang vektor n dimensi sedangkan x_i dan y_i adalah besaran skalar dalam ruang vektor n dimensi.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \qquad (1)$$

4. Jarak tersebut diurutkan dan dilakukan penentuan tetangga mana yang terdekat berdasarkan jarak minimum ke-k.
5. Kategori mayoritas dari tetangga yang terdekat digunakan sebagai hasil klasifikasi dari *data test*.

3. Hasil dan Pembahasan

3.1 Pengukuran Kinerja Sistem

Untuk mengetahui keberhasilan suatu sistem, maka perlu dilakukan pengukuran kinerja terhadap sistem tersebut. Pada penelitian ini, pengukuran kinerja klasifikasi dilakukan menggunakan *confusion matrix*. Pada dasarnya *confusion matrix* mengandung informasi yang membandingkan hasil klasifikasi yang dikerjakan oleh sistem dengan hasil klasifikasi yang seharusnya, dan digunakan untuk mengukur kinerja

Tabel 1. ***Ilustrasi pre-processing***

Process	Data
Raw Data	14.38 Situasi lalu lintas dari Slipi menuju Semanggi ramai lancar
After case folding	14.38 situasi lalu lintas dari slipi menuju semanggi ramai lancar
After cleaning	14.38 slipi menuju semanggi ramai lancar
After data transformation	14:01-15:00 slipi - semanggi ramai lancar Kamis
After tokenizing	(1) 14:01-15:00 (2) slipi-semanggi (3) ramai lancar (4) Kamis

Tabel 2. Pengelompokan Kondisi Lalu Lintas

Label	Kondisi
Lancar	lancar, lengang
Padat	ramai, ramai lancar, padat, padat merayap
Macet	tersendat, macet

Tabel 3. Contoh Tabel Hasil Akhir Data *Pre-processing*

No	Hari	Waktu	Lokasi	Kondisi
1	Kamis	14:01-15:00	slipi-semnaggi	padat

Tabel 4. Konversi Hari

Nilai	Hari
1	Senin
2	Selasa
3	Rabu
4	Kamis
5	Jum'at
6	Sabtu
7	Minggu

Tabel 5. Konversi Hari

Nilai	Hari	Nilai	Hari
1	00:01-01:00	13	12:01-13:00
2	01:01-02:00	14	13:01-14:00
3	02:01-03:00	15	14:01-15:00
4	03:01-04:00	16	15:01-16:00
5	04:01-05:00	17	16:01-17:00
6	05:01-06:00	18	17:01-18:00
7	06:01-07:00	19	18:01-19:00
8	07:01-08:00	20	19:01-20:00
9	09:01-10:00	21	20:01-21:00
10	10:00-11:00	22	21:01-22:00
11	10:01-11:00	23	22:01-23:00
12	11:01-12:00	24	23:01-24:00

classifier tersebut [7]. *Confusion matrix* yang akan digunakan dapat dilihat pada Tabel 6.

Tabel 6. Tabel *confusion matrix*

		Kelas Hasil Klasifikasi			
		Lancar	Padat	Macet	
Kelas Sebenarnya	Lancar	TP_ Lancar	*Error*	*Error*	Total Lancar
	Padat	*Error*	TP_ Padat	*Error*	Total Padat
	Macet	*Error*	*Error*	TP_ Macet	Total Macet
		Terklasifikasi Lancar	Terklasifikasi Padat	Terklasifikasi Macet	Total Data

True Positive (TP) adalah data yang diklasifikasikan sesuai dengan kelas sebenarnya, sedangkan *error* adalah data yang tidak diklasifikasikan sesuai dengan kelas sebenarnya. Nilai akurasi didapatkan dengan menggunakan rumus (2) berikut:

$$Akurasi = \left(\frac{TP-lancar + TP-padat + TP-macet}{Total\ data} \right) 100\% \quad (2)$$

Pengujian sistem ini dilakukan sebanyak 5 kali dengan skenario yang terdapat pada Tabel 1. Hasil pengujian tersebut dapat dilihat pada Tabel 7.

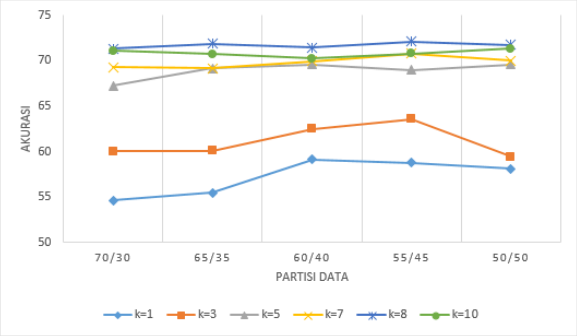
Tabel 7. Skenario Pengujian Kinerja Sistem

Pengujian	Training Data (%)	Testing Data (%)
1	50	50
2	55	45
3	60	40
4	65	35
5	70	30

3.2 Pengujian Pengaruh Nilai k Terhadap Akurasi

Pada pengujian ini, digunakan 6 nilai k pada setiap skenario dengan porsi *data training* dan *data*

testing yang berbeda-beda dari keseluruhan 1481 data. Nilai-nilai tersebut adalah 1, 3, 5, 7, 8 dan 10. Pengujian ini bertujuan untuk mengetahui pengaruh banyaknya tetangga terhadap akurasi dan untuk mengetahui nilai k yang optimal digunakan untuk mengklasifikasi kondisi lalu lintas di kota Jakarta menggunakan metode k-*nearest neighbor*. Hasil pengujian pengaruh nilai k terhadap akurasi dapat di lihat pada Gambar 3.



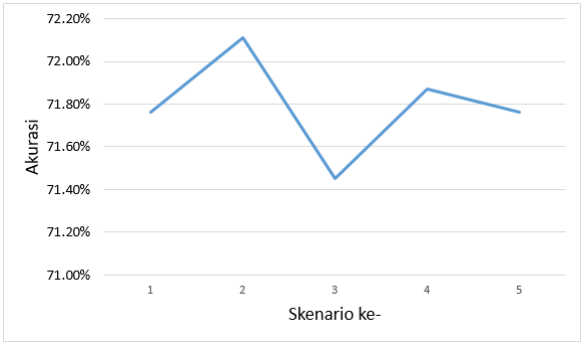
Gambar 3. Hasil Pengujian Pengaruh Nilai k Terhadap Akurasi

Pengujian dengan menggunakan total *data training* dan *testing* sebanyak 1481 data menghasilkan rata-rata akurasi tertinggi apabila k bernilai 8 yaitu dengan nilai 71.7%. Pengujian dengan nilai k = 1 menghasilkan rata-rata nilai akurasi 57.22%, k = 3 dengan rata-rata akurasi bernilai 61.14%. Rata-rata keakurasian ini terus meningkat sehingga k = 8, yaitu dengan rata-rata akurasi 68.92% apabila k = 5, 69.83% apabila k=7 dan 71.7% apabila k = 8. Rata-rata akurasi mengalami penurunan nilai pada saat k = 10. Dari hasil tersebut, dapat disimpulkan bahwa nilai k memberi pengaruh pada nilai akurasi meskipun perubahan tersebut tidak terlalu signifikan. Hasil tersebut membuktikan bahwa nilai k yang sangat besar akan menyebabkan tingkat akurasi menurun. Hal ini dikarenakan semakin besar nilai k, maka semakin banyak data yang diambil untuk proses klasifikasi sehingga semakin banyak data yang tidak relevan.

3.3 Pengujian Pengaruh Jumlah *Training Data* Terhadap Akurasi

Pada pengujian ini, jumlah *data training* dan *data testing* yang akan digunakan adalah sesuai yang telah ditetapkan di Tabel 7. Hasil pengujian sebelumnya menghasilkan nilai k = 8 sebagai nilai yang optimal untuk digunakan pada sistem ini yaitu dengan rata-rata keakurasian sebesar 71.7%. Hasil pengujian pengaruh jumlah data latih terhadap keakurasian ditunjukkan pada Gambar 4.

Pada pengujian pengaruh jumlah *data training* terhadap akurasi, skenario 2 menghasilkan akurasi



Gambar 4. Hasil Pengujian Pengaruh Jumlah Data Latih Terhadap Akurasi

tertinggi ketika porsi *data training* dan *data testing* adalah 55-45% dari total 1481 data yaitu sebesar 72.11%. Pada pengujian ini, skenario 1 menghasilkan akurasi sebesar 71.76%, skenario 3 sebesar 71.45%. Skenario 4 mengalami kenaikan akurasi menjadi 71.87%, sedangkan skenario 5 menghasilkan akurasi 71.33%. Dari hasil pengujian tersebut, dapat disimpulkan bahwa jumlah *data training* tidak selamanya mempengaruhi tingkat akurasi. Meskipun terjadi penurunan, selisih antara nilai akurasi tersebut tidak terlalu signifikan. Hal ini disebabkan oleh sebaran kelas data yang tidak merata, dimana terdapat satu kelas yang datanya lebih mendominasi berbanding dua kelas lainnya.

4. Kesimpulan

Berdasarkan penelitian yang telah dilakukan, dapat disimpulkan bahwa metode k-*nearest neighbor* dapat diimplementasikan untuk klasifikasi kondisi lalu lintas di Kota Jakarta. Nilai k yang optimal digunakan pada klasifikasi kondisi lalu lintas ini adalah 8, dengan rata-rata akurasi dari 5 pengujian bernilai 71.7%. Secara umum, pemilihan k yang terlalu kecil dan terlalu besar dapat menyebabkan penurunan akurasi. Nilai k yang terlalu kecil dan besar akan menyebabkan algoritma sensitif terhadap *noise* sehingga mengakibatkan penurunan dalam tingkat akurasi.

Penelitian selanjutnya bisa menambahkan atribut penyebab kemacetan pada proses klasifikasi, sehingga hasil klasifikasi menjadi lebih akurat dan lebih detail. Selain itu, bisa dilakukan pengambilan data dan perhitungan k-*nearest neigbor* secara *real time*.

Daftar Pustaka

[1] B. P. S. P. D. Jakarta, "Statistik transportasi dki jakarta 2015," *DKI Jakarta: Badan Pusat Statistik Provinsi DKI Jakarta*, 2015.

[2] F. Lécué, R. Tucker, V. Bicer, P. Tommasi, S. Tallevi-Diotallevi, and M. Sbodio, "Predicting

severity of road traffic congestion using semantic web technologies," in *European Semantic Web Conference*. Springer, 2014, pp. 611–627.

[3] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, S. Y. Philip *et al.*, "Top 10 algorithms in data mining," *Knowledge and information systems*, vol. 14, no. 1, pp. 1–37, 2008.

[4] J. Henrique, "Jefferson-henrique/getoldtweets-java," Apr 2016. [Online]. Available: https://github.com/Jefferson-Henrique/GetOldTweets-java

[5] S. Sumathi and S. Sivanandam, *Introduction to data mining and its applications*. Springer, 2006, vol. 29.