Analisis Kinerja Struktur Data Kd-Tree Pada Metode K-Nearest Neighbors

Yuan Lukito
Program Studi Teknik Informatika, Fakultas Teknologi Informasi
Universitas Kristen Duta Wacana
Jl. Dr. Wahidin Sudiro Husodo 5-25 Yogyakarta
Telp. (0274) 563929
yuanlukito@ti.ukdw.ac.id

Abstrak : *Metode K-Nearest Neighbors (KNN) merupakan salah satu metode yang banyak digunakan untuk melakukan klasifikasi. KNN termasuk dalam kelompok metode instance-based, sehingga jumlah data pelatihan yang digunakan akan mempengaruhi lamanya proses perhitungan hasil klasifikasi. Penelitian ini membandingkan dan menganalisis penggunaan struktur data KD-Tree dan Array pada implementasi metode KNN. Penelitian dilakukan dengan menggunakan dataset yang memiliki atribut dalam jumlah banyak. Dari hasil pengujian dapat disimpulkan bahwa struktur data KD-Tree secara umum memiliki kecepatan yang lebih baik dan relatif stabil jika dibandingkan dengan struktur data Array.*

Kata kunci : *KD-Tree, K-Nearest Neighbors, Array*

Abstract : *K-Nearest Neighbors is a commonly used classification technique that can be categorized into instance-based classification method. The performance of KNN is mostly determined by the size of the training data. This research compared and analyzed KD-Tree and Array data structures on KNN implementation. Dataset used in this research has large multidimensional features. From the experiment conducted we can conclude that KD-Tree data structure has better and relatively stable performance compared to Array data structure.*

Keywords : *KD-Tree, K-Nearest Neighbors, Array*

PENDAHULUAN

Metode K-Nearest Neighbors (K-NN) merupakan salah satu metode klasifikasi yang dapat digunakan untuk mengklasifikasikan berbagai jenis data, seperti teks, citra, suara maupun jenis data lainnya. Metode ini termasuk dalam kategori *instance-based learning*, yaitu klasifikasi dilakukan dengan cara membandingkan langsung data uji dengan data pelatihan yang telah dikumpulkan sebelumnya.

Salah satu kelemahan dari algoritma K-NN adalah proses perhitungan kemiripan yang harus dilakukan terhadap seluruh data pelatihan yang ada (Mitchel, 1997). Jika data pelatihan bertambah jumlahnya, waktu untuk melakukan klasifikasi juga akan meningkat secara proporsional. Masalah tersebut dapat diatasi jika menggunakan struktur data KD-Tree.

Penelitian ini bertujuan untuk menganalisis kinerja KD-Tree berdasarkan waktu yang dibutuhkan untuk menemukan *k* data pelatihan yang paling mirip dengan data uji. Sebagai perbandingan digunakan struktur data Array sebagai acuan kinerja standar.

K-Nearest Neighbors

Setiap data pelatihan (*instance*) diasumsikan memiliki serangkaian ciri (features) yang dapat direpresentasikan dalam bentuk vektor [1].

$$V = [c_1, c_2, c_3, c_4, \dots, c_n] \quad \text{.....[1]}$$

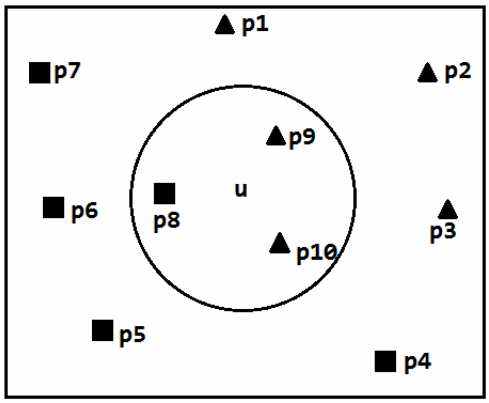
dengan: **V** adalah vektor ciri dari suatu data pelatihan, **n** merupakan jumlah ciri dan $\mathbf{c_1}$, $\mathbf{c_2}$, $\mathbf{c_3}$, $\mathbf{c_4}$, **…** $\mathbf{c_n}$ merupakan nilai dari masing-masing ciri.

Perhitungan kemiripan dilakukan dengan cara menghitung tingkat kemiripan (jarak) antara data uji dengan data pelatihan. Perhitungan tingkat kemiripan umumnya dilakukan dengan menggunakan Euclidian Distance, seperti pada persamaan [2].

$$D_{(u,p)} = \sqrt{\sum_{i=1}^{n}(cu_i - cp_i)^2} \text{ ……..[2]}$$

dengan: $D_{(u,p)}$ adalah tingkat kemiripan antara data uji (u) dengan data pelatihan (p), cu_i adalah nilai ciri yang ke-i dari data uji dan cp_i adalah nilai ciri ke-i dari data pelatihan.

Perhitungan tingkat kemiripan dilakukan terhadap data uji dengan seluruh data pelatihan yang ada. Hasil klasifikasi ditentukan oleh kelas-kelas dari sejumlah k data pelatihan yang paling mirip dengan data uji (Mitchel, 1997). Nilai dari k harus ditentukan terlebih dahulu. Pada umumnya k bernilai ganjil (1, 3, 5, 7, …). Contoh ilustrasi penentuan hasil klasifikasi dengan nilai k = 3 dapat dilihat pada Gambar 1.



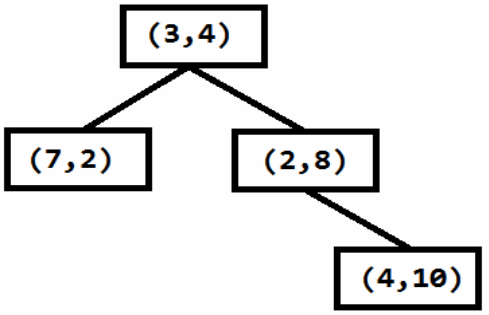
Gambar 1. Ilustrasi penentuan hasil klasifikasi untuk k = 3.

Sebuah data uji (u) akan diklasifikasikan apakah termasuk dalam kelas segitiga atau kotak. Terdapat 10 data pelatihan (p_1, p_2, …, p_{10}) yang masing-masing sudah memiliki informasi kelas (segitiga atau kotak). Dari hasil perhitungan kemiripan, terdapat tiga data pelatihan yang memiliki kemiripan paling tinggi (jaraknya lebih dekat), yaitu p_8, p_9 dan p_{10}. Dari tiga data pelatihan yang paling mirip, terdapat dua kelas segitiga (p_9 dan p_{10}) dan satu kelas kotak (p_8) sehingga hasil klasifikasi dari u adalah termasuk dalam kelas segitiga.

Pada Gambar 1 dapat dilihat perhitungan kemiripan dilakukan terhadap seluruh data pelatihan, kemudian diambil 3 yang paling mirip. Jika jumlah data pelatihan meningkat, maka waktu yang dibutuhkan untuk klasifikasi akan semakin meningkat secara proporsional (O(N)). Faktor lain yang menentukan adalah jumlah ciri yang ada pada setiap data pelatihan. Semakin banyak ciri maka perhitungan tingkat kemiripan juga akan semakin lama.

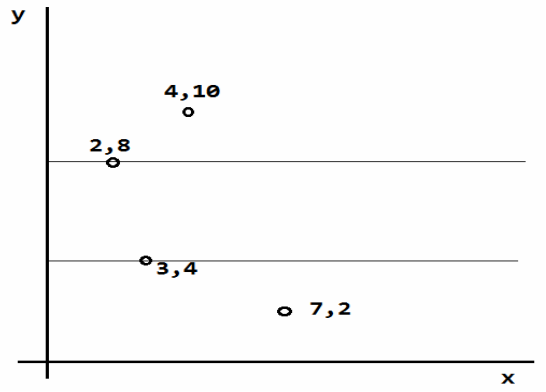
KD-Tree

KD-Tree (K-Dimensional Tree) merupakan representasi data multidimensional dalam bentuk binary tree yang bertujuan untuk memisahkan setiap data dalam suatu area tertentu berdasarkan nilai posisinya (Skiena, 2008). Contoh KD-Tree yang disusun dari data (3,4), (7,2), (2,8) dan (4,10) dapat dilihat pada Gambar 2.



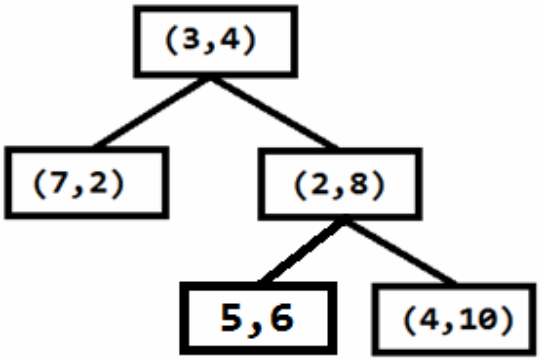
Gambar 2. Representasi KD-Tree dari beberapa data dua dimensi (x, y).

Perbedaan paling mendasar antara KD-Tree dengan Binary Tree adalah pada jumlah dimensi data yang berukuran lebih dari satu (K-Dimensional). Dari KD-Tree yang terbentuk pada Gambar 2, didapatkan area-area yang membagi masing-masing data seperti pada Gambar 3.



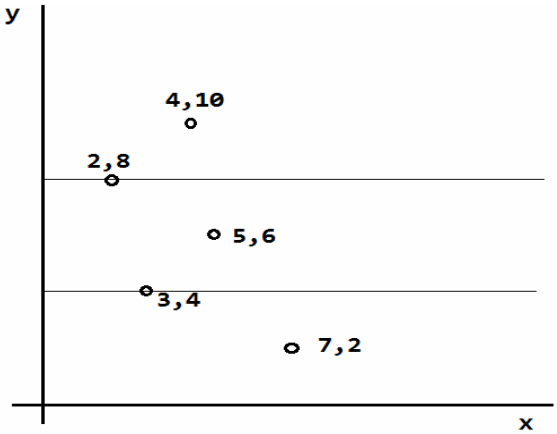
Gambar 3. Pembagian area (x,y) berdasarkan KD-Tree yang dihasilkan.

Titik (3,4) merupakan root dari KD-Tree, membagi area menjadi dua bagian (dalam sumbu y). Kemudian titik (2,8) membagi area menjadi dua bagian (dalam sumbu y). Dengan demikian setiap titik sudah berada dalam area masing-masing. Jika ada titik baru, misalnya (5,6), maka KD-Tree yang dihasilkan dapat dilihat pada Gambar 4.



Gambar 4. KD-Tree yang dihasilkan setelah penambahan titik (5,6).

Dengan adanya titik baru (5,6), area yang dihasilkan akan berubah, menjadi seperti pada Gambar 5. Titik (2,8) membagi menjadi dua area (dalam sumbu y), masing-masing berisi titik (5,6) dan 4,10).



Gambar 5. Pembagian area setelah penambahan titik (5,6).

Nearest Neighbors dengan KD-Tree

Setelah KD-Tree terbentuk maka pencarian Nearest Neighbors dapat dilakukan dengan cara memasukkan titik uji ke dalam KD-Tree, kemudian menghitung kemiripan dengan node-node terdekat (sibling node maupun parent node). Dengan demikian secara umum hanya diperlukan sebanyak k operasi perhitungan kemiripan saja, di mana k sudah ditentukan sebelumnya (K-NN).

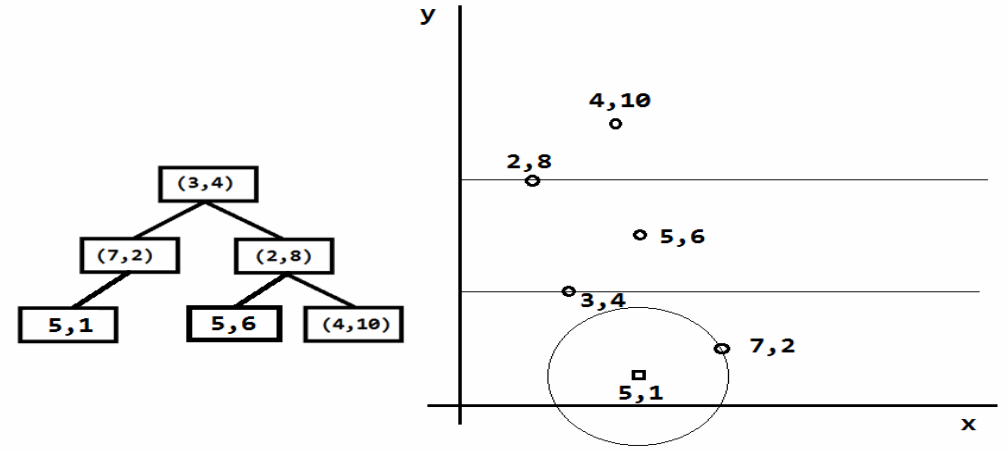
Misalkan terdapat data uji titik (5,1), untuk menentukan satu titik terdekat dapat dilakukan dengan cara menghitung jarak titik (5,1) terhadap seluruh titik yang ada. Setelah didapatkan semua jarak, diambil salah satu yang jaraknya paling kecil. Hasil perhitungan jarak keseluruhan dapat dilihat pada Tabel 1.

Tabel 1. Hasil perhitungan Euclidian Distance titik (5,1) terhadap titik-titik lainnya.

Titik	Jarak
(2,8)	7,616
(3,4)	3,606
(4,10)	9,055
(5,6)	5
(7,2)	2,236

Dari Tabel 1 dapat disimpulkan bahwa titik (7,2) memiliki jarak terdekat dengan titik (5,1). Hasil ini didapatkan setelah menghitung jarak titik (5,1) terhadap seluruh titik-titik yang ada. Jika menggunakan KD-Tree, proses perhitungan diawali dengan memasukkan titik (5,1) ke dalam KD-Tree yang sudah terbentuk sebelumnya. Hasilnya dapat dilihat pada Gambar 6. Titik (5,1) akan diletakkan di sebelah kiri *root node*, yaitu (3,4). Kemudian jika dibandingkan dengan titik (7,2) maka titik (5,1) akan diletakkan di sebelah kirinya. Untuk mencari 1 data termirip (k = 1) cukup dilakukan satu perhitungan saja, yaitu menghitung pada *sibling node* (tidak ada) atau pada *parent node* dari titik (5,1), yaitu titik (7,2).

Kedua metode perhitungan tersebut menghasilkan hasil yang sama, tetapi dengan menggunakan KD-Tree akan lebih cepat karena tidak perlu menghitung kemiripan dengan keseluruhan data yang ada.



Gambar 6. Data uji (5,1) untuk k = 1.

METODE PENELITIAN

Penelitian ini melakukan pengujian kinerja KD-Tree dengan cara mengukur waktu yang diperlukan untuk mendapatkan k data pelatihan yang paling mirip dengan data uji yang dimasukkan. Data yang digunakan dalam pengujian berupa data ruangan dan kekuatan sinyal *access point* di lingkungan Universitas Kristen Duta Wacana (Lukito & Chrismanto, 2015). Pengujian dilakukan dalam beberapa tahapan, yaitu menggunakan data pelatihan sebanyak 250, 500, 1000, 2000, 4000 dan 8000 data untuk melihat laju perkembangan waktu yang diperlukan. Jumlah ciri yang digunakan sebanyak 177 ciri, seperti terlihat pada Tabel 2.

Tabel 2. Cuplikan data pelatihan yang digunakan.

Ruangan	AP1 (dbm)	AP2 (dbm)	AP3 (dbm)	…	AP177(dbm)
Perpustakaan	-60	-70	-82		-100
LPPM	-100	-100	-100		-89
Koperasi	-70	-40	-40		-63
Biro Akademik	-90	-100	-100		-100
Fakultas Teknologi Informasi	-50	-100	-63		-77
Toko Buku UKDW	-51	-100	-80		-70
Humas UKDW	-60	-80	-100		-66

Pengukuran waktu dilakukan terhadap dua macam struktur data, yaitu Array dan KD-Tree. Penggunaan struktur data Array dilakukan untuk mendapatkan acuan kinerja standar. Implementasi KD-Tree menggunakan library KD-Tree dari Java Machine Learning (Java-ML, 2008). Dalam penelitian ini waktu yang diukur ada dua macam, yaitu waktu pembentukan data training (baik dalam bentuk Array atau KD-Tree) dan waktu yang diperlukan untuk menghitung hasil klasifikasi dengan K-NN.

Setiap percobaan dilakukan sebanyak tiga kali, kemudian hasilnya dihitung dari rata-rata tiga kali percobaan tersebut. Untuk struktur data Array hanya dilakukan pengujian dengan nilai k=1 sebagai acuan kinerja standar. Pada struktur data KD-Tree dilakukan beberapa kali pengujian dengan nilai k yang berbeda-beda, yaitu k=1, k=3 dan k=5.

HASIL DAN PEMBAHASAN

Hasil pengujian kinerja dengan menggunakan Array dapat dilihat pada Tabel 3, sedangkan hasil pengujian dengan menggunakan KD-Tree dapat dilihat pada Tabel 4. Waktu inisialisasi adalah waktu keseluruhan yang dibutuhkan mulai dari membaca data pelatihan dari suatu file *.csv sampai terbentuknya struktur data baik dalam bentuk Array maupun KD-Tree yang akan dibutuhkan pada tahap klasifikasi dengan K-Nearest Neighbors. Waktu klasifikasi adalah waktu yang diperlukan untuk menentukan hasil klasifikasi dari suatu data uji.

Tabel 3. Hasil pengujian kinerja dengan menggunakan Array untuk nilai k = 1.

Jumlah data pelatihan	Inisialisasi (ms)			Rata-rata Inisialisasi (ms)	Klasifikasi (ms)			Rata-rata klasifikasi (ms)
500	171	172	172	171,67	31	31	31	31,00
1000	188	219	203	203,33	31	31	31	31,00
2000	250	219	219	229,33	63	61	62	62,00
4000	328	297	312	312,33	110	109	110	109,67
8000	438	484	437	453,00	203	203	219	208,33

Tabel 4. Hasil pengujian kinerja dengan menggunakan KD-Tree untuk nilai k = 1.

Jumlah data pelatihan	Inisialisasi (ms)			Rata-rata Inisialisasi (ms)	Klasifikasi (ms)			Rata-rata klasifikasi (ms)
500	172	188	172	177,33	15	16	15	15,33
1000	219	234	203	218,67	15	16	15	15,33
2000	234	250	235	239,67	16	15	15	15,33
4000	312	315	328	318,33	15	16	16	15,67
8000	453	438	453	448,00	16	16	16	16,00

Dari hasil pengujian didapatkan waktu inisialisasi baik untuk struktur data Array maupun KD-Tree meningkat sesuai dengan jumlah data pelatihan yang diproses. Dari Tabel 3 dapat dilihat bahwa waktu klasifikasi semakin meningkat seiring dengan bertambahnya jumlah data pelatihan yang harus dihitung kemiripannya. Berbeda dengan hasil pada Tabel 4, secara umum waktu klasifikasi dengan struktur data KD-Tree relatif tidak berubah walaupun jumlah data pelatihan bertambah. Hasil pengujian struktur data KD-Tree untuk nilai k=3 dan k=5 secara berturut-turut dapat dilihat pada Tabel 5 dan Tabel 6.

Tabel 5. Hasil pengujian kinerja dengan menggunakan KD-Tree untuk nilai k = 3.

Jumlah data pelatihan	Inisialisasi (ms)			Rata-rata Inisialisasi (ms)	Klasifikasi (ms)			Rata-rata klasifikasi (ms)
500	171	172	172	171,67	16	15	16	15,67
1000	219	218	219	218,67	16	15	15	15,33
2000	234	250	250	244,67	16	15	16	15,67
4000	328	313	313	318,00	15	15	16	15,33
8000	438	453	453	448,00	16	16	16	16,00

Tabel 6. Hasil pengujian kinerja dengan menggunakan KD-Tree untuk nilai k = 5.

Jumlah data pelatihan	Inisialisasi (ms)			Rata-rata Inisialisasi (ms)	Klasifikasi (ms)			Rata-rata klasifikasi (ms)
500	171	172	187	176,67	15	16	16	15,67
1000	217	219	220	218,67	15	16	16	15,67
2000	235	245	255	245,00	16	15	16	15,67
4000	313	330	316	319,67	15	16	16	15,67
8000	453	438	438	443,00	16	16	16	16,00

Dari Tabel 5 dan Tabel 6 dapat dilihat waktu klasifikasi pada struktur data KD-Tree relatif tidak berubah. Waktu untuk inisialisasi secara umum ikut meningkat jika jumlah data pelatihan bertambah, baik pada struktur data Array maupun pada struktur data KD-Tree.

SIMPULAN DAN SARAN

Dari hasil pengujian dapat disimpulkan bahwa penggunaan struktur data KD-Tree dapat mengurangi waktu yang diperlukan saat melakukan klasifikasi dengan menggunakan metode K-Nearest Neighbors. Waktu yang diperlukan untuk klasifikasi juga relatif konstan walaupun nilai k berubah-ubah. Implementasi KD-Tree juga sangat menentukan, dari hasil pengujian didapatkan waktu untuk inisialisasi dan pembentukan KD-Tree relatif sama dengan waktu yang diperlukan untuk inisialisasi dan pembentukan Array.

Penelitian ini dapat dilanjutkan atau dikembangkan dengan menambahkan struktur data yang dibandingkan, misalnya Locally-Sensitive Hashing (LSH) atau Voronoi Tessellation.

DAFTAR PUSTAKA

Java Machine Learning. (2008). Class KD-Tree. Diakses pada 15 Maret 2016 dari World Wide Web: http://java-ml.sourceforge.net/ap i/0.1.7/net/sf/javaml/co re/kdtree/KDTree .html

Lukito, Y., Chrismanto, A., (2015). Perbandingan Metode-Metode Klasifikasi Untuk Indoor Positioning System. Jurnal Teknik Informatika dan Sistem Informasi. 1 (**2**): 123-131.

Mitchel, T.M. (1997) *Machine Learning*. Portland: McGraw-Hill.

Skiena, S.S. (2008) *The Algorithm Design Manual (2nd Edition)*. London: Springer-Verlag.

BIODATA PENULIS

Penulis adalah lulusan S1 dan S2 Ilmu Komputer, Universitas Gadjah Mada, Yogyakarta. Saat ini penulis adalah dosen di program studi Teknik Informatika, Universitas Kristen Duta Wacana, Yogyakarta. Bidang penelitian yang diminati adalah pengenalan pola, *machine learning*, pengolahan citra digital dan *computer vision*. Penulis dapat dihubungi melalui email yuanlukito@ti.ukdw.ac.id atau melalui website http://lecturer.ukdw.ac.id/yuan