STUDI KOMPARATIF PENERAPAN METODE *HIERARCHICAL*, K-*MEANS* DAN *SELF ORGANIZING MAPS* (SOM) C*LUSTERING* PADA BASIS DATA

Undang Syaripudin[1], Ijang Badruzaman[2], Erwan Yani[3], Dede K[4], M. Ramdhani[5]

1, 2 Teknik Informatika UIN Sunan Gunung Djati Bandung

3, 4, 5 AMIK Garut

Abstract

This study identifies the results of some test results clustering methods. The data set used in this test method Clustering. The third method of clustering based on these factors than the size of the data set and the extent of the cluster. The test results showed that the SOM algorithm produces better accuracy in classifying objects into matching groups. K-means algorithm is very good when using large data sets and compared with Hierarchical SOM algorithm. Hierarchical grouping and SOM showed good results when using small data sets compared to using k-means algorithm.

Keyword: Testing, Clustering. K-means, hierarchical, hierarchical, SOM

1. Pendahuluan

Clustering merupakan teknik pengelompokan sejumlah data atau objek ke dalam *cluster* (*group*) sehingga setiap dalam *cluster* tersebut akan berisi data yang semirip mungkin dan berbeda dengan objek dalam *cluster* yang lainnya (Santosa B., 2007). Terdapat beberapa metode *clustering* diantaranya *hierarchical, K-means, self organizing maps (SOM) clustering* (Alfina, 2012). K-*means* merupakan metode *clustering* yang paling sederhana dan umum. K-*means* mempunyai kemampuan mengelompokkan data dalam jumlah yang cukup besar dengan waktu komputasi yang relatif cepat dan efisien. Metode hierarki dapat dibedakan menjadi dua bagian, yaitu metode penggabungan (*agglomerative*) dan metode pemecahan (*devisive*). Pembentukan kelompok dalam metode hierarki, menggunakan beberapa cara, antara lain pautan tunggal (*single linkage*), pautan lengkap (*complete linkage*), dan pautan rata-rata (*average linkage*). *Self Organizing Maps* (SOM) merupakan suatu tipe *Artificial Neural Networks* yang di-*training* secara *unsupervised*. SOM menghasilkan map yang terdiri dari *output* dalam dimensi yang rendah (2 atau 3

dimensi). Map ini berusaha mencari *property* dari *input* data.

2. Data *Mining*

Data *mining* merupakan gabungan dari berbagai bidang ilmu, antara lain basis data, *information retrieval*, statistika, algoritma dan *machine learning*. Bidang ini telah berkembang sejak lama namun makin terasa pentingnya sekarang ini di mana muncul keperluan untuk mendapatkan informasi yang lebih dari data transaksi maupun fakta yang terkumpul selama bertahun-tahun. *Data mining* adalah cara menemukan informasi tersembunyi dalam sebuah basis data dan merupakan bagian dari proses *Knowledge Discovery in Databases* (KDD) untuk menemukan informasi dan pola yang berguna dalam data (Budiarti, 2006). Kegiatan data *mining* biasanya dilakukan pada sebuah data *warehouse* yang menampung data dalam jumlah besar dari suatu organisasi. Proses data *mining* mencari informasi baru, berharga dan berguna di dalam sekumpulan data bervolume besar dengan melibatkan komputer dan manusia serta bersifat iteratif baik melalui proses otomatis ataupun manual. Secara umum, data *mining* terbagi dalam 2 sifat:

a. *Predictive*: menghasilkan model berdasarkan sekumpulan data yang dapat digunakan untuk memperkirakan nilai data yang lain. Metode-metode yang termasuk *Predictive Data Mining* adalah:
   1. Klasifikasi: pembagian data ke dalam beberapa kelompok yang telah ditentukan sebelumnya.
   2. Regresi: memetakan data ke suatu *prediction variable*.
   3. *Time series Analysis*: pengamatan perubahan nilai atribut dari waktu ke waktu.

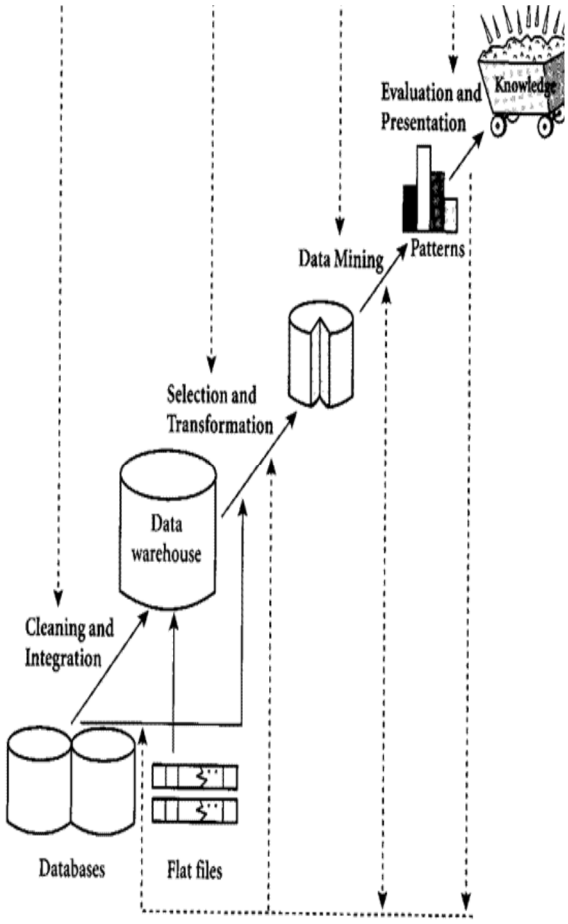
b. *Descriptive*: mengidentifikasi pola atau hubungan dalam data untuk menghasilkan informasi baru. Metode yang termasuk dalam *Descriptive* Data *Mining* adalah:
   1. *Clustering*: identifikasi kategori untuk mendeskripsikan data.
   2. *Association Rules*: identifikasi hubungan antara data yang satu dengan lainnya.
   3. *Summarization*: pemetaan data ke dalam subset dengan deskripsi sederhana.
   4. *Sequence Discovery*: identifikasi pola sekuensial dalam data.

2.1 Tahapan Data *Mining*

Dalam aplikasinya, data *mining* sebenarnya merupakan bagian dari proses *Knowledge Discovery in Database* (KDD), bukan sebagai teknologi yang utuh dan

ISSN 1979-8911

berdiri sendiri. Data *mining* merupakan suatu bagian langkah yang penting dalam proses KDD terutama berkaitan dengan ekstraksi dan penghitungan pola-pola dari data yang ditelaah, seperti ditunjukan oleh gambar 1 di bawah ini :



Gambar 1 Tahapan pada proses *knowledge discovery*

a. Data *cleaning*
   Untuk menghilangkan data *noise* (data yang tidak relevan/berhubungan langsung dengan tujuan akhir proses data *mining*, misal: data *mining* yang bertujuan untuk menganalisa hasil penjualan, maka data-data dalam kumpulan seperti ”nama pegawai”, ”umur”, dan sebagainya dapat di-*ignore*) dan tidak konsisten.

b. Data *integration*
   Untuk menggabungkan *multiple* data *source*.

c. Data *selection*
   Untuk mengambil data yang sesuai untuk keperluan analisa.

d. Data *transformation*
   Untuk mentransformasikan data ke dalam bentuk yang lebih sesuai untuk data *mining*.

e. Data *Mining*
   Proses terpenting dimana metode tertentu diterapkan untuk menghasilkan data *pattern*.

f. *Pattern evaluation*
   Untuk mengidentifikasi apakah *interenting patterns* yang didapatkan sudah cukup mewakili *knowledge* berdasarkan perhitungan tertentu.

g. *Knowledge presentation*
   Untuk mempresentasikan *knowledge* yang sudah didapatkan dari *user*.

2.2 Studi Komparatif

Studi komparatif terdiri dari dua suku kata yaitu "studi" dan "komparatif". Dalam kamus bahasa Indonesia "studi" berarti penelitian, kajian atau telaah (Depdiknas, 2007). Sedangkan "komparatif" yaitu berkenaan atau berdasarkan perbandingan (Depdiknas, 2007). Jadi jika pengertian di atas disatukan maka pengertian studi komparatif adalah penelitian ilmiah atau kajian berdasarkan dengan perbandingan. Pendapat Aswarni yang dikutip Suharsimi Arikunto (1997 : 236) menyebutkan bahwa "Penelitian komparatif akan menemukan persamaan-persamaan dan perbedaan-perbedaan tentang benda, orang, prosedur kerja, ide, kritik terhadap orang, kelompok, terhadap suatu idea atau suatu prosedur kerja".

Pendapat lain, Mohammad Nasir (1988 : 68) mengatakan bahwa "Studi atau penelitian komparatif adalah sejenis penelitian deskriptif yang ingin mencari jawaban secara mendasar tentang sebab akibat, dengan menganalisa faktor-faktor penyebab terjadinya atau munculnya suatu fenomena tertentu".

Jadi studi komparatif adalah penelitian yang bertujuan untuk membandingkan dua variabel atau lebih, untuk mendapatkan jawaban atau fakta apakah ada perbandingan atau tidak dari objek yang sedang diteliti.

3. *Clustering*

Clustering merupakan proses membuat pengelompokan sehingga semua anggota dari setiap partisi mempunyai persamaan berdasarkan matrik tertentu (Santosa, 2007). *Clustering* juga dikenal sebagai *unsupervised learning* yang membagi data menjadi kelompok-kelompok atau *clusters* berdasarkan suatu kemiripan atribut-atribut di antara data tersebut. Karakteristik tiap *cluster* tidak ditentukan sebelumnya, melainkan tercermin dari kemiripan data yang terkelompok di dalamnya.

3.1 Analisis *Clustering*

Analisis *Clustering* adalah proses pengelompokan obyek ke dalam *subsets* yang mempunyai arti dalam konteks masalah tertentu (Tias, 2009). Obyek dengan demikian diorganisir ke dalam suatu penyajian efisien dan bermanfaat. Tidak sama dengan klasifikasi, *clustering* tidak bersandar pada kelas sudah ada. *Clustering* dikenal sebagai suatu metode pelajaran pembelajaran *unsupervised* karena tidak ada informasi disajikan tentang "jawaban yang benar" untuk obyek yang manapun. Ini dapat menemukan hubungan yang sebelumnya tidak diketahui didalam suatu *dataset* yang kompleks.

Analisis *cluster* merupakan suatu teknik analisa *multivariate* untuk mencari

ISSN 1979-8911

dan mengorganisir informasi tentang variabel sehingga secara relatif dapat dikelompokkan dalam kelompok yang homogen atau "*cluster*" dapat dibentuk. *Cluster* dibentuk dengan metode kedekatan yang secara internal harus homogen (anggota adalah serupa untuk satu sama lain) dan sangat secara eksternal tak sejenis (anggotanya tidak seperti anggota dari *cluster* yang lain).

Analisis *cluster* dapat menerima suatu data masukan yang beragam. Ini biasanya disebut pengukuran "kesamaan", dapat juga disebut"kedeka tannya", dan "kemiripannya". Beberapa ahli merekomendasikan penggunaan standardisasi data, *cluster* dapat dihitung dalam skala yang berbeda dan standardisasi akan memberi pengukuran dengan menggunakan unit yang berbeda.

Seperti teknik yang lain, analisis *cluster* menghadapi permasalahan dalam beberapa banyak faktor, atau dimensi, atau berapa banyak *cluster* yang akan dihasilkan. Untuk itu akan dipilih suatu tempat dimana struktur *cluster* yang stabil untuk jarak yang jauh. Beberapa kemungkinan lain akan mencari pengelompokan grup dengan struktur cocok atau yang diharapkan.

3.2 Fungsi Jarak

Pengukuran *proximity* yang paling umum digunakan, sedikitnya untuk rasio skala adalah matrik *Minkowski*, yang mana adalah suatu generalisasi jarak antara titik di dalam *Euclidean Space*. Jarak *Euclidean* dapat dianggap sebagai jarak yang paling pendek antar dua poin-poin, dan pada dasarnya sama halnya dengan persamaan *Pythagoras* ketika digunakan di dalam 2 dimensi. Secara matematis dapat dituliskan di dalam persamaan berikut :

$$d(i,j)=\sqrt{\left|x_{i1}-x_{j1}\right|^2+\left|x_{i2}-x_{j2}\right|^2+....+\left|x_{ip}-x_{jp}\right|^2}$$

Gambar 2. Persamaan *Pythagoras*

Ketika menggunakan fungsi jarak *Euclidean* untuk membandingkan jarak, tidak diperlukan untuk mengkalkulasi akar dua sebab jarak selalu merupakan angka-angka positif. Untuk dua jarak, d_1 dan d_2 , jika :

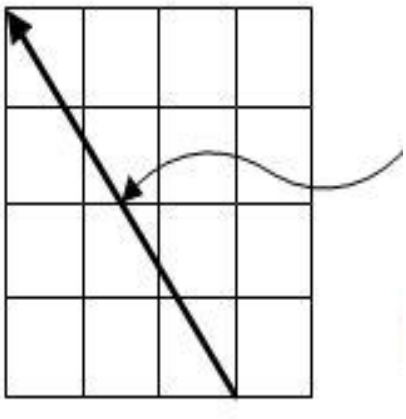
$$\sqrt{d_1} > \sqrt{d_2} \Leftrightarrow d_1 > d_2.$$

Gambar 3. Fungsi Jarak *Euclidean*

Jika sebagian dari suatu atribut obyek diukur dengan skala berbeda, maka ketika menggunakan fungsi jarak

ISSN 1979-8911

Euclidean, atribut dengan skala yang lebih besar boleh meliputi atribut yang terukur pada skala yang lebih kecil. Untuk mencegah masalah ini, nilai-nilai atribut dinormalisasi untuk terletak diantara 0 dan 1. Fungsi jarak lain mungkin lebih sesuai untuk beberapa data. Lebih jelasnya dapat dilihat pada gambar 4 representasi dari jarak terdekat dari 2 titik.



Gambar 4. Fungsi *Euclidean*

Suatu komponen penting pada algoritma *cluster* adalah mengukur jarak antara poin-poin data. Jika komponen dari data adalah semua termasuk dalam unit yang sama, mungkin jarak *Euclidean* yang sederhana cukup sukses mengelompokkan data serupa.

Bagaimanapun, bahkan dalam hal ini jarak *Euclidean* kadang-kadang dapat salah. Di samping kedua-duanya pengukuran diambil di dalam unit yang sama, suatu keputusan harus dibuat berkaitan dengan skala. Skala yang berbeda dapat menyebabkan perbedaan *clustering*.

4. K-*Means*

Tujuan dari data *clustering* ini adalah untuk meminimalisasikan *objective function* yang diset dalam proses *clustering*, yang pada umumnya berusaha meminimalisasikan variasi di dalam suatu *cluster* dan memaksimalisasikan variasi antar *cluster*. Data *clustering* menggunakan metode K-*Means* ini secara umum dilakukan dengan algoritma dasar sebagai berikut:

1. Tentukan jumlah *cluster*
2. Alokasikan data ke dalam *cluster* secara random
3. Hitung *centroid*/rata-rata dari data yang ada di masing-masing *cluster*
4. Alokasikan masing-masing data ke *centroid*/rata-rata terdekat
5. Kembali ke Step 3, apabila masih ada data yang berpindah *cluster* atau apabila perubahan nilai *centroid*, ada yang di atas nilai *threshold* yang ditentukan atau apabila perubahan nilai pada *objective function* yang digunakan di atas nilai *threshold* yang ditentukan.

4.1 Perkembangan Penerapan *K-Means*

Beberapa alternatif penerapan K-*Means* dengan beberapa pengembangan teori-teori penghitungan terkait telah diusulkan. Hal ini termasuk pemilihan:

1. *Distance space* untuk menghitung jarak di antara suatu data dan *centroid*
2. Metode pengalokasian data kembali ke dalam setiap *cluster*
3. *Objective function* yang digunakan

4.1.1 *Distance Space* Untuk Menghitung Jarak Antara Data dan *Centroid*

Beberapa *distance space* telah diimplementasikan dalam menghitung jarak (*distance*) antara data dan *centroid* termasuk di antaranya L_1 (*Manhattan/City Block*) *distance space*, L_2 (*Euclidean*) *distance space*, dan L_p (*Minkowski*) *distance space*. Jarak antara dua titik x_1 dan x_2 pada *Manhattan/City Block distance space* dihitung dengan menggunakan rumus sebagai berikut:

$$D_{L_1}(x_2, x_1) = \|x_2 - x_1\|_1 = \sum_{j=1}^{p} |x_{2j} - x_{1j}|$$

dimana:

p : Dimensi data

| . | : Nilai *absolute*

Sedangkan untuk L_2 (*Euclidean*) *distance space*, jarak antara dua titik dihitung menggunakan rumus sebagai berikut:

$$D_{L_2}(x_2, x_1) = \|x_2 - x_1\|_2 = \sqrt{\sum_{j=1}^{p} (x_{2j} - x_{1j})^2}$$

dimana:

p : Dimensi data

L_p (*Minkowski*) *distance space* yang merupakan generalisasi dari beberapa *distance space* yang ada seperti L_1 (*Manhattan/City Block*) dan L_2 (*Euclidean*), juga telah diimplementasikan. Tetapi secara umum *distance space* yang sering digunakan adalah *Manhattan* dan *Euclidean*. *Euclidean* sering digunakan karena penghitungan jarak dalam *distance space* ini merupakan jarak terpendek yang bisa didapatkan antara dua titik yang diperhitungkan, sedangkan *Manhattan* sering digunakan karena kemampuannya dalam mendeteksi keadaan khusus seperti keberadaaan *outliers* dengan lebih baik.

4.1.2 Metode Pengalokasian Ulang Data ke Dalam Masing-Masing *Cluster*

Secara mendasar, ada dua cara pengalokasian data kembali ke dalam masing-masing *cluster* pada saat proses iterasi *clustering*. Kedua cara tersebut adalah pengalokasian dengan cara tegas (*hard*), dimana data item secara tegas

dinyatakan sebagai anggota *cluster* yang satu dan tidak menjadi anggota *cluster* lainnya, dan dengan cara *fuzzy*, dimana masing-masing data item diberikan nilai kemungkinan untuk bisa bergabung ke setiap *cluster* yang ada. Kedua cara pengalokasian tersebut diakomodasikan pada dua metode *Hard K-Means* dan *Fuzzy K-Means*. Perbedaan diantara kedua metode ini terletak pada asumsi yang dipakai sebagai dasar pengalokasian.

Pengalokasian kembali data ke dalam masing-masing *cluster* dalam metode *Hard K-Means* didasarkan pada perbandingan jarak antara data dengan *centroid* setiap *cluster* yang ada. Data dialokasikan ulang secara tegas ke *cluster* yang mempunyai *centroid* terdekat dengan data tersebut. Pengalokasian ini dapat dirumuskan sebagai berikut:

$$a_{ik} = \begin{cases} 1 & d = \min\{D(x_k, v_i)\} \\ 0 & lainnya \end{cases}$$

dimana:

a_{ik} : Keanggotaan data ke-*k* ke *cluster* ke-$_i$

v_i : Nilai *centroid cluster* ke-*i*

4.1.3 *Objective Function* Yang Digunakan

Objective function yang digunakan khususnya untuk *Hard K-Means* dan *Fuzzy K-Means* ditentukan berdasarkan pada pendekatan yang digunakan dalam poin sebelumnya. Untuk metode *Hard K-Means*, *objective function* yang digunakan adalah sebagai berikut:

$$J(U,V) = \sum_{k=1}^{N} \sum_{i=1}^{c} a_{ik} D(x_k, v_i)^2$$

dimana:

N : Jumlah data

c : Jumlah *cluster*

a_{ik} : Keanggotaan data ke-*k* ke *cluster* ke-*i*

v_i : Nilai *centroid cluster* ke-*i*

5. *Hierarchical*

Clustering dengan pendekatan hierarki mengelompokkan data yang mirip dalam hierarki yang sama dan yang tidak mirip dihierarki yang agak jauh. Ada dua metode yang sering diterapkan yaitu *agglomerative hiearchical clustering* dan *divisive hierarchical clustering* (Hartini, 2005). *Agglomerative* melakukan proses *clustering* dari N *cluster* menjadi satu kesatuan *cluster*, dimana N adalah jumlah data, sedangkan *divisive* melakukan proses *clustering* yang sebaliknya yaitu dari satu *cluster* menjadi N *cluster*.

Beberapa metode *hierarchical clustering* yang sering digunakan dibedakan menurut cara mereka untuk menghitung tingkat kemiripan. Ada yang menggunakan *Single Linkage*, *Complete Linkage*, *Average*

Linkage, *Average Group Linkage* dan lain-lainnya. Seperti juga halnya dengan *partition-based clustering*, kita juga bisa memilih jenis jarak yang digunakan untuk menghitung tingkat kemiripan antar data.

Salah satu cara untuk mempermudah pengembangan *dendogram* untuk *hierarchical clustering* ini adalah dengan membuat *similarity matrix* yang memuat tingkat kemiripan antar data yang dikelompokkan. Tingkat kemiripan bisa dihitung dengan berbagai macam cara seperti dengan *Euclidean Distance Space*. Berangkat dari *similarity matrix* ini, kita bisa memilih *lingkage* jenis mana yang akan digunakan untuk mengelompokkan data yang dianalisa.

Tugas *hierarchical clustering* adalah mengatur sekumpulan objek menjadi sebuah hierarki hingga terbentuk kelompok yang memiliki kesamaan. Berikut merupakan langkah-langkah yang untuk melakakukan *hierarchical clustering*:

1. Kelompokkan setiap objek dalam sebuah *cluster*.
2. Temukan pasangan yang paling mirip untuk dimasukkan ke dalam *cluster* yang sama dengan melihat data dalam matriks kemiripan.
3. Kedua objek kemudian digabungkan dalam satu *cluster*.
4. Ulangi dari langkah kedua dan ketiga hingga tersisa sebuah *cluster*.

Untuk mengukur kemiripan dari objek-objek ini dapat dengan menggunakan *cosinus*, *kovarian*, dan korelasi (Santosa, 2006).

a. Single Linkage

Input untuk algoritma *single linkage* bisa berujud jarak atau *similarities* antara pasangan-pasangan dari objek-objek. Kelompok-kelompok dibentuk dari *entities* individu dengan menggabungkan jarak paling pendek atau *similarities* (kemiripan) yang paling besar.

Pada awalnya, kita harus menemukan jarak terpendek dalam $D = \{d_{ik}\}$ dan menggabungkan objek-objek yang bersesuaian misalnya, U dan V , untuk mendapatkan *cluster* (UV). Untuk langkah (3) dari algoritma di atas jarak-jarak antara (UV) dan *cluster* W yang lain dihitung dengan cara

$$d_{(UV)W} = min\ \{d_{UW}, d_{VW}\}$$

Di sini besaran-besaran d_{UW} dan d_{VW} berturut-turut adalah jarak terpendek antara *cluster-cluster* U dan W dan juga *cluster-cluster* V dan W .

b. Complete Linkage

Complete linkage memberikan kepastian bahwa semua item-item dalam

satu *cluster* berada dalam jarak paling jauh (similaritas terkecil) satu sama lain. Algoritma *aglomerative* pada umumnya dimulai dengan menentukan *entri* (elemen matriks) dalam $D = \{d_{ik}\}$ dan menggabungkan objek-objek yang bersesuaian misalnya *U* dan *V* untuk mendapatkan *cluster* (*UV*). Untuk langkah (3) dari algoritma di atas jarak-jarak antara *cluster* (*UV*) dan *cluster W* yang lain dihitung dengan

$$d_{(UV)W} = maks\{d_{UW}, d_{VW}\}$$

Di sini besaran-besaran d_{UW} dan d_{VW} berturut-turut adalah jarak antara tetangga terdekat *cluster-cluster U* dan *W* dan juga *cluster-cluster V* dan *W*.

c. *Average Linkage*

Average linkage memperlakukan jarak antara dua *cluster* sebagai jarak rata-rata antara semua pasangan item-item di mana satu anggota dari pasangan tersebut kepunyaan tiap cluster. Mulai dengan mencari matriks jarak $D = \{d_{ik}\}$ untuk memperoleh objek-objek paling dekat (paling mirip) misalnya *U* dan *V* . Objek objek ini digabungkan untuk membentuk *cluster (UV)*. Untuk langkah (3) dari algoritma di atas jarak-jarak antara (*UV*) dan *cluster W* yang lain ditentukan oleh

$$d_{(UV)W} = \frac{\sum_i \sum_k d_{ik}}{N_{(UV)} N_W}$$

Gambar 5. *Average Lingkage*

di mana d_{ik} adalah jarak antara objek *i* dalam *cluster (UV)* dan objek *k* dalam *cluster W* , N_{uv} dan N_w berturut-turut adalah banyaknya item-item dalam *cluster* (*UV*) dan *W*.

6. *Self Organizing Map* (SOM)

Walaupun proses *learning* yang dilakukan SOM mirip dengan *Artificial Neural Networks*, tetapi proses untuk meng-*assign input* data ke *map*, lebih mirip dengan K-*Means* dan kNN *Algorithm*. Adapun prosedur yang ditempuh dalam melakukan *clustering* dengan SOM adalah sebagai berikut:

1. Tentukan *weight* dari input data secara random
2. Pilih salah satu *input* data
3. Hitung tingkat kesamaan (dengan *Eucledian*) antara *input* data dan *weight* dari *input* data tersebut dan pilih *input* data yang memiliki kesamaan dengan *weight* yang ada (data ini disebut dengan *Best Matching Unit* (BMU))

4. Perbaharui *weight* dari *input* data dengan mendekatkan *weight* tersebut ke BMU dengan rumus:

$$Wv(t+1) = Wv(t) + Theta(v, t) \times Alpha(t) \times (D(t) - Wv(t))$$

Dimana :

a. *Wv(t)*: *Weight* pada saat ke-t

b. *Theta (v, t)*: Fungsi *neighbourhood* yang tergantung pada *Lattice distance* antara BMU dengan *neuron* v. Umumnya bernilai 1 untuk *neuron* yang cukup dekat dengan BMU, dan 0 untuk yang sebaliknya. Penggunaan fungsi *Gaussian* juga memungkinkan.

c. *Alpha (t)*: *Learning Coefficient* yang berkurang secara *monotonic*

d. *D(t)*: *Input* data

e. Tambah nilai t, sampai t < *Lambda*, dimana *Lambda* adalah jumlah iterasi

7. Perbandingan Algoritma *Clustering*

Perbandingan algoritma *clustering* dijelaskan pada tabel 1.

Tabel 1 Perbandingan Algoritma *Clustering*

Metode	Algoritma	Karakteristik
Hierarchical Clustering	1. Kelompokkan setiap objek dalam sebuah *cluster*. 2. Temukan pasangan yang paling mirip untuk dimasukkan ke dalam *cluster* yang sama dengan melihat data dalam matriks kemiripan. 3. Kedua objek kemudian digabungkan dalam satu *cluster*. 4. Ulangi dari langkah kedua dan ketiga hingga tersisa sebuah *cluster*.	1. Memberikan hasil variasi kelompok yang banyak, mulai dari masing-masing data sebagai satu kelompok hingga saat semua data bergabung sebagai kelompok tunggal. 2. Metode ini biasanya digunakan untuk alasan pendasar aplikasi, seperti pembuatan taksonomi yang membutuhkan hierarki pengelompokan data. 3. Karena menggunakan teknik yang rakus dalam perosesnya, komputasi metode ini mahal dan kompleks. 4. Penggabungan dua kelompok merupakan keputusan final karena dua kelompok yang sudah digabung tidak bisa dikembalikan seperti semula. 5. Bisa terjadi masalah untuk set data yang mengandung *noise*, dan data berdimensi tinggi. Biasanya, untuk masalah ini dibantu dengan metode lain secara parsial, seperti *k-means.*
K-Means	1. Tentukan jumlah *cluster* 2. Alokasikan data ke dalam *cluster* secara	1. *K-means* merupakan metode pengelompokan yang sederhana dan dapat digunakan dengan mudah.

Metode	Algoritma	Karakteristik
	random 3. Hitung *centroid*/rata-rata dari data yang ada di masing-masing *cluster* 4. Alokasikan masing-masing data ke *centroid*/rata-rata terdekat 5. Kembali ke Step 3, apabila masih ada data yang berpindah *cluster*	2. Pada jenis set data tertentu, *k-means* tidak dapat melakukan segmentasi data dengan baik dimana hasil segmentasinya tidak dapat memberikan pola kelompok yang mewakili karakteristik bentuk alami data. 3. *K-means* bisa mengalami masalah ketika mengelompokan data yang mengandung *outlier*.
SOM	1. Tentukan *weight* dari *input* data secara random 2. Pilih salah satu input data 3. Hitung tingkat kesamaan (dengan *Eucledian*) antara *input* data dan *weight* dari *input* data tersebut dan pilih *input* data yang memiliki kesamaan dengan *weight* yang ada (data ini disebut dengan *Best Matching Unit* (BMU) 4. Perbaharui weight dari input data dengan mendekatkan *weight* tersebut	1. SOM dapat memvisualkan hasil pengelompokan dalam bentuk topografi dua dimensi layaknya peta sehingga memudahkan pengamatan distribusi kelompok hasil pengelompokan. 2. Memerlukan penentuan fungsi keterangan, laju pembelajaran, fungsi pembelajaran, jumlah kelompok, dan jumlah iterasi yang diinginkan. Untuk penentuan parameter ini bisa digunakan cara coba-coba dengan sejumlah nilai, kemudian pilih yang terbaik. 3. Hanya cocok untuk data yang sudah diketahui jumlah kelompoknya dengan mengamati bentuk alami distribusi data. 4. Dalam memberikan hasil pengelompokan, SOM tidak menggunakan fungsi objektif tertentu seperti k-*means* dan *fuzzy* c-*means* sehingga untuk suatu kondisi yang sudah optimal pada suatu iterasi, SOM tidak akan menghentikan iterasinya selama jumlah iterasi yang ditentukan belum tercapai. Hal ini juga berlaku ketika hasil kelompok yang didapatkan belum optimal, tetapi jumlah iterasi yang ditentukan sudah tercapai sehingga hasilnya menjadi kurang sesuai dengan yang diharapkan (belum optimal). Oleh karena itu, SOM tidak menjamin konvergensi hasil pengelompokan.

8. Implementasi Data Uji

Implementasi data uji menjelaskan mengenai struktur tabel penyusunnya adapun pembuatan data dilakukan dengan menggunakan *spreadsheet* program, data uji ini merupakan tabel yang diolah menggunakan *software Microsoft Offices Excel*. Contoh data di bawah ini yang dibuat

oleh *software* tersebut. Adapun contoh implementasi dapat dilihat pada gambar 3 di bawah ini :

	A	B	C	D	E	F	G	H	I	J	K
1	product_id	customer_id	store_id	promotion_id	mount_of_year	quarter	year	store_sales	store_cost	unit_sales	fact_count
2	1	157	24	1869	12	Q4	1997	8.55	2.9925	3	1
3	1	456	15	0	6	Q2	1997	11.4	4.332	4	1
4	1	638	11	0	9	Q3	1997	8.55	2.9925	3	1
5	1	916	7	0	4	Q2	1997	11.4	4.902	4	1
6	1	923	15	0	7	Q3	1997	8.55	2.736	3	1
7	1	1312	3	0	5	Q2	1997	8.55	3.6765	3	1
8	1	1565	24	0	9	Q3	1997	8.55	4.1895	3	1
9	1	2270	11	0	11	Q4	1997	8.55	4.0185	3	1
10	1	3065	3	0	11	Q4	1997	5.7	2.508	2	1
11	1	3441	3	0	8	Q3	1997	8.55	3.42	3	1
12	1	3528	17	0	10	Q4	1997	8.55	3.8475	3	1
13	1	4461	11	0	4	Q2	1997	8.55	2.9925	3	1
14	1	4707	11	0	12	Q4	1997	8.55	4.0185	3	1
15	1	4728	7	501	1	Q1	1997	11.4	3.99	4	1
16	1	5313	24	0	3	Q1	1997	8.55	3.762	3	1
17	1	5607	6	0	6	Q2	1997	11.4	4.902	4	1
18	1	5929	15	0	8	Q3	1997	14.25	5.5575	5	1
19	1	6248	24	1860	8	Q3	1997	11.4	3.876	4	1
20	1	6666	17	0	2	Q1	1997	8.55	4.1895	3	1
21	1	7704	3	0	7	Q3	1997	5.7	2.508	2	1
22	1	8202	3	0	12	Q4	1997	8.55	4.104	3	1
23	1	9169	23	0	5	Q2	1997	11.4	5.358	4	1
24	1	9358	15	0	7	Q3	1997	8.55	4.275	3	1
25	1	9652	14	0	9	Q3	1997	5.7	1.881	2	1
26	1	9788	13	0	1	Q1	1997	8.55	4.0185	3	1

data

Gambar 4. Data Uji (menggunakan *software spreadsheet*)

9. Hasil Pengujian

8.1 Pengujian Metode *Clustering*

Data set yang digunakan untuk menguji algoritma klasterisasi diperoleh dari situs: (Http://kdd.ics.uci.edu/) atau dari situs lain, yaitu, (http://www.kdnuggets.com/datasets). Data set untuk menguji algoritma klasterisasi adalah *time series*. Data set ini disimpan dalam file ASCII, 600 baris, 60 kolom, untuk membedakan data set besar dan kecil data set dibagi dua kelompok menjadi kumpulan data (200 baris dan 20 kolom). Ketiga metode *clustering* dibandingkan berdasarkan faktor-faktor yang terdapat pada tabel 1 sebagai berikut:

Tabel 2. Pengujian data *Clustering*

Metode	Dataset	Klaster
Hierarchical	Basar dan Kecil	Banyak dan Sedikit
K-*means*	Basar dan	Banyak dan

	Kecil	Sedikit
SOM	Basar dan Kecil	Banyak dan Sedikit

Menurut jumlah *cluster* k kecuali untuk metode hierarki, semua algoritma klasterisasi dibandingkan membutuhkan pengaturan k. Disini, kinerja algoritma yang berbeda untuk berbagai k dibandingkan untuk menguji kinerja yang terkait dengan k. Untuk menyederhanakan dan untuk membuat perbandingan lebih mudah, k yang dipilih sama dengan 8, 16, 32, dan 64. Untuk membandingkan *hierarchical clustering* dengan algoritma lain, pohon hierarki dipotong pada dua tingkat yang berbeda untuk mendapatkan nomor yang sesuai cluster (8, 16, 32 dan 64). hasilnya, sebagai nilai k menjadi lebih besar kinerja algoritma SOM menjadi lebih rendah. Namun, kinerja k-*means* algoritma menjadi lebih baik dari algoritma hierarki. Dapat dilihat pada tabel 2 di bawah ini

Tabel 3. Hubungan antara jumlah *cluster* dan kinerja algoritma.

Number Of Cluster	**Performance**		
	Hirarki	K-means	SOM
8	65	63	59
16	74	71	67
32	87	84	78
64	92	89	85

Menurut ukuran data set, data set besar digunakan terdiri dari 600 baris dan 60 kolom dan data set kecil menggunakan 200 baris dan 20 kolom. Data set kecil

diekstraksi sebagai bagian dari dataset besar. Kualitas k-*means* menjadi sangat baik ketika menggunakan data set besar. Dua algoritma *hierarchical clustering* dan algoritma SOM menunjukkan hasil yang baik bila menggunakan data set kecil, hasil pengujian dapat dilihat pada tabel 3 di bawah ini.

Tabel 3. Pengaruh ukuran data pada algoritma.

K=32			
Data Size	Hirarki	K-*means*	SOM
36000	850	910	830
4000	91	95	89

10. Kesimpulan

Setelah menganalisis hasil pengujian algoritma *clustering* dan menjalankan algoritma tersebut dengan faktor dan situasi yang berbeda, maka kesimpulan yang diperoleh sebagai berikut:

1. Algoritma SOM menghasilkan akurasi yang lebih baik dalam mengelompokan objek ke dalam kelompok yang cocok dari pada algoritma k-*means* dan *Hierarchical*.
2. Algoritma *K-means* menjadi sangat baik ketika menggunakan data set besar dibanding dengan algoritma SOM dan *Hierarchical*.
3. Pengelompokan *hierarchical* dan SOM menunjukkan hasil yang baik saat menggunakan data set kecil dibanding menggunakan algoritma k-*means*.
4. Sebagai kesimpulan umum, algoritma partisi (seperti k-*means*) yang direkomendasikan untuk data set besar sementara algoritma *hierarchical clustering* dan SOM

yang direkomendasikan untuk data set kecil.

5. Aplikasi bermanfaat untuk mengetahui pengelompokan data set yang dihasilkan oleh algoritma k-*means, hierarchical* dan SOM.

Daftar Pustaka

Santosa, Budi. 2007. *Data Mining. Teknik Pemanfaatan Data untuk Keperluan Bisnis,* First Edition ed. Yogyakarta: Graha Ilmu.

Prasetyo, Eko. 2012. *Data Mining Konsep dan Aplikasi Menggunakan Matlab.* Yogyakarta : penerbit andi.

Eisen, M. 1998. *Cluster and Tree View Manual*. Stanford University. Japan.

Abu Abbas, Osama. 2007. *Comparisons Between Data Clustering Algorithms.* Computer Science Department, Yarmouk University, Jordan

K. Arai and A. R. Barakbah. 2007. "*Hierarchical K-means: an algorithm for centroids initialization for Kmeans,*". Saga University.

Alfina, Tahta. 2012. *Analisa Perbandingan Metode Hierarchical Clustering, K-means dan Gabungan Keduanya dalam Cluster Data.* Institut Teknologi Sepuluh Nopember.

Latiffaturrahman. 2010. *perbandingan hasil penggerombolan metode k-means, fuzzy k-means dan two step cluster.* Institut Pertanaian Bogor.

Wahanani, Nursinta Adi. 2012. *Optimasi Clustering K-Means Dengan Algoritma Genetika Multiobyektif.* Institut Pertanaian Bogor.

Shandy, Liesca Levy. 2008. *Perbandingan Metode Diskretisasi Data Partisi Intuitif dan K-Means Clustering Terhadap Pembuatan Pohon Keputusan.* Institut Pertanaian Bogor.

Edward. 2006. *Clustering Menggunakan Self Organizing Maps.* Institut Pertanaian Bogor.

Fatansyah. 1999. *Basis Data*. Bandung: Informatika.

ISSN 1979-8911