IMPLEMENTASI DAN ANALISA GRANULAR SUPPORT VECTOR MACHINE DENGAN DATA CLEANING (GSVM-DC) UNTUK E-MAIL SPAM FILTERING

Moh.Mahsus[1], ZK. Abdurahman Baizal, SSi., M.Kom.[2], Shaufiah, ST., MT.[3]
[1,3]Program Studi Teknik Informatika, Fakultas Informatika, Institut Teknologi Telkom, Bandung
[2]Program Studi Ilmu Komputasi, Fakultas Sains, Institut Teknologi Telkom, Bandung
Jl Telekomunikasi No 1, Terusan Buah Batu, Bandung
Telp. (022)7564108
Email : [1]manshoezz@telkom.net, [2]bayzal@gmail.com, [3]ufi@ittelkom.ac.id

ABSTRAK
E-mail spam adalah pembanjiran internet dengan banyak salinan pesan yang sama, dan memaksa pengguna internet untuk menerimanya walupun itu tidak dinginkan. Pengguna e-mail membutuhkan waktu yang lebih lama untuk membaca dan memutuskan apakah e-mail yang diterima tersebut adalah spam atau bukan. Berdasarkan motivasi tersebut e-mail spam filtering banyak dikembangkan. Banyak teknik yang digunakan untuk pembangunan sebuah e-mail spam filtering, dan salah satunya dengan menggunakan metode Support Vector Machines karena terbukti memiliki kemampuan generalisasi yang baik. Untuk meningkatkan akurasi dan efisiensi pemrosesan, SVM dimodifikasi dengan paradigma granular computing dengan memecah data menjadi bagian-bagian kecil informasi serta metode data cleaning yang digunakan untuk mengurangi jumlah data yang akan dilatih. Dengan modifikasi yang disebut Granular Support Vector Machines with Data Cleaning ini diperoleh hasil akurasi sebesar 97,2%, lebih baik daripada menggunakan SVM biasa dengan hasil akurasi sebesar 96,6%.

Kata kunci: e-mail spam, support vector machines, granular computing, data cleaning

1. PENDAHULUAN

1.1 Latar Belakang

E-mail spam filtering adalah suatu program yang digunakan untuk mendeteksi *e-mail* yang tidak diinginkan dan tidak diminta dan mencegah *e-mail* - tersebut untuk masuk kedalam kotak masuk pengguna *e-mail* (techterms, 2009). Banyak teknik yang digunakan untuk membuat *e-mail spam filtering*, salah satunya dengan menggunakan teknik klasifikasi (G. Sudipto, 2003).

Klasifikasi adalah salah satu teknik dalam *data mining* yang digunakan untuk memprediksi kelompok keanggotaan(*class*) dari setiap *instance* data. Teknik klasifikasi yang biasa digunakan untuk membangun *e-mail spam filtering* diantaranya adalah *Naïve Bayes*, *support vector machine* (SVM) dan *k-Nearest Neighbor* (kNN). Dari beberapa teknik tersebut yang paling sering digunakan untuk klasifikasi data adalah SVM.

Support Vector Machine (SVM) adalah suatu metode *supervised learning* yang digunakan untuk melakukan klasifikasi data. *Support vector machine* (SVM) memiliki hasil yang bagus dalam klasifikasi data, akan tetapi hasil tersebut akan turun ketika dataset yang digunakan mengandung banyak atribut sehingga ketika dipetakan kedalam ruang vektor akan menimbulkan *curse of dimensionality*. Selain hal tersebut, performansi SVM akan turun jika dalam suatu *dataset* jumlah *class* yang satu dengan *class* yang lain sangat berbeda jauh atau disebut *imbalance dataset*. Kondisi tersebut menyebabkan *class* yang sedikit tersebut dianggap sebagai pencilan(*outlier*) atau bahkan tidak dianggap. Penurunan performansi SVM juga akan terjadi ketika pemilihan *support vector* (SV) yang digunakan untuk pembangunan *hyperplane* banyak terdapat data yang seharusnya bukan SV tetapi dianggap SV(*noise*) jika hanya dibangun sebuah SVM untuk keseluruhan *dataset* yang digunakan. Untuk mengatasi hal - hal tersebut, maka banyak dilakukan modifikasi terhadap SVM dan juga untuk meningkatkan efektifitas dan efisiensi SVM (G. Sudipto, 2003).

Salah satu penerapan modifikasi pada SVM adalah dengan cara penggunaan paradigma *granular computing* dan teori statistik yang kemudian gabungan dari kedua cara tersebut disebut *Granular Support Vector Machines* (GSVM). Terdapat beberapa metode yang ditambahkan pada GSVM sesuai dengan tujuan pembangunan SVM. Salah satunya adalah metode *Data Cleaning* yang berfungsi untuk mengurangi jumlah data masukan yang akan digunakan untuk proses *training* SVM (Y.C. Tang, 2006)

Dataset yang digunakan adalah *dataset* EMCL PKDD 2006 *Discovery Challenge Task A* dengan rincian data *training* sebanyak 4000 data dan data *testing* sebanyak 1000 data. *Dataset* yang digunakan sebelumnya telah melalui proses *preprocessing* untuk dilakukan ekstraksi dan normalisasi. Ekstraksi atribut dilakukan dengan menggunakan modul *information gain* pada WEKA 3.5 dan normalisasi menggunakan *Term Frequency*(TF). Aplikasi yang dibangun merupakan *stand alone application* dan tidak di implementasikan dalam *e-mail server*.

Penelitian ini bertujuan untuk merancang dan membangun sistem *e-mail spam filtering* menggunakan *Granular Support Vector Machines With Data Cleaning*, serta melakukan analisa akurasi dan efisiensi waktu pada sistem yang dibangun dengan membandingkan ukuran *F-Measure* dan waktu pemrosesan dengan SVM yang tidak dimodifikasi. Selain itu juga akan dilakukan analisa jumlah *support vector* yang ditemukan dengan akurasi pengujian terhadap waktu *training* pada GSVM-DC.

2. LANDASAN TEORI

2.1 *Klasifikasi*

Klasifikasi adalah adalah salah teknik dalam data *mining* yang digunakan untuk memprediksi kelompok keanggotaan(*class*) dari setiap *instance* data. Tujuan dari klasifikasi adalah untuk melakukan analisa terhadap data *training* dan membentuk sebuah model yang digunakan untuk klasifikasi data yang akan dilakukan

Sebelum proses *training*, dokumen yang akan di klasifikasi, dilakukan transformasi data terlebih dahulu. Dokumen yang berbentuk kata – kata di ubah menjadi bentuk yang sesuai dengan klasifikasi yang akan dibangun (H. Andreas, S. Gerd, 2003). Representasi ini disebut matriks *co-occurrence*, yaitu matriks frekuensi kemunculan *term* pada tiap dokumen. Kekurangan dari representasi tersebut adalah dimensinya yang sangat besar sehingga menurunkan peformansi dari klasifikasi [8]. *Information gain* biasa digunakan dalam proses pemilihan atribut yang memiliki kontribusi besar dalam proses klasifikasi.

$$IG(t) = -\sum_{i=1}^{m} P(c_i)\log P(c_i) + P(t)\sum_{i=1}^{m} P(c_i|t)\log P(c_i|t) + P(\bar{t})\sum_{i=1}^{m} P(c_i|\bar{t})\log P(c_i|\bar{t}) \quad (1)$$

2.2 *Support Vector Machines*

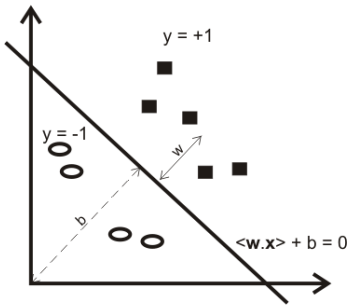
Support Vector Machine (SVM) adalah suatu metode *supervised learning* yang digunakan untuk melakukan klasifikasi data. *Support Vector Machine* bekerja dengan cara memetakan dataset masukan(*input space*) kedalam vektor yang berdimensi tinggi untuk kemudian dilakukan pemisahan antar *class* dengan menggunakan *hyperplane*. Pembentukan *hyperplane* ini berdasarkan *support vector*(SV) yang merupakan data yang memiliki jarak paling dekat dengan *hyperplan* yang dibangun (B. Steve, 2003). Untuk pembangunan SVM, diberikan data *training* dalam bentuk persamaan 2.

$$\mathcal{D} = \{(x^1, y^1), \dots, (x^l, y^l)\}, x \in \mathbb{R}^n, y \in \{-1,1\} \quad (2)$$

Dengan *hyperplane*:

$$(w.x) + b = 0 \quad (3)$$

Persamaan 3 adalah *hyperplane* yang akan dibangun untuk memisahkan *class* 1 dan -1. Jika persamaan tersebut dipenuhi, maka *input space* tersebut bisa dikatakan sebagai data yang *linearly separable* seperti pada gambar 1.



Gambar 1. Support vector machines

Fungsi *sign(x)* yang dihasilkan dari perhitungan SVM adalah:

$$f(x) = \begin{cases} -1, & \langle \boldsymbol{w}.x\rangle + b < 0 \\ +1, & \langle \boldsymbol{w}.x\rangle + b > 0 \end{cases} \quad (4)$$

2.3 OPTIMISASI SVM

Optimasi SVM dilakukan untuk membentuk *hyperplane* yang tepat agar menghasilkan akurasi yang tinggi dalam pembentukan SVM. Terdapat 2 optimasi yang dapat dilakukan untuk membentuk *hyperplane*. Yang pertama yaitu *primal form* SVM yang ditunjukkan pada persamaan (5).

$$\min_{\arg\, \boldsymbol{w},b} {}^1\!/_2\,(\boldsymbol{w}.\boldsymbol{w}) \quad subject\ to\ (\langle \boldsymbol{w}.x\rangle + b)y_i \geq 1 \quad (5)$$

Untuk membentuk *hyperplane* dengan *primal form* sulit dilakukan, karena ada dua *variable* yang masih belum ada nilainya yaitu *w* dan *b*.

Bentuk optimasi berikutnya adalah dengan menggunakan *dual form* SVM dengan menggunakan pendekatan **lagrangian**. Bentuk ini merupakan modifikasi dari bentuk *primal form* dengan menggunakan lagrang untuk pembentukan *hyperplane* dengan menambahkan konstanta $\boldsymbol{\alpha}$ dimana konstanta tersebut digunakan untuk memberikan tanda apakah suatu data tersebut *support vector* ($\alpha \geq 0$) atau bukan support vector ($\alpha < 0$). Semakin besar nilai $\boldsymbol{\alpha}$, maka semakin mirip dengan *Primal form SVM* (B. Steve, 2003). Bentuk persamaannya seperti pada persamaan (6).

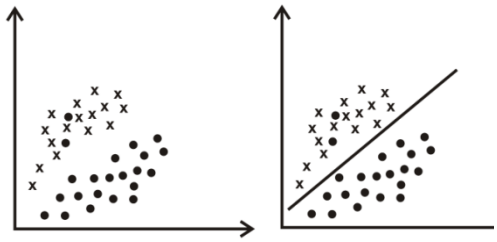
$$\max L_D = \sum_{i=1}^{l} \alpha_i - \sum_{i,j=1}^{l} \alpha_i\, \alpha_j y_i y_j \langle \boldsymbol{x}.\boldsymbol{x}\rangle$$

$$subject\ to\ \alpha_i \geq 0 \qquad (6)$$

2.4 Soft Margin

Dalam beberapa contoh SVM, rata – rata melakukan asumsi bahwa suatu data dapat di pisahkan dengan garis lurus atau data tersebut adalah *linearly separable.* Dalam kenyataanya, jarang ditemukan data yang benar – benar *linearly separable* (S. N. William, 2006). Ketika *dataset* memiliki *error* / *noise*, yang menyebabkan *misclassification*, maka perlu melakukan modifikasi SVM dengan menambahkan *soft margin* (S. N. William, 2006).

Untuk membangunan SVM menggunakan *dual form*, ditambahkan parameter *C* yang digunakan untuk membatasi nilai $\boldsymbol{\alpha}$ untuk data yang *non-linearly separable* (C. Nello, S. John, 2000)**.** Hal ini dilakukan agar didapatkan *hyperplane* yang dapat mentolelir ketika terdapat data yang terletak pada daerah *hyperplane* yang salah. Contoh hasil klasifikasi menggunakan parameter pembatas nilai $\boldsymbol{\alpha}$ ditunjukkan oleh gambar 2.



Gambar 2. Soft margin

Konstanta *C*, atau yang disebut *soft margin* ditambahkan dalam konstrain pada *dual form* SVM, sehingga persamaan *dual form* SVM akan menjadi

$$\max L_D = \sum_{i=1}^{l} \alpha_i - \sum_{i,j=1}^{l} \alpha_i\, \alpha_j y_i y_j \langle \boldsymbol{x}.\boldsymbol{x} \rangle$$
$$subject\ to\ 0 \leq \alpha_i \leq C$$
$$\sum_{i=1}^{l} \alpha_i\, y_i = 0 \qquad (7)$$

2.5 Granular Computing

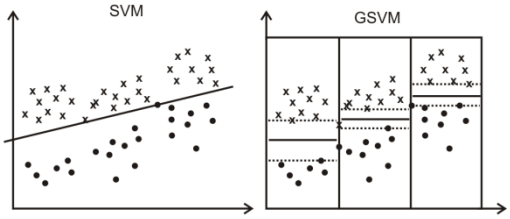
Granular computing adalah pemecahan masalah kedalam bentuk yang lebih kecil atau disebut *information granule*, kemudian menyelesaikan masalah pada tiap – tiap *information granule* tersebut (Y.C. Tang, 2006). Dalam permasalahan *data mining*, banyak algoritma yang digunakan untuk membentuk *information granule*, antara lain *decision trees algorithm* ,*association rule* dan *clustering algorithm* (Y.C. Tang, 2006).

2.5.1 Granular Support Vector Machines

Salah satu modifikasi yang dilakukan dalam SVM adalah menggunakan paradigma *granular computing*, yaitu dengan memecah *input space* menjadi bentuk yang lebih kecil (*information granule*) seperti pada gambar 3 dengan tujuan utamanya adalah memungkinkan data *non-linearly separable* menjadi *linearly separable (*Y.C. Tang, 2006), memperbesar tingkat kemampuan generalisasi SVM yang dibangun (Y.C. Tang, 2006), dan meningkatkan akurasi dari SVM (Y.C. Tang, 2006).

Langkah-langkah umum pembentukan GSVM-DC adalah sebagai berikut:

1. *Granulation*, yaitu membagi *input space* menjadi *information granule*.
2. Pemilihan *local support vector* (LSV) dan parameter *C* pada tiap *information granule* yang di buat.
3. Menggabungkan hasil LSV dan melakukan *training* dengan semua parameter *C* yang terpilih pada setiap *information granule* untuk membangun sebuah sistem.



Gambar 3 Perbandingan SVM dan GSVM pada linearly separable data

2.5.2 *Fuzzy C-Means Clustering*

Implementasi *granulation* dalam langkah pertama pembentukan GSVM digunakan *clustering algorithm*, yaitu *fuzzy c means clustering*. *Fuzzy C-Means clustering* adalah salah satu metode *clustering* yang memperbolehkan suatu data ada di dua atau lebih *cluster*(*overlap*) (Clustering, 2009). *Information granule* yang dibangun harus *overlap*, karena setiap hasil keluaran dari *information granule* digabungkan kembali untuk membentuk sebuah model klasifikasi yang digunakan untuk membangun sistem, maka dipilihlah *fuzzy c-means clustering* karena memungkinkan data untuk *overlap*. Langkah – langkah dalam *fuzzy c-means clustering* adalah (Clustering, 2009):

1. Tentukan jumlah *cluster / information granule* yang akan di bentuk.
2. Tentukan nilai dari *fuzziness parameter(wight)* > 1, yaitu nilai yang menentukan berapa banyak cluster yang mungkin terjadi *overlap* (G. Sudipto, 2003).
3. Tentukan maksimum iterasi(*t*) yang akan di lakukan.
4. Tentukan kriteria penghentian iterasi(ε)[0..1].
5. Inisialisasi matrik *V* (matrik *cluster centroid*) dengan 0.

6. Lakukan inisialisai nilai dari matrik *U*(matrik keanggotaan), biasanya dilakukan secara *random*[0..1] untuk pertama kali, dan pastikan hasil penjumlahan nilai kolom dari matrik *U* =1 dan penjumlahan baris dari matrik *U* > 1.
7. Mulai iterasi dari *i* = 1.
8. Perbaharui matrik *V* untuk tiap – tiap *cluster*.
9. Perbaharui matrik U^i dan pastikan hasil penjumlahan nilai kolom dari matrik U^i =1 dan penjumlahan baris dari matrik U^i > 1.
10. Tentukan kriteria perhentian iterasi dengan:
    a. $max||U^i - U^{i-1}|| < \varepsilon$.
    b. *i* = maksimum iterasi(*t*).
11. Jika tidak terpenuhi kriteria pemberhentian iterasi ulangi dari langkah ke-8.

2.5.3 Data Cleaning

Data cleaning adalah proses pengurangan data yang akan di proses. Salah satu masalah ketika kita memproses data latih yang cukup besar adalah waktu eksekusi yang memakan waktu lama. Oleh karena itu, untuk meningkatkan efisiensi dilakukan *data cleaning* pada data latih (Y.C. Tang, 2006).

Dalam proses *data cleaning*, ada 2 kemungkinan yang akan terjadi (Y.C. Tang, 2006):

1. Hilangnya informasi karena hilangnya informasi / data yang berguna untuk bentuk *classifier*.
2. Data menjadi bersih karena data yang tidak relevan, redundan, noise di hilangkan sehingga dapat meningkatkan peforma *classifier*

langkah – langkah *data cleaning* seperti pada gambar 2-4 adalah:

1. *k* = jumlah *information granule* yang dibentuk, dimana *k* adalah jumlah *cross validation* yang dilakukan untuk mengoptimasi parameter SVM.
2. Untuk tiap – tiap *information granule* yang terbentuk, tentukan *local support vector(*LSV*)* dan nilai *C* yang paling bagus untuk tiap – tiap *information granule* yang terbentuk dengan *linear* SVM.
3. Lakukan *cross-validation* untuk mengoptimasi parameter SVM yangdipilih, dimana tiap – tiap LSV(*i*) yang terpilih dijadikan *training datasets* dan *information granule*(*k*) digunakan sebagai *validation datasets.*
4. Kemudian gabungkan seluruh *LSV* dan gunakan parameter yang terpilih di langkah sebelumnya untuk membangun *SVM* yang optimal.

2.6 Implementasi

Proses yang dilakukan sistem dalam melakukan klasifikasi *e-mail spam* adalah melakukan *parser* data, untuk mengubah format data asli menjadi format *comma separated value*(CSV). Kemudian melakukan ekstraksi data untuk mengurangi jumlah atribut menggunakan modul *Information Gain* dari aplikasi Weka 3.5. Normalisasi data menggunakan persamaan *Term Frequency* (TF). Selanjutnya melakukan *training* data dengan *Granular Support Vector Machines with data cleaning*, untuk menemukan *hyperplanes* terbaik, dengan SVM dibangun dari *toolbox* LIBSVM. Hal ini ditunjukkan gambar 1 pada lampiran.

2.7 Kinerja Sistem

Evaluasi yang akan dilakukan menggunakan parameter *F-Measure* yang terdiri dari perhitungan *precision* dan *recall*. *Recall, precision*, dan *F-Measure* merupakan metode pengukuran efektifitas yang biasa dilakukan pada proses klasifikasi (H. Eyke, 2008). Dalam tugas akhir ini, nilai *precision* dan *recall* didasarkan pada hasil keluaran menggunakan *confusion matrix* seperti pada tabel 1.

Tabel 1. *Confusion matrix*

prediction	actual	
	positive	negative
positive'	True Positives (TP)	False Positives (FP)
negative'	False Negatives (FN)	Ture Negativas (TN)

Pada *confusion matrix* sendiri, untuk menentukan akurasi pengujian digunakan persamaan (Y.C. Tang, 2006):

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (8)$$

$$Precision\ (p) = \frac{TP}{TP + FP} \quad (9)$$

$$Recall\ (r) = \frac{TP}{TP + FN} \quad (10)$$

$$F - Measure = \frac{2 \times p \times r}{p + r} \quad (11)$$

3. PENGUJIAN DAN ANALISIS

3.1 Pengujian Performansi

Pengujian ini dilakukan untuk meneliti pengaruh jumlah *information granule* terhadap waktu *training* dan akurasi, jumlah LSV terhadap akurasi, mencari nilai konstanta *C* (*soft margin*) dan jumlah *local support vector*(LSV) terbaik, dan menganalisia performansi sistem *e-mail spam filtering* yang telah dibangun. Hasil pengukuran berdasarkan pada waktu pemrosesan, akurasi, *precision*, *recall*, dan *F-Measure*. Pengujian dengan GSVM-DC dibandingkan dengan *linear*-SVM untuk melihat performansinya.

3.2 Dokumen Uji

Dataset yang digunakan dalam pengujian ini merupakan *dataset* PKDD 2006 bagian *evaluation task_a*, dimana *dataset evaluation task_a* yang digunakan adalah gabungan dari task_a_u00_eval_lab. yang berisikan 2500 *e-mail* dan task_a_u01_eval_lab yang juga berisikan 2500 *e-mail*. Data asli dari PKDD, terdiri dari 49179 atribut. *Dataset* tersebut digabung dan digunakan sebagai dataset *training* dan *testing* dimana 4000 e-mail digunakan sebagai data *training* dan 1000 lainnya digunakan untuk data *testing*.

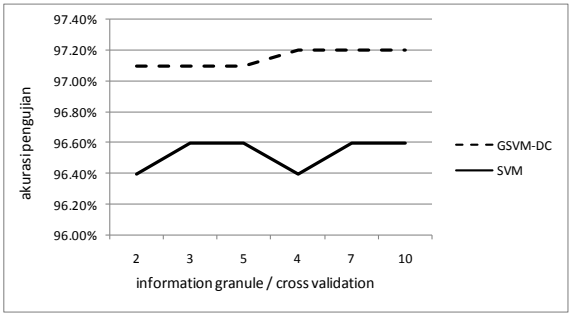
Gabungan *dataset* ini telah dilakukan *processing* dengan melakukan parser data menjadi format *comma separated value*(CSV). dilakukan perhitungan *information gain* menggunakan *tool* Weka 3.5 kemudian dilakukan pembobotan menggunakan *term frequency* untuk normalisasi data sehingga menyederhanakan perhitungan. Hasil *information gain* menggunakan Weka diperoleh 1454 atribut, dengan menghilangkan atribut yang nilai *information gain*-nya nol (0.0000)

3.3 Analisis Hasil Pengujian

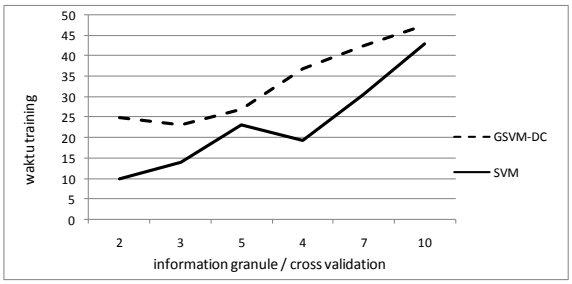
3.3.1 Analisis Pengaruh Jumlah Information granule terhadap Akurasi dan Efisiensi

Pada pengujian ini dipilah nilai epsilon(ε) = 0.1, *fuzziness parameter(weight)* = 2 dan parameter *C* di acak sebanyak 3 kali untuk setiap *information granule* yang di bentuk dengan nilai 0 sampai dengan 1.sedangkan jumlah information granule yang di bentuk dibatasi sejumlah 2,3,4,5,7 dan 10. Berdasarkan sepuluh kali pengujian pada masing-masing jumlah *information granule* untuk GSVM-DC dan linear SVM, diperoleh data empirik seperti pada gambar 4 dan gambar 5 dengan hasil sebagai berikut:

- Untuk GSVM-DC, hasil pengujian terbaik diperoleh pada parameter SVM(*C*) = 0.718852837745963, jumlah *information granule* = 4, *fuzzyness parameter* = 2, *epsilon*(ε) = 0.1 dengan waktu training 36.955026128 detik dan akurasi pengujian 97.20%.
- Untuk *linear*-SVM diperoleh hasil waktu *training* sebesar 19.30077503 detik, akurasi pengujian sebesar 96.40%.



Gambar 4. Grafik perbandingan jumlah information granule dengan akurasi pengujian



Gambar 5. Grafik perbandingan jumlah information granule dengan waktu training

3.3.2 Analisis GSVM-DC yang dibangun

Pada proses granulasi menggunakan F*uzzy C-Means clustering*, ditemukan beberapa kasus yang menyebabkan proses *training* tidak selesai dikarenakan:

1. Ditemukan 1 atau lebih *cluster* yang tidak memiliki anggota.
2. Ditemukan *cluster* yang hanya memiliki anggota dengan *class* yang sama.

Kedua hal diatas menyebabkan *information granule* yang terbentuk tidak dapat di lakukan *training* untuk mencari parameter SVM dan LSV terbaik untuk tiap – tiap *information granule*, karena untuk mendapatkannya digunakan *linear* SVM, dan SVM hanya bias di *training* jika dalam suatu kumpulan data terdapat minimal 2 *class* yang berbeda.

Untuk menghindari hal itu terjadi, maka ada beberapa cara, antara lain:

1. Proses granulasi akan diulang dari iterasi pertama.
2. Sistem menyimpan hasil iterasi sebelumnya, dan hasil tersebut digunakan sebagai hasil akhir dari proses granulasi. Jika kondisi tersebut ditemukan pada iterasi pertama, maka dilakukan cara 1.
3. *Information granule* tersebut dianggap tidak ada.
4. Jika ditemukan kondisi 2, maka data yang berada di cluster tersebut dilakukan perhitungan Euclidean distance dengan matrik V(matrik cluster centroid) terdekat. Kemudian pindahkan data tersebut ke cluster yang memiliki nilai Euclidean distance paling kecil.Sistem yang dibangun hanya menerapkan solusi 1.

Dari gambar 3-1 dapat di simpulkan GSVM memiliki akurasi lebih baikdari pada linear SVM. Pengujian dilakukan dengan menggunakan nilai SVM parameter yang sama dan jumlah cross validation yang sama. GSVM-DC memiliki akurasi rata – rata 97,15% sedangkan SVM hanya memiliki akurasi rata – rata 96.53%.

Sedangkan Berdasarkan gambar 3-2 dapat di simpulkan bahwa *linear* SVM memiliki waktu *training* lebih cepat di bandingkan dengan waktu

training GSVM-DC. GSVM-DC memiliki waktu *training* yang lebih lama karena *training* akan dilakukan sebanyak jumlah *information granule* yang dibangun, sedangkan *linear* SVM hanya melakukan *training* sekali dengan menggunakan parameter SVM yang terpilih pada GSVM-DC.

Dengan hasil ini membuktikan bahwa modifikasi SVM dengan menggunakan paradigma *granular computing* dan metode DC dapat memberikan akurasi pengujian yang lebih baik dari linear SVM dalam melakukan klasifikasi *e-mail spam filtering*.

4. KESIMPULAN DAN SARAN

4.1 Kesimpulan

1. Modifikasi SVM menggunakan paradigma *granular computing*, dimana proses granulasi menggunakan *fuzzy c-means clustering*, dan metode *data cleaning* mampu meningkatkan akurasi pengujian sebesar 0,8% - 1% di bandingkan dengan *linear* SVM untuk *dataset* PKDD 2006.
2. Dataset PKDD 2006, menghasilkan pengujian terbaik dengan parameter SVM(*C*) = 0.718852837745963, jumlah granule = 4, *fuzzyness* parameter = 2, *epsilon*(ε) = 0.1 dengan waktu *training* 36.95 detik dan akurasi 97.2%.
3. Pada proses granulasi dengan menggunakan GSVM-DC, ditemukan beberapa kondisi yang menyebabkan pelatihan tidak selesai antara lain:
    1. Ditemukan 1 atau lebih *cluster* yang tidak memiliki anggota.
    2. Ditemukan *cluster* yang hanya memiliki anggota dengan *class* yang sama.

    Kedua hal tersebut menyebabkan *information granule* yang terbentuk tidak dapat di lakukan *training* untuk mencari parameter SVM dan LSV terbaik untuk tiap – tiap *information granule*, karena untuk mendapatkannya digunakan *linear* SVM, dan SVM hanya bisa di latih jika dalam suatu kumpulan data terdapat minimal 2 *class* yang berbeda.

PUSTAKA

Y.C. Tang, 2006, Granular Support Vector Machines Based on Granular Computing, Soft Computing and Statistical Learning, *A Dissertation Submitted in Partial Fulfillment of Requirements for the Degree of Doctor of Philosophy in the College of Arts and Sciences, Georgia State University*.

S. N. William, 2006, What is A Support Vector Machines?, Nature Publishing Group, *Nature Biotechnology* Volume 24 Number 12, pages 1565-1567.

B. Steve, 2003, Support Vector Machines, Department of Computer Science and Artificial Intelligence, University of Malta. In *Proceedings of the First Computer Science Annual Worksho*p (CSAW) 2003.

C. Nello, S. John, 2000, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, Robotica 18:6:687-689 Cambridge University Press.

G. Sudipto, M. Adam, M. Nina, M. Rajeev, O. Liadan, 2003, *Clustering Data Streams: Theory and Practice*, Radical Eye Software, pages 1-4.

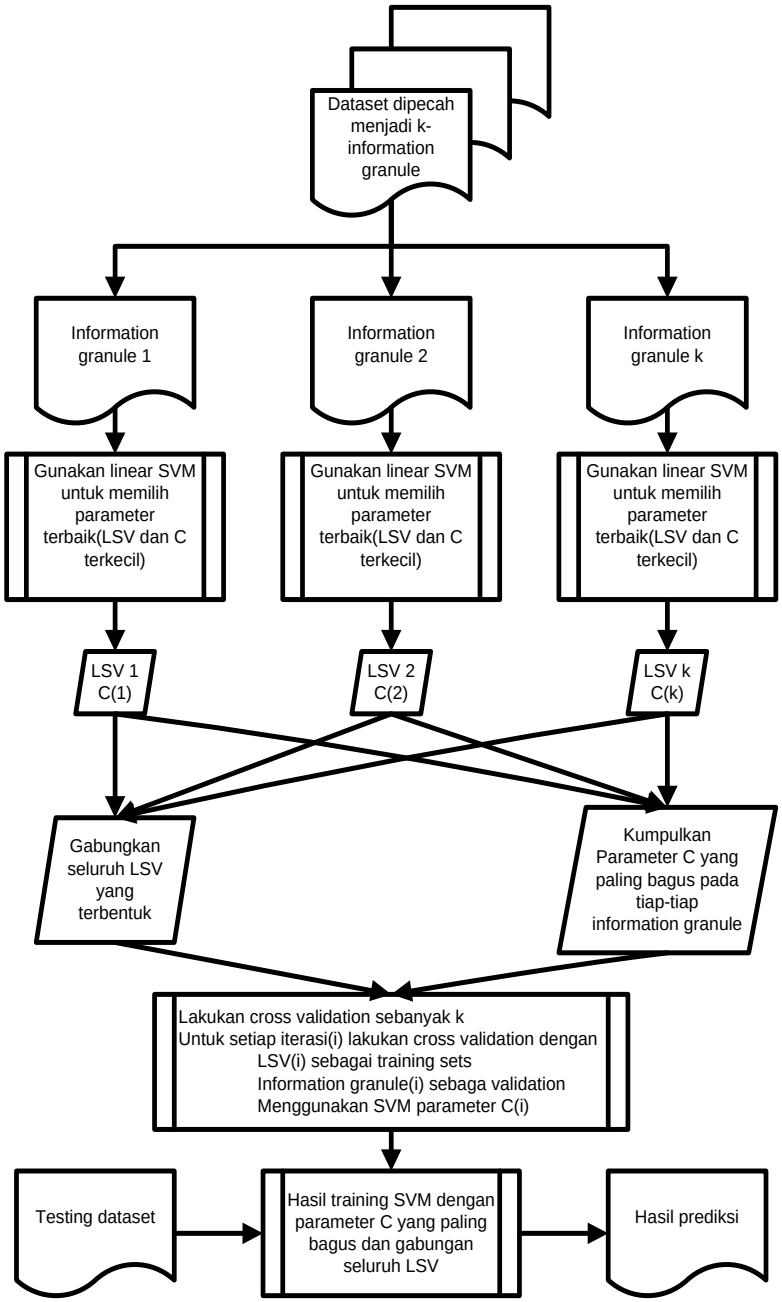
H. Eyke, 2008, *Granular Computation in Machine Learning and Data Mining*. In P. Witold, S. Andrzej, K Vladik, (editors), Handbook of Granular Computing, John Wiley & Sons, pages 889-907.

*Clustering: An Introduction***,** http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/index.html **,** diakses pada tanggal 20 oktober 2009.

H. Andreas, S. Gerd, 2003, *Conceptual Clustering of Text Clusters*, Institute of Applied Informatics and Formal Description Methods AIFB, University of Karlsruhe, http://www.aifb.unikarlsruhe.de/WBS/aho/pub/tc_fca_2002_submit.pdf

techterms@whatis.com, *what is spam filter?*, http://searchmidmarketsecurity.techtarget.com/sDefinition/0,,sid198_gci931766,00.html , Midmarket IT security definition, diakses pada tanggal 20 oktober 2009.

LAMPIRAN



Gambar 1. Bagan GSVM-DC