Terbit online pada laman web jurnal : **http://jurnal.iaii.or.id**

JURNAL RESTI

(Rekayasa Sistem dan Teknologi Informasi)

Vol. 1 No. 1 (2017) 19 – 25 | ISSN Media Elektronik : 2580-0760

Penerapan Support Vector Machine (SVM) untuk Pengkategorian Penelitian

Fithri Selva Jumeilah

Sistem Informasi, STMIK GI MDP, fithri.selva@mdp.ac.id

Abstract

The preparation of the research should be categorized in order to facilitate the search for the needy. To categorize the research required a method for text mining, one of them with the implementation of Support Vector Machines (SVM). The data used to recognize the characteristics of each category requires a collection of abstracts of research. The data will be preprocessing with several stages of case folding, stop words removing, tokenizing, and stemming. Further data that has undergone preprocessing will be converted into numerical form with for the term weighting stage. The results of term weighting then obtained data that can be used for data training and test data. The training process is done by providing input in the form of text data known category. Then by using Support Vector Machines, the input data is transformed into a knowledge model that can be used in the prediction process. From the research result, it is found that the categorization of research produced by SVM has been very good. This is evidenced by the results of the test that yields an accuracy of 90%.

Keywords: SVM, Text Mining, Preprocessing, Classification, Term Weighting

Abstrak

Penyusunan penelitian hendaknya harus perkategori agar mempermudah pencarian bagi yang membutuhkan. Untuk mengkategorikan penelitian dibutuhkan sebuah metode untuk penambangan teks, salah satunya dengan implementasi *Support Vector Machines* (SVM). Data yang digunakan untuk mengenali ciri dari tiap kategori maka dibutuhkan kumpulan dari abstrak penelitian. Data tersebut akan dilakukan *preprocessing* dengan beberapa tahapan yaitu *case folding*, *stopwords removing, tokenizing*, dan *stemming*. Selanjutnya data yang sudah mengalami *preprocessing* akan diubah menjadi bentuk numerik dengan untuk tahap *term weighting*. Hasil *term weighting* maka diperoleh data yang bisa digunakan untuk data *training* dan data uji. Proses *training* dilakukan dengan memberikan masukan berupa data teks yang diketahui kategorinya. Kemudian dengan menggunakan *Support Vector Machines*, data hasil masukan tersebut ditransformasikan ke dalam suatu model pengetahuan yang nantinya dapat digunakan dalam proses prediksi. Dari hasil penelitian diperoleh bahwa pengkategorian penelitian yang dihasilkan oleh SVM sudah sangat baik. Hal ini dibuktikan oleh hasil pengujian yang menghasilkan tingkat akurasi 90%.

Kata kunci: SVM, Penambangan Teks, *Preprocessing*, Klasifikasi, *Term Weighting*

1. Pendahuluan

Saat ini sudah banyak sekali kebijakan pemerintah yang ditujukan untuk meningkatkan penelitian. Semakin banyaknya penelitian maka semakin sulitnya penyimpanan penelitian. Kesulitan tersebut akan muncul ketika ada peneliti lain yang membutuhkan penelitian yang mendukung penelitiannya. Oleh sebab itu, sebaiknya penyimpanan penelitian dilakukan berdasarkan kategorinya. Pengkategorian penelitian dapat dilakukan secara manual dan otomatis. Untuk cara manual tentu akan membutuhkan waktu yang lebih banyak dibandingkan otomatis. Pengkategorian secara otomatis, dapat dilakukan dengan memanfaatkan *classifier* yang mampu memisahkan setiap dokumen sesuai dengan kategorinya.

Salah satu metode *classifier* yang dapat digunakan pada kategorisasi teks biasanya diadopsi dari *traditional machine learning* seperti *find similar, decision tree, Naive Bayes, Bayes Networks, Support Vector Machines (SVM)*, dan lain-lain. Diantara beberapa *classifier* tersebut SVM adalah salah satu *classifier* yang menghasilkan solusi paling baik dengan tingkat akurasi paling tinggi dibandingkan *classifier* lain, yaitu 92% untuk 10 kategori, sedangkan dengan jumlah

kategori yang sama, tingkat akurasi untuk masing-masing *classifier* yang dibandingkan yaitu: *find similar 64.6%, decision tree 88.4%, Naive Bayes 81.5%, Bayes Nets 85%* [1].

SVM telah berhasil diterapkan pada banyak kasus kategorisasi dengan akurasi tingkat tinggi. Keberhasilan tersebut meliputi: klasifikasi aroma multikelas dengan *data training* lebih dari 20% mampu melakukan pengkategorian dengan tingkat akurasi mencapai 100% [2]; selanjutnya dalam penelitiannya *web text categorization*, tingkat akurasinya antara 83.33% [3]. Secara sederhana SVM memiliki konsep mencari *hyperplane* "terbaik" yang berfungsi sebagai batas dari dua buah *class* [4]. SVM mencari ini *hyperplane* berdasarkan *support vector* dan *margin*. *Support vector* adalah titik–titik yang paling dekat dengan *separating hyperplane* sedangkan *margin* menyatakan lebar dari *separating hyperplane*.

Berdasarkan penelitian-penelitian sebelumnya, kemampuan SVM dalam mengkategorikan teks memiliki tingkat akurasi yang cukup baik. Oleh karena itu, dalam penelitian ini akan menggunakan SVM untuk pengategorian penelitian.

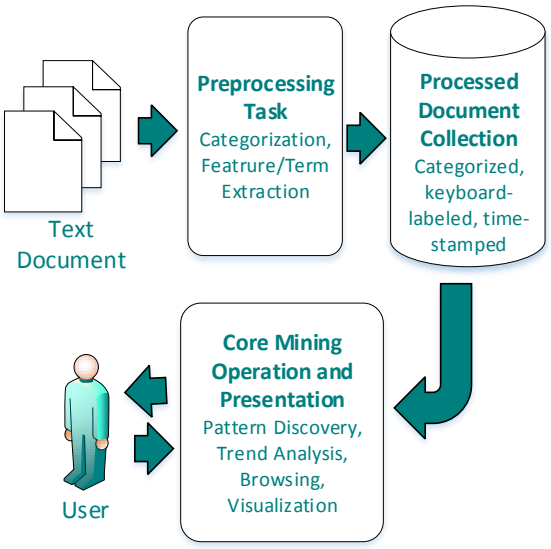
2. Tinjauan Pustaka

2.1 Text mining

Text mining adalah ilmu yang mempelajari bagaimana menarik informasi yang menarik, sesuatu yang baru, pola yang belum diketahui sebelumnya atau menemukan kembali informasi tersirat yang berasal dari kumpulan sumber-sumber data teks yang berbeda-beda [5]. *Text mining* mengekstrak informasi atau pola yang berguna dari sumber data teks melalui identifikasi dan eksplorasi dari suatu pola menarik [6]. Sumber data pada *text mining*, berupa sekumpulan dokumen. Selain itu juga, data berbentuk pola menarik yang tidak ditemukan pada *database record*, tetapi dalam teks yang tidak terstruktur.

Text mining memililki perbedaan dengan *data mining*. Perbedaan antara *text mining* dan *data mining* terletak pada sumber datanya, dimana *text mining* menggunakan sumber data yang berasal dari kumpulan dokumen atau teks yang umumnya berbentuk *unstructured text*. *Text mining* mencoba untuk mencari hubungan satu bagian teks dengan yang lainnya berdasarkan aturan-aturan tertentu.

Pada tingkat fungsional, sistem *text mining* mengikuti model umum yang diberikan oleh beberapa aplikasi *data mining* klasik, dam terdapat 4 bidang utama pada tingkat fungsional sistem *text mining*, yaitu *preprocessing tasks, core mining operations, presentation layer components and browsing functionality*, dan *refinement techniques* [6]. Arsitektur umum *text mining* terlihat pada Gambar 1 [6].

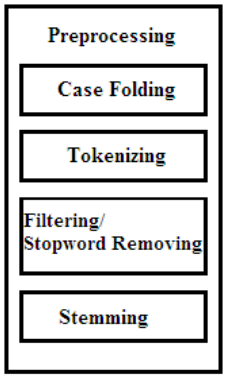


Gambar 1. Arsitektur umum text mining

2.2 Preprocessing

Dokumen pada umumnya mempunyai struktur yang sembarangan atau tidak terstruktur. Oleh karena itu, diperlukan suatu proses yang dapat mengubah bentuk data yang sebelumnya tidak terstruktur ke dalam bentuk data yang terstruktur. Proses pengubahan ini dikenal dengan istilah *text preprocessing* [6].

Dokumen mengandung beragam variasi dari bentuk huruf sampai tanda baca. Variasi huruf harus diseragamkan yaitu dengan menjadikan huruf besar saja atau huruf kecil saja. Selain itu, proses penghilangan tanda baca dilakukan untuk menghilangkan *noise* pada saat pengambilan informasi.



Gambar 2. Tahap *Preprocessing*

Proses *preprocessing* dilakukan agar data yang digunakan bersih dari *noise*, memiliki dimensi yang lebih kecil, serta lebih terstruktur, sehingga dapat diolah lebih lanjut. Tahap *preprocessing* memiliki beberapa proses, yaitu *case folding, stopwords removing, tokenizing*, dan *stemming* yang dapat dilihat pada Gambar 2 [7].

Text preprocessing yang pertama kali dilakukan adalah *case folding*. *Case folding* merupakan proses dalam *text preprocessing* yang dilakukan untuk menyeragamkan karakter pada data. Proses *case folding* adalah proses mengubah seluruh huruf menjadi huruf kecil. Pada proses ini karakter-karakter 'A'-'Z' yang terdapat pada data diubah kedalam karakter 'a'-'z'. Karakter-karakter selain huruf 'a' sampai 'z' (tanda baca dan angka-angka) akan dihilangkan dari data dan dianggap sebagai *delimeter* [8]. *Delimeter* adalah urutan satu atau lebih karakter yang digunakan untuk menentukan batas pemisah.

Sebelum data/teks dapat diproses lebih lanjut, maka data tersebut harus disegmentasi menjadi kata-kata, proses ini disebut *tokenizing*. Tahap *tokenizing* adalah tahap pemotongan *string* masukan berdasarkan kata-kata yang menyusunnya atau dengan kata lain pemecahan kalimat menjadi kata. Strategi umum yang dilakukan pada tahap *tokenizing* adalah memotong kata pada *white space* atau spasi dan membuang karakter tanda baca. Tahap *tokenizing* membagi urutan karakter menjadi kalimat dan kalimat menjadi *token*.

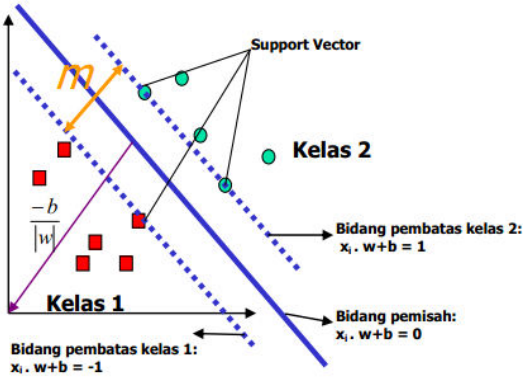
Setelah tahap *tokenizing*, maka dilakukan tahap *filtering* yaitu dengan menghapus kata-kata yang sangat umum [9]. Kata yang termasuk dalam *stopword* contohnya adalah yang, dan, di, itu, dengan, untuk, tidak, dari, dalam, akan, pada, ini, juga, saya, serta, adalah, bahwa, lain, kamu, dan lain lain. *Stemming* merupakan tahapan pada *text preprocessing* yang bertujuan untuk mengubah *term* ke bentuk akar katanya. *Stem* (akar kata) adalah bagian dari kata yang tersisa setelah dihilangkan imbuhannya (awalan dan akhiran).

2.2 Support Vector Machine

Support Vector Machine adalah sistem pembelajaran yang menggunakan ruang hipotesis berupa fungsi-fungsi linier dalam sebuah ruang fitur *(feature space)* yang berdimensi tinggi dan mengimplementasikan *learning* bias yang berasal dari teori pembelajaran statistik yang dilatih dengan algoritma pembelajaran [10,11]. Teori yang mendasari SVM telah berkembang sejak tahun 1960-an, tetapi baru diperkenalkan oleh Vapnik, Boser dan Guyon pada tahun 1992.

Secara sederhana konsep SVM adalah usaha mencari *hyperplane* "terbaik" yang berperan penting sebagai garis batas dua buah *class* [4]. SVM mencari *hyperplane* ini berdasarkan *support vectors* dan *margin*. *Support vectors* adalah seluruh vektor data yang berjarak paling mendekati *hyperplane*, sedangkan *margin* menyatakan lebar dari *separating hyperplane*.

Linearly separable data merupakan data yang dapat dipisahkan secara *linier*. Misalkan {x1,...,xn} adalah *dataset* dan y i ∈{+1,−1} adalah label kelas dari data xi, label +1 menandakan bahwa data tersebut diklasifikasikan sebagai kelas +1 dan label -1 menandakan sebaliknya. Tujuan dari SVM adalah menghasilkan sebuah model klasifikasi berupa fungsi sign(x), f(x)= y, agar dapat mengklasifikasikan data pada proses *testing*.



Gambar 3. Alternatif bidang pemisah terbaik

Dalam Gambar 3 dua kelas dapat dipisahkan oleh sepasang bidang pembatas *(hyperplane)* yang sejajar. Bidang pembatas pertama menjadi batas kelas pertama sedangkan bidang pembatas kedua adalah batas dari kelas kedua, sehingga diperoleh Persamaan 1.

$$xi\ .w + b \geq +1 \quad foryi = +1$$

$$xi\ .w + b \leq -1 \quad foryi = -1 \qquad (1)$$

Keterangan:
w : Normal bidang
b : Posisi bidang relatif terhadap pusat koordinat

Secara umum, cara kerja dari SVM adalah menemukan jarak terjauh dari *hyperplane* dengan kedua kelas. Proses penentuan jarak terjauh dilakukan berulang kali hingga menemukan *hyperplane* terbaik. Untuk itulah diperlukan optimasi pada SVM untuk menemukan jarak maksimum *hyperplane* dengan kedua kelas tersebut. Dalam pembangunan SVM, terdapat dua bentuk optimasi yang digunakan untuk menemukan *hyperplane*. Bentuk optimasi pertama yaitu *Primal Form* SVM dan yang kedua adalah *Dual Form* SVM. *Primal Form* tidak dapat digunakan dalam penelitian ini karena tidak akan pernah memenuhi konstrain.

Dual Form SVM, dibangun dengan menggunakan pendekatan *Lagrange*. Bentuk *Dual Form* merupakan perubahan dari bentuk *Primal Form* yang dimodifikasi menggunakan *Lagrange* sehingga pencarian *hyperplane* dapat dilakukan. Persamaan *Lagrange* dibangun dengan menggunakan konstanta α yang untuk selanjutnya konstanta tersebut yang digunakan sebagai penanda *support vectors*. Persoalan ini akan lebih mudah diselesaikan jika ditranformasi ke dalam persamaan *Lagrange* yang menggunakan *lagrange multiplier*. Dengan demikian permasalahan optimasi konstrain dapat diubah menjadi:

$$\max_{\alpha} L_D = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i x_j \quad (2)$$

$$\text{Subject to. } \sum_{i=1}^{n} \alpha_i y_i = 0, \alpha_i \geq 0$$

Nilai α seperti yang telah dijelaskan sebelumnya, merupakan penanda apakah data tersebut merupakan *support vectors* atau tidak yang ditandai dengan nilai $\alpha \geq 0$ merupakan *support vectors*, sedangkan sisanya memiliki nilai $\alpha_i=0$. Dengan demikian, dapat diperoleh nilai α_i yang nantinya digunakan untuk menemukan w menggunakan Persamaan (3):

$$\frac{\partial}{\partial w} L_p(w, b, \alpha) = 0 \quad \rightarrow w = \sum_{i=1}^{n} \alpha_i y_i x_i \quad (3)$$

Keterangan:
w : Normal bidang
b : Posisi bidang relatif terhadap pusat koordinat

Formula pencarian bidang yang pembatas terbaik pada Persamaan (2) adalah *quadratic programming*, sehingga nilai maksimum global dari α_i selalu dapat ditemukan. Sedangkan untuk mencari nilai bias (b) digunakan Persamaan (4):

$$b = \frac{1}{\#SV} \sum_{xi \in SV} \left(\frac{1}{yi} \right) - \sum_{xj \in SV} \alpha_j y_j k(x_j x_i)) \quad (4)$$

Keterangan:

#SV : Jumlah *Support Vector*

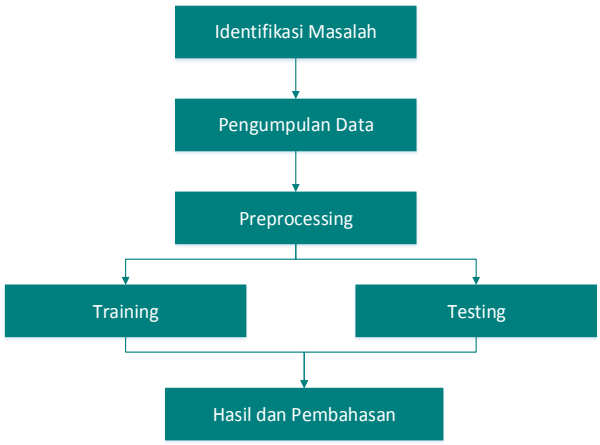
Setelah solusi permasalahan *quadratic programming* (nilai α_i) dan nilai bias (b) ditemukan. Maka kelas dari data pengujian x dapat ditentukan berdasarkan nilai dari fungsi keputusan:

$$f(x_d) = \sum_{i=1}^{ns} \alpha_i y_i x_i x_d + b \quad (5)$$

Keterangan:
ns : Jumlah *Support Vector*
xd : Data yang akan diklasifikasikan

3. Metodologi Penelitian

Untuk melaksanakan penelitian ini, terdapat beberapa langkah yaitu identifikasi masalah, pengumpulan data, *preprocessing, testing, training* dan hasil dan pembahasan. Langkah-langkah tersebut dapat dilihat pada Gambar 4.



Gambar 4. Langkah-langkah Penelitian

3.1 Identifikasi Masalah

Penelitian semakin hari akan semakin bertambah dan akan semakin menyulitkan pengorganisasian filenya dan akan menyulitkan pencarian datanya. Di berbagai perguruan tinggi dibutuhkan pengkategorian penelitian untuk memudahkan para peneliti lain mencari penelitian yang terkait dan memudahkan pekerjaan dari bagian perpustakaan.

3.2 Pengumpulan Data

Untuk mengkategorikan penelitian dibutuhkan banyak penelitian yang membahas tentang algoritma *text mining*. Selain itu, juga dibutuhkan kumpulan abstrak penelitian yang akan digunaan sebagai *data testing* dan *data training*.

3.3 Preprocessing

Dokumen pada umumnya mempunyai struktur yang sembarangan atau tidak terstruktur. Oleh karena itu, diperlukan suatu proses yang dapat mengubah bentuk data yang sebelumnya tidak terstruktur ke dalam bentuk data yang terstruktur. Tahap *preprocessing* memiliki beberapa proses, yaitu *case folding, stopwords removing, tokenizing,* dan *stemming*. Selanjutnya data yang sudah mengalami *preprocessing* akan diubah menjadi bentuk numerik dengan tahap *term weighting*. Pada penelitian ini terdapat tiga metode *term weighting* yang digunakan, yaitu *term frequency, inverse document frequency*,dan *term frequency-inverse document frequency*.

Langkah pertama yang digunakan adalah mencari tf. Metode tf melakukan pembobotan dengan menghitung frekuensi kemunculan *term*. Sebuah kata yang sering muncul pada suatu dokumen teks, maka bobot kata tersebut semakin besar dan kata tersebut dianggap sebagai kata yang sangat merepresentasikan dokumen teks tersebut dan untuk mencari tf dapat dilihat dalam Persamaan 6.

$$ig(c,t) = \left(\frac{a}{N} * \log\frac{a*N}{(a+c)*(a+b)}\right) + \left(\frac{b}{N} * \log\frac{b*N}{(b+d)*(a+b)}\right) + \left(\frac{c}{N} * \log\frac{c*N}{(a+c)*(c+d)}\right) + \left(\frac{d}{N} * \log\frac{d*N}{(b+d)*(c+d)}\right) \quad (6)$$

Keterangan :

a adalah jumlah dokumen pada *positive category* c_j yang mengandung term t_i.
b adalah jumlah dokumen pada *positive category* c_j yang tidak mengandung term t_i.
c adalah jumlah dokumen pada *negative category* $\overline{c_j}$ yang mengandung term t_i .
d adalah jumlah dokumen pada *negative category* $\overline{c_j}$ yang tidak mengandung term t_i.

$$N = a + b + c + d$$

.

Setelah itu mencari IDF, metode ini memperhatikan kemunculan *term* pada kumpulan dokumen. Pada metode ini, *term* yang dianggap penting adalah *term* yang paling sedikit kemunculannya pada sumber dokumen. Selain itu, tingkat kepentingan nilai (bobot) dari suatu *term* juga diasumsikan berbanding terbalik dengan jumlah dokumen yang memiliki *term* tersebut. Untuk mencari IDF dapat dilihat dalam Persamaan 7.

$$W(d,t) = TF(d,t) \quad (7)$$

Langkah terakhir menggunakan metode TF IDF, metode ini gabungan antara metode yang sebelumnya dan merupakan hasil perkalian antara keduanya, dapat dilihat pada Persamaan 8. Pada metode ini, nilai bobot yang tinggi akan diberikan kepada *term* yang sering muncul pada suatu dokumen, tetapi jarang muncul pada kumpulan dokumen. Selanjutnya hasil yang telah didapatkan dari proses *termweighting* akan disimpan kedalam txt.trainset.

$$IDF(t) = \log\left(\frac{N}{df(t)}\right) \quad (8)$$

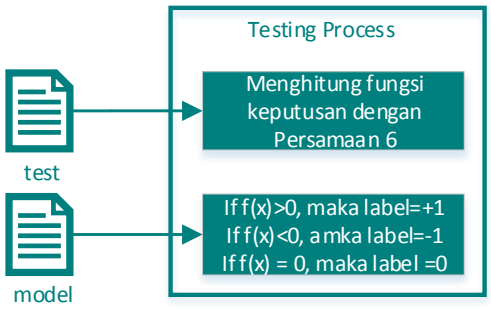
Tahapan-tahapan tersebut dilakukan secara bertahap, sehingga data yang dipakai menjadi data yang berkualitas tinggi. Data dikatakan berkualitas tinggi, apabila bersih dari *noise*, terstruktur dan berdimensi kecil.

3.4 Training dan Testing

Proses pelatihan ini diawali dengan masukan berupa teks yang telah melalui *preprocessing* terlebih dahulu. Proses pelatihan pada *Support Vector Machine* bertujuan untuk mencari nilai alpha *(support vector)* dan bias pada fungsi tujuan dari semua data sampel *training*, seperti yang terlihat pada Persamaan 2 dan Persamaan 4. Di ruang *vector, support vector* adalah kumpulan data dari kedua kategori yang posisinya paling mendekati *separating hyperplane*. Nilai alpha yang dimaksud adalah nilai yang lebih besar sama dengan 0 dan semakin besar nilai alpha maka semakin bagus *hyperplane* yang terbentuk. Setelah didapatkan nilai alpha dan bias, maka akan dicari nilai w dengan Persamaan 3. Hasil dari proses pelatihan ini akan disimpan dalam file "model" dan akan digunakan untuk proses pengujian (Gambar 5).

Gambar 5. Arsitektur Pelatihan

Proses pengujian dijalankan dengan dua macam masukan, yaitu data pengujian dan model yang didapatkan dari proses pelatihan. Pada proses pengujian, data masukan dilakukan *preprocessing* terlebih dahulu sama seperti saat proses pelatihan. Setelah itu proses pengkategorian teks dilakukan dengan menggunakan Persamaan 6. Selanjutnya akan dilihat hasil tesebut masuk dalam kelas +1 atau -1. Hasil proses pengkategorian adalah berupa label. Label tersebut menandakan kategori yang telah ditentukan sebelumnya. Sehingga data uji tersebut dapat dikategorikan ke dalam kelas tertentu. Ilustrasi Pengujian dapat dilihat pada Gambar 6.



Gambar 6. Arsitektur Pengujian

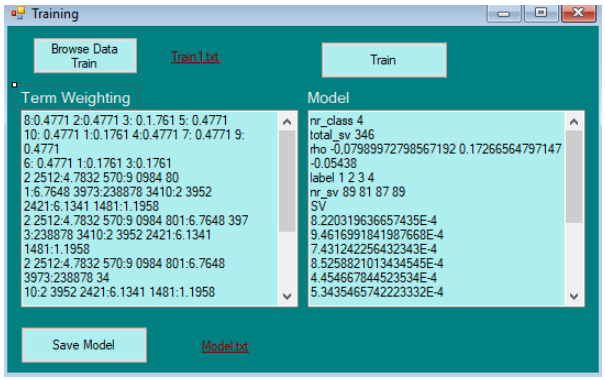
3.6 Hasil dan Pembahasan

Untuk menguji SVM maka akan dilakukan perbandingan hasil kategori SVM dengan kategori yang sebenarnya dan akan dihitung nilai akurasinya.

4. Hasil dan Pembahasan

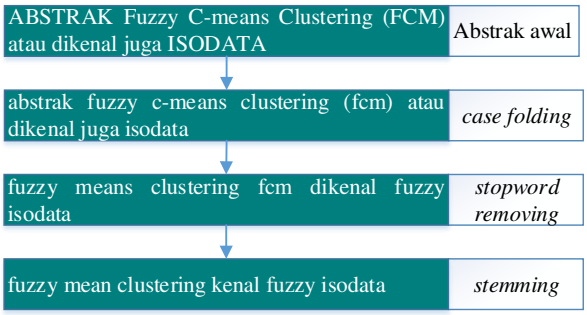
Hasil dari penelitian ini adalah berupa sebuah aplikasi yang memiliki 2 halaman utam yaitu halaman *training* dan *testing*. Untuk halam training dapat dilihat pada Gambar 7. Sebelum melakukan pengkategorian

pengguna harus menginput data *traning* terlebih dahulu yang berupa file txt dengan menekan tombol "*Browse Data Train*". File tersebut berisikan kumpulan data abstrak penelitian yang berjumlah 390 abstrak dengan 4 macam kategori. Terdapat empat kategori yang mempunyai label 1, 2, 3, dan 4 yaitu label 1 untuk Kecerdasan Buatan dan *Natural Language Processing*, label 2 untuk Pengolahan Citra dan Pengenalan Pola, label 3 untuk Sistem Informasi, dan label 4 untuk Sistem Keamanan dan Jaringan Komputer. Data abstrak penelitian yang digunakan yaitu berasal dari tugas akhir mahasiswa yang ada pada fakultas ilmu komputer dari *digital library* universitas-universitas di Indonesia. Setelah memilih data training maka aplikasi akan langsung melakukan *preprocessing*.



Gambar 7. Halaman Antar Muka *Training*

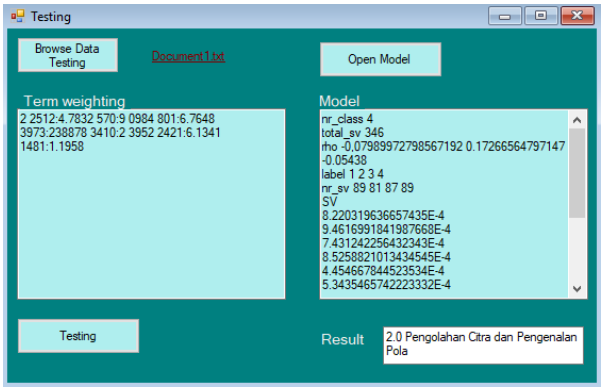
Dokumen pada umumnya mempunyai struktur yang sembarangan atau tidak terstruktur. Oleh karena itu, diperlukan suatu proses yang dapat mengubah bentuk data yang sebelumnya tidak terstruktur ke dalam bentuk data yang terstruktur. Tahapan pertama yang dilakukan pada tahap *preprocessing* adalah proses *case folding*. Proses ini mengubah data menjadi huruf kecil semua dan membuang *delimeter*. Apabila karakter dalam dokumen sudah seragam, maka akan dilakukan tahap selanjutnya yaitu *filtering* atau *stopword removing*. Semua kata yang dianggap tidak memiliki kontribusi atau yang terdapat dalam *stoplist* akan dihilangkan.



Gambar 8. Hasil Tahapan *Preprocessing*

Tahap selanjutnya pada *preprocessing* adalah *stemming*. Proses ini bertujuan untuk mendapatkan bentuk akar dari setiap kata dengan menghilangkan awalan atau akhiran. Algoritma *stemming* yang digunakan adalah algoritma Nazief dan Adriani. Pada Gambar 8 dapat dilihat contoh hasil dari setiap tahap *preprocessing*. Jika dokumen sudah terstruktur, maka kata-kata yang ada akan dicari nilai-nilai atau diberikan bobot untuk masing-masing kata yang ada dalam dokumen. Tahap ini dinamakan tahap *term weighting* dapat diartikan sebagai proses memberikan nilai atau bobot ke sebuah *term* berdasarkan kemunculannya pada suatu dokumen teks dan akan menampilkan pada *textbox Tem weighting* dan tombol Train akan aktif. Jika tombol Train telah aktif maka pengguna bisa menekannya untuk melakukan *training* dan mendapatkan modelnya.

Untuk melakukan data testing akan bisa dilakukan jika model dari hasil data *training* sudah ada. Seperti yang terlihat pada Gambar 9 pengguna bisa memilih data yang ingin diuji dan model yang ingin digunakan. Seperti seperti tahapan *training*, data *testing* juga akan melalui preprocessing secara otomatis dan hasilnya akan mucul pada *text box term weighting*. Setelah teks berhasil dimasukkan lalu tombol *testing* ditekan, maka sistem akan mulai melakukan proses pengkategorian penelitian dan menampilkan status hasil kategori di result.



Gambar 9. Halaman Antar Muka *Trainning*

Nilai akurasi dari suatu aturan, fungsi, ataupun model pengetahuan dapat diketahui dengan melihat perbandingan antara hasil prediksi yang benar dengan jumlah seluruh data yang diuji. Pengujian dilakukan dengan membandingkan jumlah data abstrak penelitian yang berhasil diuji dan jumlah data abstrak penelitian keseluruhan yang diuji. Maka, akurasi diperoleh dari Persamaan 9:

$$Akurasi = \frac{jumlah\ data\ teks\ yang\ berhasil\ diuji}{jumlah\ data\ teks\ keseluruhan\ yang\ diuji} x100\% \quad (9)$$

Jumlah data abstrak yang berhasil dikategorikan dengan benar berjumlah 36 data dari jumlah data

abstrak keseluruhan yang diuji yaitu 40 data. Dari hasil pengujian, persentasi keakuratan dapat dihitung degan membandingkan jumlah data yang berhasil diuji dengan jumlah data uji (Persamaan 9).

$$Akurasi = \frac{36}{40} x100\% = 90\%$$

Dari hasil perhitungan nilai akurasi maka didapat nilai akurasi mencapai 90%. Pada beberapa kasus, terdapat beberapa data abstrak penelitian yang tidak dapat dikenali. Data abstrak penelitian gagal dikategorikan karena memiliki pola kata yang hampir sama dengan kategori yang lain sehingga bersifat ambigu. Ada beberapa data yang memiliki pola yang jelas namun gagal dikategorikan sesuai dengan kategorinya. Hal ini dikarenakan contoh abstrak penelitian yang terdapat pada training set mungkin belum banyak memiliki variasi seperti yang terdapat pada pengujian set.

5. Kesimpulan

5.1 Simpulan

Kesimpulan dari penelitian ini adalah:

1. Algoritma *Support Vector Machine* dapat diimplementasikan dalam pembangunan Sistem Kategorisasi Topik penelitian.
2. Sistem menghasilkan tingkat akurasi sebesar 90% dari 40 data pengujian untuk 4 kategori.

5.2 Saran

Saran untuk pengembangan lebih lanjut dari penelitian ini diharapkan penelitian ini mampu diimplementasikan pada beberapa perguruan tinggi atau dilakukan pengembangan terhadap masalah pengkategorian penelitian dengan pertimbangan yang lebih komplek dengan metode yang lain.

6. Daftar Rujukan

[1] Dumais, et al. (1998), “Inductive Learning Algorithms and Representations for Text Categorization”.
[2] Rustam, et al. (2003), Klasifikasi Aroma Menggunakan SVM. Seminal Nasional Ilmu Komputer dan Teknologi Informasi, Vol 4, pp. 231-235.
[3] Hao Y, Lee Y, Harmer S et al. (2007) . Measurement of Complex Permittivity of Textile Materials for Body- Centric Wireless Communications. Conference: iET Seminar on Antennas and Propagation for Body-Centric Wireless Communications, London, April 2007.
[4] Yang, Y. and Liu, 1999, “Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval “, (SIGIR'99, pp 42--49).
[5] Even, Yair dan Zohar. 2002. Itroduction to Text Mining. University Of Illionis. Illionis.
[6] Feldman, R. & Sanger, J. 2007. The Text Mining Handbook-Advanced Approaches in Analyzing Unstructured Data, USA: New York.
[7] Langgeni, Baizal dan Firdaus, 2010, Clustering Artikel Berita Berbahasa Indonesia Menggunakan Unsupervised Feature Selection, Seminar Nasional Informatika, Yogyakarta.
[8] Raghavan dan Schutze. 2009. Introduction to Information Retrieval, Cambridge University Press.
[9] Benbrahim dan Bramer, 2009. Text and Hypertext Categorization.
[10] Joachims, T. 1998. TextCotegorization with Support Vector Machines : Learning with Many Relevant Features. University Dortmund. Germany.
[11] Yang and Joachims (2008) Text categorization. 3(5):4242