Efek Penggunaan Keterkaitan Kata pada Algoritma Similaritas Semantik Terhadap Kinerja Proses Klasifikasi Teks dengan K-Nearest Neighbour

Husni Thamrin[1)], Atiqa Sabardila[2)]
1) Teknik Informatika, 2) Pendidikan Bahasa Indonesia UMS
E-mail : Husni.Thamrin@ums.ac.id

ABSTRAK

Klasifikasi teks merupakan proses untuk mengelompokkan dokumen teks ke kelas-kelas yang telah ada. Metode k-nearest neighbour dapat digunakan dalam proses klasifikasi teks yang mengandalkan hasil perhitungan similaritas semantik untuk menentukan skor jarak/kedekatan antar dokumen teks. Perhitungan similaritas dua dokumen tidak hanya dipengaruhi oleh kesamaan kata-kata yang terkandung dalam dokumen, namun dipengaruhi juga oleh faktor keterkaitan kata di antara kedua dokumen. Tulisan ini membandingkan kinerja proses klasifikasi yang menerapkan fungsi kosinus tanpa memperhitungkan keterkaitan kata dan fungsi Dice yang memperhitungkan keterkaitan kata dengan Google bi-gram. Metode klasifikasi yang diuji adalah k-nearest neighbour. Hasil pengamatan menunjukkan bahwa penambahan faktor Google bi-gram pada fungsi Dice meningkatkan skor similaritas dua dokumen dan meningkatkan kinerja proses klasifikasi. Algoritma tanpa penambahan keterkaitan kata menghasilkan nilai F-Measure sebesar 0.648, sedangkan dengan penambahan faktor keterkaitan kata diperoleh F-Measuer sebesar 0.759.

Kata kunci: keterkaitan kata, n-gram, similaritas semantik, klasifikasi teks

A. PENDAHULUAN

Klasifikasi adalah proses untuk menentukan label atau kategori objek. Sekumpulan objek yang mempunyai kesamaan fitur biasanya dikategorikan ke dalam kelompok tertentu dan kelompok itu kemudian diberi nama atau label. Proses klasifikasi dibagi menjadi dua yaitu klasifikasi yang terpandu dan tidak terpandu (*supervised and unsupervised classification*). Klasifikasi yang terpandu mempunyai tahapan pembelajaran (*learning phase*) sebelum memasuki tahapan penerapan atau pengujian (*testing phase*). Klasifikasi yang tidak terpandu merupakan metode pengelompokan objek tanpa didahului tahap belajar (Sentosa, 2007).

Salah satu bentuk klasifikasi diterapkan terhadap objek berupa teks atau dokumen. Klasifikasi teks banyak diterapkan misalnya pada pengelompokan email ke dalam kategori spam atau tidak spam, pengelompokan berita ke dalam topik tertentu misalnya olahraga atau politik, dan penentuan opini masyarakat terhadap sebuah isu apakah positif atau negatif (Manning, dkk. 2009).

Banyak metode dapat diterapkan untuk melakukan klasifikasi teks dan dokumen. Metode yang populer antara lain adalah pohon keputusan (*decision tree*), klasifikasi berbasis aturan (*rule-based classifier*), *support vector machine (SVM)*, jaringan syaraf tiruan (*artificial neural network*), tetangga terdekat (*K-nearest neighbour*), dan algoritma Bayesian (Agarwal & Zhai, 2012).

Metode klasifikasi teks mengolah dan mencermati kata-kata yang menyusun teks

untuk menentukan kelas dokumen. Berbagai metode mempunyai variasi dalam mengolah kata penyusun, mulai dari keberadaan kata (*binary*), jumlah kata (*frequency*), hingga makna kata. Penggunaan metode yang mencermati keberadaan atau jumlah kata mengandung potensi masalah karena kata-kata yang sama dapat memiliki makna yang berbeda dalam konteks yang berbeda. Kata-kata yang berbeda dapat pula mempunyai makna yang sama.

Salah satu metode klasifikasi teks menggunakan keterkaitan kata (*word relatedness*) dalam proses perhitungan similaritas dua teks. Penelitian yang dilakukan Yazdani dan Popescu Belis (2012) menghitung keterkaitan kata berdasarkan isi dan link sebuah ensiklopedia *hypertext*. Kedua peneliti berkesimpulan bahwa pemanfaatan keterkaitan menghasilkan proses yang sangat baik, meskipun masih lebih rendah dari yang terbaik. Untuk pengukuran similaritas dokumen, kedua peneliti mendapatkan nilai korelasi lebih dari 0,6 yang merupakan angka yang mendekati metode LSA (*latent semantic analysis*).

Penerapan keterkaitan kata dalam menghitung similaritas teks telah dikaji untuk berbagai aplikasi selain klasifikasi teks. Kajian terbaru dilakukan oleh Liu dan Wang (2013) yang memanfaatkan keterkaitan kata dalam jejaring (*WordNet*). Kedua peneliti berupaya memperbaiki kinerja fungsi similaritas kosinus dengan memberikan skor kedekatan relatif dalam jejaring kata. Dengan menyetel nilai sebuah parameter pada titik optimal tertentu, dihasilkan perbaikan pada nilai kinerja dibanding metode Lesk (Lesk, 1986), Leacock-Chodorow (Leacock & Chodorow, 1998) maupun metode Wu-Palmer (Wu & Palmer, 1994).

Thamrin dan Wantoro (2014) telah pula berupaya mencermati keterkaitan kata dalam leksikon bahasa Indonesia untuk melakukan penilaian otomatis (*automatic scoring*) dalam sebuah sistem manajemen belajar online (*learning management system*). Keduanya mendapati bahwa terdapat korelasi antara skor yang diberikan terhadap jawaban siswa dengan keterkaitan kata dalam konsep sinonim, hiponim, dan hipernim.

Islam, Milios dan Keselj (2012) menggunakan Google tri-grams untuk mencermati tingkat keterkaitan antar kata. Google n-gram merupakan basis data yang berisi kata atau kumpulan kata yang sering digunakan sebagai kata kunci dalam pencarian di situs pencari Google. Basis data itu mengandung informasi tentang frekuensi ditemukannya kata-kata dalam situs (*term hit*). Dasar idenya adalah bahwa kata-kata yang lebih sering muncul bersama-sama merupakan kata yang mempunyai hubungan atau keterkaitan yang lebih tinggi. Para peneliti mengklaim bahwa hasil perhitungan dengan algoritma ini mempunyai korelasi sangat tinggi yaitu lebih dari 0,9 dan mendekati skor penilaian pakar.

Tulisan ini mengamati pengaruh penggunaan keterkaitan kata (*word relatedness*) pada perhitungan similaritas dua teks. Perhitungan similaritas tersebut digunakan pada proses klasfikasi dengan metode *k-nearest neighbour*. Hasil pengamatan menunjukkan adanya perbaikan skor similaritas dokumen dan perbaikan kinerja proses klasifikasi.

B. Metode Penelitian

Penelitian dilakukan dengan melakukan klasifikasi terhadap sejumlah teks atau dokumen menggunakan metode *k-nearest neighbour* (k tetangga terdekat). Prinsipnya, sebuah objek digolongkan ke dalam kelas tertentu berdasarkan persentase jumlah objek

terdekat yang tergolong kelas yang dipilih. Pada penelitian ini, penentuan kedekatan objek dilakukan dengan dua fungsi similaritas yang berbeda.

Penentuan kelas sebuah teks X dengan *k-nearest neighbour* dilakukan dengan langkah-langkah sebagai berikut.

1. Langkah awal (*preprocessing*) dimulai dengan membaca teks dan menghilangkan karakter selain huruf, angka, tanda titik, tanda minus (-), dan spasi. Proses ini disebut juga *case folding*. Kemudian dilakukan proses *parsing* yaitu mengurai frase dan kalimat yang menjadi daftar kata-kata. Kata yang tergolong sebagai kata yang terlalu umum (*stop word*) diabaikan. Pada penelitian ini tidak dilakukan proses *stemming* atau proses penentuan kata dasar terhadap kata turunan.
2. Melakukan perhitungan jarak atau kemiripan teks X dengan seluruh teks lain yang ada. Pada penelitian ini, perhitungan yang dimaksud adalah kemiripan teks dengan menggunakan dua algoritma similaritas semantik. Algoritma pertama adalah fungsi similaritas kosinus (*cosine similarity function*) dan yang kedua adalah algoritma similaritas Islam-Melios-Keselj (IMK).
3. Menentukan sejumlah k teks terdekat (misalnya k = 10). Dari sejumlah k teks tersebut ditentukan satu kelas yang mengandung teks terbanyak dan teks X digolongkan ke dalam kelas tersebut. Sebagai contoh, dari 10 teks yang paling mirip dengan teks X terdapat 5 teks kelas A, 3 teks kelas B, dan 2 teks kelas C, maka teks X digolongkan ke dalam kelas A.

Fungsi similaritas kosinus jika diterapkan pada dokumen teks mempunyai bentuk persamaan matematis sebagai berikut.

$$\mathrm{Sim}(D_1, D_2) = \frac{\sum_{k=1}^{d} D_{1k} D_{2k}}{\sqrt{\sum_{k=1}^{d} D_{1k}^2 \sum_{k=1}^{d} D_{2k}^2}}$$

Pada persamaan tersebut, D_1 dan D_2 adalah dua dokumen teks, d adalah jumlah total *term* (atau kata) yang ada pada kedua teks, D_{ik} adalah jumlah *term* tertentu pada dokumen i. Sebagai contoh misalkan terdapat dua teks D_1 = "Ali membeli sepatu" dan D_2 = "Sepatu Ali baru". Maka term yang ada pada kedua teks adalah <ali, membeli, sepatu, baru>, artinya d = 4. Nilai D_{1k} = <1, 1, 1, 0> dan D_{2k} = <1, 0, 1, 1>. Similaritas kedua teks adalah 2/3 ~ 0,667.

Algoritma similaritas Islam-Melios-Keselj (IMK) tidak hanya memperhatikan jumlah kata yang sama pada dua teks namun memperhatikan tingkat keterkaitan kata-kata yang tidak sama. Pada contoh di atas, kata "membeli" dan kata "baru" merupakan dua kata yang tidak mempunyai pasangan pada dokumen lainnya. Namun kata "membeli" dan "baru" mungkin mempunyai keterkaitan. Pada algoritma IMK, keterkaitan dua kata ditentukan secara relatif dengan nilai Google bi-gram, yaitu frekuensi kedua kata yang ditemukan oleh mesin pencari Google. Nilai Google bi-gram tersedia dalam bentuk basis data yang dikeluarkan secara resmi oleh perusahaan tersebut. Contoh langkah penerapan algoritma untuk menghitung similaritas dua teks D_1 = "The boy got a pair of shoes from his mom" dan D_2 = "The baby got new shoes" adalah sebagai berikut.

1. Hitung jumlah kata pada kedua teks. Panjang teks D_1 = <boy, got, pair, shoes, mom> dan D_2 = <baby, got, new, shows> adalah m = 5 dan n =4. Catatan: tahap

pre-processing telah membuang *stop word* seperti "the", "of", dan "a".

2. Hitung jumlah kata yang sama: d = 2 yaitu <got, shoes>.
3. Hilangkan kata yang sama dari kedua dokumen. D_1 = <boy, pair, mom> dan D_2 = <baby, new>.
4. Buat matriks similaritas berukuran (m-d) x (n-d). Label baris berupa kata di D_1 dan label kolom berupa kata di D_2. Tabel yang tersusun untuk contoh adalah sebagai berikut.

	baby	new
boy		
pair		
shop		

5. Isi sel matriks dengan nilai relatif bi-gram dari kata di kolom dan baris (Davies, 2011).

	baby	new
boy	1.605e-4	3.09e-5
pair	0	9.03e-5
shop	0	3.66e-6

6. Buat sebuah vektor R yang isinya diambil dari nilai terbesar dari setiap baris matriks similaritas.
   R = {1.605 x 10^{-4}, 9.03 x 10^{-5}, 3.66 x 10^{-6}}
7. Hitung similaritas:

$$\mathrm{Sim}(D_1, D_2) = \frac{(d + \sum R)(m+n)}{2mn}$$

Hasil perhitungan similaritas dengan algoritma IMK menghasilkan Sim(D_1, D_2) = 0.5501. Jika dilakukan dengan fungsi similaritas kosinus, diperoleh Sim(D_1, D_2) = 0.5477.

Penelitian ini menggunakan data contoh klasifikasi yang dapat diunduh di situs http://www.python-course.eu. Data berisi kumpulan dokumen teks yang dikelompokkan dalam dua direktori yaitu "learn" dan "test". Direktori "learn" berisi dokumen-dokumen yang sudah terklasifikasi dan menjadi dasar penentuan kelas dokumen yang lain. Direktori "test" berisi dokumen-dokumen yang akan diklasifikasi program aplikasi. Dokumen dalam direktori "test" telah diklasifikasi oleh pakar dan digunakan untuk melakukan validasi terhadap hasil klasifikasi oleh program aplikasi. Direktori "learn" mengandung 711 dokumen terdiri atas 5 kelas, yang masing-masing kelas mempunyai dokumen sebanyak 61, 198, 125, 198, dan 122. Jumlah dokumen yang akan diuji (ada di direktori "test") sebanyak 25 berkas, yaitu 5 berkas untuk setiap kelas. Topik untuk setiap kelas adalah "Clinton", "Lawyer", "Math", "Medical" dan "Music" dan selanjutnya hanya akan disebut secara berturut-turut kelas 1 hingga kelas 5.

Hasil klasifikasi oleh setiap algoritma direkapitulasi ke dalam sebuah *confusion matrix*. Nilai pada baris "Kelas 1" dan kolom "Kelas 2" berarti jumlah dokumen yang dinilai pakar termasuk kelas 1 dan diidentifikasi oleh program aplikasi sebagai dokumen kelas 2.

Analisis Pakar	Analisis oleh program				Total hasil analisis pakar
	Kelas 1	Kelas 2	...	Kelas n	
Kelas 1	M_{11}	M_{12}		M_{1n}	
Kelas 2	M_{21}	M_{22}		M_{2n}	
...					
Kelas n	M_{n1}	M_{n2}		M_{nn}	
Total hasil analisis program					

Kinerja algoritma diukur dari parameter *Precision*, *Recall* dan *F-Measure*. Pada konteks multi kelas, *Recall* terhadap kelas j menunjukkan banyaknya dokumen teridentifikasi sebagai kelas j dibanding jumlah total dokumen untuk kelas yang sama. *Precision* terhadap kelas j menunjukkan jumlah dokumen teridentifikasi secara benar sebagai kelas j dibanding jumlah dokumen yang teridentifikasi sebagai kelas j oleh program aplikasi (Sokolova & Lapalme, 2009). F-Measure merupakan parameter yang menggabungkan dua parameter lain. Nilai terbaik untuk ketiga parameter adalah 1 dan nilai terburuk 0. Secara matematis,

$$R_j = \frac{M_{jj}}{\sum M_{ji}} \quad P_j = \frac{M_{jj}}{\sum M_{kj}}$$

$$F_j = 2P_jR_j / \left(P_j + R_j\right)$$

$$F = \frac{F_j \sum M_{ji}}{n}$$

C. HASIL PENGAMATAN DAN DISKUSI

Proses klasifikasi terhadap 25 dokumen teks dengan metode *k-nearest neighbour* dengan menggunakan fungsi similaritas kosinus menghasilkan *confusion matrix* sebagai berikut.

Analisis Pakar	Analisis oleh program					Total hasil analisis pakar
	Kelas 1	Kelas 2	Kelas 3	Kelas 4	Kelas 5	
Kelas 1	3	1			1	5
Kelas 2		3			2	5
Kelas 3		1	2	1	1	5
Kelas 4				4	1	5
Kelas 5		1			4	5
Total hasil analisis program	3	6	2	5	9	

Berdasarkan matriks tersebut, diperoleh nilai *Recall*, *Precision* dan *F-Measure* sebagai berikut.

R = <0.6, 0.6, 0.4, 0.8, 0.8>

P = <1, 0.5, 1, 0.8, 0.444>

F = 0.648

Ketika menggunakan algoritma IMK, diperoleh *confusion matrix* sebagaimana tabel berikut.

Analisis Pakar	Analisis oleh program					Total hasil analisis pakar
	Kelas 1	Kelas 2	Kelas 3	Kelas 4	Kelas 5	
Kelas 1	5					5
Kelas 2		4			1	5
Kelas 3		2	2		1	5
Kelas 4				4	1	5
Kelas 5		1			4	5
Total hasil analisis program	5	7	2	4	7	

Sehingga diperoleh nilai *Recall, Precision* dan *F-Measure* sebagai berikut.

R = <1, 0.8, 0.4, 0.8, 0.8>

P = <1, 0.571, 1, 1, 0.571>

F = 0.759

Hasil perhitungan menunjukkan bahwa proses klasifikasi dengan metode *k-nearest neighbour* yang menerapkan algoritma IMK mempunyai performa yang lebih baik. Nilai *F-Measure* berada pada angka 0.759 dan secara signifikan lebih baik dari nilai *F-Measure* untuk proses klasifikasi yang menerapkan fungsi similaritas kosinus. Begitu pula nilai *Recall* dan *Precision* untuk semua kelas menunjukkan bahwa kinerja algoritma IMK lebih baik.

Jika dilihat dari formula yang digunakan dalam perhitungan similaritas, algoritma IMK menggunakan rumusan yang mirip dengan fungsi similaritas Dice untuk menghitung kemiripan dua objek. Pada fungsi Dice, dua kata yang tidak sama akan mempunyai skor nol sedangkan pada algoritma IMK terdapat penambahan skor keterkaitan kata jika kedua kata tidak sama. Kinerja fungsi Dice tidak jauh berbeda dibanding kinerja fungsi kosinus. Jika mencermati hasil penelitian oleh Hamzah, dkk. (2008) tentang klustering dokumen berita bahasa Indonesia, diperoleh gambaran bahwa fungsi similaritas kosinus mempunyai performa terbaik diukur dari nilai *F-Measure* maupun kecepatan eksekusi program dan fungsi Dice mempunyai performa sedikit di bawah fungsi kosinus. Penelitian yang dilakukan penulis menunjukkan bahwa klasifikasi yang menerapkan algoritma Dice yang dimodifikasi oleh IMK mempunyai performa lebih baik dibanding klasifikasi yang menerapkan fungsi kosinus ditinjau dari nilai *F-Measure*.

D. KESIMPULAN

Penelitian ini menyimpulkan bahwa penggunaan keterkaitan kata yang diperoleh dari Google *bi-gram* dalam fungsi Dice meningkatkan skor similaritas dua dokumen dibanding jika similaritas dihitung dengan fungsi kosinus tanpa memperhatikan keterkaitan kata. Peningkatan skor similaritas antara dua dokumen berimplikasi pada perbaikan kinerja proses klasifikasi teks dengan metode *k-nearest neighbour*. Penerapan algoritma Islam-Milios-Keselj yang merupakan fungsi Dice dengan keterkaitan kata berdasarkan Google *n-gram* menghasilkan kinerja klasifikasi dengan *F-Measure* sebesar 0.759 sedangkan penerapan fungsi similaritas kosinus menghasilkan *F-Measure* sebesar 0.648.

DAFTAR PUSTAKA

Aggarwal, C. C., dan Zhai, C. 2012. "A survey of text classification algorithms," dalam *Mining Text Data*. hal. 163-222. Springer US.

Davies, M. 2011. N-grams data from the Corpus of Contemporary American English (COCA). Diunduh dari http://www.ngrams.info pada 14 Agustus 2014.

Hamzah, A., Soesianto, F., Susanto, A., Istiyanto, J.E., 2008. "Studi Kinerja Fungsi-Fungsi Jarak dan Similaritas dalam Clustering Dokumen Teks Berbahasa Indonesia," dalam *Prosiding Seminar Nasional Informatika 2008 (semnasIF 2008)*, Yogyakarta.

Islam, I., Milios, E., Keselj, V. 2012. "Text Similarity using Google Tri-Grams," dalam *25th Canadian Conference on Advances in Artificial Intelligence*, Mei 28-30, hal. 312-317.

Leacock, C. dan Chodorow, M., 1998. "Combining Local Context and WordNet Sense Similiarity for Word Sense Disambiguation," dalam *WordNet, An Electronic Lexical Database*, The MIT Press.

Lesk, M.E., 1986. "Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to tell a Pine Cone from an Ice Cream Cone," dalam *Proceedings of the SIGDOC Conference 1986*, Toronto, Juni.

Manning, C. D., Raghavan, P., Schütze, H. 2009. *Introduction to Information Retrieval*. Cambridge University Press.

Sentosa, Budi. 2007. *Data Mining Teknik Pemanfaatan Data untuk Keperluan Bisnis*. Graha Ilmu. Surabaya.

Sokolova, M., & Lapalme, G. 2009. A systematic analysis of performance measures for classification tasks, dalam *Information Processing and Management* 45, hal. 427-437.

Thamrin, H., Wantoro, J. 2014. "An Attempt to Create an Automatic Scoring Tool of Short Text Answer in Bahasa Indonesia" dalam *Proceeding of International Conference on Electrical Engineering, Computer Science and Informatics (EECSI 2014)*, Yogyakarta, Indonesia, hal. 96-98.

Wu, Z. dan Palmer, M., 1994. "Verb Semantics and Lexical Selection." dalam *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, Las Cruces, New Mexico.

Yazdani, M., dan Popescu-Belis, A., 2012. "Computing text semantic relatedness using the contents and links of a hypertext encyclopedia," dalam *Artificial Intelligence*.