

BAB I

PENDAHULIAN

1.1 Latar Belakang

Semakin dekatnya Pemilihan Umum 2014 membuat berbagai partai politik mencari cara yang tepat untuk mengetahui pendapat masyarakat. Dimana kondisi masyarakat yang memiliki pendapat berbeda mendapat perlakuan yang berbeda pula. Baik itu pendapat yang bersifat positif, negatif, maupun netral. Salah satu media yang digunakan oleh masyarakat dalam mengutarakan pendapatnya adalah situs jejaring sosial. Dengan demikian sebuah partai politik dapat menggunakannya untuk mengetahui pendapat masyarakat dunia maya dan mengelompokkannya ke dalam beberapa kelas.

Pendapat yang telah dikumpulkan juga harus diketahui persebarannya sehingga bisa diketahui pada daerah mana saja yang dominan untuk kelas tertentu. Selain itu pendapat yang diinginkan harus bersifat update dan realtime. Sehingga dapat mengetahui bagaimana perubahan yang terjadi dari waktu ke waktu pada suatu daerah.

Klasifikasi pendapat masyarakat pada tema tugas akhir ini diharapkan dapat membantu dalam mengetahui pendapat masyarakat terhadap partai tertentu. Metode yang digunakan adalah *Expectation Maximization* yang mana merupakan pengembangan metode *Naïve Bayes*. Metode ini dipilih karena kemudahannya dalam proses penghitungan serta akurasi yang cukup tinggi.

1.2 Rumusan Masalah

Dari latar belakang yang ada, maka didapat suatu permasalahan yaitu:

1. Bagaimana mengimplementasikan klasifikasi pendapat masyarakat jejaring sosial *Twitter* terhadap partai tertentu menggunakan metode *Expectation Maximization*?
2. Bagaimana menampilkan pendapat masyarakat jejaring sosial *Twitter* dalam bentuk peta digital berdasarkan distribusi daerahnya di Indonesia?
3. Sejauh mana akurasi metode *Expectation Maximization* dalam mengklasifikasikan pendapat masyarakat jejaring sosial terhadap suatu partai tertentu?

1.3 Tujuan dan Manfaat

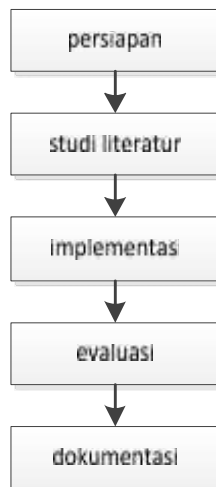
Tujuan dari penelitian ini adalah membangun aplikasi yang dapat mengumpulkan dan mengklasifikasikan pendapat masyarakat (positif, negatif, dan netral) jejaring sosial terhadap suatu partai politik tertentu kemudian menampilkannya berdasarkan daerah di Indonesia ke dalam sebuah peta digital secara *real time*.

1.4 Batasan Masalah

Batasan masalah dalam Tugas Akhir ini mencakup:

1. Pengumpulan pendapat hanya dilakukan pada pendapat masyarakat jejaring sosial *Twitter*.
2. Pengumpulan pendapat hanya dilakukan pada pendapat masyarakat jejaring sosial terhadap lima partai besar pemilu 2009, yaitu Partai Demokrat, Partai Golongan Karya (GOLKAR), Partai Demokrasi Indonesia Perjuangan (PDIP), Partai Keadilan Sejahtera (PKS), dan Partai Amanat Nasional (PAN)[1].
3. Pengklasifikasian dikelompokkan menjadi positif, negatif, dan netral menggunakan metode *Expectation Maximization*.
4. Klasifikasi hanya pada dokumen (*tweet*) dalam bahasa Indonesia.

1.5 Metodologi Penelitian



Gambar 1.1 Metodologi Penelitian

Secara umum Metodologi Penelitian adalah sebagai berikut:

1. Persiapan
Pada tahap ini dicari suatu bentuk permasalahan dimana permasalahan tersebut merupakan permasalahan baru dan belum atau kurang mendapatkan solusi. Diharapkan dengan adanya penelitian ini dapat membantu baik secara menyeluruh maupun sebagian dalam menyelesaikan masalah tersebut.
2. Studi Literatur
Pada tahap ini dilakukan pencarian tentang sumber-sumber kajian ataupun penelitian terkait yang sebelumnya pernah dilakukan dalam ranah yang serupa. Sumber di dapatkan dari buku teks, jurnal ilmiah, artikel, tugas akhir dan sumber lain baik dalam bentuk *hard-copy* maupun *soft-copy* yang terkait dengan penelitian.
3. Implementasi
Pada tahap ini implementasi metode yang didapatkan terhadap permasalahan yang ada. Mulai dari design sistem, pengumpulan variabel, coding, design, dan uji coba sistem.
4. Evaluasi
Pada tahap ini dilakukan proses evaluasi terkait hasil yang didapat apakah sesuai dengan perancangan atau tidak serta memperbaiki kesalahan yang mungkin terjadi untuk kemudian menarik kesimpulan dari hasil yang diperoleh.
5. Dokumentasi
Pada tahap ini seluruh proses dalam penelitian dicatat dan didokumentasikan agar dapat digunakan untuk mengembangkan penelitian selanjutnya.

1.6 Sistematika Penulisan

Sistematika dalam penulisan tugas akhir ini adalah :

BAB I PENDAHULUAN

Bab ini mengandung penjelasan latar belakang, rumusan masalah, tujuan dan manfaat, batasan masalah, metodologi penelitian dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini mengandung dasar teori yang mendukung pembuatan sistem klasifikasi pendapat masyarakat dunia maya terhadap partai politik dengan metode *expectation maximization*.

BAB III PERANCANGAN SISTEM

Bab ini menjelaskan tentang rancangan sistem yang akan digunakan dalam pembuatan program klasifikasi menggunakan metode *expectation maximization*. Baik perancangan algoritma maupun *database*.

BAB IV IMPLEMENTASI SISTEM

Bab ini menjelaskan tentang batasan studi kasus yang digunakan, detail hardware dan software yang digunakan serta hasil yang didapat dari proses terhadap data yang digunakan dengan menggunakan metode *expectation maximization*.

BAB V PENUTUP

Bab ini berisi tentang kesimpulan dari sistem yang telah dibuat serta hasil yang didapat dan saran terhadap penerapan sistem baru.

BAB II

STUDI PUSTAKA

2.1 Penelitian sebelumnya

2.1.1 Sistem Analisis Opini Microblogging Berbahasa Indonesia.

Penelitian yang telah dilakukan oleh Harlili dan Yudi Wibisono ini mengembangkan sistem sentimen analisis otomatis untuk *tweet* bahasa Indonesia. Dengan menggunakan klasifikasi *Naïve Bayes*, mereka menangani pemisahan *tweet* opini dan nonopini, *tweet* opini negatif dan positif, dan key word yang berkaitan dengan opini positif dan negatif tersebut.

33 ribu *tweet* (atau lebih) dalam rentang satu bulan dapat ditampilkan dalam bentuk grafik, dan difilter secara otomatis mana yang opini dan apakah opini tersebut negatif atau positif, beserta *keyword* yang relevan.

Hasil eksperimen menunjukkan akurasi untuk model klasifikasi *tweet* opini dan nonopini adalah 91.83% sedangkan untuk model klasifikasi opini negatif dan positif mencapai 93.40%. Sedangkan untuk *keyword extraction* masih kurang memuaskan karena hanya 34.48% *keyword* yang dapat diambil[2].

2.1.2 Analisis Sentimen dan Ekstraksi Topik Penentu Sentimen pada Opini terhadap Tokoh Publik.

Ismail Sunni dan Dwi Hendratmo Widyantoro juga memanfaatkan jejaring sosial *twitter* untuk mengetahui opini atau sentimen terhadap tokoh publik dan dapat mengetahui kapan terjadi perubahan sentimen dan topik apa yang menyebabkan perubahan sentimen tersebut. Pada penelitiannya mereka juga menggunakan *Naïve Bayes* dan *Tf-Idf discounted-cumulative*. Tahap praproses ternyata tidak memberi pengaruh yang signifikan (69.4%-72.8%) klasifikasi sentimen. Sementara metode *ConvertEmoticon* dan *ConvertNumber* mampu menaikkan akurasi. *Emoticon* (emotion icon) merupakan salah satu cara pengungkapan perasaan secara tekstual. Hal ini tentu akan membantu dalam menentukan sentimen suatu *tweet*. Menurut Aqsath & Ayu (2011) *ConvertNumber* dilakukan pada angka yang muncul di tengah-tengah kata[3].

2.1.3 Text Mining dengan Metode *Naïve Bayes Classifier* dan *Support Vector Machines Naïve Bayes Support Vector Machines* untuk Sentiment Analysis

Pada penelitian ini dokumen bahasa Inggris diambil dari opini film sementara dokumen bahasa Indonesia diambil dari Bali Pos. Dalam penelitiannya Ni Wayan melakukan percobaan terhadap bahasa Inggris dan bahasa Indonesia menggunakan *Naïve Bayes Classifier* (NBC) dan *Support Vector Machine* (SVM). Percobaannya membuktikan bahwa SVM lebih baik dari NBC untuk opini bahasa Inggris dan opini positif bahasa Indonesia. Untuk opini negatif bahasa Indonesia, NBC bekerja lebih baik pada data ujinya[4].

2.1.4 Twitter Sentiment Analysis: The Good the Bad and the OMG!

Penelitian Efthymios K. dan teman-temannya ini melakukan sentimen analisis pada twitter. Pada penelitian mereka, hasil terbaik mereka dapatkan dari penggunaan *n-gram*, fitur kosakata, dan fitur *microblogging* (kosakata dan emoticon). Metode *part-of-speech* tidak terlalu berfungsi pada sebuah *microblogging*[5].

2.1 Landasan Teori

Twitter API

Application Programming Interface (API). Merupakan protokol yang menyediakan semua data *Twitter* dan digunakan untuk membangun aplikasi yang dapat mengakses *Twitter*.

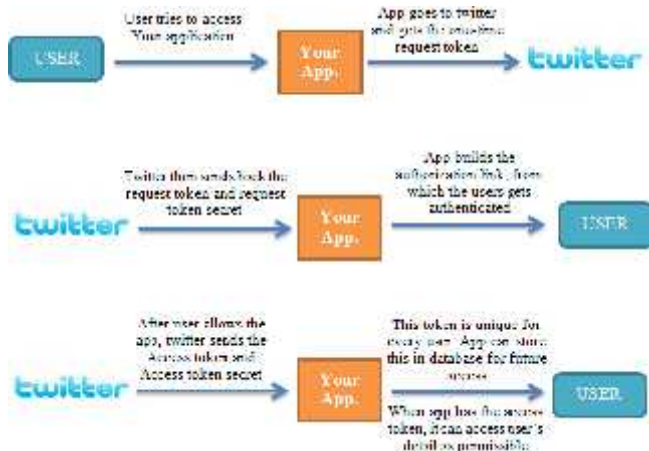
Twitter API dibagi dalam tiga bagian[6]:

1. *REST API*. Digunakan untuk aplikasi untuk melakukan post Tweets, mem-follow seseorang, create list, dan sebagainya.
2. *Search API*. Digunakan untuk melakukan pencarian, seperti yang biasa dilakukan pada *Twitter*.
3. *Streaming API*. Digunakan untuk menerima stream real-time *Tweets public* dari *Twitter*.

Dalam *Twitter API* juga dikenal istilah *OAuth (Open Authentication)*. *OAuth* adalah sebuah sistem autentifikasi/pengenalan user *Twitter* untuk menggunakan layanan berbasis *Twitter*[7].

Secara umum *OAuth* merupakan standar terbuka yang memungkinkan orang untuk berbagi sumber daya secara

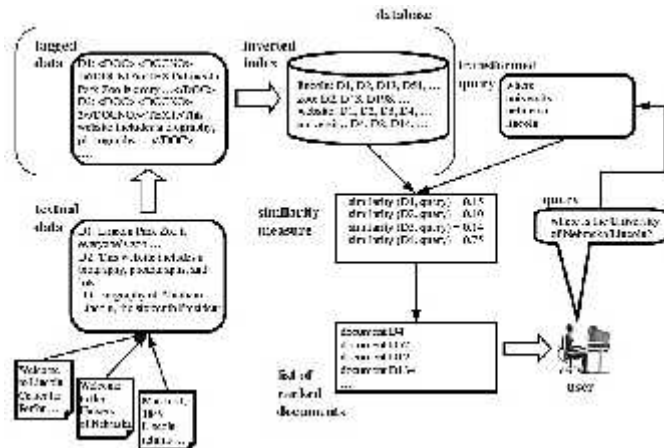
aman dari satu situs (yang mendukung *OAuth*: dalam hal ini *Twitter*) ke website lain pihak ketiga.



Gambar 2.1 Ilustrasi bagaimana Twitter mengizinkan OAuth mengakses data user[8]

2.3 Information Retrieval

ISO 2382/1 mendefinisikan *Information Retrieval* (IR) sebagai tindakan, metode dan prosedur untuk menemukan kembali data yang tersimpan, kemudian menyediakan informasi mengenai subyek yang dibutuhkan. Tindakan tersebut mencakup *text indexing*, *inquiry analysis*, dan *relevance analysis*. Data mencakup teks, tabel, gambar, ucapan, dan video. Informasi termasuk pengetahuan terkait yang dibutuhkan untuk mendukung penyelesaian masalah dan akuisisi pengetahuan[9].



Gambar 2.2 Arsitektur dasar dari sistem IR [9]

2.4 Pembuatan Index

Pembangunan *index* dari koleksi dokumen merupakan tugas pokok pada tahapan *preprocessing* di dalam IR. Kualitas *index* mempengaruhi efektifitas dan efisiensi sistem IR [10]. *Index* dokumen adalah himpunan *term* yang menunjukkan isi atau topik yang dikandung oleh dokumen. *Index* akan membedakan suatu dokumen dari dokumen lain yang berada di dalam koleksi. Ukuran *index* yang kecil dapat memberikan hasil buruk dan mungkin beberapa *item* yang relevan terabaikan. *Index* yang besar memungkinkan ditemukan banyak dokumen yang relevan tetapi sekaligus dapat menaikkan jumlah dokumen yang tidak relevan dan menurunkan kecepatan pencarian (*searching*) [11].

Pembuatan *inverted index* harus melibatkan konsep *linguistic processing* yang bertujuan mengekstrak *term-term* penting dari dokumen yang direpresentasikan sebagai *bag-of-words*. Ekstraksi *term* biasanya melibatkan dua operasi utama berikut [9]:

1. Penghapusan ***stop-words***. *Stop-word* didefinisikan sebagai *term* yang tidak berhubungan (*irrelevant*) dengan subyek utama dari database meskipun kata tersebut sering kali hadir di dalam dokumen. Contoh *stop words* adalah *a, an, the, this, that, these, those, her, his,*

its, my, our, their, your, all, few, many, several, some, every, for, and, nor, bit, or, yet, so, also, after, although, if, unless, because, on, beneath, over, of, during, beside, dan etc.

Stop-words termasuk pula beberapa kata tertentu yang didefinisikan terkait dengan topik database, misal pada database yang menampung daftar karya tulis (*paper*) penelitian terkait dengan *heart diseases*, maka kata *heart* dan *disease* sebaiknya dihapus.

2. **Stemming.** Kata-kata yang muncul di dalam dokumen sering mempunyai banyak varian morfologik. Karena itu, setiap kata yang bukan *stop-words* direduksi ke bentuk *stemmed word (term)* yang cocok. Kata tersebut distem untuk mendapatkan bentuk akarnya dengan menghilangkan awalan atau akhiran. Dengan cara ini, diperoleh kelompok kata yang mempunyai makna serupa tetapi berbeda wujud sintaktis satu dengan lainnya. Kelompok tersebut dapat direpresentasikan oleh satu kata tertentu. Sebagai contoh, kata *disease, diseases, diseased* dapat dikatakan serupa atau satu kelompok dan dapat diwakili oleh satu kata umum *disease*.

Algoritma *stemming* paling umum yang diterapkan pada bahasa Inggris dan dinyatakan efektif adalah algoritma Porter [12]. Algoritma ini terdiri dari lima fase reduksi kata yang diterapkan secara urut. Di dalam setiap fase terdapat berbagai konvensi untuk memilih aturan. Pada fase pertama, konvensi yang digunakan mengikuti aturan grup berikut:

Rule	Contoh
SS → SS	<i>caresses</i> → <i>caress</i>
IES → I	<i>ponies</i> → <i>poni</i>
SS → SS	<i>caress</i> → <i>caress</i>
S →	<i>cats</i> → <i>cat</i>

Banyak aturan berikutnya menggunakan konsep ukuran kata dengan memeriksa jumlah suku kata untuk mengetahui apakah suatu kata cukup panjang sehingga dapat diberlakukan aturan untuk menghilangkan akhirnya, bukan menghapus dari akar kata. Sebagai

contoh, aturan:

$(m > 1)$ EMENT $\rightarrow$

mengubah “*replacement*” menjadi “*replac*” tetapi tidak mengubah “*cement*” tidak menjadi “*c*”.

Menurut [13, 14] terdapat 5 langkah pembangunan *inverted index*, yaitu:

1. Penghapusan format dan *markup* dari dalam dokumen

Tahap ini menghapus semua *tag markup* dan format khusus dari dokumen, terutama pada dokumen yang mempunyai banyak *tag* dan format seperti dokumen (X)HTML. Jika isi dokumen telah berada di dalam database maka tahapan ini sering dilewatkan.

2. Pemisahan rangkaian kata (*tokenization*)

Tokenization adalah tugas memisahkan deretan kata di dalam kalimat, paragraf atau halaman menjadi *token* atau potongan kata tunggal atau *termmed word*. Tahapan ini juga menghilangkan karakter-karakter tertentu seperti tanda baca dan mengubah semua *token* ke bentuk huruf kecil (*lower case*).

3. Penyaringan (*filtration*)

Pada tahapan ini ditentukan *term* mana yang akan digunakan untuk merepresentasikan dokumen sehingga dapat mendeskripsikan isi dokumen dan membedakan dokumen tersebut dari dokumen lain di dalam koleksi. *Term* yang sering dipakai tidak dapat digunakan untuk tujuan ini, setidaknya karena dua hal. Pertama, jumlah dokumen yang relevan terhadap suatu query kemungkinan besar merupakan bagian kecil dari koleksi. *Term* yang efektif dalam pemisahan dokumen yang relevan dari dokumen tidak relevan kemungkinan besar adalah *term* yang muncul pada sedikit dokumen. Ini berarti bahwa *term* dengan frekuensi kemunculan tinggi bersifat *poor discriminator*. Kedua, *term* yang muncul dalam banyak dokumen tidak mencerminkan definisi dari topik atau sub-topik

dokumen. Karena itu, *term* yang sering digunakan dianggap sebagai *stop-word* dan dihapus[14]. Penghapusan *stop-word* dari dalam suatu koleksi dokumen pada satu waktu membutuhkan banyak waktu. Solusinya adalah dengan menyusun suatu pustaka *stop-word* atau *stop-list* dari *term* yang akan dihapus. Strategi umum penentuan *stop-list* adalah mengurutkan *term* berdasarkan frekuensi koleksi (jumlah total kemunculan setiap *term* di dalam koleksi dokumen) dan memasukkan *term* yang paling sering muncul sebagai *stop-word*[13].

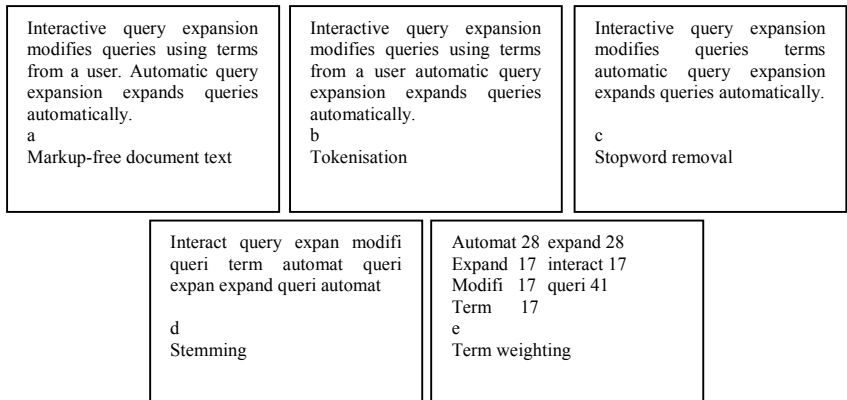
4. Konversi *term* ke bentuk akar (*stemming*)

Stemming adalah proses konversi *term* ke bentuk umumnya, sebagaimana dijelaskan sebelumnya. Dokumen dapat pula diekspansi dengan mencari sinonim bagi *term-term* tertentu di dalamnya. Sinonim adalah kata-kata yang mempunyai pengertian serupa tetapi berbeda dari sudut pandang morfologis. Seperti *stemming*, operasi ini bertujuan menemukan suatu kelompok kata terkait. Akan tetapi sinonim bekerja berdasarkan pada *thesaurus*, tidak berbagi-pakai *term stem*. Jika pengguna memasukkan query “*heart disease*” maka query diekspansi untuk mengakomodasi semua sinonim dari *disease* seperti *ailment*, *complication*, *condition*, *disorder*, *fever*, *ill*, *illness*, *infirmity*, *malady*, *sickness*, dan lain-lain [12].

5. Pemberian bobot terhadap *term* (*weighting*).

Setiap *term* diberikan bobot sesuai dengan skema pembobotan yang dipilih, apakah pembobotan lokal, global atau kombinasi keduanya. Banyak aplikasi menerapkan pembobotan kombinasi berupa perkalian bobot lokal *term frequency* dan global *inverse document frequency*, ditulis *Tf.Idf*.

Di bawah ini ilustrasi dari proses *indexing* sedangkan contoh struktur index diperlihatkan pada tabel 2.1.



Contoh lima tahap indexing pada sistem berbasis content secara urut mulai dari markup removal (a), tokenization (b), stopwords filtration (c), stemming (d) dan weighting (e)[15]

Tabel 2.1 Contoh struktur index dalam bentuk tabel[16,17]

Term	Posisi (Nomor dokumen)
automat	(1,2);(3,5);(4,3); dst s.d ke-28
expan	(1,3);(6,5);(10,13); dst s.d ke-28
expand	(7,2);(8,5);(9,1); dst s.d ke-17
interact	(21,7);(30,5);(34,2); dst s.d ke-17
modify	(12,1);(13,5);(14,12); dst s.d ke-17
query	(9,8);(13,5);(14,1); dst s.d ke-41
term	(6,2);(22,5);(23,3); dst s.d ke-17

2.5 Term Documents Matrix

Term documents matrix adalah representasi kumpulan dokumen yang akan digunakan untuk melakukan proses klasifikasi dokumen teks. Dalam *term documents matrix*, sebuah dokumen direpresentasikan sebagai kumpulan fitur dan dapat diilustrasikan sebagai $d_j = [w_{1j}, w_{2j}, \dots, w_{kj}]$ dengan d_j merupakan dokumen ke- j dan w_{kj} merupakan nilai kemunculan fitur ke- k pada dokumen d_j . Matriks ini akan berisi nilai-nilai kemunculan fitur. Nilai kemunculan fitur yang digunakan sesuai dengan jenis fitur

pembobotan *Tf-Idf*. Baris pada *term documents matrix* merupakan data dokumen, sedangkan kolom dari *term documents matrix* merupakan fitur yang digunakan. Berikut gambaran *term documents matrix*[18]:

	w_1	w_2	w_3	$\cdot$	w_k
d_1	w_{11}	w_{21}	w_{31}	$\cdot$	w_{k1}
d_2	w_{12}	w_{22}	w_{32}	$\cdot$	w_{k2}
d_3	w_{13}	w_{23}	w_{33}	$\cdot$	$\cdot$
$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$
$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$
d_j	w_{ij}	w_{2j}	$\cdot$	$\cdot$	w_{kj}

2.6 Pembobotan *TF-IDF*

Algoritma *TF-IDF* adalah suatu algoritma yang berdasarkan nilai statistik menunjukkan kemunculan suatu kata di dalam dokumen. *TF* (Term Frequency) menyatakan banyaknya suatu kata muncul dalam sebuah dokumen. Dan *DF* (*Document Frequency*) menyatakan banyaknya dokumen yang mengandung suatu kata dalam satu segmen publikasi. *TF-IDF* adalah nilai bobot dari suatu kata yang diambil dari nilai *IF* dan nilai Inverse *DF*, yang didefinisikan dengan[19]:

$$IDF(w) = \log \frac{N}{DF(w)}$$

$$IDF(w) = \log \quad (1)$$

$$TF - IDF(w,d) = TF(w,d) \times IDF(w) \quad (2)$$

Keterangan:

$TF-IDF(w,d)$: Bobot suatu kata dalam keseluruhan dokumen

w : Suatu kata (*word*)

d : Suatu dokumen (*document*)

$TF(w,d)$: frekuensi kemunculan sebuah kata w dalam dokumen d

$IDF(w)$: inverse DF dari kata w

N : jumlah keseluruhan dokumen

$DF(w)$: jumlah dokumen yang mengandung kata w

Berdasarkan rumus (1) di atas, berapapun besarnya nilai $TF(w,d)$, apabila nilai $N = DF(W)$ maka akan didapatkan hasil 0 (nol) untuk perhitungan DF . Untuk itu, dapat ditambahkan nilai 1 pada sisi DF , sehingga perhitungan $TF(w,d)$ menjadi sebagai berikut:

$$IDF(w) = \log \frac{N}{DF(w)}$$

$$TF - IDF(w,d) = TF(w,d) \times (\log (DF(w) + 1)) \quad (3)$$

Rumus (3) dinormalisasi dengan rumus (4) dengan tujuan untuk menstandarisasi nilai $TF(w,d)$ ke dalam interval 0 sampai 1. Rumus TF - DF dengan menggunakan normalisasi adalah[20]:

$$TF - IDF(w,d) = \frac{TF - IDF(w,d)}{\sqrt{\sum_{k=1}^N (TF - IDF(w,k))^2 \times \left[\log \left(\frac{N}{DF(w)} \right) - 1 \right]^2}} \quad (4)$$

Keterangan tambahan :

$TF(w,k)$: frekuensi kemunculan sebuah kata w dalam dokumen k , dimana dokumen k merujuk pada dokumen d

2.7 Klasifikasi Teks

Proses klasifikasi teks dapat dibagi ke dalam dua fase, yaitu[21]:

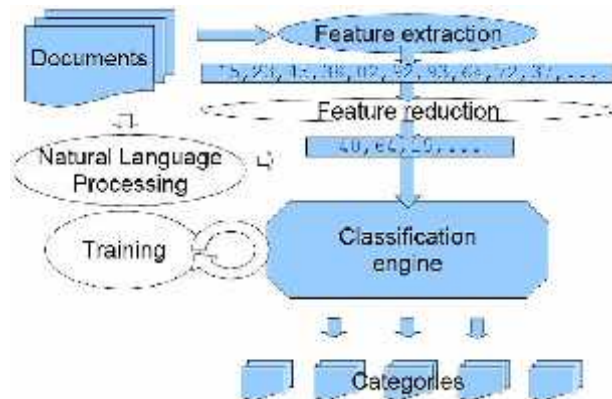
1. Fase *information retrieval* (IR) untuk mendapatkan data numerik dari dokumen teks. Langkah pertama yang dilakukan pada fase ini adalah *feature extraction*. Pendekatan yang umum digunakan adalah distribusi frekuensi kata. Nilai *numerik* yang diperoleh dapat berupa berapa kali suatu kata muncul di dalam dokumen, 1 jika kata ada di dalam dokumen atau 0 jika tidak ada (*biner*), atau jumlah kemunculan kata pada awal

dokumen. Fitur yang diperoleh dapat direduksi agar dimensi vektor menjadi lebih kecil. Beberapa pendekatan *feature reduction* dapat diterapkan seperti menghapus *stop-words*, *stemming*, dan *statistical filtering*. Teknik lebih lanjut seperti SVD dan *genetic algorithm* akan menghasilkan vektor berdimensi lebih rendah.

2. Fase klasifikasi utama. Suatu algoritma memroses data numerik di atas untuk memutuskan ke kategori mana teks baru (bukan contoh) ditempatkan. Terdapat beberapa algoritma klasifikasi yang merupakan kajian di bidang statistika dan *machine learning* yang dapat diterapkan pada fase ini, di antaranya adalah *Naïve Bayesian*, *Rocchio*, *Decision Tree*, *k-Nearest Neighbor* (k-NN), *Neural Network* (NN), dan *Support Vector Machines* (SVM). Teknik-teknik tersebut berbeda dalam mekanisme pembelajaran dan representasi model yang dipelajari [22]. *K-NN* dan *SVM* merupakan algoritma yang memberikan hasil klasifikasi terbaik dengan presisi di atas 85% [23].

Proses klasifikasi teks melibatkan banyak teknik *IR* mulai dari *preprocessing*, pengukuran kemiripan selama operasional klasifikasi sampai dengan evaluasi kinerja algoritma[24].

Gambar 4 memperlihatkan diagram proses klasifikasi teks secara garis besar.



Gambar 2.3 Diagram proses klasifikasi teks[21]

2. 8 Machine Learning

Machine learning adalah sebuah studi yang mempelajari cara untuk memprogram sebuah komputer untuk belajar.

2.3.1 *Supervised Learning*

Dalam teknik *supervised learning*, maka sebuah program harus dapat membuat klasifikasi – klasifikasi dari contoh-contoh yang telah diberikan. Misalnya sebuah program diberikan benda berupa bangku dan meja, maka setelah beberapa contoh, program tersebut harus dapat memilah-milah objek ke dalam klasifikasi yang cocok.

Kesulitan dari *supervised learning* adalah kita tidak dapat membuat klasifikasi yang benar. Dapat dimungkinkan program akan salah dalam mengklasifikasi sebuah objek setelah dilatih. Oleh karena itu, selain menggunakan *training set* kita juga memberikan *test set*. Dari situ kita akan mengukur persentase keberhasilannya. Semakin tinggi berarti semakin baik program tersebut.

Persentase tersebut dapat ditingkatkan dengan diketahuinya *temporal dependence* dari sebuah data. Misalnya diketahui bahwa 70% mahasiswa dari jurusan Teknik Informatika adalah laki-laki dan 80% mahasiswa dari jurusan Sastra adalah wanita. Maka program tersebut akan dapat mengklasifikasi dengan lebih baik[25].

2.3.2 *Unsupervised Learning*

Teknik ini menggunakan prosedur yang berusaha untuk mencari partisi dari sebuah pola. *Unsupervised learning* mempelajari bagaimana sebuah sistem dapat belajar untuk merepresentasikan pola input dalam cara yang menggambarkan struktur statistik dari keseluruhan pola input.

Berbeda dari *supervised learning*, *unsupervised learning* tidak memiliki target output yang eksplisit atau tidak ada pengklasifikasian input.

Dalam *machine learning*, teknik *unsupervised* sangat penting. Hal ini dikarenakan cara bekerjanya mirip dengan cara bekerja otak manusia. Dalam melakukan pembelajaran,

tidak ada informasi dari contoh yang tersedia. Oleh karena itu, unsupervised learning menjadi esensial[25].

2.3.3 *Semi Supervised Learning*

Seperti namanya, *semi-supervised* learning merupakan tempat antara *unsupervised* dan *supervised*. Bahkan, sebagian besar strategi pembelajaran *semi-supervised learning* didasarkan pada pengembangan keduanya untuk menambahkan informasi tambahan khas dari pola pembelajaran lainnya.

Secara khusus, semi-supervised learning memiliki beberapa perbedaan pengaturan, di antaranya[26]:

- **Semi-supervised classification.** Dikenal juga sebagai klasifikasi dengan data berlabel dan tidak berlabel (atau sebagian data berlabel), ini merupakan pengembangan untuk masalah supervised learning. Data pelatihan terdiri dari keduanya, l berlabel contoh $\{(x_i, y_i)\}_{i=1}^l$ dan u contoh tidak berlabel $\{x_j\}_{j=l+1}^{l+u}$. Salah satunya mengasumsikan bahwa ada lebih banyak lagi data yang tidak berlabel daripada data berlabel, yaitu $u \gg l$. Tujuan dari semi-supervised adalah untuk melatih classifier f dari kedua data berlabel dan tidak berlabel, sehingga lebih baik daripada supervised classifier pada pelatihan data berlabel saja.
- **Constrained clustering.** Ini adalah pengembangan untuk unsupervised clustering. Data training terdiri dari data tidak berlabel yaitu $\{x_i\}_{i=1}^n$, seperti "supervised information" dalam meng-cluster. Sebagai contoh, informasi tersebut dapat disebut *must-link*, yang kedua contoh x_i, x_j harus dalam *cluster* yang sama, dan *cannot-link* terbatas, dimana x_i, x_j tidak bisa dalam *cluster* yang sama. Salah satunya juga dapat membatasi ukuran *clusters*. Tujuan dari *constrained clustering* adalah

untuk mendapatkan *clustering* yang lebih baik daripada *clustering* dari data tidak berlabel saja.

Supervised Learning dibagi menjadi dua, yaitu[18]:

Fully supervised learning, yaitu teknik klasifikasi dimana semua dokumen training telah diketahui kategorinya. *Naïve Bayes* adalah contoh dari teknik *fully supervised*.

Semi supervised learning, yaitu teknik klasifikasi dimana pembelajaran dilakukan dari dokumen training yang telah diketahui kategorinya dan dokumen training yang belum diketahui kategorinya. *Expectation Maximization* adalah teknik semi supervised learning yang paling populer digunakan untuk klasifikasi teks

2.9 Bayesian Learning

Naïve Bayes merupakan salah satu metode *machine learning* yang menggunakan perhitungan probabilitas. Konsep dasar yang digunakan oleh *Naïve Bayes* adalah Teorema Bayes, yaitu melakukan klasifikasi dengan melakukan perhitungan nilai probabilitas $p(C = c_i | D = d_j)$, yaitu probabilitas kategori c_i jika diketahui dokumen d_j . Klasifikasi dilakukan untuk menentukan kategori $c \in C$ dari suatu dokumen $d \in D$ dimana $C = \{c_1, c_2, c_3, \dots, c_i\}$ dan $D = \{d_1, d_2, d_3, \dots, d_j\}$. Penentuan dari kategori sebuah dokumen dilakukan dengan mencari nilai maksimum dari $p(C = c_i | D = d_j)$ pada $P = \{p(C = c_i | D = d_j) | c \in C \text{ dan } d \in D\}$. Nilai probabilitas $p(C = c_i | D = d_j)$ dapat dihitung dengan persamaan[27]:

$$p(C = c_i | D = d_j) = \frac{P(C = c_i \cap D = d_j)}{P(D = d_j)}$$

$$= \frac{p(D = d_j | C = c_i) \times p(C = c_i)}{p(D = d_j)}$$

Dengan $p(D = d_j | C = c_i)$ merupakan nilai probabilitas dari kemunculan dokumen d_j jika diketahui dokumen tersebut berkategori c_i , $p(C = c_i)$ adalah nilai probabilitas

kemunculan kategori c_i , dan $p(D = d_j)$ adalah nilai probabilitas kemunculan dokumen d_j .

2.10 Expectation Maximization

Expectation maximization tidak jauh berbeda dengan *Naïve Bayes*. Hanya saja metode ini juga menggunakan unlabeled documents sebagai document trainingnya setelah didapat model probabilistiknya.

Expectation Maximization memiliki dua tahapan dalam melakukan klasifikasi dokumen teks, yang pertama adalah *expectation step* yaitu tahapan memperkirakan kategori dari setiap *unlabeled documents* dengan menentukan probabilitas $p(c_i | d_j)$ dengan perhitungan [18, 28]:

$$p(c_i | d_j) = \frac{p(c_i) \prod_{k=1}^{|d_j|} p(w_{kj} | c_i)}{\sum_{r=1}^{|C|} p(c_r) \prod_{k=1}^{|d_j|} p(w_{kj} | c_r)}$$

} Tahap Expectation

dengan $|d_j|$ merupakan jumlah kata pada dokumen d_j .

Setelah semua *unlabeled documents* memiliki kategori perkiraan, tahap berikutnya adalah *maximization step* yaitu tahap untuk melakukan *update* terhadap parameter klasifikasi yaitu probabilitas $p(w_{kj} | c_i)$ dan probabilitas $p(c_i)$ dengan perhitungan [18, 28]:

Meng-update nilai probabilistik awal:

$$p(w_{kj} | c_i) = \frac{1 + \sum_{j=1}^{|D|} N(w_{kj}, d_j) p(c_i | d_j)}{|W| + \sum_{s=1}^{|W|} \sum_{j=1}^{|D|} N(w_s, d_j) p(c_i | d_j)}$$

Dan

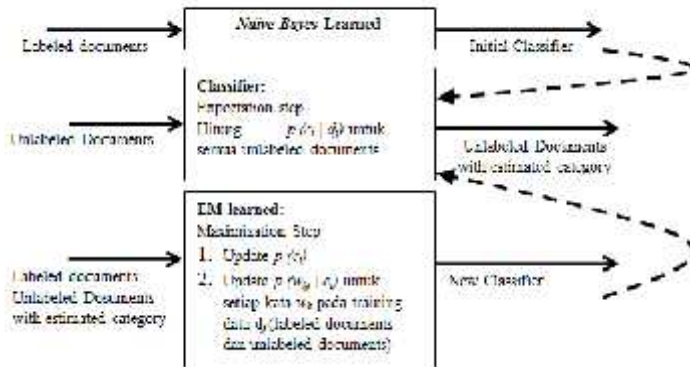
Meng-update nilai probabilistik untuk setiap kategori yang ada:

$$p(c_i) = \frac{1 + \sum_{j=1}^{|D|} p(c_i | d_j)}{|C| + |D|}$$

Dengan $N(w_{kj}, d_j)$ adalah nilai kata atau fitur w_k pada dokumen d_j dan $|W|$ merupakan jumlah keseluruhan

} Tahap Maximization

kata/fitur yang digunakan. Dua tahap tersebut akan terus dilakukan hingga perubahan parameter probabilitas $p(w_{kj} | c_i)$ dan $p(c_i)$ tidak melebihi batasan yang ditentukan dari iterasi sebelumnya. Proses yang terjadi pada algoritma Expectation Maximization dapat dilihat pada Gambar 5.



Gambar 2.4 Algoritma Klasifikasi Expectation Maximization[18]

Dari studi pustaka di atas maka dapat dikatakan metode Expectation Maximization merupakan metode yang membentuk sebuah probabilitas sekumpulan dokumen training. Probabilitas yang dibentuk pertama kali diperoleh dari dokumen training berlabel dan diperbaharui setelah melabeli dokumen training tidak berlabel. Sehingga probabilitas akhir yang didapatkan tidak secara mutlak nilai dari label manual, tetapi juga merupakan perhitungan yang mengandung nilai-nilai probabilitas label yang dihitung melalui *expectation step*.

