BAB 2
LANDASAN TEORI

2.1. Artikel

Artikel merupakan sebuah karangan faktual (non fiksi), tentang suatu masalah secara lengkap yang panjangnya tidak ditentukan, untuk dimuat di surat kabar, majalah, bulletin dan sebagainya dengan tujuan untuk menyampaikan gagasan dan fakta guna meyakinkan, mendidik, menawarkan pemecahan suatu masalah, atau menghibur. Artikel termasuk termasuk tulisan kategori views (pandangan), yaitu tulisan yang berisi pandangan, ide, opini, penilaian penulisannya tentang suatu masalah atau peristiwa.

2.2. Berita

Berita adalah cerita atau keterangan mengenai kejadian atau peristiwa yang hangat. Berita haruslah sesuai dengan kenyataan yang ada, tidak dibuat-buat (fiktif), dan terbaru/terkini. Berita merupakan salah satu cara berkomunikasi melalui peristiwa penting, terbaru, dan menarik. Kita dapat menjumpai berita dimana saja seperti di koran, majalah, internet, televisi, radio, bahkan di mading sekolah.

Berita merupakan sebuah uraian tentang fakta dan atau pendapat yang mengandung nilai berita dan yang sudah disajikan melalui media massa periodik [4]. Berita haruslah memiliki unsur yaitu baru, penting, bermakna, berpengaruh, menyangkut hidup orang banyak, relevan, dan menarik. Ada juga yang mengatakan bahwa berita merupakan apa saja yang ingin dan perlu diketahui orang atau lebih luas lagi oleh masyarakat.

2.3. *Text mining*

Text mining merupakan area baru dan menarik dari penelitian ilmu komputer yang mencoba menyelesaikan krisis informasi yang berlebihan dengan menggabungkan teknik dari data mining, pembelajaran mesin, pengalahan bahasa alami, pencarian informasi, dan manajemen pengetahuan" [9]. Text mining adalah bagian dari data mining dimana Text mining mencoba menggali pengetahuan yang terdapat pada sekumpulan teks atau sering disebut dokumen yang susunannya tidak terstruktur. Adapun tahapan utama yang dipakai pada text mining:

1. *Text Preprocessing*, pada tahap awal ini dilakukan pemrosesan terhadap data teks untuk diubah kedalam bentuk token atau kata yang sering disebut dengan tokenisasi. Data dalam bentuk dokumen, paragraf ataupun kalimat dipecah menjadi token-token sebagai masukan untuk proses selanjutnya yaitu *Text Transformation*. Dalam tokenisasi ini, dilakukan seleksi pada tanda baca, kapitalisasi, karakter spesial dan lain sebagaianya sehingga hasil yang didapatkan berupa kumpulan token.
2. *Text Transformation*, pada tahap ini proses yang dilakukan adalah melakukan *stemming* dan *stopword*. Stemming melakukan pemotongan atau penghilangan imbuhan pada token sehingga hasil yang didapat merupakan kata dasar dari kata tersebut, contoh pada kata “dipukul” dan “memukuli” memiliki kata dasar “pukul”. *Stemming* ini dilakukan untuk menurunkan jumlah daftar kata pada indeks. Selain itu juga terdapat proses *stopword*, dimana dilakukan penghilangan pada kata sambung yang sangat umum berada pada dokumen, contoh kata “yang” dan “pada”. Kata sambung tersebut dihilangkan karena terlalu bersifat umum sehingga tidak dapat mencirikan sebuah dokumen tertentu.
3. *Feature Selection*, melakukan tahap seleksi fitur dengan menghapus kata-kata yang yang dianggap tidak menggambarkan isi dari dokumen yang dapat dilakukan dengan melihat frekuensi kemunculan kata tersebut [10].

4. *Data Mining / Pattern Discovery*, merupakan tahap terpenting dalam text mining dimana dilakukan penerapan metode data mining dengan tujuan untuk menemukan pengetahuan yang tersimpan dalam teks[10].

2.3.1. *Preprocessing*

Preprocessing adalah tahapan untuk mempersiapkan teks menjadi data yang akan diolah di tahapan berikutnya. Inputan awal pada proses ini adalah berupa dokumen. Pada umumnya preprocessing memiliki beberapa tahapan yaitu *case folding, tokenizing, stop word removal, stemming*, dan lain-lain. *Preprocessing* pada penelitian ini terdiri dari beberapa tahapan, yaitu: proses pemecahan kalimat, proses *case folding*, proses *tokenizing* kata, dan proses *stop word removal*

2.3.1.1 Pemecahan Kalimat

Memecah dokumen menjadi kalimat-kalimat merupakan langkah awal tahapan preprocessing. Pemecahan kalimat yaitu proses memecah string teks dokumen yang panjang menjadi kumpulan kalimat-kalimat. Dalam memecah dokumen menjadi kalimat kalimat menggunakan fungsi split(), dengan tanda titik ".", tanda tanya "?" dan tanda tanya "!" sebagai delimiter untuk memotong string dokumen [11].

Tabel 2.1 Contoh Pemecah Kalimat

Kalimat	Hasil pemecahan kalimat
Manajemen transaksi elektronik. Pengetahuan antar individu. Dalam manajemen pengetahuan, terdapat transfer pengetahuan elektronik	Manajemen transaksi elektronik Pengetahuan antar individu Dalam manajemen pengetahuan terdapat transfer pengetahuan elektronik

2.3.1.2 Case Folding

Dokumen mengandung berbagai variasi dari bentuk huruf sampai tanda baca. Variasi huruf harus diseragamkan (menjadi huruf besar saja atau huruf kecil saja) dan tanda baca dihilangkan untuk menghilangkan *noise* pada saat pengambilan informasi. Hal ini dapat dilakukan dengan *case folding*. *Case folding* adalah tahapan proses mengubah semua huruf dalam teks dokumen menjadi huruf kecil, serta menghilangkan karakter selain a-z.[11].

Tabel 2.2 Contoh *Case Folding*

Kalimat	**Hasil *case folding***
Manajemen transaksi elektronik Pengetahuan antar individu Dalam manajemen pengetahuan terdapat transfer pengetahuan elektronik	manajemen transaksi elektronik pengetahuan antar individu dalam manajemen pengetahuan terdapat transfer pengetahuan elektronik

2.3.1.3 Tokenizing

Tokenizing adalah proses pemotongan string input berdasarkan tiap kata yang menyusunnya. Pemecahan kalimat menjadi kata-kata tunggal dilakukan dengan men-scan kalimat dengan pemisah (*delimiter*) *white space* (spasi, tab, dan *newline*)[11].

Tabel 2.3 Contoh *Tokenizing*

Kalimat	Hasil *Tokenizing*
manajemen transaksi elektronik pengetahuan antar individu dalam manajemen pengetahuan terdapat transfer pengetahuan elektronik	Manajemen transaksi elektronik pengetahuan antar individu dalam manajemen pengetahuan terdapat transfer pengetahuan elektronik

2.3.1.4 *Stoplist* / *Stop Word Removal*

Penghapusan *Stopword* merupakan proses penghilangan kata *stopword*. *Stopword* adalah kata - kata yang sering kali muncul dalam dokumen namun arti dari kata kata tersebut tidak deskriptif dan tidak memiliki keterkaitan dengan tema tertentu. Misalnya “di”, ”oleh”, “pada”, ”sebuah”, ”karena” dan lain sebagainya[11].

Tabel 2.4 Contoh *Stopword*

Kalimat	Hasil *Stop Word Removal*
manajemen transaksi elektronik pengetahuan antar individu dalam manajemen pengetahuan terdapat transfer pengetahuan elektronik	manajemen transaksi elektronik pengetahuan individu manajemen transfer

2.3.1.5 Stemming

Menurut Zaman B. dan E Winarko [12] *stemming* adalah proses pemetaan dari penguraian berbagai bentuk kata baik itu *prefix*, *sufix*, maupun gabungan antara *prefix* dan *sufix* (*confix*), menjadi bentuk kata dasarnya. Algoritma *stemmer* yang diperkenalkan Nazief dan Adriani didefinisikan sebagai berikut (13):

1. Di awal proses *stemming* dan setiap langkah yang selanjutnya dilakukan, lakukan pengecekan hasil proses stemming kata yang di-input-kan pada langkah tersebut ke kamus kata dasar. Jika kata ditemukan, berarti kata tersebut sudah berbentuk kata dasar dan proses stemming dihentikan. Jika tidak ditemukan, maka langkah selanjutnya dilakukan.
2. Hilangkan *Inflection Suffixes* (“-lah”, “-kah”, “-ku”, “-mu”, atau “-nya”). Jika berupa *particles* (“-lah”, “-kah”, “-tah” atau “-pun”) maka langkah ini diulangi lagi untuk menghapus *Possesive Pronouns* (“-ku”, “-mu”, atau “-nya”), jika ada.
3. Hapus *Derivation Suffixes* (“-i”, “-an” atau “-kan”). Jika kata ditemukan di kamus, maka algoritma berhenti. Jika tidak maka ke langkah 3a
   a. Jika “-an” telah dihapus dan huruf terakhir dari kata tersebut adalah “-k”, maka “-k” juga ikut dihapus. Jika kata tersebut ditemukan dalam kamus maka algoritma berhenti. Jika tidak ditemukan maka lakukan langkah 3b.
   b. Akhiran yang dihapus (“-i”, “-an” atau “-kan”) dikembalikan, lanjut ke langkah 4.
4. Hilangkan *derivation prefixes*.
   a. Langkah 4 berhenti jika :

      Terjadi kombinasi awalan dan akhiran yang terlarang, awalan yang dideteksi saat ini sama dengan awalan yang dihilangkan sebelumnya, tiga awalan telah dihilangkan.
   b. Identifikasikan tipe awalan dan hilangkan. Awalan terdiri dari dua tipe : Standar (“di-”, “ke-”, “se-”) yang dapat langsung dihilangkan dari

kata, kompleks (“me-”, “be-”, “pe”, “te-”) adalah tipe-tipe awalan yang dapat bermorfologi sesuai kata dasar yang mengikutinya.

c. Cari kata yang telah dihilangkan awalannya ini di dalam kamus kata dasar. Apabila tidak ditemukan, maka langkah 4 diulangi kembali, apabila ditemukan, maka keseluruhan proses dihentikan.

5. Apabila setelah langkah 4 kata dasar masih belum ditemukan, maka proses *recoding* dilakukan dengan mengacu pada aturan pada Tabel 2.1. *Recoding* dilakukan dengan menambahkan karakter *recoding* di awal kata yang dipenggal. Pada Tabel 2.1, karakter *recoding* adalah huruf kecil setelah tanda hubung („-“) dan terkadang berada sebelum tanda kurung. Sebagai contoh, kata “menangkap” (aturan 15), setelah dipenggal menjadi “nangkap”. Karena tidak valid, maka recoding dilakukan dan menghasilkan kata “tangkap”.
6. Jika semua langkah gagal, maka input kata yang diuji pada algoritma ini dianggap sebagai kata dasar.

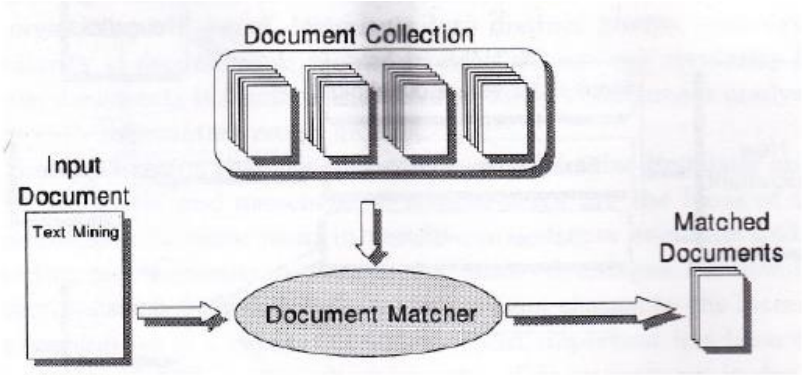
Tabel 2.5 Aturan pemenggalan Awalan Stemmer Nazief dan Adriani

Aturan	Format Kata	Pemenggalan
1	berV...	ber-V... | be-rV...
2	berCAP...	ber-CAP...dimana C!=“r“ & P!=“er“
3	berCAerV...	ber-CaerV... dimana C!=“r“
4	Belajar	bel-ajar
5	beC1erC2...	be-C1erC2... dimana C1!={“r“|“l“}
6	terV...	ter-V... | te-rV...
7	terCerV...	ter-CerV... dimana C!=“r“
8	terCP...	ter-CP... dimana C!=“r“ dan P!=“er“
9	teC1erC2...	te-C1erC2... dimana C1!=“r”
10	me{l|r|w|y}V...	me-{l|r|w|y}V...
11	mem{b|f|v}...	mem-{b|f|v}...
12	mempe{r|l}...	mem-pe...
13	mem{rV|V}...	me-m{rV|V}... | me-p{rV|V}...

14	men{c|d|j|z}...	men-{c|d|j|z}...
15	menV...	me-nV... | me-tV
16	meng{g|h|q}...	meng-{g|h|q}...
17	mengV...	meng-V... | meng-kV...
18	menyV...	meny-sV…
19	mempV...	mem-pV... dimana V!=„e“
20	pe{w|y}V...	pe-{w|y}V...
21	perV...	per-V... | pe-rV...
22	perCAP	per-CAP... dimana C!=“r“danP!=“er“
23	perCAerV...	per-CAerV... dimana C!=“r“
24	pem{b|f|V}...	pem-{b|f|V}...
25	pem{rV|V}...	pe-m{rV|V}... | pe-p{rV|V}...
26	pen{c|d|j|z}...	pen-{c|d|j|z}...
27	penV...	pe-nV... | pe-tV...
28	peng{g|h|q}...	peng-{g|h|q}...
29	pengV...	peng-V... | peng-kV...
30	penyV...	peny-sV…
31	pelV...	pe-lV... kecuali “pelajar” yang menghasilkan “ajar”
32	peCerV...	per-erV... dimana C!={r|w|y|l|m|n}
33	peCP...	pe-CP... dimana C!={r|w|y|l|m|n} dan P!=“er“

2.4. *Information Retrieval*

Information Retrival merupakan pengorganisasian dan pencarian informasi dari sejumlah besar dokumen berbasis teks" [14]. Pencarian informasi dilakukan dengan memberikan query teks pencarian yang diinginkan untuk dicocokkan kedalam kumpulan dokumen sehingga ditemukan dokumen-dokumen yang relevan. Sesuai dengan cara kerja tersebut, Information Retrieval sering dikaitkan dengan search engine. Konsep dasar yang digunakan pada *Information Retrieval* adalah mengukur kesamaan antara dua dokumen, akan dilihar sejauh mana kemiripan yang ada pada dokumen tersebut [15].



Gambar 2. 1 Pengambilan dokumen yang cocok

Pada gambar diatas, dimasukkan dokumen yang akan dicari terhadap koleksi dokumen, dilakuakan proses pencocokan antara dokumen yang akan dicari (dokumen masukan) dengan masing-masing koleksi dokumen, selanjutnya didapatkan dokumen yang sesuai dengan dokumen input pada koleksi dokumen. Tujuan pencarian informasi ini adalah mencari dokumen yang sesuai dengankebutuhan user.

Dalam menghitung performasi hasil pencarian dokumen dapat diukur dengan menghitung variable-varibel tertentu, yaitu:

2.4.1. *Precision*

Precision merupakan sebuah tingkat ketepatan hasil pencarian pada kumpulan dokumen (dataset)[10]. Dalam perhitungannya akan ditentukan seberapa banyak dokumen yang relevan atau sesuai dengan kebutuhan user diantara dokumen yang telah didapatkan pada hasil pencarian. Precision dapat ditentukan dengan rumus [10]:

$$\text{Precision} = \frac{|\{\text{Relevan}\} \cap \{\text{Retrieved}\}|}{|\{\text{Retrieved}\}|} \quad (2.1)$$

Keterangan : {*Relevan* } = kumpulan dokumen yang relevan

{*Retrieved*} = kumpulan dakumen yang didapat atau ditemukan

2.4.2. *Recall*

Recall merupakan tingkat keberhasilan dalam menemukan hasil dokumen yang relevan terhadap jumlah dokumen relevan yang sesungguhnya[10]. Recall dapat dihitung dengan rumus[10]:

$$\text{Recall} = \frac{|\{\text{Relevan}\} \cap \{\text{Retrieved}\}|}{|\{\text{Relevan}\}|} \quad (2.2)$$

Keterangan : {*Relevan* } = kumpulan dokumen yang relevan

{*Retrieved*} = kumpulan dakumen yang didapat atau ditemukan

2.4.3. *F Measure*

Adapun perhitungan yang menggabungkan nilai recall dan precision yaitu *F-Measure* sehingga dapat mewakili kedua nilai tersebut. *F-measure* memiliki rumus [16]:

$$\text{F1} = \frac{2 * \text{precision} * \text{recall}}{precision + recall} \quad (2.3)$$

2.5. *Classifier*

klasifikasi adalah penyusunan bersistem dalam kelompok atau golongan menurut kaidah atau standar yang ditetapkan. Tahapan proses klasifikasi sendiri adalah dengan membagi data latih dan data uji adalah dengan membagi jumlah data yang akan dilakukan klasifikasi data sesuai record yang akan diinputkan.

2.5.1. Algoritma TF-IDF

Metode *Term Frequency-Inverse Document Frequency* (TF-IDF) adalah cara pemberian bobot hubungan suatu kata (term) terhadap dokumen. Untuk dokumen tunggal tiap kalimat dianggap sebagai dokumen. Metode ini menggabungkan dua konsep untuk perhitungan bobot, yaitu *Term frequency* (TF) merupakan frekuensi kemunculan kata (t) pada kalimat (d). Document frequency (DF) adalah banyaknya kalimat dimana suatu kata (t) muncul.

Frekuensi kemunculan kata di dalam dokumen yang diberikan menunjukkan seberapa penting kata itu di dalam dokumen tersebut. Frekuensi dokumen yang mengandung kata tersebut menunjukkan seberapa umum kata tersebut. Bobot kata semakin besar jika sering muncul dalam suatu dokumen dan semakin kecil jika muncul dalam banyak dokumen[11]. Pada aloritma TF-IDF digunakan rumus untuk menghitung bobot (W) masing masing dokumen terhadap kata kunci dengan rumus yaitu :

Wdt = tf dt * IDFt(2.4)

Dimana:

d = dokumen ke-d

t = kata ke-t dari kata kunci

W= bobot dokumen ke-d terhadap kata ke-t

tf = banyaknya kata yang dicari pada sebuah dokumen

IDF = Inversed Document Frequency

IDF = log2 (D/df)

D = total dokumen

df = banyak dokumen yang mengandung kata yang dicari

Setelah bobot (W) masing-masing dokumen diketahui, maka dilakukan proses sorting/pengurutan dimana semakin besar nilai W, semakin besar tingkat similaritas dokumen tersebut terhadap kata kunci, demikian sebaliknya.

Inverse Document Frequency memperhatikan kemunculan *term* pada

kumpulan dokumen. Pada metode ini, term yang dianggap bernilai/berharga adalah term yang jarang muncul pada koleksi/ kumpulan dokumen[20]. Persamaan IDF adalah sebagai berikut:

$$I\text{df}(t) = log\ (N/df(t))+1 \quad (2.5)$$

Dimana df(t) adalah banyak dokumen yang mengandung term t. TF*IDF merupakan kombinasi metode TF dengan metode IDF. Sehingga persamaan TF*IDF adalah sebagai berikut:

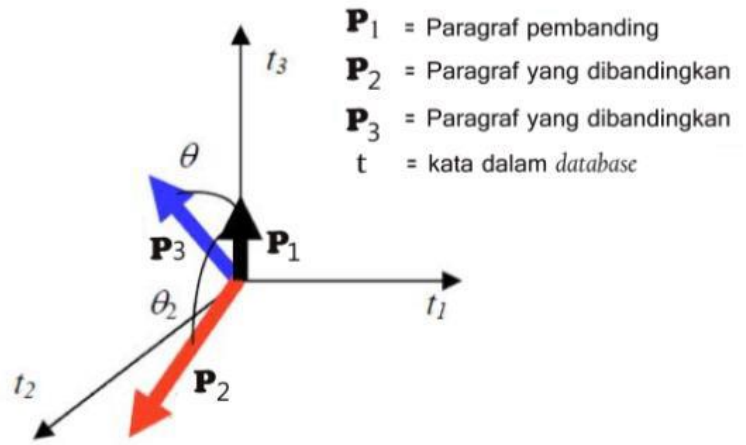
$$TF * I\text{df}(d, t) = TF(d, t) * IDF(t) \quad (2.6)$$

Perhitungan bobot *query relevance* merupakan bobot hasil perbandingan kemiripan (similaritas) antara *query* yang dimasukkan oleh user terhadap keseluruhan kalimat. Sedangkan bobot *similarity* kalimat, merupakan bobot hasil perbandingan kemiripan antar kalimat.

2.5.2. Algoritma *Vector Space Model* (VSM)

Vector Space Model (VSM) adalah metode untuk melihat tingkat kedekatan atau kesamaan (similarity) term dengan cara pembobotan term. Dokumen dipandang sebagi sebuah vektor yang memiliki *magnitude* (jarak) dan direction (arah). Pada *Vector Space Model*, sebuah istilah direpresentasikan dengan sebuah dimensi dari ruang vektor. Relevansi sebuah dokumen ke sebuah query didasarkan pada similaritas diantara vektor dokumen dan vektor query[11].

Vektor Space Model (VSM) digunakan dalam menggambarkan dokumen berbasis teks sebagai vektor dari identifier dimana model ini sering digunakan pada information retrieval [10]. Kalimat yang ada pada dokumen dibagi menjadi term-term dan dibuat vektor dimana keberadaan term dapat di nyatakan 1 bila ada dan 0 bila tidak ada.



Gambar 2. 2 *Vector Space Model*

Dimana : ti : Kata di database

Di : Dokumen

Q : Kata Kunci

Cara kerja dari *vector space model* adalah dengan menghitung nilai cosines sudut dari dua vector, yaitu vektor kata kunci terhadap vektor tiap dokumen. Perhitungan vektor space model menggunakan persamaan (2.7), (2.8) dan (2.9).

$$\cos \theta_{D_i} = Sim(Q, D_i) \qquad (2.7)$$

Dimana : Q : query (kata kunci)

Di : dokumen ke-i

$$Sim(Q, D_i) = \frac{\Sigma_j W_{i,j} W_{q,j}}{\sqrt{\Sigma_j W_{i,j}^2} \sqrt{\Sigma_i W_{q,j}^2}} \quad (2.8)$$

Dimana : Di : dokumen ke-i

Q : query (kata kunci)

J : Kata diseluruh dokumen

$$\cos \theta_{D_i} = \frac{Q - D_i}{|Q| * |D_i|} \quad (2.9)$$

Dimana : Di : Dokumen ke-i

Q : Query (kata kunci)

|Q| : Vektor Q

|Di| : Vektor Di

2.6. Analisi Tersetruktur

Analisis terstruktur akan dibahas tentang pengertian dan perangkat pemodelan analisis terstruktur

a. Pengertian Analisis Tersetruktur

Analisis Terstruktur (Structured Analysis), merupakan salah satu teknik analisis yang mengunakan pendekatan berorientasi fungsi. Analisis ini terfokus pada aliran data dan proses bisnis perangkat dan perangkat lunak. Analisis ini disebut process oriented. Analisis struktur sederhana dalam konsep [19].

Teknik ini mempunyai sekumpulan petunjuk dan perangkat komunikasi grafis yang memungkinkan analis sistem mendefinisikan spesifikasi fungsional perangkat lunak secara terstruktur. Pada metode ini, semua fungsi sistem direpresentasikan sebagai sebuah proses transformasi informasi, dan disusun secara hirarkis sesuai tingkat abstraksinya (sistem maupun perangkat lunak) yang hasilnya ditujukan untuk entitas-entitas eksternal. Prinsip dari teknik ini adalah dekomposisi fungsi dari sistem berdasarkan aliran data dan proses-prosesnya untuk mendapatkan produk analisis yang dapat diubah dan

diperbaiki secara mudah (highly maintainable). Dalam bukunya itu, DeMarco mendefinisikan Analisis Terstruktur sebagai teknik untuk mendeskripsikan spesifikasi sistem baru melalui Data Flow Diagrams, Data Dictionary, Structured English, dan Data Structure Diagrams. Spesifikasi sistem tersebut dinyatakan dalam suatu dokumen yang disebut Spesifikasi Terstuktur (Structured Specification).

b. Perangkata Pemodelan Analisis Tersetruktur

Perangkat Pemodelan Analisis Terstruktur adalah alat bantu pemodelan yang digunakan untuk menggambarkan hasil pelaksanaan Analisis Terstruktur. Perangkat Analisis Terstruktur antara lain adalah:

1. Diagram Konteks

Diagram konteks adalah diagram yang terdiri dari suatu proses dan menggambarkan ruang lingkup suatu sistem. Diagram konteksmerupakan level tertinggi dari DFD yang menggambarkan seluruh input ke sistem atau output dari sistem. Ia akan memberi gambaran tentang keseluruhan sistem. Sistem dibatasi oleh boundary (dapat digambarkan dengan garis putus). Dalam diagram konteks hanya ada satu proses. Tidak boleh ada store dalam diagram konteks. Diagram konteks berisi gambaran umum (secara garis besar) sistem yang akan dibuat. Secara kalimat, dapat dikatakan bahwa diagram konteks ini berisi siapa saja yang memberi data (dan data apa saja) ke sistem, serta kepada siapa saja informasi (dan informasi apa saja) yang harus dihasilkan sistem.

2. Diagram Aliran Data atau Data Flow Diagram (DFD)

Diagram untuk menggambarkan aliran data dalam sistem, sumber dan tujuan data, proses yang mengolah data tersebut, dan tempat penyimpanan datanya. Keuntungan dari diagram arus data adalah memungkinkan pengembangan sistem dari level yang paling tinggi dan

memecah menjadi level yang lebih rendah.Beberapa simbol yang digunakan DFD adalah sebagai berikut :

a. Entitas luar (external entity)

Suatu yang berada diluar sistem, tetapi ia memberikan data kedalam sistem atau memberikan data dari sistem, disimbolkan dengan suatu kotak notasi. External Entity tidak termasuk bagian dari sistem. Bila sistem informasi dirancang untuk suatu bagian lain yang masih terkait menjadi external entity.

b. Proses (Process)

Proses merupakan apa yang dikerjakan oleh sistem. Proses dapat mengolah data atau aliran data masuk menjadi aliran data keluar. Proses berfungsi mentransformasikan suatu atau beberapa data keluaran sesuai dengan spesifikasi yang diinginkan. Setiap proses memiliki satu atau beberapa masukan serta menghasilkan satu atau beberapa data kelurahan. Proses sering juga disebut bubble.

c. Arus data (Data Flow)

Arus data merupakan tempat mengalirnya informasi dan digambarkan dengan garis yang menghubungkan komponen dari sistem. Arus data ditunjukan dengan arah panah dan garis diberi nama atas arus data yang mengalir. Arus data ini mengalir diantara proses, data store dan menunjukan arus data dari data yang berupa masukan untuk sistem atau hasil proses sistem.

d. Simpanan data (Data Store)

Simpanan data merupakan tempat penyimpanaan data yang ada dalam sistem. Data store dapat disimbolkan dengan dua garis sejajar atau dua garis dengan salah satu sisi samping terbuka. Proses dapat mengambil data dari atau memberikan data ke simpanan data (database).

3. Kamus Data (Data Dictionary)

Merupakan suatu tempat penyimpanan (gudang) dari data dan informasi yang dibutuhkan oleh suatu sistem informasi. Kamus data digunakan untuk mendeskripsikan rincian dari aliran data atau informasi yang mengalir dalam sistem, elemen-elemen data, file maupun basis data (tempat penyimpanan) dalam DFD.

4. Diagram Entitas-Relasi atau Entity-Relationship Diagram (ERD)

ERD merupakan notasi grafis dalam pemodelan data konseptual yang mendeskripsikan hubungan antara penyimpanan. ERD digunangan untuk memodelkan struktur data dan hubungan antar data, karena hal ini relative kompleks. Dengan ERD kita dapat menguji model dengan mengabaikan proses yang dilakukan. ERD menggunakan sejumlah notasi dan simbol untuk menggambarkan struktur dan hubungan antar data, pada dasarnya ada 3 macam simbol yang digunakan:

a. Entity

Entity adalah suatu objek yang dapat di identifikasi dalam lingkaran pemakai, sesuatu yang penting bagi pemakai dalam konteks sistem yang akan dibuat.

b. Atribut

Entity mempunyai elemen yang disebut atribut dan berfungsi mendeskripsikan karakter entity.

c. Relasi

Relasi merupakan gugusan entitas yang berhubungan antar entitas atau beberapa entitas.

2.7. Bahasa pemrograman yang di gunakan

Dalam pembuatan aplikasi klasifikasi berita otomatis, menggunakan bahas pemrograman C#.

2.7.1 Bahasa Program C#

C# (dibaca: C sharp) merupakan sebuah bahasa pemrograman yang berorientasi objek yang dikembangkan oleh Microsoft sebagai bagian dari inisiatif kerangka .NET Framework. Bahasa pemrograman ini dibuat berbasiskan bahasa C++ yang telah dipengaruhi oleh aspek-aspek ataupun fitur bahasa yang terdapat pada bahasa-bahasa pemrograman lainnya seperti Java, Delphi, Visual Basic, dan lain-lain) dengan beberapa penyederhanaan. Menurut standar ECMA-334 C# Language Specification, nama C# terdiri atas sebuah huruf Latin C (U+0043) yang diikuti oleh tanda pagar yang menandakan angka # (U+0023). Tanda pagar # C# (dibaca: C sharp) merupakan sebuah bahasa pemrograman yang berorientasi objek yang dikembangkan oleh Microsoft sebagai bagian dari inisiatif kerangka .NET Framework. Bahasa pemrograman ini dibuat berbasiskan bahasa C++ yang telah dipengaruhi oleh aspek-aspek ataupun fitur bahasa yang terdapat pada bahasa-bahasa pemrograman lainnya seperti Java, Delphi, Visual Basic, dan lain-lain) dengan beberapa penyederhanaan. Menurut standar ECMA-334 C# Language Specification, nama C# terdiri atas sebuah huruf Latin C (U+0043) yang diikuti oleh tanda pagar yang menandakan angka # (U+0023). Tanda pagar # yang digunakan memang bukan tanda kres dalam seni musik (U+266F), dan tanda pagar # (U+0023) tersebut digunakan karena karakter kres dalam seni musik tidak terdapat di dalam keyboard standar.

C# kadang-kadang dapat disebutkan sebagai bahasa pemrograman yang paling mencerminkan dasar dari CLR dimana semua program-program .NET berjalan, dan bahasa ini sangat bergantung pada kerangka tersebut sebab ia secara spesifik didisain untuk mengambil manfaat dari fitur-fitur yang tersedia pada CLR.

2.8. *Tools* Yang Digunakan

Dalam pembuatan aplikasi klasifikasi berita otomatis, menggunakan Microsoft Visual Studio 2010 sebagai pembuatan aplikasi dan menggunakan tools DevExpress Universal sebagai editor tampilan pada aplikasi yang akan dibuat.

2.9.1 *Microsoft Visual Studio*

Microsoft Visual Studio merupakan sebuah perangkat lunak yang dapat digunakan untuk melakukan pembuatan maupun pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dengan bentuk aplikasi console, aplikasi windows, ataupun aplikasi web. Visual Studio mencakup kompiler, SDK, Integrated Development Environment (IDE), dan dokumentasi (umumnya berupa MSDN Library). Kompiler yang dimasukkan ke dalam paket Visual Studio antara lain adalah : Visual C++, Visual C#, Visual Basic, Visual Basic .NET, Visual InterDev, Visual J++, Visual J#, Visual FoxPro, dan Visual SourceSafe.



Gambar 2. 3 Logo Microsoft Visual Studio