BAB 2
LANDASAN TEORI

2.1 Analisis Sentimen

Analisis sentimen merupakan bidang studi mengenai analisa suatu pendapat, opini, penilaian, sentimen, maupun emosi terhadap entitas tertentu seperti produk, layanan, organisasi, topik, dan sebagainya [10]. Tujuan utama dalam analisis sentimen yaitu untuk mengelompokkan polaritas dari teks yang ada dalam dokumen, kalimat, atau pendapat mengarah ke arah yang positif atau negatif [11].

Analisis Sentimen bisa dikatakan merupakan suatu teknik untuk mengolah dan menggali suatu data yang bentuknya teks untuk mengetahui informasi yang terkandung didalamnya. Dengan melakukan analisis sentimen, maka informasi tersebut dapat didapatkan dengan cepat dan tepat [8]. Hasil dari analisis sentimen dapat digunakan di berbagai bidang, salah satunya adalah pada bidang bisnis. Pendapat dari seseorang terhadap suatu produk bisa menjadi sumber informasi untuk mengembangkan produknya. Dengan mengkategorikan suatu teks, maka bisa dipisahkan antara sentimen positif dan negatif dari masyarakat, dengan ini pembisnis dapat mengembangkan usahanya [2].

Secara umum analisis sentimen terbagi kedalam tiga level diantaranya [10]:

1. Level Dokumen (*Document Level*)
   Pada level ini ditujukan untuk mengklasifikasikan suatu dokumen secara keseluruhan apakah tergolong ke positif atau negatif.
2. Level Kalimat *(Sentence Level)*
   Pada level ini ditujukan untuk mengklasifikasikan setiap kalimat cenderung ke arah positif, negatif atau netral.
3. Level Aspek *(Entity and Aspect level)*
   Sama halnya dengan level kalimat dengan tambahan pada aspek apa sentimen tersebut ditujukan.

Analisis sentimen ini sejatinya termasuk ke ranah *text mining* maupun *natural language processing* yang hingga pada saat ini penerapannya, sudah mulai menerapkan pola *deep learning* yang setiap waktu terus berkembang, yang diharapkan seiring perkembangannya akan menangani hal yang belum terjawab [12]. Namun pada penelitian masih akan menerapkan *machine learning* pada analisis sentimen

2.2 Analisis Sentimen Berbasis Aspek

Analisis sentimen level aspek atau *aspect based sentiment analysis* (*ABSA*) secara umum dapat diartikan dengan proses menganalisa suatu teks melihat kecenderungannya kearah positif atau negatif, dimana bukan hanya menentukan bagaimana sentimennya, tetapi juga pada aspek apa sentimen tersebut ditujukan [2]. Melakukan analisis sentimen hanya pada level kalimat sering kali dirasa kurang lengkap karena hasilnya tidak mengidentifikasi target dari opini tersebut. Maka dari itu, analisis sentimen berbasis aspek bertujuan untuk mendeteksi polaritas teks berdasarkan pada aspek tertentu.

Secara sederhana analisis sentimen berbasis aspek bekerja dengan beberapa tahap, namun diantara tahapnya hal yang terpenting dan harus ada yaitu [9] :

1. *Aspect extraction*

   Pada tahap ini mengekstrak atau mengenali aspek yang telah ditentukan sebelumnya. Sebagai contoh pada kalimat “tempatnya nyaman jadi betah nongkrong disini”. Pada kalimat tersebut aspeknya adalah “tempat” karena pada kalimat tersebut yang dibicarakan adalah tempat tersebut.
2. *Aspect sentiment classification*

   Pada tahap ini menentukan apakah pendapat dari berbagai aspek kedalam sentimen positif, atau negatif. Sebagai contoh pada kalimat yang sama dengan yang sebrlumnya “tempatnya nyaman jadi betah nongkrong disini”, pada kalimat tersebut dapat ditarik hasil bahwa aspeknya adalah “tempat” dan sentimennya adalah “positif”.

Pada tahap mengekstrak atau mengidentifikasi aspek maupun klasifikasi sentimen dapat dilakukan salah satunya dengan menggunakan pendekatan *machine learning* yaitu *supervised learning*.

2.3 Machine Learning

Machine Learning atau yang lebih dikenal dengan pembelajaran mesin dapat didefinisikan sebuah metode agar sebuah komputer dapat dengan sukses memprediksi menggunakan pegalaman yang sebelumnya. Pembelajaran mesin membangun model komputasi secara otomatis dengan memproses data yang latih. Lalu melakukan proses prediksi, klasifikasi, kluster pada data uji dengan hasil pelatihan dari data latih [13].

Menurut Teguh Wahyono secara umum *machine learning* adalah salah satu cabang dari kecerdasan buatan, terutama khusus yang mempelajari mengenai bagaimana komputer mampu belajar dari data yang ada untuk meningkatkan kecerdasannya [14].

Menurut teknik pembelajaannya, secara umum machine learning dibagi menjadi *supervised learning* dan *unsupervised learning* [15]

1. *Supervised Learning*

   Supervised Learning atau pembelajaran dengan pengawasan yaitu proses pembelajaran yang membutuhkan atau memerlukan guru. Yang dimaksud dengan guru disini adalah sesuatu yang memiliki pengetahuan. Dengan demikian teknik ini memerlukan data latih yang sudah berlabel.

   Tujuan dari teknik ini adalah dapat mengidentifikasi labael dari data input yang baru (data uji) baik itu melakukan klasifikasi maupun prediksi.
2. *Unsupervised Learning*

   Unsupervised Learning atau pembelajaran tanpa pengawasan yaitu proses pembelajaran yang tidak membutuhkan atau memerlukan guru untuk

mengawasi proses belajar. Teknik ini tidak memerlukan data latih berlabel untuk melakukan pembelajaran.

Teknik ini tidak memiliki kelas target dan bertujuan untuk mengelompokan suatu objek yang memiliki suatu fitur yang hampir sama atau dalam area tertentu.

2.4 Modified K-Nearest Neighbor (MKNN)

Metode *Modified K-Nearest Neighbor* (*MKNN*) adalah pengembangan dari algoritma *K-Nearest Neighbor*, dengan penambahan beberapa tahapan proses yaitu perhitungan nilai validitas dan perhitungan bobot (*Weight Voting*). Kedua tambahan tahapan ini berpengaruh pada hasil klasifikasi yang akan dihasilkan. Pada penelitiannya Fakihatin kedua tambahan tersebut disebutkan dapat mengatasi kelemahan dari data yang memiliki masalah dalam *outlier* atau pencilan [6].

Perhitungan nilai validitas pada data latih menggunakan persamaan (2.1) sebagai berikut :

$$Validity(x) = \frac{1}{K}\sum_{i=0}^{n} S(lbl(x), lbl(Ni(x))) \qquad (2.1)$$

Dimana

K : jumlah tetangga terdekat

$lbl(x)$: kelas x

$lbl(Ni(x))$: kelas selain x

Fungsi dari S digunakan untuk mengkalkulasi kesamaan antara titik x dan data ke-i dari tetangga terdekat, yang dituliskan dengan persamaan (2.2) berikut :

$$S(a,b) = \begin{cases} 1\ jika\ a = b \\ 0\ jika\ a \neq b \end{cases} \qquad (2.2)$$

Dimana :

a : kelas a pada data traing

b : kelas lain selain daripada a pada data tetangga terdekat

Sedangkan untuk perhitungan bobot atau *weight voting* menggunakan persamaan (2.3) berikut :

$$W(i) = Validity(i) \times \frac{1}{d_e + 0.5} \quad (2.3)$$

Dimana :

W(i) : nilai *weight voting* ke-i

Validity(i) : nilai validitas ke-i

d_e : jarak euclidean

Dengan adanya dua tahapan tersebut, maka dapat meningkatkan akurasi dari metode sebelumnya. Sehingga algoritma *MKNN* ini bisa dikatakan lebih kuat dari metode *K-NN* tradisional yang hanya didasarkan pada jarak [16].

2.5 Euclidean Distance

Pengukuran jarak memegang peran yang amat penting juga dalam melihat dan menentukan kemiripan antar data. Pengukuran jarak dilakukan untuk mengetahui dengan cara yang sedemikian rupa bahwa suatu data bersifat saling terkait, mirip, ataupun tidak mirip.

Euclidean Distance merupakan salah satu cara untuk menghitung jarak skalar jauh atau dekatnya suatu data dengan data yang lain. *Euclidean Distance* melakukan perhitungan jarak dari 2 (dua) buah titik yang berada dalam *Euclidean space* (baik bidang *euclidean* dua dimensi, tiga dimensi, dan seterusnya). Semakin kecil jarak euclidean antara dua buah data maka dapat disimpulkan semakin mirip pula kedua data tersebut.

Persamaan untuk menghitung jarak dengan *Euclidean Distance* menggunakan persamaan (2.4) berikut :

$$D(a,b) = \sqrt{\sum_{k=1}^{d} (a_k - b_k)^2} \quad (2.4)$$

Dimana :

D(a,b) : jarak skalar antara a dan b

d : ukuran dimensi

a : ciri pada data a

b : ciri pada data b

Pada penelitian lain yang melakukan perbandingan antara perhitungan *euclidean distance* dengan *manhattan distance* dan *minkowski distance* dalam hal melakukan klastering dengan menerapkan algoritma *K-means*, menyebutkan bahwa ketiganya memiliki tingkat akurasi yang tinggi, dengan *euclidean distance* memiliki nilai paling tinggi. Hasil tersebut yaitu 84.47% (untuk *euclidean distance*), 83.85% (untuk *manhattan distance*), dan 83.85% (untuk metode *minkowski*). Dengan demikian, bisa disimpulkan bahwa metode *euclidean* merupakan metode terbaik untuk diterapkan dalam algoritma *K-Means Clustering* [17].

2.6 Web Scrapping

Web Scraping merupakan suatu teknik yang dilakukan untuk mendapatkan beberapa informasi dari suatu situs web dengan otomatis. Tujuan utama dari *web scrapping* adalah untuk mencari dan mengumpulkan beberapa jenis informasi tertentu, mengekstrak, serta menggabungkannya. Informasi yang didapatkan tersebut dapat digunakan untuk tujuan atau maksud tertentu [18].

2.7 Preprocessing

Preprocessing merupakan beberapa tahapan permulaan yang dilakukan sebelum suatu algoritma dijalankan. Banyak faktor yang memengaruhi keberhasilan *Machine Learning (ML)* pada tugas tertentu. Tahapan *preprocessing* atau praproses ini biasanya berdampak yang signifikan pada performa suatu algoritma terutama yang sifatnya *supervised learning*. Diharapkan dengan dilakukannya tahapan praproses ini data menjadi lebih terstruktur dan siap untuk dilakukan proses selajutnya [19].

a) Case folding

Tahapan yang berfungsi untuk mengubah semua huruf besar menjadi huruf kecil. Seluruh huruf yang ada pada data termasuk huruf awal kalimat akan berubah menjadi huruf kecil. Proses *Case folding* ini juga dikenal dengan konsep *lowercase.* Berikut salah satu contoh dari proses *case folding* terlihat pada Tabel 2.1 berikut :

Tabel 2.1 Contoh Proses Case folding

Sebelum *case folding*	Setelah *case foldiing*
Harga yang murah untuk kantong Mahasiswa :) rekomended pokoknya.	harga yang murah untuk kantong mahasiswa :) rekomended pokoknya..

b) Normalisasi

Merupakan tahapan yang berfungsi untuk melakukan normalisasi pada kata yang bersifat bahasa asing atau bahasa yang tidak baku menjadi baku dalam bahasa indonesia. Pada tahap ini juga dilakukan proses untuk mengubah emotikon menjadi kata.

Untuk tahapan proses normalisasi pada penelitian ini akan menggunakan pendekatan berbasis aturan atau *rule based*. Bekerja dengan melakukan pemeriksaan pada data, jika pada data terdapat kata yang terdapat pada kamus maka kata tersebut akan diubah terlebih dahulu ke bentuk normalnya (baik bahasa asing, kata tidak baku maupun emotikon). Untuk kamus dibuat sendiri disusun sedemikian rupa menyesuaikan dengan data yang ada. Untuk isi kamus bahasa bersumber dari google translate. Sedangkan untuk kamus data baku bersumber dari situs daring Kamus Besar Bahasa Indonesia (KBBI).

Pada penelitian sebelumnya didapat kamus yang berisi berbagai jenis emotikon yang umum digunakan dalam bahasa indonesia untuk di ubah ke dalam bentuk kata. Jumlah emotikon tersebut adalah sembilan belas

emotikon [20] . Namun pada penelitian kali ini, berdasarkan data yang ada, maka ada sedikit tambahan emotikon pada kamus tersebut.

Berikut merupakan Tabel 2.2 merupakan proses normalisasi emotikon yang diusulkan :

Tabel 2.2 Tabel Normalisasi emotikon

Emotikon	Hasil normalisasi
:) , :)) , :-) , :-)) , =) , =)) , 8) , 8))	esenyum
:D , :-D , =D	etawa
:-(, :(	esedih
;-) , ;)	ekedip
:-P , :P	eejek
:-/ , :/ , -_-	eragu
:| , :-|	eharu
:v , :-v	ekesal

Berikut merupakan salah satu contoh dari proses normalisasi pada salah satu data yang ada terlihat pada Tabel 2.3 brikut:

Tabel 2.3 Contoh proses normalisasi

Sebelum normalisasi	Setelah normalisasi
harga yang murah untuk kantong mahasiswa :) rekomended pokoknya.	harga yang murah untuk kantong mahasiswa esenyum rekomendasi pokoknya.

c) Filtering

Tahapan yang berfungsi untuk melakukan penyaringan data dari karakter, simbol, angka selain huruf yang tidak diperlukan. Cara kerja proses *filtering* yaitu mengecek data yang ada, jika mengandung karakter simbol ataupun angka maka akan dihapus. Berikut salah satu contoh dari proses *filtering* terlihat pada Tabel 2.4 berikut :

Tabel 2.4 Contoh Proses *Filtering*

Sebelum *filtering*	Setelah *filtering*
harga yang murah untuk kantong mahasiswa esenyum rekomendasi pokoknya.	harga yang murah untuk kantong mahasiswa esenyum rekomendasi pokoknya

d) Tokenizing

Tokenizing merupakan proses pengambilan kata yang menjadi penyusun suatu kalimat. Karakter-karakter pemisah tersebut bisa berupa spasi, tanda baca, angka, dan karakter selain huruf. Berikut data setelah dilakukan proses *tokenizing* terlihat pada Tabel 2.5 berikut :

Tabel 2.5 Contoh *Proses Tokenizing*

Sebelum *Tokenizing*	Setelah *Tokenizing*
harga yang murah untuk kantong mahasiswa esenyum rekomendasi pokoknya	[“harga”, ”yang”, ”murah”, ”untuk”, “kantong”, “mahasiswa”, “esenyum”, “rekomendasi”, “pokoknya”]

e) Stemming

Tahapan *Stemming* merupakan suatu tahapan yang dilakukan untuk membuat kata yang berimbuhan menjadi ke kata dasarnya. Suatu aturan bahasa diterapkan untuk menghilangkan tersebut. Untuk melakukan proses *stemming*, pada penelitian ini akan menggunakan *library Sastrawi*. Sastrawi merupakan suatu *library stemmer* sederhana yang dibuat untuk dapat digunakan dengan mudah.

Pada *library Sastrawi* proses stemming dilakukan dengan menggunakan kamus kata dasar yang berasal dari kateglo.com. Satrawi

stemmer menerapkan algoritma Nazief dan Adriani, dengan beberapa peningkaan menggunakan Algoritma *CS (Confix Stripping), ECS (Enhanced Confix Stripping)* serta *Modified ECS*. [21]

Secara garis besar cara kerja dari algoritma Nazief dan Adriani adalah sebagai berikut [22] :

1. Mencari kata pada kamus, jika ditemukan, kata tersebut dianggap sebagai kata dasar yang benar, lalu algoritma dihentikan.
2. Menghilangkan *Inflection Suffixes* diantaranya (“-lah”, “-kah”, “-ku”, “-mu”, atau “-nya”). Apabila ada kata berupa *particles* (“-lah”, “-kah”, “-tah” atau “-pun”), maka ulangi langkah ini untuk menghapus *Possesive Pronouns* (“-ku”, “-mu”, atau “-nya”).
3. Menghapus *Derivational Suffix* (“-i” atau ”-an”,”). Apabila kata ada pada kamus kata dasar, maka algoritma berhenti. Jika tidak, maka lanjut ke langkah 3a:

   a. Apabila akhiran “-an” telah dihapus dan huruf terakhir dari kata tersebut adalah “-k”, maka “-k” juga dihapus. Apabila kata terdapat dalam kamus, algoritma berhenti. Namun ika tidak, maka lakukan langkah 3b.

   b. Akhiran yang dihapus (“-i”, “- an” atau “-kan”) dikembalikan, lanjut ke langkah 4.
4. Hapus *Derivational Prefix* (“be-”,”di-”,”ke-”,”me-”,”pe-“,”se-” dan “te-“). Apabila kata yang didapat ditemukan didalam database kata dasar, maka proses dihentikan, jika tidak, maka lakukan *recoding.*
5. Apabila semua langkah telah dilakukan tetapi kata dasar tersebut tidak ditemukan pada kamus, maka algoritma ini mengembalikan kata yang asli sebelum dilakukan stemming.

Berikut salah satu contoh dari proses *stemming* terlihat pada Tabel 2.6 berikut.

Tabel 2.6 Contoh *Proses Stemming*

Sebelum *stemming*	Setelah *stemming*
[“harga”, ”yang”, ”murah”, ”untuk”, “kantong”, “mahasiswa”, “esenyum”. “rekomendasi”, “pokoknya”]	[“harga”, ”yang”, ”murah”, ”untuk”, “kantong”, “mahasiswa”, “esenyum”. “rekomendasi”, “pokok”]

f) Stopword Removal

Tahapan *Stopword removal* merupakan tahap untuk menghilangkan kata kata yang kurang penting atau tidak relevan. Proses *Stopword removal* pada penelitian ini akan menggunakan *library Sastrawi*. Dimana kamus data *stopwordlist* bersumber dari kamus bawaan dari *library sastrawi* itu sendiri. Cara kerja *stopword removal* adalah dengan mengecek setiap data, jika pada data tersebut mengandung kata yang ada pada *stopwordlist* maka kata tersebut akan dihilangkan, jika tidak maka kata akan tetap dan tidak dihilangkan Berikut salah satu contoh dari proses *stopword removal* terlihat pada tabel 2.7 berikut.

Tabel 2.7 Contoh Proses *Stopword Removal*

Sebelum *stopword removal*	*Setelah stopword removal*
[“harga”, ”yang”, ”murah”, ”untuk”, “kantong”, “mahasiswa”, “esenyum”. “rekomendasi”, “pokok”]	[“harga”, ,”yang”, ”murah”, “kantong”, “mahasiswa”, “esenyum”. “rekomendasi”, “pokok”]

g) *TF-IDF* (*Term Frequency Inverse Document Frequency*)

Teks representasi merupakan prosedur yang penting dan utama dalam melakukan klasifikasi teks. Salah satunya yang paling banyak digunakan adalah *TF-IDF,* karena bisa dikataka metode ini cukup mudah dipahami. Secara kualitas statistik *TF-IDF* sudah bisa dikatakan cukup baik.

Pembobotan kata dengan *TF-IDF* ditujukan untuk mengeksrak atau mengenali fitur atau ciri yang ada pada data [23].

Perhitungan *TF* (*Term Frequency*) menunjukan jumlah munculnya pada suatu dokumen/teks, sedangkan *DF* (*Document Frequency*) merupakan jumlah dari suatu dokumen/teks yang memiliki/berisi suatu *term*. Persamaan *TF-IDF* dapat dirumuskan dengan persamaan 2.4 berikut :

$$W_{(i,j)} = tf_{(i,j)} * IDF_j \quad (2.4)$$

Dimana :

$W_{(i,j)}$: bobot kata *j* terhadap dokumen *i*

$tf_{(i,j)}$: banyaknya kemunculan kata *j* pada dokumen *i*

IDF_j : *Inversed Document Frequency*

Sedangkan untuk *IDF* nya sendiri memiliki persamaannya tersendiri persamaan 2.5 yaitu :

$$IDF_j = \log\left(\frac{D}{DF_j}\right) \quad (2.5)$$

Dimana :

IDF_j : *Inversed Document Frequency*

D : Jumlah Dokumen

DF_j : jumlah dokumen yang berisi *term j*

Pada persamaan 2.5 jika nilai *D = DFj* maka hasil dari *log* 1 akan menghasilkan 0, maka dari itu, nilai 1 akan ditambahkan pada nilai IDF. Pada penelitian ini proses perhitungan pembobotan kata *TF – IDF* juga akan memanfaatkan *library* dari *sklearn*. *Sklearn* merupakan suatu *library* yang menyediakan pustaka pembelajaran mesin yang bersifat terbuka *(open source)* untuk bahasa pemograman python.

Perhitungan *IDF* pada *library sklearn* ini menggunakan logaritma natural, sehingga perhitungan *IDF* pada penelirian kali ini akan menggunakan persamaan 2.6 berikut [24] :

$$IDF_j = \ln\left(\frac{D}{DF_j}\right) + 1 \qquad (2,6)$$

Dimana :

IDF_j : *Inversed Document Frequency*

D : Jumlah Dokumen

DF_j : jumlah dokumen yang berisi *term j*

2.8 Python

Python merupakan salah satu bahasa pemograman yang cukup populer di kalangan para *programmer*, tertutama bagi programmer yang menggeluti di bidang *machine learning*. Bahasa pemograman python ini bersifat *open source*, dimana sebagian besar dari keseluruhannya telah menggunakan lisensi *GFL-compatible*, python telah berpindah-pindah/ migrasi dari satu perusahaan ke perusahaan lain.

Python muncul tidak lebih dulu dibandingkan dengan bahasa pemograman lainya, kendati demikian sejatinya python telah berkembang dengan cukup pesat dengan berbagai keunggulannya seperti [14] :

1. Keterbacaan atau *readibility*

   Bahasa pemograman python tergolong sederhana, mudah dipahami, dan diingat, sehingga mempermudah pengembang aplikasi.
2. Efisien

   Dengan memiliki aspek keterbacaan, memberikan efisiensi pada programmer dalammembuat program.
3. Multifungsi

   Bahasa python dapat digunakan untuk pembuatan web, robotik, serta aplikasi dengan kecerdasan buatan.
4. Interoperabilitas

   Bahasa pemograman python dapat berinteraksi dengan bahasa pemograman lainnya.

5. Dukungan Komunitas

   Python memiliku ruang lingkup komunitas yang cukup luas, sehingga mempermudah pengguna *share* untuk berbagi tentang bahasa ini.

2.9 Pengukuran Akurasi

Salah satu alat yang dapat sebagai ukuran performansi adalah *confusion matrix. Confusion matrix* berisi ringkasan kinerja klasifikasi dari suatu sistem *classifier* dengan menggunakan sejumlah data. *Confusion matrix* berbentuk seperti tabel yang mencakup *matrix* dua dimensi, dimana dimensi yang pertama diisi oleh kelas sebenarnya dari suatu objek dan di dimensi yang lain diisi oleh kelas yang dihasilkan oleh *classifier*. *Confusion matrix* yang biasa digunakan, biasanya menggunakan 4 kombinasi nilai terlihat pada Tabel 2.8 berikut [25] .

Tabel 2.8 Tabel *Confusion Matrix*

		Kelas Prediksi	
		Positive	*Negative*
Kelas Aktual	*Positive*	**TP**	**FN**
	Negative	**FP**	**TN**

Dengan keterangan :

TP = *True Positive* atau jumlah tupel positif yang dilabeli dengan benar oleh *clasiffier*.

TN = *True Negative* atau jumlah tupel negatif yang dilabeli dengan benar oleh *clasiffier*.

FP = *False Positive* atau jumlah tupel positif yang salah dilabeli oleh *clasiffier*.

FN = *False Negative* atau jumlah tuple negatif yang salah dilabeli oleh *clasiffier*.

Sedangkan untuk mengukur dari segi akurasi itu sendiri menggunakan persamaan 2.7 yaitu :

$$accuracy = \frac{(TP+TN)}{(TP+TN+FN+FN)} \quad (2.7)$$

2.10 Pemodelan

Pemodelan merupakan suatu proses yang dilakukan untuk memodelkan suatu sistem yang akan dibangun, dimana bagian - bagian dari sistem tersebut akan dipresentasikan kedalam bentuk atau gambaran tertentu. Konsep pemodelan yang akan digunakan dalam penelitian kali ini bersifat pemograman terstruktur, yang dimana konsep yang akan digunakan tersebut diantaranya *flowchart*, diagram konteks dan *data flow diagram (DFD).*

2.10.1 Flowchart

Flowchart merupakan suatu alat untuk memetakan program yang menunjukkan alur atau urutan tindakan dalam suatu proses dalam bentuk yang mudah dibaca, dipahami dan dikomunikasikan. *Flowchart* pada sistem menggambarkan secara grafik dari runtunan, dan prosedur pada suatu program. *Flowchart* ini berguna untuk memudahkan dalam penyelesaian suatu masalah, khususnya masalah yang perlu dipahami dan dievaluasi lebih lanjut. Flowchart ini dibedakan menjadi beberapa jenis yaitu *flowchart* sistem, *flowchart* dokumen, *flowchart* skematik, *flowchart* program dan *flowchart* proses . Dimana setiap jenis dari *flowchart* tersebut memiliki fungsi dan kegunaan masing – masing. [26]

2.10.2 Diagram Konteks

Diagram konteks merupakan tingkatan tertinggi dalam suatu diagram aliran data yang menggambarkan hubungan sistem dengan lingkungan luarnya. Diagram konteks ini hanya memuat satu proses yang menunjukkan sistem secara keseluruhan baik alur data masukan maupun keluaran dari sistem. [27]

2.10.3 Data Flow Diagram

Data Flow Diagram (DFD) adalah metode analisis dan desain terstruktur. *DFD* ini merupakan alat visual untuk menggambarkan model logika dan mengekspresikan transformasi data dalam suatu sistem. *DFD* mencakup mekanisme yang memodelkan arus data atau aliran data. *DFD* ini merupakan alat perancangan sistem yang berorientasi pada alur data yang dapat digunakan untuk gambaran analisa maupun rancangan sistem yang mudah dikomunikasikan oleh profesional sistem kepada pengguna maupun *programmer*. [27]