BAB 2
TINJAUAN PUSTAKA

2.1 Ekstraksi Informasi

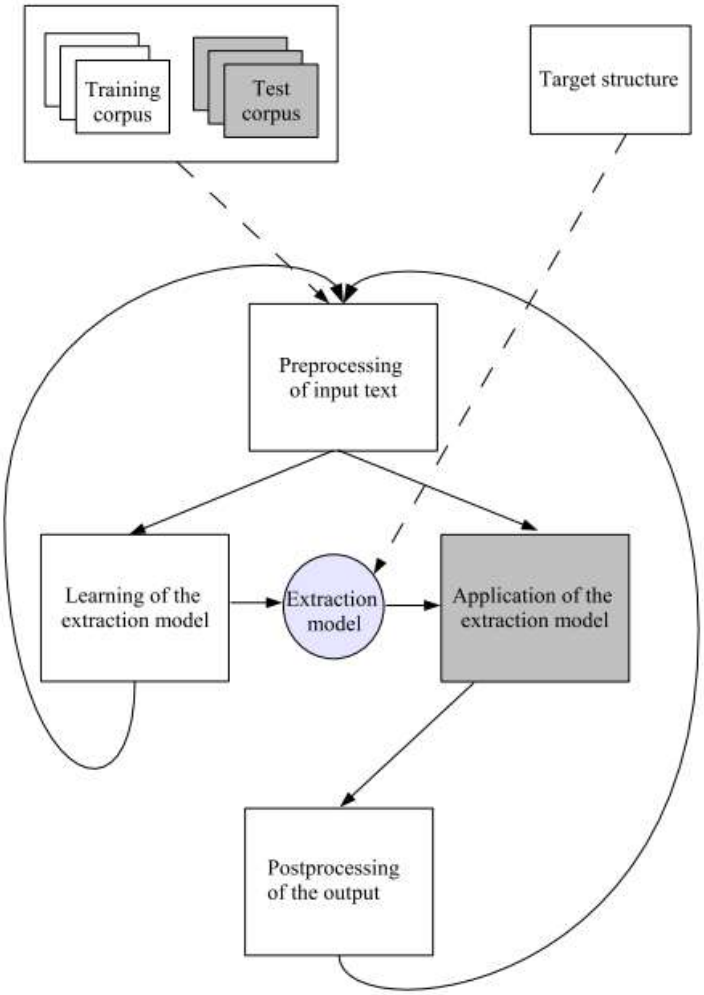
Teknologi ekstraksi informasi (*Information Extraction*) adalah teknologi yang berkaitan dengan cara menjadikan dokumen teks tidak terstruktur dengan domain tertentu ke dalam sebuah struktur informasi yang relevan. Dengan kata lain, tujuan utama dari IE adalah mencari informasi-informasi yang relevan dengan domain dan tidak memperdulikan informasi tidak relevan [9]. Secara garis besar, proses ekstraksi informasi terdiri dari dua tahap, yaitu mengidentifikasi data yang relevan, kemudian menyimpannya ke dalam bentuk terstruktur untuk digunakan kemudian [10].

Sebagai contoh, dilakukan proses ekstraksi informasi terhadap sebuah email mengenai undangan untuk menghadiri kuliah. Seperti yang dapat dilihat pada gambar 2.1, hasil dari proses ekstraksi informasi tersebut adalah informasi mengenai pembicara, tempat dan waktu diadakannya kuliah tersebut. Riset dan pengembangan dari IE sebagian besar termotivasi karena adanya *Message Understanding Conferences* (MUC) dan *Automatic Content Extraction* (ACE) [11].

Gambar 2.1 Contoh Proses Ekstraksi Informasi

Menurut Christian Siefkes terdapat 3 pendekatan pembelajaran mesin untuk ekstraksi informasi berdasarkan klasifikasi yaitu *rule learning, knowledge-based* dan *statistical* [11]. Pada penelitian ini menggunakan pendekatan *statistical* dengan metode *maximum entropy markov model*.

Algoritma mempelajari aturan-aturan ekstraksi berdasarkan dokumen teks sebagai data latih yang telah diberi anotasi mengenai entitas informasi yang akan diekstrak. Peran manusia diperlukan untuk memberikan label atau anotasi sesuai entitas yang akan diekstrak pada dokumen. Berikut ini alur proses secara umum sistem ekstraksi informasi dengan algoritma pembelajaran mesin pada gambar 2.2.



Gambar 2.2 Alur Proses Sistem Ekstraksi Informasi Secara Umum

Alur proses dari gambar 2.2 tersebut [10]:

1. *Preprocessing* data masukan

Data masukan berupa teks yang tidak terstruktur, *natural language text*. Informasi penting didapatkan dengan analisis linguistik, karena menghasilkan kata kunci dan fitur ciri penting untuk mengidentifikasi informasi. Analisis linguistik yang digunakan diantaranya tokenisasi, pembagian kalimat (*sentence splitting*), analisis morpologi, *parsing* dan *named entity*. Pada penelitian ini menggunakan tokenisasi dan ekstraksi fitur termasuk *named entity*.

1.1.Tokenisasi

Keadaan awal dalam bentuk karakter yang terhubung dengan tujuan untuk mengidentifikasi bagian dasar dari *natural language* seperti kata, tanda baca dan pemisah. Hasil dari token yang memiliki makna dan terhubung sebagai dasar untuk proses linguistik dan teks berikutnya.

1.2.Ekstraksi Fitur

Menurut Prihatini, ekstraksi fitur merupakan proses untuk mencari nilai - nilai fitur yang terkandung dalam dokumen [12]. Fitur dapat diartikan sebagai ciri dari setiap data yang dikenali oleh sistem sehingga menghasilkan nilai fitur. Ekstraksi fitur merupakan topik penting dalam klasifikasi, karena fitur-fitur yang baik akan sanggup meningkatkan tingkat akurasi, sementara fitur-fitur yang tidak baik cenderung memperburuk tingkat akurasi [13].

Ada 3 kelompok fitur yang akan digunakan dengan total 15 fitur, yaitu fitur lokal, fitur tata letak dan *named entity*. Fitur lokal adalah karakteristik yang terdapat dalam karakter setiap baris kalimat 7 fitur. Fitur tata letak adalah posisi suatu baris kalimat dalam bagian dokumen 3 fitur. Fitur *named enity* adalah fitur yang diekstrak dari dokumen berdasarkan aturan tertentu 3 fitur [14] sedangkan untuk fitur LOWERCASE dan EIGHTDIGIT merupakan fitur yang ditentukan oleh peneliti.

2. Pembelajaran/pelatihan dan aplikasi model ekstraksi informasi

Algoritma pembelajaran digunakan untuk membentuk model ekstraksi informasi berdasarkan hasil pelatihan data masukan dan penentuan struktur awal seperti anotasi atau label yang telah ditentukan. Hasil pelatihan diaplikasikan ke data pengujian untuk menghasilkan informasi sesuai dengan struktur awal yang ditentukan.

3. *Postprocessing*

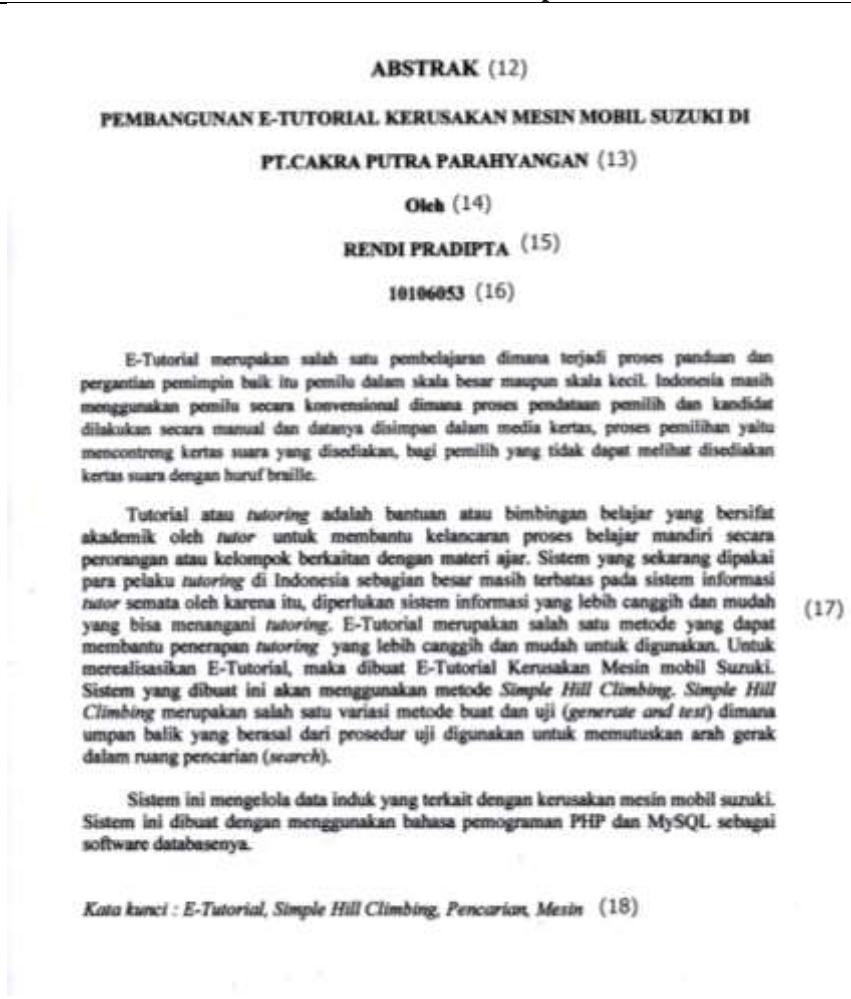
Secara umum struktur target direpresentasikan dengan relasi dalam basis data, oleh karena itu pemrosesan hasil keluaran berkaitan dengan mengisi struktur target dengan informasi relevan yang dihasilkan. Proses pengisian tersebut mencakup normalisasi ke dalam format tertentu contohnya untuk representasi tanggal dan waktu. Kemungkinan fakta yang ditemukan dalam dokumen teks dibutuhkan untuk proses penggabungan fakta (*instance unification*).

2.2 Dokumen Karya Ilmiah

Dokumen adalah surat tertulis atau tercetak yang dipakai sebagai bukti keterangan [15]. Karya ilmiah adalah karangan ilmu pengetahuan yang menyajikan fakta dan ditulis menurut metodologi penulisan yang baik dan benar [14]. Kamus Besar Bahasa Indonesia menjelaskan bahwa karya ilmiah berupa makalah berisi tulisan tentang suatu pokok yang dimaksudkan untuk dibacakan di muka umum dan sering disusun untuk diterbitkan. Pengertian karya ilmiah lainnya adalah makalah yang berisi karangan termasuk tugas peserta didik selama dalam pendidikan di sekolah [15]. Jenis karya ilmiah ada 2 yaitu hasil penelitian contoh skripsi, tesis, disertasi, buku dan makalah dan tinjauan atau usulan/gagasan sendiri contoh buku pelajaran, diktat dan modul. Skripsi termasuk hasil penelitian yang mengemukakan pendapat penulis berdasarkan pendapat orang lain. Pendapat yang diajukan harus didukung oleh data dan fakta empiris-objektif, baik berdasarkan penelitian langsung (observasi lapangan) maupun penelitian tidak langsung (studi kepustakaan) [16]. Pada penelitian ini dokumen karya ilmiah yang digunakan untuk data masukan adalah dokumen skripsi pada bagian halaman depan atau sampul dan abstrak dari program studi Teknik Informatika secara acak yang memiliki 17 kategori atau kelas. Pada halaman sampul terdiri dari kategori Judul Penelitian (Sampul), Jenis Penelitian, Kalimat Pengajuan, *Other*, Penulis (Sampul), NIM (Sampul), Program Studi, Fakultas, Universitas, Kota dan Tahun. Pada halaman abstrak terdiri dari Jenis Halaman, Judul Penelitian (Abstrak), *Other*, Penulis (Abstrak), NIM (Abstrak), Isi Abstrak dan Kata Kunci. Tabel 2.1 menmpilkan bagian – bagian kategori pada lembar sampul dan abstrak beserta kelasnya.

Tabel 2.1 Bagian-bagian Kategori dari Dokumen Karya Ilmiah

Lembar Sampul Skripsi	No	Kategori	Kelas
PEMBANGUNAN E-TUTORIAL KERUSAKAN MESIN MOBIL SUZUKI BERBASIS WEB DI PT.CAKRA PUTRA PARAHYANGAN TASIKMALAYA (1) SKRIPSI (2) Diajukan untuk Menempuh Ujian Akhir Sarjana Program Strata Satu Jurusan Teknik Informatika Fakultas Teknik dan Ilmu Komputer Universitas Komputer Indonesia (3) Oleh : (4) RENDI PRADIPTA (5) 10106053 (6) JURUSAN TEKNIK INFORMATIKA (7) FAKULTAS TEKNIK DAN ILMU KOMPUTER (8) UNIVERSITAS KOMPUTER INDONESIA (9) BANDUNG (10) 2011 (11)	1	Judul Penelitian (Sampul)	0
	2	Jenis Penelitian	1
	3	Kalimat Pengajuan	2
	4	*Other*	16
	5	Penulis (Sampul)	3
	6	NIM (Sampul)	4
	7	Prodi	5
	8	Fakultas	6
	9	Universitas	7
	10	Kota	8
	11	Tahun	9

Lembar Abstrak Skripsi	No	Kategori	Kelas
	12	Jenis Halaman	10
	13	Judul Penelitian (Abstrak)	11
	14	*Other*	15
	15	Penulis (Abstrak)	12
	16	NIM (Abstrak)	13
	17	Isi Abstrak	14
	18	Kata Kunci	16

ABSTRAK (12)

PEMBANGUNAN E-TUTORIAL KERUSAKAN MESIN MOBIL SUZUKI DI PT.CAKRA PUTRA PARAHYANGAN (13)

Oleh (14)

RENDI PRADIPTA (15)

10106053 (16)

E-Tutorial merupakan salah satu pembelajaran dimana terjadi proses panduan dan pergantian pemimpin baik itu pemilu dalam skala besar maupun skala kecil. Indonesia masih menggunakan pemilu secara konvensional dimana proses pendataan pemilih dan kandidat dilakukan secara manual dan datanya disimpan dalam media kertas, proses pemilihan yaitu mencontreng kertas suara yang disediakan, bagi pemilih yang tidak dapat melihat disediakan kertas suara dengan huruf braille.

Tutorial atau *tutoring* adalah bantuan atau bimbingan belajar yang bersifat akademik oleh *tutor* untuk membantu kelancaran proses belajar mandiri secara perorangan atau kelompok berkaitan dengan materi ajar. Sistem yang sekarang dipakai para pelaku *tutoring* di Indonesia sebagian besar masih terbatas pada sistem informasi *tutor* semata oleh karena itu, diperlukan sistem informasi yang lebih canggih dan mudah yang bisa menangani *tutoring*. E-Tutorial merupakan salah satu metode yang dapat membantu penerapan *tutoring* yang lebih canggih dan mudah untuk digunakan. Untuk merealisasikan E-Tutorial, maka dibuat E-Tutorial Kerusakan Mesin mobil Suzuki. Sistem yang dibuat ini akan menggunakan metode *Simple Hill Climbing*. *Simple Hill Climbing* merupakan salah satu variasi metode buat dan uji (*generate and test*) dimana umpan balik yang berasal dari prosedur uji digunakan untuk memutuskan arah gerak dalam ruang pencarian (*search*). (17)

Sistem ini mengelola data induk yang terkait dengan kerusakan mesin mobil suzuki. Sistem ini dibuat dengan menggunakan bahasa pemograman PHP dan MySQL sebagai software databasenya.

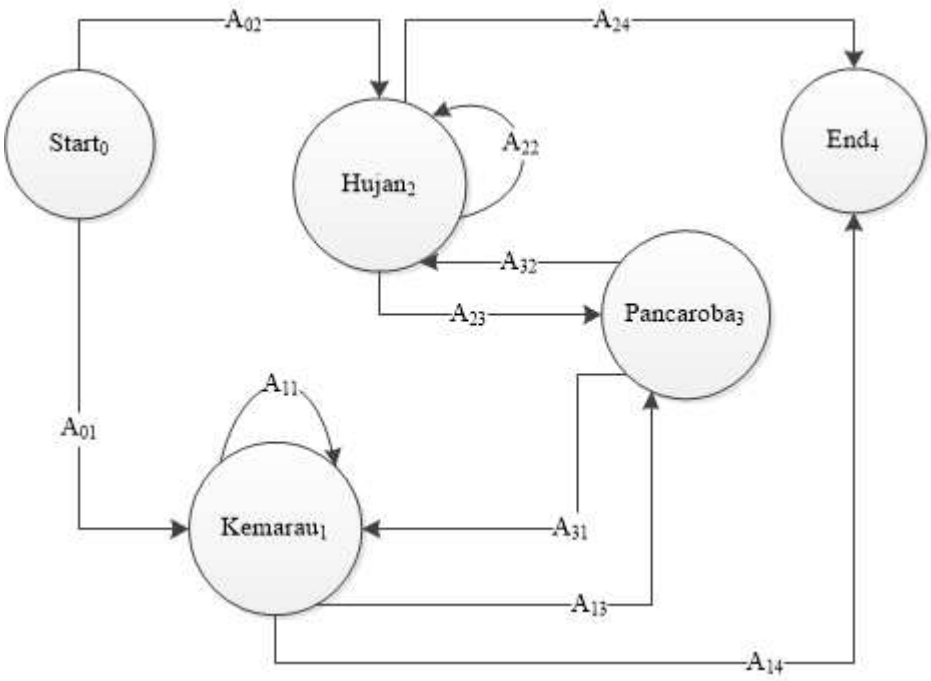
Kata kunci : E-Tutorial, Simple Hill Climbing, Pencarian, Mesin (18)

2.3 Klasifikasi

Supervised Learning disebut juga klasifikasi atau pembelajaran secara induktif dalam pembelajaran mesin (*machine learning*). Pembelajaran dilakukan dari data yang telah dikumpulkan sebelumnya dan menggambarkan keadaan sebelumnya dalam pengaplikasian di dunia nyata [17]. Menurut Zdravko, tujuan klasifikasi untuk membuat sebuah pemetaan (disebut juga model atau hipotesis) diantara sebuah set dokumen dan set label kelas. Hasil pemetaan digunakan untuk menentukan kelas dokumen baru (belum diberi label kelas awal) secara otomatis [18]. Salah satu pengklasifikasi yang akan digunakan dalam penelitian ini adalah pengklasifikasi sekuens (*Sequence Classifier*). Pengklasifikasi sekuens adalah model yang bertugas untuk menentukan sebuah label atau kelas ke setiap unit dalam sebuah sekuens, sehingga memetakan sebuah sekuens observasi ke sekuens label [19]. Algoritma yang termasuk dalam pengklasifiksi sekuens adalah *Hidden Markov Model* (HMM) dan *Maximum Entropy Markov Model* (MEMM).

1.3.1 Hidden Markov Model

Markov Model disebut juga *Markov Chain* atau *Markov Process* merupakan bagian dari proses stokastik yang memiliki properti *Markov*, sehingga apabila diberikan data masukan keadaan saat ini, keadaan akan datang dapat diprediksi dan ia lepas dari keadaan masa lalu. Dengan kata lain, kondisi masa depan dituju dengan menggunakan probabilitas. Model bagian dari *Finite state* atau *Finite Automaton*. *Finite Automaton* adalah kumpulan state yang transisi antar state-nya dilakukan berdasarkan observasi. Pada *Markov Chain*, setiap busur antar state berisi probabilitas kemungkinan jalur tersebut akan diambil. Jumlah probabilitas semua busur yang keluar dari sebuah simpul adalah satu.



Gambar 2.3 Contoh Probabilitas Transisi Dari Hidden Markov Model

Pada gambar 2.3, A_{ij} menunjukkan probabilitas transisi dari state i ke state j. Contoh, simpul $Start_0$ memiliki dua kemungkinan A_{01} dan A_{02}, sehingga jumlah probabilistik $A_{01} + A_{02} = 1$. Hal ini berlaku juga untuk simpul-simpul yang lain. *Markov Chain* berperan untuk menghitung probabilitas suatu kejadian teramati,yang secara umum dirumuskan berikut:

$$P(q_i \dots q_{i-1}) = P(q_i|q_{i-1}) \tag{2.1}$$

Karena setiap A_{ij} merepresentasikan probabilitas dari $P(q_i|q_{i-1})$ dan nilai i atau j dapat sama maka diberlakukan aturan batasan:

$$\sum_{j=1}^{n} a_{ij} = 1 \quad \forall i \tag{2.2}$$

Markov chain bermanfaat untuk menghitung probabilitas urutan kejadian yang dapat diamati. Untuk mengetahui urutan kejadian yang tersembunyi menggunakan algoritma *Hidden Markov Model*.

Hidden Markov Model (HMM) merupakan model statistik dimana sistem yang dimodelkan diasumsikan sebagai markov proses dengan kondisi yang tidak terobservasi, oleh karena itu urutan langkah yang dibuat oleh HMM memberikan

informasi menunjuk langkah yang dilewati model, bukan kepada parameter dari model. HMM merupakan variasi dair *finite state machine* yang memiliki kondisi tersembunyi himpunan *hidden state* $W = w_1 w_2, \ldots, w_n$, suatu nilai keluaran himpunan observasi $O = O_1, O_2, \ldots, O_n$, proabilitas transisi $A = a_{11} a_{12}, \ldots, a_{n1} \ldots a_{nn}$ setiap a_{ij} merepresentasikan perpindahan probabilitas tersebut dari state I ke state j, probabilitas emisi atau *likelihood observation* $B = b_i(O_t)$ merupakan probabilitas observasi O_t dibangkitkan oleh state ke i, probabilitas awal $\pi = \pi_1, \pi_2, \ldots, \pi_n$ dimana $\sum_{j=1}^{n} \pi_i = 1 \quad \forall$i.

Rumus (2.1) *Hidden Markov Model* untuk menghitung keterurutan kelas target terbaik $P(T|W)$ berdasarkan aturan *Bayes* dan likelihood $P(W|T)$ sebagai berikut [17]:

$$\hat{T} = \arg\max_{T} P(T|W) \tag{2.3}$$
$$= \arg\max_{T} P(W|T)P(T)$$
$$= \operatorname{argmax}_{T} \prod_{i} P(W_i|T_i) \prod_{i} P(T_i|T_{i-1})$$

$W = W_1^n$ keterurutan kata dan $T = T_1^n$ keterurutan kelas target, $\hat{T}$ menentukan keterurutan terbaik dari kelas target. $P(kata_i|target_i)$ probabilitas emisi dan $P(target_i|target_{i-1})$ probabilitas transisi.

1.3.2 Maximum Entropy Markov Model

Maxium Entropy Markov Model atau MEMM adalah sekuens model yang diadaptasi dari *MaxEnt* (*multinomial logistic regresstion*) *classifier*. Berdasarkan *logistic regression*, MEMM adalah model sekuens yang diskriminatif sedangkan HMM (*Hidden Markov Model*) model sekuens yang generatif [19].

Pada MEMM menggunakan *Maximum Entropy* untuk proses klasifikasinya sehingga rumus pelatihan untuk menentukan kelas target terbaik $P(T|W)$ rumus (2.4) sebagai berikut [19]:

$$\hat{T} = \arg\max_{T} P(T|W) \tag{2.4}$$
$$= \arg\max_{T} P(t|t', w)$$

$$= \underset{T}{\operatorname{argmax}} \frac{\exp(\sum_j \theta_j f_j(w_i, t_i))}{\sum_{t' \in seluruh\ kelas\ T} \exp(\sum_j \theta_j f_j(w_i, t'))}$$

t_i adalah kelas ke-i dari data yang ditunjuk. W adalah kata dari seluruh dataset. T' adalah seluruh kelas awal yang telah ditentukan. θ adalah vektor bobot awal. $f_i(w_i, t_i)$ adalah fungsi indikator yang menghasilkan nilai 0 jika syarat tidak terpenuhi dan 1 jika syarat terpenuh [19]. Indikator yang diambil berdasarkan kelas ke-i dan ekstraksi fitur ke-i.

Multinomial Logistic Regression atau disebut juga *Maximum Entropy Model* bagian dari pengklasifikasi exponensial atau *log-linear*, mengekstrak set kata, mengkombinasikan secara linear (mengkalikan setiap kata dengan bobot dan menjumlahkannya) kemudian menerapkan sebuah fungsi pada kombinasi tersebut [17]. Proses pelatihan dalam *logistic regression* menggunakan fungsi objektif untuk meminimalkan nilai *error* pada data latih, yaitu *cross entropy loss function* (2.5) [17]:

$$L_{CE}(\hat{y}, y) = -\sum_{k=1}^{K} 1\{y = k\} \log p(y = k|x) \tag{2.5}$$

1{} sebagai fungsi untuk menentukan jika indeks kelas sebenernya ke k dari data latih sama dengan indeks *softmax* ke k maka nilai fungsi adalah 1 jika tidak memenuhi syarat maka nilai fungsi 0. Nilai *cross entropy* dihitung rata-rata seluruh datanya untuk mendapatkan nilai *error* secara keseluruhan. Untuk mengoptimalkan dan memperbarui nilai bobot θ berdasarkan hasil *cross entropy* jika belum mendekati 0 maka dihitung nilai *gradient descent*. Berikut proses perhitungan *gradient descent* pada rumus (2.6) [19]:

$$\frac{\partial L_{CE}}{\partial W_k} = (1\{y = k\} - p(y = k|x))X_k \tag{2.6}$$

$$\theta' = \theta - \eta \frac{\partial L_{CE}}{\partial W_k}$$

Sama dengan *cross entropy* untuk fungsi 1{}, X_k berdasarkan matriks fungsi fitur untuk setiap k kelas, η *learning rate* nilai bobot jika terlalu tinggi nilainya maka

proses pelatihan akan bobot θ akan terlalu besar dan melewat nilai *cross entropy*. Jika terlalu kecil nilainya maka membuat proes pelatihan menjadi sangat lambat dan perubahan bobot yang sangat kecil. θ' perbaruan bobot [19]. Proses untuk menentukan urutan kelas berdasarkan data observasi disebut *decoding* dan sebagai metode pengujian. Pada penelitian ini menggunakan algoritma Viterbi untuk *decoder*-nya. Berikut ini rumus (2.7) untuk proses pengujian Viterbi [19]:

$$arg\max_{T} P(T|W) = \prod_{i=1} \mathrm{P}(t_i|w_1 \dots w_n, t_i\,; \theta) \tag{2.7}$$

1. Inisalisasi:

   $\pi(*,*,0) = 1$
2. Rekursi:

   Untuk setiap kelas $j\epsilon\{1 \dots k\}$, untuk setiap token $t\,\epsilon\,[1 \dots n]$ dan untuk kelas sebelumnya $i\,\epsilon\,[1 \dots k]$

   $v_t(j) = \max_{i=1\dots k}(v_{t-1}(i) * \mathrm{P}(\mathrm{t}_j|w_1 \dots w_t, t_i\,; \theta))$
3. Terminasi:

   Nilai tertinggi : P* = $\mathrm{argmax}_{i\,\epsilon 1\dots k}\, v_t(i)$

2.1 Tahap Pengujian

Untuk mengevaluasi performa dari hasil proses klasifikasi menggunakan nilai dari F-measure, Precision dan Recall dengan rumus (2.8) sebagai berikut [19]:

$$F - measure = \frac{2 * recall * precision}{recall + precision} \tag{2.8}$$

F-measure adalah *harmonic mean* dari nilai *precision* dan *recall*. *Precision* adalah jumlah sampel berkategori positif diklasifikasi benar dibagi dengan total sampel yang diklasifikasi sebagai sampel positif. *Recall* adalah jumlah sampel diklasifikasi positif dibagi total sampel dalam data uji dengan set kategori yang positif. Mengukur akurasi dapat ditampilkan pada *confussion matrix*. *Confussion*

Matrix berisi tentang klasifikasi aktual dan yang telah diprediksi yang dilakukan oleh sebuah sistem klasifikasi. Kinerja sebuah sistem klasifikasi umumnya dievaluasi dengan menggunakan data dalam matriks. Tabel 2.2 berikut menunjukan *confussion matrix* untuk klasifikasi dua kelas. Setelah data diuji oleh algoritma *viterbi* hasil klasifikasi akan diuji menggunakan pengukuran standar *information retrieval* [20].

Tabel 2.2. Definisi Parameter TP, FP, FN, TN

Category Set		***Expert Judgement***	
		Yes	***No***
System Judgement	***Yes***	*True Positives*	*False Positives*
	No	*False Negatives*	*True Negatives*

True Positives adalah jumlah record positif yang terklasifikasi sebagai positif oleh sistem. *False Positives* adalah jumlah *record* positif yang terklasifikasi sebagai negatif oleh sistem. *False Negatives* adalah jumlah record negatif yang terklasifikasi sebagai positif oleh sistem. *True Negatives* adalah record negatif yang terklasifikasi sebagai negatif oleh sistem. Setiap kolom dari *confussion matrix* merupakan contoh di kelas yang telah diprediksi, sedangkan setiap baris merupakan mewakili contoh di kelas sebenarnya. Setelah didapat semua nilai tersebut(*True Positives, False Positive, False Negatives, True Negatives*), bisa dihitung nilai akurasinya [20].

$$Akurasi = \frac{TP + TN}{TP + FP + TN + FN} \quad \textbf{(2.9)}$$

$$Precision = \frac{TP}{TP + FP} \quad \textbf{(2.10)}$$

$$Recall = \frac{TP}{TP + FN} \quad \textbf{(2.11)}$$

2.2 Natural Language Processing (NLP)

Alat preprocessing yang tepat dalam banyak studi Natural Language Processing (NLP) sangat penting untuk dilakukan memberikan akurasi yang lebih baik. Tahap preprocessing leksikal, seperti pendeteksian kata-kata dasar (Stemming) dan deteksi jenis kata (penandaan POS) berdampak besar bagi sistem komputasi bahasa itu membutuhkan penentuan struktur kalimat. Dalam bahasa Indonesia, penelitian tentang Stemming dan Penandaan POS masih dilakukan, baik dengan menggunakan metode statistik atau aturan tertentu. Beberapa masalah yang dihadapi untuk pengolahan tersebut adalah kurangnya corpus di Indonesia dan Indonesia ketidaklengkapan aturan yang tersedia. Penelitian tentang stemming, pertama kali diterbitkan oleh Julie Beth Lovins pada tahun 1968 [21].

2.3 Pemodelan Sistem

Suatu sistem memiliki beberapa proses sehingga membutuhkan model untuk memperjelas gambaran yang terjadi pada proses tersebut. Berikut ini pemodelan sistem yang akan digunakan pada penelitian ini, terdiri dari Blok Diagram, UML (*Unified Modeling Language*) dan *Flowchart*.

1.5.1 Blok Diagram

Proses yang terjadi di dalam suatu sistem, dapat digambarkan dengan mudah melalui data masukan, proses, serta keluaran yang dihasilkan. Model yang memiliki ketiga tahapan dinamakan dengan model blok diagram. Blok diagram digunakan untuk merepresentasikan suatu sistem atau sejumlah blok yang berhingga dalam rangkaian beberapa proses menggunakan blok. Elemen yang terdapat pada diagram blok terdiri dari sebab akibat input dan output [22]. Berikut gambaran blok diagram secara umum.

Gambar 2.4 Block Diagram

Penggunaan blok diagram pada penelitian ini untuk menggambarkan beberapa proses yang terjadi pada sistem dari mulai data masukan sampai hasil yang diharapkan.

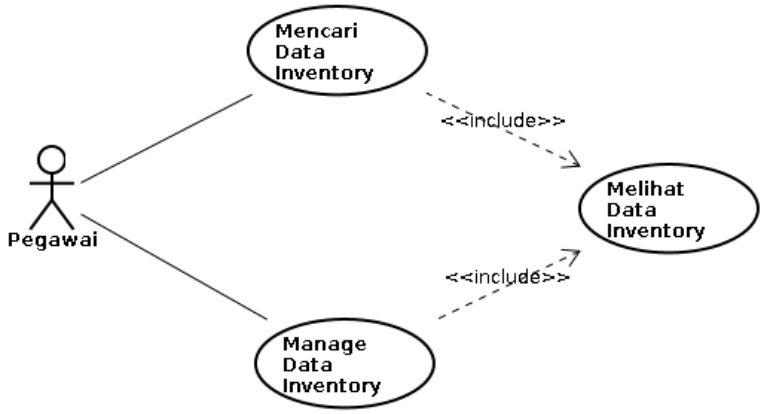
1.5.2 Unified Modeling Language (UML)

Unified Modeling Language (UML) adalah keluarga notasi grafis yang membantu dalam menggambarkan dan merancang sistem perangkat lunak terutama yang dibangun dengan menggunakan orientasi objek. Definisi ini adalah definisi yang telah disederhanakan. Pada kenyataannya, pendapat orang-orang berbeda tentang UML berbeda satu sama lain. Hal ini disebabkan oleh sejarah dan perspektif yang berbeda tentang membuat proses rekayasa perangkat lunak yang efektif [23].

Diagram-diagram UML yang digunakan pada sistem ekstraksi informasi adalah *use case diagram, activity diagram, class diagram,* dan *sequence diagram.*

1.5.2.1 Use Case Diagram

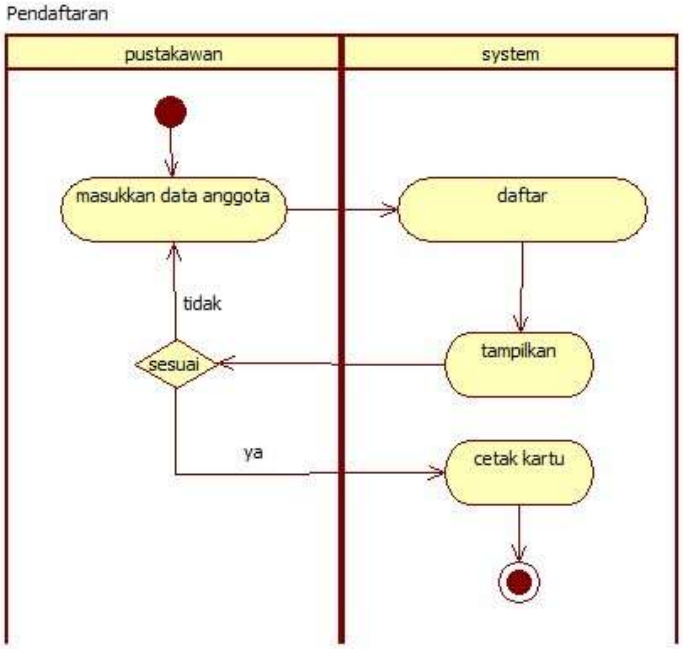
Use Case adalah teknik untuk menggambarkan kebutuhan fungsional dari sebuah sistem. *Use case* menggambarkan interaksi antara pengguna dengan sistem, menyediakan sebuah cerita bagaimana sebuah sistem digunakan [23]. Dalam *Use case*, para pengguna disebut sebagai aktor. Aktor merupakan sebuah peran yang dimainkan seorang pengguna dalam kaitannya dengan sistem. Aktor tidak harus berwujud manusia atau makhluk hidup, jika sebuah sistem melakukan sebuah layanan untuk sistem komputer lain, sistem lain tersebut merupakan aktor.



Gambar 2.5 Contoh *Use Case* Diagram

1.5.2.2 Activity Diagram

Activity diagram adalah teknik untuk menggambarkan logika prosedural, proses bisnis, dan jalur kerja. Dalam beberapa hal, *activity diagram* memainkan peran mirip diagram alir, tetapi perbedaan prinsip antara notasi diagram alir adalah *activity diagram* mendukung *behavior* paralel [23]. Node pada sebuah activity diagram disebut sebagai *action*, sehingga diagram tersebut menampilkan sebuah *activity* yang tersusun dari *action*.

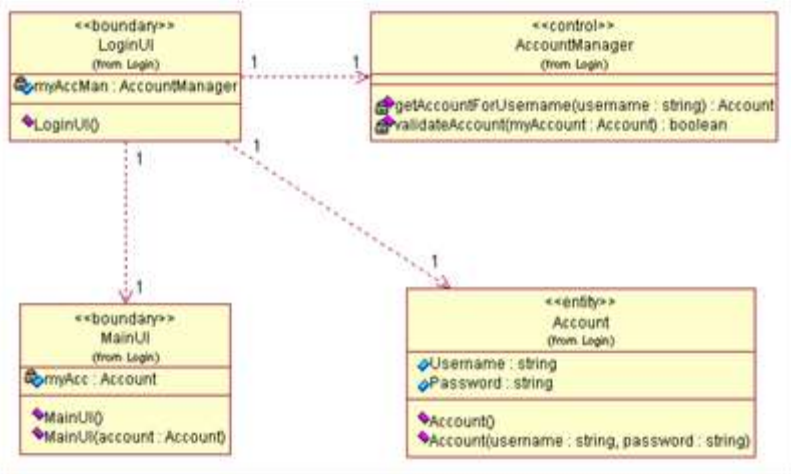


Gambar 2.6 Contoh *Activity Diagram*

1.5.2.3 Class Diagram

Class diagram menggambarkan jenis-jenis objek yang terdapat pada sistem dan berbagai macam relasi statis yang ada di antara objek-objek tersebut. *Class diagram* juga menunjukkan properti-properti dan operasi-operasi pada suatu *class* dan batasan atau *constraint* yang berlaku pada cara objek-objek tersebut saling berhubungan [23].

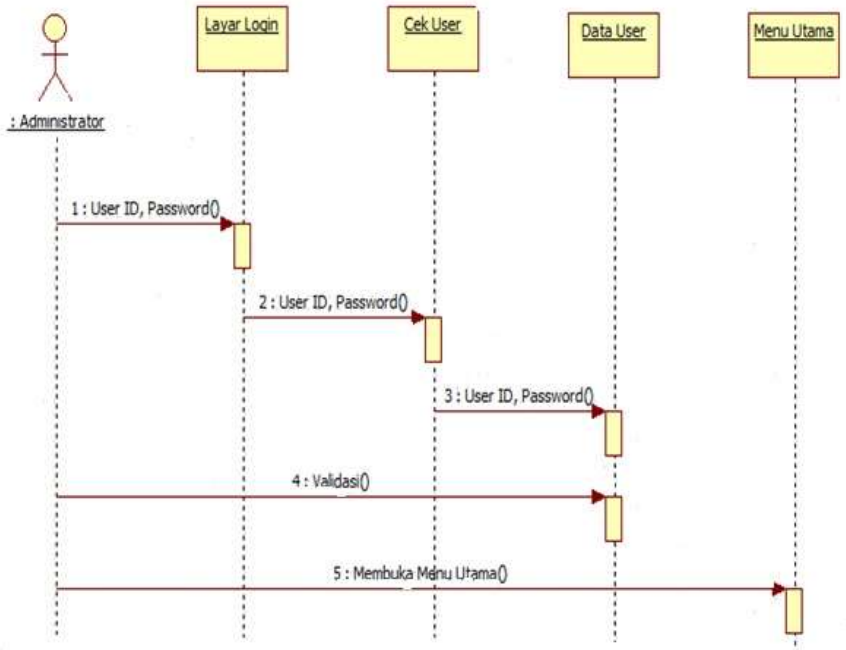
Multiplicity dari suatu properti adalah indikasi seberapa banyak objek yang dapat mengisi properti tersebut. Secara umum, *multiplicity* didefinisikan dengan batas bawah dan batas atas. Batas bawah berupa angka positif dan nol, sedangkan batas atas berupa angka positif dan * (tidak terbatas) [23].



Gambar 2.7 Contoh *Class Diagram*

1.5.2.4 Sequence Diagram

Interaction diagrams menggambarkan bagaimana kumpulan objek dapat berkolaborasi dalam tindakan tertentu. UML mendefinisikan beberapa bentuk *interaction diagram*, salah satunya yang paling umum adalah *sequence diagram*. Secara khusus, *sequence diagram* menangkap perilaku dari suatu skenario. Diagram ini menunjukkan sejumlah objek contoh dan pesan-pesan yang melewati objek-objek tersebut dalam suatu *use case* [23].



Gambar 2.8 Contoh *Sequence Diagram*

2.4 OOP (Object Oriented Programming)

Pemrograman berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapann prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktifitas analisis beorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek.

Pada saat ini, metode berorientais objek banyak dipilih karena metodologi lama banyak menimbulkan masalah seperti adanya kesulitan pada saat mentranformasi hasil dari satu tahap pengembangan ke tahap berikutnya, misalnya pada metode pendekatan terstruktur, jenis aplikasi yang dikembangkan saat ini berbeda dengan masa lalu. Aplikasi yang dikembangkan saat ini sangat beragam (aplikasi bisnis, *real-time*, *utility* dan sebagainya) dengan platform yang berbedabeda, sehingga menimbulkan tuntutan kebutuhan metodologi

pengembangan yang dapat mengakomodasi ke semua jenis aplikasi. Keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut:

a. Meningkatkan Produktivitas

   Karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (*reuseable*).

b. Kecepatan Pengembangan

   Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan mennyababkan berkurangnya kesalahan pada saat pengkodean

c. Kemudahan Pemeliharaan

   Karena dengan model objek, pola-pola yang cendrung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering diubah-ubah.

d. Adanya Konsistensi

   Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.

e. Meningkatkan Kualitas Perangkat Lunak

   Karena adanya pendekatan pengembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat pengambangannya, perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

Berikut beberapa contoh bahasa pemograman yang mendukung pemrograman berorinentasi objek adalah:

a. *Smalltalk*

   Smalltalk adalah salah satu bahasa pemograman yang diekmbangkan untuk mendukung pemrograman beroirentasi objek.

b. Bahasa Pemrograman *Eiffel*

   Eiffel merupakan bahsa pemrograman yang kembangkan untuk mendukung pemrograman berorientasi objek oleh Bertrand Meyer dan *compiler*.

c. Bahasa Pemrograman (*Web*) PHP

Php dibuat pertama kali oleh seorang perekayasa perangkat (software engineering) yang bernama *Rasmus Lerdoff*.

d. Bahasa Pemrograman C++

C++ merupakan pengembangan lebih lanjut dari bahasa pemrograman C untuk mendukung pemrograman berorientasi objek.

e. Bahasa Pemrograman Java

Java dikembangkan oleh perusahaan Sun Microsystem. Java menurut definisi dari Sun Microsystem adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer *standalone* ataupun pada lingkungan jaringan.

2.5 Bahasa Pemrograman

Bahasa pemrograman merupakan bahasa yang digunakan dalam pembangunan sistem ekstraksi informasi menggunakan algoritma *MEMM*. Bahasa pertama yang digunakan adalah *Java* sebagai pemrosesan ekstraksi fitur dan algoritma *MEMM*.

1.11.1 Java

Java adalah bahasa pemrograman serbaguna. Java dapat digunakan untuk membuat suatu program sebagaimana Anda membuatnya dengan bahasa seperti Pascal atau C++. Yang lebih menarik, Java juga mendukung sumber daya internet yang saat ini populer, yaitu *World Wide Web* atau yang sering disebut web. Java juga mendukung aplikasi *client*/*server*, baik dalam jaringan *local* (LAN) maupun jaringan berskala luas (WAN) [24].

Java dikembangkan oleh Sun Microsystems pada Agustus 1991, dengan nama semula Oak. Konon Oak adalah pohon semacam Jati yang terlihat dari jendela tempat pembuatnya, James Gosling, bekerja. Ada yang mengatakan bahwa Oak adalah singkatan dari "*Object Application Kernel*", tetapi ada yang menyatakan hal itu muncul setelah nama Oak diberikan. Pada Januari 1995, karena nama Oak dianggap kurang komersial, maka diganti menjadi Java [24].

Dalam sejumlah literatur disebutkan bahwa Java merupakan hasil perpaduan sifat dari sejumlah bahasa pemrograman, yaitu C, C++, Object-C,

SmallTalk, dan *Common* LISP. Selain itu Java juga dilengkapi dengan unsur keamanan. Yang tak kalah penting adalah bahwa Java menambahkan paradigma pemrograman yang sederhana. Jika telah mengenal C atau C++, yang mengandalkan pointer, Java justru meninggalkannya sehingga anda akan memperoleh kemudahan saat menggunakannya [24].

Program java bersifat tidak bergantung pada *platform*, artinya, Java dapat dijalankan pada sembarang komputer dan bahkan pada sembarang sistem operasi. Ketidakbergantungan pada platform sering dinyatakan dengan istilah portabilitas. Yang menarik, tingkat portabilitas java tidak hanya sebatas pada program sumber (*source code*), melainkan juga pada tingkat kode biner yang disebut *bytecode*. Dengan demikian bila telah mengkompilasi program java pada komputer berbasis operasi Windows, dapat juga menjalankan hasil kompilasi pada Macintosh secara langsung, tanpa perlu mengkompilasi ulang [24].