BAB 2
TINJAUAN PUSTAKA

2.1. Ekstraksi Informasi

Ekstraksi informasi merupakan suatu bidang ilmu untuk pengolahan bahasa alami, dengan cara mengubah teks tidak terstruktur menjadi informasi dalam bentuk terstruktur [6]. Proses ekstraksi informasi dapat dilakukan dengan dua pendekatan, yaitu berbasis aturan dan statistika. Pendekatan berbasis aturan dilakukan dengan menerapkan aturan-aturan yang dibuat oleh pakar, sedangkan statistik dilakukan dengan pembelajaran terhadap sejumlah data latih [6].

Pada tabel 2.1 terdapat sebuah kategori didalam kategori tersebut ada judul, penulis(sampul), nim(sampul), dll. Kemudian, supaya mesin dapat mengenali kategori-kategori tersebut dibuatlah aturan-aturan.

2.2. Dokumen Karya Tulis Ilmiah

Dokumen adalah surat penting atau berharga yang sifatnya tertulis atau tercetak yang berfungsi atau dapat di pakai sebagai bukti atau katerangan. Karya tulis ilmiah adalah berbagai macam tulisan yang dilakukan oleh seseorang atau kelompok dengan menggunakan tata cara ilmiah [7]. Dengan kata lain karya tulis ilmiah laporan tertulis hasil kegiatan ilmiah. Pada penelitian ini dokumen karya tulis ilmiah skripsi Universitas Komputer Indonesia.

Skripsi adalah karya tulis ilmiah yang mengemukakan pendapat penulis berdasarkan pendapat orang lain yang didukung oleh data dan fakta untuk melengkapi syarat guna memperoleh gelar sarjana dari suatu perguruan tinggi [7].

Pada penelitian ini, dataset yang akan digunakan yaitu sampul dan abstrak pada dokumen karya tulis ilmiah skripsi Universitas Komputer Indonesia. Pada dokumen sampul terdapat beberapa kategori, yaitu Judul Penelitian(sampul), Jenis Penelitian, Kalimat Pengajuan, Penulis(sampul), NIM(sampul), Program Studi dan Fakultas. Sedangkan pada lembar abstrak terdiri dari Judul Halaman Abstrak, Judul Penelitian (abstrak), other, Penulis (abstrak), NIM (abstrak), Isi Abstrak, Kata Kunci [3]. Untuk lebih jelasnya lihat tabel 2.1 dan 2.2.

Tabel 2.1 Kategori-kategori Yang Ada Pada Lembar Sampul

Lembar Sampul Skripsi	No	Kategori	Kelas
PEMBANGUNAN APLIKASI MOBILE ANDROID PEREKOMENDASI MASAKAN RUMAHAN BERDASARKAN HARI RAYA MENGGUNAKAN API GOOGLE CALENDER SKRIPSI Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana (S1) SINTIA DEWIE CHANIAGO 10113073  PROGRAM STUDI TEKNIK INFORMATIKA FAKULTAS TEKNIK DAN ILMU KOMPUTER UNIVERSITAS KOMPUTER INDONESIA 2018	1	Judul Penelitian Sampul	0
	2	Jenis Penelitian	1
	3	Kalimat Pengajuan	2
	4	Penulis (Sampul)	3
	5	NIM (Sampul)	4
	6	Program Studi	5
	7	Fakultas	6
	8	Universitas	7
	9	Tahun	8

Tabel 2.2 Kategori-kategori Yang Ada Pada Lembar Abstrak

Lembar Abstrak Skripsi	No	Kategori	Kelas
ABSTRAK **PEMBANGUNAN APLIKASI MOBILE ANDROID PEREKOMENDASI MASAKAN RUMAHAN BERDASARKAN HARI RAYA MENGGUNAKAN API GOOGLE CALENDER** Oleh: **Sintia Dewie Chaniago** **10113073** Belum adanya media informasi khusus untuk melakukan pencarian masakan rumahan dan masakan hari raya, belum adanya media *portable* yang dapat memudahkan setiaip memasak masakan rumahan dan hari raya yang kemudian ingin dengan mudah untuk menjualnya. karena itu dibangun aplikasi *android* dengan menggunakan teknologi API yang memanfaatkan google *calnder* untuk mengetahui tanggal – tanggal di hari raya dan menggunakan teknologi Google *Cloud Messaging* untuk memberikan notifikasi adanya masakan hari raya. Salah satu bentuk implementasi pengguna teknologi untuk menjual dan membeli masakan rumahan. Ditarik kesimpulan bahwa aplikasi *mobile* dan *android* yang dibangun agar mudah dipahami, dan menarik sesuai dengan tujuan penelitian. Untuk mengembangkan sistem yang sudah dibangun dapat ditambahkan proses input data dari penjual maupun pembeli sehingga aplikasi dapat berkembang dan semakin banyak yang terdaftar pada aplikasi ini. **Kata kunci** : Aplikasi mobile android, google calender, google cloud messaging, rekomendasi masakan rumahan	1	Judul Halaman Abstrak	9
	2	Judul Penelitian (Abstrak)	10
	3	*Other*	11
	4	Penulis (Abstrak)	12
	5	NIM (Abstrak)	13
	6	Isi Abstrak	14
	7	Kata Kunci	15

2.3. Penelitian Terdahulu

Penelitian sebelumnya yang sudah pernah dilakukan terkait dengan algoritma CRF (*Conditional Random Field*) perlu dipaparkan untuk rujukan. Adapaun penelitian sebelumnya sebagai berikut.

1. Pada penelitian yang dilakukan oleh Fuchun Peng dan Andrew McCallum dengan judul "Information Extraction From Research Papers Using Conditional Random Field" pada tahun 2005 menggunakan Conditional Random Field (CRF) untuk tugas mengekstrak berbagai bidang umum dari header dan kutipan makalah penelitian. CRF menyediakan cara yang berprinsip untuk menggabungkan berbagai fitur lokal, fitur leksikon eksternal dan fitur tata letak global. Pada artikel ini juga dibuat eksplorasi empiris beberapa faktor, termasuk variasi pada gaussian, laplace, hiperbolik-L1 priors untuk meningkatkan regularisasi dan beberapa kelas fitur. Pada dataset patokan standar, kami mencapai kinerja seni yang baru, mengurangi kesalahan rata-rata F1 sebesar 36% dan tingkat kesalahan kata sebesar 78%. Secara keseluruhan akurasi kata yang didapat 98,3% sesuai dengan pengurangan tingkat kesalahan kata 78% [4].
2. Pada penelitian yang dilakukan oleh Kenji Hirohata, Sophia Ananiadou, Naoaki Okazaki dan Mitsuru Ishizuka dengan judul "Identifying Sections in Scientific Abstracts Using Conditional Random Field" pada penelitian ini menyajikan pendekatan baru untuk mengkategorikan kalimat dalam abstrak ilmiah menjadi 4 bagian, yaitu : tujuan, metode, hasil dan kesimpulan. Memformalkan tugas kategorisasi sebagai masalah pelabelan menggunakan CRF untuk membubuhi keterangan label ke dalam kalimat abstrak. Pada penelitian ini didapat akurasi per-kalimat 95,5% dan 68,8% akurasi per-akurasi abstrak [8].
3. Pada penelitian yang dilakukan oleh Fivip Saefulloh dengan judul "Part of Speech Tagger untuk Bahasa Indonesia Menggunakan Conditional Random Field" pada penelitian ini metode yang digunakan untuk membangun POS Tagger adalah Conditional Random Field (CRF) dan dari hasil pengujian yang dilakukan rata-rata akurasi yang dihasilkan adalah 78,20% [5].

2.4. Tokenisasi

Tokenisasi adalah proses memisahkan kalimat menjadi kata per kata, paragraf atau halaman menjadi *token* atau potongan tunggal atau termmed word yang berdiri sendiri [3].

Misalkan saja contoh kalimat dari tabel 2.3 kalimat tersebut akan ditokenisasi dengan cara memanfaatkan setiap spasi. Maka akan seperti pada tabel 2.3 berikut ini.

Tabel 2.3 Tokenisasi Kata

Kalimat	Kata
Pembangunan Aplikasi E-Commerce Di Butik Sephira	Pembangunan
	Aplikasi
	E-Commerce
	Di
	Butik
	Sephia

Pada tabel 2.3 contoh dari tokenisasi. Pada kolom kata tersebut hasil dari tokenisasi dari kolom kalimat yang berada pada tabel sebelah kiri.

2.5. Ekstraksi Fitur

Fitur merupakan elemen paling penting untuk membangun model ini. Pemilihan fitur yang tepat dapat meningkatkan akurasi pelabelan. Pada penelitian ini sendiri terdapat 15 fitur yang digunakan. 13 fitur diantaranya merujuk pada penelitian yang dilakukan oleh F. Peng [4]. Sedangkan dua fitur yaitu WORD dan EIGHDIGIT merujuk pada penelitian yang dilakukan oleh Firdam [3].

Adapun penjelasan dari ke 15 fitur pada tabel 2.4 berikut ini.

Tabel 2.4 Ekstraksi Fitur

No	Nama Fitur	Keterangan
1	INITCAPS	Mengenali setiap token yang hurufnya di awali dengan kapital.
2	ALLCAPS	Mengenali setiap token yang semua hurufnya kapital.
3	CONTAINSDIGIT	Mengenali setiap token yang mengandung digit.
4	ALLDIGIT	Mengenali setiap token yang semuanya digit.
5	CONTAINSDOTS	Mengenali setiap token yang mengandung titik.
6	LOWERCASE	Mengenali setiap token yang hurufnya kecil semua.
7	PUNCTUATION	Mengenali setiap token yang mengandung tanda tertentu

		seperti titik, koma, titik dua, titik koma, tanda kurung dan tanda seru.
8	EIGHDIGIT	Fitur ini dikhususkan mengenali token yang memiliki digit dengan panjang 8 digit.
9	WORD	Fitur ini dikhususkan untuk memberikan bobot pada token untuk kelas JENIS_PENELITIAN dan KALIMAT_PENGAJUAN.
10	LINE_START	Mengenali posisi token pada indeks array awal.
11	LINE_IN	Mengenali posisi token pada indeks array tengah.
12	LINE_END	Mengenali posisi token pada indeks array akhir.
13	PERSON	Mengenali token nama seseorang.
14	ORGANIZATION	Mengenali token sebuah organization.
15	YEAR	Mengenali ciri token tahun.

Pada tabel 2.4 semua fitur akan diberi bobot 1 atau 0. Pemberian bobot 1 apabila setiap token termasuk ke dalam salah satu fitur dan akan di berikan 0 jika tidak termasuk ke dalam salah satu fitur tersebut. Berikut contoh ekstraksi fitur pada tabel 2.5

Tabel 2.5 Contoh Ekstraksi Fitur

No	Nama Fitur	Kata					
		PEMBANGUNAN	APLIKASI	E-COMMERCE	DI	BUTIK	SEPHIA
1	INITCAPS	1	1	1	1	1	1
2	ALLCAPS	1	1	1	1	1	1
3	CONTAINSDIGIT	0	0	0	0	0	0
4	ALLDIGIT	0	0	0	0	0	0
5	CONTAINSDOTS	0	0	0	0	0	0
6	LOWERCASE	0	0	0	0	0	0
7	PUNCTUATION	0	0	1	0	0	0
8	EIGHDIGIT	0	0	0	0	0	0
9	WORD	0	0	0	0	0	0
10	LINE_START	1	0	0	0	0	0
11	LINE_IN	0	0	0	0	0	0
12	LINE_END	0	0	0	0	0	0
13	PERSON	0	0	0	0	0	0
14	ORGANIZATION	0	0	0	0	0	0
15	YEAR	0	0	0	0	0	0

2.6. Bahasa Pemrograman

Penelitian ini bahasa pemograman PHP (*Personal Home Page*). *Personal Home Page* (PHP) adalah bahasa pemrograman berbasis web yang memiliki kemampuan untuk memproses data secara dinamis. PHP dikatakan sebagai sebuah *server-side embedded script language* artinya sintaks-sintaks dan perintah yang kita berikan akan sepenuhnya dijalankan oleh server disertakan pada halaman HTML biasa. Aplikasi-aplikasi yang dibangun oleh PHP pada umumnya akan memberikan hasil pada *web browser*, akan tetapi prosesnya secara keseluruhan dijalankan di server [3].

2.7. Perangkat Lunak Pendukung

Perangkat lunak pendukung adalah suatu kebutuhan perangkat yang digunakan untuk membangun sistem ekstraksi informasi menggunakan CRF. Beberapa perangkat pendukung sebagai berikut.

2.7.1. XAMPP

XAMPP merupakan paket PHP berbasis open source ang dikembangkan oleh sebuah komunitas Open Source. Dengan menggunakan XAMPP kita tidak perlu lagi melakukan penginstallan program yang lain. Karena semua kebutuhan telah disediakan oleh XAMPP. Beberapa paket ang telah disediakan adalah Apache, MySql, Filezila dan Phpmyadmin [3].

2.7.2. MySQL

MySQL adalah sebuah implementasi dari sistem manajemen basisdata relasional (RDBMS) yang didistribusikan secara gratis. Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersiat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basisdata yang telah ada sebelumnya SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoprasian basisdata, terutama untuk pemilihan atau seleksi dan

pemasikan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis [3].

2.7.3. Notepad++

Notepad++ merupakan aplikasi teks editor yang gratis serta powerful yang dapat digunakan oleh seorang pengembang aplikasi (programmer) untuk menuliskan sebuah kode-kode program. Notepad++ mendukung banyak bahasa pemrograman , diantaranya: Assambly, C, C++, C#, CSS, HTML, Java, Javascript, Pascal, Perl, PHP, Python, Ruby, SQL dan lain sebagainya. Notepad++ ini memiliki banyak kelebihan bila dibandingkan dengan notepad bawaan windows yang pertama, seperti memiliki GUI yang baik dan menarik. Selain itu, Notepad++ juga dapat ditambahkan plugin yang bisa semakin mempermudah pekerjaan programmer [1].

2.8. Regular Expression

Regular Expressions (Regex) merupakan suatu metode yang sangat baik untuk memanipulasi data text. Pada penelitian ini *Regex* digunakan pada proses ekstraksi fitur untuk memberikan bobot pada data berbentuk text dengan cara membuat aturan *Regex.* Setiap kata akan diberikan bobot dari fitur-fitur yang ada pada tabel 2.5. Adapun simbol-simbol yang dapat digunakan untuk pembuatan aturan *Regex* sebagai berikut.

Simbol	Fungsi
/	mengawali dan mengakhiri (mengapit) *pattern*
^	mencocokkan *pattern* yang terletak pada awal subjek
$	mencocokkan *pattern* yang terletak pada akhir subjek
.	mencocokkan dengan karakter apapun, kecuali baris baru
.*?	mencocokkan dengan karakter apapun termasuk baris baru
[ ]	membuka dan menutup definisi *character class*
|	tanda pemisah dari untuk opsi alternatif
( )	membuka dan menutup sub-*pattern*
\	karakter *escape*
{x, y}	pembilang repetisi dengan nilai minimal x dan maksimal y
?	pembilang repetisi minimal nol dan maksimal satu {0, }
*	pembilang repetisi minimal nol dan maksimal tidak terbatas {0, 1}
+	pembilang repetisi minimal satu dan maksimal tidak terbatas {1, }

Gambar 2.1 Simbol Aturan Regex

2.9. CSV

Comma Separated Values (*CSV*) adalah format dasar yang terdiri dari file text yang berisi line dan value. Pada file CSV, terdapat beberapa istilah seperti *Value*, *Line*, *Header*, *Row*, dan *Cell*. *Line* adalah setiap baris *header* yang tidak termasuk sebagai row. *Cell* merupakan kolom dan row merupakan baris [4].

Pada penelitian ini, file *CSV* digunakan untuk data *training* yang akan digunakan sebagai data masukan pada proses *preprocessing training*. Tujuan menggunkan file *CSV* adalah supaya data dapat tersusun dengan rapih.

2.10. TXT

Plain Text (*TXT*) merupakan jenis teks murni yang hanya berupa karakter text tanpa ada format lainnya. Pada penelitian file dengan format *TXT* digunakan untuk data masukan proses *preprocessing testing.* Tujuan menggunkan file *TXT* karena *TXT* tidak memiliki kriteria tipe *font*, warna dan ukuran.

2.11. Data Flow Diagram (DFD)

Data Flow Diagram (DFD) adalah representasi grafik yang menggambarkan aliran informasi dan trasformasi informasi yang diaplikasikan sebagai data yang mengalir dari masukan (*input*) dan keluaran (*output*) [1].

2.12. Cnditional Random Field (CRF)

Conditional Random Field (*CRF*) adalah kerangka kerja probabilistik untuk pelabelan dan segmentasi data tersetruktur seperti urutan, pohon dan kisi. Kelebihan utama *CRF* atas *Hidden Markov Model* (*HMM*) adalah sifat kondisional (*conditional*), sehingga mengurangi ketergantungan asumsi yang dibutuhkan oleh *HMM* untuk memastikan inferensi mudah dikerjakan [4]. Selain itu, *CRF* juga menghidari masalah pada label bias, kelemahan yang ditunjukan oleh *maximum entropy markov model* (MEMM) dan model *conditional markov* lainnya yang merupakan *directed graphical models* [5]

$$P(y|x) = \frac{1}{z(x)} \prod_c \psi_c (y_c, x) \quad (2.1)$$

Conditional Random Field (CRF) merupakan model grafis yang tidak diarahkan dilatih untuk memaksimalkan probabilitas bersyarat. Model probabilistik untuk menghitung probabilitas $p(y|x)$ dari nilai *input* yang diinisialisaikan dengan $x = (x_1, x_2, \dots, x_n)$ yang juga disebut data observasi dan nilai output yang diinisialisasikan dengan $y = (y_1, y_2, \dots, y_n)$ (Lafferty et al, 2001) [5]. Berikut persamaan dari persamaan probabiltas kondisional.

$$Z(x) = \sum_y \prod_c \psi_c(y_c, x) \tag{2.2}$$

Dimana $Z(x)$ adalah konstanta normalisasi yang membuat probabilitas semua urutan state menjadi satu. Pada fungsi potensial terdapat dua jenis yaitu *node* potensial $\phi(y_t, x)$ dan *edge* potensial $\psi(y_t, y_{t+1}, x)$. Berikut ini adalah persamaan *node* potensial [9].

$$\phi_t(y_t, x) = \exp\left(\sum_k \lambda_k f_k(y_t, x, t)\right) \tag{2.3}$$

Dimana :

x : data masukan berupa kata

y_t : kelas untuk data x (contoh jenis penelitian)

λ_k : parameter fitur *node*

t : indeks fitur kata ke-t dari data input (contohnya)

$f_k(y_t, x, t)$: fitur *node* ke-k

Sedangkan untuk *edge* potensial dapat dituliskan sebagai berikut [9].

$$\psi_t(y_t, y_{t+1}, x) = \exp\left(\sum_k \lambda_{k'} f_{k'}(y_t, y_{t+1}, x, t)\right) \tag{2.4}$$

Dimana :

x : data masukan berupa kata

y_t : kelas untuk data x (contoh jenis penelitian)

y_{t+1} : kelas setelah y_t

t : indeks fitur kata ke-t dari data input (contohnya)

$\lambda_{k'}$: parameter fitur *edge*

$f_{k'}(y_t, y_{t+1}, t)$: fitur *edge* ke-k

Pada *algoritma Conditional Random Field* (CRF) terdapat beberapa proses yang akan dilakukan sebagai berikut.

2.12.1. Node dan Edge

Tahap pertama yang dilakukan pada algoritma *Conditional Random Field* (CRF) adalah menentukan bobot *node* dan *edge*. Menentukan bobot *node* dan *edge* untuk nantinya dijadikan data masukan untuk mencari node dan edge potensial. *Node* dan *edge* didapat dari ekstraksi fitur pada 2.5. *Node* merupakan asosiasi antara fitur y dengan masukan x. Berikut ini merupakan contoh dari *node* [9], yaitu:

$$f_1(y_t, x, t)\begin{cases} 1 \; if \; y_t = judul\ penelitian, x = INITCAPS \\ 0 \quad otherwise \end{cases}$$

Pada contoh *node* diatas, fitur ke-1 akan bernilai 1 jika label kelasnya jenis penelitian dan fiturnya yaitu *INITCAPS* kata yang sedang di proses adalah pembangunan, jika bukan maka akan bernilai 0.

Sedangkan *edge* merupakan asosiasi antara label kelas kata untuk kata yang sedang diproses dengan label kelas kata untuk kata setelahnya. Berikut ini adalah contoh dari *edge*, yaitu:

$$f_1(y_t, y_{t+1}, x, t)\begin{cases} 1 \; if \; y_t = judul\ penelitian, y_{t+1} = judul\ penelitian\ penelitian \\ 0 \quad otherwise \end{cases}$$

Pada contoh *edge*, fitur ke-1 diatas bernilai 1 karena label kelas kata untuk kata yang sedang diproses merupakan jenis penelitian dan untuk kelas kata setelahnya adalah jenis penelitian jadi akan bernilai 1.

2.12.2. Training CRF

Pada tahap training supaya mendapatkan nilai yang optimal pada parameter fitur node dan parameter fitur edge. Nilai parameter didapatkan dengan menggunakan *maximum likelihood* yang didapat dituliskan [9]:

$$\mathcal{L} = \sum_{t\in[1,T]}\sum_{k}\lambda_k f_k(y_t, x, t) - \sum_{t\in[1,T-1]}\sum_{k'}\lambda_{k'} f_{k'}(y_t, y_{t+1}, x, t) - \log Z(x) - \sum_{k}\frac{\lambda_k^2}{2\sigma^2} \quad (2.5)$$

Dimana :

$t \in [1,T]$: banyaknya kata

λ_k , $\lambda_{k'}$: parameter fitur ke-k

$f_k(y_t, x, t)$, $f_{k'}(y_t, y_{t+1}, x, t)$: fitur ke-k

$Z(x)$: fungsi partisi

σ : standar deviasi

Supaya mendapatkan parameter λ untuk setiap fitur, maka digunakan turunan *log-likelihood* [9], yaitu :

$$G_k^x = \sum_{t\in[1,T]}\left(f_k(y_t, x, t) - \sum_{y_t} P_t(y_t|x) f_k(y_t, x, t)\right) - \frac{\lambda_k}{\sigma^2} \quad (2.6)$$

$$G_{k'}^x = \sum_{t\in[1,T-1]}\left(f_{k'}(y_t, y_{t+1}, x, t) - \sum_{y_t}\sum_{y_{t+1}} P_t(y_t, y_{t+1}|x) f_{k'}(y_t, y_{t+1}, x, t)\right) - \frac{\lambda_{k'}}{\sigma^2} \quad (2.7)$$

Dimana :

G_k^x : nilai gradien ke-k untuk fitur *node*

$G_{k'}^x$: nilai gradien ke-k untuk fitur *edge*

$f_k(y_t, x, t)$: fitur *node* ke-k

$f_{k'}(y_t, y_{t+1}, x, t)$: fitur *edge* ke-k

$P_t(y_t|x)$: probabilitas label y_t terhadap data x

$P_t(y_t, y_{t+1}|x)$: probabilitas label y_t diikuti label y_{t+1} ketika data x

λ_k : parameter fitur *node* ke-k

$\lambda_{k'}$: parameter fitur *edge* ke-k

σ : Standar deviasi

Pada persamaan 2.6 dan 2.7 di atas adalah persamaan *stochastic gradient method* yang akan digunakan untuk mengitung $P_t(y_t|x)$ dan $P_t(y_t, y_{t+1}|x)$ pada prosedure two-pass yaitu *forward-backward* terhadap data.

1. Forward Pass

Pada tahap *forward pass* melakukan penulusuran dari indeks t=1 hingga t=T. Untuk mendapatkan nilai seluruh probabilitas lokal diperlukan penulusuran.

Untuk variabel *forward* dilambangkan oleh $\alpha_t[y_t]$. Apabila t=1 inisialisasi $\alpha_1[y_1] = 1/S$ untuk seluruh label y_t, dimana S adalah jumlah seluruh label kelas kata. Apabila $t \geq 2$ maka akan digunakan persamaan sebagai berikut [9]:

$$\alpha_t[y_t] = k_t \sum_{x_{t-1}} \alpha_{t-1}[y_{t-1}] \phi_t(y_t, x) \psi_{t-1}(y_{t-1}, y_t, x) \quad (2.8)$$

Dimana :

$\alpha_t[y_t]$: variabel *forward pass* ke-t

k_t : faktor penskalaan untuk memastikan $\sum_{y_t} \alpha_t[y_t] = 1$

$\alpha_{t-1}[y_{t-1}]$: variabel *forward* pass sebelumnya

$\phi_t(y_t, x)$: nilai *node* potensial ke-t

$\psi_{t-1}(y_{t-1}, y_t, x)$: nilai *edge* potensial sebelumnya

Persamaan diatas yaitu untuk mendapatkan nilai forward pass diperlukan data masukan seperti k_t didapat dengan membagi nilai 1 dengan seluruh jumlah hasil label kelas pada kata tersebut. $\phi_t(y_t, x)$ yaitu didapat dari persamaan 2.3, sedangkan $\psi_{t-1}(y_{t-1}, y_t, x)$ didapat dari persamaan 2.4.

2. *Backward Pass*

Pada *backward pass* hampir sama dengan *forward pass*. Perbedaannya hanya cara penelusurannya saja. Kalau backward mulai dari indeks t=T sampai t=1. Adapun variabel *backward* dilambangkan $\beta_t[y_t]$. Apabila t=T, diinisialisasikan

$\beta_T[y_T] = 1/S$ untuk semua label y_T ketika $t < T$ maka akan digunaka persamaan sebagai berikut[1]:

$$\beta_t[y_t] = \mu_t \sum_{y_{t+1}} \beta_{t+1}[y_{t+1}] \phi_{t+1}(y_{t+1}, x) \psi_t(y_t, y_{t+1}, x) \quad (2.9)$$

Dimana :

$\beta_t[y_t]$: nilai *backward pass* ke-t

μ_t : faktor penskalaan untuk memastikan $\sum_{y_t} \beta_t[y_t] = 1$

$\beta_{t+1}[y_{t+1}]$: nilai variabel *backward pass* setelahnya

$\phi_{t+1}(y_{t+1}, x)$: nilai *node* potensial setelahnya

$\psi_t(y_t, y_{t+1}, x)$: nilai *edge* potensial ke-t

Persamaan diatas yaitu untuk mendapatkan nilai forward pass diperlukan data masukan seperti $\mu_t$didapat dengan membagi nilai 1 dengan seluruh jumlah hasil label kelas pada kata tersebut. $\phi_{t+1}(y_{t+1}, x)$ yaitu didapat dari persamaan 2.3, sedangkan $\psi_t(y_t, y_{t+1}, x)$ didapat dari persamaan 2.4.

3. Probabilitas *Node* dan Probabilitas *Edge*

Supaya mendapatkan nilai probabilitas *node* yang menghitung probabilitas label kelas kata y_t terhadapa data masukan x, dapat dihitung dengan menggunakan persamaan [9]:

$$P_t(y_t|x) = w_t \alpha_t[y_t] \phi_t(y_t, x) \beta_t[y_t] \quad (2.10)$$

Dimana :

$P_t(y_t|x)$: probabilitas *node* ke-t

w_t : faktor normalisasi untuk memastikan $\sum_{y_t} P_t(y_t|x) = 1$

$\alpha_t[y_t]$: nilai *forward pass* variabel ke-t

$\phi_t(y_t, x)$: nilai *node* potensial ke-t

$\beta_t[y_t]$: nilai *backward pass* ke-t

Pada persamaan diatas, terdapat data masukan yang diperlukan seperti $\alpha_t[y_t]$ didapat dari persamaan 2.8, $\phi_t(y_t, x)$ didapat dari persamaan 3.2 dan $\beta_t[y_t]$ didapat dari persamaan 2.9.

Sedangkan untuk mengitung probabilitas edge yang menghitung probabilitas label y_t untuk data x diikuti oleh label y_{t+1} yang merupakan label kelas kata untuk kata setelahnya dapat dihitung menggunkan persamaan [9]:

$$P_t(y_t, y_{t+1}|x) = \gamma_t \alpha_t[y_t]\, \phi_t(y_t,x) \psi_t(y_t,y_{t+1},x) \phi_{t+1}(y_{t+1},x) \beta_{t+1}[y_{t+1}] \quad (2.11)$$

Dimana:

$P_t(y_t, y_{t+1}|x)$: probabilitas $edge$ ke-t

γ_t : faktor normalisasi untuk memastikan $\sum_{y_t} P_t(y_t, y_{t+1}|x) = 1$

$\alpha_t[y_t]$: nilai *forward pass* variabel ke-t

$\phi_t(y_t, x)$: nilai *node* potensial ke-t

$\psi_t(y_t, y_{t+1}, x)$: nilai *edge* potensial ke-t

$\phi_{t+1}(y_{t+1}, x)$: nilai *node* potensial setelahnya

$\beta_{t+1}[y_{t+1}]$: nilai *backward pass* setelahnya

Persamaan diatas yaitu untuk mendapatkan nilai probabilitas edge diperlukan data masukan seperti $\phi_t(y_t, x)$ yaitu didapat dari persamaan 2.3, sedangkan $\psi_t(y_t, y_{t+1}, x)$ didapat dari persamaan 2.4, $\beta_t[y_t]$ didapat dari persamaan 2.9.

2.12.3. Testing CRF

Pada tahap testing adalah tahap dimana untuk menentukan label kelas kata untuk seluruh data, bukan hanya satu node saja. Secara teori yaitu untuk menentukan label yang memungkinkan untuk diberikan terhadap data masukan. Pada langkah ini digunakan metode *Viterbi Decoding*. Ada dua langkah prosedur pada metode ini, yaitu [9]:

1. *Maximal Forward Pass*

Maximal forward pass digunakan untuk menemukan probabilitas paling optimal untuk data masukannya dengan cara mempertimbangkan hubungan antara label kelas kata untuk kata yang sedang diproses dengan label kelas kata untuk

kata setelahnya. Apabila t = 1, inisialisasi $\alpha_1^{max}[y_1] = 1/S$ untuk seluruh label y_1 apabila $t \geq 2$ maka akan digunakan rumus persamaan berikut [9].

$$\alpha_t^{max}[y_t] = k_t \max_{y_{t-1}} (\alpha_{t-1}[y_{t-1}^{max}]\phi_t(y_t, x)\psi_{t-1}(y_{t-1}, y_t, x)) \quad (2.12)$$

Dimana :

$\alpha_t^{max}[y_t]$: nilai maksimal untuk $\alpha_t[y_t]$

k_t : faktor penskalaan untuk memastikan $\sum_{y_t} \alpha_t^{max}[y_t] = 1$

$\alpha_{t-1}[y_{t-1}^{max}]$: nilai maksimal untuk label kelas kata untuk kata

$\phi_t(y_t, x)$: nilai *node* potensial ke-t

$\psi_{t-1}(y_{t-1}, y_t, x)$: nilai *edge* potensial sebelumnya

2. Backtracking Pass

Proses *backtracking* digunakan untuk menemukan label paling optimal. Adapun persamaannya adalah sebagai berikut [9].

$$y_t^* = \arg max_{y_t}(\alpha_t^{max}[y_t]\phi_t(y_t, x)) \quad (2.13)$$

Dimana :

$\alpha_t^{max}[y_t]$: nilai maksimal untuk $\alpha_t[y_t]$

$\phi_t(y_t, x)$: nilai node potensial ke-t

2.13. Nilai Akurasi

Nilai akurasi diperoleh dengan mengukur nilai kebenaran token yang diklasifikasikan, dibagi dengan jumlah token, dikali 100%. Berikut rumus perhitungan Nilai akurasi [10].

$$Akurasi\ (\%) = \frac{Keseluruhan\ data\ terklasifikasi\ dengan\ benar}{Keseluruhan\ testing\ data}\ x\ 100\%$$

Rumus diatas menjelaskan bahwa "Keseluruhan data terklasifikasi dengan benar" merupakan jumlah keseluruhan kelas yang terklasifikasikan dengan benar antara data sesungguhnya dan data prediksi. Sedangkan untuk "Keseluruhan *testing data*", merupakan jumlah keseluruhan data yang dijadikan sebagai data untuk diklasifikasikan. Pada penelitian ini, yang dimaksud dengan

data merupakan token. Jadi, pengukuran akan dilakukan terhadap token yang telah memiliki kelas.

2.14. Nilai Error

Nilai error merupakan nilai yang diperoleh ketika beberapa kelas tidak terklasifikasi dengan benar. Berikut rumus perhitungan error [10].

$$Error\ (\%) = \frac{Keseluruhan\ data\ tidak\ terklasifikasi\ dengan\ benar}{Keseluruhan\ testing\ data}\ x\ 100\%$$

Rumus diatas menjelaskan bahwa “Keseluruhan data tidak terklasifikasi dengan benar” merupakan jumlah keseluruhan data yang tidak terklasifikasikan dengan benar dari perbandingan antara data sesungguhnya dan data prediksi. Sedangkan untuk “Keseluruhan *testing data”,* merupakan jumlah dari keseluruhan data yang dijadikan sebagai data untuk diklasifikasik.

2.15. Natural Language Processing

Preprocessing adalah sebuah alat yang tepat dalam menangani banyak kasus studi *Natural Language Processing* (NLP) sangat penting dilakukan untuk memberikan akurasi yang lebih baik. Pada tahap preproccessing leksikal, seperti untuk mendeteksi kata dasar dan deteksi jenis kata berdampak pada sistem komputasi bahasa itu membutuhkkan penentuan struktur kalimat. Dalam bahasa Indonesia, penelitian stemming dan penanda POS masih dilakukan, baikdengan metode statistik atau aturan tertentu. Adapun beberapa masalah yang dihadapi untuk pengolahan tersebut adalah kuranya korpus di Indonesia dan ketidaklengkapan aturan tersedia. Penelitian tentang stemming, pertama kali diterbitkan oleh Julia Beth Lovins pada tahun 1968 [11].