BAB 3
ANALISIS DAN PERANCANGAN

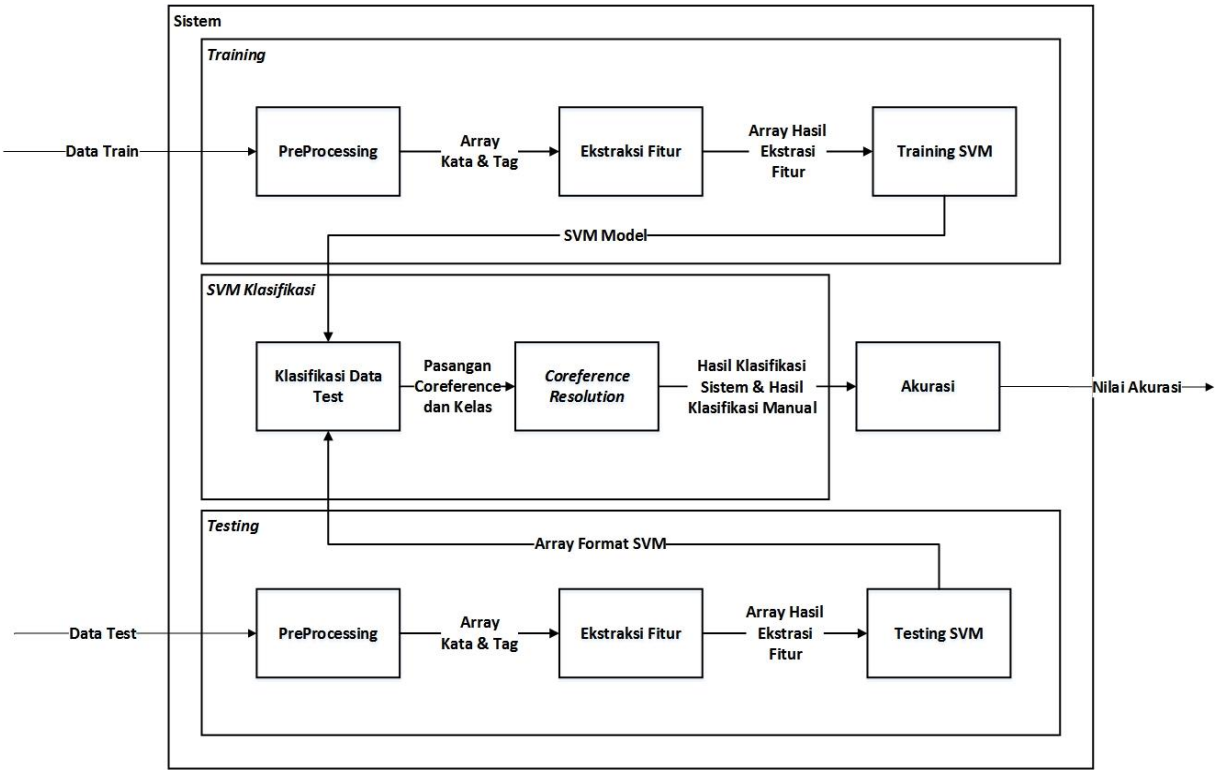
3.1. Analisis Masalah

Berdasarkan uraian pada latar belakang masalah penelitian ini, masalah yang diangkat adalah belum dilakukannya penelitian *coreference resolution* pada bahasa Indonesia dengan menggunakan metode SVM. Penelitian yang terkait dengan *coreference resolution* menggunakan metode SVM sudah pernah dilakukan sebelumnya. Penelitian yang dilakukan oleh Gao Junwei dengan judul "*Research of Noun Phrase Coreference Resolution*" bahasa yang digunakan pada penelitiannya adalah bahasa China. Penelitiannya menggunakan 15 fitur, beberapa diantaranya adalah *Distance, StringMatch, Alias, i-pronoun, j-pronoun, i-properNP, j-properNP* dan lainnya. Pada beberapa fitur yang digunakannya yang memberikan nilai cukup tinggi adalah *StringMatch, Alias* dan *Similarity.* Hasil dari penelitiannya memiliki performansi *precision* sebesar 52.97%, *recall* sebesar 49.95% dan *F-measure* sebesar 51.41% [1].

Berdasarkan hasil pengamatan terhadap novel "Surga Sungsang", terdapat banyak pengulangan nama seseorang dan kata ganti. Oleh karena itu penelitian ini menggunakan beberapa fitur yaitu *i-pronoun, i-propernoun, j-pronoun dan j-propernoun* untuk jenis kata sedangkan untuk relasi antar keduanya memanfaatkan fitur *distance of words, distance of senrences* dan *string_match*.

3.2. Analisis Sistem

Analisis sistem merupakan penguraian dari sistem utama ke dalam sub-sub sistem dengan tujuan untuk mengidentifikasi kebutuhan-kebutuhan yang diperlukan agar dapat dibangun sebuah aplikasi untuk mengetahui akurasi metode yang digunakan. Sistem yang akan dibangun memiliki beberapa tahapan. Tahapan-tahapan tersebut yaitu tahap *training*, tahap *testing*, tahap klasifikasi dengan menggunakan SVM serta penghitungan akurasi *coreference resolution* yang dihasilkan. Gambaran umum dari aplikasi yang akan dibangun dapat dilihat pada gambar 3.1 berikut ini.



Gambar 3. 1 *Gambaran Sistem*

Tahap sistem *coreference resolution* dimulai dari penerimaan teks masukan yang berupa data teks dengan format .txt yang kemudian dilakukan tahap *preprocessing* yang terdiri dari *case folding, filtering,* tokenisasi kalimat, tokenisasi kata, *stemming,* dan POS *tagging*. Hasil dari *preprocessing* berupa potongan-potongan kata dari data masukan dan informasi POS *tag* yang dimilikinya. Kemudian sistem akan memprosesnya dengan melakukan ekstraksi fitur. Beberapa fitur yang digunakan diantaranya adalah jenis kata dari *tag* yang dihasilkan diantaranya adalah *i-pronoun, i-propernoun, j-pronoun, j-propernoun* serta untuk menyatakan relasi antar keduanya adalah *distance sentence* dan *distance term.*

Selanjutya dilakukan tahap training dengan metode *Support Vector Machine* dengan memanfaatkan label.txt sebagai kunci pengklasifikasian. Dari tahap ini SVM akan menghasilkan model dari data train yang akan digunakan pada data testing. Setelah tahap training dilakukan, selanjutnya adalah tahap testing dengan inputan berupa teks novel dengan format .txt yang akan melalui tahap preprocessing dan ekstrasi fitur seperti pada tahap sebelumnya. Kemudian SVM

akan mengklasifikasikan data testing dengan SVM yang memanfaatkan train model svm yang duhasilkan sebelumnya. Pada tahap klasifikasi ini akan mengeluarkan hasil klasifikasi coreference resolution dari data testing novel yang diinputkan sebelumnya. Setelah itu adalah proses penghitungan akurasi, penghitungan ini dilakukan untuk mengetahui tingkat akurasi klasifikasi dengan menggunakan metode SVM.

3.3. Analisis Data Masukan

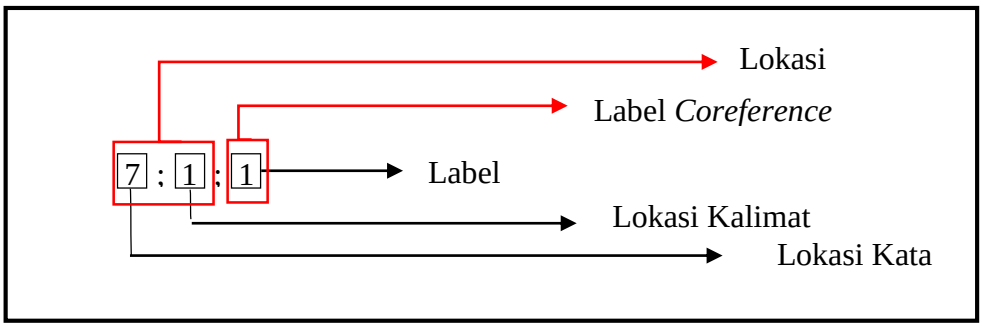
Data masukan yang diperlukan untuk menjalankan sistem ini berupa teks novel sebagai data yang akan diolah dengan format .txt. Untuk data masukan training diberikan dua data yang pertama adalah teks dari novel yang digunakan dan yang kedua adalah data label untuk menyatakan *coreference* yang akan digunakan pada tahap klasifikasi. Berikut adalah contoh teks yang akan digunakan pada tahap *training* yang merupakan potongan teks dari novel yang berjudul "*Surga Sungsang*".

> *SEJAK itu* ***Abu₁ Jenar₁*** *memang tak pernah lagi menyambangi makam. Jika* ***ia₂*** *berkunjung ke kampung itu, paling-paling* ***ia₂*** *mengajak* ***Kiai₂ Siti₂*** *dan penduduk kampung bercakap-cakap di masjid.*

Dari teks di atas terdapat tiga kata benda yang merupakan nama seseorang yaitu ***Abu₁***, ***Jenar₁***, ***Kiai₂*** dan ***Siti₂*** dan dua kata ganti yaitu ***ia₂*** dan ***ia₂***. Untuk dapat mendeteksi kata tersebut diberikan label sebagai kunci pengklasifikasian.

Label terdiri dari tiga angka yang diberi tanda ';' (titik koma) sebagai pemisah. Angka yang pertama menunjukkan lokasi kata dari sebuah kalimat, angka yang kedua menunjukkan lokasi kalimat dari sebuah teks dan angka yang terakhir merupakan label yang dapat menunjukkan bahwa kata tersebut *coreference* dengan kata lain yang memiliki label sama. Jika kata *i* dan kata *j* memiliki label yang sama, maka kata *i* dan kata *j* merupakan *coreference*. Dan jika kata *i* dan kata *j* tidak memiliki label yang sama, maka kata *i* dan kata *j* bukan *coreference.* Label dibatasi

sesuai dengan banyaknya *coreference* dalam kata tersebut. Format penulisan label yang digunakan, dapat dilihat pada gambar berikut.



Gambar 3. 2 *Data Label*

Untuk dapat melihat keterkaitan antara teks novel dan data label perhatikan tabel berikut ini.

Tabel 3. 1 Data Masukan

Teks Novel	**Teks Label**
SEJAK itu **Abu Jenar** memang tak pernah lagi menyambangi makam. 1 2 3 4 5 6 7 8 9 10 Jika **ia** berkunjung ke kampung itu, paling-paling **ia** mengajak **Kiai Siti** 1 2 3 4 5 6 7 8 9 10 11 dan penduduk kampung bercakap-cakap di masjid. 12 13 14 15 16 17	3;1;1 4;1;1 2;2;1 8;2;1 10;2;2 11;2;2

Potongan teks di atas terdiri dari dua kalimat, kalimat pertama diberi penanda berwarna merah sedangkan kalimat ke-dua diberi penanda berwarna biru. Untuk dapat mendeteksi kata ***Abu₁*** diberikan label ***3;1;1*** dimana angka pertama menunjukkan lokasi kata dari kalimat tersebut, karena kata ***Abu₁*** adalah kata ke 3 maka pada angka pertama diberikan nilai 3. Angka ke-dua menunjukkan lokasi kalimat, karena kata tersebut berada pada kalimat pertama maka pada angka ke-dua diberikan nilai 1. Selanjutnya untuk mengisi angka ke-tiga yang merupakan pelabelan kata, karena kata ***Abu₁*** merupakan kata benda orang pertama dari teks di atas maka diberikan label 1. Begitu pula pada ***Jenar₁*** dan kata ganti ***ia*** pada kalimat berikutnya diberi label 1 karena merupakan satu orang yang sama dengan ***Abu₁***.

Selanjutnya adalah kata ***Kiai$_2$*** pada kalimat pertama diberikan label ***10;2;2*** karena kata ***Kiai$_2$*** merupakan kata ke-10 dari kalimat kedua. Sedangkan untuk label karena ***Kiai$_2$*** merupakan kata benda orang ke-dua maka diberikan nilai 2 pada label. Maka diperoleh label ***Kiai$_2$*** adalah ***10;2;2***. Begitu pula pada kata ***Siti$_2$*** yang ada pada kalimat ke-dua diberikan nilai label 2 seperti kata ***Kiai$_2$*** yang karena ***Siti$_1$*** dan ***Kiai$_2$*** merupakan orang yang sama maka diberikan nilai label yang sama.

Sedangkan contoh data masukan untuk tahap *testing* digunakan potongan teks dari novel yang berjudul ”*Dilan*”. Berikut adalah potongan teks yang akan digunakan pada tahap *testing.*

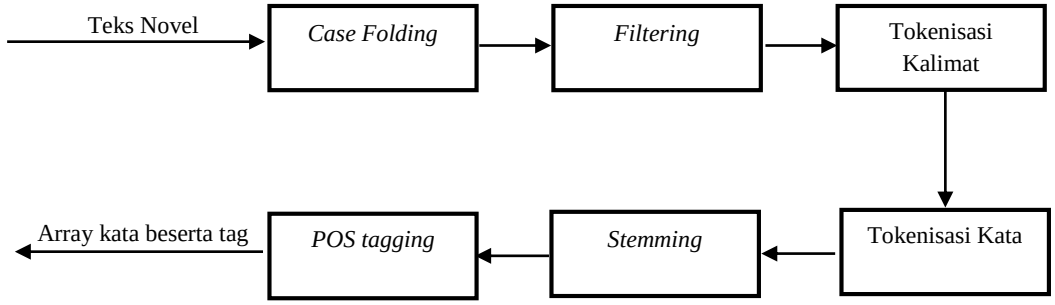
> *Dan **Burhan** membenarkan berita itu, **dia** tahu dari **Ajis**, tapi **Burhan** juga bingung karena belum tahu bagaimana kejadian sebenarnya.*

Dari teks di atas terdapat dua kata benda yang merupakan nama seseorang yaitu ***Burhan$_1$, dia, Ajis*** dan ***Burhan$_2$***. Sistem ini harus melakukan pengklasifikasian terhadap data di atas termasuk *coreference* atau bukan *coreference*.

Berdasarkan pengamatan dari data novel yang ada, maka perlu dilakukan tahap *preprocessing* sebelum melakukan klasifikasi. Adapun beberapa contoh data tersebut yang selanjutnya akan digunakan pada tahap *preprocessing.*

3.4. Analisis *Preprocessing*

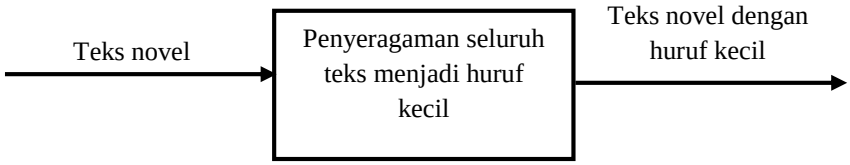
Tahap *preprocessing* merupakan tahapan untuk mempersiapkan data masukan yang akan diolah pada tahap selanjutnya. Berdasarkan dari analisis data masukan yang telah dijelaskan, maka perlu untuk melakukan tahap *preprocessing* agar dapat memaksimalkan hasil dari tahap selanjutnya. Tahap *preprocessing* pada penelitian ini terdiri dari beberapa tahapan, yaitu *case folding, filtering,* tokenisasi kalimat, tokenisasi kata, *stemming,* dan POS *tagging.* Tahap *preprocessing* dapat dilihat pada gambar di bawah ini.



Gambar 3. 3 *Blok Diagram* Preprocessing

3.4.1. *Case Folding*

Case Folding merupakan proses menyeragamkan semua huruf pada teks yaitu dengan mengubah huruf kapital (*uppercase*) menjadi huruf kecil (*lowercase*) atau sebaliknya. Gambaran proses dari tahap *case folding* dapat dilihat pada gambar berikut ini.



Gambar 3. 4 *Blok Diagram Case Folding*

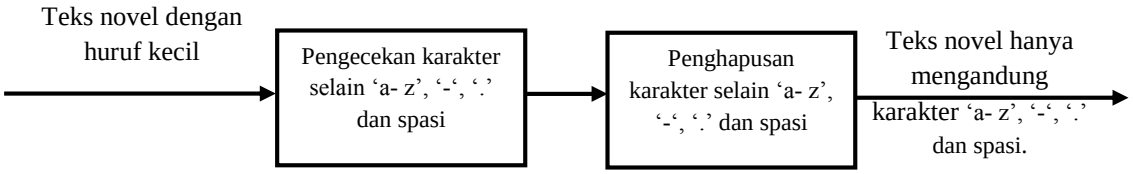
Berikut adalah contoh dokumen teks novel sebelum dan sesudah melalui tahap *case folding.*

Tabel 3. 2 Contoh Tahap *Case Folding*

Sebelum	Sesudah
SEJAK itu **A**bu **J**enar memang tak pernah lagi menyambangi makam. **J**ika ia berkunjung ke kampung itu, paling-paling ia mengajak **K**iai **S**iti dan penduduk kampung bercakap-cakap di masjid.	**sejak** itu **a**bu **j**enar memang tak pernah lagi menyambangi makam. **j**ika ia berkunjung ke kampung itu, paling-paling ia mengajak **k**iai **s**iti dan penduduk kampung bercakap-cakap di masjid.

3.4.2. *Filtering*

Tahap ini diperlukan untuk mendapatkan hasil *term* dengan tidak adanya karakter lain selain karakter ‘a’ sampai ‘z’, ‘-‘, ‘.’ dan spasi. Contohnya kata ***itu,*** pada teks di atas akan menjadi ***itu*** karena karakter ‘***,***’ bukan termasuk batasan karakter dari tahap *filtering* maka karakter tersebut akan dihapus. Berikut tahapan yang dilakukan dapat dilihat pada gambar berikut.



Gambar 3. 5 *Blok Diagram* Filtering

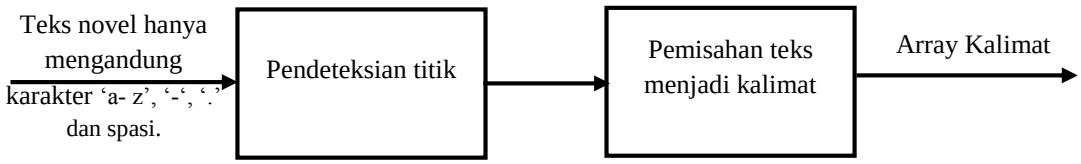
Berikut adalah contoh dokumen teks novel sebelum dan sesudah melalui tahap f*iltering*.

Tabel 3. 3 Contoh Filtering

Sebelum	Sesudah
sejak itu abu jenar memang tak pernah lagi menyambangi makam. jika ia berkunjung ke kampung itu, paling-paling ia mengajak kiai siti dan penduduk kampung bercakap-cakap di masjid.	sejak itu abu jenar memang tak pernah lagi menyambangi makam. jika ia berkunjung ke kampung itu paling-paling ia mengajak kiai siti dan penduduk kampung bercakap-cakap di masjid.

3.4.3 Tokenisasi Kalimat

Tokenisasi kalimat digunakan untuk memisahkan teks ke dalam bentuk satuan kalimat. Sehingga teks akan dijadikan sebagai ukuran untuk penentuan fitur *distance of sentences* pada tahap ekstraksi fitur. Tokenisasi kalimat pada sistem ini menggunakan delimeter titik (“.”). Proses tokenisasi kalimat dapat dilihat pada gambar berikut ini.



Gambar 3. 6 Blok Diagram Tokenisasi Kalimat

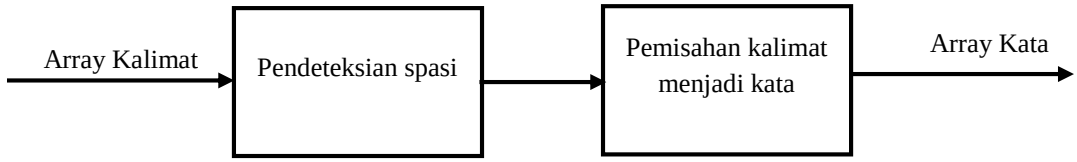
Berikut adalah contoh dokumen teks novel sebelum dan sesudah melalui tahap tokenisasi kalimat.

Tabel 3. 4 Contoh Tokenisasi Kalimat

Sebelum	**Sesudah**	
sejak itu abu jenar memang tak pernah lagi menyambangi makam. jika ia berkunjung ke kampung itu paling-paling ia mengajak kiai siti dan penduduk kampung bercakap-cakap di masjid.	**No**	**Kalimat**
	1	sejak itu abu jenar memang tak pernah lagi menyambangi makam.
	2	jika ia berkunjung ke kampung itu paling-paling ia mengajak kiai siti dan penduduk kampung bercakap-cakap di masjid.

3.4.4. Tokenisasi Kata

Tokenisasi kata digunakan untuk memisahkan dari setiap kalimat menjadi satuan kata. Sehingga teks akan dijadikan sebagai ukuran untuk penentuan fitur *distance of words* pada tahap ekstraksi fitur. Pemisahan kalimat menggunakan delimeter spasi (‘ ’). Tokenisasi kata dilakukan untuk membuat kata menjadi *array* sehingga memudahkan sistem dalam proses POS *tagging.* Proses tokenisasi kata dapat dilihat pada gambar berikut.



Gambar 3. 7 Blok Diagram Tokenisasi Kata

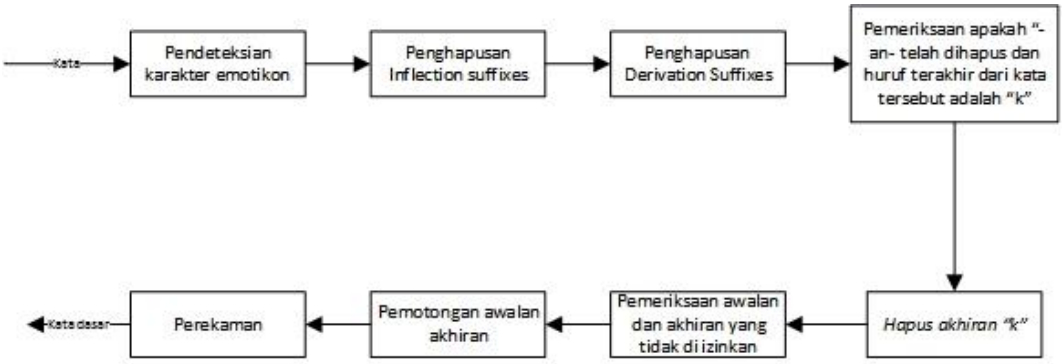
Berikut adalah contoh dokumen teks novel sebelum dan sesudah melalui tahap tokenisasi kata.

Tabel 3. 5 Contoh Tokenisasi Kata

No	Sebelum Tokenisasi	Sesudah Tokenisasi		
1	sejak itu abu jenar memang tak pernah lagi menyambangi makam.	sejak	memang	menyambangi
		itu	tak	makam.
		abu	pernah	
		jenar	lagi	
2	jika ia berkunjung ke kampung itu paling-paling ia mengajak kiai siti dan penduduk kampung bercakap-cakap di masjid.	jika	paling-paling	penduduk
		ia	ia	kampong
		berkunjung	mengajak	bercakap-cakap
		ke	kiai	di
		kampung	siti	masjid.
		itu	dan	

3.4.5. *Stemming*

Dalam sebuah novel terdapat beberapa penggunaan imbuhan di setiap katanya. *Stemming* digunakan untuk mengubah kata berimbuhan menjadi kata dasar. Pada proses ini menggunakan *library stemmer* untuk bahasa Indonesia dari sastrawi yang telah mengadopsi aturan Nazief & Andriani [15]. Sebagai contoh, kata ***memberi*** memiliki kata dasar yaitu ***beri***. Proses *stemming* dapat dilihat pada gambar berikut.



Gambar 3. 8 Blok Diagram *Stemming*

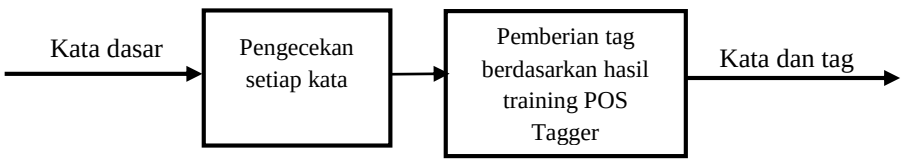
Berikut adalah contoh dokumen teks novel sebelum dan sesudah melalui tahap *stemming*.

Tabel 3. 6 Contoh *Stemming*

No	Sebelum	Sesudah
1	sejak	sejak
	itu	itu
	abu	abu
	jenar	jenar
	memang	memang
	tak	tak
	pernah	pernah
	lagi	lagi
	menyambangi	**sambang**
	makam.	makam.
2	jika	jika
	ia	ia
	berkunjung	**kunjung**
	ke	ke
	kampung	kampung
	itu	itu
	paling-paling	**paling**
	ia	ia
	mengajak	**ajak**
	kiai	kiai
	siti	siti
	dan	dan
	penduduk	**duduk**
	kampung	kampung
	bercakap-cakap	**cakap**
	di	di
	masjid.	masjid

3.4.6. POS *Tagging*

POS *tagging* adalah proses pemberian *tag* jenis kata seperti kata ***siti*** merupakan kata benda nama orang maka *tag* yang diberikan oleh *tagger* adalah ***NNP***. POS *Tagger* yang digunakan untuk mengidentifikasi jenis kata dan entitasnya pada penelitian ini menggunakan aplikasi Indonesia NER. Berikut tahapan yang dilakukan dapat dilihat pada gambar berikut.



Gambar 3. 9 Blok Diagram POS Tagging

Berikut adalah contoh dokumen teks novel sebelum dan sesudah melalui tahap POS *tagging*.

Tabel 3. 7 Contoh POS Tagging

ID	Kata	POS Tag
1	sejak	IN
	itu	DT
	abu	NNP
	jenar	NNP
	memang	RB
	tak	NEG
	pernah	NN
	lagi	RB
	sambang	VBT
	makam	NN
2	jika	SC
	ia	PRP
	kunjung	VBI
	ke	IN
	kampung	NN
	itu	DT
	paling	RB
	ia	PRP
	ajak	VBT
	kiai	NNP
	siti	NNP
	dan	CC
	duduk	VBI
	kampung	NN
	cakap	NN
	di	IN
	masjid	NN

3.5. Ekstraksi Fitur

Fitur merupakan atribut yang digunakan untuk mewakili data yang digunakan. Kata kunci digunakan untuk mengekstak fitur, dimana fitur merupakan hasil ekstraksi yang diasumsikan memberikan informasi mengenai data. Pada penelitian ini digunakan 7 fitur, dimana fitur-fitur tersebut telah digunakan pada penelitian lain seperti pada penelitian yang dilakukan oleh Gao Junwey [1]. Keenam fitur yang dijelaskan di bawah ini akan digunakan untuk proses klasifikasi menggunakan SVM.

Tabel 3. 8 Fitur *Coreference Resolution*

Fitur yang mendeskripsikan co-1(i)	
i-Pronoun	Mendeskripsikan nilai i yang dideteksi oleh POS tag dari entitasnya adalah PRP. Jika i adalah kata ganti nilainya adalah 1 jika tidak maka nilainya 0.
i-Proper Noun Phrase	Mendeskripsikan nilai i yang dideteksi oleh POS tag dari entitasnya adalah NNP. Jika i adalah proper noun nilainya adalah 1 jika tidak maka nilainya 0.
Fitur yang mendeskripsikan co-2 (j)	
j-Pronoun	Mendeskripsikan nilai j yang dideteksi oleh POS tag dari entitasnya adalah PRP. Jika j adalah kata ganti nilainya adalah 1 jika tidak maka nilainya 0.
j-Proper Noun Phrase	Mendeskripsikan nilai j yang dideteksi oleh POS tag dari entitasnya adalah NNP. Jika j adalah proper noun nilainya adalah 1 jika tidak maka nilainya 0.
Fitur untuk relasi antar keduanya	
Distance Of Words (DOW)	Mendeskripsikan jarak kalimat antara i dan j. Nilai yang akan mengisi pada fitur ini adalah letak kata *j* dikurangi kata ke *i* lalu dibagi jumlah kata pada dokumen.
Distance Of Sentences (DOS)	Mendeskripsikan jarak kalimat antara i dan j. Nilai yang akan mengisi pada fitur ini adalah letak kalimat pada kata *j* dikurangi letak kalimat pada kata ke *i* lalu dibagi jumlah kalimat pada dokumen.
String_match	Mendeskripsikan string pada i dan j. Jika string i sesuai dengan string j, maka nilainya adalah 1 jika tidak nilainya 0.

Ekstraksi fitur terbagi menjadi dua yaitu ekstraksi fitur untuk data *training* dan ekstraksi fitur untuk data *testing.* Berikut adalah contoh data masukan untuk training.

SEJAK itu ***Abu₁ Jenar₁*** *memang tak pernah lagi menyambangi makam. Jika* ***ia₂*** *berkunjung ke kampung itu, paling-paling* ***ia₂*** *mengajak* ***Kiai₂ Siti₂*** *dan penduduk kampung bercakap-cakap di masjid.*

Pada ekstraksi fitur data *training* memanfaatkan label.txt untuk membuat pasangan *i* dan *j.* Jika kata *i* memiliki data label yang berbeda dengan kata *j* maka

akan dijadikan pasangan. Contohnya adalah kata ***Abu₁*** memiliki label ***3;1;1*** dan ***Jenar₁*** memiliki label ***4;1;1*** karena label yang dimiliki ***Abu₁*** dan ***Jenar₁*** berbeda maka ***Abu₁*** dan ***Jenar₁*** akan dijadikan pasangan *i* dan *j*. Dimana *i* adalah ***Jenar₁*** dan *j* adalah ***Abu₁*** karena letak *j* harus berada setelah kata *i* (*j* > *i*) .

Setelah tahap *preprocessing* maka didapatkan kata dan tag. Berdasarkan label yang dimiliki maka sistem akan mencari kata yang dimaksud. ***Jenar₁*** sebagai *i* memiliki tag NNP sehingga fitur i-pronoun diberi nilai 0 dan i-propernoun diberi nilai 1. Begitu pun dengan ***Abu₁*** sebagai *j, k*arena ***Abu₁*** memiliki tag NNP sehingga fitur j-pronoun diberi nilai 0 dan j-propernoun diberi nilai 1. Untuk mencari nilai *Distance Of Words* dengan rumus sebagai berikut. Dengan jarak kata *i* didapat dari jarak kata *i* dari awal kalimat, begitupun dengan kata *j*.

$$DOW = \frac{\text{jarak kata } i - \text{ jarak kata } j}{jumlah\ kata\ dalam\ teks\ novel} = \frac{4 - 3}{27} = 0.04$$

Sedangkan untuk mencari nilai *Distance Of Sentences* dengan rumus sebagai berikut. Dengan jarak kalimat *i* didapat dari jarak kalimat *i* dari awal teks novel, begitupun dengan kalimat *j*.

$$DOS = \frac{\text{jarak kalimat } i - \text{ jarak kalimat } j}{jumlah\ kalimat\ dalam\ teks\ novel} = \frac{0}{2} = 0$$

Berikut adalah contoh hasil ekstraksi fitur dari data *training* yang sudah melalui tahap *preprocessing.*

Tabel 3. 9 Contoh Ekstraksi Fitur data *training*

i	***Tag - i***	***j***	***Tag - j***	**Fitur *i***		**Fitur *j***		**Fitur Relasi**		
				***i*- Pron**	***i*- Prop**	***j*- Pron**	***j*- Prop**	**DOW**	**DOS**	**Str_ match**
(jenar) 1	NNP	(abu) 1	NNP	0	1	0	1	0.04	0	0
(ia) 2	PRP	(abu) 1	NNP	1	0	0	1	0.33	0.50	0
(ia) 2	PRP	(jenar) 1	NNP	1	0	0	1	0.30	0.50	0
(ia) 2	PRP	(abu) 1	NNP	1	0	0	1	0.56	0.50	0
(ia) 2	PRP	(jenar) 1	NNP	1	0	0	1	0.52	0.50	0
(ia) 2	PRP	(ia) 2	PRP	1	0	1	0	0.22	0	1
(kiai) 2	NNP	(abu) 1	NNP	0	1	0	1	0.63	0.50	0
(kiai) 2	NNP	(jenar) 1	NNP	0	1	0	1	0.59	0.50	0

(kiai) 2	NNP	(ia) 2	PRP	0	1	1	0	0.30	0	0
(kiai) 2	NNP	(ia) 2	PRP	0	1	1	0	0.07	0	0
(siti) 2	NNP	(abu) 1	NNP	0	1	0	1	0.67	0.50	0
(siti) 2	NNP	(jenar) 1	NNP	0	1	0	1	0.63	0.50	0
(siti) 2	NNP	(ia) 2	PRP	0	1	1	0	0.33	0	0
(siti) 2	NNP	(ia) 2	PRP	0	1	1	0	0.11	0	0
(siti) 2	NNP	(kiai) 2	NNP	0	1	0	1	0.04	0	0

Sedangkan ekstraksi fitur untuk data *testing*, pasangan *i* dan *j* dicari berdasarkan hasil POS *tagging.* Jika kata tersebut memiliki tag NNP untuk kata benda nama orang dan PRP untuk kata ganti maka akan dijadikan kandidat pasangan *i* dan *j.* Berikut adalah contoh data masukan untuk *testing.*

*Dan **Burhan$_1$** membenarkan berita itu, **dia$_1$** tahu dari **Ajis$_1$**, tapi **Burhan$_1$** juga bingung karena belum tahu bagaimana kejadian sebenarnya.*

Berikut adalah hasil ekstraksi fitur dari data *testing* di atas.

Tabel 3. 10 Contoh Ekstraksi Fitur data *testing*

i	***Tag - i***	***j***	***Tag - j***	**Fitur *i***		**Fitur *j***		**Fitur Relasi**		
				***i*- Pron**	***i*- Prop**	***j*- Pron**	***j*- Prop**	**DOW**	**DOS**	**Str_ match**
(dia) 1	PRP	(burhan) 1	NNP	1	0	0	1	0.21	0	0
(ajis) 1	NNP	(burhan) 1	NNP	0	1	0	1	0.37	0	0
(ajis) 1	NNP	(dia) 1	PRP	0	1	1	0	0.16	0	0
(burhan) 1	NNP	(burhan) 1	NNP	0	1	0	1	0.47	0	1
(burhan) 1	NNP	(dia) 1	PRP	0	1	1	0	0.26	0	0
(burhan) 1	NNP	(ajis) 1	NNP	0	1	0	1	0.11	0	0

3.6. Analisis Klasifikasi Menggunakan Metode SVM

Pada tahap klasifikasi, setiap pasangan *i* dan *j* akan diberikan label positif dan negatif. Label positif digunakan untuk pasangan yang merupakan *coreference* sedangkan label negatif digunakan untuk pasangan yang merupakan *not-coreference*. Pada tahap klasifikasi tersebut menggunakan metode *Support Vector Machine.*

3.6.1. SVM Training

Pada tahap ini, data training dibagi menjadi dua yaitu teks novel dan label. Label dibuat dengan setiap pasangan co-1 dan co-2 yang merupakan coreference diberikan label yang sama. Sebagai cotoh, dapat dilihat pada di bawah ini.

Tabel 3. 11 Contoh Data Training dalam Format SVM

Format SVM (x_i)	Label (y_i)
0 1 0 1 0.04 0 0	1
1 0 0 1 0.33 0.50 0	1
1 0 0 1 0.30 0.50 0	1
1 0 0 1 0.56 0.50 0	1
1 0 0 1 0.52 0.50 0	1
1 0 1 0 0.22 0 1	1
0 1 0 1 0.63 0.50 0	-1
0 1 0 1 0.59 0.50 0	-1
0 1 1 0 0.30 0 0	-1
0 1 1 0 0.07 0 0	-1
0 1 0 1 0.67 0.50 0	-1
0 1 0 1 0.63 0.50 0	-1
0 1 1 0 0.33 0 0	-1
0 1 1 0 0.11 0 0	-1
0 1 0 1 0.04 0 0	1

Setelah diubah menjadi format SVM maka dilakukan tahap kernelisasi dengan menggunakan kernel *linier* yang didefinisikan K($x_i x_j$)= $x_i^T x_j$. Dimana *x* merupakan hasil ekstraksi fitur yang telah diubah menjadi format SVM yang dapat dilihat pada tabel 3.11 di atas. Lakukan perhitungan dengan rumus $x_i^T x_i$ hingga $i = n$.

$$x_1{}^T x_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}^T \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$$

$$= [1\ 0\ 0\ 1\ 0.04\ 0\ 0] \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$$

$$= 1 + 0 + 0 + 1 + 0.002 + 0 + 0$$
$$= 2.002$$

Berikut adalah hasil dari perhitungan di atas.

Tabel 3. 12 Hasil Perhitungan $x_i^T x_j$ pada *Training*

i	x_i	x_i^T	j	x_j	$x_i^T x_j$
1	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$	[0 1 0 1 0.04 0 0]	1	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$	2.002
			2	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.33 \\ 0.50 \\ 0 \end{bmatrix}$	1.013
			3	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.30 \\ 0.50 \\ 0 \end{bmatrix}$	1.012
			4	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.56 \\ 0.50 \\ 0 \end{bmatrix}$	1.022
			5	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.52 \\ 0.50 \\ 0 \end{bmatrix}$	1.021
			6	$\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0.22 \\ 0 \\ 1 \end{bmatrix}$	0.009

			7	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.63 \\ 0.50 \\ 0 \end{bmatrix}$	2.025
			8	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.59 \\ 0.50 \\ 0 \end{bmatrix}$	2.023
			9	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.30 \\ 0 \\ 0 \end{bmatrix}$	1.012
			10	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.07 \\ 0 \\ 0 \end{bmatrix}$	1.003
			11	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.67 \\ 0.50 \\ 0 \end{bmatrix}$	2.027
			12	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.63 \\ 0.50 \\ 0 \end{bmatrix}$	2.025
			13	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.33 \\ 0 \\ 0 \end{bmatrix}$	1.013

			14	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.11 \\ 0 \\ 0 \end{bmatrix}$	1.004
			15	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$	2.002
		. . .			
15	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$	[0 1 0 1 0.04 0 0]	1	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$	2.002
			2	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.33 \\ 0.50 \\ 0 \end{bmatrix}$	1.013
			3	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.30 \\ 0.50 \\ 0 \end{bmatrix}$	1.012
			4	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.56 \\ 0.50 \\ 0 \end{bmatrix}$	1.022
			5	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.52 \\ 0.50 \\ 0 \end{bmatrix}$	1.021

			6	$\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0.22 \\ 0 \\ 1 \end{bmatrix}$	0.009
			7	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.63 \\ 0.50 \\ 0 \end{bmatrix}$	2.025
			8	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.59 \\ 0.50 \\ 0 \end{bmatrix}$	2.024
			9	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.30 \\ 0 \\ 0 \end{bmatrix}$	1.012
			10	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.07 \\ 0 \\ 0 \end{bmatrix}$	1.003
			11	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.67 \\ 0.50 \\ 0 \end{bmatrix}$	2.027
			12	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.63 \\ 0.50 \\ 0 \end{bmatrix}$	2.025

			13	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.33 \\ 0 \\ 0 \end{bmatrix}$	1.013
			14	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.11 \\ 0 \\ 0 \end{bmatrix}$	1.004
			15	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$	2.002

Setelah dilakukan pada seluruh nilai x, maka matriks yang terbentuk dari hasil perhitungan adalah sebagai berikut.

$$x_i^T x_j = \begin{bmatrix} x_1^T x_1 & \cdots & x_1^T x_{15} \\ \vdots & \ddots & \vdots \\ x_{15}^T x_1 & \cdots & x_{15}^T x_{15} \end{bmatrix}$$

$$= \begin{bmatrix} 2.002 & 1.013 & 1.012 & 1.022 & 1.021 & 0.009 & 2.052 & 2.024 & 1.012 & 1.003 & 2.027 & 2.025 & 1.013 & 1.004 & 2.002 \\ 1.013 & 2.359 & 2.349 & 2.435 & 2.422 & 1.073 & 1.458 & 1.445 & 0.099 & 0.023 & 1.471 & 1.458 & 0.109 & 0.036 & 1.013 \\ 1.012 & 2.349 & 2.34 & 2.418 & 2.406 & 1.066 & 1.439 & 1.427 & 0.09 & 0.021 & 1.451 & 1.439 & 0.099 & 0.033 & 1.012 \\ 1.022 & 2.435 & 2.418 & 2.564 & 2.541 & 1.123 & 1.603 & 1.58 & 0.168 & 0.039 & 1.625 & 1.603 & 0.185 & 0.062 & 1.022 \\ 1.021 & 2.435 & 2.406 & 2.541 & 2.520 & 1.114 & 1.578 & 1.557 & 0.156 & 0.036 & 1.598 & 1.578 & 0.172 & 0.057 & 1.021 \\ 0.009 & 1.073 & 1.066 & 1.123 & 1.114 & 3.048 & 0.139 & 0.13 & 1.066 & 1.015 & 0.147 & 0.139 & 1.073 & 1.024 & 0.009 \\ 2.025 & 1.458 & 1.439 & 1.603 & 1.578 & 0.139 & 2.647 & 2.622 & 1.189 & 1.044 & 2.672 & 2.647 & 1.208 & 1.069 & 2.025 \\ 2.024 & 1.445 & 1.427 & 1.580 & 1.557 & 0.13 & 2.622 & 2.598 & 1.177 & 1.041 & 2.645 & 2.622 & 1.195 & 1.065 & 2.024 \\ 1.012 & 0.099 & 0.09 & 0.168 & 0.156 & 1.066 & 1.189 & 1.177 & 2.09 & 2.021 & 1.201 & 1.189 & 2.099 & 2.033 & 1.012 \\ 1.003 & 0.023 & 0.021 & 0.039 & 0.036 & 1.015 & 1.044 & 1.041 & 2.021 & 2.005 & 1.047 & 1.044 & 2.023 & 2.008 & 1.003 \\ 2.027 & 1.471 & 1.451 & 1.625 & 1.598 & 0.147 & 2.672 & 2.645 & 1.201 & 1.047 & 2.699 & 2.672 & 1.221 & 1.074 & 2.027 \\ 2.025 & 1.458 & 1.439 & 1.603 & 1.578 & 0.139 & 2.647 & 2.622 & 1.189 & 1.044 & 2.672 & 2.647 & 1.208 & 1.069 & 2.025 \\ 1.013 & 0.109 & 0.099 & 0.185 & 0.172 & 1.073 & 1.208 & 1.195 & 2.099 & 2.023 & 1.221 & 1.208 & 2.109 & 2.036 & 1.013 \\ 1.004 & 0.036 & 0.033 & 0.062 & 0.057 & 1.024 & 1.069 & 1.065 & 2.033 & 2.008 & 1.074 & 1.069 & 2.036 & 2.012 & 1.004 \\ 2.002 & 1.013 & 1.012 & 1.022 & 1.021 & 0.009 & 2.025 & 2.024 & 1.012 & 1.003 & 2.027 & 2.025 & 1.013 & 1.004 & 2.002 \end{bmatrix}$$

Tahap selanjutnya adalah melakukan perhitungan terhadap y_i dengan melakukan hal yang sama seperti perhitungan terhadap nilai x_i. Untuk nilai y_i adalah nilai dari label yang telah diberikan, dapat dilihat pada tabel 3.11. Lakukan perhitungan sebagai berikut dengan rumus $y_i^T y_i$ hingga $i = n$.

$$y_1^T y_1 = [1]^T [1] = [1] [1] = 1$$

Berikut adalah hasil dari perhitungan di atas.

Tabel 3. 13 Hasil perhitungan $y_i{}^T y_j$ pada *Testing*

i	y_i	$y_i{}^T$	j	y_j	$y_i{}^T y_j$
1	[1]	[1]	1	[1]	1
			2	[1]	1
			3	[1]	1
			4	[1]	1
			5	[1]	1
			6	[1]	1
			7	[−1]	-1
			8	[−1]	-1
			9	[−1]	-1
			10	[−1]	-1
			11	[−1]	-1
			12	[−1]	-1
			13	[−1]	-1
			14	[−1]	-1
			15	[1]	1
		. . .			
15	[1]	[1]	1	[1]	1
			2	[1]	1
			3	[1]	1
			4	[1]	1
			5	[1]	1
			6	[1]	1
			7	[−1]	-1
			8	[−1]	-1
			9	[−1]	-1
			10	[−1]	-1
			11	[−1]	-1
			12	[−1]	-1
			13	[−1]	-1
			14	[−1]	-1
			15	[1]	1

Sehingga matriks yang terbentuk dari hasil perhitungan $y_i^T y_j$ hasilnya adalah sebagai berikut.

$$y_i^T y_j = \begin{bmatrix} y_1^T y_1 & \cdots & y_1^T y_{15} \\ \vdots & \ddots & \vdots \\ y_{15}^T y_1 & \cdots & y_{15}^T y_{15} \end{bmatrix}$$

$$= \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 \\
-1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 \\
-1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 \\
-1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 \\
-1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 \\
-1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 \\
-1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 \\
-1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 \\
1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1
\end{bmatrix}$$

Selanjutnya adalah tahap mengubah setiap pasangan *i* dan *j* menjadi nilai vektor (*support vector*) = $\begin{bmatrix} x \\ y \end{bmatrix}$ untuk mendapatkan nilai a_i. Nilai *x* didapatkan menggunakan persamaan 2.19 kernel linear sebagai berikut.

$$\sum_{i=1,j=1}^{n} x_i^T x_j \ (i, = 1,..,\text{n}) \qquad (2.19)$$

$$x_1 = x_1^T x_1 + x_1^T x_2 + \cdots + x_1^T x_{15} = 2.002 + 1.013 + \cdots + 2.002 = 20.214$$

$$x_2 = x_2^T x_1 + x_2^T x_2 + \cdots + x_1^T x_{15} = 1.013 + 2.359 + \cdots + 1.013 = 18.762$$

$$\vdots$$

$$x_{15} = x_1^T x_{15} + x_2^T x_{15} + \cdots + x_{15}^T x_{15} = 2.002 + 1.013 + \cdots + 2.002 = 20.191$$

Sehingga didapatkan nilai x untuk setiap pasangan sebagai berikut.

Tabel 3. 14 Nilai *x* Untuk Setiap Pasangan

x_i	Nilai
x_1	20.214
x_2	18.762
x_3	18.602
x_4	19.99
x_5	19.777
x_6	12.175
x_7	25.364

x_8	25.151
x_9	16.602
x_{10}	15.374
x_{11}	25.578
x_{12}	25.364
x_{13}	16.762
x_{14}	15.587
x_{15}	20.191

Begitu pun dengan nilai *y* didapatkan menggunakan persamaan 2.20 kernel *linear* untuk y_i yaitu sebagai berikut.

$$\sum_{i=1,j=1}^{n} y_i^T y_j \ (i, = 1, \dots, \mathrm{n}) \qquad (2.20)$$

$$y_1 = y_1^T y_1 + y_1^T x_2 + \cdots + y_1^T x_3 = 1 + 1 + \cdots + 1 = -1$$

$$y_2 = y_2^T y_1 + y_2^T x_2 + \cdots + y_1^T x_3 = 1 + 1 + \cdots + 1 = -1$$

$$\vdots$$

$$y_{15} = y_{15}^T y_1 + y_{15}^T x_2 + \cdots + y_{15}^T x_{15} = 1 + 1 + \cdots + 1 = -1$$

Sehingga didapatkan nilai y_i untuk setiap pasangan sebagai berikut.

Tabel 3. 15 Nilai *y* Untuk Setiap Pasangan

y_i	**Nilai**
y_1	-1
y_2	-1
y_3	-1
y_4	-1
y_5	-1
y_6	-1
y_7	1
y_8	1
y_9	1
y_{10}	1
y_{11}	1
y_{12}	1

y_{13}	1
y_{14}	1
y_{15}	-1

Setelah didapatkan nilai x_i dan y_i, lalu subtitusi nilai x_i dan y_i ke dalam persamaan trick phi 2.18 berikut.

$$\phi \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{cases} \sqrt{{x_n}^2 + {y_n}^2} > 2, maka \begin{bmatrix} \sqrt{{x_n}^2 + {y_n}^2} - x_i + |x_i - y_i| \\ \sqrt{{x_n}^2 + {y_n}^2} - y_i + |x_i - y_i| \end{bmatrix} \\ \sqrt{{x_n}^2 + {y_n}^2} \leq 2, maka \begin{bmatrix} x_i \\ y_i \end{bmatrix} \end{cases} \quad (2.18)$$

Nilai x_n yang didapat dari x_{15} dan y_n dari y_{15} yang disubtitusikan ke dalam persamaan sebagai berikut.

$$\sqrt{{x_{15}}^2 + {y_{15}}^2} = \sqrt{20.191^2 + (-1)^2} = 20.216$$

Karena $\sqrt{{x_n}^2 + {y_n}^2} > 2$, maka $\phi \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} \sqrt{{x_n}^2 + {y_n}^2} - x_i + |x_i - y_i| \\ \sqrt{{x_n}^2 + {y_n}^2} - y_i + |x_i - y_i| \end{bmatrix}$

$$\phi \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} \sqrt{{x_{15}}^2 + {y_{15}}^2} - x_1 + |x_1 - y_1| \\ \sqrt{{x_{15}}^2 + {y_{15}}^2} - y_1 + |x_1 - y_1| \end{bmatrix}$$

$$= \begin{bmatrix} \sqrt{20.191^2 + (-1)^2} - 20.214 + |20.214 - (-1)| \\ \sqrt{20.191^2 + (-1)^2} - (-1) + |20.214 - (-1)| \end{bmatrix}$$

$$= \begin{bmatrix} 21.216 \\ 42.43 \end{bmatrix}$$

Setelah itu masing-masing *support vector* diberi nilai bias 1, untuk mendapatkan jarak tegak lurus yang optimal dengan mempertimbangkan vektor positif, serta membantu mendapatkan nilai b atau nilai *hyperplane* yaitu sebagai berikut. Sehingga didapatkan hasil sebagai berikut.

Tabel 3. 16 Tabel ϕ untuk setiap pasangan

x_n^2	y_n^2	$\sqrt{x_n^2 + y_n^2}$	$\phi \begin{bmatrix} x_i \\ y_i \end{bmatrix}$	b	Si	$\phi \begin{bmatrix} x_i \\ y_i \\ b \end{bmatrix}$
407.685	-1	20.216	$\begin{bmatrix} 21.216 \\ 42.43 \end{bmatrix}$	1	S_1	$\begin{bmatrix} 21.216 \\ 42.43 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 21.216 \\ 40.978 \end{bmatrix}$	1	S_2	$\begin{bmatrix} 21.216 \\ 40.978 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 21.216 \\ 40.818 \end{bmatrix}$	1	S_3	$\begin{bmatrix} 21.216 \\ 40.818 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 21.216 \\ 42.206 \end{bmatrix}$	1	S_4	$\begin{bmatrix} 21.216 \\ 42.206 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 21.216 \\ 41.993 \end{bmatrix}$	1	S_5	$\begin{bmatrix} 21.216 \\ 41.993 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 21.216 \\ 34.391 \end{bmatrix}$	1	S_6	$\begin{bmatrix} 21.216 \\ 34.391 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 19.216 \\ 43.58 \end{bmatrix}$	1	S_7	$\begin{bmatrix} 19.216 \\ 43.58 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 19.216 \\ 43.367 \end{bmatrix}$	1	S_8	$\begin{bmatrix} 19.216 \\ 43.367 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 19.216 \\ 34.818 \end{bmatrix}$	1	S_9	$\begin{bmatrix} 19.216 \\ 34.818 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 19.216 \\ 33.59 \end{bmatrix}$	1	S_{10}	$\begin{bmatrix} 19.216 \\ 33.59 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 19.216 \\ 43.794 \end{bmatrix}$	1	S_{11}	$\begin{bmatrix} 19.216 \\ 43.794 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 19.216 \\ 43.58 \end{bmatrix}$	1	S_{12}	$\begin{bmatrix} 19.216 \\ 43.58 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 19.216 \\ 34.978 \end{bmatrix}$	1	S_{13}	$\begin{bmatrix} 19.216 \\ 34.978 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 19.216 \\ 33.803 \end{bmatrix}$	1	S_{14}	$\begin{bmatrix} 19.216 \\ 33.803 \\ 1 \end{bmatrix}$
			$\begin{bmatrix} 21.216 \\ 42.407 \end{bmatrix}$	1	S_{15}	$\begin{bmatrix} 21.216 \\ 42.407 \\ 1 \end{bmatrix}$

Setelah mendapat nilai *support vector bias*, kemudian tahap selanjutnya adalah mencari nilai a_i, didapatkan dengan mengalikan setiap data vektor menggunakan persamaan 2.22 yaitu sebagai berikut.

$$\sum_{i=1,j=1}^{n} a_i {S_i}^T S_j = y_i \qquad (2.22)$$

Lakukan perhitungan berikut dengan rumus di atas hingga $i = n$.

$$a_1 {S_1}^T S_1 = a_1 \begin{bmatrix} 21.216 \\ 42.43 \\ 1 \end{bmatrix}^T \begin{bmatrix} 21.216 \\ 42.43 \\ 1 \end{bmatrix} = 450.116a_1 + 1800.3a_1 + 1a_1 = 2251.38a_1$$

...

$$a_{15} {S_{15}}^T S_{15} = a_{15} \begin{bmatrix} 21.216 \\ 42.43 \\ 1 \end{bmatrix}^T \begin{bmatrix} 21.216 \\ 42.43 \\ 1 \end{bmatrix} = 450.116a_{15} + 1800.3a_{15} + 1a_{15} = 2251.38a_{15}$$

Maka didapatlah hasil sebagai berikut untuk nilai $i = 1$.

$$a_1 {S_1}^T S_1 + a_2 {S_1}^T S_2 + \cdots + a_1 {S_{15}}^T S_{15} = y_1$$

$$2251.38a_1 + 2189.8a_2 + \cdots + 2252.4\, a_{15} = -1 \quad \ldots \quad \text{(Persamaan 1)}$$

Lakukan perhitungan di atas hingga $i = n$. Dengan menggunakan aplikasi Maple 2016 maka didapatkan nilai untuk variable a_1 sampai a_{15} sebagai berikut.

Tabel 3. 17 Nilai untuk variabel a1 sampai a15

a_i	Nilai
a_1	180.677
a_2	402.925
a_3	−55.154
a_4	568.038
a_5	−40.367
a_6	−101.419
a_7	−450.864
a_8	−30.225
a_9	−399.38
a_{10}	−38.572
a_{11}	−38.288

a_{12}	-112.032
a_{13}	-112.032
a_{14}	-96.994
a_{15}	101.786

Karena nilai ai adalah bernilai nol atau positif, maka nilai $a_3, a_5, a_6, a_7, a_8, a_9, a_{10}, a_{11}, a_{12}, a_{13}$ dan a_{14} tidak dihitung ketahap selanjutnya. Setelah didapatkan nilai ai masukkan ke persamaan 2.23 yaitu sebagai berikut.

$$\widetilde{W} = \sum_{i=1,j=1}^{n} a_i S_i \qquad (2.23)$$

$$\widetilde{W} = 180.667\begin{bmatrix}21.216\\42.43\\1\end{bmatrix} + 402.925\begin{bmatrix}21.216\\40.978\\1\end{bmatrix} + 568.038\begin{bmatrix}21.216\\42.206\\1\end{bmatrix} + 101.786\begin{bmatrix}21.216\\42.407\\1\end{bmatrix} = \begin{bmatrix}24555.984\\52467.812\\1253.416\end{bmatrix}$$

Maka didapatkan nilai w dan b adalah sebagai berikut.

$$w = \begin{matrix}24555.984\\52467.812\end{matrix} \quad \text{dan} \quad b = 1253.416$$

3.6.2. SVM *Testing*

Berikut adalah contoh data *testing* yang akan diklasifikasikan.

Tabel 3. 18 *Data Masukan* Testing

Data Testing
*Dan **Burhan$_1$** membenarkan berita itu, **dia$_1$** tahu dari **Ajis$_1$**, tapi **Burhan$_1$** juga bingung karena belum tahu bagaimana kejadian sebenarnya.*

Setelah melewati proses *preprocessing* hingga tahap ekstraksi fitur maka didapatkan data yang dapat dilihat pada tabel berikut.

Tabel 3. 19 Hasil Ekstraksi Fitur Data Training

i	*Tag - i*	*j*	*Tag - j*	Fitur *i*		Fitur *j*		Fitur Relasi		
				i- Pron	*i-* Prop	*j-* Pron	*j-* Prop	DOW	DOS	Str_ match
(dia) 1	PRP	(burhan) 1	NNP	1	0	0	1	0.21	0	0
(ajis) 1	NNP	(burhan) 1	NNP	0	1	0	1	0.37	0	0
(ajis) 1	NNP	(dia) 1	PRP	0	1	1	0	0.16	0	0
(burhan) 1	NNP	(burhan) 1	NNP	0	1	0	1	0.47	0	1
(burhan) 1	NNP	(dia) 1	PRP	0	1	1	0	0.26	0	0
(burhan) 1	NNP	(ajis) 1	NNP	0	1	0	1	0.11	0	0

Untuk melakukan *testing* kita memerlukan data training untuk melakukan tahap klasifikasi data *testing.* Untuk mempersingkat, hanya akan diambil salah satu contoh saja untuk dilakukan pengujian yaitu pasangan yang pertama dengan *i* adalah dia dan *j* adalah burhan.

Tabel 3. 20 Contoh Data Testing dalam Format SVM

Format SVM (x_i)	Label (y_i)
0 1 0 1 0.04 0 0	1
1 0 0 1 0.33 0.50 0	1
1 0 0 1 0.30 0.50 0	1
1 0 0 1 0.56 0.50 0	1
1 0 0 1 0.52 0.50 0	1
1 0 1 0 0.22 0 1	1
0 1 0 1 0.63 0.50 0	-1
0 1 0 1 0.59 0.50 0	-1
0 1 1 0 0.30 0 0	-1
0 1 1 0 0.07 0 0	-1
0 1 0 1 0.67 0.50 0	-1
0 1 0 1 0.63 0.50 0	-1
0 1 1 0 0.33 0 0	-1
0 1 1 0 0.11 0 0	-1
0 1 0 1 0.04 0 0	1
1 0 0 1 0.21 0 0	0

Selanjutnya melakukan kernelisasi menggunakan fungsi kernel linier sama seperti yang dilakukan pada tahap *training*. Dimana x_n merupakan hasil ekstraksi fitur yang telah diubah menjadi format SVM yang dapat dilihat pada tabel 3.19 di atas. Lakukan perhitungan dengan rumus $x_i^T x_i$ hingga $i = n$.

$$x_1{}^T x_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}^T \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$$

$$= [1\ 0\ 0\ 1\ 0.04\ 0.50\ 0] \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$$

$$= 1 + 0 + 0 + 1 + 0.002 + 0 + 0$$

$$= 2.002$$

Berikut adalah hasil dari perhitungan di atas.

Tabel 3. 21 Hasil Perhitungan $x_i{}^T x_j$ pada *Training*

i	x_i	$x_i{}^T$	j	x_j	$x_i{}^T x_j$
1	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$	[0 1 0 1 0.04 0 0]	1	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$	2.002
			2	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.33 \\ 0.50 \\ 0 \end{bmatrix}$	1.013
			3	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.30 \\ 0.50 \\ 0 \end{bmatrix}$	1.012
			4	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.56 \\ 0.50 \\ 0 \end{bmatrix}$	1.022

			5	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.52 \\ 0.50 \\ 0 \end{bmatrix}$	1.021
			6	$\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0.22 \\ 0 \\ 1 \end{bmatrix}$	0.009
			7	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.63 \\ 0.50 \\ 0 \end{bmatrix}$	2.025
			8	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.59 \\ 0.50 \\ 0 \end{bmatrix}$	2.023
			9	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.30 \\ 0 \\ 0 \end{bmatrix}$	1.012
			10	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.07 \\ 0 \\ 0 \end{bmatrix}$	1.003
			11	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.67 \\ 0.50 \\ 0 \end{bmatrix}$	2.027

			12	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.63 \\ 0.50 \\ 0 \end{bmatrix}$	2.025
			13	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.33 \\ 0 \\ 0 \end{bmatrix}$	1.013
			14	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.11 \\ 0 \\ 0 \end{bmatrix}$	1.004
			15	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$	2.002
			16	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.11 \\ 0 \\ 0 \end{bmatrix}$	1.008
. . .					
16		[0 1 0 1 0.11 0 0]	1	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$	1.008

	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.11 \\ 0 \\ 0 \end{bmatrix}$		2	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.33 \\ 0.50 \\ 0 \end{bmatrix}$	2.569
			3	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.30 \\ 0.50 \\ 0 \end{bmatrix}$	2.563
			4	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.56 \\ 0.50 \\ 0 \end{bmatrix}$	2.618
			5	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.52 \\ 0.50 \\ 0 \end{bmatrix}$	2.609
			6	$\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0.22 \\ 0 \\ 1 \end{bmatrix}$	1.046
			7	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.63 \\ 0.50 \\ 0 \end{bmatrix}$	1.632
			8	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.59 \\ 0.50 \\ 0 \end{bmatrix}$	1.633

			9	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.30 \\ 0 \\ 0 \end{bmatrix}$	0.063
			10	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.07 \\ 0 \\ 0 \end{bmatrix}$	0.015
			11	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.67 \\ 0.50 \\ 0 \end{bmatrix}$	1.641
			12	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.63 \\ 0.50 \\ 0 \end{bmatrix}$	1.632
			13	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.33 \\ 0 \\ 0 \end{bmatrix}$	0.069
			14	$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0.11 \\ 0 \\ 0 \end{bmatrix}$	0.023
			15	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.4 \\ 0 \\ 0 \end{bmatrix}$	1.008

			16	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.11 \\ 0 \\ 0 \end{bmatrix}$	2.044

Setelah dilakukan pada seluruh nilai x_i, maka matriks yang terbentuk dari hasil perhitungan adalah sebagai berikut.

$$x_i^T x_j = \begin{bmatrix} x_1^T x_1 & \cdots & x_1^T x_{16} \\ \vdots & \ddots & \vdots \\ x_{15}^T x_1 & \cdots & x_{15}^T x_{16} \end{bmatrix}$$

$$= \begin{bmatrix}
2.002 & 1.013 & 1.012 & 1.022 & 1.021 & 0.009 & 2.052 & 2.023 & 1.012 & 1.003 & 2.027 & 2.025 & 1.013 & 1.004 & 2.002 & 1.008 \\
1.013 & 2.359 & 2.349 & 2.435 & 2.422 & 1.073 & 1.457 & 1.444 & 0.099 & 0.023 & 1.471 & 1.458 & 0.109 & 0.036 & 1.013 & 2.569 \\
1.012 & 2.349 & 2.34 & 2.418 & 2.406 & 1.066 & 1.439 & 1.427 & 0.09 & 0.021 & 1.451 & 1.439 & 0.099 & 0.33 & 1.012 & 2.563 \\
1.022 & 2.435 & 2.418 & 2.564 & 2.541 & 1.123 & 1.602 & 1.58 & 0.168 & 0.039 & 1.625 & 1.603 & 0.185 & 0.062 & 1.022 & 2.618 \\
1.021 & 2.422 & 2.406 & 2.541 & 2.520 & 1.114 & 1.577 & 1.556 & 0.156 & 0.036 & 1.598 & 1.578 & 0.172 & 0.057 & 1.021 & 2.609 \\
0.009 & 1.073 & 1.066 & 1.123 & 1.114 & 3.048 & 0.139 & 0.13 & 1.066 & 1.015 & 0.147 & 0.139 & 1.073 & 1.024 & 0.009 & 1.046 \\
2.025 & 1.458 & 1.439 & 1.603 & 1.578 & 0.139 & 2.647 & 2.622 & 1.189 & 1.044 & 2.672 & 2.647 & 1.208 & 1.069 & 2.025 & 1.632 \\
2.024 & 1.445 & 1.427 & 1.580 & 1.557 & 0.13 & 2.622 & 2.598 & 1.177 & 1.041 & 2.645 & 2.622 & 1.195 & 1.065 & 2.024 & 1.624 \\
1.012 & 0.099 & 0.09 & 0.168 & 0.156 & 1.066 & 1.189 & 1.177 & 2.09 & 2.021 & 1.201 & 1.189 & 2.099 & 2.033 & 1.012 & 0.063 \\
1.003 & 0.023 & 0.021 & 0.039 & 0.036 & 1.015 & 1.044 & 1.041 & 2.021 & 2.005 & 1.047 & 1.044 & 2.023 & 2.008 & 1.003 & 0.015 \\
2.027 & 1.471 & 1.451 & 1.625 & 1.598 & 0.147 & 2.672 & 2.645 & 1.201 & 1.047 & 2.699 & 2.672 & 1.221 & 1.074 & 2.027 & 1.641 \\
2.025 & 1.458 & 1.439 & 1.603 & 1.578 & 0.139 & 2.647 & 2.622 & 1.189 & 1.044 & 2.672 & 2.647 & 1.208 & 1.069 & 2.025 & 1.632 \\
1.013 & 0.109 & 0.099 & 0.185 & 0.172 & 1.073 & 1.208 & 1.195 & 2.099 & 2.023 & 1.221 & 1.208 & 2.109 & 2.036 & 1.013 & 0.069 \\
1.004 & 0.036 & 0.033 & 0.062 & 0.057 & 1.024 & 1.069 & 1.065 & 2.033 & 2.008 & 1.074 & 1.069 & 2.036 & 2.012 & 1.004 & 0.023 \\
2.002 & 1.013 & 1.012 & 1.022 & 1.021 & 0.009 & 2.025 & 2.024 & 1.012 & 1.003 & 2.027 & 2.025 & 1.013 & 1.004 & 2.002 & 1.008 \\
2.069 & 2.063 & 2.021 & 2.109 & 1.046 & 1.132 & 1.124 & 0.063 & 0.015 & 1.141 & 0.069 & 1.132 & 0.069 & 0.023 & 1.008 & 2.044
\end{bmatrix}$$

Tahap selanjutnya adalah melakukan perhitungan terhadap y_i dengan melakukan hal yang sama seperti perhitungan terhadap nilai x_i. Untuk nilai y_i adalah nilai dari label yang telah diberikan, dapat dilihat pada tabel 3.11. Lakukan perhitungan sebagai berikut dengan rumus $y_i^T y_i$ hingga $i = n$.

$$y_1^T y_1 = [1]^T [1] = [1] [1] = 1$$

Berikut adalah hasil dari perhitungan *y* di atas.

Tabel 3. 22 Hasil perhitungan $y_i^T y_j$ pada *Testing*

i	y_i	y_i^T	j	y_j	$y_i^T y_j$
1	[1]	[1]	1	[1]	1
			2	[1]	1
			3	[1]	1
			4	[1]	1
			5	[1]	1
			6	[1]	1

			7	[−1]	-1
			8	[−1]	-1
			9	[−1]	-1
			10	[−1]	-1
			11	[−1]	-1
			12	[−1]	-1
			13	[−1]	-1
			14	[−1]	-1
			15	[1]	1
			16	[0]	0
			.		
			.		
			.		
16	[0]	[1]	1	[1]	0
			2	[1]	0
			3	[1]	0
			4	[1]	0
			5	[1]	0
			6	[1]	0
			7	[−1]	0
			8	[−1]	0
			9	[−1]	0
			10	[−1]	0
			11	[−1]	0
			12	[−1]	0
			13	[−1]	0
			14	[−1]	0
			15	[1]	0
			16	[0]	0

Sehingga matriks yang terbentuk dari hasil perhitungan $y_i^T y_j$ hasilnya adalah sebagai berikut.

$$y_i^T y_j = \begin{bmatrix} y_1^T y_1 & \cdots & y_1^T y_{16} \\ \vdots & \ddots & \vdots \\ y_{16}^T y_1 & \cdots & y_{15}^T y_{16} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 0 \\ -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & 0 \\ -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & 0 \\ -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & 0 \\ -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & 0 \\ -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & 0 \\ -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & 0 \\ -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Selanjutnya adalah tahap mengubah setiap pasangan *i* dan *j* menjadi nilai vektor (*support vector*) untuk mendapatkan nilai a_i. Nilai x_i didapatkan menggunakan persamaan 2.19 kernel linear sebagai berikut.

$$\sum_{i=1,j=1}^{n} x_i^T x_j \ (i, = 1,..,\text{n}) \qquad (2.19)$$

$$x_1 = x_1^T x_1 + x_1^T x_2 + \cdots + x_1^T x_{16} = 2.002 + 1.013 + \cdots + 2.004 = 22.218$$

$$x_2 = x_2^T x_1 + x_2^T x_2 + \cdots + x_1^T x_{15} = 1.013 + 2.359 + \cdots + 1.536 = 20.299$$

$$\vdots$$

$$x_{16} = x_1^T x_{16} + x_2^T x_{16} + \cdots + x_{16}^T x_{16} = 2.002 + 1.013 + \cdots + 2.012 = 22.559$$

Sehingga didapatkan nilai x_i untuk setiap pasangan sebagai berikut.

Tabel 3. 23 Nilai *x* Untuk Setiap Pasangan

x_i	Nilai
x_1	22.218
x_2	20.2985
x_3	20.135
x_4	21.552
x_5	21.334
x_6	12.199
x_7	27.9335
x_8	27.7155
x_9	17.635
x_{10}	16.3815

x_{11}	28.1515
x_{12}	27.9335
x_{13}	17.7985
x_{14}	16.5995
x_{15}	22.218
x_{16}	22.559

Begitu pun dengan nilai y_i didapatkan menggunakan persamaan 2.20 kernel *linear* untuk y_i yaitu sebagai berikut.

$$\sum_{i=1,j=1}^{n} y_i^T y_j \ (i, = 1, \dots, \mathrm{n}) \qquad (2.20)$$

$$y_1 = y_1^T y_1 + y_1^T x_2 + \cdots + y_1^T x_{16} = 1 + 1 + \cdots + 1 = -1$$

$$y_2 = y_2^T y_1 + y_2^T x_2 + \cdots + y_1^T x_{16} = 1 + 1 + \cdots + 1 = -1$$

$$\vdots$$

$$y_{16} = y_{16}^T y_1 + y_{16}^T x_2 + \cdots + y_{16}^T x_{16} = 0 + 0 + \cdots + 0 = 0$$

Sehingga didapatkan nilai x untuk setiap pasangan sebagai berikut.

Tabel 3. 24 Nilai *y* Untuk Setiap Pasangan

y_i	**Nilai**
y_1	-1
y_2	-1
y_3	-1
y_4	-1
y_5	-1
y_6	-1
y_7	1
y_8	1
y_9	1
y_{10}	1
y_{11}	1
y_{12}	1
y_{13}	1
y_{14}	1

y_{15}	-1
y_{16}	0

Setelah didapatkan nilai x_i dan y_i, lalu subtitusi nilai x_i dan y_i ke dalam persamaan trick phi 2.18 berikut.

$$\phi\begin{bmatrix}x_i\\y_i\end{bmatrix} = \begin{cases} \sqrt{{x_n}^2 + {y_n}^2} > 2, maka \begin{bmatrix}\sqrt{{x_n}^2 + {y_n}^2} - x_i + |x_i - y_i|\\ \sqrt{{x_n}^2 + {y_n}^2} - y_i + |x_i - y_i|\end{bmatrix} \\ \sqrt{{x_n}^2 + {y_n}^2} \leq 2, maka \begin{bmatrix}x_i\\y_i\end{bmatrix} \end{cases} \quad (2.18)$$

Nilai x_n yang didapat dari x_{16} dan y_n dari y yang disubtitusikan ke dalam persamaan sebagai berikut.

$$\sqrt{{x_{16}}^2 + {y_{16}}^2} = \sqrt{22.559^2 + 0^2} = 22.559$$

Karena $\sqrt{{x_n}^2 + {y_n}^2} > 2$, maka $\phi\begin{bmatrix}x_i\\y_i\end{bmatrix} = \begin{bmatrix}\sqrt{{x_n}^2 + {y_n}^2} - x_i + |x_i - y_i|\\ \sqrt{{x_n}^2 + {y_n}^2} - y_i + |x_i - y_i|\end{bmatrix}$

$$\phi\begin{bmatrix}x_1\\y_1\end{bmatrix} = \begin{bmatrix}\sqrt{{x_{16}}^2 + {y_{15}}^2} - x_1 + |x_1 - y_1|\\ \sqrt{{x_{16}}^2 + {y_{15}}^2} - y_1 + |x_1 - y_1|\end{bmatrix}$$

$$= \begin{bmatrix}\sqrt{22.559^2 + 0^2} - 22.218 + |22.218(-1)|\\ \sqrt{22.559^2 + 0^2} - (1) + |20.218 - (-1)|\end{bmatrix}$$

$$= \begin{bmatrix}3.012\\26.23\end{bmatrix}$$

Selanjutnya untuk mendapatkan nilai kelas dari data testing, lakukan perhitungan ke persamaan $\widetilde{W}^T.\ \phi t\begin{bmatrix}x_t\\y_t\end{bmatrix}$ dengan $\widetilde{W}^T$ didapatkan dari hasil training.

$$\widetilde{W}^T.\ \phi t\begin{bmatrix}x_t\\y_t\end{bmatrix} = \begin{bmatrix}24555.984\\52467.812\end{bmatrix}^T \text{ x } \begin{bmatrix}3.012\\26.23\end{bmatrix} = 1450172.03$$

Berdasarkan hasil *training* yang dilakukan sebelumnya didapat nilai b=1253.416 sehingga proses klasifikasi memiliki aturan sebagai berikut.

Tabel 3. 25 Aturan Klasifikasi SVM

$\widetilde{W}^T . \ \phi t \begin{bmatrix} x_t \\ y_t \end{bmatrix}$	**Kelas**
$\widetilde{W}^T . \ \phi t \begin{bmatrix} x_t \\ y_t \end{bmatrix} \leq 1253.416$	-1 (Bukan *Coreference)*
$\widetilde{W}^T . \ \phi t \begin{bmatrix} x_t \\ y_t \end{bmatrix} > 1253.416$	1 (*Coreference)*

Hasil *testing* adalah 1450172.03 sehingga data *testing* lebih besar dari pada hasil nilai b pada tahap *training* yaitu 1253.416 yang berarti data *testing* untuk pasangan ***dia*** dan ***burhan*** atermasuk dalam kelas 1 (*coreference*). Berikut adalah contoh hasil *testing* SVM.

Tabel 3. 26 Hasil Testing SVM

i	***Tag - i***	***j***	***Tag - j***	**Format SVM**	**Hasil Klasifikasi**
(dia) 1	PRP	(burhan) 1	NNP	1 0 0 1 0.21 0 0	Coreference
(ajis) 1	NNP	(burhan) 1	NNP	0 1 0 1 0.37 0 0	Bukan Coreference
(ajis) 1	NNP	(dia) 1	PRP	0 1 1 0 0.16 0 0	Bukan Coreference
(burhan) 1	NNP	(burhan) 1	NNP	0 1 0 1 0.47 0 1	Bukan Coreference
(burhan) 1	NNP	(dia) 1	PRP	0 1 1 0 0.26 0 0	Bukan Coreference
(burhan) 1	NNP	(ajis) 1	NNP	0 1 0 1 0.11 0 0	Bukan Coreference

Pada penelitian ini, tahapan *training* dan *testing* tersebut dilakukan dengan menggunakan *library* LIBSVM untuk PHP yang dibuat oleh Gabriel R Couto, adapun dalam penggunaannya yaitu data *training* yang sudah diberi bobot dan label. Sedangkan data *testing* hanya membutuhkan data *input* yang merupakan array yang dapat dilihat pada gambar 3.10.

```
 $data=[[0,0,0,0,0],[0.5,0.5,0.5,0.5,0.5],
[0.7,0.7,0.7,0.7,0.7],[1,1,1,1,1]];
$labels = [-1,-1,1,1];
function svm_train($data, $labels) {
  $svm = new Svm();
  $svm->train($data, $labels);
  $svm->save('svm_model.svm');
}
 $input = [[0,0,0,0,0],[1,1,1,1,1]];
function svm_test($input) {
  $svm = Svm::load('svm_model.svm');
  $predictions = $svm->predict($input);

  return $predictions;
}
```

Gambar 3. 10 Training *dan* Testing *dengan LIBSVM*

3.7. Analisis Rencana Pengujian

Pengujian sistem dilakukan untuk memastikan bahwa sistem yang dibuat sesuai dengan rancangan sistem yang telah dibuat sebelumnya. Pada tahap ini, hasil analisis pendeteksian *Coreference Resolution* menggunakan sistem yang telah dibuat dan pendeteksian secara manual dapat dilihat pada tabel berikut.

Tabel 3. 27 Hasil Analisis Pendeteksian *Coreference Resolution*

i	j	Sistem	Manual	Hasil
(dia) 1	(burhan) 1	Coreference	Coreference	Benar
(ajis) 1	(burhan) 1	Bukan Coreference	Bukan Coreference	Benar
(ajis) 1	(dia) 1	Bukan Coreference	Bukan Coreference	Benar
(burhan) 1	(burhan) 1	Bukan Coreference	Coreference	Salah
(burhan) 1	(dia) 1	Bukan Coreference	Coreference	Salah
(burhan) 1	(ajis) 1	Bukan Coreference	Bukan Coreference	Benar

Untuk mengukur pengujian pada penelitian ini menggunakan perhitungan nilai akurasi yang dihasilkan sistem *coreference resolution* dengan menggunakan metode *Support Vector Machine* dan *coreference resolution* secara manual. Dengan menggunakan rumus 2.29 sebagai berikut.

$$Akurasi = \frac{Jumlah\ yang\ diklasifikasi\ dengan\ benar}{Jumlah\ sampel\ data\ uji}\ x\ 100\% \qquad (2.29)$$

Sehinga akurasi yang dihasilkan adalah sebagai berikut.

$$Akurasi = \frac{Jumlah\ yang\ diklasifikasi\ dengan\ benar}{Jumlah\ sampel\ data\ uji}\ x\ 100\% = \frac{4}{6} = 66\%$$

3.8. Analisis Kebutuhan Non Fungsional

Analisis non fungsional yang dilakukan dibagi ke dalam 2 tahap, yaitu analisis kebutuhan perangkat keras dan analisis kebutuhan perangkat lunak.

3.8.1 Analisis Kebutuhan Perangkat Keras

Analisis kebutuhan perangkat keras (*hardware*) pada penelitian ini merupakan kebutuhan perangkat keras yang digunakan dalam pembangunan aplikasi ini. Adapun spesifikasi minimum perangkat keras berdasarkan spesifikasi minimum untuk aplikasi yang digunakan dalam sistem *coreference resolution* seperti XAMPP dan *Macromedia Dreamweaver* 8 dapat dilihat pada tabel 3.28 berikut.

Tabel 3. 28 Spesifikasi Kebutuhan Perangkat Keras

No	Perangkat Keras	Spesifikasi
1	*Processor*	Intel Pentium III
2	Memori (RAM)	256 MB
3	*Harddisk*	300 MB

Sedangkan spesifikasi minimum perangkat keras yang digunakan untuk pengguna berdasarkan aplikasi yang digunakan untuk pengguna seperti *browser Chrome* adalah sebagai berikut.

Tabel 3. 29 Spesifikasi Minimum Perangkat Keras untuk Pengguna

No	Perangkat Keras	Spesifikasi
1	*Processor*	Intel Pentium III
2	Memori (RAM)	128 MB
3	*Harddisk*	200 MB

3.8.2 Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak (*software*) pada penelitian ini merupakan *tools* yang digunakan untuk pembangunan aplikasi ini. Adapun *tools* yang digunakan berdasarkan aplikasi yang digunakan baik dalam pembangunan sistem maupun penggunaan dapat dilihat pada tabel 3.29 berikut.

Tabel 3. 30 Spesifikasi Kebutuhan Perangkat Lunak

No	Perangkat Lunak	Spesifikasi
1	Sistem Operasi	Windows XP atau versi yang lebih baru
2	Web Browser	Google Chrome Versi 54.0.2840.99

3.9. Analisis Kebutuhan Non Fungsional

Analsis kebutuhan fungsional merupakan analisis reaksi sistem terhadap input tertentu, serta perilaku sistem ketika dalam situasi tertentu. Diagram yang akan digambarkan pada bagian ini adalah Diagram Konteks, DFD level 1, DFD level 2, dan Deskripsi Proses.

3.9.1. Analisis Penyimpanan Data

Dalam penelitian ini data yang dibutuhkan sebagai input data *training* adalah teks novel dan label dengan format file .txt. File novel tersebut akan diupload oleh pengguna dan disimpan di dalam folder, sedangkan label sudah disimpan oleh sistem di dalam folder yang sama. Pada tahap *training* akan menghasilkan model SVM yang akan disimpan dengan format file .svm contohnya adalah model_SVM.svm. Untuk data yang dibutuhkan sebagai input proses *testing* adalah teks novel dengan format file .txt. File ini juga harus diupload terlebih dahulu oleh

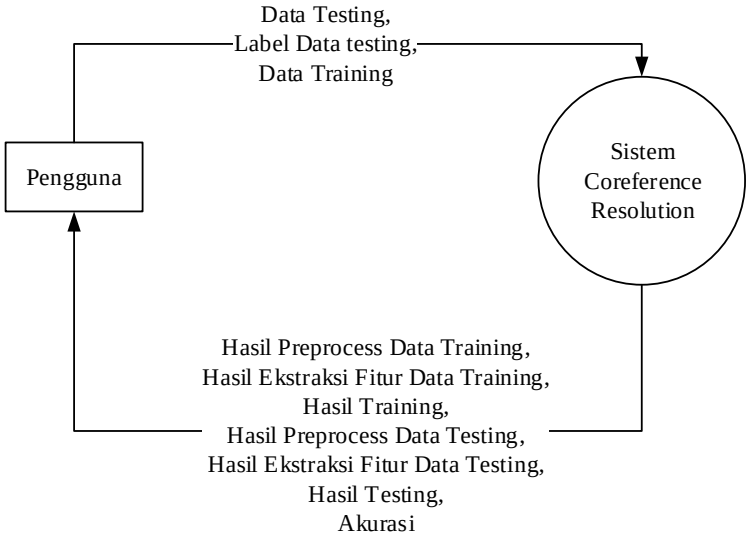
pengguna yang kemudian sistem akan menyimpannya di dalam suatu folder. Setiap proses pada tahap *testing* seperti hasil *preprocessing* hingga hasil klasifikasi akan disimpan di dalam sebuah file dengan format file .result. Proses dan file yang dilibatkan dapat dilihat pada tabel berikut.

Tabel 3. 31 Penyimpanan Data

Nama Proses	File Input (format file)	File Output (format file)	Lokasi Penyimpanan
Training	Teks Novel (.txt)	SVM Model (.svm)	Folder dalam sistem
	Label (.txt)	Hasil setiap proses (.result)	
Testing	SVM Model (.svm)	Hasil setiap proses (.result)	Folder dalam sistem
	Teks Novel (.txt)		

3.9.2. Diagram Konteks

Diagram konteks menggambarkan sistem secara keseluruhan sistem *Coreference Resolution* dengan *user* sebagai entitas eksternal atau pengguna sistem. Diagram konteks sistem ini dapat dilihat pada gambar berikut.



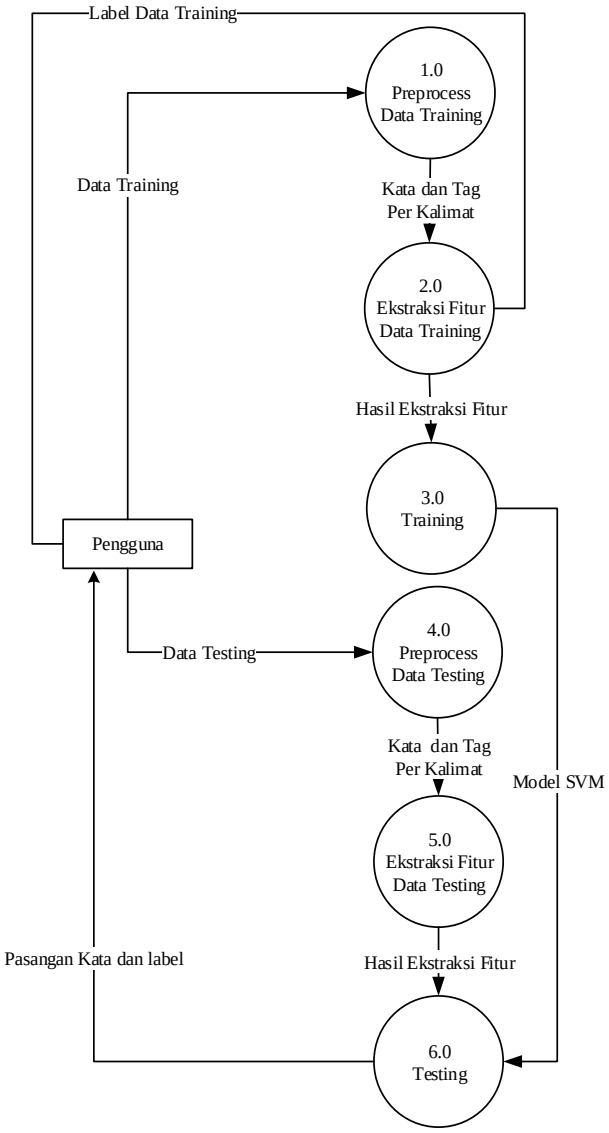
Gambar 3. 11 Diagram Konteks

3.9.3. Data Flow Diagram

Data Flow Diagram (DFD) merupakan model sistem yang menunjukkan fungsional dimana tiap transformasi menunjukkan fungsi atau proses. Berikut ini merupakan DFD untuk sistem *Coreference Resolution* untuk novel bahasa Indonesia menggunakan metode SVM.

3.9.3.1. DFD Level 1

Pada DFD level 1 ini kita dapat melihat apa saja yang dapat dilakukan pengguna terhadap sistem, seperti yang terlihat pada gambar berikut

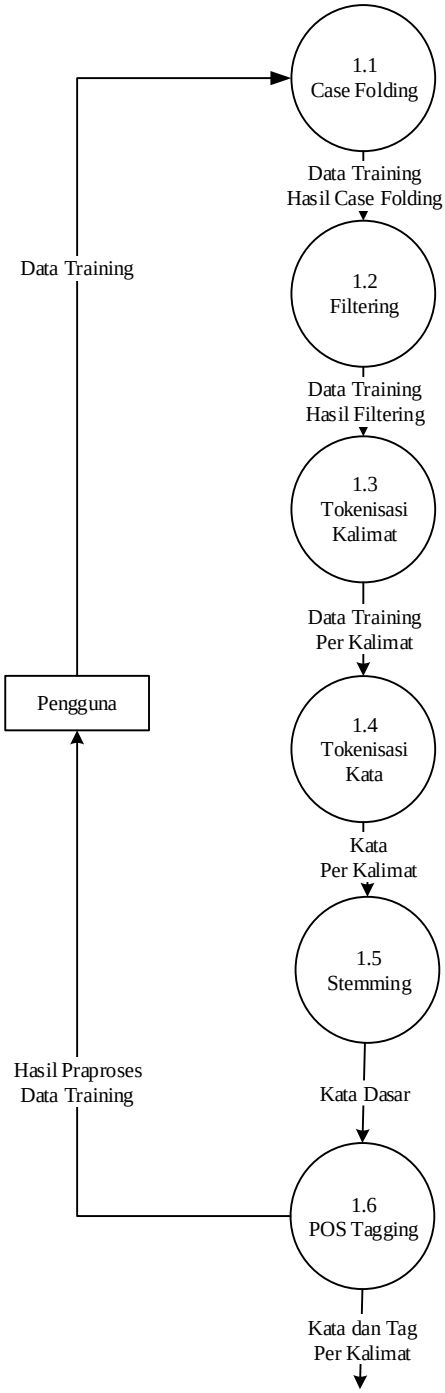


Gambar 3. 12 DFD Level 1

3.9.3.2. DFD Level 2

1. **Preprocess Data Training**

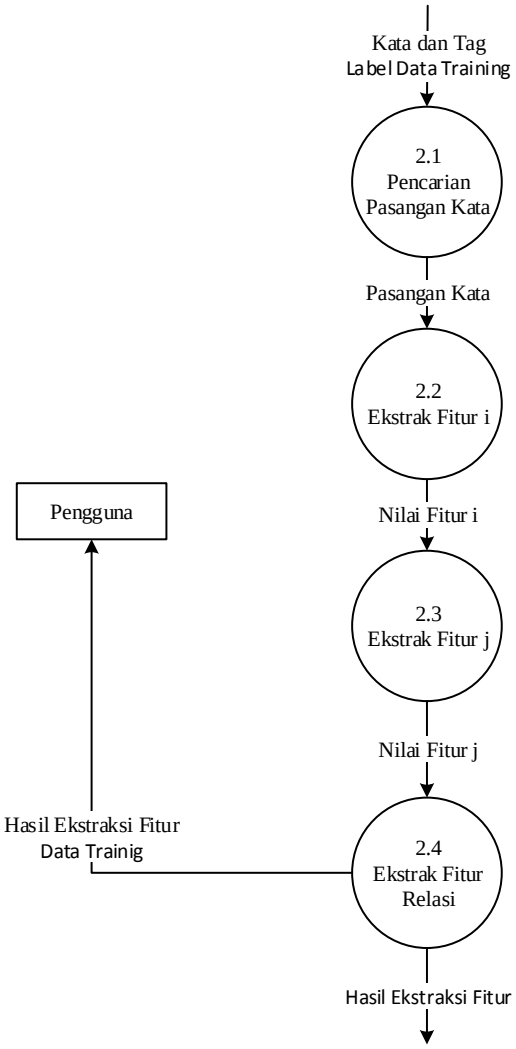
DFD Level 2 untuk praproses data *training* dapat dilihat pada gambar berikut.



Gambar 3. 13 DFD Level 2 Preprocessing Data *Training*

1. Ekstraksi Fitur

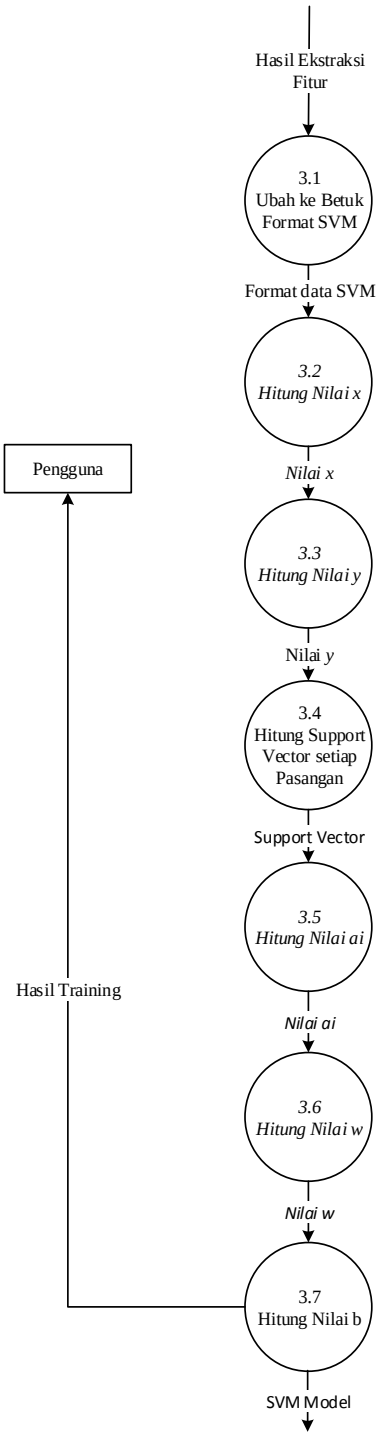
DFD Level 2 untuk proses ekstraksi fitur dapat dilihat pada gambar berikut.



Gambar 3. 14 DFD Level 2 Ekstraksi Fitur Training

2. SVM *Training*

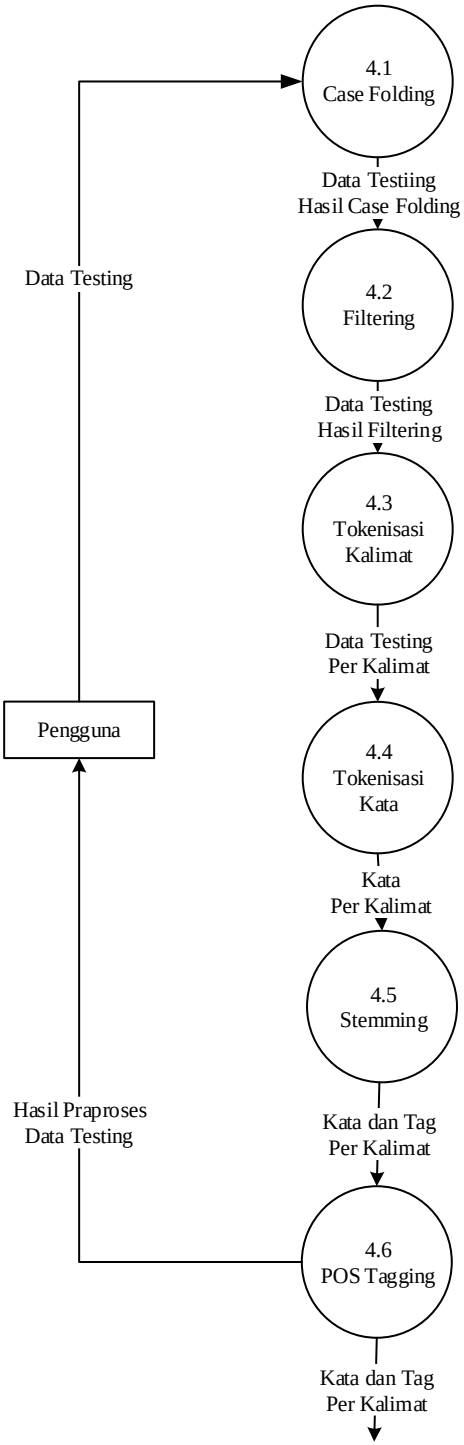
DFD Level 2 untuk proses *training* dapat dilihat pada gambar berikut.



Gambar 3. 15 DFD Level 2 SVM *Training*

3. Preprocess Data Testing

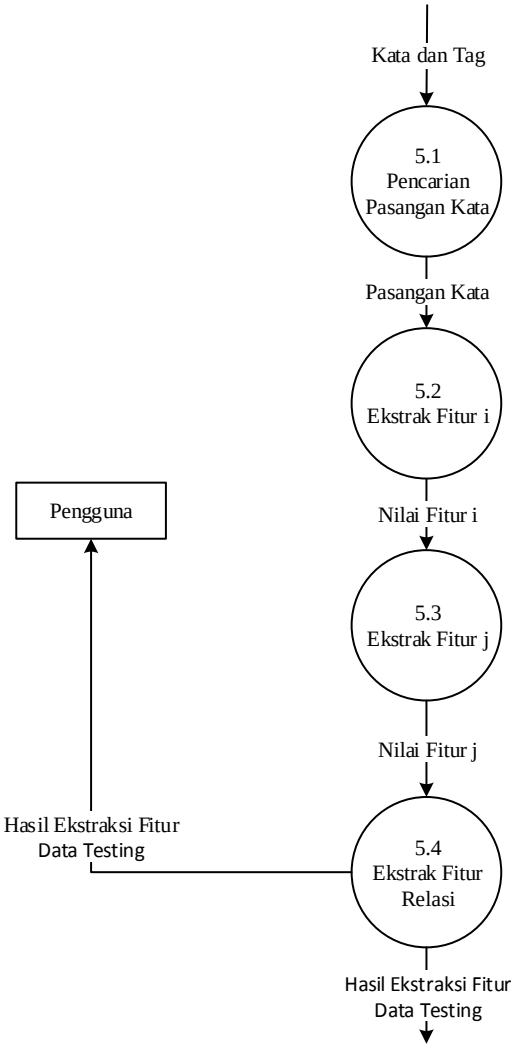
DFD Level 2 untuk praproses data *testing* dapat dilihat pada gambar berikut.



Gambar 3. 16 DFD Level 2 Preprocessing Data *Testing*

4. Ekstraksi Fitur Data *Testing*

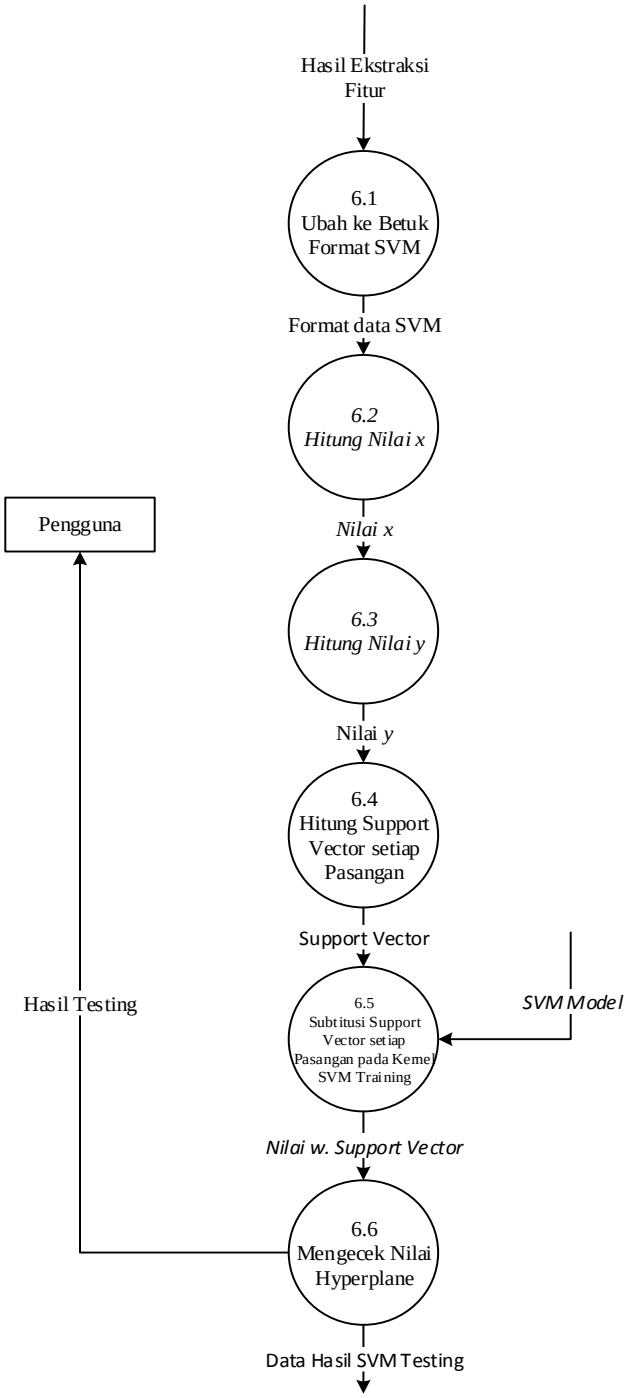
DFD Level 2 untuk proses ekstraksi fitur data *testing* dapat dilihat pada gambar berikut.



Gambar 3. 17 DFD Level 2 Ekstraksi Fitur Data Testing

5. SVM *Testing*

DFD Level 2 untuk proses *testing* dapat dilihat pada gambar berikut.



Gambar 3. 18 DFD Level 2 SVM *Testing*

3.9.4. Spesifikasi Proses

Spesifikasi proses merupakan keterangan atau deskripsi dari semua proses yang terdapat di DFD. Berikut ini merupakan spesifikasi proses untuk DFD pada gambar 3.12 di atas.

Tabel 3. 32 Spesifikasi Proses

No	Nama Proses	Keterangan
1	Nomor Proses	1.0
	Nama Proses	Preprocess Data *Training*
	Sumber	Pengguna
	Input	Data *Training*
	Output	Kata dan tag per kalimat
	Logika Proses	1. Lakukan *case folding* pada data *training.* 2. *Filtering* 3. Tokenisasi kalimat 4. Tokenisasi kata *5. Stemming* 6. *POS Tagging*
2	Nomor Proses	2.0
	Nama Proses	Ekstraksi Fitur Data *Training*
	Sumber	Pengguna
	Input	Kata dan tag per kalimat
	Output	Hasil ekstraksi fitur data *training*
	Logika Proses	1. Pencarian pasangan kata berdasarkan label data *training*. 2. Fitur *i-pronoun* bernilai 1 apabila kata *i* memiliki tag PRP dan bernilai 0 jika tidak 3. Fitur *i-propernoun* bernilai 1 apabila kata *i* memiliki tag NNP dan bernilai 0 jika tidak 4. Fitur *j-pronoun* bernilai 1 apabila kata *j* memiliki tag PRP dan bernilai 0 jika tidak 5. Fitur *j-propernoun* bernilai 1 apabila kata *j* memiliki tag NNP dan bernilai 0 jika tidak 6. Fitur *distance* bernilai 0 jika kata *i* dan *j* tidak terpisah kalimat, bernilai 0,n jika kata *i* dan *j* terpisah kalimat sebanyak n 7. Fitur *str_match* bernilai 1 apabila kata *i* dan kata *j* memiliki susunan karakter yang sama
3	Nomor Proses	3.0
	Nama Proses	*Training*
	Sumber	Pengguna
	Input	Hasil ekstraksi fitur
	Output	Model SVM

	Logika Proses	1. Ubah ke bentuk SVM 2. Hitung nilai *x* 3. Hitung nilai *y* 4. Hitung Support Vector tiap pasangan 5. Hitung nilai *ai* 6. Hitung nilai *w* 7. Hitung nilai b
4	Nomor Proses	4.0
	Nama Proses	Praproses Data *Testing*
	Sumber	Pengguna
	Input	Data *testing*
	Output	Kata dan tag per kalimat
	Logika Proses	1. Lakukan *case folding* pada data *testing* 2. *Filtering* 3. Tokenisasi kalimat 4. Tokenisasi kata *5. Stemming* 6. *POS Tagging*
5	Nomor Proses	5.0
	Nama Proses	Ekstraksi Fitur Data *Testing*
	Sumber	Pengguna
	Input	Pasangan kata dan tag
	Output	Hasil ekstraksi fitur
	Logika Proses	1. Pencarian pasangan kata berdasarkan label data *training*. 2. Fitur *i-pronoun* bernilai 1 apabila kata *i* memiliki tag PRP dan bernilai 0 jika tidak 3. Fitur *i-propernoun* bernilai 1 apabila kata *i* memiliki tag NNP dan bernilai 0 jika tidak 4. Fitur *j-pronoun* bernilai 1 apabila kata *j* memiliki tag PRP dan bernilai 0 jika tidak 5. Fitur *j-propernoun* bernilai 1 apabila kata *j* memiliki tag NNP dan bernilai 0 jika tidak 6. Fitur *distance* bernilai 0 jika kata *i* dan *j* tidak terpisah kalimat, bernilai 0,n jika kata *i* dan *j* terpisah kalimat sebanyak n 7. Fitur *str_match* bernilai 1 apabila kata *i* dan kata *j* memiliki susunan karakter yang sama
6	Nomor Proses	6.0
	Nama Proses	*Testing*
	Sumber	Pengguna
	Input	Data hasil preprocessing testing SVM Model
	Output	Hasil Klasifikasi dan Akurasi
	Logika Proses	1. Ubah ke bentuk SVM 2. Hitung nilai *x*

		3. Hitung nilai *y* 4. Hitung Support Vector tiap pasangan 5. Subtitusi Support Vector setiap Pasangan pada Kernel SVM Training 6. Cek Nilai hyperplane 7. Hitung akurasi
7	Nomor Proses	1.1
	Nama Proses	*Case Folding*
	Sumber	Pengguna
	Input	Data *training*
	Output	Data *training* hasil *case folding*
	Logika Proses	1. Ubah semua karakter huruf pada data *training* menjadi huruf kecil semua.
8	Nomor Proses	1.2
	Nama Proses	Filtering
	Sumber	Pengguna
	Input	Data *training* hasil *case folding*
	Output	Data *training* tanpa karakter selain ‘a’-‘z’, ‘-‘, ‘.’ dan spasi
	Logika Proses	1. Pengecekan setiap karakter 2. Penghapusan karakter selain ‘a’-‘z’, ‘-‘, ‘.’ dan spasi
9	Nomor Proses	1.3
	Nama Proses	Tokenisasi Kalimat
	Sumber	Pengguna
	Input	Data *training* hasil *filtering*
	Output	Data *training* per kalimat
	Logika Proses	1. Pencarian delimiter titik ‘.’ 2. Pemisahan kalimat
10	Nomor Proses	1.4
	Nama Proses	Tokenisasi Kata
	Sumber	Pengguna
	Input	Data *training* per kalimat
	Output	Kata per kalimat
	Logika Proses	1. Pencarian delimiter spasi ‘ ‘ 2. Pemisahan kata
11	Nomor Proses	1.5
	Nama Proses	Stemming
	Sumber	Pengguna
	Input	Kata per kalimat
	Output	Kata dan tag per kalimat
	Logika Proses	1. Pengecekan kata berimbuhan 2. Pengubahan menjadi kata dasar.
12	Nomor Proses	1.6
	Nama Proses	POS *Tagging*

	Sumber	Pengguna
	Input	Kata dasar
	Output	Kata dan tag
	Logika Proses	1. Pengecekan jenis kata berdasarkan hasil training POS Tagger
13	Nomor Proses	2.1
	Nama Proses	Pencarian Pasangan Kata
	Sumber	Pengguna
	Input	Kata dan tag, Label Data Training
	Output	Pasangan Kata
	Logika Proses	1. Pengecekan Label Data Training 2. Jika kata *i* dan kata *j* susunan labelnya berbeda maka dijadikan pasangan
14	Nomor Proses	2.2
	Nama Proses	Ekstrak fitur *i*
	Sumber	Pengguna
	Input	Kata *i*
	Output	Nilai fitur *i*
	Logika Proses	1. Fitur *i-pronoun* bernilai 1 apabila kata *i* memiliki tag PRP dan bernilai 0 jika tidak 2. Fitur *i-propernoun* bernilai 1 apabila kata *i* memiliki tag NNP dan bernilai 0 jika tidak
15	Nomor Proses	2.3
	Nama Proses	Ekstrak fitur *j*
	Sumber	Pengguna
	Input	Kata *j*
	Output	Nilai fitur *j*
		1. Fitur *j-pronoun* bernilai 1 apabila kata *j* memiliki tag PRP dan bernilai 0 jika tidak 2. Fitur *j-propernoun* bernilai 1 apabila kata *j* memiliki tag NNP dan bernilai 0 jika tidak
16	Nomor Proses	2.4
	Nama Proses	Ekstrak Fitur Relasi
	Sumber	Pengguna
	Input	Pasangan Kata
	Output	Nilai fitur relasi
	Logika Proses	1. Fitur *distance* bernilai 0 jika kata *i* dan *j* tidak terpisah kalimat, bernilai 0,n jika kata *i* dan *j* terpisah kalimat sebanyak n 2. Fitur *str_match* bernilai 1 apabila kata *i* dan kata *j* memiliki susunan karakter yang sama
17	Nomor Proses	3.1
	Nama Proses	Ubah ke Format SVM
	Sumber	Pengguna
	Input	Hasil Ekstraksi Fitur

	Output	Format SVM
	Logika Proses	1. Pengecekan hasil ekstraksi fitur setiap pasangan kata 2. Pengubahan ke Format SVM
18	Nomor Proses	3.2
	Nama Proses	Hitung Nilai *x*
	Sumber	Pengguna
	Input	Data Format SVM
	Output	Nilai *x*
		1. Pencarian nilai x_i berdasarkan rumus kernel linier
19	Nomor Proses	3.3
	Nama Proses	Hitung Nilai *y*
	Sumber	Pengguna
	Input	Data Label
	Output	Kata per kalimat
	Logika Proses	1. Pencarian nilai *yi* berdasarkan rumus kernel linear
20	Nomor Proses	3.4
	Nama Proses	Hitung Support Vector setiap Pasangan
	Sumber	Pengguna
	Input	Nilai x dan nilai y
	Output	Support Vector setiap Pasangan
	Logika Proses	1. Menggunakan rumus kernel trick phi dengan mengambil nilai x dan y
21	Nomor Proses	3.5
	Nama Proses	Hitung Nilai *ai*
	Sumber	Pengguna
	Input	Support Vector setiap Pasangan
	Output	Persamaan linier
	Logika Proses	1. Buat persamaan linier dari setiap Support Vector 2. Cari nilai *ai*
22	Nomor Proses	3.6
	Nama Proses	Hitung Nilai w
	Sumber	Pengguna
	Input	Nilai *ai*
	Output	Nilai *w*
	Logika Proses	1. Menggunakan rumus W dengan memanfaatkan nilai *ai* yang bernilai nol atau positif
23	Nomor Proses	3.7
	Nama Proses	Hitung nilai *b*
	Sumber	Pengguna
	Input	Nilai W

	Output	Nilai b
	Logika Proses	1. Mengambil Nila b dengan persamaan W
24	Nomor Proses	4.1
	Nama Proses	*Case Folding*
	Sumber	Pengguna
	Input	Data *training*
	Output	Data *training* hasil *case folding*
	Logika Proses	1. Ubah semua karakter huruf pada data *training* menjadi huruf kecil semua.
25	Nomor Proses	4.2
	Nama Proses	Filtering
	Sumber	Pengguna
	Input	Data *training* hasil *case folding*
	Output	Data *training* tanpa karakter selain ‘a’-‘z’, ‘-‘, ‘.’ dan spasi
	Logika Proses	3. Pengecekan setiap karakter 4. Penghapusan karakter selain ‘a’-‘z’, ‘-‘, ‘.’ dan spasi
26	Nomor Proses	4.3
	Nama Proses	Tokenisasi Kalimat
	Sumber	Pengguna
	Input	Data *training* hasil *filtering*
	Output	Data *training* per kalimat
	Logika Proses	3. Pencarian delimiter titik ‘.’ 4. Pemisahan kalimat
27	Nomor Proses	4.4
	Nama Proses	Tokenisasi Kata
	Sumber	Pengguna
	Input	Data *training* per kalimat
	Output	Kata per kalimat
	Logika Proses	3. Pencarian delimiter spasi ‘ ‘ 4. Pemisahan kata
28	Nomor Proses	4.5
	Nama Proses	Stemming
	Sumber	Pengguna
	Input	Kata per kalimat
	Output	Kata dan tag per kalimat
	Logika Proses	3. Pengecekan kata berimbuhan 4. Pengubahan menjadi kata dasar.
29	Nomor Proses	4.6
	Nama Proses	POS *Tagging*
	Sumber	Pengguna
	Input	Kata dasar
	Output	Kata dan tag

	Logika Proses	2. Pengecekan jenis kata berdasarkan hasil training POS Tagger
30	Nomor Proses	5.1
	Nama Proses	Pencarian Pasangan Kata
	Sumber	Pengguna
	Input	Kata dan tag, Label Data Training
	Output	Pasangan Kata
	Logika Proses	1. Pengecekan tag setiap kata 2. Jika Tag NNP atau PRP maka akan menjadi kandidat 3. Jika kata *i* dan kata *j* kunci kata dan kalimat berbeda maka dijadikan pasangan
31	Nomor Proses	5.2
	Nama Proses	Ekstrak fitur *i*
	Sumber	Pengguna
	Input	Kata *i*
	Output	Nilai fitur *i*
	Logika Proses	1. Fitur *i-pronoun* bernilai 1 apabila kata *i* memiliki tag PRP dan bernilai 0 jika tidak 2. Fitur *i-propernoun* bernilai 1 apabila kata *i* memiliki tag NNP dan bernilai 0 jika tidak
32	Nomor Proses	5.3
	Nama Proses	Ekstrak fitur *j*
	Sumber	Pengguna
	Input	Kata *j*
	Output	Nilai fitur *j*
		1. Fitur *j-pronoun* bernilai 1 apabila kata *j* memiliki tag PRP dan bernilai 0 jika tidak 2. Fitur *j-propernoun* bernilai 1 apabila kata *j* memiliki tag NNP dan bernilai 0 jika tidak
33	Nomor Proses	5.4
	Nama Proses	Ekstrak Fitur Relasi
	Sumber	Pengguna
	Input	Pasangan Kata
	Output	Hasil Ekstraksi Fitur
	Logika Proses	1. Fitur *distance* bernilai 0 jika kata *i* dan *j* tidak terpisah kalimat, bernilai 0,n jika kata *i* dan *j* terpisah kalimat sebanyak n 2. Fitur *str_match* bernilai 1 apabila kata *i* dan kata *j* memiliki susunan karakter yang sama
34	Nomor Proses	6.1
	Nama Proses	Ubah ke Format SVM
	Sumber	Pengguna
	Input	Hasil Ekstraksi Fitur

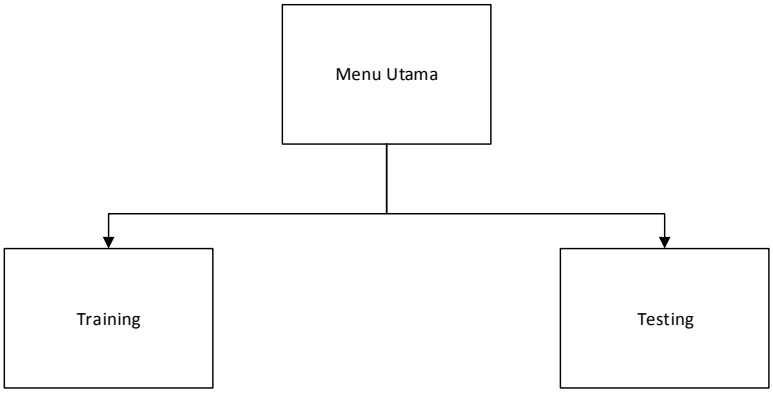
	Output	Format SVM
	Logika Proses	3. Pengecekan hasil ekstraksi fitur setiap pasangan kata 4. Pengubahan ke Format SVM
35	Nomor Proses	6.2
	Nama Proses	Hitung Nilai *x*
	Sumber	Pengguna
	Input	Data Format SVM
	Output	Nilai *x*
		2. Pencarian nilai x_i berdasarkan rumus kernel linier
36	Nomor Proses	6.3
	Nama Proses	Hitung Nilai *y*
	Sumber	Pengguna
	Input	Data Label
	Output	Kata per kalimat
	Logika Proses	2. Pencarian nilai *yi* berdasarkan rumus kernel linear
37	Nomor Proses	6.4
	Nama Proses	Hitung Support Vector setiap Pasangan
	Sumber	Pengguna
	Input	Nilai x dan nilai y
	Output	Support Vector setiap Pasangan
	Logika Proses	2. Menggunakan rumus kernel trick phi dengan mengambil nilai x dan y
38	Nomor Proses	6.5
	Nama Proses	Subtitusi Support Vector setiap Pasangan pada Kernel SVM Training
	Sumber	Pengguna
	Input	Support Vector kernel Training
	Output	Nilai w . Support Vector Testing
	Logika Proses	3. Mengalikan Nilai Support Vector terhadap nilai w pada Training
39	Nomor Proses	6.6
	Nama Proses	Pengecekan Nilai Hyperplane
	Sumber	Pengguna
	Input	Nilai w. Support Vector Testing
	Output	Hasil Klasifikasi
	Logika Proses	1. Mengecek nilai b hasil training 2. Kelas 1 *coreference* apabila nilai w. support vector testing lebih besar dari nilai b dan kelas -1 jika sebaliknya

3.10. Perancanngan Sistem

Pada bagian ini akan dijelaskan perancangan sistem yang untuk mengimplementasikan hasil analisis yang telah dilakukan. Sistem yang dibuat akan digunakan untuk mensimulasikan metode yang digunakan. Perancangan sistem yang dibuat terdiri dari perancangan struktur tabel, perancangan menu, perancangan antarmuka, dan jaringan semantik.

3.10.1. Perancangan Menu

Berikut ini adalah perancangan struktur menu yang akan diterapkan pada aplikasi. Adapun perancangan struktur menu dapat dilihat pada gambar berikut ini.



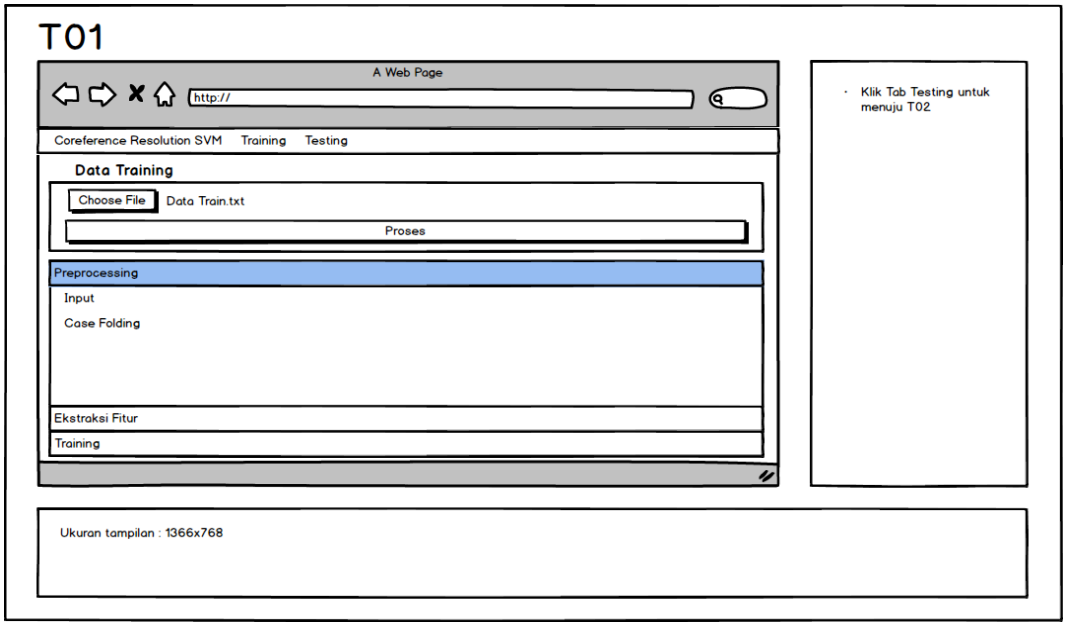
Gambar 3. 19 Perancangan Menu

3.10.2. Perancangan Antarmuka

Perancangan antarmuka merupakan suatu bentuk tampilan dari aplikasi yang akan dibuat untuk kebutuhan interface dengan pengguna. Adapun perancangan antarmuka dapat dilihat pada gambar di bawah ini.

1. Antarmuka Training

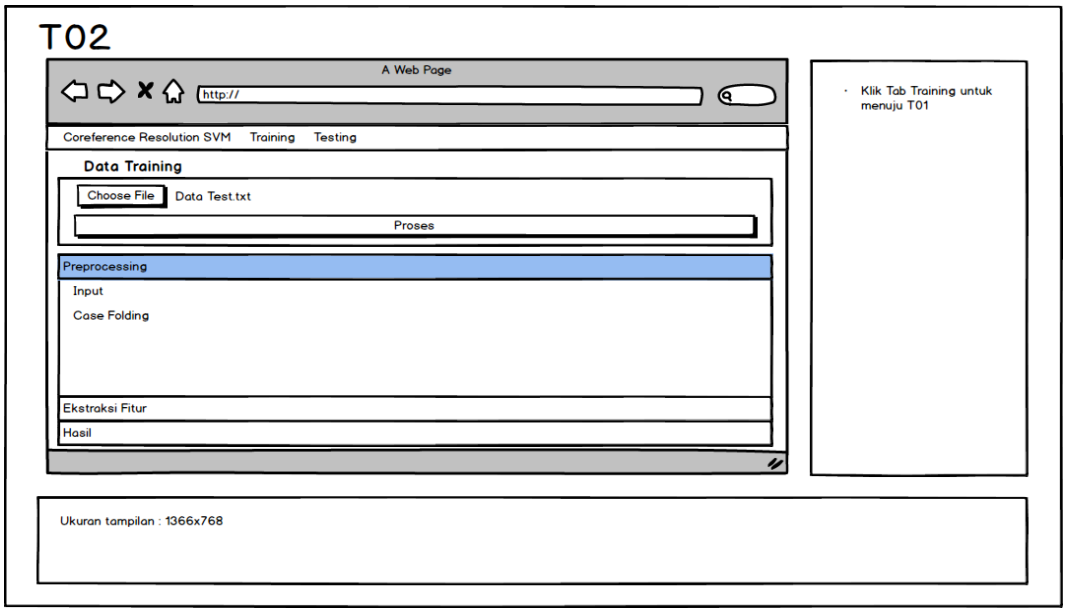
Berikut adalah antarmuka halaman training :



Gambar 3. 20 Perancangan Antar Muka Training

2. Antarmuka Testing

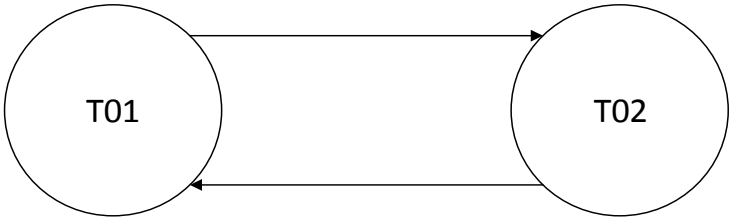
Berikut adalah antarmuka halaman testing :



Gambar 3. 21 Perancangan Antar Muka Testing

3.9.3. Jaringan Semantik

Jaringan semantik menggambarkan keterhubungan dari navigasi menu dari satu antarmuka ke antarmuka lain. Jaringan semantik yang terbentuk pada sistem ini dapat dilihat pada gambar di bawah ini.



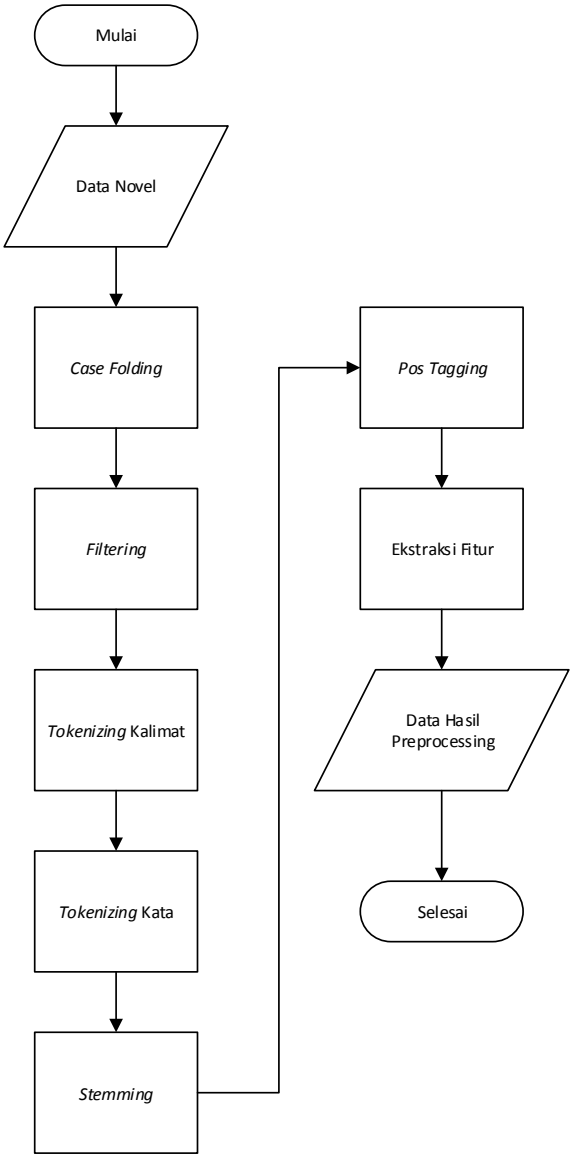
Gambar 3. 22 Jaringan Semantik

3.9.4. Perancangan Prosedural

Perancangan prosedural merupakan perancangan yang dilakukan untuk menetapkan detail tahap-tahap yang akan dinyatakan ke dalam suatu perangkat lunak.

3.9.4.1. *Preprocessing*

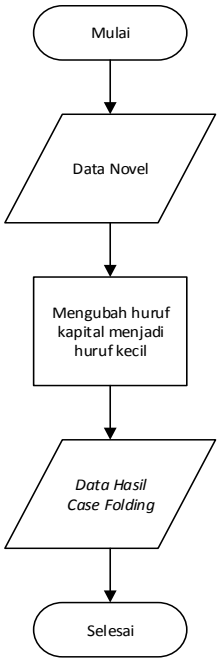
Preprocessing merupakan proses untuk mempersiapkan data menjadi data yang siap diolah. Proses *preprocessing* yang dilakukan akan mempengaruhi hasil klasifikasi karena dari *preprocessing* ini akan diambil nilai bobotnya dengan memanfaatkan ekstraksi fitur. Berikut adalah gambar prosedural *preprocessing*.



Gambar 3. 23 Perancangan Prosedural *Preprocessing*

1. ***Case Folding***

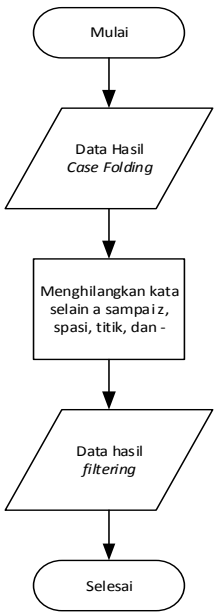
   Berikut adalah perancangan procedural tahap *case folding :*



Gambar 3. 24 Perancangan Prosedural *Case Folding*

2. ***Filtering***

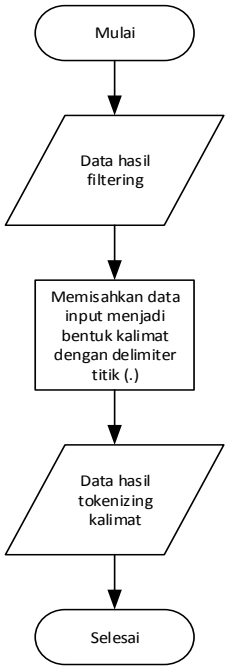
   Berikut adalah perancangan procedural tahap *filtering :*



Gambar 3. 25 Perancangan Prosedural *Filtering*

3. *Tokenizing* Kalimat

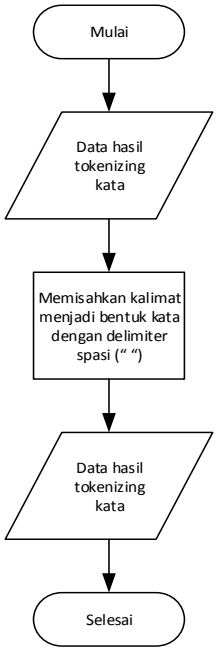
Berikut adalah perancangan procedural tahap *tokenizing kalimat :*



Gambar 3. 26 Perancangan Prosedural *Tokenizing* Kalimat

4. *Tokenizing* Kata

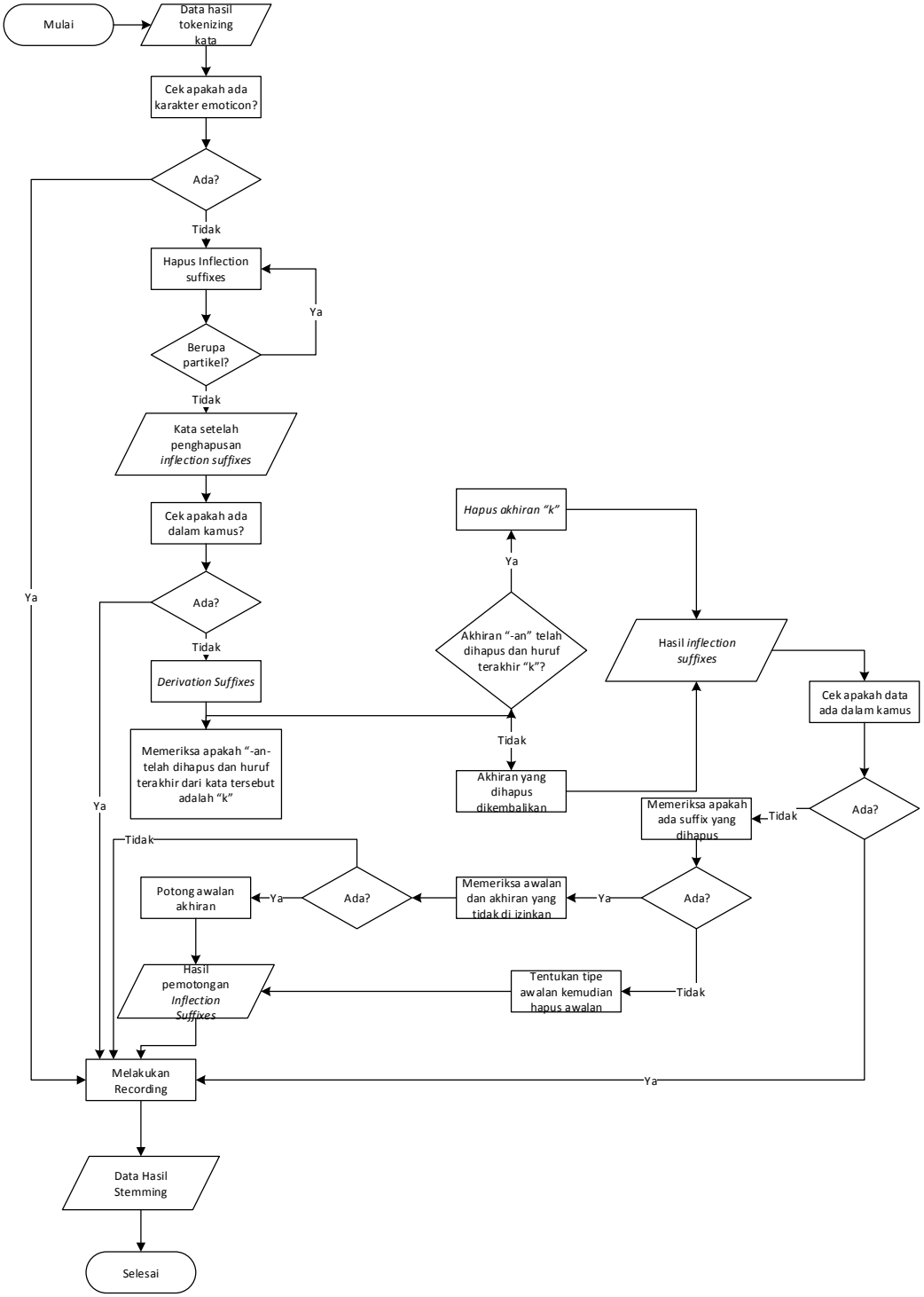
Berikut adalah perancangan procedural tahap *case kata :*



Gambar 3. 27 Perancangan Prosedural *Tokenizing* Kata

5. *Stemming*

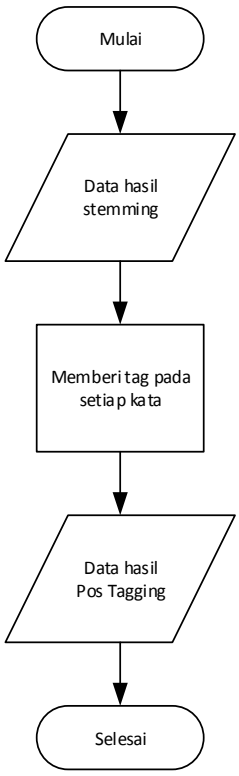
Berikut adalah perancangan procedural tahap *stemming :*



Gambar 3. 28 Perancangan Prosedural *Stemming*

6. POS *Tagging*

Berikut adalah perancangan procedural tahap POS *Tagging :*



Gambar 3. 29 Perancangan Prosedural POS *Tagging*