PENERAPAN METODE SUPPORT VECTOR MACHINE PADA PART OF SPEECH TAG BAHASA INDONESIA

Asep Sumpena Nugraha[1], Ken Kinanti Purnamasari[2]

[1,2]Universitas Komputer Indonesia
Jl. Dipati Ukur No.112-116, Lebakgede, Coblong, Kota Bandung, Jawa Barat 40132
E-mail : asepsn7@gmail.com[1], ken.kinanti@email.unikom.ac.id[2]

ABSTRAK

POS Tag merupakan pengklasifikasian kata ke dalam kelas katanya. Dalam pembuatan POS Tag terdapat kesulitan terkait ambiguasi karena struktur atau *grammar* suatu bahasa seringkali berbeda atau beragam. Masalah tersebut bisa diatasi dengan machine learning. Metode yang digunakan dalam penelitian ini yaitu Support Vector Machine atau yang sering disebut dengan SVM. Dalam proses SVM untuk membatasi antar kelasnya menggunakan Kernel Linier. Pada dasarnya SVM merupakan machine learning yang hanya bisa mengklasifikasikan dua kelas saja namun setelah dikembangkan, SVM dapat mengklasifikasikan lebih dari dua kelas dengan sebuah teknik yang bernama multiclass. Setelah melihat beberapa penelitian mengenai multiclass diputuskan dalam penelitian ini akan menggunakan Multiclass One Versus All (OVA). Data masukan yang digunakan sebanyak 50.005 token pada proses *training* sedangkan untuk keseluruhan proses *testing* totalnya sebanyak 12.111 token menghasilkan akurasi sebesar 54,29%. Hal tersebut diakibatkan oleh fitur pada saat pengambilan informasi masih kurang efektif.
Kata kunci: Part of Speech Tag, Ambiguasi, Support Vector Machine, Kernel Linier, Multiclass One Versus All.

1. PENDAHULUAN

Part of Speech Tag atau yang sering disebut POS Tag bertujuan untuk mengklasifikasikan setiap kata ke dalam kelas kata yang sesuai (kata kerja, kata sifat, dll). POS Tag berperan terhadap bidang NLP lainnya seperti *Question-Answering, Speech Recognition, information retrieval system, machine translation system, word sense disambiguation system*, dan sebagainya [1].

Penelitian POS Tag telah dilakukan pada teks bahasa Indonesia. Pada penelitian tersebut, metode yang digunakan adalah rule based dengan kelas kata berjumlah 23 buah dan memiliki tingkat akurasi sebesar 79% [2]. Berdasarkan hasil dari penelitian tersebut, Rule Based memiliki masalah terkait ambiguasi yang mengakibatkan penurunan tingkat akurasi. Ambiguasi merupakan kata yang memiliki lebih dari satu kelas kata [3]. Hal ini disebabkan oleh sulitnya pembuatan aturan Rule Based dalam menangani struktur dan *grammar* bahasa. Masalah tersebut dapat ditangani dengan *machine learning.* Ada beberapa metode yang belum diterapkan pada POS Tag bahasa Indonesia diantaranya adalah SVM, KNN, dan Decision Tree. Dalam penelitian mengenai komparasi metode pada POS Tag bahasa thailand, metode yang paling baik dari sisi akurasinya adalah SVM diikuti oleh Decision Tree dan yang terakhir yaitu KNN [4]. Penelitian lainnya yang menunjukkan SVM memiliki akurasi yang lebih baik dalam membandingkan ketiga metode tersebut adalah penelitian yang dilakukan oleh Amr [5].

Penelitian mengenai penanganan ambiguasi telah dilakukan dengan menggunakan SVM dapat kita lihat pada penelitian yang dilakukan oleh Gimenez untuk POS Tag bahasa Inggris dan bahasa Spanyol. Nilai akurasi yang didapat sangatlah baik yaitu sebesar 97,16% untuk bahasa Inggris dan 98,86% untuk bahasa Spanyol [6] [7]. Selain penelitian yang dilakukan oleh Gimenez, SVM pada POS Tag bahasa Inggris juga telah dilakukan oleh Nakagawa. Data yang digunakan sebanyak 2.416 kalimat atau 56.684 kata. Penelitian itu menghasilkan akurasi sebesar 97,11% [8].

Berdasarkan penelitian-penelitian tersebut, POS Tag bahasa Indonesia yang akan dibuat menggunakan SVM kemungkinan memiliki akurasi yang tinggi juga. Dengan alasan lainnya juga bahwa metode SVM belum pernah diterapkan pada POS Tag bahasa Indonesia maka diputuskan untuk membuat penelitian mengenai sistem POS Tagger bahasa Indonesia menggunakan SVM.

2. ISI PENELITIAN

Pada bagian isi penelitian ini akan dijelaskan mengenai POS Tag, metode penelitian, arsitektur sistem, data masukan, tokenisasi, ekstraksi fitur, *training* SVM, *testing* SVM, dan hasil pengujian.

2.1 POS Tag

POS Tag dapat diartikan dengan menentukan kategori morfosintaktis setiap kata dalam kalimat yang diberikan [7]. POS Tag juga dapat diartikan dengan pengklasifikasian dengan mengkategorikan setiap kata ke dalam kelas katanya.

Tabel 1. Contoh POS Tag

Teks	Token	Kelas
Saya memegang tongkat.	Saya	PRP
	memegang	VB
	tongkat	NN
	.	Z

Pada Tabel 1 terdapat teks "Saya memegang tongkat.". Sebelum diklasifikasikan, teks tersebut dibagi menjadi beberapa token. Setelah itu token-token diklasifikasikan berdasarkan kelas kata yang sesuai. "Saya" termasuk ke dalam kelas kata PRP (Pronomina Persona) dimana dalam kelas tersebut diartikan sebagai kata ganti orang, contoh lainnya adalah dia, kamu, aku, mereka, kalian. Kata kedua "memegang" termasuk ke dalam kelas kata VB (Kata Kerja) yang artinya sedang melakukan suatu aktivitas atau pekerjaan. Kata ketiga yaitu "tongkat" yang merupakan kelas kata NN (Noun/kata benda) yang tidak lain adalah berupa benda atau yang sering digunakan sebagai subjek maupun objek. Token terakhir adalah "." yang termasuk ke dalam kelas kata tanda baca atau Z.

Jumlah kelas POS Tag tiap bahasa biasanya berbeda-beda, hal ini disebabkan oleh bentuk tata bahasa yang beragam. Kelas yang digunakan pada penelitian ini disesuaikan dengan kelas-kelas yang ada pada data *training* yang telah dibuat oleh UI POS Tag dalam bentuk korpus dengan format TSV [9] yaitu berjumlah 23. Kelas-kelas tersebut diantaranya.

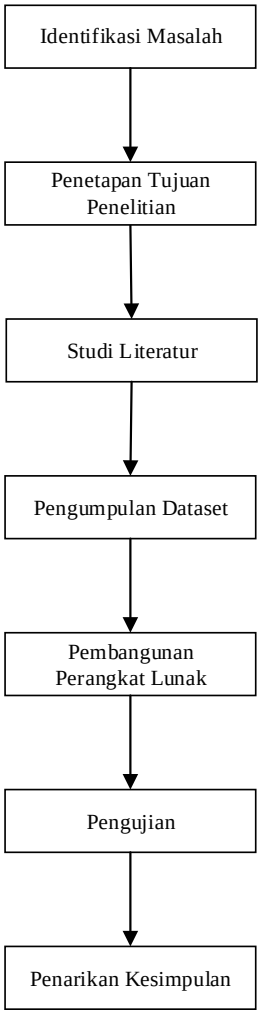
Tabel 2. Kelas Target

Tag	Deskripsi
CC	Kata penghubung atau bisa disebut juga *coordinating conjunction*, biasanya digunakan untuk menghubungkan satu kalimat dengan kalimat lainnya.
CD	*Cardinal Number*, merupakan suatu numerik atau kata yang menunjukan sebuah numerik. Contohnya adalah 1, dua, beberapa.
OD	*Ordinal Number*, merupakan kata atau nilai yang mengindikasikan posisi. Perbedaan kelas ini dengan kelas CD, jika CD lebih ke numerik atau jumlah tetapi OD lebih ke menunjukkan posisi. Contohnya adalah ke-1, kedua, terakhir.
DT	Biasanya disimpan di depan kata benda untuk menandainya. Bisa berupa pasti atau tak tentu. Kelas ini bisa juga disebut *Determiner*
FW	Foreign Word, merupakan kata dari bahasa asing yang tidak terdapat dan belum diadaptasi dan diserap oleh kamus bahasa Indonesia atau KBBI.
IN	Kelas kata ini hampir sama dengan kelas kata CC yang merupakan kata penghubung namun biasanya disimpan di depan preposisinya dan menghasilkan kata preposesi.
JJ	Merupakan kata dimana deskripsi, modifikasi, atau beberapa properti spesifik dari frasa noun. Kelas ini biasanya disebut dengan *adjective.*
MD	Modal dan kata kerja bantu
NEG	Merupakan kata yang bersifat negatif atau penolakan
NN	Noun, merupakan kata yang menunjukan manusia, binatang, konsep, arah, berkaitan dengan waktu, dan mata uang.
NNP	Merupakan nama spesifik dari seseorang, geografi, negara, organisasi, institusi atau perusahaan, hari, bulan, kompetisi, dan simbol stok. Kelas kata ini juga dapat disebut dengan proper nomina.
NND	Merupakan kata yang mengindikasikan pada kata timbangan, membuat sebuah hal yang tadinya tidak bisa dihitung menjadi bisa dihitung.
PR	Dalam bahasa inggris sering disebut dengan *Demonstrative pronoun*, mengimplikasikan penunjukan tempat objek.
PRP	Kata ganti orang, baik itu untuk perorangan maupun lebih dari satu orang seperti kami
RB	Kata yang menerangkan kata sifat dan kata kerja.
RP	Particle, merupakan partikel yang biasanya terdapat pada kalimat deklaratif, interogatif, ataupun imperative.
SC	*Subordinating conjunction* atau bisa juga disebut dengan subordinator, merupakan penghubung antara 2 atau lebih klausa yang biasanya dibagi menjadi klausa utama dan klausa pendukung.
SYM	*Symbol*, token yang masuk dalam kelas kata ini adalah simbol mata uang seperti "$".
UH	Kata untuk menyeru.
VB	Kata kerja atau aktivitas yang dilakukan.
WH	Kata yang bersifat tanya atau *Question word*.
X	Tipe kata yang termasuk dalam kelas kata ini adalah kesalahan ejaan atau *typo* ataupun kata yang tidak termasuk pada kelas kata lainnya.

Z	Tanda baca atau yang sering disebut dengan Punctuation.

2.2 Metode Penelitian

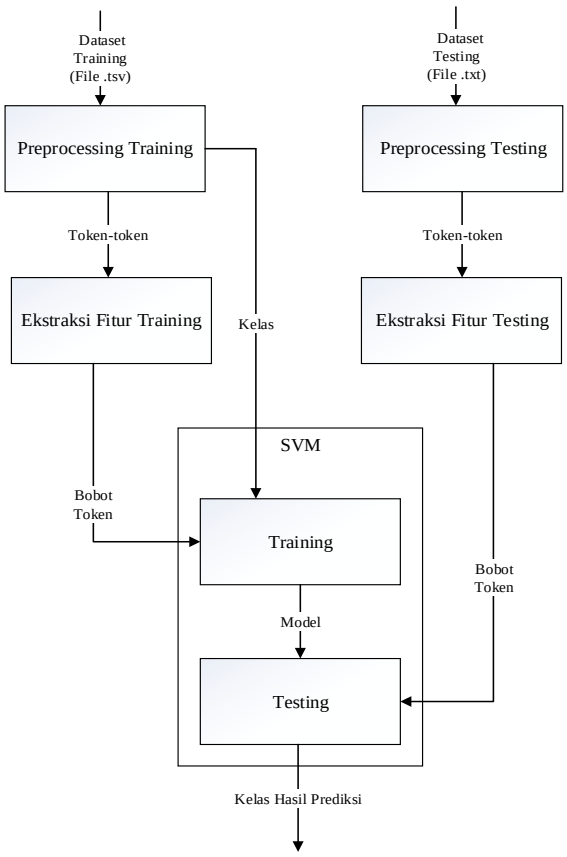
Metode penelitian yang digunakan adalah Deskriptif [10]. Metode ini memiliki tujuh tahap diantaranya yaitu identifikasi masalah, penetapan tujuan penelitian, pengumpulan dataset, pembangunan perangkat lunak, pengujian, dan penarikan kesimpulan.



Gambar 1. Metode Penelitian

2.3 Arsitektur Sistem

Dalam pembangunan sistem untuk mengimplementasikan Support Vector Machine pada Part of Speech Tag bahasa Indonesia memiliki beberapa tahapan. Tahapan-tahapan tersebut bisa dilihat pada Gambar 2.



Gambar 2. Arsitektur Sistem

2.4 Data Masukan

Data *training* diambil dari penelitian rule based sebelumnya yang terdiri dari token dan kelasnya dengan pemisah berupa tab dalam format file .tsv.

Jamrud	NNP
Bahia	NNP
kemudian	RB
dibawa	VB
ke	IN
Las	NNP
Vegas	NNP
untuk	SC
dilihat	VB
oleh	IN
calon	NN
pembeli	NN
namun	CC
disita	VB

oleh	IN
polisi	NN
Los	NNP
Angeles	NNP
.	Z

Gambar 3. Data *Training*

Sedangkan untuk data *testing*-nya diambil dari Panlocalization. Format file yang digunakan adalah .txt. Isi dari file tersebut yaitu teks bahasa indonesia.

Korea Selatan dan Jepang terus membawa keuntungan.

Gambar 4. Data *Testing*

2.5 Tokenisasi

Pada proses *preprocessing* digunakan tokenisasi(*tokenizing*), proses ini membagi teks ke dalam token-token. Tanda baca maupun simbol dikategorikan ke dalam kelas kata 'Z' atau 'SYM' [11]. Tokenisasi *training* dan tokenisasi *testing* memiliki proses yang berbeda. Pada tokenisasi *training*, data dipisahkan dengan tab karena file yang digunakan berupa file .tsv.

Gambar 5. Tokenisasi training

Sedangkan pada tokenisasi *testing* setiap teksnya dipisahkan berdasarkan tanda baca walaupun tanda baca tersebut diambil sebagai kelas kata juga. Berikut alur tokenisasi *testing.*

Gambar 6. Tokenisasi Testing

2.6 Ekstraksi Fitur

Fitur bisa diartikan sebagai ciri/informasi, jadi proses ekstraksi fitur adalah proses mengambil ciri/informasi yang terkandung dalam sebuah data [12]. Ekstraksi fitur berguna untuk membedakan antara kelas satu dengan kelas lainnya. Fitur-fitur yang digunakan pada penelitian ini ada 14 fitur [3] [6] [7] [13] diantaranya.

Tabel 3. Fitur

Fitur	Ketentuan	Contoh
Caps	Bernilai *True* jika awal huruf dari token yang diperiksa berupa kapital	***G**ates*, ***D**ia*
In_Cap	Bernilai True jika token yang diperiksa mengandung kapital kecuali awal huruf	*i**P**hone*
All_Cap	Bernilai True jika semua huruf dari token yang diperiksa berupa kapital	***SBY***
All_Low	Bernilai True jika semua huruf dari token yang diperiksa berupa huruf kecil	***Selalu, bekerja***
Num	Bernilai True jika token yang diperiksa merupakan bilangan numerik.	***1, 20***
Hyp	Bernilai True jika token yang diperiksa mengandung simbol ("-").	*Bolak-balik, kejar-kejaran*
me-	Bernilai True jika token yang diperiksa mengandung imbuhan "me"	***Me**lakukan, **me**melihara*
pe-	Bernilai True jika token yang diperiksa mengandung imbuhan "pe"	***Pe**laksana-an, **pe**meriksaan*
ke-	Bernilai True jika token yang diperiksa mengandung imbuhan "ke"	***Ke**laparan, **ke**keliruan*
se-	Bernilai True jika token yang diperiksa mengandung imbuhan "se"	***Se**ekor*
be-	Bernilai True jika token yang diperiksa mengandung imbuhan "be"	***Ber**main, **ber**napas*
di-	Bernilai True jika token yang diperiksa mengandung imbuhan "di"	***Di**diamkan, **di**rendam*
-an	Bernilai True jika token yang diperiksa mengandung imbuhan "an"	*Pengaku**an,** kekeliru**an***
-kan	Bernilai True jika token yang diperiksa mengandung imbuhan kan	*Panas**kan,** kobar**kan***

2.7 SVM

Support Vector Machine (SVM) adalah sebuah machine learning yang pertama kali diperkenalkan oleh Vapnik. SVM merupakan supervised machine learning yang dapat menyelesaikan berbagai masalah seperti kategori teks, tulisan tangan, digit recognition, tone recognition, klasifikasi gambar dan deteksi objek, dan klasifikasi data [14].

Dalam metode SVM, poin utamanya yaitu untuk mengoptimalkan yang namanya *hyperplane*. *Hyperplane* digunakan sebagai batas yang memisahkan *support vector* kelas satu dengan *support vector* kelas lainnya. Dengan mengoptimalkan *Support vector* khususnya support vector yang berdekatan antara kelas satu dengan kelas lainnya dijadikan sebagai tolak ukur untuk batas klasifikasi agar *hyperplane* yang akan dibuat menjadi optimal. *Vector* ini berasal dari dataset yang sudah diubah menjadi nilai *vector* melalui *vectorization* setelah proses ekstraksi fitur dan dijadikan sebagai *support vector*. Sebagai contoh pada dataset *training* terdiri dari x dan y dalam bentuk $\{(x_1, y_1), \dots, (x_n, y_n)\}$ dimana x disebut dengan *vector* dan y adalah label kelasnya [15].

2.7.1 Kernel

Kernel pada metode Support Vector Machine adalah pemisah antara kelas satu dengan kelas lainnya. Ada beberapa kernel untuk support vector machine diantaranya Linear, Polinomial, dan Radial Basis Function(RBF). Dalam penelitian-penelitian sebelumnya, kernel yang sering digunakan adalah Linear dan Radial Basis Function(RBF) [16].

a. Linear

Kernel linear menggunakan garis lurus sebagai pembatas/*hyperplane* antar kelas. Kernel linier hanya membutuhkan dua variabel diantaranya x_i dan x_j. Variabel tersebut merupakan *vector* dari hasil *vectorization* terhadap nilai bobot ekstraksi fitur. Dalam perhitungannya, nilai *vector* x_i yang ditranspose terlebih dahulu sebelum dikalikan dengan x_j. Sama halnya dengan kernel linear terhadap label hasil pelabelan pada kelas target.

$$k\left(x_i, x_j\right) = x_i^T x_j \qquad (1)$$

b. RBF

Dalam penyelesaiannya RBF membutuhkan parameter gamma dan C. Gamma berfungsi sebagai batas keputusan dan wilayah keputusan, sebagai contoh jika gamma bernilai kecil maka batas keputusan akan kecil namun wilayah keputusan akan menjadi luas dan begitupun sebaliknya. Nilai gamma yang digunakan harus lebih besar dari nol. C berfungsi sebagai penalti terhadap kesalahan dalam klasifikasi. *x* diambil dari *vector* hasil *vectorization*. Exp adalah eksponen dari hasil perhitungan x dan gamma.

$$exp(-\gamma||x_i - x||^2), \gamma > 0 \qquad (2)$$

c. Polinomial

Kernel Polinomial memiliki dua parameter berbeda dari kernel lain. Parameter r adalah parameter bebas yang jika diisi r sama dengan nol maka disebut homogen. Sedangkan parameter d adalah derajat/quadrat yang umumnya diisi dengan d sama dengan 2.

$$\mathrm{k}(x_i, x) = \left(y.\ x_i^T\ x + r\right)^d \qquad (3)$$

Dalam penelitian ini, kernel yang digunakan adalah kernel linear dengan mengacu pada penelitian yang dilakukan oleh Gimenez yang memiliki hasil akurasi diatas 90%.

2.7.2 Multiclass

Terdapat dua teknik multi class yang sering digunakan pada SVM yaitu One Versus One (OVO) dan One Versus All (OVA). OVA yang dimaksud yaitu membandingkan satu dengan semua selain dirinya yang dianggap menjadi satu kesatuan. Multi class ini digunakan karena sejatinya SVM adalah machine learning yang hanya mengklasifikan dua kelas saja secara linear.

Pada penelitian-penelitian sebelumnya, diketahui bahwa akurasi OVA lebih baik dibanding OVO walaupun dari segi kecepatan OVO lebih cepat [17]. Maka dari itu, pada penelitian yang akan dilakukan mengenai penerapan SVM terhadap POS Tag bahasa Indonesia ini menggunakan OVA. Dengan metode tersebut diharapkan dapat memberikan akurasi yang tinggi.

Tabel 4. Multiclass One Versus All

$h_i = 1$	$h_1 = -1$	Hipotesis
Kelas 1	Bukan Kelas 1	$f^1(g) = (w^1)g + b^1$
Kelas 2	Bukan Kelas 2	$f^2(g) = (w^2)g + b^2$
Kelas 3	Bukan Kelas 3	$f^3(g) = (w^3)g + b^3$
Kelas 4	Bukan Kelas 4	$f^4(g) = (w^4)g + b^4$

Dalam menerapkan metode OVA akan dibangun z buah model SVM biner. z disini adalah jumlah kelas. Dalam mengklasifikasikan hal tersebut dapat dilihat pada persamaan berikut [18].

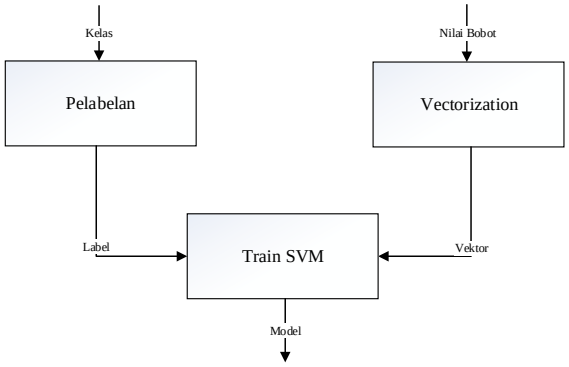
$$Kelas\ g = arg\ \max_{r=1..z}\left(\left(w^{(r)}\right)^T . \varphi\begin{bmatrix}p_i\\q_i\end{bmatrix} + b^{(r)}\right) \qquad (4)$$

Persamaan tersebut digunakan pada tahap terakhir proses testing setelah *support vector* dari data *testing* didapatkan. $\left(w^{(r)}\right)^T$ adalah *hyperplane* yang jumlahnya sebanyak *z* atau jumlah kelas dari data *training*. *Hyperplane* merupakan model atau pembatas kelas dari hasil perhitungan *training* SVM. Selain itu, ada juga hasil perhitungan *training* SVM lainnya $b^{(r)}$ atau bias. Support vector data testing $\varphi\begin{bmatrix}p_i\\q_i\end{bmatrix}$ dimasukan ke dalam persamaan sebagai

komponen dari data *testing* yang akan diklasifikasikan. Hasil klasifikasi ditentukan dengan *arg max* bahwa nilai tertinggilah yang akan diambil dari hasil perhitungan semua *hyperplane* dengan *support vector* data *testing* sebagai kelas targetnya atau juga bisa disebut sebagai kelas prediksi.

2.7.3 *Training* SVM

Pada tahap *training* terdapat beberapa proses yang dilakukan. Proses-proses tersebut diantaranya *Vectorization*, Pelabelan, dan SVM. Berikut alur *training* SVM.



Gambar 7. *Training* SVM

Pada proses *training* SVM langkah pertama adalah mengubah bobot fitur hasil ekstraksi fitur menjadi format yang bisa diterima oleh SVM yaitu nilai yang berbentuk *vector*. Proses tersebut disebut dengan *vectorization*. Langkah selanjutnya adalah memberi label pada kelas dengan 1 atau -1. Dalam penelitian ini proses pelabelan menggunakan multiclass One Versus All(OVA). Setelah itu masukan pada kernel, pada penelitian ini kernel yang digunakan adalah kernel linear.

$$\sum_{i=1}^{n} x_i x_j = x_i^T x_j (i, j = 1, \dots, n) \quad (5)$$

$$\sum_{i=1}^{n} y_i y_j = y_i^T y_j (i, j = 1, \dots, n) \quad (6)$$

Nilai *x* diambil dari vector hasil vectorization. Sedangkan nilai *y* diambil dari hasil pelabelan kelas. Lakukan langkah tersebut dari *i* sama dengan 1 hingga ke-*n*, begitupun dengan *j*. *n* disini adalah jumlah token atau data. Setelah itu buat matriks dengan menggunakan persamaan berikut.

$$C = \begin{bmatrix} x_i^T x_j & \cdots & x_i^T x_{jn} \\ \vdots & \ddots & \vdots \\ x_{in}^T x_j & \cdots & x_{in}^T x_{jn} \end{bmatrix} \quad (7)$$

$$D = \begin{bmatrix} y_i^T y_j & \cdots & y_i^T y_{jn} \\ \vdots & \ddots & \vdots \\ y_{in}^T y_j & \cdots & y_{in}^T y_{jn} \end{bmatrix} \quad (8)$$

Pada persamaan tersebut, nilai-nilai dari matriks C diambil dari Persamaan 5 dan nilai-nilai dari D diambil dari Persamaan 6. *i* dimulai dari 1 hingga ke-*in*, sedangkan *j* dimulai dari 1 hingga ke-*jn*. *in* dan *jn* merupakan jumlah dari data. setelah selesai tambahkan tiap baris matriksnya sehingga didapat nilai *k* dan *l*.

$$k_i = x_i^T x_1 + \cdots + x_i^T x_j \quad (9)$$

$$l_i = y_i^T y_1 + \cdots + y_i^T y_j \quad (10)$$

Lakukan hal tersebut dari *i* sama dengan 1 hingga ke-*n*. *n* adalah jumlah keseluruhan data *training* Selanjutnya lakukan persamaan berikut.

$$\varphi \begin{bmatrix} k_i \\ l_i \end{bmatrix} = \begin{cases} \sqrt{{k_n}^2 + {l_n}^2} > 2, maka \begin{bmatrix} \sqrt{{k_n}^2 + {l_n}^2} - k_i + |k_i - l_i| \\ \sqrt{{k_n}^2 + {l_n}^2} - l_i + |k_i - l_i| \end{bmatrix} \\ \sqrt{{k_n}^2 + {l_n}^2} \leq 2, maka \begin{bmatrix} k_i \\ l_i \end{bmatrix} \end{cases} \quad (11)$$

Perhitungan dimulai dengan mengecek apakah hasilnya akan bernilai lebih dari 2 atau bisa jadi sama dengan maupun kurang dari 2. Jika hasilnya lebih dari 2 maka rumus yang digunakan adalah yang berada diatas, namun jika tidak maka hasilnya sama dengan $\begin{bmatrix} k_i \\ l_i \end{bmatrix}$ [19]. Nilai *k* didapat dari Persamaan 9 sedangkan nilai *l* didapat dari Persamaan 10. Lakukan hal tersebut dari i sama dengan *1* hingga *i* sama dengan *n*. *n* adalah jumlah keseluruhan data training

Persamaan 11 akan menghasilkan nilai *support vector training*. Dalam mencari nilai α_i, *support vector* ditambahkan nilai bias 1 agar tegak lurus sempurna. Lalu lakukan persamaan berikut.

$$\sum_{i=1,j=1}^{n} \alpha_i s_i^T s_j = l_i \quad (12)$$

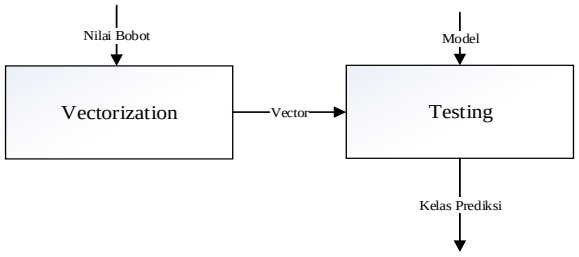
Lakukan langkah tersebut dari *i* sama dengan 1 hingga ke-*n*, begitupun dengan *j*. Dengan menggunakan software mapple maka didapat nilai a. Jika nilai α_i telah didapatkan, maka langkah selanjutnya adalah mencari nilai *w* dan *b* yang baru sebagai *hyperplane*.

$$w = \sum_{i=1}^{n} \alpha_i s_i \quad \text{dimana } \alpha_i \geq 0 \quad (13)$$

Dalam menghitung hyperplane, α_i yang digunakan harus bernilai positif. Dengan menggunakan persamaan tersebut nantinya akan didapat nilai w dan b secara bersamaan. Nilai tersebut dianggap sebagai *hyperplane*, kumpulan *hyperplane* disebut dengan model yang akan digunakan pada proses *testing* SVM.

2.7.4 *Testing* SVM

Testing adalah proses pemasukan data *testing* ke dalam model yang sudah dibuat selama tahap *training*. Pada tahap ini, data *testing* berupa bobot fitur setelah *preprocessing* dan ekstraksi fitur.



Gambar 8. *Testing* SVM

Sama halnya dengan proses *training*, bobot harus diubah menjadi *vector*, sedangkan untuk kelas diberi label. Pada proses pelabelan kelas untuk data *testing* dianggap sama dengan 0, sedangkan label untuk kelas data yang lainnya sama seperti pada proses *training* dimana token data *testing* disatukan dengan token-token data *training* untuk mencari nilai *support vector testing* [15]. Setelah itu cari variabel yang dibutuhkan dalam perhitungan kernel. Lakukan perhitungan berikut dengan *i* dari 1 hingga ke-*n*, begitupun dengan *j*. Nilai *n* adalah jumlah data.

$$\sum_{i=1}^{n} g_i g_j = g_i^T g_j (i, j = 1, \dots, n) \quad (14)$$

$$\sum_{i=1}^{n} h_i h_j = h_i^T h_j (i, j = 1, \dots, n) \quad (15)$$

Setelah itu petakan ke dalam matriks. Matriks E untuk vector bobot *testing* dan Matriks F untuk kelas *testing*.

$$E = \begin{bmatrix} g_i^T g_j & \cdots & g_i^T g_{jn} \\ \vdots & \ddots & \vdots \\ g_{in}^T g_j & \cdots & g_{in}^T g_{jn} \end{bmatrix} \quad (16)$$

$$F = \begin{bmatrix} h_i^T h_j & \cdots & h_i^T h_{jn} \\ \vdots & \ddots & \vdots \\ h_{in}^T h_j & \cdots & h_{in}^T h_{jn} \end{bmatrix} \quad (17)$$

Lalu cari nilai p_i dan q_i dengan menghitung Persamaan 18 dan Persamaan 19. $g_i^T g_j$ diambil dari tiap baris Matriks E dan $h_i^T h_j$ diambil dari tiap baris Matriks F.

$$p_i = g_i^T g_j + \cdots + g_i^T g_j \quad (18)$$

$$q_i = h_i^T h_j + \cdots + h_i^T h_j \quad (19)$$

Setelah didapat nilai kernel p_i dan q_i. maka cari *support vector*. Berikut persamaan untuk mencari *support vector testing*.

$$\varphi \begin{bmatrix} p_i \\ q_i \end{bmatrix} = \begin{cases} \sqrt{{p_n}^2 + {q_n}^2} > 2, \; maka \begin{bmatrix} \sqrt{{p_n}^2 + {q_n}^2} - p_i + |p_i - q_i| \\ \sqrt{{p_n}^2 + {q_n}^2} - q_i + |p_i - q_i| \end{bmatrix} \\ \sqrt{{p_n}^2 + {q_n}^2} \leq 2, \; maka \begin{bmatrix} p_i \\ q_i \end{bmatrix} \end{cases} \quad (20)$$

Sama halnya dengan proses training sebelum mulai menghitung *support vector*, cek terlebih dahulu apakah hasil dari $\sqrt{{p_n}^2 + {q_n}^2}$ lebih dari 2 atau kurang maupun sama dengan 2. Jika hasilnya lebih dari 2 maka gunakan rumus yang atas jika tidak maka support vectornya sama dengan $\begin{bmatrix} p_i \\ q_i \end{bmatrix}$. Tahap terakhir dari proses *testing* yaitu masukan *support vector* data *testing* ke dalam model *hyperplane* yaitu Persamaan 4. Hasil nilai terbesar pada perhitungan tersebut merupakan kelas prediksinya. Kelas prediksi ini akan dibandingkan dengan kelas aslinya untuk mendapatkan jumlah kelas prediksi benar dalam penghitungan nilai akurasi.

2.8 Hasil Pengujian

Pengujian dilakukan berdasarkan jumlah prediksi benar dari hasil klasifikasi token dan kelas tokennya dengan data *training* yang berjumlah 50.005 token dan data *testing* yang berjumlah 12.111 token.

Tabel 5. Hasil Pengujian

Nama	Jumlah Token	Jumlah Kelas Prediksi Benar
Testing1.txt	1.012	561
Testing2.txt	1.581	978
Testing3.txt	1.001	522
Testing4.txt	1.254	654
Testing5.txt	1.364	718
Testing6.txt	1.110	635
Testing7.txt	1.010	506
Testing8.txt	1.607	796
Testing9.txt	1.025	572
Testing10.txt	1.147	633
Total	12.111	6.575

Berdasarkan hasil pengujian yang telah dilakukan, maka hitung nilai akurasi dengan membagi total dari jumlah token dengan total jumlah kelas prediksi benar dan kalikan hasilnya dengan seratus persen. Nilai akurasi yang didapat adalah sebesar 54,29%.

3. PENUTUP

Hasil yang diperoleh melalui pengujian menunjukkan bahwa akurasi rata-rata yang didapatkan POS Tagger Indonesia menggunakan SVM pada penelitian ini adalah sebesar 54,29%. Hal tersebut dipengaruhi oleh fitur all_low pada ekstraksi fitur yang tidak efektif dalam membedakan antar kelasnya sehingga diperlukan fitur lainnya untuk membantu fitur 'all_low' tersebut agar lebih efektif.

Dikarenakan masih banyak kekurangan pada penelitian ini, ada beberapa saran yang dapat dilakukan untuk pengembangan selanjutnya. Berikut saran yang bisa diambil.

a. Pengecekan kata dasar sebelum proses pengambilan informasi fitur imbuhan. Jika token merupakan kata dasar maka semua fitur imbuhan bernilai False. Misalnya kata 'aku' merupakan kata dasar maka tidak memiliki imbuhan apapun. Jika token yang diperiksa bukan termasuk kata dasar maka token tersebut kemungkinan memiliki imbuhan seperti kata 'merusak' yang memiliki imbuhan 'me'.
b. Penambahan atau pengubahan fitur dalam proses ektraksi fitur. Misalnya menambahkan fitur imbuhan 'ter', 'sasi', dan 'iah'.

DAFTAR PUSTAKA

[1] R. J. P. M. C. Padma, "Morpheme Based Parts of Speech Tagger For Kannada," 2016.

[2] F. Rashel, A. Luthfi, A. Dinakaramani dan R. manurung, "Building an Indonesian Rule-Based Part-of-Speech Tagger," 2014.

[3] K. K. Purnamasari dan I. S. Suwardi, "Rule-based Part of Speech Tagger for Indonesian Language," *IOP Conference Series: Materials Science and Engineering,* vol. 407, 2018.

[4] T. Nomponkrang dan C. Sanrach, "The Comparison of Algorithms for Thai-Sentence Classification," 2016.

[5] A. E. Mohamed, "Comparative Study of Four Supervised Machine Learning Techniques for Classification," 2017.

[6] J. Gimenez dan L. Marquez, "SVMTool: A general POS tagger generator based on Support Vector Machine," 2004.

[7] J. Gimenez dan L. Marquez, "Fast and Accurate Part-of-Speech Tagging: SVM Aproach Revisited," 2004.

[8] T. Nakagawa, T. Kudo dan Y. Matsumoto, "Revision Learning and its Application to Part-of-Speech Tagging," 2002.

[9] F. R. A. L. R. M. Arawinda Dinakaramani, "Designing an Indonesia Part of Speech Tagset and Manually Tagged Indonesia Corpus," 2014.

[10] M. Nazir, Metode Penelitian, Bogor: Ghalia Indonesia, 2014.

[11] F. Saefulloh, "Part Of Speech Untuk Bahasa Indonesia Menggunakan Conditional Random Field(CRF)," 2017.

[12] A. L. Sari, "Coreference Resolution Dengan Menggunakan Metode SVM Pada Novel Bahasa Indonesia," 2017.

[13] H. Ramza, M. S. Kadar, F. Abdurrahman dan M. S. Ab-Rahman, "Analisis Hubungan Imbuhan Me dan Ber pada Setiap Perenggan dalam Tulisan Cerita Pendek "Kembali Kasih"," 2013.

[14] L. B. Durgesh K. Srivastava, "Data Classification Using Support Vector Machine," 2005.

[15] A. N. Amalia, "Implementasi Support Vector Machine (SVM) Pada Klasifikasi Laporan Skripsi (Studi Kasus: Teknik Informatika UNIKOM)," 2016.

[16] N. Guenther dan M. Schonlau, "Support Vector Machine," 2016.

[17] J. Milgram, M. Cheriet dan R. Sabourin, ""One Against One" or "One Again All": Which One is Better for Handwriting Recognition with SVMs?," 2006.

[18] G. M. Foody dan A. Mathur, "A Relative Evaluation Of Multi-Class Image Classification By Support Vector Machines," 2004.

[19] B. P. Utama, "Support Vector Machine Dalam Sistem Pendeteksi Kepribadian Berdasarkan Pola Tanda Tangan," 2017.