EKSTRAKSI INFORMASI DOKUMEN KARYA TULIS ILMIAH MENGGUNAKAN ALGORITMA LEARNING VECTOR QUANTIZATION

Firdamdam Sasmita[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 : firdamdamsasmita2211@gmail.com[1], ken.kinanti@email.unikom.ac.id[2]

ABSTRAK

Mendeteksi setiap komponen dokumen karya tulis ilmiah sulit dilakukan untuk formatnya yang beragam. Penelitian sebelumnya yang terkait pengenalan dokumen karya tulis ilmiah telah dilakukan untuk dokumen karya tulis ilmiah skripsi bahasa Indonesia. Berdasarkan pengujian pada 50 dokumen tahun 2013 dan 2017, diperoleh rata-rata akurasi sebesar 57%. Permasalahan tersebut dapat diatasi jika menggunakan *machine learning*. Algoritma *machine learning* yang digunakan pada penelitian ini adalah *LVQ*. Pada penelitian sebelumnya di bidang klasifikasi teks dan pengolahan citra, *LVQ* memiliki akurasi yang unggul dibandingkan dengan *SVM, KNN, FFBP, RBF, Naive Bayes*, dan *C4.5*. Algoritma *LVQ* memiliki kemungkinan memperoleh akurasi yang lebih baik apabila dilakukan di bidang ekstraksi informasi. Berdasarkan pengujian pada 40 dokumen karya tulis ilmiah skripsi tahun 2011 sampai 2018, diperoleh rata-rata akurasi token-kelas sebesar 78%, akurasi kelas-token sebesar 19%. Perolehan akurasi token-kelas disebabkan oleh penggunaan fitur pembobotan. Sedangkan, perolehan akurasi kelas-token disebabkan oleh pengaruh algoritma, salah ejaan, dan munculnya simbol tidak beraturan.

Kata kunci : Ekstraksi Informasi, *Learning Vector Quantization*, *LVQ*, Dokumen karya tulis ilmiah, Skripsi.

1. PENDAHULUAN

Ekstraksi informasi adalah suatu proses untuk mengubah informasi tidak terstruktur yang terdapat dalam teks ke dalam data terstruktur [1]. Ekstraksi informasi dilakukan dengan mendeteksi komponen-komponen yang ada pada setiap dokumen, salah satunya dokumen karya tulis ilmiah. Mendeteksi setiap komponen pada dokumen karya tulis ilmiah sulit dilakukan untuk dokumen yang formatnya beragam, karena dibutuhkan suatu pembuatan aturan ekstraksi yang sesuai untuk masing-masing dokumen.

Penelitian sebelumnya yang terkait pengenalan dokumen karya tulis ilmiah telah dilakukan untuk dokumen skripsi berbahasa Indonesia [2]. Penelitian tersebut melakukan ekstraksi informasi menggunakan sistem berbasis aturan untuk mendapatkan identitas pada dokumen skripsi, diantaranya cover, abstrak, dan abstract. Berdasarkan pengujian 3 dokumen tahun 2017, diperoleh akurasi sebesar 100%. Sedangkan, apabila pengujian dilakukan terhadap 50 dokumen yang disertakan dengan tahun 2013, diperoleh rata-rata akurasi sebesar 57%. Penurunan akurasi yang terjadi disebabkan oleh sistem berbasis aturan yang tidak mampu menangani dokumen dengan format beragam pada tahun 2013. Oleh karena itu, permasalahan tersebut dapat diatasi jika menggunakan *machine learning*.

Pada penelitian sebelumnya, penggunaan *machine learning* di bidang ekstraksi informasi telah dilakukan pada makalah ilmiah [3]. Begitupun pada bidang lainnya, seperti klasifikasi teks [4] dengan komparasi algoritma *SVM* (*Support Vector Machine*), *LVQ* (*Learning Vector Quantization*), dan *KNN* (*K-Nearest Neigbour*). Penelitian tersebut menyatakan bahwa algoritma yang lebih unggul adalah algoritma *LVQ* dengan perolehan akurasi sebesar 80%. Sama halnya seperti penelitian yang telah dilakukan oleh Novizon dan Zulkarnain [5] pada pengolahan citra dengan komparasi algoritma *FFBP*(*Feed-Forward Back-Propagation*), *RBF* (*Radial Basis Function*), dan *LVQ*, algoritma *LVQ* menghasilkan akurasi sebesar 100%. Sedangkan pada penelitian yang telah dilakukan oleh Umer dan Khiyal [6] di bidang klasifikasi teks dengan komparasi algoritma *LVQ*, *RBF*, *KNN*, *Naive Bayes*, dan C4.5, algoritma *LVQ* menghasilkan akurasi sebesar 84%. Oleh karena itu, algoritma *LVQ* memiliki kemungkinan untuk memperoleh akurasi yang lebih baik apabila dilakukan pada bidang ekstraksi informasi.

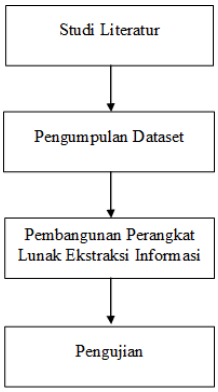
Dengan demikian, penelitian ini akan membangun sistem ekstraksi informasi menggunakan algoritma *LVQ* untuk dokumen karya tulis ilmiah berbahasa Indonesia, dengan batasan dokumen yang akan digunakan pada penelitian ini adalah dokumen karya tulis ilmiah skripsi Program Studi Teknik Informatika, Fakultas Teknik dan Ilmu Komputer, Universitas Komputer Indonesia.

2. ISI PENELITIAN

Isi penelitian menjelaskan mengenai metode penelitian, dokumen karya tulis ilmiah skripsi, arsitektur sistem, tokenisasi, segmentasi data, ekstraksi fitur, algoritma *LVQ*, *plotting* per kelas, dan hasil pengujian.

2.1 Metode Penelitian

Pada penelitian ini ada empat tahapan alur kerja penelitian, diantaranya studi literatur, pengumpulan *dataset*, pembangunan perangkat lunak, serta pengujian. Berikut blok diagram alur kerja penelitian.



Gambar 1. Blok Diagram Alur Kerja Penelitian

2.2 Dokumen Karya Tulis Ilmiah Skripsi

Pada penelitian ini, dokumen karya tulis ilmiah skripsi Program Studi Teknik Informatika, Fakultas Teknik dan Informatika, Universitas Komputer Indonesia digunakan sebagai *dataset* yang memiliki beberapa kategori pada lembar sampul dan abstrak. Jumlah *dataset* yang digunakan sebanyak 80 data, 40 dokumen dari tahun 2011 sampai dengan 2018 dan 40 dokumen dari tahun 2011 sampai dengan 2018. Berikut bagian – bagian kategori pada lembar sampul dan abstrak dapat dilihat pada Tabel 1.

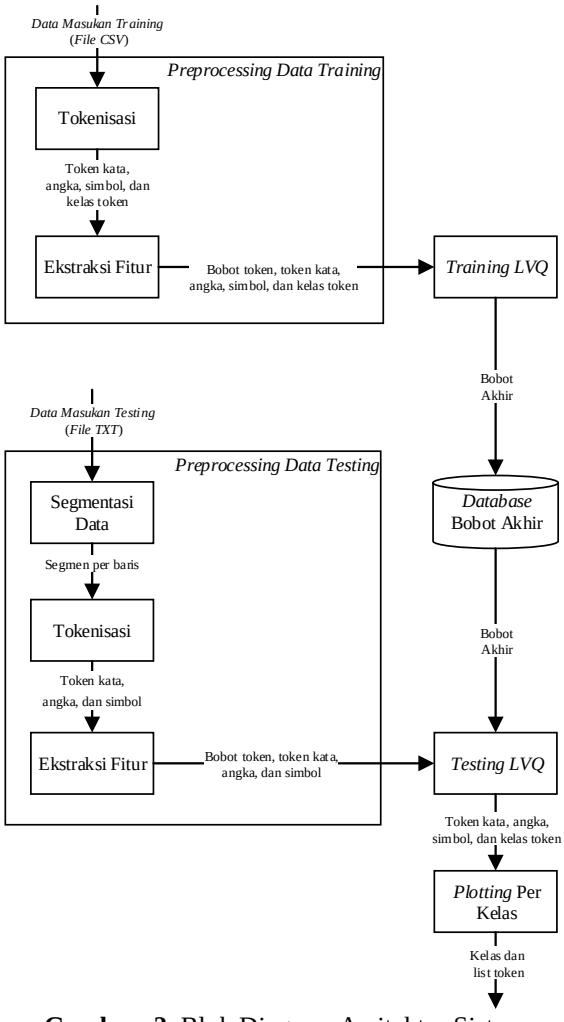
Tabel 1. Kategori Lembar Sampul Dan Abstrak

Lembar			
No	**Sampul**	**No**	**Abstrak**
1	Judul Penelitian (Sampul)	10	Judul Halaman Abstrak
2	Jenis Penelitian	11	Judul Penelitian (Abstrak)
3	Kalimat Pengajuan	12	Other
4	Penulis (Sampul)	13	Penulis (Abstrak)
5	NIM (Sampul)	14	NIM (Abstrak)
6	Program Studi	15	Isi Abstrak
7	Fakultas	16	Kata kunci
8	Universitas		
9	Tahun		

Beberapa kategori pada Tabel 1 direpresentasikan menjadi kelas berbentuk angka untuk proses pembelajaran dan klasifikasi algoritma *LVQ*. Total kelas yang akan digunakan pada penelitian ini sebanyak 16 kelas, terhitung dari nomor pada lembar sampul sampai dengan lembar abstrak.

2.3 Arsitektur Sistem

Pembangunan sistem ekstraksi informasi terdiri dari beberapa proses. Proses utama yang terdapat pada sistem ekstraksi informasi menggunakan algoritma *LVQ* terdiri dari data masukan *training* dan *testing*, *preprocessing data training* dan *testing*, *training LVQ*, *testing* LVQ, dan *Plotting* Per Kelas. Untuk beberapa proses lainnya, dapat dilihat pada blok diagram arsitektur sistem berikut ini.



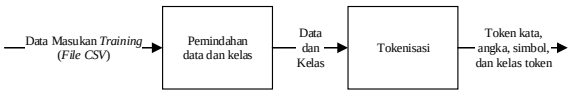
Gambar 2. Blok Diagram Arsitektur Sistem

Pada tahapan *preprocessing data training*, data masukan yang akan digunakan berupa *file CSV*. *File* tersebut berisi sekumpulan data hasil pemberian label (*plotting*) dari beberapa kategori yang terdapat pada *file PDF* hasil *scanning* lembar sampul dan abstrak skripsi yang dilakukan di luar sistem, misalkan kategori Judul. *File CSV* kemudian akan melalui

proses tokenisasi dan ekstraksi fitur. Tokenisasi akan menghasilkan token kata, angka, dan simbol, serta kelas token dari proses memindahkan sekumpulan baris data pada *file CSV* ke sebuah *array*. Token yang dihasilkan akan diberikan bobot melalui proses ekstraksi fitur. Setiap bobot token, token kata, angka, simbol dan kelas token akan melalui proses *training LVQ*, sehingga menghasilkan bobot akhir yang disimpan pada database bobot akhir, yang berbentuk *file CSV*.

2.4 Tokenisasi

Suatu proses untuk memisahkan setiap kata dari rangkaian kalimat dinamakan dengan proses tokenisasi. Menurut Amin, token seringkali disebut sebagai istilah (term) atau kata. Sebuah token merupakan suatu urutan karakter dari dokumen tertentu yang dikelompokkan sebagai unit semantik yang berguna untuk diproses [23]. Berikut blok diagram tokenisasi.



Gambar 3. Blok Diagram Tokenisasi

Pada penelitian ini, tokenisasi yang dilakukan tidak melalui tahapan *filtering* untuk menghilangkan sejumlah term yang tidak penting karena term-term yang tidak penting atau tidak memiliki makna justru sangat penting untuk ekstraksi fitur *punctuation*.

2.5 Segmentasi Data

Segmentasi data adalah proses yang digunakan untuk menghapus baris kosong dan menghasilkan data segmen per baris. Proses ini hanya dilakukan pada tahapan *preprocessing data testing* yang data masukannya berupa file teks. Berikut blok diagram segmentasi data.

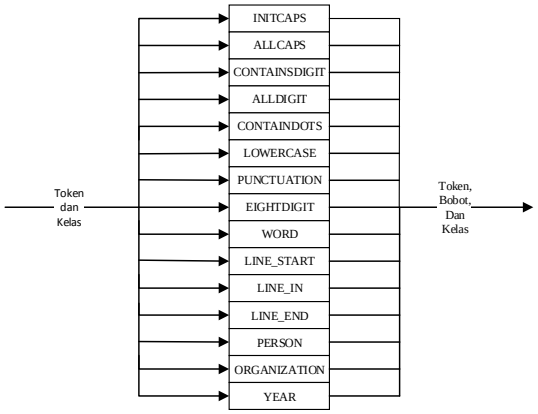


Gambar 4. Blok Diagram Segmentasi Data

Pada Gambar 4, beberapa baris kosong pada data masukan *testing* melalui proses penghapusan baris kosong. Kemudian, barisan data tanpa baris kosong dipisahkan untuk menjadi beberapa baris segmen.

2.6 Ekstraksi Fitur

Menurut Prihatini, ekstraksi fitur merupakan proses untuk mencari nilai - nilai fitur yang terkandung dalam dokumen [7]. Fitur dapat diartikan sebagai ciri dari setiap data yang dikenali oleh sistem sehingga menghasilkan nilai fitur. Ekstraksi fitur merupakan topik penting dalam klasifikasi, karena fitur-fitur yang baik akan sanggup meningkatkan tingkat akurasi, sementara fitur-fitur yang tidak baik cenderung memperburuk tingkat akurasi [8]. Berikut blok diagram ekstraksi fitur.



Gambar 5. Blok Diagram Ekstraksi Fitur

Pada penelitian ini, ekstraksi fitur digunakan untuk memberikan nilai/bobot pada setiap token berdasarkan fitur yang telah ditentukan. Ekstraksi fitur yang digunakan sebanyak 15 fitur. Untuk 13 fitur merujuk pada penelitian yang dilakukan oleh Peng [9], sedangkan untuk fitur LOWERCASE, dan EIGHTDIGIT merupakan fitur yang dibuat oleh penulis. Berikut penjelasan dari kelima belas fitur yang digunakan pada penelitian ini dapat dilihat pada Tabel 2.

Tabel 2. Ekstraksi Fitur

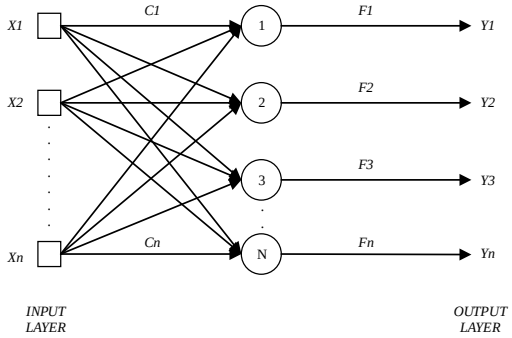
No	Ekstraksi Fitur	Keterangan
1	INITCAPS	Mengenali setiap token yang hurufnya diawali dengan kapital.
2	ALLCAPS	Mengenali setiap token yang semua hurufnya kapital.
3	CONTAINSDIGIT	Mengenali setiap token yang mengandung digit.
4	ALLDIGIT	Mengenali setiap token yang semuanya digit.
5	CONTAINSDOTS	Mengenali setiap token yang mengandung titik.
6	LOWERCASE	Mengenali setiap token yang semuanya huruf kecil.
7	PUNCTUATION	Mengenali setiap token yang mengandung tanda tertentu seperti titik, koma, titik dua, titik koma, tanda kurung, dan tanda seru.

8	EIGHTDIGIT	Fitur tambahan pada penelitian ini, fitur ini dikhususkan untuk mengenali token yang memiliki digit dengan panjang 8 digit.
9	WORD	Fitur tambahan pada penelitian ini, fitur ini dikhususkan untuk memberikan bobot pada token untuk kelas JENIS_PENELITIAN dan KALIMAT_PENGAJUAN.
10	LINE_START	Mengenali posisi token pada indeks array awal.
11	LINE_IN	Mengenali posisi token pada indeks array tengah.
12	LINE_END	Mengenali posisi token pada indeks array akhir.
13	PERSON	Mengenali token nama seseorang.
14	ORGANIZATION	Mengenali token sebuah organisasi.
15	YEAR	Mengenali ciri token tahun.

Setiap fitur pada Tabel 2.2 akan memberikan bobot 1 atau 0. Bobot 1 diberikan oleh setiap fitur jika token termasuk pada fitur. Begitupula dengan token yang tidak termasuk pada fitur, akan diberikan bobot 0.

2.4 *Learning Vector Quantization*

Learning Vector Quantization (*LVQ*) merupakan salah satu bagian algoritma dari Jaringan Syaraf Tiruan (JST), dan biasa disebut dengan Jaringan *LVQ*. Jaringan *LVQ* dibuat oleh Teuvo Kohonen untuk versi *supervised learning* dari algoritma *Self-Organizing Maps* (*SOM*) [10]. Supervised atau active learning adalah proses belajar yang membutuhkan guru, sesuatu yang memiliki pengetahuan tentang lingkungan [16]. Jaringan *LVQ* termasuk pada lapisan yang kompetitif, terdiri dari dua layer dan bisa bekerja untuk melakukan klasifikasi multi kelas [11], yang berarti jika 2 vektor input mendekati sama, maka lapisan kompetitif akan meletakkan kedua vektor input tersebut ke dalam kelas yang sama. Algoritma *LVQ* dapat disebut sebagai *Single Layer Feedforward* yang berarti jarngan syaraf dengan arsitektur lapis-tunggal umpan-maju [15]. Jaringan LVQ disebut sebagai Single Layer Feedforward karena "single-layer" yang dimaksud mengacu pada lapisan node output secara langsung [17]. Berikut struktur jaringan *LVQ*.



Gambar 6. Struktur Jaringan LVQ

Pada Gambar 1, prinsip kerja algoritma *LVQ* adalah melakukan pembelajaran secara otomatis untuk mengklasifikasikan vektor – vektor input kedalam kelas-kelas tertentu. berikut parameter dan langkah – langkah untuk melakukan *training* dan *testing LVQ* [12] [10].

Tabel 3. Algoritma Deskriptif *Training* dan *Testing LVQ*

Training LVQ
1. Inisialisasi *Codebook Vectors*, *Epoch*, *Max Epoch*, *Learning Rate* awal, *Learning Rate* selanjutnya, *Learning Rate* Baru, Pengurangan *Learning Rate*, dan Error minimum yang diharapkan. 2. Tetapkan kondisi awal untuk $Epoch = 0$. 3. Kerjakan $Epoch$ sampai $MaxEpoch$ jika ($Epoch < MaxEpoch$) atau ($\alpha > Eps$). 4. Tentukan nilai jarak minimum dengan *Euclidean Distance* [11] : $$d_{r,b} = \sqrt{\sum_{j=1}^{J} \left(x_{r,j} - c_{b,j}\right)^2} \quad (1)$$ 5. Perbaharui c_w dengan ketentuan [11] : - Jika $kelas\ x_r\ sama\ dengan\ kelas\ c_w$ maka : $c_w = c_w + \alpha(t) \cdot (x_r - c_w)$ (2) atau bisa dibaca dengan [4] $c_{b,j(new)} = c_{b,j(old)} + \alpha(t) \cdot \left(x_{r,j} - c_{b,j(old)}\right)$ (3) - Jika $kelas\ x_r tidak\ sama\ dengan\ kelas\ c_w$ maka: $c_w = c_w - \alpha(t) \cdot (x_{rj} - c_w)$ (4) atau bisa dibaca dengan [4] $c_{b,j(new)} = c_{b,j(old)} - \alpha(t) \cdot (x_{rj} - c_{b,j(old)})$ (5)

Untuk semua *codebook vectors* lainnya tidak berubah :

$c_b = c_b \; untuk \; b \; tidak \; sama \; dengan \; w$ (6)

6. Pengurangan *Learning Rate* (α) , dengan rumus [13] :

$$\alpha(t+1) = \alpha(t) - \frac{\alpha(0)-u}{K} \quad (7)$$

7. Penambahan *epoch*, dengan rumus [12] :

$$Epoch = Epoch + 1 \quad (8)$$

Testing LVQ

1. Inisialisasi *Sample Vector* sebagai $x_{r,j}$.
2. Persiapkan bobot akhir yang telah diperoleh pada tahapan *training* sebagai $c_{b,j}$.
3. Tentukan nilai jarak minimum dengan *Euclidean Distance* [10] :

$$d_{r,b} = \sqrt{\sum_{j=1}^{J} \left(x_{r,j} - c_{b,j}\right)^2} \quad (9)$$

4. Klasifikasikan nilai $d_{r,b}$ sebagai kelas dari $x_{r,j}$.

Pada rumus nomor ke (1) dan (9), J merupakan total neuron ke-j, $x_{r,j}$ merupakan *sample vector* x_r, $c_{b,j}$ merupakan *codebook vectors* c_b, dan $d_{r,b}$ merupakan nilai jarak minimum antara $x_{r,j}$ dengan $c_{b,j}$. Sedangkan, pada rumus nomor ke (7), t berupa waktu (Menunjukan perubahan bobot pada satu waktu sesuai dengan banyaknya *input* yang masuk [14] atau bisa dilihat sebagai iterasi [13]), $\alpha(0)$ merupakan *learning rate* awal yang ditentukan secara konstan, $\alpha(t+1)$ merupakan *learning rate* selanjutnya dari waktu sebelumnya, $\alpha(t)$ merupakan *learning rate* baru untuk waktu tertentu, u merupakan nilai pengurangan *learning rate*, dan K merupakan jumlah kelas.

Perlu diketahui bahwa jika terdapat nilai minimum yang sama pada masing – masing bobotnya, maka untuk nilai minimum pertama kali yang diperoleh itulah yang menang. Mengingat bahwa pada algoritma *LVQ* ini mempunyai konsep competitive learning [27]. Proses *training LVQ* membutuhkan *codebook vectors* dan parameter – parameternya, yaitu *epoch*, *maxepoch*, learningrate, dan lain-lain. Berikut beberapa parameter yang digunakan pada algoritma *LVQ* dapat dilihat pada Tabel 4.

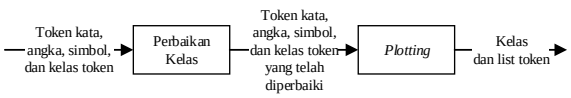
Tabel 4. Parameter Algoritma *LVQ*

No	Parameter	Simbol
1	*Codebook vectors*	C
2	*Epoch*	$epoch$
3	*Maxepoch*	$maxepoch$
4	*Learning rate* pada *epoch* 0	$\alpha(0)$
5	*Learning rate* selanjutnya	$\alpha(t+1)$
6	*Learning rate* pada waktu tertentu	$\alpha(t)$
7	Jumlah kelas	K
8	Pengurangan *Learningrate*	$Pengurangan \; LearningRate$
9	*Eps*	Eps

Codebook vectors dapat dikatakan sebagai bobot awal [12] atau *reference vectors* [13]. *Codebook vectors* mewakili masing – masing kelas yang dipilih menjadi acuan untuk proses *training*.

2.5 *Plotting* Per Kelas

Setiap token yang telah memiliki masing-masing kelas akan dilakukan *plotting* per kelas. *Plotting* per kelas dilakukan untuk menghasilkan serangkaian data berdasarkan kelas token, seperti kategori judul penelitian (sampul) yang direpresentasikan sebagai kelas 0. Sebelum melakukan *plotting*, sistem melalui proses untuk memperbaiki kelas token yang tidak sesuai dengan kelas aslinya. Berikut blok diagram *plotting* per kelas.



Gambar 7. Blok Diagram Plotting Per Kelas

Pada Gambar 7, token kata, angka, simbol akan diperbaiki kelasnya. Perbaikan kelas dilakukan berdasarkan hasil klasifikasi dari masing-masing token yang tidak sesuai dengan kelas aslinya.

2.5.1 Perbaikan Kelas

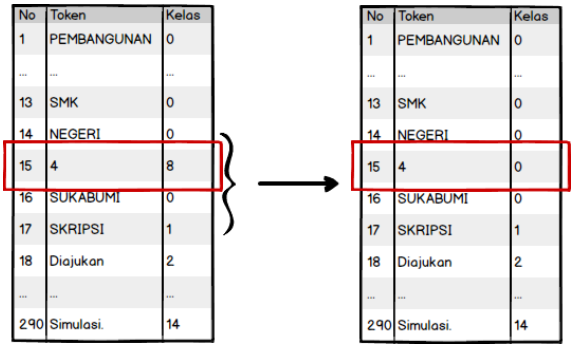
Perbaikan kelas merupakan suatu proses untuk memperbaiki ketidaksesuaian kelas token hasil klasifikasi algoritma *LVQ* dengan kelas aslinya. Perbaikan dilakukan untuk memperoleh akurasi yang lebih baik terhadap proses *plotting*. Aturan yang diterapkan untuk memperbaiki kelas token dapat dilihat pada Tabel 5.

Tabel 5 Aturan Perbaikan Kelas

No	Kondisi	Aksi
1	$(T_i \neq T_{i-1})$ AND $(T_i = T_{i+1})$ AND $(T_i > T_{i-1})$	Mengecek kelas token berdasarkan urutannya.
2	$(T_i \neq T_{i-1})$ AND $(T_i \neq T_{i+1})$ AND $(T_i < T_{i-1})$	Mengecek kelas token berdasarkan mayoritas kelas token.

3	$(T_i \neq T_{i-1})$ AND $(T_i \neq T_{i+1})$ AND $(T_i > T_{i-1})$ AND $(T_i \neq T_{i-1+1})$	Mengecek kelas token berdasarkan urutannya.
4	$(T_i \neq T_{i-1})$ AND $(T_i \neq T_{i+1})$ AND $(T_i > T_{i-1})$ AND $(T_i < T_{i+1})$	Mengecek kelas token berdasarkan kelas token yang dikhususkan untuk 1 token tertentu.
5	$(T_i \neq T_{i-1})$ AND $(T_i \neq T_{i+1})$ AND $(T_i < T_{i-1})$ AND $(T_i \neq T_{i-1+1})$	Mengecek kelas token berdasarkan urutannya.
6	$(T_i = T_{i-1})$ AND $(T_i = T_{i+1})$	Mengecek kelas token berdasarkan urutannya.
7	$(T_i = T_{i-1})$ AND $(T_i \neq T_{i+1})$	Mengecek kelas token berdasarkan mayoritas kelas token.

Pada Tabel 5, simbol T menunjukan *variable* kelas ke-i dengan beberapa variasi parameternya, seperti T_{i-1}, T_{i+1}, dan T_{i-1+1}. Berikut salah satu contoh perbaikan kelas untuk aturan nomor 3.



No	Token	Kelas
1	PEMBANGUNAN	0
...	...	...
13	SMK	0
14	NEGERI	0
15	4	8
16	SUKABUMI	0
17	SKRIPSI	1
18	Diajukan	2
...	...	...
290	Simulasi.	14

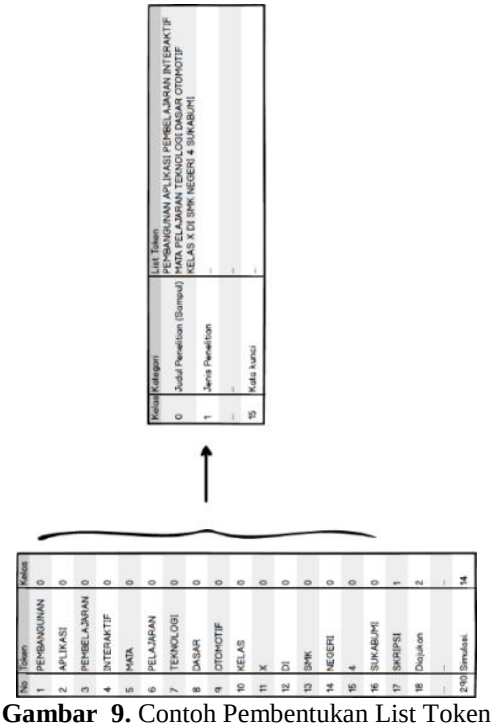
No	Token	Kelas
1	PEMBANGUNAN	0
...	...	...
13	SMK	0
14	NEGERI	0
15	4	0
16	SUKABUMI	0
17	SKRIPSI	1
18	Diajukan	2
...	...	...
290	Simulasi.	14

Gambar 8. Contoh Perbaikan Kelas

Pada Gambar 8, token angka '4' memiliki kelas 8 yang tidak sesuai dengan kelas aslinya. Kelas asli dari token angka '4' adalah kelas ke-0.

2.5.2 *Plotting*

Plotting merupakan proses pembentukan list token berdasarkan kelas token. Setiap token yang memiliki kelas token hasil perbaikan kelas akan digabungkan berdasarkan kelas token. Berikut contoh pembentukan list token.



Kelas	Kategori	List Token
0	Judul Penelitian (Sumpul)	PEMBANGUNAN APLIKASI PEMBELAJARAN INTERAKTIF MATA PELAJARAN TEKNOLOGI DASAR OTOMOTIF KELAS X DI SMK NEGERI 4 SUKABUMI
1	Jenis Penelitian	...
...	...	...
15	Kata kunci	...

No	Token	Kelas
1	PEMBANGUNAN	0
2	APLIKASI	0
3	PEMBELAJARAN	0
4	INTERAKTIF	0
5	MATA	0
6	PELAJARAN	0
7	TEKNOLOGI	0
8	DASAR	0
9	OTOMOTIF	0
10	KELAS	0
11	X	0
12	DI	0
13	SMK	0
14	NEGERI	0
15	4	0
16	SUKABUMI	0
17	SKRIPSI	1
18	Diajukan	2
...	...	...
290	Simulasi.	14

Gambar 9. Contoh Pembentukan List Token

Pada Gambar 9, sistem akan mendeteksi setiap kelas token yang memiliki kesamaan kelas token 0, lalu menggabungkannya berdasarkan kesamaan kelas tersebut. Kelas token 0 merupakan kelas token yang sudah melalui proses perbaikan kelas seperti pada Gambar 8. Setiap token yang memiliki kelas 0 digabungkan, sehingga membentuk list token dengan kategori Judul Penelitian (Sampul).

2.7 Hasil Pengujian

Pengujian dilakukan dengan bentuk pengujian token-kelas dan kelas-token. Keseluruhan dokumen yang telah memiliki nilai akurasi akan dihitung rata-rata nilai akurasi dan errornya. Berikut hasil pengujian nilai akurasi dan error.

2.7.1 Pengujian Token-Kelas

Setiap token kata, angka, dan simbol yang telah memiliki kelas divalidasi kebenarannya, lalu dihitung tingkat kebenaran dan kesalahannya, sehingga menghasilkan nilai akurasi dan error. Berikut hasil klasifikasi dengan bentuk pengujian token-kelas dapat dilihat pada Tabel 6.

Tabel 6. Hasil Bentuk Pengujian Token-Kelas

No	Nama Dokumen	Akurasi	Error
1	1.txt	80%	20%
2	2.txt	87%	13%
3	3.txt	76%	24%
4	4.txt	82%	18%
5	5.txt	82%	18%
6	6.txt	75%	25%
7	7.txt	76%	24%
8	8.txt	72%	28%

9	9.txt	71%	29%
10	10.txt	70%	30%
11	11.txt	71%	29%
12	12.txt	71%	29%
13	13.txt	77%	23%
14	14.txt	77%	23%
15	15.txt	81%	19%
16	16.txt	82%	18%
17	17.txt	86%	14%
18	18.txt	84%	16%
19	19.txt	81%	19%
20	20.txt	83%	17%
21	21.txt	81%	19%
22	22.txt	79%	21%
23	23.txt	80%	20%
24	24.txt	78%	22%
25	25.txt	82%	18%
26	26.txt	78%	22%
27	27.txt	81%	19%
28	28.txt	86%	14%
29	29.txt	76%	24%
30	30.txt	81%	19%
31	31.txt	78%	22%
32	32.txt	73%	27%
33	33.txt	71%	29%
34	34.txt	67%	33%
35	35.txt	69%	31%
36	36.txt	83%	17%
37	37.txt	81%	19%
38	38.txt	77%	23%
39	39.txt	81%	19%
40	40.txt	82%	18%
Rata-Rata		78%	22%

2.7.2 Pengujian Kelas-Token

Pengujian yang dilakukan dengan bentuk pengujian kelas-token berdasarkan hasil *plotting* per kelas yang sudah dilakukan sebelumnya. Beberapa list token berdasarkan kelasnya terdiri dari token-token yang sudah melalui proses perbaikan kelas. Berikut hasil klasifikasi dengan bentuk pengujian token-kelas dapat dilihat pada Tabel 7.

Tabel 7. Hasil Bentuk Pengujian Kelas-Token

No	Nama Dokumen	Akurasi	Error
1	1.txt	0%	100%
2	2.txt	13%	88%
3	3.txt	0%	100%
4	4.txt	0%	100%
5	5.txt	13%	88%
6	6.txt	6%	94%
7	7.txt	19%	81%
8	8.txt	0%	100%
9	9.txt	19%	81%
10	10.txt	6%	94%
11	11.txt	6%	94%
12	12.txt	0%	100%
13	13.txt	0%	100%
14	14.txt	6%	94%
15	15.txt	6%	94%
16	16.txt	13%	88%
17	17.txt	0%	100%
18	18.txt	6%	94%
19	19.txt	6%	94%
20	20.txt	13%	88%
21	21.txt	0%	100%
22	22.txt	0%	100%
23	23.txt	0%	100%
24	24.txt	0%	100%
25	25.txt	6%	94%
26	26.txt	0%	100%
27	27.txt	0%	100%
28	28.txt	6%	94%
29	29.txt	13%	88%
30	30.txt	13%	88%
31	31.txt	6%	94%
32	32.txt	6%	94%
33	33.txt	6%	94%
34	34.txt	6%	94%
35	35.txt	6%	94%
36	36.txt	0%	100%
37	37.txt	13%	88%
38	38.txt	13%	88%
39	39.txt	13%	88%
40	40.txt	13%	88%
Rata-Rata		6%	94%

2.8 Analisis Hasil Pengujian

Tinggi rendahnya akurasi dan error yang diperoleh memiliki beberapa penyebab. Analisis yang dilakukan terhadap hasil pengujian dengan bentuk pengujian token-kelas menyimpulkan bahwa yang pertama adalah karena data masukan yang digunakan oleh sistem adalah file teks, maka terdapat keterbatasan pada fitur yang digunakan. Kedua, nilai akurasi yang rendah diperoleh karena pada dokumen testing terdapat beberapa simbol tidak beraturan yang dihasilkan oleh proses konversi sebelumnya. Ketiga, berdasarkan pengamatan yang dilakukan, dampak yang mempengaruhi pada penurunan nilai akurasi selalu disebabkan oleh fitur *ORGANIZATION* yang dikhususkan untuk memberikan pembobotan pada kategori Program Studi, Fakultas, dan Universitas. Karena, pada algoritma *LVQ* memiliki jaringan kompetitif, maka jika Program Studi diasumsikan menang dengan pembobotan yang sama antara Fakultas dan Universitas, Fakultas dan Universitas akan terklasifikasikan pada kelas yang sama seperti Program Studi. Keempat, minimnya penggunaan ekstraksi fitur yang dapat mengenali beberapa token secara spesifik terhadap banyaknya jumlah kelas, yaitu 16 kelas, sehingga untuk memberikan bobot pada setiap token tidak signifikan.

Sedangkan, berdasarkan pengamatan, nilai akurasi yang diperoleh tidak pernah melebihi 50%,

karena yang pertama adalah adanya kesalahan ejaan/typo pada token kata, seperti Teknologi Dasar menjadi Teknologi Dasr. Kedua, munculnya tanda spasi untuk token kata maupun token angka, seperti 10111654 menjadi 1011(spasi)1654. Ketiga, munculnya simbol tidak beraturan yang disebabkan oleh proses konversi dokumen, sehingga simbol tidak beraturan menjadi penghalang/noise terhadap suatu data, seperti ___… WW. _ _ N...-yy n-r.

3 PENUTUP

Berdasarkan pengujian fungsionalitas sistem dan pengukuran akurasi yang telah dilakukan, sistem ekstraksi informasi menggunakan algoritma *Learning Vector Quantization* (*LVQ*) telah berhasil dibangun dengan perolehan akurasi token-kelas sebesar 78% dan akurasi kelas-token sebesar 6%. Perolehan akurasi token-kelas disebabkan oleh penggunaan fitur pembobotan yang tidak signifikan, sehingga algoritma *LVQ* tidak dapat melakukan klasifikasi secara benar. Sedangkan, perolehan akurasi kelas-token disebabkan oleh pengaruh klasifikasi algoritma *LVQ* token-kelas, adanya salah ejaan/typo, dan munculnya simbol tidak beraturan hasil konversi dari *file PDF*.

Saran untuk pengembangan selanjutnya menggunakan data masukan dengan format .docx atau .html agar dapat menggunakan beberapa fitur tambahan seperti mendeteksi bold, italic, underline, serta beberapa *font style* lainnya, diperlukan suatu fitur tambahan yang signifikan untuk pembobotan kategori Fakultas, Program Studi, Universitas, dan dibutuhkan pengecekan dan pengkoreksian kesalahan ejaan atau typo pada hasil konversi *file PDF*.

DAFTAR PUSTAKA

[1] D. Jurafsky dan J. H. Martin, "*Chapter* 21 - *Information Extraction*," dalam *Speech and Language Processing*, 2017, pp. 1 - 31.

[2] D. Mustaqwa, "Implementasi Ekstraksi Informasi Pada Dokumen Teks Skripsi Menggunakan Metode Rule Based," Skripsi, Universitas Komputer Indonesia, Bandung, Jawa Barat, 2018.

[3] A. I. Riaddy, Y. S. S.Si, M.T. dan A. A. S.Si, M.Si, "Ekstraksi Informasi pada Makalah Ilmiah dengan Pendekatan *Supervised Learning*," *e-Proceeding of Engineering*, vol. 3, no. No. 1, pp. 1184 - 1190, 2016.

[4] M. T. Pilevar, H. F. dan M. S. , "*Classification of Persian textual documents using Learning Vector Quantization*".

[5] N. dan Z. Abdul-Malek, "*Neural Networks for Fault Classification: Comparison between Feed-Forward Back-Propagation, RBF, and LVQ Neural Network*," *Applied Mechanics and Materials*, vol. 818, pp. 96 - 100, 2016.

[6] M. F. Umer dan M. S. H. Khiyal, "*Classification of Textual Documents Using Learning Vector Quantization*," *Information Technology Journal*, vol. 6, no. 1, pp. 154-159, 2007.

[7] P. M. Prihatini, "Implementasi Ekstraksi Fitur Pada Pengolahan Dokumen Berbahasa Indonesia," Jurnal Matrix, vol. 6, no. 3, pp. 174-178, 2016.

[8] M. Indrawijaya, L. dan R. Adipranata, "Aplikasi Ekstraksi Fitur Citra Hufur Jawa Berdasarkan Morfologinya".

[9] F. Peng dan A. MCCallum, "*Accurate Information Extraction from Research Papers using Conditional Random Fields*," pp. 1-8, 2004.

[10] I. N. D. Silva, D. H. Spatti, R. A. Flauzino, L. H. B. Liboni dan S. F. d. Reis Alves, "*Artificial Neural Networks A Practical Course*," dalam 9.3 *LVQ Networks (Learning Vector Quantization)*, Brazil, *Springer International Publishing Switzerland*, 2017, pp. 176 - 181.

[11] G. R. Lloyd, R. G. Brereton, R. Faria dan J. C. Duncan, "*Learning Vector Quantization for Multiclass Classification: Application to Characterization of Plastics*," J. Chem. Inf. Model., vol. 47, pp. 1553-1563, 2007.

[12] E. Sabrina dan I. G. P. A. Buditjahjanto, "Klasifikasi Penyakit Diabetic Retinopathy menggunakan Metode *Learning Vector Quantization* (LVQ)," Jurnal Teknik Elektro, vol. 6, no. Nomor 02, pp. 97 - 104, 2017.

[13] M. T. Martin-Valdivia, M. Garcia-Vega dan L. A. Urena-Lopez, "*LVQ for text categorization using a multilingual linguistic resource*," *Neurocomputing*, vol. 55, pp. 665 - 679, 2002.

[14] F. R. Hariri, E. Utami dan A. Amborowati, "*Learning Vector Quantization* untuk Klasifikasi Abstrak Tesis," Citec Journal, vol. 2, no. No. 2, pp. 128 - 143, 2015.

[15] M. W. Dessy and I. Afrianto, "Perbandingan Metode Jaringan Syaraf Tiruan *Backpropagation* dan *Learning Vector Quantization* Pada Pengenalan Wajah," Jurnal Komputer dan Informatika (KOMPUTA), pp. 45-51, 2012.

[16] Suyanto, "*Artificial Intelligence*," in *Searching, Reasoning, Planning*, dan *Learning* Revisi Kedua, Bandung, Informatika, 2014, p. 179.

[17] S. Haykin, "*Neural Networks and Learning Machines Third Edition*," *in Neural Networks and Learning Machines*, New Jersey, Pearson Education, 2008, p. 21.