BAB 2
LANDASAN TEORI

2.1 *Text Mining*

Text mining merupakan sebuah gambaran dari teknologi yang dapat menganalisis dan memproses data teks bersifat struktur atau tidak terstruktur [11]. Dengan hal inilah yang membedakan text mining dan data mining karena data mining mengelolah data yang bersifat terstruktur. Proses *text mining* pada dasarnya merupakan pengembangan dari penelitian data mining akan tetapi yang menjadi perbedaan adalah alur yang digunakan *text mining* yang diambil dari sekumpulan teks tidak struktur [7]. Pengunaan text mining pada umumnya digunakan dale3eram kasus peringkasan teks otomatis, *information retrieval*, *information extraction*, *clustering*, klasifikasi dan sebagainya. Secara umum terdapat tahapan dalam text mining adalah *text preprocessing* yang merupakan tahap awal dengan merangkum rutinitas dan proses data yang akan digunakan pada *knowledge discovery* sistem *text mining* dan *feature selection* yang merupakan tahapan dengan bertujuan mengurangi kata – kata yang tidak diperlukan atau tidak penting maka pengklasifikasian lebih akurat dan efektif [8].

2.2 Peringkasan Teks Otomatis

Peringkasan Teks Otomatis adalah pembuatan ringkasan yang dilakukan menggunakan komputer untuk memilih informasi yang penting pada dokumen aslinya dari satu atau banyak dokumen [1]. Ringkasan yang baik harus dapat mudah dipahami, lebih singkat dan terdiri dari gagasan utama. Umumnya ada dua peringkasan teks yang dilakukan dan dikenal diantaranya *single document summarization* dan *multi document summarization*. Pendekatan dalam peringkasan teks yang dihasilkan menggunakan dua teknik diantaranya sebagai berikut.

1. Teknik *Ekstraksi*

Teknik *ekstraksi* adalah peringkasan teks dengan pengurutan kalimat yang akan disalin, terdiri dari bagian kalimat penting atau gagasan utama dari dokumen teks aslinya [9].

2. Teknik *Abstraksi*

Teknik *abstraksi* adalah peringkasan teks dengan mengambil gagasan atau informasi yang penting dari dokumen teks asli dan kemudian menyalin ulang dengan kalimat baru pada ringkasan teks yang bukan dari dokumen aslinya [9].

2.3 *Preprocessing*

Preprocessing merupakan tahapan masukan untuk mengelolah teks dalam dokumen yang merupakan bahan mentah dalam proses pada langkah selanjutnya setelah input data masukan. Pada tahap ini proses yang dilakukan antara lain *tokenizing* kalimat, *casefolding, filtering, tokenizing* kata dan *stopword removal*. Alur dari tahapan *preprocessing* dapat dilihat pada gambar 2.1 sebagai berikut.



Gambar 2. 1 Tahapan *Preprocessing*

2.3.1 *Tokenizing* Kalimat

Tokenizing kalimat merupakan proses memecah kalimat dalam teks dalam dokumen yang panjang menjadi sekumpulan kalimat-kalimat [10]. Pada proses yang digunakan dalam pemisahan kalimat dengan tanda titik (.), tanda tanya (?), dan tanda seru (!). Contoh hasil *tokenizing* kalimat dapat dilihat pada tabel 2.1 sebagai berikut.

Tabel 2. 1 Contoh Hasil *Tokenizing* Kalimat

Teks	Hasil *Tokenizing* Kalimat
Teknologi Informasi memberikan harapan baru bagi manusia. Teknologi Informasi menawarkan pemecahan yang cepat, akurat dan konsisten	Teknologi Informasi memberikan harapan baru bagi manusia.
	Teknologi Informasi menawarkan pemecahan yang cepat, akurat dan konsisten

Pada tabel 2.1 hasil *tokenizing* kalimat teks awal yang tersusun beberapa kalimat yang tersusun dalam paragraf, terpecah atau terpotong berdasarkan delimiter tanda titik (.) menjadi sekumpulan kalimat – kalimat.

2.3.2 *Case Folding*

Case folding adalah proses mengubah teks menjadi karakter dengan karakter kecil dan hanya karakter huruf ‘a’ sampai ‘z’ yang akan diterima [11]. Contoh *case folding* dapat dilihat pada tabel 2.2 sebagai berikut.

Tabel 2. 2 Contoh Hasil *Case Folding*

Kalimat	Hasil *Case folding*
Teknologi **I**nformasi memberikan harapan baru bagi manusia.	teknologi informasi memberikan harapan baru bagi manusia
Teknologi **I**nformasi menawarkan pemecahan yang cepat, akurat dan konsisten	teknologi informasi menawarkan pemecahan yang cepat, akurat dan konsisten

Pada tabel 2.2 hasil *case folding* yang awalnya kalimat terdiri dari karakter besar dan kecil menjadi kalimat dengan karakter kecil.

2.3.3 *Filtering*

Filtering adalah proses menghilangkan karakter selain huruf yang dianggap sebagai *delimiter*. List karakter delimiter yang dihapus dapat dilihat pada tabel 2.3 sebagai berikut.

Tabel 2. 3 Karakter *Delimiter* Yang Dihapus

Karakter					
!	&	„	;	]	,
@	*	“	/	|	`
#	(	-	?	\	%
$	)	<	{	+	^
_	.	>	}	[	:
=	,				

Adapun contoh hasil *filtering* dapat dilihat pada tabel 2.4 sebagai berikut.

Tabel 2. 4 Contoh Hasil *Filtering*

Kalimat	Hasil *Filtering*
teknologi informasi memberikan harapan baru bagi manusia	teknologi informasi memberikan harapan baru bagi manusia
teknologi informasi menawarkan pemecahan yang cepat**,** akurat dan konsisten	teknologi informasi menawarkan pemecahan yang cepat akurat dan konsisten

Pada tabel 2.4 hasil *filtering* terdapat tanda koma (,) pada kalimat kedua yang dimana tanda tersebut termasuk karakter delimiter yang dihapus maka hasil *filtering* koma (,) tersebut dihapus.

2.3.4 *Tokenizing* kata

Tokenizing kata adalah proses pemecahan kalimat menjadi kata-kata tunggal. Proses pemecahan kalimat menjadi kata dilakukan berdasarkan *delimiter* yang menyusunn yaitu *white space* (” ”). Proses ini bertujuan agar dapat melakukan

proses *stopword removal*. Contoh *tokenizing* kata dapat dilihat pada tabel 2.5 sebagai berikut.

Tabel 2. 5 Contoh Hasil *Tokenizing* Kata

Kalimat	Hasil *Tokenizing kata*	
teknologi informasi memberikan harapan baru bagi manusia	teknologi	baru
	informasi	bagi
	memberikan	manusia
	harapan	
teknologi informasi menawarkan pemecahan yang cepat akurat dan konsisten	teknologi	cepat
	informasi	akurat
	menawarkan	dan
	pemecahan	konsisten
	yang	

Pada tabel 2.5 hasil *tokenizing* kata kalimat yang tersusun dari berbagai kata terpecah atau terpotong berdasarkan *white space* (" ").

2.3.5 *Stopword Removal*

Stopword Removal merupakan tahapan penghilangan kata yang tidak memiliki makna atau kurang berarti dan sering muncul dalam sekumpulan kata-kata [11]. *Stopword* dapat berupa kata penghubung, kata ganti, preposisi, dan lain - lain. Pada penelitian ini *stopword* yang digunakan adalah kamus *stopword* yang diambil dari rekomendasi Fadillah Z. Tala [12]. Contoh dari *stopword removal* dapat dilihat pada table 2.6 sebagai berikut.

Tabel 2. 6 Contoh Hasil *Stopword Removal*

Kata inputan		*Stopword Removal*	
teknologi	informasi	teknologi	cepat
informasi	menawarkan	informasi	akurat
memberikan	pemecahan	harapan	konsisten
harapan	**yang**	manusia	
baru	cepat	teknologi	
bagi	akurat	informasi	
manusia	**dan**	menawarkan	
teknologi	konsisten	pemecahan	

Pada tabel 2.6 hasil *stopword removal* kata yang dicetak tebal dan digaris bawah termasuk kata yang terdapat pada kamus *stopword* maka kata tersebut dihilangkan.

2.4 Ekstraksi Fitur

Ekstraksi fitur merupakan proses pengolahan kalimat yang menghasilkan nilai-nilai bobot. Nilai bobot hasil ekstraksi fitur merukapan masukan yang digunakan

input layer pada *neural network*. Ekstraksi fitur yang digunakan adalah posisi kalimat, *positive keyword*, kemiripan antar kalimat, panjang kalimat dan bobot koneksi antar kalimat.

2.4.1 Posisi Kalimat

Posisi kalimat adalah letak kalimat dari kalimat pada dokumen. Pada penelitian ini diasumsikan bahwa kalimat pertama adalah kalimat terpenting. Berikut adalah rumus yang digunaka perhitungan posisi kalimat [13] :

$$\text{Score}_{f1}(s_i) = \frac{X}{N} \tag{2.1}$$

Asumsikan $\text{Score}_{f1}(s_i)$ adalah nilai bobot posisi kalimat dari kalimat dalam dokumen, N adalah total jumlah kalimat dalam semua kalimat dan X adalah nilai posisi kalimat dari N yang merupakan banyaknya kalimat pada satu dokumen dan untuk nilai selanjutnya dikurang satu dari total N hingga nilai satu. Berikut ini contoh dari penggunaan fitur ekstraksi posisi kalimat:

Kantor berita Amaq yang mendukung ISIS menyebutkan, seorang pengebom bunuh diri ISIS telah menargetkan para milisi Syiah. Sebelumnya, ISIS juga kerap melakukan serangan bom di Baghdad dan wilayah-wilayah Irak lainnya, khususnya menargetkan warga Syiah. Pada Februari lalu, kelompok ISIS mengklaim dua bom bunuh diri di Kota Sadr yang menewaskan 70 orang.

Kalimat diatas menunjukkan terdiri dari tiga (N) untuk jumlah kalimat dan untuk nilai (X) kalimat pertama adalah tiga dan nilai kalimat berikutnya dikurang satu (-1). Nilai bobot posisi kalimatnya adalah 3/3 = 1 untuk kalimat pertama, 3/2 = 0.6 untuk kalimat kedua dan 1/3 = 0.3 untuk kalimat ketiga.

2.4.2 *Positive Keyword*

Positive keyword merupakan kata yang sering muncul pada satu kalimat dengan kalimat yang lainya kalimat [2]. Mendapatkan kemunculan kata digunakan perhitungan *term frequency* dalam satu kalimat dan kalimat lainnya di satu dokumen.

$$\text{Score}_{f2}(s_i) = \frac{S_i(positive\ keyword)}{\sum_{i=0}^{n} S_i\ (positive\ keword)} \tag{2.2}$$

Asumsikan $\text{Score}_{f2}\,(s_i)$ adalah nilai bobot *positive keyword* dari kalimat, $S_i(positif\ keyword)$ adalah jumlah kata dalam suatu kalimat yang mengandung keyword dibagi dengan $\sum_{i=0}^{n} S_i\,(positive\ keword)$ jumlah kata dalam seluruh kalimat yang mengandung keyword, dengan keyword merupakan banyaknya kata yang muncul dalam suatu dokumen. Berikut ini contoh dari penggunaan fitur ekstraksi positive *keyword* kalimat:

Kantor berita Amaq yang mendukung ISIS menyebutkan, seorang pengebom bunuh diri ISIS telah menargetkan para milisi Syiah. Sebelumnya, ISIS juga kerap melakukan serangan bom di Baghdad dan wilayah-wilayah Irak lainnya, khususnya menargetkan warga Syiah. Pada Februari lalu, kelompok ISIS mengklaim dua bom bunuh diri di Kota Sadr yang menewaskan 70 orang.

Hasil *Preprocessing*:

Kalimat ke 1: Kantor berita amaq mendukung **isis** pengebom **bunuh** **isis** menargetkan milisi syiah

Kalimat ke 2: **isis** kerap serangan **bom** baghdad wilayah wilayah irak menargetkan warga syiah

Kalimat ke 3: februari kelompok **isis** mengklaim **bom** **bunuh** kota sadr menewaskan 70 orang

Kata “isis” muncul 4 kali, “bom” muncul 2 kali, dan “bunuh” muncul 2 kali maka diketahui bahwa ada 3 *positive keyword*. Berdasarkan kata tersebut dihitung kembali banyaknya kata pada tiap kalimat dan dibagi dengan banyak kata yang mengandung positive keyword tersebut. Nilai bobot *positive keyword* untuk kalimat pertama memiliki 2 *positive keyword* yang muncul dalam 1 kalimat maka bobotnya 2/3 = 0.666, kalimat 2 memiliki 2 *positive keyword* maka bobotnya 2/3 = 0.666 dan kalimat 3 memiliki 3 *positive keyword* maka bobotnya 3/3 = 1. Kalimat yang tidak mengandung positive keyword maka bobotnya adalah 0.

2.4.3 Kemiripan Antar Kalimat

Kemiripan antar kalimat adalah kata yang muncul atau sama pada kalimat yang sama dengan kalimat lain dalam satu dokumen [2]. Mendapatkan bobot dari kemiripan antar kalimat diperoleh dari persamaan 2.3 sebagai berikut.

$$\text{Score}_{f3}\,(s_i) = \frac{\sum \text{Sim}\,(s_i, s_j)}{\max(\sum \text{Sim}\,(s_i, s_j))} \tag{2.3}$$

Asumsikan $\text{Score}_{f3}\,(s_i)$ adalah nilai bobot dari kemiripan kalimat dari kalimat ke-. Dimana $\sum \text{Sim}\,(s_i, s_j)$ merupakan kemiripan antara kalimat ke-i dan kalimat ke-j dibagi dengan $\max(\sum \text{Sim}\,(s_i, s_j))$ merupakan jumlah kemiripan antara kalimat ke-i dan kalimat ke-j. Mendapatkan s_i, s_j dapat diperoleh dari persamaan 2.4 sebagai berikut.

$$\text{Sim}\,(s_i, s_j) = \frac{\sum_{i=1}^{n} W_{it} * W_{ij}}{\sqrt{\sum_{i=1}^{n} {W_i}^2} * \sqrt{\sum_{i=1}^{n} {W_j}^2}} \tag{2.4}$$

Asumsikan $\text{Sim}\,(s_i, s_j)$ adalah nilai bobot jarak anatar kalimat ke-i dan kalimat ke-j yang dimana $\sum_{i=1}^{n} W_{it} * W_{ij}$ merupakan jumlah perhitungan *scalar* dari pembobotan kalimat dibagi dengan $\sqrt{\sum_{i=1}^{n} {W_i}^2} * \sqrt{\sum_{i=1}^{n} {W_j}^2}$ merupakan jumlah perhitungan *vector* dari pembobotan kalimat.

Perhitungan bobot kalimat yang dilakukan berdasarkan banyaknya kemunculan dari suatu kata dalam sebuah dokumen dan jumlah kemunculan pada kalimat lain dalam satu dokumen yang diperoleh dari persamaan 2.5 sebagai berikut.

$$IDF = \log \frac{tf}{d_f} \tag{2.5}$$

Asumsikan N adalah banyaknya kalimat dokumen dan d_f adalah total kemunculan kata pada kalimat dan untuk menghitung bobot kalimat (w) diperoleh dari persamaan 2.6 sebagai berikut.

$$w_i = tf_{dt} * IDF_t \tag{2.6}$$

Asumsikan w_i adalah jumalah bobot kata pada kalimat d ke-d terhadap kata t ke-t dari perhitungan kemunculan *term frekuensi* dan *inverse document frequency* [11].

Berikut ini contoh dari penggunaan fitur ekstraksi kemiripan antar kalimat:

Kalimat 1 : Kami pergi makan dan rekreasi.

Kalimat 2 : Kami pergi lapangan.

Kalimat 3 : Kami sedang memancing.

Kata “Kami” terjadi kemunculan pada tiga kalimat, kata “pergi” terjadi kemunculan pada dua kalimat di kalimat satu dan dua. Jarak antar kalimat dihitung untuk mendapatkan nilai bobot jarak antar kalimat yang nantinya bobot tersebut sebagai bobot kemiripan antar kalimat.

2.4.4 Panjang Kalimat

Panjang kalimat merupakan proses untuk mengetahui kesamaan jumlah kata yang terjadi dalam kalimat dalam kalimat panjang dalam dokumen [2]. Mendapatkan bobot dari panjang kalimat diperoleh dari persamaan 2.7 sebagai berikut.

$$\text{Score}_{f4}\,(s_i) = \frac{jumlah\ kata}{Jumlah\ kata\ terpanjang} \tag{2.7}$$

Asumsikan, $\text{Score}_{f4}\,(s_i)$ adalah nilai bobot panjang kalimat dari kalimat ke-dimana didapatkan dari jumlah kata pada kalimat dibagi dalam jumlah kata terpanjang dalam dokumen. Berikut ini contoh dari penggunaan fitur ekstraksi panjang kalimat:

Kalimat 1 : Kami pergi makan dan rekreasi.

Kalimat 2 : Kami pergi lapangan.

Kalimat 3 : Kami sedang memancing.

Kalimat diatas memiliki jumlah kata yang diantaranya lima kata untuk kalimat pertama, tiga kata untuk kalimat kedua dan tiga kata untuk kalimat ketiga. Kata paling banyak terjadi pada kalimat pertama dengan lima kalimat, maka nilai bobot panjang kalimat adalah 5/5 = 1 untuk kalimat pertama, 3/5 = 0.6 untuk kalimat kedua dan 3/5 = 0.6 untuk kalimat ketiga.

2.4.5 Bobot Koneksi Antar Kalimat

Fungsi fitur teks ini adalah menjumlahkan bobot kata dalam suatu kalimat dengan kata yang sama dari kalimat lain dalam suatu dokumen [2]. Dalam menghitung bobot yang dilakukan yaitu dengan menggunakan perhitungan dari TF – IDF (*Term Frequency - Inverse Document Frequency*).

$$\text{TF} - \text{IDF}_{(w)} = w_i = tf_i * idf_i = tf_i * \left(log\frac{N}{n_i}\right) \tag{2.8}$$

Asumsikan $\mathrm{TF}-\mathrm{IDF}_{(w)}$ adalah nilai bobot kata pada kalimat. tf_i adalah frekuensi kemunculan kata dalam dokomuen, N adalah jumlah dari kalimat dan n_i adalah jumlah dari kalimat yang mengandung kata.

Setelah mendapatkan perhitungan dari bobot kata yang menggunakan perhitungan TF – IDF. Adapun perhitungan untuk mendapatkan bobot koneksi antar kalimat sebagai berikut.

$$\mathrm{Score}_{f5}\,(s_i) = \frac{jumlah\ bobot}{jumlah\ bobot\ terbesar} \tag{2.9}$$

Asumsikan $\mathrm{Score}_{f5}\,(s_i)$ adalah nilai bobot koneksi antar kalimat, jumlah bobot adalah jumlah bobot kata pada kalimat $\mathrm{TF}-\mathrm{IDF}_{(w)}$ dan jumlah bobot terbesar adalah bobot terbesar dari $\mathrm{TF}-\mathrm{IDF}_{(w)}$.

Berikut ini contoh dari penggunaan fitur ekstraksi bobot koneksi antar kalimat:

Kalimat 1 : Kami pergi makan dan rekreasi.

Kalimat 2 : Kami pergi lapangan.

Kalimat 3 : Kami sedang memancing.

Kalimat diatas dihitung bobot frekuensinya dengan persamaan TF-IDF. Setelah mendapatkan bobot frekuensinya bobot dengan nilai terbesar dijadikan pembanding dengan jumlah bobot pada tiap kalimat.

2.5 *Neural Network*

Neural network merupakan pembelajaran dari kecerdasan buatan yang memiliki karakteristik meniru arsitektur jaringan syaraf manusia. Pembelajaran merupakan proses tambahan pengetahuan pada *neural network* yang bersifat kesinambungan sehingga pembelajaran yang dihasilkan menjadi masukan pola tertentu tergantung dari nilai bobot sambungan antar *neuron* dalam jaringan syaraf tiruan tersebut. Lapisan n*eural network* terdiri dari beberapa lapis yaitu lapisan masukan (*input layer*), lapisan tersembunyi (*hidden layer*) dan lapisan keluaran (*output layer*). Setiap lapisan memiliki jumlah *neuron* yang berbeda-beda dan setiap neuron menerapkan fungsi aktivasi terhadap input jaringan [14].

2.5.1 Fungsi Aktivasi

Fungsi aktivasi adalah fungsi yang digunakan untuk memformulasikan *output* dari tiap *neuron* fungsi aktivasi dalam *backpropagation* mempunyai karakteristik penting diantaranya fungsi aktivasi harus bersifat *kontinu*, *terdiferensial* dengan mudah dan tidak turun [15]. Beberapa fungsi aktivasi yang selalu digunakan sebagai berikut.

1. Fungsi *aktivasi linier*

Fungsi aktivasi linier adalah fungsi yang selalu digunakan apabila keluaran dari jaringan berupa sembarang bilangan *real*. Fungsi aktivasi linier umumnya digunakan pada *neuron output* bukan hanya pada range [0, 1] atau [-1, 1].

$$f(x) = x \tag{2.10}$$

2. Fungsi aktivasi *sigmoid biner*

Fungsi aktivasi sigmoid biner adalah fungsi aktivitas yang sering dugunakan karena fungsi nilainya terletak antara 0 dan 1 dan dapat diturunkan dengan mudah. Output dari fungsi ini adalah pengkonversian input yang bernilai *continue* menjadi bilangan biner (0 atau 1).

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.11}$$

3. Fungsi aktivasi *sigmoid bipolar*

Fungsi aktivasi *sigmoid bipolar* merupakan fungsi pengembangan dari *sigmoid biner* yang memiliki fungsi nilai terletak antara -1 dan 1.

$$f(x) = \frac{1 - e^{-x}}{(1 + e^{-x})} \tag{2.12}$$

4. Fungsi aktivasi tangen hiperbola

Fungsi aktivasi yang memiliki fungsi nilai yang sama dengan bipolar yaitu antara -1 dan 1.

$$f(x) = \frac{2}{1 + e^{-x}} - 1 \tag{2.13}$$

Fungsi aktivasi yang digunakan dalam penelitian ini adalah fungsi aktivitas sigmoid biner, karena digunakan untuk jaringan syaraf yang dilatih dengan menggunakan metode *backpropagation.*

Keterangan:

$f(x)$ = Data hasil normalisasi

x = Data awal

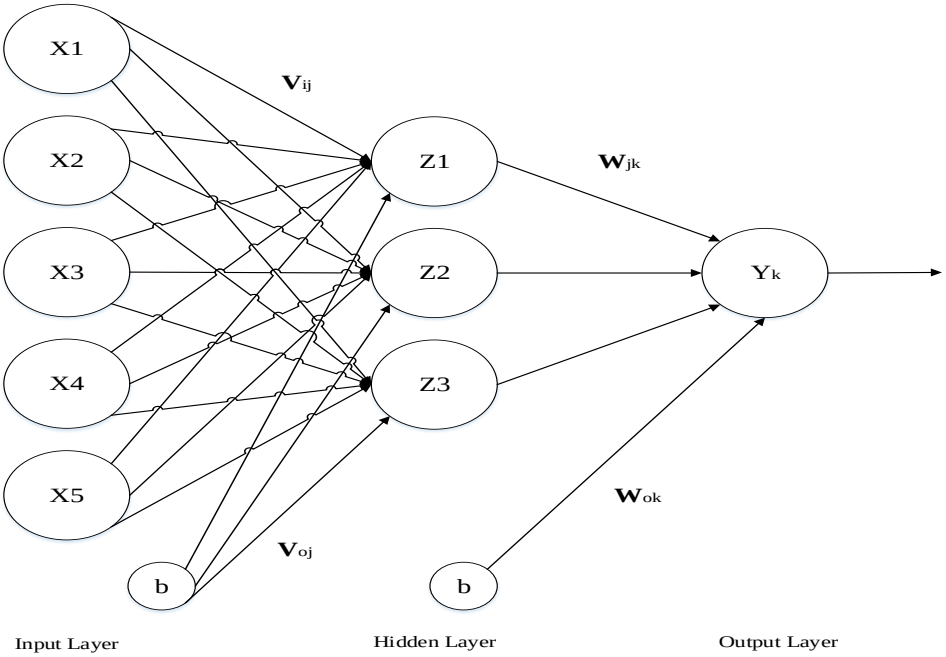
e = Eksponensial

2.5.2 *Backpropagation*

Backpropagation merupakan algoritma pembelajaran *supervised learning* yang digunakan *perceptron* dengan banyak lapisan untuk mengubah nilai bobot pada *neuron* yang terhubung pada lapisan tersembunyinya [16]. *Backpropagation* terdiri dari banyak lapisan diantaranya lapisan masukan (*input layer*), lapisan yang terdiri dari neuron input baik dari *neuron* pertama sampai *neuron network*. Lapisan tersembunyi (*hidden* layer), terdiri dari minimal satu lapis layer mulai dari neuron tersembunyi pertama sampai neuron ke-j. Menentukan jumlah *neuron* pada *hidden layer* merupakan bagian penting dalam arsitektur jaringan saraf menggunakan terlalu sedikit *neuron* pada lapis tersembunyi berkemungkinan menghasilkan *underfittin* dan sebaliknya jika menggunakan terlalu banyak *neuron* pada lapis tersembunyi berkemungkinan menghasilkan *overfitting.* Menurut Simon S. Haykin [17] yaitu, jumlah *hiden neuron* terdiri dari dua sampai sembilan sudah dapat menghasilkan hasil yang baik dalam jaringan tetapi pada dasarnya tidak ada batasan dalam menentukan jumlah *hidden neuron*. Menurut Jeff Heaton [18] yaitu, terdapat beberapa aturan dalam menentukan jumlah *neuron* pada *hidden* layer salah satunya jumlah *hidden neuron* harus 2/3 dari ukuran *input* layer ditambah dari ukuran *output layer*. Lapisan keluaran (*output layer*) memiliki jumlah lapisan satu yang terdidiri dari *neuron output* pertama sampai *neurorn output* ke-m. Jumlah dari *neuron output* tergantung tipe dan performa dari jaringan saraf itu sendiri.

Pelatihan *backpropagation* terdiri dari tiga fase diantaranya propagasi maju yang merupakan sinyal masukan untuk dipropagasikan ke *hidden layer* menggunakan fungsi aktivasi yang telah ditentukan hingga menghasilkan keluaran jaringan. Propagasi mundur adalah propagasi mulai dari garis yang berhubungan langsung dengan *neuron* di *layer* keluaran. Perubahan bobot adalah bobot semua garis yang dimodifikasi secara bersamaan. Ketiga fase tersebut diulang terus hingga kondisi penghentian terdipenuhi. Kondisi penghentian yang sering dipakai adalah jumlah maksimal iterasi (*epoch*) atau minimal kesalahan (*error*). Contoh arsitektur

backpropagation (MAJU) yang dimana input terdiri dari lima neuron (X) dengan tiga hidden layer dan satu output dilihat pada gambar 2.2 sebagai berikut.



Gambar 2. 2 Contoh Arsitektur *Backpropagation* (Maju)

Keterangan:

X = Nilai masukan (*input layer*)

V_{ij} = Nilai bobot sambungan pada lapisan *input* X ke unit Z

Z = Bobot pada lapisan tersembunyi (*hidden layer*)

W_{jk} = Nilai bobot sambungan pada lapisan tersembunyi Z ke unit Y

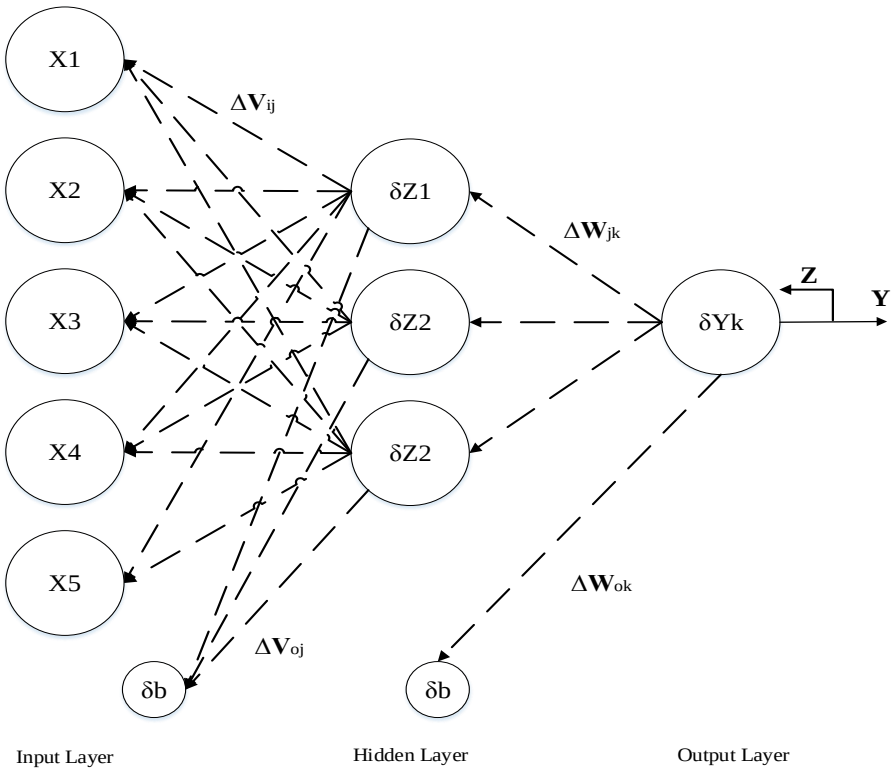
Y_k = Keluaran (*output*)

b = Bias

b_{0j} = Bobot bias pada lapisan tersembunyi x ke unit Z

b_{0k} = Bobot bias pada lapisan keluaran Z ke unit Y

Pada gambar 2.2 adalah contoh arsitektur *backpropagation* (maju) yang dimana proses berikutnya merupakan perbaikan nilai bobot pada propagation mundur. Perhitungan galat dimulai dari nilai Y yang dimana dihitung nilai galat Y (output) ke Z (hidden layer) dan X (input layer) untuk menemukan nilai bobot baru yang digunakan kembali pada *backpropagation* (maju). Contoh arsitektur *backpropagasi* mundur dapat dilihat pada gambar 2.3 sebagai berikut.



Gambar 2. 3 Contoh Arsitektur *Backpropagation* (Mundur)

Keterangan:

X = Masukan (input)

ΔV_{ij} = Bobot perbaikan sambungan Z ke unit X

δZ = Bobot perbaikan pada lapisan tersembunyi

ΔW_{jk} = Nilai perbaikan sambungan dari Y ke unit z

δY_k = Keluaran (output)

b = Bias

Δb_{oj} = Bobot bias pada lapisan tersembunyi Z ke unit b

Δb_{ok} = Bobot bias pada lapisan keluaran Y ke unit b

Proses perbaikan bobot pada lapisan tersembunyi diproses kembali kepada proses pelatihan *backpropagation* maju dengan nilai bobot baru pada bobot penghubung kepada *hiden layer* dan *output layer* sampai kondisi yang di tentukan terpenuhi.

2.5.3 Pelatihan *Backpropagation*

Secara umum langkah – langkah dalam pelatihan *neural network* menggunakan algoritma *backpropagation* adalah sebagai berikut.

Langkah 0:

Berikan inisialisasi penimbang (diberikan nilai kecil secara acak)

Langkah 1:

Jika kondisi berhenti tidak tercapai, maka ulangi langkah 2 sampai 9 hingga kondisi iterasi terpenuhi (kondisi berhenti jika perluangan tercapai maksimal atau *mean square error* (MSE) lebih kecil dari rasio pembelajaran).

Langkah 2:

Untuk masing – masing pasangan data latih (*training data*) lakukan langkah 3 sampai 8.

Propagasi maju

Langkah 3:

Masukan masing – masing unit menerima sinyal masukan (X_i) dan sinyal disebarkan ke unit – unit bagian berikutnya (*hidden layer* atau lapisan tersembunyi).

Langkah 4:

Masing – masing dari unit dalam lapisan tersembunyi dikalikan dengan *factor* penimbang dan dijumlahkan serta ditambahkan (inisialisasi pada langkah 0) dengan bias:

$$Z_{inj} = V_{oj} + \sum_{i=1}^{n} X_i V_{ij} \qquad (2.14)$$

V_{oj} bias pada fungsi tersembunyi dan kemudian hitung dengan sesuai fungsi aktivasi yang digunakan.

$$Z_j = f(Z_{inj}) \qquad (2.15)$$

Bila fungsi sigmoid yang digunakan adalah fungsi sigmoid biner maka gunakan persamaan (2.11)

Langkah 5:

Masing – masing unit keluaran Y_k, (k = karena banyak output yang digunakan satu, maka outputnya keluarnya adalah satu). Unik (Z_p) dikalikan dengan faktor penimbang dan dijumlahkan.

$$Y_{ink} = W_{ok} + \sum_{j=1}^{p} Z_i W_{ij} \qquad (2.16)$$

W_{ok} bias pada unit keluaran dan hitung kembali dengan fungsi aktivasi *backpropagation* dan galatnya.

$$Y_k = f(\,Y_{ink}\,) \tag{2.17}$$

Bila fungsi sigmoid yang digunakan adalah fungsi sigmoid biner maka maka gunakan persamaan (2.11)

Propagasi mundur

Langkah 6:

Masing – masing unit dari keluaran Y_k menerima pola target yang sesuai dengan pola masukan saat pelatihan dan dihitung galatnya.

$$\delta_k = (T_k - Y_k)f'(1 - \,Y_k) \tag{2.18}$$

Dan hitung perbaikan faktor penimbang kemudian memperbaiki W_{jk}

$$\Delta W_{jk} = a.\,\delta_k.\,z_j \tag{2.19}$$

Menghitung perbaikan koreksi

$$\Delta W_{ok} = a.\,\delta_k \tag{2.20}$$

Dan menggunakan nilai dari δ_k pada semua unit lapisan sebelumnya.

Langkah 7:

Masing – masing penimbang yang menghubungkan unit – unit dari lapisan keluaran dengan unit – unit pada lapisan tersembunyi (*hidden layer*) ke unit – unit lapisan berikutnya.

$$\delta_in_j = \sum_{k=1}^{n} \delta_k\, W_{jk} \tag{2.21}$$

Dan selanjutnya dikalikan dengan turunan fungsi aktivitasnya untuk menghitung galat.

$$\delta_z = \delta'_{in_{jf}}(y_inj) \tag{2.22}$$

Menghitung perbaikan penimbang yang digunakan untuk memperbaiki V_{ij}

$$\Delta V_{ij} = a.\,\delta_z X_i \tag{2.23}$$

Memperbaiki penimbang dan bias

$$\Delta V_{oj} = a.\,\delta_z \tag{2.24}$$

Langkah 8:

Masing – masing dari keluaran unit Y_k diperbaiki bias dan penimpangnya

$$W_{jk}(baru) = W_{jk}(lama)\Delta W_{jk} \quad (2.25)$$

Masing masing dari unit tersembunyi Z_{jk}, diperbaiki bias dan penimbangnya

$$V_{ij}(baru) = V_{ij}(lama)\Delta V_{ij} \quad (2.26)$$

Langkah 9:

Periksa kondisi akhir iterasi (pemberhentian) pengujian iterasi dilakukan dengan menghitung nilai dari *error*, apakah batas error minimum telah tercapai dan apakah batas iterasi maksimum telah tercapai [19]. Rumus pencarian nilai *error* sebagai berikut.

$$E = \frac{1}{X}\sum_{i=1}^{X}(T_k - Y_k)^2 \quad (2.27)$$

Daftar keterangan dari notasi *backpropagation* diatas dapat dilihat pada table 2.7 sebagai berikut.

Tabel 2. 7 Keterangan Notasi *Backpropagation*

Notasi	Keterangan
T_k	Pola keluaran target dari pelatihan.
X	Lapisan masukan
Z	Lapisan tersembunyi
Y	Lapisan keluaran
Z_in_j	Keluaran untuk unit Zj
Y_in_k	Net masukan untuk unit Yk
Z_j	Nilai aktifasi dari unit Zj
W_{ok}	Nilai penimbang sambungan pada bias untuk unit Yk
W_{jk}	Nilai penimbang sambungan dari Z ke unit Yk
V_{ok}	Nilai penimbang sambungan pada bias untuk unit Z
V_{ij}	Nilai penimbang sambungan pada X untuk unit Z
∇W_{ik}	Nilai bobot perbaikan sambungan Yk ke Z
ΔV_{ij}	Nilai bobot perbaikan sambungan Z ke X
∇W_{ok}	Nilai penimbang sambungan pada bias untuk unit Z
$\Delta \mathrm{V}_{oj}$	Nilai penimbang sambungan pada bias untuk unit X

Tabel 2. 7 Keterangan Notasi *Backpropagation* (Lanjutan)

Notasi	Keterangan
δk	Faktor pengaturan nilai penimbang sambungan pada lapisan keluaran
δz	Faktor pengaturan nilai penimbang sambungan pada lapisan tersembunyi
α	*learning rate*
E	Total galat

2.5.3 Pengujian *Backpropagation*

Setelah pelatihan, jaringan saraf *backpropagation* diaplikasikan dengan hanya menggunakan tahap perambatan maju dari algoritma pelatihan. Prosedur aplikasinya adalah sebagai berikut.

Langkah 0: Berikan inisialisasi penimbang (bobot akhir dari pelatihan).

Langkah 1: Untuk tiap vektor masukan, lakukan langkah 2 sampai 4.

Langkah 2: Masukan nilai aktivasi unit masukan X .

Langkah 3: Hitung fungsi lapisan tersembunyi Z beserta biasnya

$$Z_{inj} = V_{oj} + \sum_{i=1}^{n} X_i V_{ij} \quad (2.14)$$

$$Z_{inj} = f(Z_{inj}) \quad (2.15)$$

Langkah 4: Hitung fungsi lapisan output Y beserta biasnya

$$Y_{ink} = W_{ok} + \sum_{j=1}^{p} Z_i W_{ij} \quad (2.16)$$

$$Y_k = f(Y_{inj}) \quad (2.17)$$

Langkah 5: Jika hasil perhitungan output $Y_k \geq 0{,}5$ maka $Y_k = 1$, dan jika $Y_k = 0$. Nilai 1 adalah nilai sebagai ringkasan dan 0 adalah bukan sebagai ringkasan.

2.6 Pengujian

Pengujian yang dilakukan dalam penelitian ini menggunakan standar pengukuran akurasi yang biasa digunakan dalam penelitian peringkasan teks diantaranya *recall, precision* dan *f-measure* [20].

2.6.1 *Recall*

Recall adalah tingkat keberhasilan dalam sebuah ringkasan. Adapun rumus untuk menghitung *recall* dapat dilihat pada persamaan berikut:

$$r = \frac{correct}{correct + missed} \quad (2.28)$$

Asumsikan r adalah nilai tingkat keberhasilan *recall*, *correct* adalah banyaknya kalimat yang berhasil di ekstrak sistem sesuai dengan yang diekstrak manusia dan missed adalah jumlah kalimat yang diekstrak manusia tetapi tidak terdapat dalam kalimat yang diekstrak sistem.

2.6.2 *Precision*

Precision badalah tingkat ketepadan dalam sebuah ringkasan. Adapun, rumus untuk menghitung *precision* dapat dilihat pada persamaan berikut.

$$p = \frac{correct}{correct + wrong} \quad (2.29)$$

Asumsikan p adalah nilai tingkat ketepatan, correct adalah banyaknya kalimat yang berhasil di ekstrak sistem sesuai dengan yang diekstrak manusia dan wrong adalah jumlah kalimat yang diekstrak sistem tetapi tidak terdapat dalam kalimat yang diekstrak manusia

2.6.3 *F-Measure*

F-Measure adalah perhitungan yang dihasilkan dari evaluasi recall dan precision. Nilai *recall* dan *precision* pada suatu keadaan dapat memiliki bobot yang berbeda. Adapun perhitungan *f-measure* dapat dilihat pada persamaan berikut.

$$f - measure = \frac{2 * precision * recall}{precision + recall} \quad (2.30)$$

2.7 Pemodelan

Pemodelan adalah proses pengembangan model abstrak dari sistem yang dimana tiap model memperlihatkan gambaran berbeda dari sistem. Pemodelan yang digunakan dalam penelitian sebagai berikut.

2.7.1 *Flowchart*

Flowchart adalah bagan yang menunjukkan alur program atau langkah prosedur sistem secara logika. Dalam menggambarkan langkah prosedur dalam sistem digambarkan dengan simbol dan setiap simbol menyatakan gambaran proses tertentu, bagan alur program merupakan alur yang digunakan pada sistem.

2.7.2 Diagram Konteks

Diagram konteks adalah diagram yang terdiri dari suatu proses dengan menggambarkan ruang lingkup dari suatu sistem. Diagram konteks merupakan bagian data flow diagram di level tertinggi yang menggambarkan seluruh input atau output sistem dan memberi gambaran tentang keseluruan sistem. Dalam diagram konteks hanya ada satu proses serta tidak boleh ada store dalam diagram konteks [21].

2.7.3 Data Flow Diagram

Data Flow Diagram adalah diagram yang mengambarkan arus data sistem dengan menggunakan notasi simbol. Data flow diagram digunakan untuk mengambarkan secara logika dan menjelaskan arus data dari mulai pemasukan sampai dengan keluaran. Tingkat arus data mulai dari diagram konteks yang menjelaskan secara umum suatu sistem dari level 0 hingga dikembangkan menjadi level 1 sampai sistem tergambarkan keseluruhan secara rinci. Gambaran ini tidak tergantung pada perangkat keras, perangkat lunak, struktur data atau organisasi file [21].

2.8 Bahasa Pemograman

Bahasa pemrograman adalah instruksi atau aturan standar yang disusun sedemikian rupa sehingga memungkinkan pengguna membuat program yang dapat dijalankan dengan instruksi atau aturan tersebut [21]. Berikut adalah beberapa beberapa contoh bahasa pemrograman yaitu C, C++, C#, Pascal, Java, JavaScript, PHP, SQL dan Python. Bahasa Pemrograman yang digunakan pada peringkasan teks ini adalah menggunakan PHP.

2.8.1 *Hypertext Preprocessor* (PHP)

Hypertext Prepocessor (PHP) adalah bahasa pemograman yang banyak digunakan dalam situs website. Pemograman PHP sangat cocok dikembangkan dalam lingkungan web, karena PHP bisa dilekatkan pada script HTML atau sebaliknya dan dapat dikombinasikan dengan JavaScript. PHP tergolong juga sebagai bahasa pemograman yang berbasis server diaman semua script PHP diletakkan diserver dan diterjemahkan oleh web server terlebih dahulu, kemudian hasil terjemahan itu dikirim ke browser *client*. Secara teknologi, bahasa

pemograman PHP memiliki kesamaan dengan bahasa ASP (*Active Server Pag*e), Cold Fusion, JSP (*Java Server Page*), ataupun *Perl* [21].

2.9 Perangkat Lunak Pendukung

Beberapa perangkat lunak dan bahasa pemrograman yang digunakan untuk mendukung dalam pengembangan aplikasi yang dibuat.

2.9.1 XAMPP

XAMPP adalah satu paket software web server yang terdiri dari Apache, Mysql, PHP dan phpMyadmin. Pengaturan dalam XAMPP bagi pemula sangat mudah dioperasikan dan proses instalasi XAMPP sangat mudah, karena XAMPP melakukan instalasi dan konfigurasi secara otomatis terhadap melakukan konfigurasu Apache, PHP dan MySQL. Perangkat lunak yang digunakan adalah XAMPP 3.2.2.

2.9.2 *Sublime Text Editor*

Sublime Text Editor merupakan salah satu editor *teks* untuk mengedit atau membuat aplikasi selain itu mendukung pada operation system seperti Linux, Mac Os X, dan juga windows. Banyak fitur yang tersedia pada sublime text editor diantarnya minimap, membuka script secara *side* by *side*, bracket *highlight* sehingga tidak kesulitan dalam mencari pasangannya, kode *snippets*, *drag* and *drop* direktori ke sidebar terasa mirip dengan *TextMate* untuk Mac OS. Perangkat lunak yang digunakan adalah *Sublim Text* versi 3.