BAB III. METODOLOGI PENELITIAN

Pada bab ini akan dibahas metode penelitian yang digunakan dan langkah – langkah yang dilakukan dalam rangka mencapai tujuan yang diharapkan dalam penelitian ini

3.1 Waktu dan Tempat Penelitian

Penelitian dilakukan di kampus Politeknik Negeri Malang. Penelitian dilaksanakan selama 5 bulan dimulai pada bulan Januari 2021 sampai dengan Mei 2021.

3.2 Teknik Pengumpulan Data

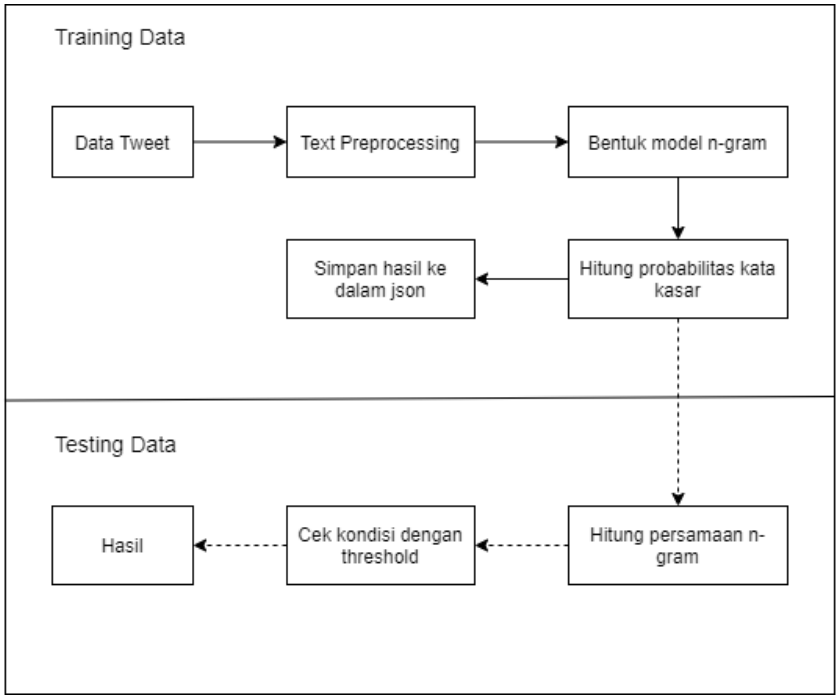
Data penelitian yang diolah pada penelitian ini berupa kata kasar, kata alay (*slang word*) pada media sosial dan kata baku dari Internet.

1. Pengumpulan Data Sekunder (*Secondary Data Collection*)

Pengambilan data penelitian dilakukan dengan cara pengumpulan data sekunder (data yang didapat secara tidak langsung atau melalui perantara media) yang berupa dataset kosa kata kasar dan kata alay pada twitter yang didapat dari situs resmi Kaggle: https://www.kaggle.com/ilhamfp31/indonesian-abusive-and-hate-speech-twitter-text dan beberapa tambahan data kata alay yang didapat dari github: https://github.com/louisowen6/NLP_bahasa_resources serta data kata baku yang telah dikumpulkan dari situs https://penerbitdeepublish.com/kata-baku-dan-tidak-baku/ dan https://bocahkampus.com/contoh-kata-baku-dan-tidak-baku sebagai data pembanding. Data *tweet* yang digunakan pada penelitian ini menggunakan perbandingan 70:30 (70% data latih, 30% data uji) dari 1000 data *tweet*.

3.3. Teknik Pengolahan Data

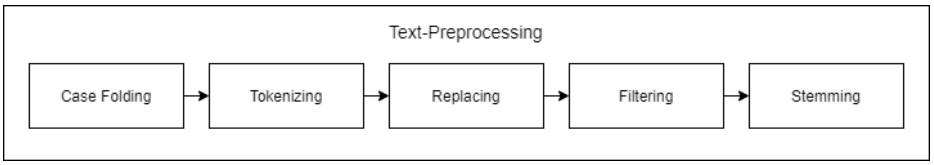
Data pesan akan diproses dengan teknik *text mining* melalui tahap *training* dan *testing* dengan skenario seperti Gambar 3.1 berikut.



Gambar 3.1 Skenario pengolahan data

1. *Training Data*

- *Text-Preprocessing*



Gambar 3.2 *Text-Preprocessing*

- *Case Folding*

Data tweet atau pesan akan disetarakan bentuk standarnya menjadi huruf kecil. Contohnya dapat dilihat pada Tabel 3.1.

Tabel 3.1 Proses *Case Folding*

Contoh Data yang dikirim	**Hasil *Case Folding***
Bener2 berasa yang paling suci nih si anying.	bener2 berasa yang paling suci nih si anying.

- *Tokenizing*

Data tweet atau pesan akan diurai dari kalimat menjadi kumpulan kata yang menyusunnya. Karakter seperti angka, tanda baca dan selain huruf akan dihilangkan pada tahap ini. Contohnya dapat dilihat pada Tabel 3.2.

Tabel 3.2 Proses *Tokenizing*

Hasil *Case Folding*	**Hasil *Tokenizing***
bener2 berasa yang paling suci nih si anying.	bener berasa yang paling suci nih si anying

- *Replacing*

Sebelum memasuki tahap *filtering*, penulis melakukan tahap *replacing* untuk mengganti kata alay menjadi kata formalnya. Contohnya dapat dilihat pada Tabel 3.3.

Tabel 3.3 Proses *Replacing*

Hasil *Tokenizing*	**Hasil *Replacing***
bener berasa yang paling suci nih si anying	benar berasa yang paling suci ini si anying

- *Filtering*

Pada tahap ini, digunakan algoritma *stopword* dengan membuang kata tidak penting pada data tweet atau pesan. Contohnya dapat dilihat pada Tabel 3.4.

Tabel 3.4 Proses *Filtering*

Hasil *Replacing*	**Hasil *Filtering***
benar berasa yang paling suci ini si anying	berasa suci anying

- *Stemming*

Pada tahap ini, kata hasil *filtering* yang memiliki imbuhan akan diubah menjadi kata dasarnya. Digunakan *library sastrawi stemmer* untuk mengecek kata dasar Bahasa Indonesia. Contohnya dapat dilihat pada Tabel 3.5.

Tabel 3.5 Proses *Stemming*

Hasil *Filtering*	**Hasil *Stemming***
berasa	rasa
suci	suci
anying	anying

- Membentuk model *N-Gram*

Setelah tahap *text-preprocessing*, pembentukan model *N-Gram*. Pada tahap ini, setiap kata akan dipotong-potong menjadi n bagian. Jika *bigram* maka n = 2, sedangkan *trigram* n = 3.

Contoh: kata “anying”,

Bentuk model *bigram* dari kata “anying”: an, ny, yi, in, ng

Bentuk model *trigram* dari kata “anying”: any, nyi, yin, ing

- Hitung probabilitas kata kasar

Setelah melalui tahap pemodelan *N-Gram*, penghitungan probabilitas dengan *N-gram*. Agar pemakaian daya komputasi dan waktu tidak terlalu besar, maka penulis hanya menggunakan model *bigram* dan *trigram*.

Fitur *bigram* dapat dihitung dengan rumus:

$$P(w_i|w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})} \quad (3.1)$$

Untuk *trigram* dapat dihitung dengan rumus:

$$P(w_i|w_{i-2}, w_{i-1}) = \frac{c(w_{i-2}, w_{i-1}, w_i)}{c(w_{i-2}, w_{i-1})} \quad (3.2)$$

Deskripsi:

w_i : jumlah kemunculan *bigram/trigram* kata selanjutnya

w_{i-1}, w_{i-2} : jumlah kemunculan *bigram/trigram* kata yang dideteksi

Hasil model *bigram* dari kata “anying”: an, ny, yi, in, ng

$$P(nj|an) = \frac{c(any)}{c(an)} = \frac{2}{22} = 0.09$$

$$P(ji|nj) = \frac{c(nyi)}{c(nj)} = \frac{2}{2} = 1$$

$$P(in|ji) = \frac{c(yin)}{c(yi)} = \frac{2}{6} = 0.33$$

$$P(ng|in) = \frac{c(ing)}{c(in)} = \frac{5}{6} = 0.83$$

Maka probabilitas dari kata “anying”: 0.09 + 1 + 0.33 + 0.83 = 2.25

2. *Testing Data*

Untuk proses pada *testing* sama dengan proses *training*, hanya saja setelah menghitung probabilitas kata kasar akan dilanjutkan dengan membandingkan nilai probabilitas kata kasar pada proses *testing* dengan probabilitas kata kasar *data training* yang sudah disimpan.

- Hitung persamaan *N-Gram*

$$simQ = \sum_{i=a}^{z} \frac{(fi - fRi)^2}{fRi} \quad (3.3)$$

Deskripsi:

fi : probabilitas *n-gram* ke-i dari kata yang terdeteksi

fRi : probabilitas *n-gram* ke-i dari kata di *data training*

Contoh probabilitas dari kata yang terdeteksi “anying”: 1.47 dan kata “anying” pada training: 2.25

$$simQ = \frac{(fi - fRi)^2}{fRi} = \frac{(1.47 - 2.25)^2}{2.25} = 0.2704$$

Semakin kecil hasil dari perhitungan, maka semakin besar kemiripan dengan *data training*.

- Cek kondisi dengan *threshold*

Terakhir mengecek hasil persamaan dengan *threshold*, jika nilai persamaan lebih kecil dari *threshold* maka hasilnya kata tersebut terindikasi kata kasar yang mana kata tersebut akan di-*filter* dengan simbol ‘*’. Sebaliknya jika nilai lebih besar maka kata akan ditandai saja.

Contoh hasil kondisi *True*:

benar berasa yang paling suci ini si **********

Contoh hasil kondisi *False*:

benar berasa yang paling suci ini si **anying**

3.4. Uji Coba Sistem

Pengujian sistem dilakukan akan dilakukan ketika semua tahap perancangan dan implementasi sistem sudah selesai, hal ini bertujuan untuk mengetahui seluruh sistem dibangun bekerja sesuai fungsinya dengan menggunakan teknik pengujian

Black box. *Black box testing* adalah metode pengujian fungsionalitas perangkat lunak tanpa tahu struktur atau cara kerjanya.

Adapun hal yang dilakukan dalam pengujian dari sistem ini meliputi beberapa tahapan, yaitu:

1. Pengujian sistem apakah sudah sesuai atau belum dengan rancangan awal.
2. Pengujian akurasi metode *N-gram* terhadap penilaian kata buruk.

   Pengujian dilakukan dengan menghitung nilai *recall* dan *precision* terlebih dahulu dengan rumus persamaan 2.1 dan 2.2. Hasil dari recall dan precision akan digabung menjadi *f-measure* di mana teknik ini digunakan untuk mengukur tingkat keberhasilan sistem temu kembali. Terakhir menghitung *accuracy*. Kedua perhitungan ini dapat menggunakan rumus 2.3 dan 2.4.

Tabel 3.6 *Confusion Matrix* pada Uji Coba Sistem

		Prediksi	
		Prediksi Benar	Prediksi Salah
Aktual	Aktual Benar	16	7
	Aktual Salah	4	7

$$Recall = \frac{TP}{TP + FN} = \frac{16}{16 + 7} = 2.285$$

$$Precision = \frac{TP}{TP + FP} = \frac{16}{16 + 4} = 0.8$$

$$F - Measure = \frac{2 \times (Recall \times Precision)}{Recall + Precision} = \frac{2 \times (2.285 \times 0.8)}{2.285 + 0.8} = \frac{3.656}{3.085} = 1.185$$

$$Accuracy = \frac{TP + FP}{TP + FP + TN + FN} = \frac{16 + 4}{16 + 4 + 7 + 7} = 0.588$$