BAB 3
METODOLOGI PENELITIAN

Pada bab ini, dijelaskan gambaran umum metodologi penelitian yang menjelaskan alur penelitian.

3.1 Metodologi Penelitian

Berikut adalah tahapan-tahapan dalam metode yang dilaksanakan dalam pelaksanaan penelitian.

1. Telaah Literatur

   Dalam tahap ini, dilakukan studi mengenai proses-proses yang akan dilakukan, yaitu *Preprocessing*, *Word Embedding* menggunakan *FastText*, *Convolutional Neural Network*, serta evaluasi model.

2. Pengumpulan Data

   Dataset diperoleh dari *Cyberbullying Classification* yang didapatkan dari internet di *website* Kaggle dari penelitian terkait[41]. Dataset terbagi atas enam kelas, yaitu, *not_cyberbullying*, *gender*, *religion*, *other_cyberbullying*, *age*, dan *ethnicity*.

3. Perancangan dan Pembuatan Sistem

   Pada tahap ini, dilakukan dengan membuat rancangan program yang akan dibangun dalam bentuk *flowchart*, kemudian akan dilakukan implementasi terhadap program analisis sentimen menggunakan FastText sebagai *word embedding* dan klasifikasi menggunakan *Convolutional Neural Network*. Bahasa pemrograman yang digunakan adalah Python, dengan *Google Colab* sebagai *environment* pemrograman.

4. Evaluasi Model

   Program analisis sentimen yang telah dirancang kemudian akan dievaluasi menggunakan parameter-parameter metrik evaluasi, yaitu akurasi, *F1-score*, *precision*, dan *recall*.

5. Penulisan Laporan Penelitian

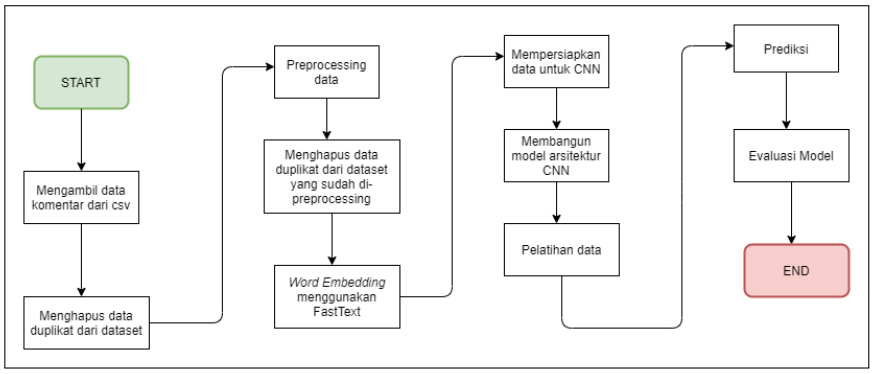
Pada tahap ini, penulisan laporan penelitian ditulis untuk mendokumentasikan hasil penelitian sehingga penelitian terdokumentasi dengan baik dan dapat digunakan sebagai bahan referensi untuk penelitian atau keperluan mendatang.

3.2 Perancangan Sistem

Langkah-langkah dalam perancangan sistem untuk program analisis sentimen terdiri atas pembuatan *flowchart* sebagai berikut.

A Flowchart Utama

Gambar 3.1 menunjukkan *flowchart* utama dari sistem analisis sentimen yang dibuat. Dimulai dari pengambilan data berbentuk *csv*, menghapus data duplikat dari dataset, melakukan *pre-processing*, menghapus data duplikat dari dataset yang sudah di-*preprocess*, melakukan *Word Embedding* menggunakan *FastText*, mempersiapkan data untuk model (mengubah data teks menjadi *integer*). Kemudian akan dilakukan pembangunan arsitektur model CNN sebelum melakukan pelatihan, prediksi, dan evaluasi model.



Gambar 3.1. *Flowchart* utama

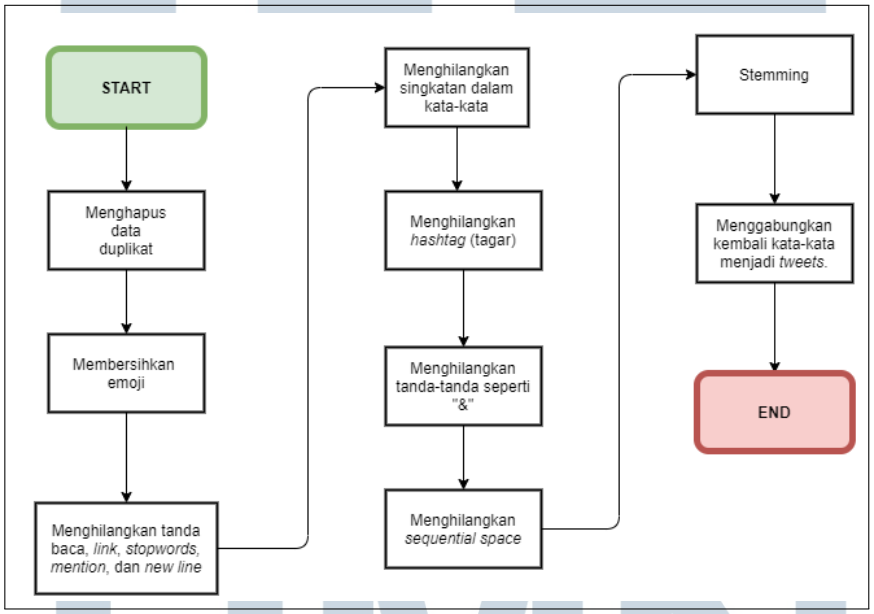
B Flowchart Preprocessing

Gambar 3.2 menunjukkan *flowchart* untuk tahap *preprocessing*. Pada proses ini, ada tahap-tahap: penghapusan data duplikat, membersihkan *emoji*, menghilangkan tanda baca, *link*, *stopwords*, *mention*, serta *new line*, menghilangkan singkatan dalam kata-kata, menghilangkan *hashtag* atau tagar, menghilangkan tanda-tanda seperti ”&”, menghilangkan *sequential space*,

UNIVERSITAS MULTIMEDIA NUSANTARA

Stemming, dan menggabungkan kembali kata-kata menjadi *tweets* yang sudah melewati *preprocessing*.

Penghapusan data duplikat dilakukan supaya tidak ada kelebihan informasi untuk diproses. Kemudian, dilanjutkan dengan penghapusan tanda-tanda seperti *emoji*, tanda baca, *link* website dalam *tweets*, *stopword*, *mention* pada akun lain, serta *new line*. Dilakukan juga penghilangan singkatan dalam kata-kata, *hashtag*, dan *Stemming*. *Stemming* adalah proses mengembalikan kata-kata tertentu menjadi bentuk dasarnya. Akhirnya, setelah *preprocessing*, komentar dibentuk kembali dari kata-kata yang sudah melewati *preprocessing*.

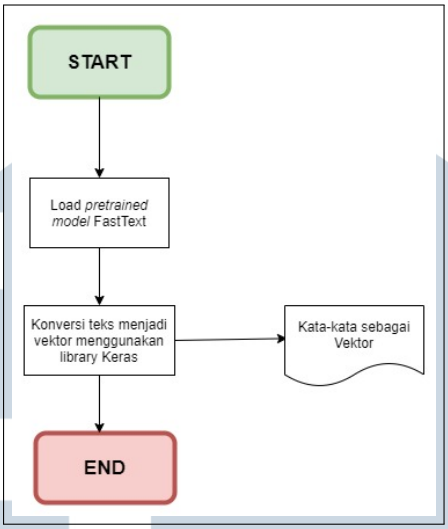


Gambar 3.2. *Flowchart preprocessing*

C Flowchart Word Embedding

Setelah melakukan *preprocessing* pada data, dilanjutkan dengan proses *Word Embedding* menggunakan *library* dan *function* yang telah disediakan oleh Keras yang ditunjukkan pada Gambar 3.3.

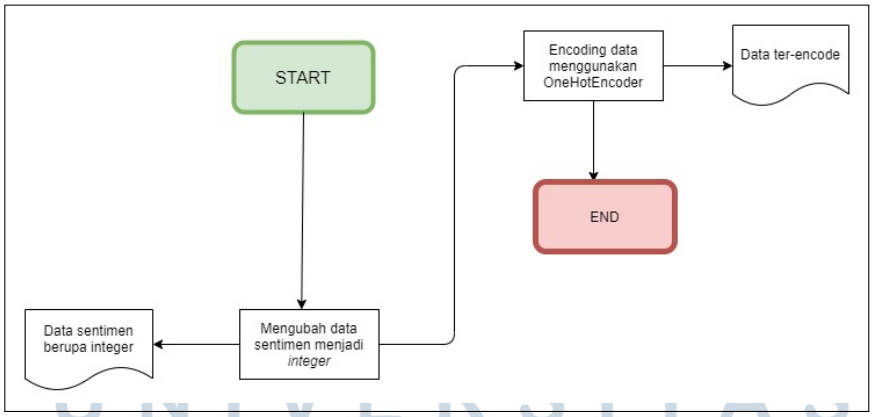
UNIVERSITAS MULTIMEDIA NUSANTARA



Gambar 3.3. *Flowchart word embedding*

D Flowchart Encoding Target

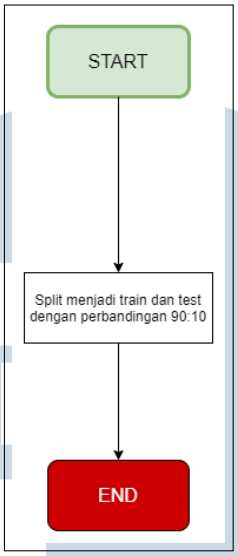
Setelah melakukan *word embedding* pada target, dilakukan *encoding* pada fitur untuk mengubah data sentimen menjadi *integer* yang dapat dipelajari oleh model pembelajaran, karena model *machine learning* tidak dapat bekerja menggunakan data kategorikal. Proses ditunjukkan pada Gambar 3.4.



Gambar 3.4. *Flowchart* untuk *encoding* sentimen

E Flowchart Pembagian Data

Supaya sebuah model dapat melakukan prediksi, diperlukan proses pembagian *dataset*. Gambar 3.5 menunjukkan proses pembagian *dataset* dari data fitur dan label yang sudah di-*preprocessing* dan di-*encoding* sebelumnya.



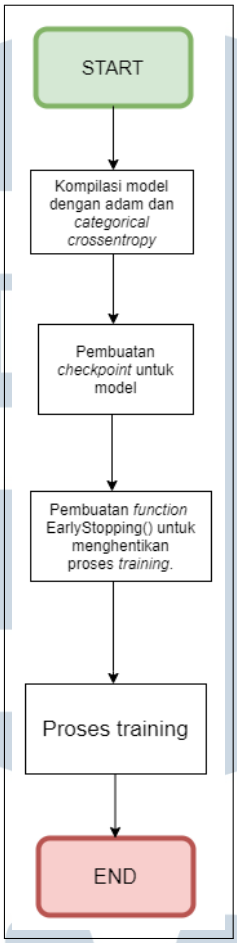
Gambar 3.5. *Flowchart* pembagian data

Selanjutnya, *dataset* dibagi menjadi data *train* dan *test*. Data *train* digunakan untuk melakukan pelatihan pada *dataset*, sedangkan data *test* digunakan untuk menguji performa dari model. Pada penelitian, perbandingan antara *test* dan *train* yang digunakan adalah 90:10. Proses ini dilakukan dengan bantuan *library* scikitlearn menggunakan *function* `train_test_split` dengan parameter `X` (fitur), `y` (target), `test_size` bernilai 0.1, `stratify` untuk memastikan bahwa set *train* dan *test* memiliki sampel proporsional dalam tiap kelas dalam *array* y. Parameter `shuffle` yang digunakan untuk mengacak data menggunakan *random_state* bernilai 100.

F Flowchart Training Model

Proses pelatihan dimulai dengan pembuatan `ModelCheckPoint`untuk menyimpan model dengan akurasi terbaik ke dalam format file h5 pada saat proses *training* dan `EarlyStopping` untuk menghentikan proses *training* saat sudah mencapai `val_loss` dengan nilai paling kecil. Kemudian, model dikompilasi dengan menggunakan menggunakan `categorical crossentropy` sebagai *loss function* dalam perhitungan nilai *loss* pada *multi-class classification*[42] dan Adam sebagai *optimizer* dengan *learning rate* sebesar 0.01. Adam merupakan salah satu metode optimasi *gradient descent* yang menggunakan dua momentum dalam melakukan perhitungan *learning rate* yang lebih adaptif[43]. Terakhir, model di-*training* menggunakan *training set* sebanyak 200 *epochs*. Alur proses *training* dapat

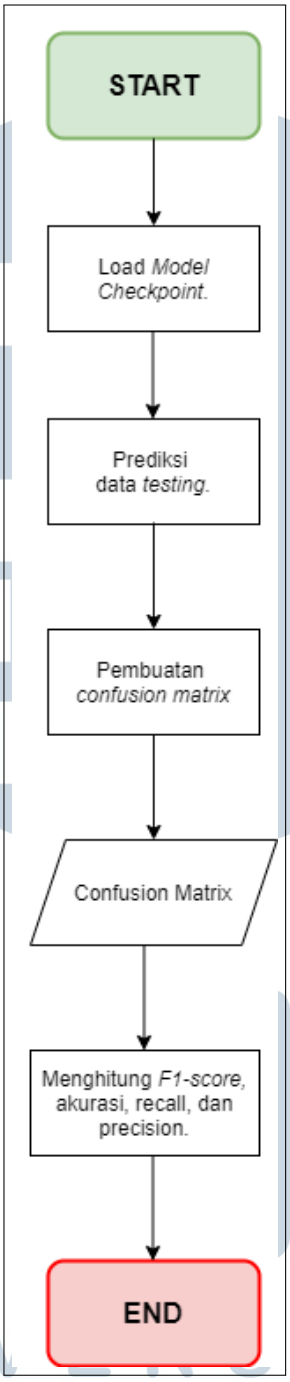
dilihat di Gambar 3.6.



Gambar 3.6. *Flowchart* pelatihan data

G Flowchart Evaluasi

Dalam pengujian sebuah model untuk mengetahui apakah model tersebut berjalan dengan baik, diperlukan proses evaluasi model. Pada penelitian ini, model dievaluasi dengan menghitung nilai akurasi, yang diperoleh dari *confusion matrix* hasil prediksi yang dilakukan pada set *testing* dari model yang disimpan oleh `ModelCheckPoint()` pada proses sebelumnya. Nilai kelas data testing dan nilai kelas hasil dari prediksi model akan digunakan untuk perhitungan metrik evaluasi yang dalam melakukan evaluasi model yang telah di-*training*. Perhitungan ini dilakukan dengan menggunakan *confusion matrix*, yang nantinya akan menampilkan *confusion matrix*, *accuracy*, *f1-score*, *precision score*, dan *recall score*. Alur proses untuk evaluasi model dapat dilihat pada Gambar 3.7.



Gambar 3.7. *Flowchart* evaluasi model

UNIVERSITAS MULTIMEDIA NUSANTARA