**DETEKSI EMOSI PADA MEDIA SOSIAL TWITTER MENGGUNAKAN LONG SHORT TERM MEMORY (LSTM) DAN FASTTEXT**



**Disusun oleh :**

**M. Alfa Riza**

**123170027**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK INDUSTRI**

**UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN”**

**YOGYAKARTA**

**2021**

# PERNYATAAN BEBAS PLAGIASI

Saya yang bertanda tangan di bawah ini :

Nama : M. Alfa Riza

NIM : 123170027

Fakultas/Prodi : Fakultas Teknik Industri/ Informatika.

Dengan ini saya menyatakan bahwa judul Proposal Tugas Akhir

**Deteksi Emosi pada Twitter Menggunakan Metode Long Short Term Memory (LSTM) dan Fasttext.**

Adalah hasil kerja keras saya sendiri dan benar bebas dari plagiasi kecuali cuplikan serta ringkasan yang terdapat didalamnya telah saya jelaskan sumbernya (Sitasi) dengan jelas. Apabila pernyataan ini terbukti tidak benar maka saya akan bersedia menerima sanksi sesuai peraturan Mendiknas RI No 17 tahun 2010 dan Peraturan Perundang-undangan yang berlaku.

Demikian surat pernyataan ini saya buat dengan penuh tanggungjawab.

Purwodadi, . . . . . . . . . . . . . .

Yang membuat pernyataan

…………………………..

NIM. 123170027

# DAFTAR ISI

HALAMAN SAMPUL i

[PERNYATAAN BEBAS PLAGIASI ii](#_Toc69588668)

[DAFTAR ISI iii](#_Toc69588669)

[DAFTAR TABEL v](#_Toc69588670)

[DAFTAR GAMBAR vi](#_Toc69588671)

[DAFTAR PERSAMAAN viii](#_Toc69588672)

[DAFTAR LAMPIRAN ix](#_Toc69588673)

[BAB I PENDAHULUAN 1](#_Toc69588674)

[1.1 Latar Belakang 1](#_Toc69588675)

[1.2 Perumusan Masalah 2](#_Toc69588676)

[1.3 Batasan Masalah 2](#_Toc69588677)

[1.4 Tujuan Penelitian 2](#_Toc69588678)

[1.5 Manfaat Penelitian 3](#_Toc69588679)

[1.6 Tahapan Penelitian 3](#_Toc69588680)

[1.6.1 Rencana dan Tahapan Penelitian 3](#_Toc69588681)

[1.6.2 Metode Pengembangan Sistem 3](#_Toc69588682)

[1.6.3 Pengujian Sistem 4](#_Toc69588683)

[1.6.4 Pengujian Penelitian 4](#_Toc69588684)

[1.6.5 Sistematika Penulisan 4](#_Toc69588685)

[BAB II TINJAUAN PUSTAKA 6](#_Toc69588686)

[2.1 Emosi 6](#_Toc69588688)

[2.2 Deteksi Emosi 6](#_Toc69588689)

[2.3 Media Sosial 6](#_Toc69588690)

[2.4 *Twitter* 6](#_Toc69588691)

[2.5 *Web Scraping* 7](#_Toc69588692)

[2.6 *Preprocessing* 7](#_Toc69588693)

[2.6.1 *Case Folding* 7](#_Toc69588694)

[2.6.2 *Remove Punctuation* 8](#_Toc69588695)

[2.6.3 *Remove Number* 8](#_Toc69588696)

[2.6.4 *Tokenizing* 8](#_Toc69588697)

[2.6.5 *Stopword Removal* 9](#_Toc69588698)

[2.6.6 *Stemming* 9](#_Toc69588699)

[2.7 Klasifikasi 12](#_Toc69588700)

[2.8 *Word Embedding* 12](#_Toc69588701)

[2.9 *Word2vec* 13](#_Toc69588702)

[2.9.1 *Continous bag of words* 13](#_Toc69588703)

[2.9.2 *Skip gram model* 14](#_Toc69588704)

[2.10 Glo-Ve 15](#_Toc69588705)

[2.11 *Fasttext* 16](#_Toc69588706)

[2.12 *Deep Learning* 18](#_Toc69588707)

[2.13 *Long Short Term Memory* 19](#_Toc69588708)

[2.14 *Confusion Matrix* 25](#_Toc69588709)

[2.15 Penelitian Sebelumnya 26](#_Toc69588710)

[BAB III METODOLOGI PENELITIAN DAN PENGEMBANGAN SISTEM 28](#_Toc69588711)

[3.1 Metodologi Penelitian 28](#_Toc69588713)

[3.1.1 Pengumpulan Data 28](#_Toc69588714)

[3.1.2 Labelling Data 28](#_Toc69588715)

[3.1.3 *Preprocessing* 29](#_Toc69588716)

[3.1.4 Pengujian Arsitektur 36](#_Toc69588717)

[3.1.5 *Word embedding* 36](#_Toc69588718)

[3.1.6 LSTM 41](#_Toc69588719)

[3.1.7 Pengujian 50](#_Toc69588720)

[3.2 Metodologi Pengembangan Sistem 51](#_Toc69588721)

[3.2.1 Analisis Kebutuhan Sistem 52](#_Toc69588722)

[3.2.2 Kebutuhan Fungsional 53](#_Toc69588723)

[3.2.3 Kebutuhan Non-fungsional 53](#_Toc69588724)

[3.2.4 Proses Desain 53](#_Toc69588725)

[3.2.5 Perancangan Sistem 54](#_Toc69588726)

[3.2.6 Perancangan Pengujian 62](#_Toc69588727)

[3.3 Jadwal Penelitian 63](#_Toc69588728)

[DAFTAR PUSTAKA x](#_Toc69588729)

[LAMPIRAN xiv](#_Toc69588730)

# DAFTAR TABEL

[Tabel 2.1 *Case Folding* 8](#_Toc69578513)

[Tabel 2.2 *Remove Punctuation* 8](#_Toc69578514)

[Tabel 2.3 *Remove number* 8](#_Toc69578515)

[Tabel 2.4 *Tokenizing* 8](#_Toc69578516)

[Tabel 2.5 *Stopword Removal* 9](#_Toc69578517)

[Tabel 2.6 Kata awalan akhiran dilarang 10](#_Toc69578518)

[Tabel 2.7 Aturan peluruhan kata dasar 10](#_Toc69578519)

[Tabel 2.8 Aturan peluruhan kata dasar (lanjutan) 11](#_Toc69578520)

[Tabel 2.9 *Stemming* 12](#_Toc69578521)

[Tabel 2.10 Ilustrasi *confusion matrix* 24](#_Toc69578522)

[Tabel 2.11 Penelitian sebelumnya 26](#_Toc69578523)

[Tabel 2.12 Penelitian sebelumnya (lanjutan) 27](#_Toc69578524)

[Tabel 3.1 Ilustrasi *Labelling* 29](#_Toc69578525)

[Tabel 3.2 Contoh *preprocessing* 36](#_Toc69578526)

[Tabel 3.3 Contoh Hasil *Word2vec* 38](#_Toc69578527)

[Tabel 3.4 Contoh hasil *fasttext* 41](#_Toc69578528)

[Tabel 3.5 Hasil *forget gate* (*ft*) tiap orde 48](#_Toc69578529)

[Tabel 3.6 Hasil *input gate* (*it*) tiap orde 48](#_Toc69578530)

[Tabel 3.7 Hasil Kandidat baru () tiap orde 48](#_Toc69578531)

[Tabel 3.8 Hasil *Cell state* (*Ct*) tiap orde 48](#_Toc69578532)

[Tabel 3.9 Hasil *Output gate* (*Ot*) tiap orde 48](#_Toc69578533)

[Tabel 3.10 Hasil *output final* (*ht*) tiap orde 49](#_Toc69578534)

[Tabel 3.11 Pengujian 50](#_Toc69578535)

[Tabel 3.12 Pengujian (lanjutan) 51](#_Toc69578536)

[Tabel 3.13 Kebutuhan perangkat keras (*hardware*) 53](#_Toc69578537)

[Tabel 3.14 Kebutuhan perangkat lunak (*software*) 53](#_Toc69578538)

[Tabel 3.15 Tabel *users* 59](#_Toc69578539)

[Tabel 3.16 Tabel *rekaps* 59](#_Toc69578540)

[Tabel 3.17 Detail perancangan pengujian sistem 63](#_Toc69578541)

[Tabel 3.18 Jadwal penelitian 63](#_Toc69578542)

[Tabel 3.19 Jadwal Penelitian (lanjutan) 64](#_Toc69578543)

# DAFTAR GAMBAR

[Gambar 2.1 Hubungan antar kata *word2vec* 13](#_Toc69155681)

[Gambar 2.2 Ilustrasi CBOW Sumber: (Rong, 2016) 14](#_Toc69155682)

[Gambar 2.3 Ilustrasi *skip gram model* Sumber: (Rong, 2016) 15](#_Toc69155683)

[Gambar 2.4 Vektor kata semangat Sumber: (Alfariqi et al., 2020) 17](#_Toc69155684)

[Gambar 2.5 Cara kerja *Hierarchical Softmax* 17](#_Toc69155685)

[Gambar 2.6 Arsitektur *fasttext* Sumber: (Joulin et al., 2016) 18](#_Toc69155687)

[Gambar 2.7 Arsitektur *deep learning* Sumber: (Xing and Du, 2019) 18](#_Toc69155688)

[Gambar 2.8 Arsitektur LSTM 19](#_Toc69155689)

[Gambar 2.9 Alur informasi pada *forget gate* 20](#_Toc69155691)

[Gambar 2.10 Proses *input gate layer* 21](#_Toc69155693)

[Gambar 2.11 Langkah mendapatkan nilai *cell state* baru 22](#_Toc69155695)

[Gambar 2.12 *Output gate* 23](#_Toc69155697)

[Gambar 3.1 Metodologi Penelitian 28](#_Toc69155699)

[Gambar 3.2 Flowchart *Preprocessing* 29](#_Toc69155700)

[Gambar 3.3 Flowchart *case folding* 30](#_Toc69155701)

[Gambar 3.4 Flowchart *remove punctuation* 31](#_Toc69155702)

[Gambar 3.5 Flowchart *remove number* 33](#_Toc69155703)

[Gambar 3.6 Flowchart *tokenizing* 34](#_Toc69155704)

[Gambar 3.7 Flowchart *stopword removal* 35](#_Toc69155705)

[Gambar 3.8 Flowchart *stemming* 36](#_Toc69155706)

[Gambar 3.9 Flowchart *word2vec* 38](#_Toc69155707)

[Gambar 3.10 Visualisasi *word2vec* 38](#_Toc69155708)

[Gambar 3.11 Flowchart Glo-Ve 39](#_Toc69155709)

[Gambar 3.12 Flowchart fasttext 40](#_Toc69155710)

[Gambar 3.13 Visualisasi *fasttext* 41](#_Toc69155711)

[Gambar 3.14 *Flowchart* LSTM 42](#_Toc69155712)

[Gambar 3.15 Metodologi Pengembangan Sistem 52](#_Toc69155713)

[Gambar 3.16 Arsitektur sistem 54](#_Toc69155714)

[Gambar 3.17 DFD level 0 55](#_Toc69155715)

[Gambar 3.18 DFD level 1 56](#_Toc69155716)

[Gambar 3.19 DFD level 2 proses login dan register 57](#_Toc69155717)

[Gambar 3.20 DFD level 2 proses *preprocessing* dan deteksi emosi dengan model LSTM 57](#_Toc69155718)

[Gambar 3.21 DFD level 2 proses rekap hasil 58](#_Toc69155719)

[Gambar 3.22 *Entity Relationship Diagram* (ERD) 58](#_Toc69155720)

[Gambar 3.23 Relasi Antar Tabel 59](#_Toc69155721)

[Gambar 3.24 Halaman Home 60](#_Toc69155722)

[Gambar 3.25 Halaman Hasil 61](#_Toc69155723)

[Gambar 3.26 Halaman Login 61](#_Toc69155724)

[Gambar 3.27 Halaman Register 62](#_Toc69155725)

[Gambar 3.28 Halaman Hasil 63](#_Toc69155726)

# DAFTAR PERSAMAAN

[Persamaan 2.1 Bentuk *Matrix Co-occurent* 15](#_Toc69579628)

[Persamaan 2.2 Rumus *Soft Constraints* 15](#_Toc69579629)

[Persamaan 2.3 Rumus *Cost Function* 16](#_Toc69579630)

[Persamaan 2.4 Fungsi Pembobotan 16](#_Toc69579631)

[Persamaan 2.5 Representasi Vektor 17](#_Toc69579632)

[Persamaan 2.6 *Forget Gate* 20](#_Toc69579633)

[Persamaan 2.7 *Input Gate* 22](#_Toc69579634)

[Persamaan 2.8 Kandidat Baru 22](#_Toc69579635)

[Persamaan 2.9 *Cell State* Orde t 23](#_Toc69579636)

[Persamaan 2.10 *Output Gate* 24](#_Toc69579637)

[Persamaan 2.11 Hasil *Output* Keseluruhan 24](#_Toc69579638)

[Persamaan 2.12 Fungsi *Sigmoid* 24](#_Toc69579639)

[Persamaan 2.13 Fungsi *Tanh* 24](#_Toc69579640)

[Persamaan 2.14 Rumus *Accuracy* 25](#_Toc69579641)

[Persamaan 2.15 Rumus *Precision* 26](#_Toc69579642)

[Persamaan 2.16 Rumus *Recall* 26](#_Toc69579643)

# DAFTAR LAMPIRAN

# BAB I PENDAHULUAN

## Latar Belakang

Deteksi emosi penting diberbagai bidang seperti pendidikan, bisnis, rekrukmen karyawan. Emosi dapat dideteksi melalui suara, ekspresi wajah, gerakan tangan, gerakan tubuh, detak jantung, tekanan darah, teks (Consoli, 2009). Teks relatif lebih mudah digunakan untuk deteksi emosi karena emosi dipicu oleh situasi tertentu serta mengsgambarkan emosi dari situasi tanpa kata-kata (Bata et al., 2015). Model emosi yang sangat populer adalah model emosi Ekman. Model Ekman membagi emosi menjadi 6 label emosi yaitu bahagia, marah, takut, jijik, sedih, dan terkejut (Ekman, 1999), label-label ini bersifat universal pada budaya yang berbeda.

Saat ini media sosial menjadikan penggunanya cenderung berekspresi emosi melalui postingan teks. Salah satu media sosial yang memiliki laju pertumbuhan pengguna tertinggi di indonesia adalah *twitter*, pengguna aktif *twitter* di Indonesia menempati posisi ketiga di Asia Pasifik dari tahun 2012 sampai tahun 2018 (Saputri et al., 2019). Maka dari itu *twitter* merupakan media sosial yang dipilih untuk dijadikan data penelitian ini.

Penelitian yang berkaitan dengan deteksi emosi pernah dilakukan oleh Fanesya(2019) menggunakan metode *Naïve Bayes* dengan kombinasi fitur N-gram, namun penelitian ini hanya menghasilkan akurasi tertinggi 55,54% (Fanesya et al., 2019). Penelitian yang dilakukan oleh Rohman (2019) menghasilkan akurasi kurang maksimal sebesar 55,54% dari 34.872 kata untuk lexicon dengan emolex, sedangkan NLP menghasilkan hanya akurasi 61,53% (Rohman et al., 2019). Rohman (2019) melakukan penelitian menggunakan Term Frequency-Inverse Document Frequency(TF-IDF) hanya menghasilkan akurasi 59% dari 6 emosi, namun jika 1 emosi saja dan menghasilkan akurasi cukup baik sebesar 87,23% (Rohman et al., 2019). Penelitian yang dilakukan oleh Ardiada(2019) menggunakan *Support Vector Machine*(SVM) dan *K-Nearest Neighbour*. SVM menghasilkan *precision* 45,64%, *recall* 50,20%, dan akurasi cukup bagus sebesar yaitu 81,04%, sedangkan untuk KNN menghasilkan *precision*, *recall*, dan akurasi yang kurang maksimal yaitu *precision* 34,21%, *recall* 45,95%, akurasi 79,70% (Ardiada et al., 2019). Penelitian Haryadi (2019) membandingkan antara metode SVM, Long Short Term Memory (LSTM) dan Nested Long Short Term Memory (NLSTM). SVM menghasilkan akurasi, *precision*, *recall* terendah yaitu akurasi 98,679%, *precision* 98,53%, *recall* 98,22%, sedangkan akurasi tertinggi didapatkan oleh NLSTM dengan *word embedding word2vec* menghasilkan akurasi 99,167%, *precision* 99,21%, *recall* 98,83%, namun *precision* dan *recall* tertinggi didapatkan oleh LSTM dengan *word embedding word2vec* menghasilkan *precision* 99,22%, *recall* 98,86% sedangkan akurasi tidak jauh berbeda dengan NLSTM yaitu 99,154% (Haryadi dan Kusuma, 2019). Penelitian Dwi (2020) menggunakan *Long Short Term Memory* (LSTM) dengan *word embedding* Glo-Ve menghasilkan akurasi hanya 50% saja, *precision* hanya33%, *recall* hanya 38%, karena dalam proses *training* terjadi *underfitting* (Dwi et al., 2020). Penelitian sebelumnya menunjukkan NLSTM memiliki akurasi yang paling baik, namun LSTM menunjukkan *precision* dan *recall* yang paling baik, hal ini berarti LSTM memiliki performa yang lebih baik. LSTM efektif untuk mengolah data sekuensial seperti teks, karena memiliki *memory cell* (Miedema, 2018).

LSTM dengan *word embedding word2vec* banyak digunakan karena *word2vec* mampu mengenali kata-kata yang mirip, *word embedding* Glo-Ve juga banyak digunakan karena mampu memperoleh hubungan semantik antarkata dengan *co-occurrence* (Nurdin et al., 2020), namun *word2vec* dan Glo-Ve tidak dapat menangani kesalahan penulisan sehingga menurunkan akurasi (Lim et al., 2020). Hal ini terjadi karena *word2vec* dan Glo-Ve tidak dapat merepresentasikan vektor dari kata yang belum pernah ditemui, masalah ini dinamakan *out of vocabulary*(OOV), *word embedding fasttext* dapat menangani masalah OOV dengan memperhatikan informasi *subword* menggunakan *n-gram* (Bojanowski et al., 2017). Sehingga penelitian ini menggunakan metode *Long Short Term Memory* (LSTM) untuk mendapatkan *precision*, *recall* yang baik dan menggunakan *word embedding fasttext* agar mendapatkan akurasi yang lebih baik.

Penelitian ini akan menerapkan *Long Short Term Memory* (LSTM) dengan *word embedding* *fasttext* untuk mendeteksi emosi dengan 6 label emosi yaitu, bahagia, marah, takut, jijik, sedih, dan terkejut. Penelitian ini akan menggunakan data yang bersumber dari *twitter*. Penelitian ini diharapkan dapat menghasilkan sebuah aplikasi yang dapat mengklasifikasikan emosi menjadi bahagia, marah, takut, jijik, sedih, dan terkejut yang memiliki akurasiyang lebih baik, sehingga hasil deteksi emosi tersebut dapat berguna dalam mengambil keputusan di berbagai bidang seperti pendidikan, bisnis, rekrukmen karyawan.

## Perumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, maka didapatkan rumusan masalah sebagai berikut:

Evaluasi tingkat akurasi, *precision*, *recall* algoritma *Long Short Term Memory* (LSTM) dengan *word embedding fasttext* untuk mendeteksi emosi pada media sosial *twitter*.

## Batasan Masalah

Pada penelitian ini, permasalahan dibatasi sebagai berikut :

1. Data penelitian didapatkan dari media sosial *twitter* dari tahun 2018 sampai tahun 2021.
2. Data penelitian hanya menggunakan Bahasa Indonesia.
3. Pada penelitian ini emosi yang diklasifikasi menjadi 6 label sesuai model Ekman yaitu bahagia, marah, takut, jijik, sedih, dan terkejut.
4. Data penelitian diambil dari akun beberapa *influencer* di Indonesia sebagai data utama dan data trending selama 5 April 2021 sampai dengan 11 April 2020 sebagai data pendukung.

## Tujuan Penelitian

Tujuan penelitian ini adalah:

Menghitung tingkat akurasi, *precision*, *recall* algoritma *Long Short Term Memory*(LSTM) dengan *word embedding fasttext* untuk deteksi emosi pada media sosial *twitter*.

## Manfaat Penelitian

Manfaat yang didapatkan dari penelitian ini adalah dapat mendeteksi emosi pada media sosial *twitter* sehingga dapat digunakan mengambil keputusan di berbagai bidang.

## Tahapan Penelitian

Tahapan penelitian yang dipakai pada penelitian ini sebagai berikut :

### Rencana dan Tahapan Penelitian

Rencana dan tahapan penelitian sebagai berikut:

1. Studi Literatur

Studi literatur dilakukan dengan cara mencari dan menghimpun data ataupun sumber-sumber pustaka yang dapat mendukung penelitian serta memberikan informasi untuk menyelesaikan permasalahan pada penelitian ini yang bersumber dari jurnal, artikel, skripsi, tesis dan paper yang dapat dipertanggungjawabkan.

1. Pengumpulan Data

Data yang akan digunakan bersumber dari media sosial *twitter*, data diambil menggunakan teknik *web scrapping* dan data tersebut diberi label secara manual.

1. Analisis Sistem

Analisis sistem ini merupakan penganalisaan terhadap kebutuhan dalam pembuatan sistem. Sehingga dapat diketahui kebutuhan-kebutuhan apa saja yang diperlukan dalam pembuatan sistem pada penelitian ini.

1. Perancangan Sistem  
   Pada bagian perancangan ini akan melakukan pemodelan terhadap sistem yang akan dibuat berdasarkan hasil analisis yang telah dilakukan sebelumnya.
2. Implementasi Perangkat Lunak  
   Tahap ini akan mengimplementasikan sistem yang sudah dirancang sebelumnya.
3. Pengujian dan Analisis

Pada tahap ini akan dilakukan pengujian terhadap implementasi yang telah dikerjakan. Kemudian akan dilakukan analisis berdasarkan hasil dari pengujian.

1. Kesimpulan dan Saran

Pada penelitian ini akan diberikan kesimpulan dari hasil penelitian yang telah dilakukan dan akan menyertakan saran yang dapat dikembangkan untuk penelitian selanjutnya sehingga mendapatkan hasil yang lebih baik.

### Metode Pengembangan Sistem

Metode pengembangan sistem yang digunakan pada penelitian ini adalah *prototyping*. *Prototyping* digunakan pada penelitian ini untuk mengembangkan model menjadi final, sehingga sistem yang dikembangkan lebih cepat dan lebih hemat dibandingkan metode pengembangan sistem lainnya. *Prototyping* memiliki beberapa tahapan, yaitu (Pressman, 2015) :

1. *Communication*

Tahap awal pengembangan sistem melakukan komunikasi dan kolaborasi dengan pelanggan atau pemangku kepentingan untuk memahami tujuan dan kebutuhan sistem sehingga membantu menentukan fitur dan fungsi sistem.

1. *Planning*

Tahap *planning* berguna untuk membantu pengembang saat pembuatan sistem. Tahap *planning* mendeskripsikan tugas teknis, resiko yang mungkin terjadi, sumber daya yang dibutuhkan, hasil produk, dan jadwal pengerjaan sistem.

1. *Model*

Tahap ini pengembang membuat *model* dari sistem yang akan dibuat sehingga pengembang dapat memahami kebutuhan sistem dan design yang sesuai untuk menunjang kebutuhan tersebut.

1. *Construction*

Tahap *construction*, pengembang memulai pembuatan sistem tahap sebelumnya, selain itu tahap ini juga melakukan pengujian atau *testing* untuk menemukan kesalahan pada pembuatan sistem.

1. *Deployment*

Tahap ini sistem yang telah dibuat dikirimkan kepada pelanggan baik semua fitur selesai maupun sebagian untuk mendapatkan evaluasi produk dan memberikan *feed back* berdasarkan evaluasi.

### Pengujian Sistem

Pengujian sistem yang digunakan pada penelitian ini adalah metode *black box testing*, yaitu pengujian yang menekankan fungsionalitas tanpa mengetahui *coding* dari sistem tersebut. *Black box testing* bertujuan untuk mengukur kinerja dari sistem yang telah dibangun.

### Pengujian Penelitian

Pengujian penelitian yang digunakan pada penelitian ini adalah *confusion matrix*. *Confusion matrix* digunakan untuk mengukur tingkat performa model klasifikasi dengan menghitung nilai *precision*, *recall*, *accuracy*. Terdapat empat istilah pada *confusion matrix* yang merepresentasikan hasil, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), *False Negative* (FN).

### Sistematika Penulisan

Sistematika penulisan yang digunakan dalam menyusun laporan penelitian ini adalah sebagai berikut :

**Bab I Pendahuluan**

Pada bagian ini membahas masalah kehidupan, *research gap*, masalah metode, dan solusi masalah metode. Bab ini terdiri dari latar belakang masalah, perumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, tahapan penelitian, dan sistematika penulisan.

**Bab II Tinjauan Pustaka**

Tinjauan pustaka memuat tentang uraian dan pembahasan tentang teori, konsep, model, metode, sistem, atau analisis pustaka ilmiah yang berkaitan dan bersifat fundamental terhadap masalah yang akan diselesaikan. Teori-teori yang sesuai dengan penelitian ini antara lain media sosial, deteksi emosi, *deep learning*, *Long Short Term Memory*(LSTM), *word embedding*, *fasttext.*

**Bab III Metodologi Penelitian dan Pengembangan Sistem**

Pada bagian ini akan membahas mengenai jenis penelitian, strategi dan rancangan penelitian, analisis kebutuhan sistem, metode penelitian, akuisisi data, pengolahan data, perancangan sistem, simulasi komputasi, pengembangan sistem, pengujian sistem.

**Bab IV Hasil dan Pembahasan Implementasi**

Pada bab ini akan merealisasikan metodologi penelitian menjadi produk akhir, selain itu bab ini berisi hasil pelaksanaan dan pengujian metode penelitian yang disajikan secara sistematis dan logis. Pembahasan implementasi juga membahas mengenai hasil penelitian yang diperoleh untuk menjawab masalah penelitian. Selain itu berisi pemahaman baru yang didapatkan dari hasil penelitian.

**Bab V Penutup**

Pada bagian ini berisi hasil penelitian untuk membuktikan hasil mampu menjawab rumusan masalah sehingga dapat ditarik kesimpulan berdasarkan tujuan. Saran juga dibahas pada bagian ini sehingga dapat dikembangkan pada penelitian lebih lanjut.

# BAB II TINJAUAN PUSTAKA



## Emosi

Emosi menurut Jabreel dan Moreno(2019) adalah kunci perasaan dan pikiran orang (Jabreel dan Moreno, 2019). Menurut Graind (2019) perasaan intens yang diarahkan pada sesuatu atau seseorang sebagai respon terhadap peristiwa internal maupun eksternal bagi individu digambarkan sebagai emosi (Gaind et al., 2019). Emosi bersifat umum dan penting dalam semua aspek kehidupan manusia.

Variasi jenis emosi dapat didasarkan kepribadian, jenis kelamin, lokasi, etnis, budaya, situasi, di samping banyak parameter psikologis, sosial dan individu lainnya (Rohman et al., 2020). Model emosi yang sangat populer adalah model emosi Ekman. Model Ekman membagi emosi menjadi 6 label emosi yaitu bahagia, marah, takut, jijik, sedih, dan terkejut (Ekman, 1999), sebenarnya banyak model emosi yang lain namun pada model ekman label-label emosi bersifat universal pada budaya yang berbeda.

## Deteksi Emosi

Deteksi Emosi merupakan bagian dari area yang lebih luas dari komputasi afektif bertujuan untuk memungkinkan komputer mengenali dan mengekspresikan emosi (Picard, 1997). Pendekatan klasifikasi teks digunakan untuk melakukan deteksi emosi (Fanesya et al., 2019). Penelitian deteksi emosi memiliki banyak kegunaan dalam mengambil keputusan seperti di bidang pendidikan untuk memprediksi keadaan emosi siswa dengan platform e-learning sehingga dapat digunakan untuk meningkatkan kualitas belajar (Daouas dan Lejmi, 2018), bidang bisnis digunakan untuk review kualitas produk (Nurlaila et al., 2017), selain itu deteksi emosi juga bermanfaat untuk rekrutmen karyawan, koseling, dan prosfil psikologis karena merupakan bagian dari prediksi kepribadian (Dandannavar et al., 2018). Emosi dapat dideteksi melalui suara, ekspresi wajah, gerakan tangan, gerakan tubuh, detak jantung, tekanan darah, teks (Consoli, 2009).

## Media Sosial

Media sosial merupakan platform komunikasi, interaksi, dan berbagi informasi yang populer melalui internet (Wilson et al., 2012). Media sosial dapat digunakan oleh penggunanya untuk berekspreksi (Sari dan Haranto, 2019). Pada media sosial, setiap acara, berita atau aktivitas di seluruh dunia dapat dibagikan, didiskusikan, diposting dan dikomentari oleh jutaan orang (Gaind et al., 2019). Teks, gambar, suara , dan video dapat dibagikan di media sosial (Kotler et al., 2012). Berdasarkan penjelasan tersebut dapat disimpulkan bahwa media sosial merupakan platform berbagi informasi, acara, aktifitas berupa teks, gambar, suara, video melalui internet untuk mengekspresikan penggunanya, dimana informasi, acara, aktifitas tersebut dapat didiskusikan, dan dikomentari oleh jutaan orang. Contoh media sosial seperti *facebook*, *twitter*, *instagram*, *line*.

## Twitter

Twitter adalah salah satu media sosial dimana pengguna dapat membaca dan berbagi pesan singkat dengan jumlah maksimal 280 karakter, *twitter* termasuk media sosial dengan kategori *microblogging* (Setiawan dan Setyohadi, 2017), hal ini yang membuat *twitter* berbeda dengan media sosial yang lain. *Twitter* didirikan oleh Jack Dorsey sejak 2006, *twitter* kini telah berkembang menjadi media sosial populer (Juwiantho et al., 2020). *Twitter* memiliki laju pertumbuhan pengguna tertinggi di indonesia, pengguna aktif *twitter* di Indonesia menempati posisi ketiga di Asia Pasifik dari tahun 2012 sampai tahun 2018 (Saputri et al., 2019). Oleh karena itu *twitter* dipilih menjadi objek pada penelitian ini.

## Web Scraping

*Web scraping* atau *screen scraping* merupakan proses pengambilan dan analisis sebuah dokumen semi-terstruktur tertentu yang diambil dari internet, biasanya berupa halaman-halaman web dalam bahasa *markup* seperti HTML, XHTML untuk digunakan bagi kepentingan lain (Turland, 2010).

*Web scaping* pertama kali dilakukan dengan cara manual, yaitu menyalin data dari website ke komputer kita atau penyimpanan lokal. Namun, cara ini kurang efektif jika digunakan untuk mengambil data dalam jumlah banyak, karena membutuhkan tenaga dan waktu yang banyak. Cara lain *web scraping* dapat dilakukan dengan *coding*, aplikasi ataupun *extension browser*. Pada penelitian ini menggunakan *coding* untuk *web scraping* dengan memanfaatkan *twitter* dan *library* dari bahasa pemprograman *python*.

Menurut Josi dkk (2014) langkah-langkah dalam melakukan *web scraping* ada beberapa tahapan, sebagai berikut (Josi et al., 2014) :

1. *Create scraping template* : mempelajari dokumen HTML dari website yang akan diambil informasinya untuk tag HTML yang mengapit informasi yang akan diambil.
2. *Explore site navigation* : mempelajari teknik navigasi pada website yang akan diambil informasinya untuk ditirukan pada aplikasi *web scraper* yang akan dibuat.
3. *Automate navigation and extraction* : berdasarkan informasi yang didapat pada tahap dan tahap 2, aplikasi *web scaper* dibuat untuk mengotomatisasi pengambilan informasi dari website yang ditentukan.
4. *Extracted data and package* : informasi yang didapat dari tahap 3 disimpan dalam format data tertentu, seperti *database*, csv.

## Preprocessing

*Preprocessing* data seringkali mempengaruhi kinerja dari *machine learning*, sehingga lebih efektif (S. B. Kotsiantis, 2006). *Preprocessing* berguna untuk membersihkan data dari derau dan bagian tidak informatif yang tidak dibutuhkan, sehingga teks siap untuk diklasifikasi (Haddi et al., 2013). *Preprocessing* akan menjadikan data bersih sehingga proses pembuatan vektor kata dan klasifikasi lebih akurat (Nurrohmat dan SN, 2019).

### Case Folding

*Case Folding* adalah mengubah seluruh huruf dalam teks tersebut menjadi *lower case* atau huruf kecil semua (Salam et al., 2018), dengan huruf yang diterima dari “a” sampai “z”. *Case folding* bertujuan agar terdapat standarisasi dalam penulisan. Misalnya terdapat data “Emosi Saya” setelah proses *case folding* menjadi “emosi saya”. Contoh lain dapat dilihat pada Tabel 2.1.

#### Tabel 2.1 Case Folding

|  |  |
| --- | --- |
| Data sebelum *case folding* | “Suaranya bagus, apalagi kalau dIiringi gitaR” |
| Data setelah *case folding* | “suaranya bagus, apalagi kalau diiringi gitar” |

### Remove Punctuation

*Remove punctuation* adalah menghapus tanda baca pada teks dengan tujuan mengurangi beban pemprosesan klasifikasi karena dianggap tidak penting dan termasuk *delimiter*, contoh tanda baca yang yang dihapus titik (.), koma(,), tanda tanya (?), *slash* (/), *hastag* (#), tanda seru (!) dan lain-lain. Misal terdapat data “sabar pak?” setelah melalui *remove punctuation* menjadi “sabar pak” Contoh lain dapat dilihat pada Tabel 2.2.

#### Tabel 2.2 Remove Punctuation

|  |  |
| --- | --- |
| Data sebelum *remove punctuation* | “suaranya bagus !!!, apalagi kalau diiringi gitar” |
| Data setelah *remove punctuation* | “suaranya bagus apalagi kalau diiringi gitar” |

### Remove Number

*Remove number* adalah menghapus angka pada suatu teks, penghapusan ini sebab angka dianggap tidak memiliki arti dan termasuk *delimiter*, mirip seperti *remove punctuation* hanya berbeda pada objek yang dihapus. Misalnya terdapat data “dokumen2 yang dikumpulkan” setelah melalui proses *remove number* menjadi “dokumen yang dikumpulkan”.Contoh lain dapat dilihat pada Tabel 2.3.

#### Tabel 2.3 Remove number

|  |  |
| --- | --- |
| Data sebelum remove number | “barang yang dijual bagus2” |
| Data setelah remove number | “barang yang dijual bagus” |

### Tokenizing

*Tokenizing* atau tokenisasiadalah pemotongan kalimat berdasarkan tiap-tiap kata penyusunnya, biasanya pemotongan berdasarkan *whitespace* seperti spasi, tab, dan enter. Tiap-tiap kata hasil *tokenizing* disebut token (Juwiantho et al., 2020). Misalnya terdapat data “ada saya semua bahagia” setelah melalui proses *tokenizing* akan menghasilkan 4 token, yaitu : “ada”, “saya”, “semua”, “bahagia”. Contoh lain dapat dilihat pada Tabel 2.4.

#### Tabel 2.4 *Tokenizing*

|  |  |
| --- | --- |
| Sebelum proses *Tokenizing* | konten yang bermutu namun sayang suaranya kurang jelas |
| Setelah proses *Tokenizing* | [‘konten’,’yang’,’bermutu’,’namun’,’sayang’,’suaranya’,  ’kurang’,’jelas’] |

### Stopword Removal

*Stopword removal* atau *filtering* merupakan proses untuk menghilangkan kata-kata yang tidak memiliki arti atau makna, namun tidak akan mengubah makna dari komentar tersebut (Juwiantho et al., 2020). *Stopword removal* akan mengurangi ukuran indeks, waktu pemprosesan klasifikasi, dan *noise* dari suatu data. Biasanya *stopword* berupa kata ganti orang dan kata hubung, seperti “aku”, “kamu”, “kita”, “dan”, “atau”. Contoh proses *stopword removal* yang datanya berasal dari proses *tokenizing* sebelumnya dapat dilihat pada Tabel 2.5.

#### Tabel 2.5 Stopword Removal

|  |  |
| --- | --- |
| Sebelum Stop Removal | [‘konten’,’yang’,’bermutu’,’namun’,’sayang’,’suaranya’,  ’kurang’,’jelas’] |
| Setelah Stop Removal | [‘konten’,’bermutu’,’sayang’,’suaranya’,’kurang’,’jelas’] |

### Stemming

*Stemming* atau *lemmatization* merupakan proses untuk mentransformasi kata kerja yang berimbuhan pada suatu dokumen menjadi kata dasar (*root word*), dengan menghapus awalan, akhiran maupun sisipan . *Stemming* bertujuan untuk mengurangi varian kata dengan makna yang hampir sama pada suatu dokumen, serta meningkatkan performa pada tahap *information retrieval*. Pada penelitian ini akan menggunakan *library* sastrawi, karena data pada penelitian ini menggunakan bahasa Indonesia. *Library* sastrawi menerapkan algoritma Nazief dan Adriani. Berikut langkah-langkah algoritma Nazief dan Adriani dalam *stemming* sebagai berikut (Nazief dan Adriani, 2005) :

1. Pertama – tama, periksa kata dengan kamus kata, jika pemeriksaan ada kata tersebut akan dianggap kata dasar maka proses berhenti, jika tidak maka akan lanjut proses ke-2.
2. Pada tahap ke-2, hapus *inflectional suffixes* yaitu (“-kah”, “-lah”, “-tah”, “-pun”), kemudian hapus *inflectional possessive pronoun suffixes* seperti (“-ku”, “-mu”, “-nya”), kemudian data hasil *infectional suffixes* dan *inflectional possessive pronoun suffixes* dicek pada kamus kata, jika hasilnya ada maka proses akan berhenti, jika tidak maka lanjut pada proses ke-3. Contoh ada data “hasilnyapun” setelah dihapus *inflectional suffixes* menjadi “hasilnya”, kemudian data dihapus *inflectional possessive pronoun suffixes* menjadi “hasil”.
3. Tahap ke-3, hapus *derivational suffixes* (imbuhan turunan) yaitu (“-i”, “-kan”, “-an”), setelah proses penghapusan *derivational suffixes* (“-i”, “-kan”, “-an”)kata akan dicari pada kamus kata, jika ada maka proses berhenti, namun jika tidak kembalikan huruf yang dihapus sebelum *derivational suffixes* (“-i”, “-kan”, “-an”), kemudian lanjut pada proses ke-4. Contoh data “menggabungkan” setelah penghapusan *derivational suffixes* (“-i”, “-kan”, “-an”)menjadi “menggabung”, karena tidak ada dalam kamus, maka hapus huruf sebelum *derivational suffixes* (“-i”, “-kan”, “-an”)menjadi “menggabun”, setelah dicek tidak ada dalam kamus, maka kembalikan kata yang dihapus tadi menjadi “menggabung” dan lanjut pada proses ke-4.
4. Tahap ke-4, hapus *derivational prefixes* yaitu (“be-“, “di-“, “ke-“, “me-“, “pe-“, “se-“, “te-“). Jika pada tahap ke-3 ada *derivational suffixes* (“-i”, “-kan”, “-an”)yang dihapus maka ke langkah 4a, jika tidak ada yang dihapus ke langkah 4b.
5. kata dicek pada Tabel 2.6, jika ada maka proses berhenti, jika tidak maka lanjut ke tahap 4b.

#### Tabel 2.6 Kata awalan akhiran dilarang

|  |  |
| --- | --- |
| Awalan | Akhiran |
| be- | -i |
| di- | -an |
| ke | -i, -kan |
| me- | -an |
| se- | -i, -kan |
| te- | -an |

**Sumber: (Nazief dan Adriani, 2005)**

1. Untuk i sama dengan 1 sampai 3, tentukan tipe awalan kemudian hapus awalan. Jika kata dasar belum ditemukan juga lakukan tahap ke-5, jika sudah maka proses berhenti. Aturan peluruhan kata dapat dilihat pada Tabel 2.7 dan Tabel 2.8.

#### Tabel 2.7 Aturan peluruhan kata dasar

|  |  |  |
| --- | --- | --- |
| Aturan | Awalan | Akhiran |
| 1 | berV . . . | ber-V . . . | be-rV |
| 2 | belajar | bel-ajar |
| 3 | berClerC2 | be-ClerC2 . . dimana C1!= {‘r’|’l’} |
| 4 | terV. . . | ter-V . . | te-rV . . |
| 5 | terCer. . . | terCer. . . dimana C!=’r’ |
| 6 | teClerC2 | te-CleC2 . . . dimana C1!=’r’ |
| 7 | me{I|r|w|y}V. . . | me-{I|r|w|y}V. . . |
| 8 | mem{b|fv}. . . | mem-{b|f|v}. . . |
| 9 | mempe. . . | m-pe. . |
| 10 | mem{r|V|V}. . . | me-m{r|V|V}. . . | me-p{r|V|V}. . . |
| 11 | men{c|d|j|z}. . . | Men-{c|d|j|z}. . . |
| 12 | menV. . . | Me-nV. . . | me-tV. . . |
| 13 | meng{g|h|q|k}. . . | Meng-{g|h|q|k}. . . |
| 14 | mengV. . . | meng-V. . . | meng-kV. . . |
| 15 | mengeC. . . | meng-C. . . |

#### Tabel 2.8 Aturan peluruhan kata dasar (lanjutan)

|  |  |  |
| --- | --- | --- |
| 16 | menyV. . . | me-ny. . . | men-sV. . . |
| 17 | memV. . . | mem-pV. . . |
| 18 | pe{w|y}V. . . | pe-{w|y}V. . . |
| 19 | perV. . . | per-V. . . | pe-rV. . . |
| 20 | pem{b|f|v}. . . | pem-{b|f|v}. . . |
| 21 | pem{rV|V}. . . | Pe-m{rV|V}. . . | pe-p{rV|V}. . . |
| 22 | pen{c|d|j|z}. . . | Pen-{c|d|j|z}. . . |
| 23 | penV. . . | Pe-nV. . . | pe-tV. . . |
| 24 | peng{g|h|q}. . . | Peng-{g|h|q} |
| 25 | pengV. . . | Peng-V | peng-kV |
| 26 | penyV. . . | Pe-nya | peny-sV |
| 27 | pelV. . . | Pe-IV. . .; kecuali untuk kata “pelajar” |
| 28 | peCP. . . | Pe-CP. . . dimana C!={r|w|y|I|m|n} dan P!=’er’ |
| 29 | perCerV. . . | Per-CerV. . . dimana C!={r|w|y|I|m|n} |

Langkah-langkah penentuan tipe awalan sebagai berikut :

1. Jika awalannya adalah: “di-”, “ke-“ atau “se-“ maka tipe awalannya secara berturut-turut adalah “di-“, “ke-“ atau “se-”.
2. Jika awalannya adalah: “te-“, “me-“, “be-“ atau “pe-“ maka dibutuhkan sebuah proses tambahan untuk menentukan tipe awalannya.
3. Jika dua karakter pertama bukan “di-“, “ke-“, “se-“, “te-“, “be-“, “me-“ atau “pe-“ maka berhenti.
4. Jika tipe awalan adalah “none” maka berhenti. Hapus awalan jika ditemukan.
5. Tahap ke-5 melakukan *recording*, pada awal kata yang dipenggal, diberi tambahan karakter *recording* Tabel 2.7dan Tabel 2.8. Jika terdapat kombinasi awalan yang dihilangkan sebelumnya atau tiga awalan yang telah dihilangkan maka proses berhenti
6. Tahap ke-6, jika semua Langkah sudah selesai, namun belum menemukan kata dasar, maka akan dikembalikan ke kata awal dan dianggap kata awal merupakan kata dasar, proses selesai.

Contoh proses *stemming* dapat dilihat pada Tabel 2.9.

#### Tabel 2.9 *Stemming*

|  |  |
| --- | --- |
| Sebelum Stemming | [‘konten’,’bermutu’,’sayang’,’suaranya’,’kurang’,’jelas’] |
| Setelah Stemming | [‘konten’,’mutu’,’sayang’,’suara’,’kurang’,’jelas’] |

## Klasifikasi

Klasifikasi merupakan proses mendapatkan model atau fungsi dari analisis atau mempelajari himpunan dokumen teks yang belum diketahui kelasnya, model atau fungsi dapat digunakan untuk mengkategorikan dokumen teks lain yang belum diketahui kelasnya kedalam satu atau lebih kelas yang telah ditentukan berdasarkan isinya (Sebastiani, 2002). Klasifikasi bertujuan untuk mengkategorikan kelas dari suatu dokumen teks yang belum diketahui kelasnya (Gaikwad et al., 2014). Berdasarkan jumlah label klasifikasi dapat dibagi menjadi 2 kategori, yaitu :

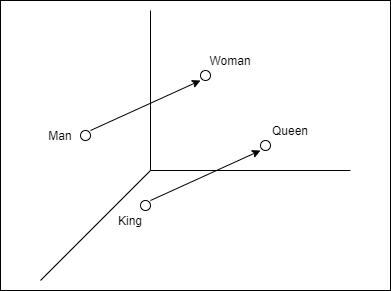
1. *Binaryclass*, mengklasifikasikan menjadi 2 kelas saja misalnya positif dan negatif, hoax dan fakta.
2. *Multiclass*, mengklasifikasikan menjadi lebih dari 3 kelas misalnya bahagia, sedih, marah.

## Word Embedding

*Word embedding* merupakan suatu teknik memetakan kata-kata berdasarkan suatu kamus yang sudah ada sehingga menghasilkan vektor-vektor angka yang berisi angka rill (Halim et al., 2020), hal ini dilakukan karena *deep learning* tidak dapat memproses data *string* atau teks, oleh karena itu data harus diubah dari data *string* atau teks menjadi vektor angka menggunakan *word embedding*. *Word embedding* juga dapat menangkap makna semantik dan sintaktik kata (Nurdin et al., 2020). Hasil dari *word embedding* dapat digunakan ke dalam ruang geografis yang dinamakan ruang *embedding* (Faadilah, 2020). Penelitian yang dilakukan oleh Utomo (2020) mendapatkan algoritma LSTM memiliki akurasi yang lebih baik jika menggunakan *word embedding* sebesar 86,76%, sedangkan tanpa *word embedding* hanya mendapatkan akurasi sebesar 84,14% (Utomo, 2020). Misalnya terdapat data “indonesia tanah air ku” akan diubah menjadi vektor menghasilkan kamus kata [“indonesia”, ”tanah”, “air”, “ku”]. Metode *one-hot encoding* akan merepresentasikan kata dengan angka 1, dan angka 0 untuk kata lainnya dalam bentuk vektor. Sehingga kata “indonesia” akan direpresentasikan dalam vektor menjadi [ 1, 0, 0, 0], kata “tanah” menjadi [ 0, 1, 0, 0], kata “air” menjadi [ 0, 0, 1, 0], kata “ku” menjadi [ 0, 0, 0, 1]. Selain metode *one-hot encoding* terdapat metode lain seperti *word2vec*, GloVe, *fasttext*. Penelitian Nurdin (2020) membandingkan *word2vec*, *Global vectors for word representation* (Glo-Ve), *fasttext* dengan *Convolutional Neural Network* (CNN) untuk klasifikasi teks, menghasilkan *fasttext* mendapatkan nilai terbaik dengan f-measure 0.979, sedangkan word2vec mendapatkan nilai f-measure 0,925 dan GloVe mendapatkan nilai f-measure 0,958 (Nurdin et al., 2020).

## Word2vec

*Word2vec* merupakan salah satu metode *word embedding* yang diperkenalkan oleh Mikolov dkk pada tahun 2013. *Word2vec* memetakan setiap kata ke dalam vektor yang dapat membawa makna semantik atau sintatikal dari kata tersebut. Proses memetakan kata pada *word embedding* menggunakan *unsupervised neural network* yang terdiri dari sebuah *hidden layer* dan *fully connected layer* (Nurdin et al., 2020). *Word2vec* memiliki kelebihan dalam similaritas makna kata yang didapatkan dari memperhatikan kesamaan kata-kata disekitar kata target (Lim et al., 2020), karena kelebihan itu metode *word2vec* sangat populer. Misalnya ada kata “king” cenderung dimiliki kata “man” maka kata “woman” akan sama artinya dengan kata “queen”, hubungan kata ini dapat dilihat pada gambar 2.1. *Word2vec* memiliki 2 teknik yaitu *continous bag of words* dan *skip gram model*.

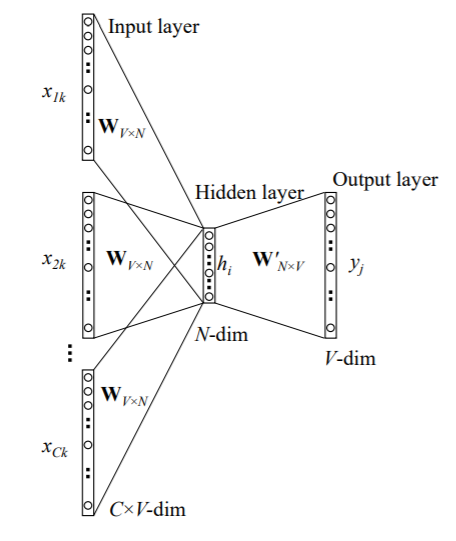


Gambar 2.1 Hubungan antar kata *word2vec*

### Continous bag of words

*Continous bag of words* (CBOW)merupakan salah satu arsitektur *word2vec* yang digunakan untuk memprediksi kata saat ini berdasarkan konteksnya (Utomo, 2020), kelebihan dari arsitektur CBOW adalah memiliki waktu *training* yang lebih cepat dan tingkat akurasi yang lebih baik untuk kata yang sering muncul. CBOW menggunakan 3 *layer* yaitu *input layer*, *hidden layer*, *output layer*. Langkah langkah CBOW mulai dari Kata yang akan masuk ke *input layer* akan dikonversikan terlebih dahulu ke bentuk vektor. Kata yang telah berbentuk vektor masuk ke *input layer* untuk pembobotan. Hasil pembobotan diproyeksikan ke *hidden layer*. Langkah terakhir menghitung *error* dengan membandingkan antara *output* dengan data target, tujuannya untuk memperbaiki nilai berdasarkan propagasi balik dari gradient kesalahannya. Ilustrasi CBOW dapat dilihat pada Gambar 2.2.

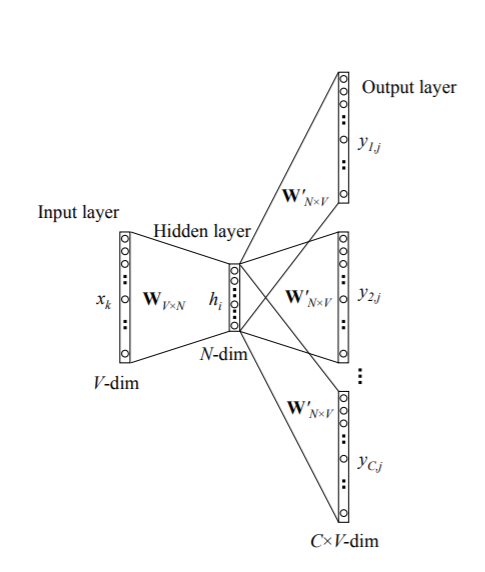
Gambar 2.2 Ilustrasi CBOW  
Sumber: (Rong, 2016)



### Skip gram model

*Skip gram model* merupakan salah satu arsitektur *word2vec*, arsitektur ini merupakan kebalikan dari arsitektur CBOW, dimana arsitektur ini berguna untuk memprediksi konteks (*output*) yang berhubungan disekitar *current word* kata input, kelebihan arsitektur ini adalah dapat bekerja dengan baik meski data *training* hanya sedikit dan arsitektur ini mampu merepresentasikan kata yang jarang muncul atau langka. Ilustrasi *Skip gram model* dapat dilihat pada Gambar 2.3.

Gambar 2.3 Ilustrasi *skip gram model*  
Sumber: (Rong, 2016)



## Glo-Ve

*Global vectors for word representation* (Glo-Ve)merupakan salah satu metode *word embedding* yang mengandalkan *co-occurrence* kata atau statistik kemunculan kata dalam kumpulan kata atau korpus yang ditangkap langsung oleh model untuk memperoleh hubungan semantik antar kata dalam korpus. Glo-Ve menggunakan metode *global matrix factorization* yang mewakili jumlah kemunculan atau frekuensi dalam suatu korpus (Pennington et al., 2014). Alur kerja Glo-ve terdiri dari langkah-langkah sebagai berikut (Pennington et al., 2014) :

1. Bentuk matriks *word co-occurence* X dimana didalam matriks tersebut berisi kumpulan statistik *word co-occurence*. Matriks tersebut terdapat elemen Xij yang merepresentasikan jumlah kata *i* muncul dalam konteks kata *j*. Penjabaran langkah ini menurut Pennington (2014) dapat dilihat pada persamaan 2.1.

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . (2.1)

Keterangan :

= vektor kata utama.

= vektor kata konteks.

= probabilitas k dalam konteks i.

= matriks word co-occurrence X.

1. Langkah selanjutnya menentukan *soft constraints* untuk setiap pasangan kata, rumus *soft constraints* menurut Pennington (2014) dapat dilihat pada persamaan 2.1.

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . (2.2)

Keterangan :

= vektor kata utama.

= vektor kata konteks.

= bias skalar kata utama.

= bias skalar kata konteks.

= matriks word co-occurrence X.

1. Langkah selanjutnya menentukan *cost function*, rumus *cost function* menurut Pennington (2014) dapat dilihat padapersamaan 2.3.

. . . . . . . . . . . . . . . . . . . . . . . . (2.3)

Keterangan :

= vektor kata utama.

= vektor kata konteks.

= bias skalar kata utama.

= bias skalar kata konteks.

= matriks word co-occurrence X.

= fungsi pembobotan.

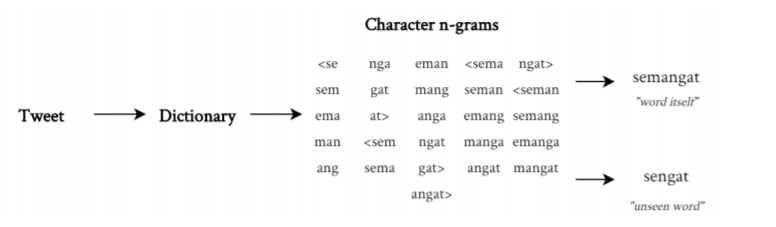
Fungsi pembobotan yang berguna untuk mencegah pembelajaran hanya dari pasangan kata yang sangat umum. Rumus *f* menurut Pennington (2014) dapat dilihat pada persamaan 2.4.

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . (2. 4)

## Fasttext

*Fasttext* merupakan metode *word embedding* pengembangan dari *word2vec*. *Fasttext* memliki kelebihan yaitu dapat menangani masalah *out of vocabulary* dimana masalah ini tidak dapat diselesaikan oleh *word embedding word2vec* dan *Glo-Ve*, *fasttext* dapat menangani masalah tersebut karena ketika ada kata yang tidak muncul selama *training* model sehingga tidak ditemukan *embedding* vektornya, maka kata tersebut akan dipecah menggunakan *n-gram* untuk mendapatkan *embedding* vektornya. Sedangkan pada *word embedding* lain jika terdapat kata yang tidak muncul selama *training* model, maka akan menghasilkan *error*. *Fasttext* mempelajari representasi kata dengan memperhatikan informasi *subword* menggunakan *n-gram* ke dalam model skipgram. Hal inilah yang membuat *fasttext* dapat menangkap kata-kata yang lebih pendek dan memahami sufiks dan prefix dari kata (Bojanowski et al., 2017). Langkah- langkah fasttext menurut Bojanowski (2017) sebagai berikut :

1. Representasikan nilai vektor kata dengan menjumlahkan nilai dari tiap *n-gram*. *N-gram* pembentuk kata ini yang memungkinkan memungkinkan muncul pada *n-gram* yang berada dalam korpus, sehingga dapat menangani kata yang belum pernah ditemui. Misalnya n = 3, dalam kata semangat menghasilkan vektor kata pada gambar 2.4.



Gambar 2.4 Vektor kata semangat   
Sumber: (Alfariqi et al., 2020)

Representasi vektor kata menurut Bojanowski (2017) dapat dihitung menggunakan persamaan 2.5.

Keterangan :

hw = kata yang ingin dicari vektor representasinya.

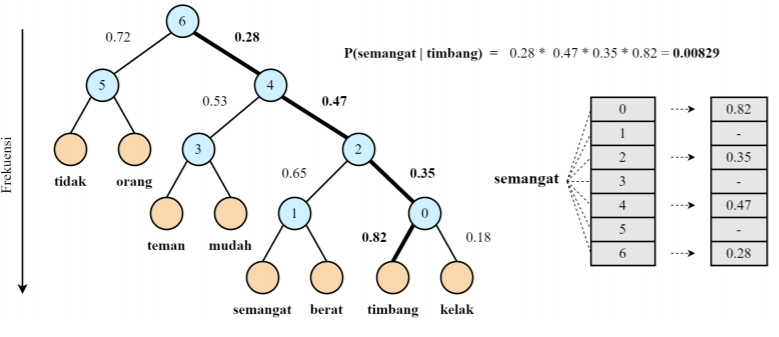
Gw = kumpulan *n-gram* yang muncul pada kata tersebut.

zg = representasi vektor tiap *n-gram*.

vc = vektor kata dari *context word* c.

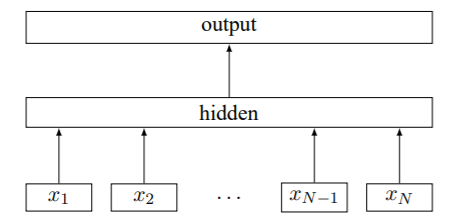
1. Langkah selanjutnya, masuk lapisan *Hierarchical Softmax* dengan metode *Huffman Tree*. Metode ini membangun *binary tree* yang memposisikan *leaf node* dengan *frekuensi* kemunculan sedikit pada level kedalaman yang lebih tinggi, dan sebaliknya. Metode ini juga mempercepat proses komputasi. Cara kerja *Hierarchical Softmax* dapat dilihat pada Gambar 2.5.

Gambar 2.5 Cara kerja *Hierarchical Softmax*



Sumber: (Alfariqi et al., 2020)

Arsitektur *fasttext* dapat dilihat pada Gambar 2.6.

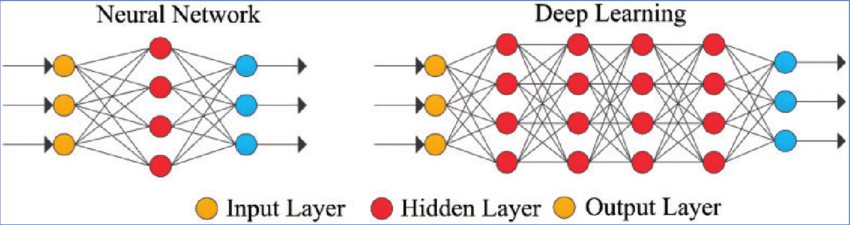


Gambar 2.6 Arsitektur *fasttext*  
Sumber: (Joulin et al., 2016)

## Deep Learning

*Deep learning* merupakan salah satu bidang *machine learning* berbasis jaringan syaraf tiruan atau *neural network*, *deep learning* terdiri dari beberapa *hidden layer* yang membentuk tumpukan (Nurfita dan Ariyanto, 2018). *Deep learning* membutuhkan waktu yang sedikit untuk melakukan *training* karena *deep learning* mampu menangani masalah gradien semakin rendah pada propagasi balik (Ahmad, 2017). Arsitektur *deep learning* dapat dilihat pada gambar 2.5.

Gambar 2.7 Arsitektur *deep learning*  
Sumber: (Xing and Du, 2019)

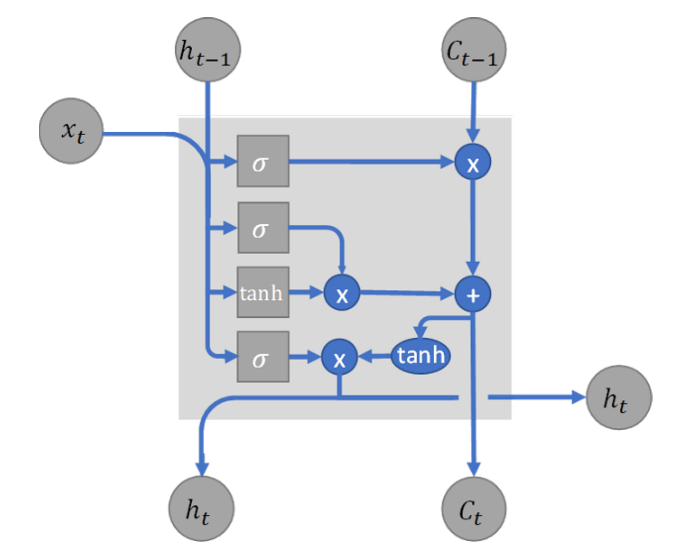


*Deep learning* efektif dalam menangani berbagai permasalahan dengan data skala besar seperti *speech recognition*, *image recognition*, *natural language comprehension*, *playing games, biomedical applications* (Temizel et al., 2020). Hasil training pada deep learning berupa model yang digunakan untuk prediksi. *Deep learning* memiliki kelebihan yaitu, model yang dihasilkan dapat membedakan suatu kelas dengan mudah, hal ini karena deep learning memiliki *feature engineering* yang berfungsi untuk mengekstrak pola yang penting dari suatu data. Contoh algoritma yang menerapkan deep learning seperti *Convolutional Neural Network* (CNN), *Recurrent Neural Network* (RNN), *Long Short Term Memory* (LSTM).

## Long Short Term Memory

*Long Short Term Memory* (LSTM) merupakan algoritma *deep learning* pengembangan dari arsitektur RNN. RNN memiliki masalah *vanishing gradient*, LSTM mampu mengatasi masalah *vanishing gradient* dengan *memory cell* dan *gate units* (*input gate*, *forget gate*, *output gate*) sehingga LSTM dapat membaca, menyimpan, dan memperbaharui informasi (Rao dan Spasojevic, 2016). Arsitektur LSTM dapat dilihat pada gambar 2.8.

Gambar 2.8 Arsitektur LSTM



Sumber: (Miedema, 2018)

Pada gambar 2.8 terdapat beberapa simbol yang mewakilkan proses LSTM, seperti persegi abu-abu melambangkan *layer neural network*, lingkaran biru melambangkan operasi *pointwise*, garis horizontal pada kanan diagram adalah *cell state*.

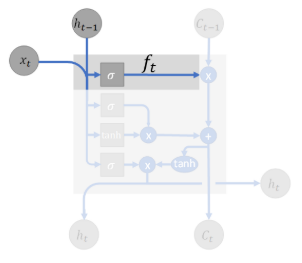
Terdapat 3 jenis *gate* pada LSTM yaitu, *forget gate*, *input gate*, *output gate*. *Forget gate* berfungsi untuk menentukan informasi mana yang dihapus dari *cell*. *Input gate* berfungsi untuk menentukan nilai dari *input* untuk diperbaharui pada *state memory*. *Output gate* berfungsi untuk menentukan *output* berdasarkan *input* dan memori pada *cell* (Utomo, 2020). Langkah-langkah LSTM sebagai berikut (Hochreiter dan Schmidhber., 1997):

1. Langkah pertama, LSTM akan menentukan informasi mana yang harus dibuang dari *cell state*, bagian ini disebut dengan *forget gate*. Langkah ini menggunakan *output* dari langkah sebelumnya (ht-1) dan *input* (xt) untuk diproses dengan fungsi aktivasi *sigmoid*. pada proses ini akan menghasilkan *output* berupa nilai 0 atau nilai 1 pada *cell state Ct-1*, nilai 0 berarti informasi akan disingkirkan, sedangkan nilai 1 berarti akan dipertahankan. Persamaan *forget gate* menurut Hochreiter dan Schmidhuber (1997)diuraikan pada persamaan 2.6, sedangkan alur informasi pada *forget gate* dapat dilihat pada gambar 2.9.

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . (2.6)

Keterangan :

*ft* : forget gate  
σ : fungsi *sigmoid*  
W*f*: weight pada forget gate  
U*f* : recrrent weight pada forget gate  
*ht-1*: *output* pada langkah sebelum orde t  
*xt* : nilai input pada orde t  
*bf* : nilai bias pada *forget gate*

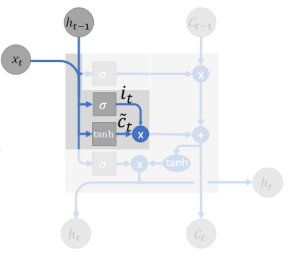


Gambar 2.9 Alur informasi pada *forget gate*

Sumber: (Miedema, 2018)

1. Langkah kedua, menentukan informasi yang akan ditambahkan pada *cell state*. Pada langkah ini memiliki memiliki dua bagian, yang pertama *layer sigmoid* dinamakan *input gate layer* berfungsi untuk menentukan nilai mana yang akan diperbaharui. Selanjutnya, *layer tanh* membuat nilai kandidat baru (𝐶̌𝑡)untuk dimasukkan ke *cell state*. *Output* dari *input gate layer* dan *tanh layer* akan digabungkan untuk memperbaharui *cell state*. Proses pada langkah ini digambarkan pada gambar 2.10. Sedangkan, menurut Hochreiter dan Schmidhuber (1997) persamaan *input gate layer* dan persamaan nilai kandidat konteks dapat dilihat pada persamaan 2.7 dan persamaan 2.8.

Gambar 2.10 Proses *input gate layer*



Sumber: (Miedema, 2018)

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .(2.7)

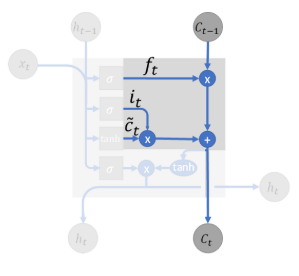
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . (2.8)

Keterangan :

it : input gate.  
𝐶̌𝑡 : kandidat baru.  
𝜎 : fungsi *sigmoid*.  
Wi : weight pada input gate.  
Ui : recurrent weight pada input gate.  
*bi* : nilai bias pada *input gate*.  
*Wc* : *weight* pada kandidat konteks.  
*Uc* : *recurrent weight* pada kandidat baru.  
*bc* : nilai bias pada kandidat konteks.  
*ht-1* : nilai *output* sebelum orde t.  
*xt* : nilai *input* pada orde t.

1. Langkah ketiga, memperbaharui *cell state* lama *Ct-1* menjadi *cell state* baru *Ct*. Pada langkah sebelumnya, telah didapatkan nilai *input gate* dan kandidat konteks. Langkah ketiga akan terjadi proses perkalian *cell state* lama *Ct-1* dengan nilai *forget gate ft*, kemudian hasil perkalian *cell state* lama Ct-1 dengan nilai *forget gate* ditambah dengan hasil perkalian nilai *input gate* dengan nilai kandidat konteks 𝐶̌𝑡, akan menghasilkan *cell state* baru (*Ct*). Proses ini diuraikan pada gambar 2.11. Persamaan pada langkah mendapatkan *cell state* baru (*Ct*) menurut Hochreiter dan Schmidhuber (1997) dapat dilihat pada persamaan 2.9.

Gambar 2.11 Langkah mendapatkan nilai *cell state* baru



Sumber: (Miedema, 2018)

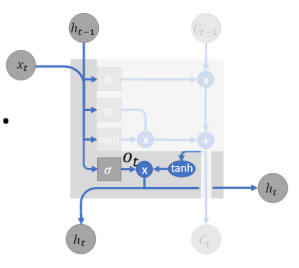
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .(2.9)

Keterangan :

*Ct* : cell state orde t  
*ft* : forget gate  
*Ct-1* : *cell state* sebelum orde t  
*it* : nilai *input* pada orde t  
𝐶̌𝑡 : kadidat konteks

1. Langkah terakhir, menentukan *output* hasil keseluruhan proses (ht). Pertama *layer sigmoid* digunakan *output* sebelumnya (ht-1) dan *input* (xt) untuk menentukan nilai *output gate* (Ot), nilai *output gate* (Ot) antara 0 dan 1 yang menunjukkan bagian yang menjadi *output* dari *cell state*, selanjutnya *cell state* (Ct) diubah menggunakan fungsi aktivasi *tanh*, *cell state* ini mendapatkan nilai antara -1 dan 1. Nilai *cell state* kemudian dikalikan dengan nilai *output gate* (Ot) menghasilkan nilai *output* (ht). Proses ini dijelaskan pada gambar 2.12. Persamaan proses ini menurut Hochreiter dan Schmidhuber (1997) dijabarkan pada persamaan 2.10 dan persamaan 2.11.

Gambar 2.12 *Output gate*



Sumber: (Miedema, 2018)

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .(2.10)

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .(2.11)

Keterangan :

Ot : output gate  
𝜎 : fungsi *sigmoid*  
Wo : nilai weight untuk output gate  
Uo : recurrent weight pada output gate  
*ht-1* : nilai *output* sebelum orde t  
*xt* : nilai *input* orde t  
*bo*: nilai bias pada *output gate*  
Ct : cell state

Dari penjelasan proses LSTM, terdapat dua fungsi aktivasi yaitu, *sigmoid* dan *tanh*. Fungsi aktivasi *sigmoid* dijabarkan pada persamaan 2.12, sedangkan fungsi aktivasi *tanh* dijabarkan pada persamaan 2.13.

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . (2.12)

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . (2.13)

Keterangan :

𝜎 : fungsi *sigmoid*.  
x : data input.  
e : fungsi eksponensial.

## Confusion Matrix

*Confusion matrix* merupakan cara untuk menguji keakuratandengan menghitung nilai *precision*, *recall*, *accuracy*. Menurut Han dan Kamber (2006) *confusion matrix* merupakan alat untuk menganalisis *classifier* mengenali *tuple* dari kelas yang berbeda dengan baik atau tidak. Terdapat empat istilah pada *confusion matrix* yang merepresentasikan hasil, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), *False Negative* (FN). Tabel ilustrasi *confusion matrix* dapat dilihat pada Tabel 2.10.

#### Tabel 2.10 Ilustrasi *confusion matrix*

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Actual Values | |
| Positive | Negative |
| Predicted Values | Positive | True Positive (TP) | False Negative(FN) |
| Negative | False Positive (FP) | True Negative (TN) |

Keterangan :

*True Positive* (TP) : jumlah data positive yang diprediksi benar.

*True Negative* : jumlah data negative yang diprediksi benar.

*False Positive* : jumlah data negatif yang diprediksi salah.

*False Negative* : jumlah data positive yang diprediksi salah.

Berdasarkan tabel *confusion matrix* maka dapat digunakan untuk mengukur *accuracy*, *precision*, *recall*. Menurut Han dan Kamber (2006) persamaan *accuracy, precision*, *recall* dijabarkan sebagai berikut (Han dan Kamber, 2006):

1. **Accuracy**

*Accuracy* merupakan nilai yang merepresentasikan tingkat kedekatan antara data yang diprediksi dengan data aktual. Perhitungan *accuracy* dengan membandingkan data prediksi benar dengan keseluruhan data. Rumus *accuracy* dapat dilihat pada persamaan 2.14.

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . (2.14)

1. **Precision**

*Precision* merupakan perbandingan antara data yang diprediksi benar *positive* dengan data yang diprediksi *positive*. Rumus untuk menghitung *precision* dapat dilihat pada persamaan 2.15.

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .(2.15)

1. **Recall***Recall* merupakan perbandingan antara data yang diprediksi benar *positive* dengan data yang benar *positive*. Rumus untuk menghitung *recall* dapat dilihat pada persamaan 2.16.

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . (2.16)

## Penelitian Sebelumnya

Penelitian yang dilakukan oleh Fanesya dkk (2019) menggunakan metode *naïve bayes* dan kombinasi fitur *n-gram*, *dataset* yang digunakan berasal dari *twitter* dan menggunakan model Ekman, penelitian ini menghasilkan akurasi tertinggi 55,5%.

Penelitian yang dilakukan oleh Rohman dkk (2019) membandingkan metode *lexicon* dengan *emolex* dengan metode *Natural Language Processing* (NLP), *dataset* yang digunakan berasal dari *facebook* dan model emosi yang digunakan adalah model Ekman. Penelitian yang dilakukan oleh Rohman dkk (2019) menghasilkan akurasi 55,54% dari 34.872 kata untuk *lexicon* dengan *emolex*, sedangkan untuk NLP menghasilkan akurasi 61,53%.

Rohman dkk (2019) melakukan penelitian dengan *dataset* yang berasal dari *facebook* dan model emosi yang digunakan adalah model Ekman, penelitian Rohman dkk (2019) menggunakan metode *Term Frequency-Inverse Document Frequency* (TF-IDF) dengan akurasi 59%.

Penelitian yang dilakukan oleh Ardiada dkk (2019) membandingkan metode *Support Vector Machine* (SVM) dengan metode *K-Nearest Neighbor* (KNN), *dataset* yang digunakan berasal dari *twitter* dan model emosi yang digunakan model 6 emosi (jijik, malu, marah, sedih, senang, takut). Penelitian yang dilakukan oleh Ardiada dkk (2019) menghasilkan *precision* 45,64%, *recall* 50,20%, dan akurasi 81,04% untuk metode SVM, sedangkan KNN *precision* 34,21%, *recall* 45,95%, serta akurasi 79,70%.

Penelitian yang membandingkan SVM tidak hanya dilakukan oleh Ardiada dkk (2019), namun juga dilakukan oleh Haryadi dkk (2019). Penelitian Haryadi dkk (2019) membandingkan metode *Support Vector Machine* (SVM), *Long Short Term Memory* (LSTM) dengan *word embedding word2vec* dan *Nested Long Short Term Memory* (Nested LSTM) dengan *word embedding word2vec*. Penelitian ini menggunakan *dataset* dari *twitter* dan model 7 emosi (marah, takut, bahagia, cinta, sedih, terkejut, bersyukur). Penelitian ini menghasilkan sebagai berikut, SVM memiliki akurasi 98,679%, *precision* 98,53%, *recall* 98,22%, *f1-score* 98,37%. Nested LSTM akurasi 99,167%, *precision* 99,21%, *recall* 98,83%, *f1-score* 99,02%. LSTM memiliki akurasi 99,154%, namun memiliki *precision*, *recall*, *f1-score* paling tinggi dengan *precision* 99,22%, *recall* 98,86%, *f1-score* 99,04%.

Penelitian yang menggunakan metode LSTM juga pernah dilakukan oleh Dwi dkk (2020), penelitian tersebut menggunakan metode LSTM dengan *word embedding Global vector representation for words* (Glo-Ve). *Dataset* yang digunakan berasal dari *twitter* dan model emosi yang digunakan adalah model 5 emosi (marah, cinta, sedih, takut, bahagia). Penelitian ini hanya menghasilkan akurasi 50%, *precision* 33%, *recall* 38%, *f-1 score* 35%, hal ini terjadi karena selama proses *training* terjadi *underfitting*.

Rangkuman penelitian yang telah dijabarkan diatas dapat dilihat pada tabel 2.11 dan tabel 2.12.

#### Tabel 2.11 Penelitian sebelumnya

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Penulis | Metode | Dataset dan Model | Hasil | Keterangan |
| 1 | Fera Fanesya, Randy Cahya Wihandika, Indriati (2019) | Naïve bayes dan kombinasi fitur n-gram | *Twitter* dan model Ekman | Penelitian ini menghasilkan akurasi 55,5%. |  |
| 2 | Rohman, Utami, Raharjo (2019) | Lexicon dengan emolex dan Natural Language Processing | *facebook* dan Ekman | Akurasi 55,54% dari 34.872 kata untuk lexicon dengan emolex, NLP menghasilkan akurasi 61,53% |  |
| 3 | Arif Nur Rohman, Riska Dwi Handayani, Ryan Dwi Y.P, Kusrini (2019) | Term Frequency-Inverse Document Frequency | *facebook* dan Ekman | Akurasi 59% dari 6 emosi | hal ini karena jumlah data antar emosi tidak sama, sehingga peneliti menguji dengan 1 emosi saja dan menghasilkan 87,23% |
| 4 | Ardiada, Sudarma, Giriantari (2019) | Support Vector Machine(SVM) dan K-Nearest Neighbor(KNN) | *Twitter* dan model 6 emosi(jijik, malu, marah, sedih, senang, takut). | *precision* 45,64%, *recall* 50,20%, dan akurasi 81,04% untuk SVM, sedangkan KNN *precision* 34,21%, *recall* 45,95%, serta akurasi 79,70% |  |

#### Tabel 2.12 Penelitian sebelumnya (lanjutan)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Penulis | Metode | Dataset dan Model | Hasil | Keterangan |
| 5 | Daniel Haryadi, Gede Putra Kusuma (2019) | Support Vector Machine (SVM), Long Short Term Memory (LSTM) dengan word embedding word2vec dan Nested Long Short Term Memory (Nested LSTM) dengan word embedding word2vec | *twitter* dan model 7 emosi (marah, takut, bahagia, cinta, sedih, terkejut, bersyukur) | SVM memiliki akurasi 98,679%, *precision* 98,53%, *recall* 98,22%, *f1-score* 98,37%. Nested LSTM akurasi 99,167%, *precision* 99,21%, *recall* 98,83%, *f1-score* 99,02%. LSTM memiliki akurasi 99,154%, namun memiliki *precision*, *recall*, *f1-score* paling tinggi dengan *precision* 99,22%, *recall* 98,86%, *f1-score* 99,04% |  |
| 6 | Adinda Dwi L, Budi Harijanto, Faisal Rahutomo (2020) | Long Short Term Memory (LSTM) dengan word embedding Global vector representation for words (Glo-Ve) | *Twitter* dan model 5 emosi (marah, cinta, sedih, takut, bahagia) | Akurasi 50%, *precision* 33%, *recall* 38%, *f-1 score* 35%. | Hasil penelitian kurang maksimal karena selama proses *training* terjadi *underfitting*. |
| 7 | M. Alfa Riza | Long Short Term Memory (LSTM) dengan word embedding Fasttext | *Twitter* dan model ekman |  |  |

Penelitian terdahulu memiliki kesamaan dalam tema penelitian dan objek penelitian, namun pada penelitian ini memiliki beberapa perbedaan, yaitu :

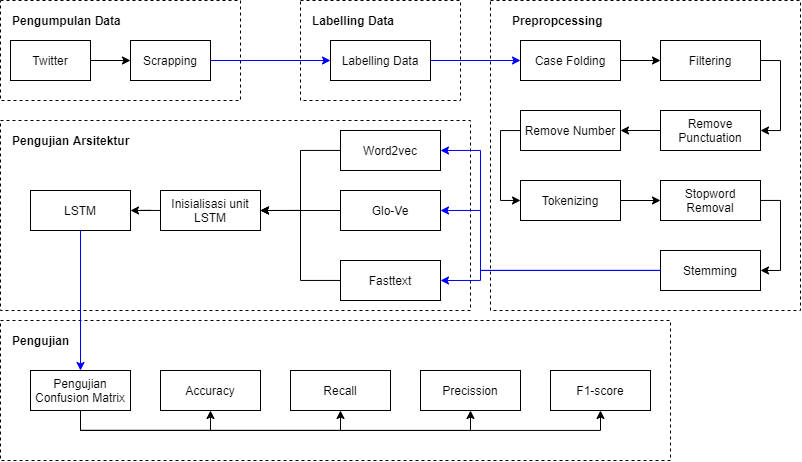
1. Penelitian sebelumnya belum menangani masalah kesalahan pengetikan (OOV).
2. Penelitian ini menggunakan *word embedding fasttext* untuk menangani kesalahan pengetikan (OOV).

# BAB III METODOLOGI PENELITIAN DAN PENGEMBANGAN SISTEM



## Metodologi Penelitian

Pada bab ini akan membahas mengenai metodologi penelitian yang digunakan pada penelitian ini. Penelitian ini termasuk penelitian non-implementatif berupa hasil analisis pengujian *word embedding* dan arsitektur LSTM. Metodologi penelitian yang akan digunakan pada penelitian ini dapat dilihat pada gambar 3.1.



Gambar 3.1 Metodologi Penelitian

### Pengumpulan Data

Data yang akan digunakan pada penelitian ini adalah data sekunder yang bersumber dari media sosial *twitter* menggunakan teknik *web scrapping*, data yang diambil untuk data penelitian berupa *tweet* dari beberapa *influencer* sebagai data utama dan data *trending* selama 1 minggu sebagai data pendukung. *Influencer* dipilih karena untuk menjadi *influencer* dibutuhkan proses pengungkapan diri (*self disclosure*) dimedia sosial, dimana salah satu bentuk *self disclosure* adalah mengekspresikan emosi (Saifulloh dan Siregar, 2019). *Web scrapping* dilakukan menggunakan bahasa pemprograman *python* dengan *library selenium*/*tweepy* dan menghasilkan dataset dengan format csv/xlsx.

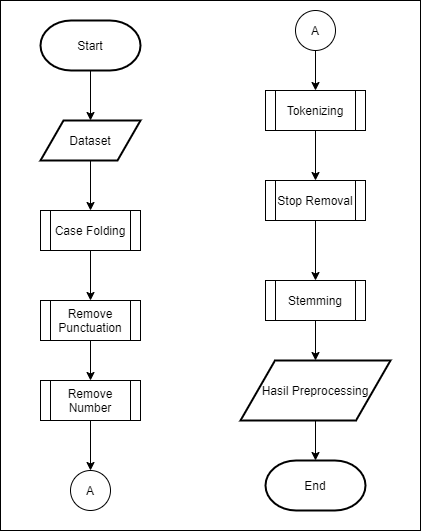
### Labelling Data

Data yang didapatkan dari hasil *web scrapping* kemudian diberikan label emosi berdasarkan model Ekman, yaitu bahagia, sedih, takut, jijik, marah, terkejut. Labelling akan dilakukan oleh beberapa mahasiswa yang cukup sering bermain twitter, hal ini dilakukan agar data yang diperoleh untuk training model tidak bersifat subjektif. Ilustrasi labelling data dapat dilihat pada tabel 3.1.

#### Tabel 3.1 Ilustrasi Labelling

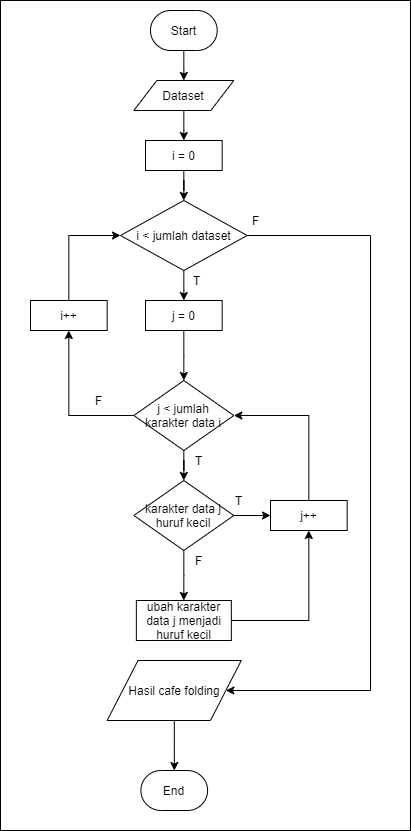
|  |  |
| --- | --- |
| Teks | Emosi |
| emang cocok banget mukanya senyum terus bahagia selalu | Bahagia |
| Sedih banget pas pamitan ke bapak kos | Sedih |
| Kalau ketemu anjing kabur aja, takut | Takut |
| “GaSuKa gElAy” oposih jijik dewe aku mahan | Jijik |
| dIAJAK FOTO MALAH BERANTEM | Marah |
| Terkejut bila tengok Berita tadi | Terkejut |

### Preprocessing

 Data yang sudah diperoleh dari proses *web scrapping* dan sudah diberi label, selanjutnya masuk pada proses *preprocessing*. *Preprocessing* berguna untuk membersihkan data dari derau dan bagian tidak informatif yang tidak dibutuhkan, sehingga teks siap untuk diklasifikasi (Haddi et al., 2013). *Preprocessing* akan menjadikan data bersih sehingga proses pembuatan vektor kata dan klasifikasi lebih akurat (Nurrohmat dan SN, 2019). Tahapan *preprocessing* yang akan digunakan pada penelitian ini adalah *case folding*, *remove punction*, *remove number*, *tokenizing*, *stop removal*, *stemming*. *Flowchart* dari proses *preprocessing* dapat dilihat pada gambar 3.2.

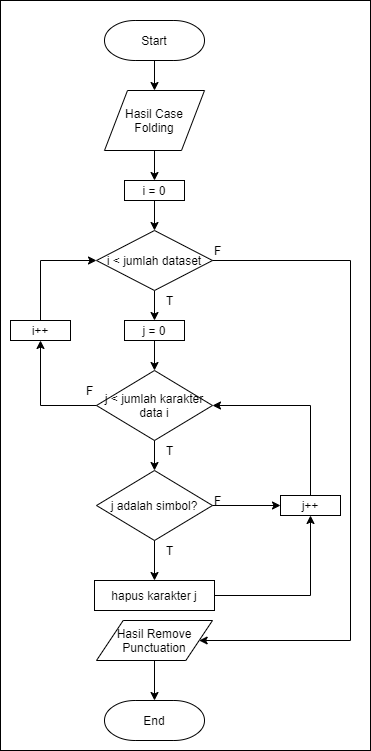
Gambar 3.2 Flowchart Preprocessing

1. **Case folding**

Langkah pertama pada proses *preprocessing* adalah *case folding*. *Case folding* merupakan proses mengubah seluruh huruf dalam teks tersebut menjadi *lower case* atau huruf kecil semua (Salam et al., 2018), dengan huruf yang diterima dari “a” sampai “z”. *Case folding* bertujuan agar terdapat standarisasi dalam penulisan. *Flowchart* dari proses *case folding* dapat dilihat pada gambar 3.3.

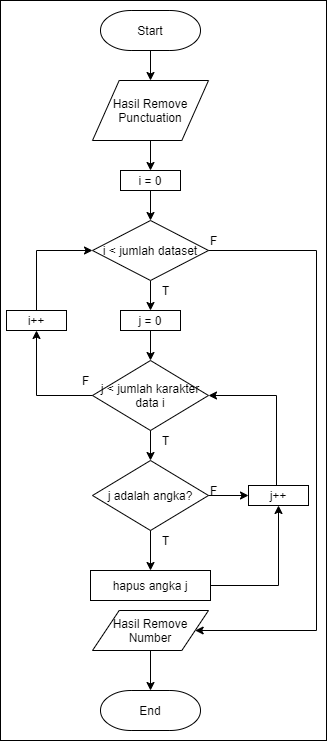
Gambar 3.3 Flowchart case folding

1. **Remove punctuation**

 Data yang telah melalui proses *case folding*, selanjutnya data akan diproses pada *remove punctuation*. *Remove punctuation* merupakan proses menghapus tanda baca pada teks dengan tujuan mengurangi beban pemprosesan klasifikasi karena dianggap tidak penting dan termasuk *delimiter*, contoh tanda baca yang yang dihapus titik (.), koma(,), tanda tanya (?), *slash* (/), *hastag* (#), tanda seru (!) dan lain-lain. *Flowchart* dari proses *remove punctuation* dapat dilihat pada gambar 3.4.

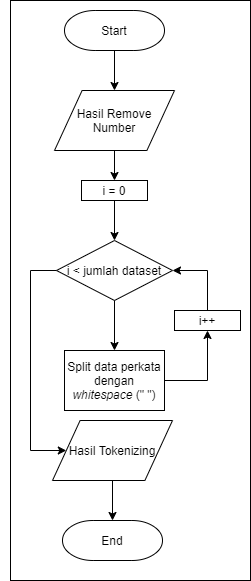
Gambar 3.4 Flowchart remove punctuation

1. **Remove number**

 Data hasil dari proses *remove punctuation* kemudian diproses ke proses *remove number*. *Remove number* merupakan proses menghapus angka pada suatu teks, penghapusan ini sebab angka dianggap tidak memiliki arti dan termasuk *delimiter*, mirip seperti *remove punctuation* hanya berbeda pada objek yang dihapus. *Flowchart* proses *remove number* dapat dilihat pada gambar 3.5.

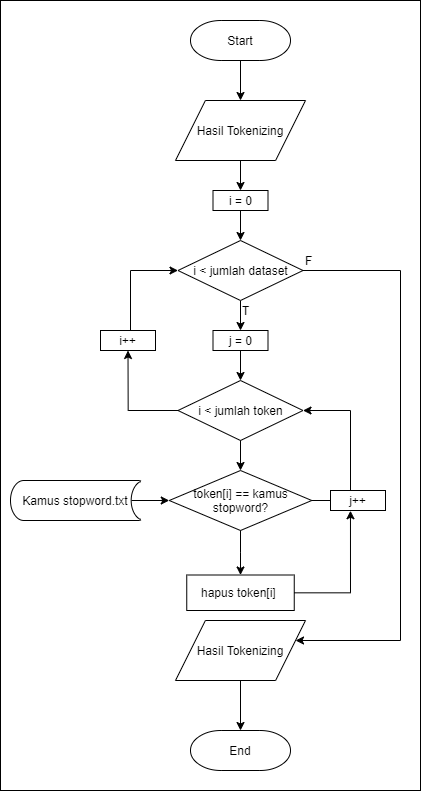
Gambar 3.5 Flowchart remove number

1. **Tokenizing**

 Data hasil dari proses *remove number* kemudian diproses ke proses *tokenizing*. *Tokenizing* merupakan pemotongan kalimat berdasarkan tiap-tiap kata penyusunnya, biasanya pemotongan berdasarkan *whitespace* seperti spasi, tab, dan enter. Tiap-tiap kata hasil *tokenizing* disebut token (Juwiantho et al., 2020). *Flowchart* dari proses *tokenizing* dapat dilihat pada gambar 3.6.

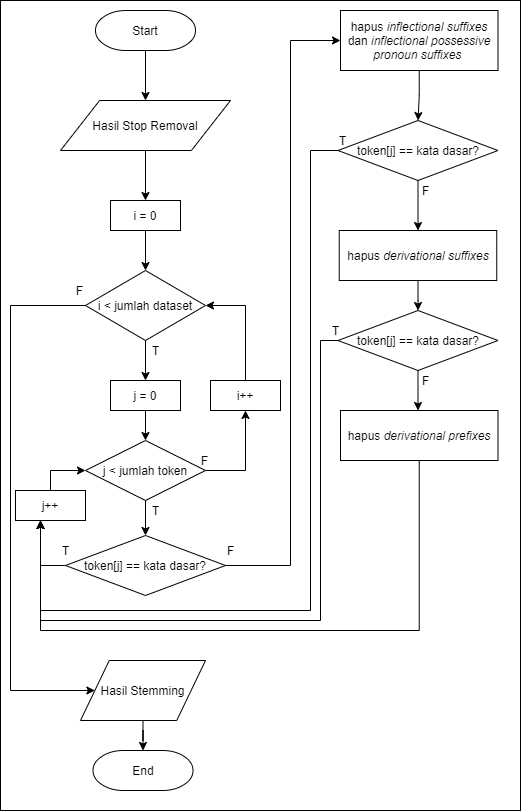
Gambar 3.6 Flowchart tokenizing

1. **Stopword Removal**

 Data hasil dari proses *tokenizing* kemudian diproses ke proses *stopword removal*. *Stopword Removal* merupakan proses untuk menghilangkan kata-kata yang tidak memiliki arti atau makna, namun tidak akan mengubah makna dari komentar tersebut (Juwiantho et al., 2020). *Stopword removal* akan mengurangi ukuran indeks, waktu pemprosesan klasifikasi, dan *noise* dari suatu data. Biasanya *stopword* berupa kata ganti orang dan kata hubung, seperti “aku”, “kamu”, “kita”, “dan”, “atau”. *Flowchart* dari proses *stopword removal* dapat dilihat pada gambar 3.7.

Gambar 3.7 Flowchart stopword removal

1. **Stemming**

 Data hasil dari proses *stopword removal* kemudian masuk ke proses *stemming*. *Stemming* merupakan proses untuk mentransformasi kata kerja yang berimbuhan pada suatu dokumen menjadi kata dasar (*root word*), dengan menghapus awalan, akhiran maupun sisipan . *Stemming* bertujuan untuk mengurangi varian kata dengan makna yang hampir sama pada suatu dokumen, serta meningkatkan performa pada tahap *information retrieval*. Pada penelitian ini akan menggunakan *library* sastrawi, karena data pada penelitian ini menggunakan bahasa Indonesia. *Flowchart* proses *stemming* dapat dilihat pada gambar 3.8.

Gambar 3.8 Flowchart stemming

Contoh data sebelum dan sesudah *preprocessing* dapat dilihat pada tabel 3.2

#### Tabel 3.2 Contoh *preprocessing*

|  |  |
| --- | --- |
| Teks asli (sebelum *preprocessing)* | “Hore, akhirnya timnas Indonesia menang dengan skor 2-0 !!” |
| Teks setelah case folding | “hore, akhirnya timnas indonesia menang dengan skor 2-0 !!” |
| Teks setelah remove punctuation | “hore, akhirnya timnas indonesia menang dengan skor 20” |
| Teks setelah remove number | “hore akhirnya timnas indonesia menang dengan skor” |
| Teks setelah *tokenizing* | [“hore”, “akhirnya”, “timnas”, “indonesia”, “menang”, “dengan”, “skor”] |
| Teks setelah stopword removal | [“hore”, “akhirnya”, “timnas”, “indonesia”, “menang”, “skor”] |
| Teks setelah *stemming* | [“hore”, “akhir”, “timnas”, “indonesia”, “menang”, “skor”] |
| Teks setelah preprocessing | [“hore”, “akhir”, “timnas”, “indonesia”, “menang”, “skor”] |

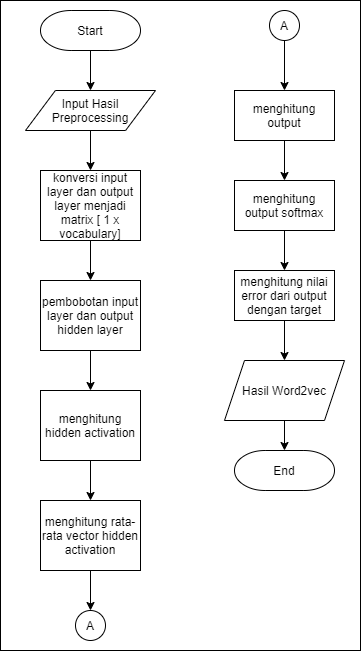
### Pengujian Arsitektur

Pengujian arsitektur bertujuan untuk menemukan arsitektur terbaik sehingga mendapatkan hasil *accuracy*, *recall*, *precission* yang maksimal. Pengujian arsitektur menggunakan beberapa parameter, diantaranya jumlah *unit* / *neuron*, fungsi aktivasi, optimasi, *epoch*, jumlah *dropout*. Parameter yang diuji adalah jumlah jumlah *unit* / *neuron* dan jumlah *dropout*. Penelitian ini akan menggunakan jumlah *unit* / *neuron* sebanyak 50 *unit* sampai 200 *unit*, sedangkan untuk *dropout* akan menggunakan 20, 30, dan 50. *Dropout* berguna untuk memutus sebagian koneksi antar unit secara random sehingga mengurangi *overfitting* secara signifikan. Fungsi aktivasi pada penelitian ini menggunakan fungsi aktivasi *sigmoid*. Optimasi yang digunakan pada penelitian ini adalah *Adaptive Moment Estimation* (ADAM). Jumlah *epoch* yang digunakan pada penelitian ini sebanyak 50 *epoch*. Fungsi *loss* yang digunakan adalah fungsi *Loss Categorical Cross Entropy*, karena pada penelitian ini terdapat lebih dari 2 kelas. Pembagian antara data *training* dan data *testing* pada penelitian ini menggunakan 70% data *training* dan 30% data *testing* dengan, hal ini dikarenakan jumlah pembagian 70:30 dan 80:20 dapat menghasilkan model terbaik (Gholamy et al., 2018).

### Word embedding

Bagian ini akan menjelaskan tahapan dari proses *word embedding*, hasil dari *preprocessing* kemudian masuk ke proses *word embedding*. *Word embedding* merupakan suatu teknik memetakan kata-kata berdasarkan suatu kamus yang sudah ada sehingga menghasilkan vektor-vektor angka yang berisi angka rill (Halim et al., 2020), hal ini dilakukan karena *deep learning* tidak dapat memproses data *string* atau teks, oleh karena itu data harus diubah dari data *string* atau teks menjadi vektor angka menggunakan *word embedding*. Penggunaan *word embedding* berpengaruh terhadap akurasi yang dihasilkan oleh model. Penelitian ini akan membandingkan *word embedding word2vec*, *Glo-Ve*, *fasttext* untuk mencapai tujuan penelitian.

1. **Word2vec**

Word embedding word2vec banyak digunakan karena kelebihannya dalam mengenali kata-kata yang mirip berdasarkan sematiknya (Nurdin et al., 2020). Word embedding word2vec menggunakan unsupervised neural network yang terdiri dari sebuah hidden layer dan fully connected layer (Nurdin et al., 2020). Hasil dari preprocessing kemudian masuk ke proses word2vec. Model word2vec yang akan digunakan pada penelitian ini adalah model Continous bag of words (CBOW). Flowchart proses word2vec dapat dilihat pada gambar 3.9.

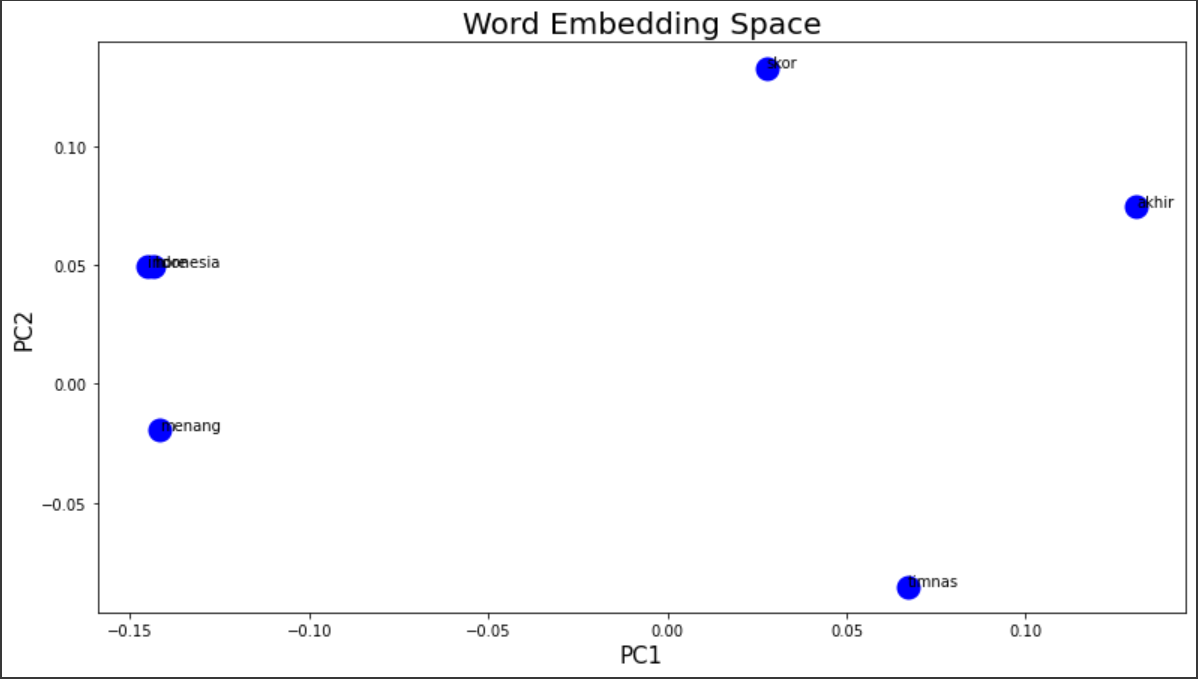
Gambar 3.9 Flowchart word2vec

Contoh hasil *word2vec* dengan vektor berdimensi 5 dengan kalimat yang sama dengan hasil proses *preprocessing*, contoh dapat dilihat pada tabel 3.2.

#### Tabel 3.3 Contoh Hasil *Word2vec*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Kata | Dimensi | | | | |
| D1 | D2 | D3 | D4 | D5 |
| hore | 0.084777 | 0.057706 | -0.082217 | -0.054878 | -0.096413 |
| akhir | -0.094585 | 0.068689 | 0.090819 | -0.059799 | 0.088703 |
| timnas | -0.076862 | -0.039967 | 0.068271 | 0.027309 | -0.035721 |
| indonesia | 0.095575 | -0.025707 | -0.084364 | -0.094829 | -0.006755 |
| menang | 0.085284 | -0.021515 | -0.033860 | -0.067008 | -0.096384 |
| skor | -0.022820 | 0.086830 | -0.074412 | -0.014862 | 0.085389 |

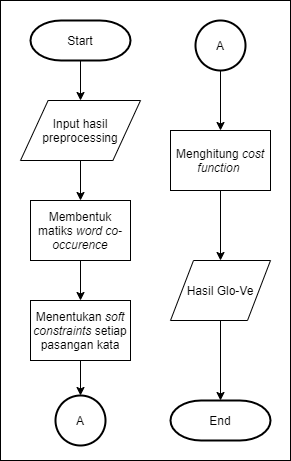
Vektor vektor hasil proses *word embedding word2vec* dapat divisualisasikan secara 2 dimensi yang menggambarkan hubungan antar kata. Visualisasi vektor vektor hasil *word2vec* dapat dilihat pada gambar 3.10.



Gambar 3.10 Visualisasi *word2vec*

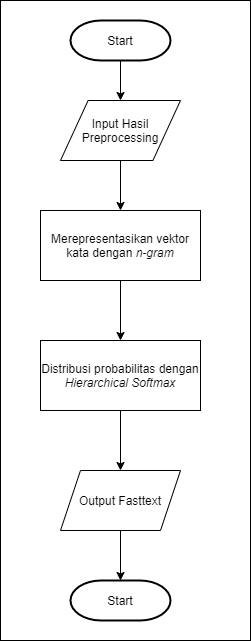
1. **Glo-Ve**

Hasil dari proses *preprocessing* kemudian masuk ke *word embedding Glo-Ve* untuk dipetakan menjadi vektor angka. *Global vectors for word representation* (Glo-Ve)merupakan salah satu metode *word embedding* yang mengandalkan *co-occurrence* kata atau statistik kemunculan kata dalam kumpulan kata atau korpus yang ditangkap langsung oleh model untuk memperoleh hubungan semantik antar kata dalam korpus. Glo-Ve menggunakan metode *global matrix factorization* yang mewakili jumlah kemunculan atau frekuensi dalam suatu korpus (Pennington et al., 2014). *Flowchart word embedding Glo-Ve* dapat dilihat pada gambar 3.11.

Gambar 3.11 Flowchart Glo-Ve

1. **Fasttext**

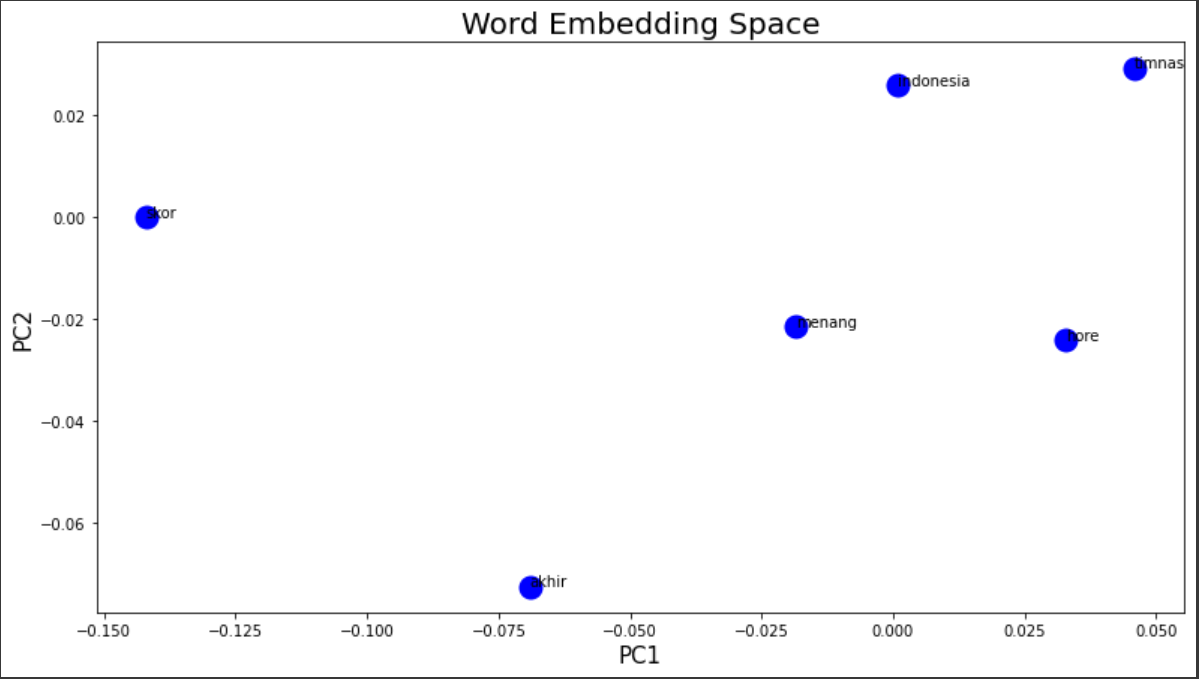
Hasil dari proses *preprocessing* kemudian masuk ke proses *word embedding fasttext* untuk mengubahnya ke vektor angka. *Fasttext* merupakan metode *word embedding* pengembangan dari *word2vec*. *Fasttext* memliki kelebihan yaitu dapat menangani masalah *out of vocabulary* dengan memperhatikan informasi *subword* menggunakan *n-gram* ke dalam model skipgram(Bojanowski et al., 2017). *Flowchart* proses *word embedding fasttext* dapat dilihat pada gambar 3.12.

Gambar 3.12 Flowchart fasttext

Contoh hasil *fasttext* dengan vektor berdimensi 5 dengan kalimat yang sama dengan hasil proses *preprocessing*, contoh dapat dilihat pada tabel 3.3. Vektor vektor hasil proses *word embedding word2vec* dapat divisualisasikan secara 2 dimensi yang menggambarkan hubungan antar kata. Visualisasi vektor vektor hasil *fasttext* dapat dilihat pada gambar 3.13.

#### Tabel 3.4 Contoh hasil *fasttext*

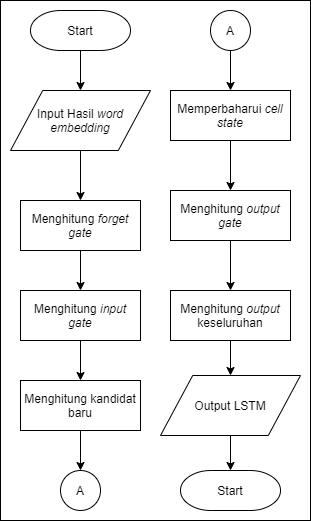
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Kata | Dimensi | | | | |
| D1 | D2 | D3 | D4 | D5 |
| hore | -0.015188 | 0.006831 | -0.000868 | -0.033525 | -0.036368 |
| akhir | 0.026232 | -0.020445 | 0.043318 | 0.081208 | -0.070539 |
| timnas | 0.041364 | 0.038119 | -0.043051 | -0.017365 | -0.014987 |
| indonesia | 0.026756 | -0.005694 | -0.003622 | -0.020880 | 0.005214 |
| menang | -0.005905 | 0.024995 | 0.015284 | 0.050226 | 0.000018 |
| skor | 0.041438 | -0.059298 | 0.087633 | 0.081472 | 0.033918 |



Gambar 3.13 Visualisasi *fasttext*

### LSTM

Hasil *output word embedding* kemudian masuk pada proses LSTM. LSTM memiliki *memory cell* dan *gate units* (*input gate, forget gate, output gate*) untuk menangani masalah masalah *vanishing gradient*. *Forget gate* berfungsi untuk menentukan informasi mana yang dihapus dari *cell*. *Input gate* berfungsi untuk menentukan nilai dari *input* untuk diperbaharui pada *state memory*. *Output gate* berfungsi untuk menentukan *output* berdasarkan *input* dan memori pada *cell* (Utomo, 2020). *Flowchart* proses LSTM dapat dilihat pada gambar 3.14.

Gambar 3.14 *Flowchart* LSTM

Analisis mengenai *Long Short Term Memory* (LSTM) lebih lanjut dibahas dalam contoh perhitungan algoritma LSTM dengan data yang didapatkan dari hasil proses *preprocessing* dan hasil *word embedding fasttext* pada tabel 3.4. Data hasil *preprocessing* dan *word embedding fasttext* terdapat 6 kata, yaitu “hore”, “akhir”, “timnas”, “indonesia”, “menang”, “skor”, dimana kata-kata tersebut akan menjadi *input* dari LSTM secara berurutan. Contohnya hasil *output* dari proses “hore” akan menjadi input dari kata “akhir”. Contoh perhitungan LSTM yang digunakan sebanyak 5 *neuron*. *Neuron* berguna dalam pembentukan *weight* tiap gerbang (*W*), *recurrent weight* tiap gerbang(*U*), dan bias tiap gerbang (*b*). Proses perhitungan *input* kata “hore” dengan 5 *neuron*, yaitu :

1. Menghitung forget gate (*ft*)

Proses menghitung *forget gate* menggunakan rumus pada persamaan 2.6. Variabel *weight forget gate* (*Wf*), *recurrent weight forget gate* (*Uf*), *bias forget gate* (*bf*) di inisialisasi awal dengan nilai *random*, sedangkan untuk variabel *output* proses sebelumnya (*ht-1*) merupakan *output* dari proses sebelumnya, namun karena pada contoh ini adalah kata awal maka nilai *ht-1* adalah matriks dengan nilai 0, dan *xt* merupakan input yang didapatkan dari hasil *word embedding fasttext*. Sehingga masing masing variabel dijabarkan sebagai berikut :

Nilai *weight forget gate* (*Wf*)kemudian dikalikan dengan *input* (*xt*), hasil dari perkalian tersebut dejabarkan sebagai berikut :

Selanjutnya, nilai *recurrent weight forget gate* (*Uf*)dikalikan dengan *output* dari proses sebelumnya (*ht-1*), karena *output* dari proses sebelumnya (*ht-1*) tidak ada maka nilainya 0, sehingga hasil *recurrent weight forget gate* (*Ui*)dikalikan dengan *output* dari proses sebelumnya (*ht-1*) adalah 0, hasil tersebut dijabarkan sebagai berikut :

Hasil proses dan kemudian ditambahkan dengan nilai *bias* (*bi*), dijabarkan sebagai berikut :

Nilai *forget gate* (*ft*)didapatkan dengan menghitung hasil dengan fungsi aktivasi *sigmoid*. Fungsi aktivasi *sigmoid* akan menghasilkan nilai antara 0 dan 1, proses menghitung *forget gate* (*ft*)dijabarkan sebagai berikut :

1. Menghitung *input gate* (*it*)

Proses selanjutnya menghitung *input gate* menggunakan rumus pada persamaan 2.7, sama seperti *forget gate* ada beberapa variabel yang dibutuhkan yaitu *weight input gate* (*Wi*), *recurrent weight input gate* (*Ui*), *input* (*xt*), *output* dari proses sebelumnya (*ht-1*), dan *bias input gate* (*bi*). Nilai *Wi, Ui, bi* untuk insialisasi awal bernilai *random*, nilai variabel *Wi, Ui, bi* dijabarkan sebagai berikut :

Langkah selanjutnya mengalikan *input*(*xt*)dengan *weight input gate* (*Wi*), langkah ini dijabarkan sebagai berikut :

Selanjutnya, nilai *recurrent weight input gate* (*Ui*)dikalikan dengan *output* dari proses sebelumnya (*ht-1*), karena *output* dari proses sebelumnya (*ht-1*) tidak ada maka nilainya 0, sehingga hasil *recurrent weight input gate* (*Ui*)dikalikan dengan *output* dari proses sebelumnya (*ht-1*) adalah 0, hasil tersebut dijabarkan sebagai berikut :

Nilai hasil dan kemudian ditambah dengan nilai *bias input gate* (*bi*), proses ini dijabarkan sebagai berikut :

Nilai *input gate* (*it*) didapatkan dengan menghitung nilai dengan fungsi aktivasi *sigmoid*. Fungsi aktivasi *sigmoid* akan menghasilkan nilai antara 0 dan 1, proses perhitungan *input gate* (*it*) dijabarkan sebagai berikut :

1. Menghitung nilai kandidat baru ()

Proses selanjutnya menghitung nilai kandidat baru () menggunakan rumus pada persamaan 2.8, ada beberapa variabel yang digunakan dalam proses ini yaitu *weight* kandidat konteks (*Wc*), *recurrent weight* kandidat konteks (*Uc*), *input* (*xt*), *output* proses sebelumnya (*ht-1*), dan nilai *bias* kandidat konteks (*bc*). Nilai *Wc*, *Uc*, *bi* untuk inisialisasi awal adalah *random*. Nilai *Wc*, *Uc*, *bi* dijabarkan sebagai berikut :

Nilai kandidat baru didapatkan dengan menghitung nilai dengan fungsi aktivasi *tanh*, fungsi aktivasi *tanh* menghasilkan nilai antara 1 dan -1. Pertama mengalikan nilai *input* (*xt*) dengan nilai *weight* kandidat konteks (*Wc*), proses ini dijabarkan sebagai berikut :

Proses selanjutnya, nilai *recurrent weight* kandidat konteks (*Uc*)dikalikan dengan *output* dari proses sebelumnya (*ht-1*), karena *output* dari proses sebelumnya (*ht-1*) tidak ada maka nilainya 0, sehingga hasil *recurrent weight* kandidat konteks(*Uc*)dikalikan dengan *output* dari proses sebelumnya (*ht-1*) adalah 0, hasil tersebut dijabarkan sebagai berikut :

Proses selanjutnya, hasil dari proses dan ditambahkan dengan nilai *bias* kandidat konteks (), penjabaran proses ini dijabarkan sebagai berikut :

Hasil dari proses kemudian dihitung menggunakan fungsi aktivasi *tanh* untuk menghasilkan nilai kandidat baru (). Penjabaran perhitungan nilai kandidat baru () sebagai berikut :

z

1. Menghitung *cell state*

Langkah selanjutnya adalah memperbaharui *cell state* lama *Ct-1* menjadi *cell state* baru *Ct*. Nilai *cell state* baru dihitung menggunakan persamaan 2.9, karena nilai *cell state* sebelumnya belum ada, maka nilainya 0, nilai *cell state* lama dijabarkan sebagai berikut :

Langkah selanjutnya mengalikan nilai *cell state* lama dengan nilai *forget gate*, langkah ini dijabarkan sebagai berikut :

Nilai *input gate* dikalikan dengan nilai kadidat baru yang diperoleh pada proses sebelumnya, langkah ini dijabarkan sebagai berikut :

Nilai *cell state* dihitung dengan menjumlahkan hasil dari proses dan , penjabaran proses ini dijabarkan sebagai berikut :

1. Menghitung output gate (Ot)

Langkah selanjutnya adalah menghitung *output gate*, langkah ini dihitung menggunakan persamaan 2.10. Variabel yang dibutuhkan pada langkah ini yaitu *weight ouput gate (Wo*), *recurrent weight output gate* (*Uo*), *input* (*xt*), *output* dari proses sebelumnya (*ht-1*), dan *bias output gate* (*bo*). Nilai *Wo*, *Uo*, *bo* pada inisialisasi awal bernilai *random* yang dijabarkan sebagai berikut :

Langkah selanjutnya mengalikan *input* (*xt*) dengan *weight output gate* (*Wo*), penjabaran langkah ini sebagai berikut :

Selanjutnya mengalikan *output* proses sebelumnya dengan *recurrent weight output gate*, penjabaran proses pengalian sebagai berikut :

Selanjutnya nilai dari hasil proses dan ditambahkan dengan nilai *bias output gate*, penjabaran proses ini sebagai berikut :

Menghitung *output gate* didapatkan dengan menghitung dengan fungsi aktivasi *sigmoid*, dimana fungsi aktivasi ini akan menghasilkan nilai antara 0 sampai 1. Proses menghitung *output gate* dijabarkan sebagai berikut :

1. Menghitung output final (*ht*)

Langkah terakhir adalah menghitung *output final* (*ht*) menggunakan persamaan 2.11, pertama menghitung nilai *tanh*(Ct) yang dijabarkan sebagai berikut :

Selanjutnya, menghitung *output gate* dengan mengalikan *output gate* dengan , proses ini dijabarkan sebagai berikut :

Proses *input* kata “hore” menghasilkan *output final* , *output final* ini akan digunakan untuk proses *input* kata selanjutnya, yaitu kata “akhir”, hasil *output final* kata “akhir” akan digunakan pada proses *input* kata “timnas” dan begitu seterusnya. Hasil keseluruhan proses dari awal hingga akhir untuk kata [“hore”, “akhir”, “timnas”, “indonesia”, “menang”, “skor”] dapat dilihat pada tabel 3.5, tabel 3.6, tabel 3.7, tabel 3.8, tabel 3.9.

#### Tabel 3.5 Hasil *forget gate* (*ft*) tiap orde

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Orde(*t*) | *Input*(*xt*) | *Forget gate* (ft) | | | | |
| 0 | hore | 0,508226 | 0,491192 | 0,494533 | 0,500566 | 0,504109 |
| 1 | akhir | 0,539047 | 0,48413 | 0,497744 | 0,511704 | 0,498023 |
| 2 | timnas | 0,435014 | 0,479691 | 0,602584 | 0,496556 | 0,511039 |
| 3 | indonesia | 0,52241 | 0,511705 | 0,518073 | 0,530234 | 0,505841 |
| 4 | menang | 0,160368 | 0,54972 | 0,524729 | 0,532748 | 0,525823 |
| 5 | skor | 0,537819 | 0,557364 | 0.503936 | 0,567526 | 0,534936 |

#### Tabel 3.6 Hasil *input gate* (*it*) tiap orde

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Orde(*t*) | *Input*(*xt*) | *Input gate* (it) | | | | |
| 0 | hore | 0,499075 | 0,507994 | 0,484948 | 0,504131 | 0,500975 |
| 1 | akhir | 0,682631 | 0,531195 | 0,478373 | 0,499388 | 0,505449 |
| 2 | timnas | 0,474113 | 0,596975 | 0,494946 | 0,557532 | 0,484612 |
| 3 | indonesia | 0,486321 | 0,516592 | 0,504831 | 0,510159 | 0,504823 |
| 4 | menang | 0,541772 | 0,555008 | 0,513857 | 0,533343 | 0,535446 |
| 5 | skor | 0,163149 | 0,544335 | 0,079544 | 0,550523 | 0,54317 |

#### Tabel 3.7 Hasil Kandidat baru () tiap orde

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Orde(*t*) | Input(*xt*) | Kandidat baru() | | | | |
| 0 | hore | 0,469511 | 0,035555 | 0,021698 | 0,025733 | 0,013274 |
| 1 | akhir | 0,168775 | 0,475897 | -0,25554 | -0,081948 | 0,615148 |
| 2 | timnas | 0,051175 | 0,008288 | 0,178389 | 0,106098 | 0,065152 |
| 3 | indonesia | -0,05184 | 0,0211608 | 0,038442 | 0,111437 | 0,035598 |
| 4 | menang | 0,0905463 | 0,134515 | 0,0510476 | 0,159938 | 0,0945864 |
| 5 | skor | 0,004678 | 0,141765 | 0,125673 | 0,124838 | 0,157632 |

#### Tabel 3.8 Hasil *Cell state* (*Ct*) tiap orde

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Orde(*t*) | Input(*xt*) | Kandidat baru() | | | | |
| 0 | hore | 0,234321 | 0,018062 | 0,010522 | 0,012973 | 0,00665 |
| 1 | akhir | 0,241521 | 0,261538 | -0,117006 | 0,301301 | 0,526422 |
| 2 | timnas | 0,129328 | 0,130405 | 0,017903 | 0,208766 | 0,300595 |
| 3 | indonesia | 0,042351 | 0,07766 | 0,028682 | 0,167545 | 0,170024 |
| 4 | menang | 0,055847 | 0,117348 | 0,041281 | 0,174561 | 0,1400484 |
| 5 | skor | 0,030799 | 0,142573 | 0,030799 | 0,156694 | 0,160538 |

#### Tabel 3.9 Hasil *Output gate* (*Ot*) tiap orde

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Orde(*t*) | Input(*xt*) | Output gate (Ot) | | | | |
| 0 | hore | 0,50845 | 0,497704 | 0,499944 | 0,508844 | 0,483251 |
| 1 | akhir | 0,340574 | 0,514624 | 0,513813 | 0,390576 | 0,462021 |
| 2 | timnas | 0,52287 | 0,462506 | 0,293463 | 0,521102 | 0,518733 |
| 3 | indonesia | 0,522442 | 0,506951 | 0,529952 | 0,510473 | 0,495368 |
| 4 | menang | 0,535088 | 0,527644 | 0,544744 | 0,510951 | 0,502957 |
| 5 | skor | 0,508765 | 0,526352 | 0,511101 | 0,544413 | 0,51488 |

#### Tabel 3.10 Hasil *output final* (*ht*) tiap orde

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Orde(*t*) | Input(*xt*) | Output final (ht) | | | | |
| 0 | hore | 0,117579 | 0,008961 | 0,00526 | 0,006601 | 0,003213 |
| 1 | akhir | 0,080693 | 0,131606 | -0,059846 | 0,114244 | 0,232165 |
| 2 | timnas | 0,026734 | 0,003833 | 0,051802 | 0,055081 | 0,033749 |
| 3 | indonesia | 0,022112 | 0,039291 | 0,015196 | 0,084736 | 0,083422 |
| 4 | menang | 0,029852 | 0,061635 | 0,022475 | 0,088297 | 0,069981 |
| 5 | skor | 0,000948 | 0,074539 | 0,015736 | 0,084615 | 0,081954 |

Setelah mendapatkan *output final* (*ht*) langkah selanjutnya masuk ke *dense layer* untuk menentukan hasil klasifikasi emosi. *Dense layer* mendapatkan *input* dari hasil *output final* (*ht*) orde terakhir. Persamaan *dense layer* menurut Hochreiter dan Schmidhuber (1997) dapat dilihat pada persamaan 3.1.

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .(3.1)

Keterangan :  
 Y = dense layer  
 ht = output final  
 Wy = weight dense layer  
 by = bias dense layer

Nilai *weight dense layer* (*Wy*) dan nilai *bias* (*by*)dinisialisasi dengan nilai *random*, *weight dense layer* (*Wy*)dan *bias* (*by*)memiliki 6 *neuron* sesuai dengan jumlah target emosi. Nilai *weight dense layer* (*Wy*) dan nilai *bias* (*by*)dijabarkan sebagai berikut :

Perhitungan *dense layer* (*Y*) menggunakan persamaan 3.1, penjabaran proses perhitungan *dense layer* sebagai berikut :

Hasil *dense layer* kemudian dicari nilai *probabilitas* tertinggi, dari hasil *dense layer* nilai *probabilitas* tertinggi didapatkan pada index ke 0, sehingga pada kata “Hore, akhirnya timnas Indonesia menang dengan skor 2-0 !!” memiliki emosi bahagia.

### Pengujian

Tahap terakhir adalah pengujian sesuai dengan perancangan arsitektur yang dibangun dan *word embedding* yang dibandingkan, pengujian akan menggunakan *confusion matrix* untuk mendapatkan nilai *accuracy*, *precision*, *recall*,dan *f1-score*. Arsitektur terbaik dari masing-masing *word embedding* akan dibandingkan untuk mengetahui tujuan penelitian tercapai atau tidak serta menganalisis hasil dari penelitian ini, tabel pengujian perancangan arsitektur dan *word embedding* dapat dilihat pada tabel 3.11.

#### Tabel 3.11 Pengujian

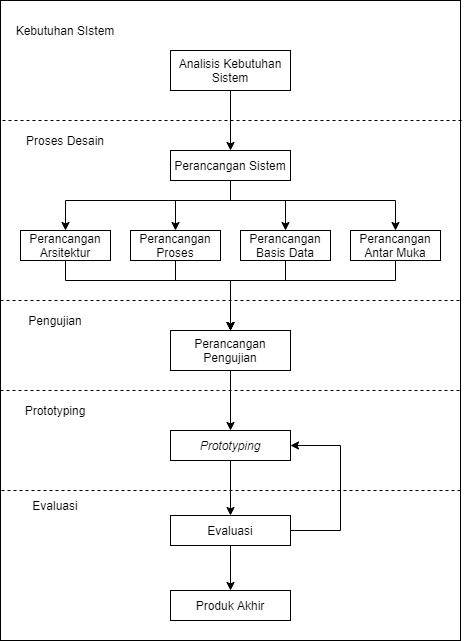
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *Word embedding* | *Unit* | *Dropout* | *accuracy* | *precission* | *recall* | *f1-score* |
| *Word2vec* | 50 | 20 |  |  |  |  |
| 30 |  |  |  |  |
| 50 |  |  |  |  |
| 100 | 20 |  |  |  |  |
| 30 |  |  |  |  |
| 50 |  |  |  |  |
| 150 | 20 |  |  |  |  |
| 30 |  |  |  |  |
| 50 |  |  |  |  |
| 200 | 20 |  |  |  |  |
| 30 |  |  |  |  |
| 50 |  |  |  |  |

#### Tabel 3.12 Pengujian (lanjutan)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *Word embedding* | *Unit* | *Dropout* | *accuracy* | *precission* | *recall* | *f1-score* |
| Glo-Ve | 50 | 20 |  |  |  |  |
| 30 |  |  |  |  |
| 50 |  |  |  |  |
| 100 | 20 |  |  |  |  |
| 30 |  |  |  |  |
| 50 |  |  |  |  |
| 150 | 20 |  |  |  |  |
| 30 |  |  |  |  |
| 50 |  |  |  |  |
| 200 | 20 |  |  |  |  |
| 30 |  |  |  |  |
| 50 |  |  |  |  |
| *Fasttext* | 50 | 20 |  |  |  |  |
| 30 |  |  |  |  |
| 50 |  |  |  |  |
| 100 | 20 |  |  |  |  |
| 30 |  |  |  |  |
| 50 |  |  |  |  |
| 150 | 20 |  |  |  |  |
| 30 |  |  |  |  |
| 50 |  |  |  |  |
| 200 | 20 |  |  |  |  |
| 30 |  |  |  |  |
| 50 |  |  |  |  |

## Metodologi Pengembangan Sistem

Metode pengembangan sistem yang digunakan pada penelitian ini adalah *prototyping*. *Prototyping* digunakan pada penelitian ini untuk mengembangkan model menjadi final, sehingga sistem yang dikembangkan lebih cepat dan lebih hemat dibandingkan metode pengembangan sistem lainnya. Metodologi pengembangan sistem yang digunakan pada penelitian ini dapat dilihat pada gambar 3.15.

Gambar 3.15 Metodologi Pengembangan Sistem

### Analisis Kebutuhan Sistem

Analisis kebutuhan sistem mengidentifikasi kebutuhan yang dibutuhkan sistem untuk mengembangkan sistem dengan cara mendefinisikan sistem kedalam bagian-bagian komponen yang lebih kecil. Analisis kebutuhan sistem dibagi menjadi 2, yaitu kebutuhan fungsional dan kebutuhan non-fungsional.

### Kebutuhan Fungsional

Kebutuhan yang berkaitan langsung dengan sistem dinamakan kebutuhan fungsional, kebutuhan fungsionalitas ini memuat gambaran umum yang dapat dilakukan oleh sistem, gambaran umum sistem yang akan dibuat pada penelitian ini sebagai berikut :

1. Sistem dapat memproses teks yang diinputkan dengan *Long Short Term Memory*(LSTM) dan *word embedding fasttext*.
2. Sistem dapat menghasilkan emosi yang cukup akurat.
3. Sistem dapat melakukan proses *login* dan *register*.
4. Sistem dapat menyimpan teks dan hasil emosi.
5. Sistem dapat menampilkan hasil rekap emosi.

### Kebutuhan Non-fungsional

Kebutuhan yang tidak berkaitan langsung dengan sistem dan menggambarkan properti yang dimiliki oleh sistem dinamakan kebutuhan non-fungsional. Kebutuhan non-fungsional terdiri dari kebutuhan perangkat keras (*hardware*) dan kebutuhan perangkat lunak (*software*) :

1. Kebutuhan perangkat keras (*hardware*)

Kebutuhan perangkat keras (*hardware*) yang dibutuhkan dan spesifikasi untuk pengembangan sistem pada penelitian ini dapat dilihat pada tabel 3.13.

#### Tabel 3.13 Kebutuhan perangkat keras (*hardware*)

|  |  |  |
| --- | --- | --- |
| No | Perangkat keras (*hardware*) | Keterangan |
| 1. | Processor | Intel Core i5-8265U 8th Gen |
| 2. | RAM | 4 GB DDR4 |
| 3. | Storage | Harddisk 500GB |

1. Kebutuhan perangkat lunak (*software*)

Kebutuhan perangkat lunak (*software*) yang dibutuhkan dan spesifikas untuk pengembangan sistem pada penelitian ini dapat dilihat pada tabel 3.14.

#### Tabel 3.14 Kebutuhan perangkat lunak (*software*)

|  |  |  |
| --- | --- | --- |
| No | Perangkat lunak (*software*) | Keterangan |
| 1. | OS system | Windows |
| 2. | Google colab | Notebook cloud |
| 3. | Chrome | Web browser |
| 4. | python | Bahasa pemprograman |
| 5. | XAMPP | DBMS |

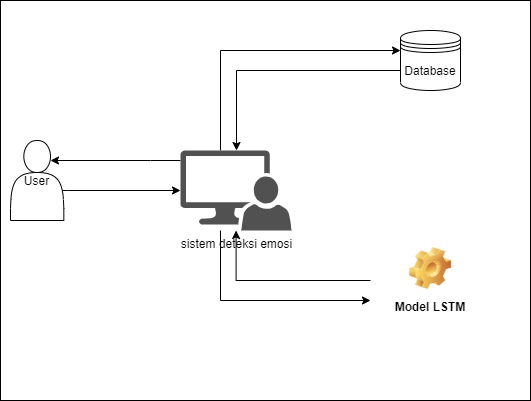
### Proses Desain

Proses desain akan menjelaskan proses perancangan sistem yang berguna untuk memberikan gambaran umum sisem sehingga mempermudah dalam membangun sistem. Proses ini terdiri dari perancangan sistem dan perancangan pengujian.

### Perancangan Sistem

Perancangan sistem membahas mengenai rancangan sistem yang akan dibangun pada penelitian ini. Perancangan sistem terdiri dari perancangan arsitektur, perancangan proses, perancangan basis data, dan perancangan antar muka. Penjelasan lebih detail mengenai perancangan arsitektur, perancangan proses, perancangan basis data, dan perancangan antar muka sebagai berikut :

1. **Perancangan Arsitektur**

Arsitektur sistem yang akan dibuat terdiri dari *user*, *database*, model LSTM, dan sistem deteksi emosi. *User* merupakan aktor yang dapat menginput teks pada sistem untuk mendeteksi emosi. Sistem deteksi emosi memproses teks yang diinput oleh *user* menggunakan model LSTM dan menampilkan hasil kepada *user*. Hasil deteksi emosi akan disimpan didalam *database*. Ilustrasi arsitektur sistem yang akan dibuat dapat dilihat pada gambar 3.16.

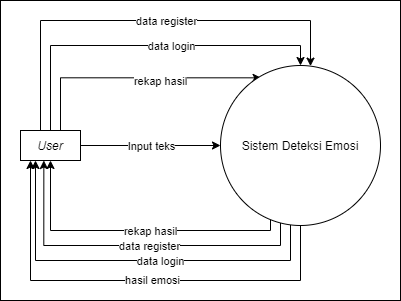
Gambar 3.16 Arsitektur sistem

1. **Perancangan proses**

Proses perancangan sistem pada penelitian ini menggunakan *Data Flow Diagram* (DFD) untuk menggambarkan aliran data pada sistem. DFD yang digunakan pada sistem ini sebagai berikut :

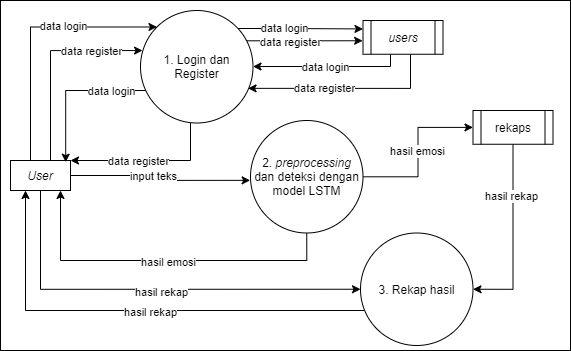
1. DFD level 0 (Diagram Context)

DFD level 0 atau *Diagram Context* menggambarkan sistem secara *basic* dan aliran *input* sampai *output* saja, oleh karena itu tidak ada *database*. Sistem ini hanya memiliki 1 entitas saja yaitu *user*. DFD level 0 pada sistem ini dapat dilihat pada gambar 3.17.

Gambar 3.17 DFD level 0

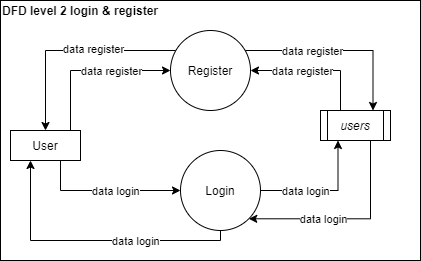
1. DFD level 1

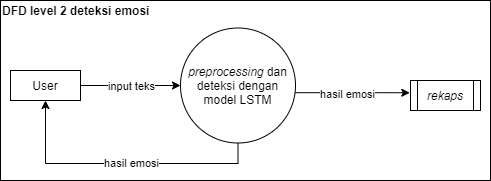
DFD level 1 menggambarkan aliran data yang lebih kompleks dari pada DFD level 1, dengan memecah beberapa proses pada DFD level 0. Sistem ini memilik 3 proses yaitu *login* dan *register*, *preprocessing* dan deteksi emosi dengan model LSTM, dan rekap hasil. DFD level 1 juga menggambarkan *database* yang akan digunakan sistem, pada sistem ini menggunakan 2 tabel *database* yaitu, *users* dan *rekaps*. DFD level 1 pada sistem ini dapat dilihat pada gambar 3.18.

Gambar 3.18 DFD level 1

1. DFD level 2

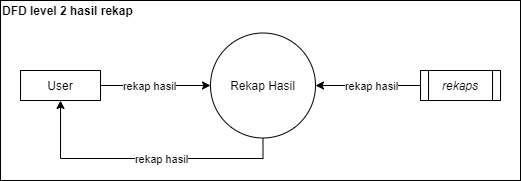
DFD level 2 menguraikan proses-proses pada DFD level 1 menjadi lebih rinci, terdapat 3 proses yang diuraikan pada DFD level 2 sistem ini, yaitu proses *login* dan *register*, proses *preprocessing* dan deteksi dengan model LSTM, dan proses rekap hasil. Proses *login* dan *register* akan *store* data *login* dan data *register*  ke *data base users*. DFD level 2 proses *login* dan *register* dapat dilihat pada gambar 3.19.

Gambar 3.19 DFD level 2 proses login dan register

DFD level 2 proses *preprocessing* dan deteksi emosi dengan model LSTM menggambarkan proses *preprocessing* dan deteksi emosi dengan model LSTM kemudian hasil emosi di *store* ke database *rekaps*. DFD level 2 proses *preprocessing* dan deteksi emosi dengan model LSTM dapat dilihat pada gambar 3.20.

Gambar 3.20 DFD level 2 proses *preprocessing* dan deteksi emosi dengan model LSTM

DFD level 2 proses rekap hasil menggambarkan proses *user request* rekap hasil, kemudian sistem mengambil data pada *data base rekaps* dan menampilkan ke *user*. DFD level 2 proses rekap hasil dapat dilihat pada gambar 3.21.

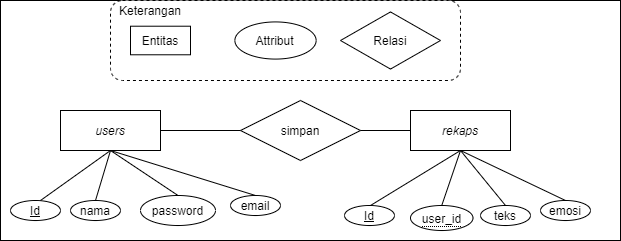
Gambar 3.21 DFD level 2 proses rekap hasil

1. **Perancangan *data base***

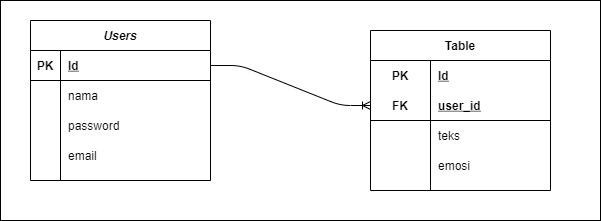
Perancangan *data base* beguna untuk menjabarkan *data base* yang digunakan pada sistem, sistem ini akan menggunakan *web server Apache* dan *Data base Management System* (DBMS) menggunakan *mySQL*. Perancangan *data base* menggunakan *Entity Relationship Diagram* (), Relasi Antar Tabel (RAT), dan Struktur Tabel. Penjabaran perancangan *data base* sebagai berikut :

1. Entity Relationship Diagram (ERD)

*Entity Relationship Diagram* (ERD) memvisualisasikan basis data saling terhubung yang dilambangkan dengan simbol-simbol. Simbol-simbol tersebut adalah entitas, atribut, dan relasi. Entitas pada perancangan *data base* sistem ini terdapat 2, yaitu *users* dan *rekaps*. Penjabaran ERD yang akan diterapkan pada sistem ini dapat dilihat pada gambar 3.22.

Gambar 3.22 Entity Relationship Diagram (ERD)

1. Relasi Antar Tabel (RAT)

Relasi Antar Tabel (RAT) merepresentasikan hubungan tabel satu ke tabel yang lain dengan menghubungkan *PRIMARY KEY* dengan *FOREIGN KEY*, RAT berguna untuk mengatur operasi pada suatu *data base*. RAT pada sistem yang akan dibangun dapat dilihat pada gambar 3.23.

Gambar 3.23 Relasi Antar Tabel

1. Struktur Tabel

Struktur tabel yang akan digunakan pada sistem ini sebagai berikut :

1. Tabel *Users*

Tabel *users* memiliki 4 kolom yaitu id, nama, password, email. Penjabaran mengenai tabel *users* dapat dilihat pada tabel 3.15.

#### Tabel 3.15 Tabel *users*

|  |  |  |  |
| --- | --- | --- | --- |
| Kolom | Tipe data | Constraints | Keterangan |
| Id | Integer(11) | Primary key , Auto Increment | Id user |
| nama | varchar(191) |  | nama user |
| password | varchar(191) |  | Password *user* |
| email | varchar(191) |  | email *user* |

1. Tabel *Rekaps*

Tabel *rekaps* memiliki 4 kolom yaitu id, user\_id, teks, emosi. Penjabaran mengenai tabel *rekaps* dapat dilihat pada tabel 3.16.

#### Tabel 3.16 Tabel *rekaps*

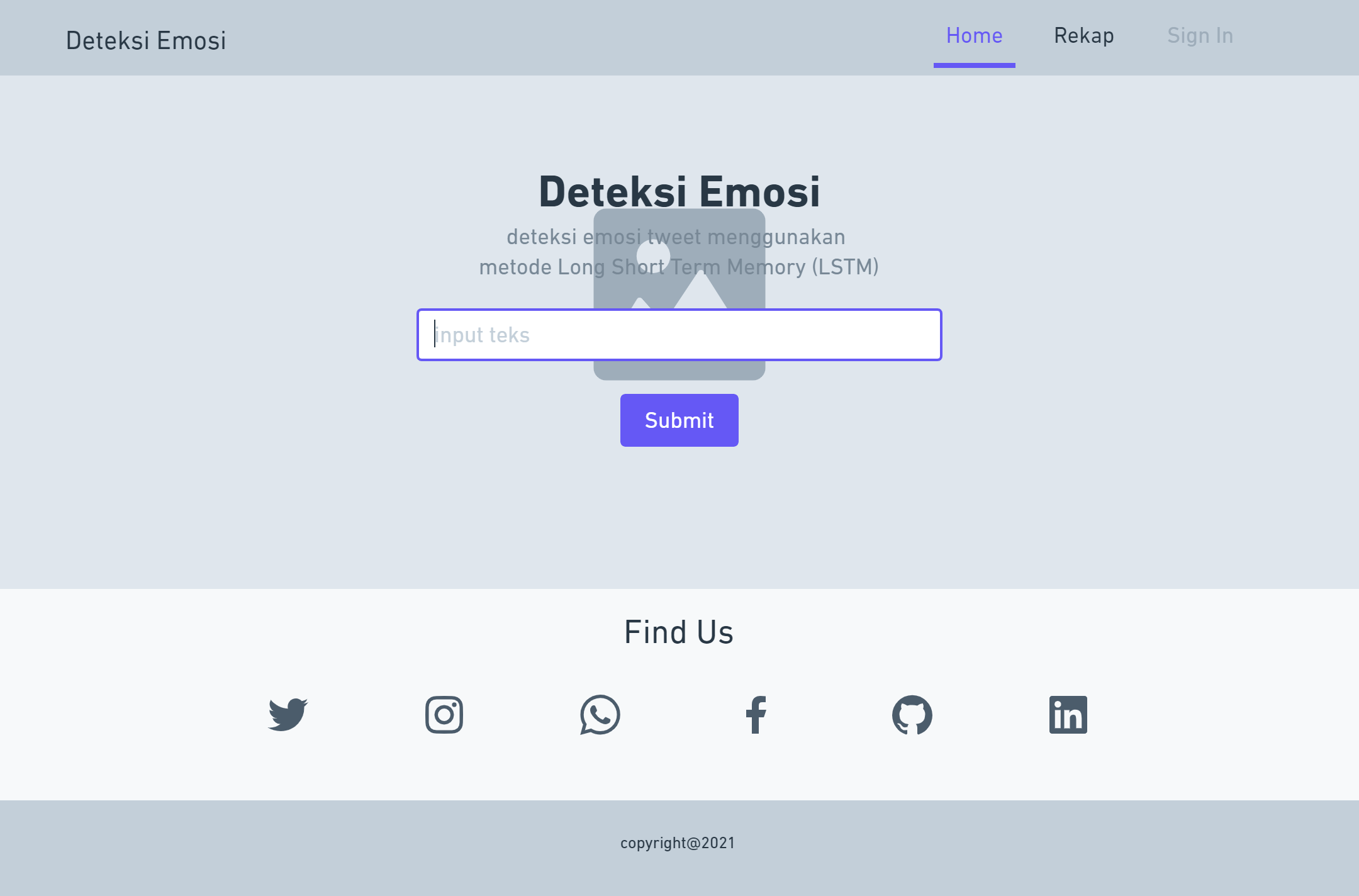
|  |  |  |  |
| --- | --- | --- | --- |
| Kolom | Tipe data | Constraints | Keterangan |
| Id | Integer(11) | Primary key , Auto Increment | Id rekaps |
| user\_id | Integer(11) | Foreign key | Id user |
| teks | varchar(191) |  | teks rekaps |
| emosi | varchar(191) |  | emosi rekaps |

1. **Perancangan antar muka**

Perancangan antar muka merupakan mekanisme komunikasi antara user dengan sistem. Perancangan antar muka yang akan diterapkan pada sistem ini terdiri dari beberapa bagian sebagai berikut :

1. Halaman Home

Halaman ini merupakan halaman yang akan diakses pertama kali oleh user, terdiri dari form untuk menginput teks dan button untuk mengetahui hasil emosi.

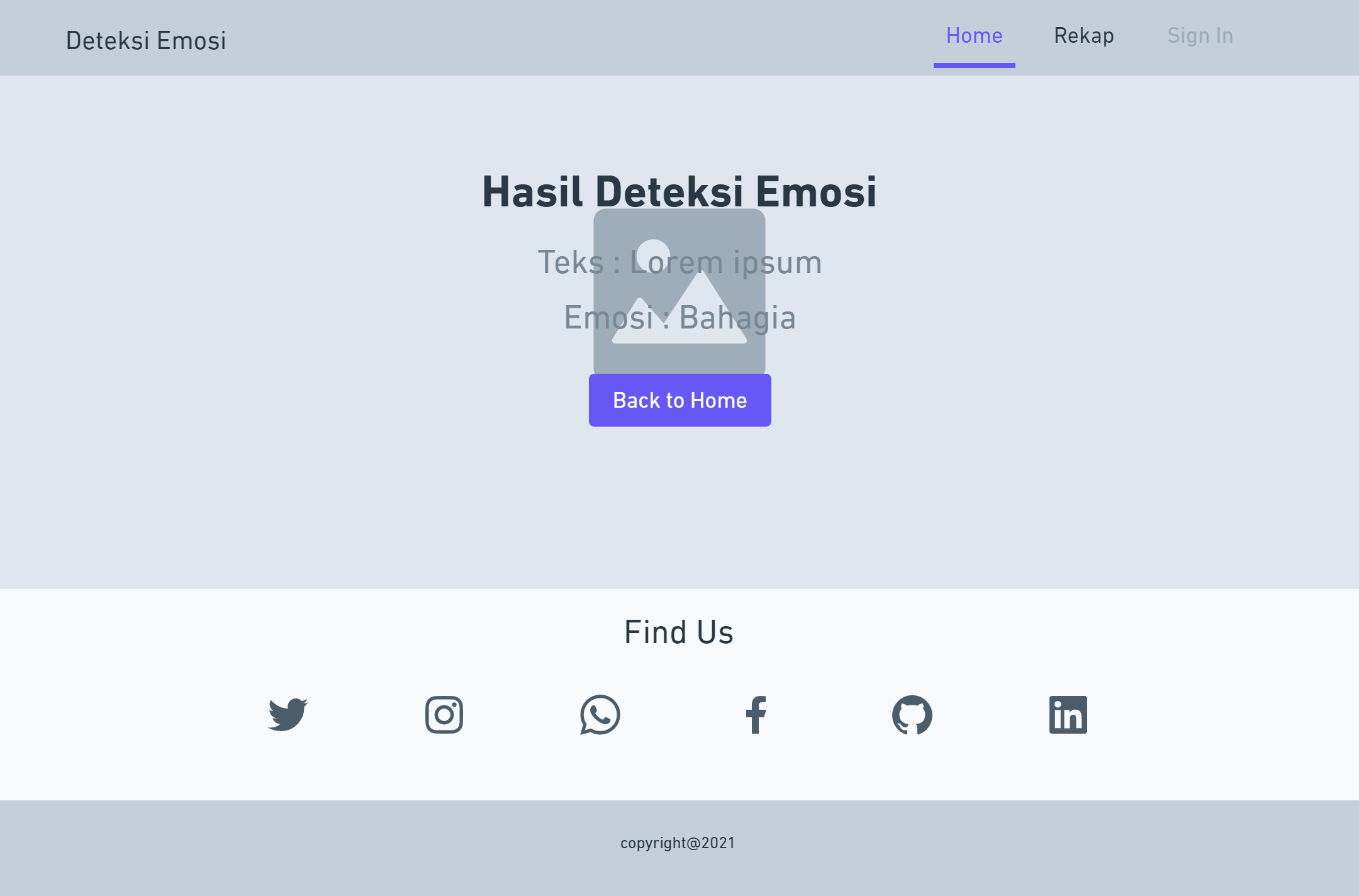


Gambar 3.24 Halaman Home

1. Halaman Hasil

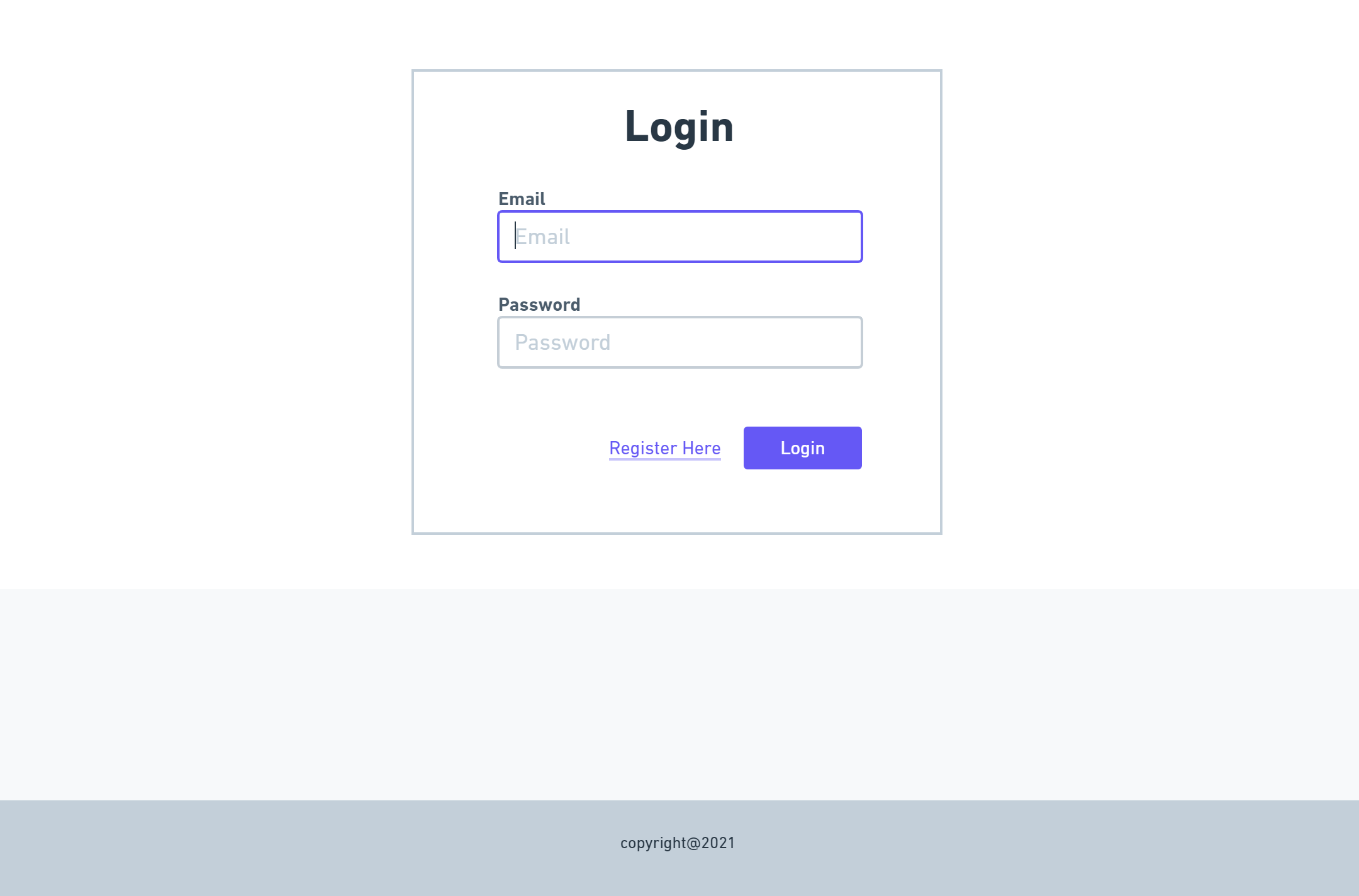
Halaman ini akan menampilkan teks yang diinput pada halaman home serta hasil emosi bedasarkan model LSTM.

Gambar 3.25 Halaman Hasil



1. Halaman Login

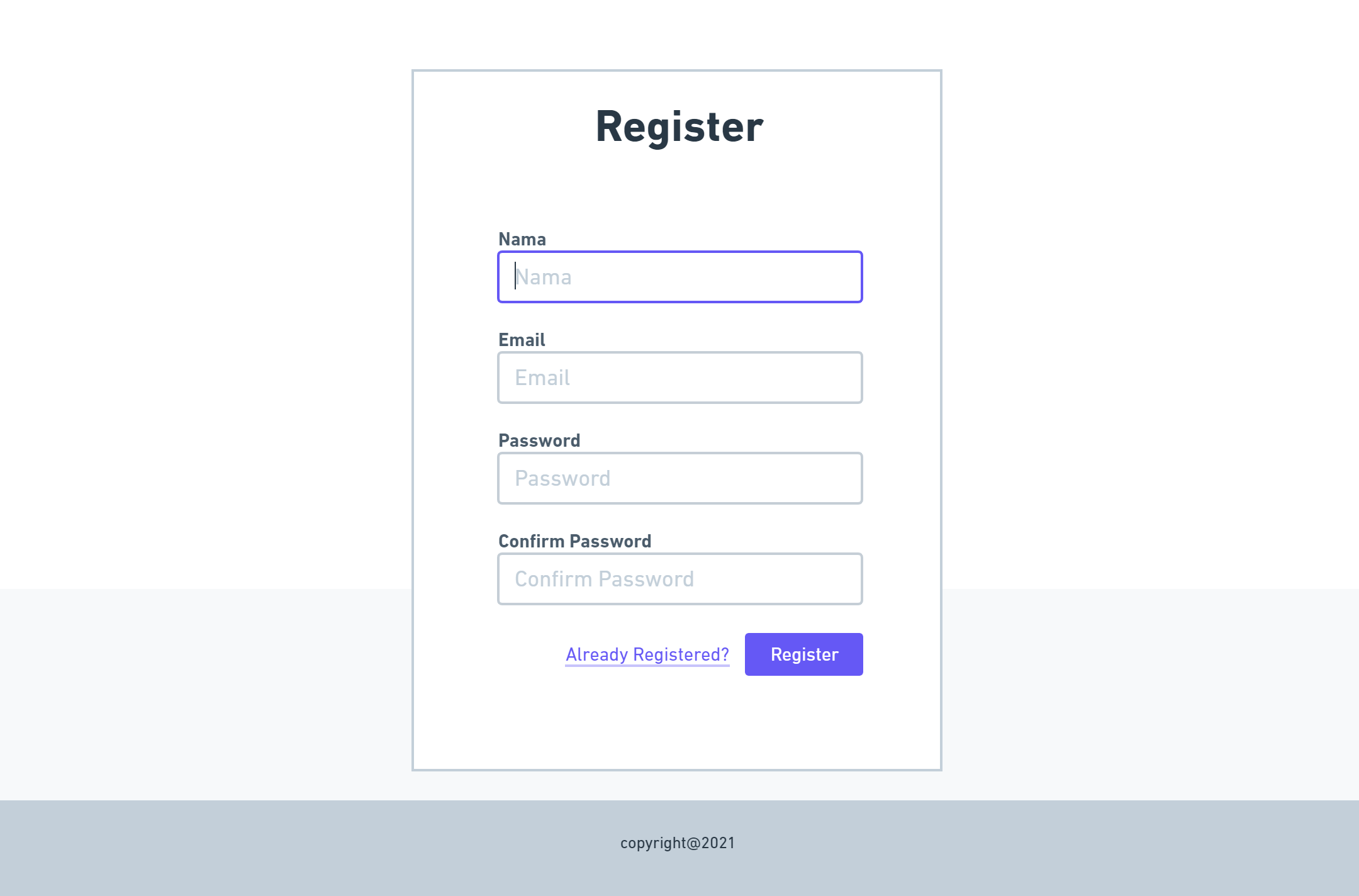
User dapat melakukan login jika sudah memiliki akun melalui halaman login dengan menginput email dan password dengan benar.



Gambar 3.26 Halaman Login

1. Halaman Register

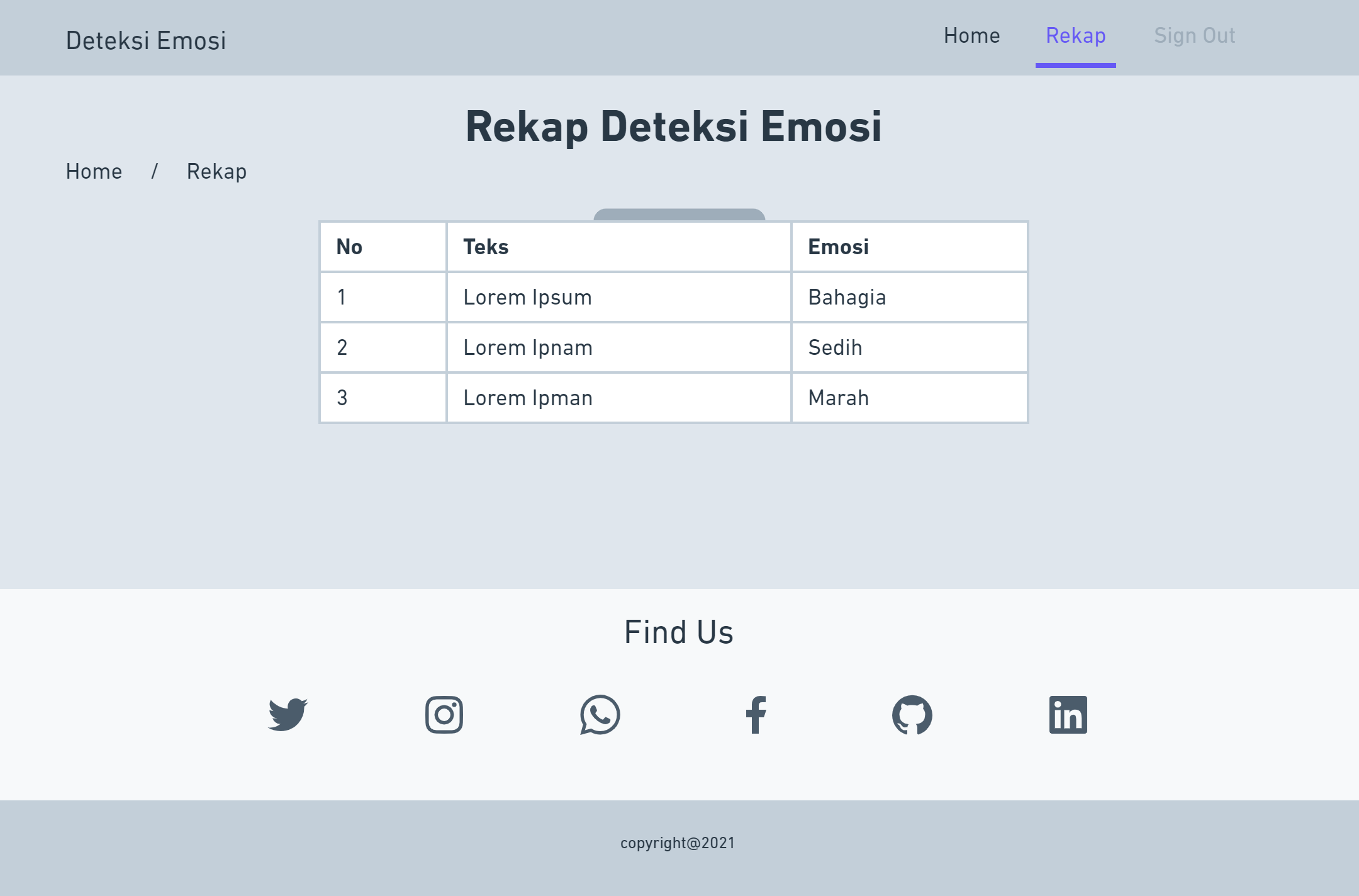
User yang belum memiliki akun dapat mendaftarkan akun terlebih dahulu melalui halaman register dengan menginput nama, email, password, dan konfirmasi password.



Gambar 3.27 Halaman Register

1. Halaman Rekap Hasil

User dapat melihat hasil rekap terhadap teks yang telah diinput dengan mengakses halaman rekap hasil, rekap hasil disajikan dalam bentuk tabel.



Gambar 3.28 Halaman Hasil

### Perancangan Pengujian

Perancangan pengujian berguna untuk menguji sistem terhadap kesesuaian rancangan yang telah dibuat. Pengujian sistem yang digunakan pada penelitian ini adalah metode *black box testing*, yaitu pengujian yang menekankan fungsionalitas tanpa mengetahui *coding* dari sistem tersebut. *Black box testing* bertujuan untuk mengukur kinerja dari sistem yang telah dibangun. Detail perancangan pengujian sistem dapat dilihat pada tabel 3.17.

#### Tabel 3.17 Detail perancangan pengujian sistem

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Halaman | Pengujian | Hasil | |
| Berhasil | Gagal |
| 1. | Halaman utama | *User* dapat menginput teks |  |  |
| Sistem dapat melakukan validasi input |  |  |
| 2. | Halaman hasil | Sistem dapat menampilkan hasil deteksi emosi |  |  |
| 3. | Halaman login | *User* dapat melakukan login |  |  |
| Sistem dapat melakukan validasi input |  |  |
| 4. | Halaman Register | *User* dapat melakukan register |  |  |
| Sistem dapat melakukan validasi input |  |  |
| 5. | Halaman Rekap | Sistem dapat menampilkan hasil rekap |  |  |

## Jadwal Penelitian

Jadwal penelitian yang dibuat sesuai dengan metodologi penelitian dan metodologi pengembangan sistem, jadwal penelitian lebih detail dapat dilihat pada tabel 3.18 dan tabel 3.19.

#### Tabel 3.18 Jadwal penelitian

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Kegiatan | April | | | | Mei | | | | Juni | | | | Juli | | | | Agustus | | | | September | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| 1 | Pengumpulan data | | | | | | | | | | | | | | | | | | | | | | | | |
| Scrapping *twitter* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 | Labelling | | | | | | | | | | | | | | | | | | | | | | | | |
| Labelling Data |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | Preprocessing | | | | | | | | | | | | | | | | | | | | | | | | |
| Case Folding |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Remove Punctuation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Remove Number |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Tokenizing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Stopword Removal |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Stemming |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

#### Tabel 3.19 Jadwal Penelitian (lanjutan)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 4 | Pengujian Arsitektur | | | | | | | | | | | | | | | | | | | | | | | | |
| Word Embedding (word2vec, Glo-Ve, fasttext) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Inisialiasi Unit |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| LSTM |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5 | Pengujian | | | | | | | | | | | | | | | | | | | | | | | | |
| Pengujian Confussion Matrix |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6 | Analisis Kebutuhan Sistem | | | | | | | | | | | | | | | | | | | | | | | | |
| Analisis kebutuhan sistem |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 7 | Proses Desain | | | | | | | | | | | | | | | | | | | | | | | | |
| Perancangan Arsitektur |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Perancangan Proses |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Perancangan Antar Muka |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 8 | Pengujian Sistem | | | | | | | | | | | | | | | | | | | | | | | | |
| Perancangan Pengujian |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 9 | Prototyping | | | | | | | | | | | | | | | | | | | | | | | | |
| Prototyping |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 10 | Evaluasi | | | | | | | | | | | | | | | | | | | | | | | | |
| Evaluasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Produk Akhir |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# DAFTAR PUSTAKA

Ahmad, A. (2017). *Mengenal Artificial Intelligence, Machine Learning, Neural Network, dan Deep Learning*. *October*.

Alfariqi, F., Maharani, W., & Husen, J. H. (2020). *Klasifikasi Sentimen pada Twitter dalam Membantu Pemilihan Kandidat Karyawan dengan Menggunakan Convolutional Neural Network dan Fasttext Embeddings Pendahuluan*. *7*(2), 8052–8062.

Ardiada, D., Sudarma, M., & Giriantari, D. (2019). *Text Mining pada Sosial Media untuk Mendeteksi Emosi Pengguna Menggunakan Metode Support Vector Machine dan K-Nearest Neighbour*. *18*(1), 55–60.

Arliyanti Nurdin, Bernadus Anggo Seno Aji, Anugrayani Bustamin, Z. A. (2020). Perbandingan kinerja Word Embedding Word2Vec, Glove dan FastText pada klasifikasi teks. *Jurnal TEKNOKOMPAK*, *14*(2), 74--79.

Bata, J., Suyoto, & Pranowo. (2015). Leksikon Untuk Deteksi Emosi Dari Teks Bahasa Indonesia. *Seminar Nasional Informatika 2015 (SemnasIF 2015)*, *2015*(November), 195–202.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). *Enriching Word Vectors with Subword Information*. http://www.isthe.com/chongo/tech/comp/fnv

Consoli, D. (2009). Emotions That Influence Purchase Decisions And Their Electronic Processing. *Annales Universitatis Apulensis Series Oeconomica*, *2*(11), 1–45.

Dandannavar, P. S., Mangalwede, S. R., & Kulkarni, P. M. (2018). Social Media Text - A Source for Personality Prediction. *Proceedings of the International Conference on Computational Techniques, Electronics and Mechanical Systems, CTEMS 2018*, 62–65. https://doi.org/10.1109/CTEMS.2018.8769304

Daouas, T., & Lejmi, H. (2018). Emotions recognition in an intelligent elearning environment. *Interactive Learning Environments*, *26*(8), 991–1009. https://doi.org/10.1080/10494820.2018.1427114

Dwi L, Adinda., Harijanto, Budi., Rahutomo, Faisal. (2020). *Implementasi Deep Learning Untuk Deteksi Ekspresi Emosi pada Twitter*.

Faadilah, A. (2020). *Tokopedia Di Google Play Store*.

Fanesya, F., Wihandika, R. C., & Indriati. (2019). *Deteksi Emosi Pada Twitter Menggunakan Metode Naïve Bayes Dan Kombinasi Fitur*. *3*(7), 6678–6686.

Gaind, B., Syal, V., & Padgalwar, S. (2019). *Emotion Detection and Analysis on Social Media*. http://arxiv.org/abs/1901.08458

Gholamy, A., Kreinovich, V., & Kosheleva, O. (2018). Why 70/30 or 80/20 Relation Between Training and Testing Sets: A Pedagogical Explanation. In *Part of the Computer Sciences Commons Comments*. https://digitalcommons.utep.edu/cs\_techrephttps://digitalcommons.utep.edu/cs\_techrep/1209

Haddi, E., Liu, X., & Shi, Y. (2013). The role of text pre-processing in sentiment analysis. *Procedia Computer Science*, *17*, 26–32. https://doi.org/10.1016/j.procs.2013.05.005

Halim, K., Novianus Palit, H., & Tjondrowiguno, A. N. (2020). Penerapan Recurrent Neural Network untuk Pembuatan Ringkasan Ekstraktif Otomatis pada Berita Berbahasa Indonesia. *Jurnal Infra*, *8*(1), 221–227.

Han, J., & Kamber, M. (n.d.). *Data Mining: Concepts and Techniques*.

Haryadi, D., & Kusuma, G. P. (2019). Emotion detection in text using nested Long Short-Term Memory. *International Journal of Advanced Computer Science and Applications*, *10*(6), 351–357. https://doi.org/10.14569/ijacsa.2019.0100645

Hochreiter, Sepp dan Schmidhber, Jurgen. (1997). Long Short-Term Memory. https://doi.org/10.1162/neco.1997.9.8.1735

Jabreel, M., & Moreno, A. (2019). A deep learning-based approach for multi-label emotion classification in Tweets. *Applied Sciences (Switzerland)*, *9*(6). https://doi.org/10.3390/app9061123

Josi, A., & Andretti Abdillah, L. (n.d.). *PENERAPAN TEKNIK WEB SCRAPING PADA MESIN PENCARI ARTIKEL ILMIAH*.

Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). *Bag of Tricks for Efficient Text Classification*. https://github.com/facebookresearch/fastText

Juwiantho, H., Setiawan, E. I., Santoso, J., & Purnomo, M. H. (2020). Sentiment Analysis Twitter Bahasa Indonesia Berbasis Word2vec Menggunakan Deep Convolutional Neural Network. *Jurnal Teknologi Informasi Dan Ilmu Komputer (JTIIK)*, *7*(1), 181–188. https://doi.org/10.25126/jtiik.202071758

Kotler, P., Keller, K. L., Hall, P., Columbus, B., New, I., San, Y., Upper, F., River, S., Cape, A., Dubai, T., Madrid, L., Munich, M., Montreal, P., Delhi, T., Sao, M. C., Sydney, P., Kong, H., Singapore, S., & Tokyo, T. (2012). *Marketing Management*.

Kowalska, M., & Wróbel, M. (2017). Basic Emotions. In *Encyclopedia of Personality and Individual Differences* (pp. 1–6). https://doi.org/10.1007/978-3-319-28099-8\_495-1

Lim, E., Istts, T. I., Setiawan, E. I., & Istts, T. I. (2020). *Stance Classification Post Kesehatan di Media Sosial Dengan FastText Embedding dan Deep Learning*. 65–73.

Miedema, F. (2018). *Sentiment Analysis with Long Short-Term Memory networks*.

Nazief, B. A. A. & Adriani, M. (2005). Confix- stripping: Approach to Stemming Algorithm for Bahasa Indonesia. *Conferences in Research and Practice in Information Technology Series*, *38*(4).

Nurfita, R. D., & Ariyanto, G. (2018). Implementasi Deep Learning Berbasis Tensorflow Untuk Pengenalan Sidik Jari. *Emitor: Jurnal Teknik Elektro*, *18*(01), 22–27. https://doi.org/10.23917/emitor.v18i01.6236

Nurlaila, A., Wiranto, & Saptono, R. (2017). Classification Of Customers Emotion Using Naive Bayes Classifier ( Case Study : Natasha Skin Care ). *Jurnal Ilmiah Teknologi Dan Informasi*, *6*(2), 92–97.

Nurrohmat, M. A., & SN, A. (2019). Sentiment Analysis of Novel Review Using Long Short-Term Memory Method. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, *13*(3), 209. https://doi.org/10.22146/ijccs.41236

Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, *June 2018*, 1532–1543. https://doi.org/10.3115/v1/d14-1162

Picard, R. W. (1997). *Affective Computing*. http://www.media.mit.edu/˜picard/

Pressman, Roger S., B. R. M. (2015). Software Engineering A Practitioner’s Approach. In *McGraw-Hill Education, 2 Penn Plaza, New York, NY 10121* (Vol. 1, Issue 4). www.mhhe.com/pressman.

Rao, A., & Spasojevic, N. (2016). *Actionable and Political Text Classification using Word Embeddings and LSTM*. http://arxiv.org/abs/1607.02501

Rohman, A. N., Handayani, R. D., & Kusrini, K. (2020). Deteksi Emosi Media Sosial Menggunakan Term Frequency-Inverse Document Frequency. *CSRID (Computer Science Research and Its Development Journal)*, *11*(3), 140–148.

Rohman, A. N., Utami, E., & Raharjo, S. (2019). *View of Deteksi Kondisi Emosi pada Media Sosial Menggunakan Pendekatan Leksikon dan Natural Language Processing*. Jurnal Eksplora Informatika. https://doi.org/https://doi.org/10.30864/eksplora.v9i1.277

Rong, X. (2016). *word2vec Parameter Learning Explained*. http://bit.ly/wevi-online.

S. B. Kotsiantis, D. K. and P. E. P. (2006). *Data Preprocessing for Supervised Leaning* .

Saifulloh, M., & Siregar, M. U. (2019). Pengungkapan Diri Gofar Hilman Sebagai Influencer Melalui Media Instagram. *Pustaka Komunikasi*, *2*(2), 167–180. https://doi.org/https://doi.org/10.32509/pustakom.v2i2.869

Salam, A., Zeniarja, J., & Khasanah, R. S. U. (2018). Analisis Sentimen Data Komentar Sosial Media Facebook Dengan K-Nearest Neighbor (Studi Kasus Pada Akun Jasa Ekspedisi Barang J&T Ekpress Indonesia). *Prosiding SINTAK*, 480–486.

Saputri, M. S., Mahendra, R., & Adriani, M. (2019). Emotion Classification on Indonesian Twitter Dataset. *Proceedings of the 2018 International Conference on Asian Language Processing, IALP 2018*, *November*, 90–95. https://doi.org/10.1109/IALP.2018.8629262

Sari, B. W., & Haranto, F. F. (2019). Implementasi Support Vector Machine Untuk Analisis Sentimen Pengguna Twitter Terhadap Pelayanan Telkom Dan Biznet. *Jurnal Pilar Nusa Mandiri*, *15*(2), 171–176. https://doi.org/10.33480/pilar.v15i2.699

Sebastiani, F. (2002). Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, *34*(1), 1–47. https://doi.org/10.1145/505282.505283

Setiawan, R. A., & Setyohadi, D. B. (2017). Analisis Komunikasi Sosial Media Twitter sebagai Saluran Layanan Pelanggan Provider Internet dan Seluler di Indonesia. *Journal of Information Systems Engineering and Business Intelligence*, *3*(1), 16. https://doi.org/10.20473/jisebi.3.1.16-25

Temizel, C., Canbaz, C. H., Saracoglu, O., Putra, D., Baser, A., Erfando, T., Krishna, S., & Saputelli, L. (2020). *Production Forecasting in Shale Reservoirs Using LSTM Method in Deep Learning*. *July*. https://doi.org/10.15530/urtec-2020-2878

Turland. (2010). *php|architect’s Guide to Web Scraping with PHP | php[architect]*. https://www.phparch.com/books/phparchitects-guide-to-web-scraping-with-php/

Utomo, B. (2020). *Kinerja Deep Learning dalam Analisis Sentimen*.

Vijay Gaikwad, S. (2014). Text Mining Methods and Techniques. In *International Journal of Computer Applications* (Vol. 85, Issue 17).

Wilson, C., Sala, A., Puttaswamy, K. P. N., & Zhao, B. Y. (2012). Beyond social graphs: User interactions in online Social networks and their implications. *ACM Transactions on the Web*, *6*(4). https://doi.org/10.1145/2382616.2382620

Xing, W., & Du, D. (2019). Dropout Prediction in MOOCs: Using Deep Learning for Personalized Intervention. *Journal of Educational Computing Research*, *57*(3), 547–570. https://doi.org/10.1177/0735633118757015

# LAMPIRAN