**PROPOSAL TUGAS AKHIR**

**SENTIMENT ANALYSIS UJARAN KEBENCIAN TWITTER MENGGUNAKAN PENDEKATAN BIDIRECTIONAL LSTM**



Oleh:

Adrianus Charlie Hadirria Agelsadewa

195314174

**Program Studi Informatika**

**Fakultas Sains dan Teknologi**

**Universitas Sanata Dharma**

**Yogyakarta**

**2022**

**HALAMAN ABSTRAK DAN PENGESAHAN PEMBIMBING**

**<Judul>**

# Abstrak

|  |
| --- |
| Isi bagian ini dengan abstrak dari proposal tugas akhir anda.  Abstrak akan berisi paparan singkat mengenai latar belakang persoalan yang diangkat, rumusan masalah, tujuan penelitian, rencana data yang akan digunakan, dan rencana langkah penelitian. |

**Disetujui pada tanggal : <isikan tanggal pengesahan>**

**oleh:**

**Pembimbing Tugas Akhir**

**<tanda tangan calon dosen pembimbing>**

**<Nama Dosen>**

Daftar Isi

[Abstrak 2](#_Toc128565231)

[Bab I. Pendahuluan 6](#_Toc128565232)

[I.1. Latar Belakang 6](#_Toc128565233)

[I.2. Rumusan Masalah 9](#_Toc128565234)

[I.3. Tujuan 9](#_Toc128565235)

[I.4. Manfaat Penelitian 9](#_Toc128565236)

[I.5. Batasan Masalah 9](#_Toc128565237)

[I.6. Sistematika Penulisan 10](#_Toc128565238)

[BAB II TINJAUAN PUSTAKA DAN LANDASAN TEORI 12](#_Toc128565239)

[II.1. Tinjauan Pustaka 12](#_Toc128565242)

[II.2. Landasan Teori 17](#_Toc128565246)

[II.2.1 Sentiment Analysis 17](#_Toc128565247)

[II.2.2 Jaringan Syaraf Tiruan 18](#_Toc128565248)

[II.2.3 Deep Learning 22](#_Toc128565249)

[II.2.4 Long Short-Term Memory (LSTM) 34](#_Toc128565250)

[II.2.5 Bidirectional Long Sshort-Term Memory (LSTM) 37](#_Toc128565251)

[II.2.6 Preprocessing 38](#_Toc128565252)

[BAB III METODE PENELITIAN 40](#_Toc128565253)

[III.1. Tahapan Penelitian 40](#_Toc128565255)

[III.2. Pengumpulan Data 40](#_Toc128565256)

[III.3. Preprocessing 43](#_Toc128565257)

[III.3.1 Lowercasing 44](#_Toc128565258)

[III.3.2 Pembersihan Karakter Non-Alphanumeric 44](#_Toc128565259)

[III.3.3 Pembersihan Karakter Khusus 45](#_Toc128565260)

[III.3.4 Normalisasi Kata-Kata “Alay” 46](#_Toc128565261)

[III.3.5 Stemming 47](#_Toc128565262)

[III.3.6 Stop Word Removal 49](#_Toc128565263)

[III.4. Penerjemahan Data 50](#_Toc128565264)

[III.5. Labelling Data 51](#_Toc128565265)

[III.6. Pemisahan Data 53](#_Toc128565266)

[III.6.1 Tokenizing 54](#_Toc128565267)

[III.7. Perancangan Model 57](#_Toc128565268)

[III.7.1 Embedding Layer 58](#_Toc128565269)

[III.7.2 Dense Layer 59](#_Toc128565270)

[III.7.3 Flatten Layer 60](#_Toc128565271)

[III.7.4 LSTM Layer 60](#_Toc128565272)

[III.7.5 Bidirectional LSTM Layer 66](#_Toc128565273)

[III.8. Evaluasi Model 66](#_Toc128565274)

[III.9. Skenario Pengujian 67](#_Toc128565275)

[III.9.1 Layer Model 67](#_Toc128565276)

[III.9.2 Activation Function 68](#_Toc128565277)

[III.9.3 Optimizer 68](#_Toc128565278)

[III.9.4 Loss Function 68](#_Toc128565279)

[III.9.5 Skenario Simpel Model 1 68](#_Toc128565280)

[III.9.6 Skenario Simpel Model 2 69](#_Toc128565281)

[III.9.7 Skenario Kompleks Model 1 70](#_Toc128565282)

[III.9.8 Skenario Kompleks Model 2 70](#_Toc128565283)

[III.9.9 Pengujian Skenario 71](#_Toc128565284)

[BAB IV IMPLEMENTASI DAN HASIL PENELITIAN 72](#_Toc128565285)

[IV.1. Pengumpulan Data 72](#_Toc128565287)

[IV.2. Preprocessing Data 74](#_Toc128565288)

[IV.2.1 Lowercasing 74](#_Toc128565289)

[IV.2.2 Pembersihan Karakter Non-Alphanumeric 75](#_Toc128565290)

[IV.2.3 Pembersihan Karakter Khusus 76](#_Toc128565291)

[IV.2.4 Normalisasi Kata-Kata Alay 77](#_Toc128565292)

[IV.2.5 Stemming 78](#_Toc128565293)

[IV.2.6 Stopword Removal 79](#_Toc128565294)

[IV.3. Penerjemahan Data 81](#_Toc128565295)

[IV.4. Labelling Data 82](#_Toc128565296)

[IV.5. Tokenizing Data 84](#_Toc128565297)

[IV.6. Pemisahan Data 86](#_Toc128565298)

[Daftar Pustaka 87](#_Toc128565299)

[References 87](#_Toc128565300)

[Jadwal Kerja 89](#_Toc128565301)

# Pendahuluan

Bagian ini akan berisi mengenai latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi penelitian dan sistematika penulisan.

## Latar Belakang

Pada zaman sekarang, informasi dan teknologi sudah berkembang pesat, tak terkecuali di Indonesia. Di Indonesia sendiri, sekarang sudah memasuki industry 4.0 dimana semuanya sudah serba digital, informasi bisa berkembang cepat melalui media digital hanya dalam hitungan detik saja. Dilansir dari (Rosyid, 2022) pengguna media sosial di Indonesia mencapai 191 Juta per Januari 2022. Hal ini membuktikan bahwa mayoritas penduduk Indonesia sudah menggunakan media sosial untuk mendapatkan serta berbagi informasi melalui media sosial.

Saat ini, banyak sekali ditemukan tweets di media sosial Twitter yang mengandung ujar kebencian salah satunya oleh pengguna Twitter di Indonesia, khususnya yang membahas tentang hacker Bjorka akhir-akhir ini. Pengguna Twitter ini membaca, menerima dan bahkan mengirimkan ujar kebencian di media sosial terkait dengan trending topic tersebut. Ujaran kebencian *(hate speech)* merupakan tindakan komunikasi yang dilakukan oleh individu atau kelompok tertentu dalam bentuk provokasi, hasutan, hinaan, penistaan, pencemaran nama baik, serta penyebaran berita bohong dalam aspek seperti ras, warna kulit, gender, etnis, cacat fisik, orientasi seksual, kewarganeraaan, agama, dan lain-lain (Permatasari & Subyantoro, 2020). Penyebab dari ujar kebencian sendiri adalah kesalahpahaman dan emosi dalam menerima suatu informasi, tidak sependapat dengan orang lain, dan adanya kebencian pribadi terhadap orang lain. Ujar kebencian bisa menyebabkan seseorang menjadi tidak nyaman, bahkan sampai bisa menimbulkan perpecahan politik. Selain itu, ujar kebencian juga dapat membuat sebuah lingkungan di masyarakat terbagi menjadi beberapa kelompok. Untuk itu, sebagai pengguna sosial media, khususnya Twitter, perlu adanya sebuah urgensi untuk mengetahui tweet yang mengandung unsur ujar kebencian atau tidak di tengah ramainya topik dan pembicaraan tentang hacker Bjorka yang menjadi trending topik Twitter.

Pengguna media sosial Twitter di Indonesia kurang lebih sudah mencapai 18,45 juta (Rizaty, 2022). Twitter sendiri merupakan jaringan informasi constant yang menghubungkan penggunanya dengan cerita, ide, pendapat dan berita terbaru apa saja yang dianggap menarik oleh banyak orang. Twitter memiliki fitur yang bernama *tweets* yang memungkinkan penggunanya untuk berbagi pendapat dan pikirannya tersebut ke platform dan pengguna Twitter. Tak hanya itu, banyak dijumpai pengguna Twitter yang mengatakan bahwa Twitter sebagai media sosial yang bebas dan tepat untuk menyampaikan pemikiran dan pendapat mereka akan suatu hal tak terkecuali ujar kebencian juga.

Terdapat salah satu metode yang dapat untuk mendeteksi ujar kebencian yaitu menggunakan sentiment analisis dengan teknik *deep learning* dan algoritma LSTM. Sentiment analysis sendiri merupakan suatu teknik natural language processing yang digunakan untuk menentukan status suatu data, entah itu mengandung muatan positif, netral, atau justru negative (algorit.ma, 2022). Sentiment analysis memungkinkan kita untuk memprediksi sebuah tweet yang mengandung ujar kebencian berdasarkan suatu kata yang terkandung dalam satu kalimat. Deep learning sendiri merupakan Teknik machine learning yang mengajarkan komputer untuk melakukan apa yang terjadi secara alami pada manusia, yaitu dengan belajar dengan memberi contoh. Deep learning bekerja dengan cara mempelajari kebiasaan/pola suatu data tanpa melakukan ekstraksi fitur terhadap data tersebut. Deep learning sendiri biasanya digunakan untuk klasifikasi suatu data text, gambar, suara dan lain-lain. Deep learning dilatih dengan sekumpulan data yang cukup besar dan berlabel dengan arsitektur jaringan syaraf yang berisi banyak lapisan. Bidirectional LSTM atau bidirectional long shortterm memory merupakan metode dalam deep learning yang merupakan salah satu modifikasi dari RNN (Recurent Neural Network) yang mampu mengingat kumpulan informasi yang telah disimpan dalam jangka Panjang sekaligus menghapus informasi yang sudah tidak relevan (algorit.ma, 2022). LSTM lebih efisien dalam memproses, memprediksi, sekaligus mengklasifikasikan data berdasarkan urutan waktu tertentu. Bidirectional LSTM bekerja secara dua arah, yaitu dari masa lampau ke masa depan dan dari masa depan ke masa lampau dalam menyimpan suatu informasi. Dengan metode Bidirectional LSTM ini, akan memprediksi sebuah tweet yang mengandung ujar kebencian dengan lebih akurat dan lebih optimal, khususnya dalam data text.

## Rumusan Masalah

Berangkat dari hal itu, maka diperlukan pendekatan untuk mengenali ujaran kebencian di Twitter menggunakan sentiment analysis dengan algoritma deep learning dan algoritma Bidirectional LSTM.

## Tujuan

Dari rumusan masalah terebut, ditentukan tujuan penelitian sebagai berikut:

1. Melakukan sentiment analisis ujaran kebencian tweets pengguna Twitter di Indonesia menggunakan algoritma LSTM dan Bidirectional LSTM.
2. Menghitung akurasi optimal dari algoritma LSTM dan Bidirectional LSTM.
3. Mengetahui parameter deep learning yang mampu memberikan hasil paling optimal.

## Manfaat Penelitian

Dapat membuat sebuah model deep learning menggunakan algoritma LSTM dan Bidirectional LSTM yang dapat mendeteksi/mengenali ujaran kebencian di Twitter.

## Batasan Masalah

Batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Dataset yang digunakan merupakan data tweets dari pengguna Twitter di Indonesia
2. Dataset yang digunakan berbahasa Indonesia
3. Dataset merupakan tweets yang membahas tentang kejadian fenomenal hacker Bjorka

## Sistematika Penulisan

1. BAB I PENDAHULUAN

Diuraikan latar belakang masalah, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah dan sistematika penulisan penelitian ini.

1. BAB II LANDASAN TEORI

Teori-teori yang relevan dan mendukung penelitian ini yang berhubungan dengan *deep learning* khususnya metode Bidirectional LSTM*,* sentiment analysis.

1. BAB III METODOLOGI PENELITIAN

Berisi bagaimana pengolahan data untuk penelitian dan rencana/Langkah-langkah yang akan ditempuh dalam melakukan penelitian ini. Selain itu, akan dijelaskan proses implementasi dari metode *deep learning* metode Bidirectional LSTM untuk dapat melakukan sentiment analysis serta memaparkan arsitektur model yang akan digunakan untuk penelitian ini.

1. BAB IV IMPLEMENTASI DAN ANALISA HASIL

Implementasi serta hasil dari penelitian ini. Setelah itu, akan melakukan pembahasan dan Analisa terhadap hasil pengujian-pengujian yang telah dilakukan.

1. BAB V PENUTUP

Menyimpulkan percobaan-percobaan dalam penelitian yang telah dilakukan dan juga akan diuraikan saran dari penulis untuk pengembangan dari penelitian ini.

1. BAB VI

Berisi tentang referensi yang digunakan dalam penelitian ini.

# BAB II TINJAUAN PUSTAKA DAN LANDASAN TEORI

Pada Bab II ini, akan diuraikan tinjauan Pustaka dari beberapa referensi yang terkait dengan penelitian ini serta teori-teori yang mendukung dan relevan dengan penelitian ini, seperti sentiment analysis, jaringan syaraf tiruan, deep learning, LSTM, Bidirectional LSTM.



## Tinjauan Pustaka

Mengklasifikasi ujaran kebencian di Twitter merupakan hal yang penting agar tidak mudah tergiring oleh opini kebencian oleh oknum-oknum yang menyebarkan ujaran tersebut demi sebuah kepentingan. Oleh karena itu, dibutuhkan sebuah keahlian khusus untuk melakukan sentiment analysis/klasifikasi ujaran kebencian. Beberapa penelitian tentang sentiment analysis telah dilakukan terkait sentiment analysis twitter. Penelitian oleh (Ahmad, Ali, & Ahtab, 2017) yang membahas tentang sentiment analysis tweets menggunakan support vector machine (SVM) mendapatkan akurasi 85%. Selanjutnya terdapat juga penelitian oleh (Fauzi, 2018) yang membahas tentang penggunaan random forest untuk sentiment analysis Bahasa Indonesia mendapatkan skor OOB/akurasi sebesar 82,9%. Kemudian, terdapat penelitian oleh (Fitri, Andreswari, & Hasibuan, 2019) yang membahas tentang sentiment analysis twitter menggunakan algoritma Naïve Bayes, Decision Tree, dan Random Forest yang memberikan masing-masing akurasi 83.43%, 82.91%, dan 82.91%. Terdapat juga penelitian oleh (Jianqiang, Xiaolin, & Xuejun, 2018) membahas tentang sentiment analysis menggunakan algoritma Deep Convolutional Neural Network yang mendapatkan akurasi terbaik sebesar 87.62%. Penelitian yang dilakukan oleh (Huq, Ali, & Rahman, 2017) membahas tentang sentiment analysis menggunakan KNN dan support vector machine dengan menggunakan 5-fold cross validation mendapatkan akurasi terbaik 84,32% dan 77,97%. Terdapat penelitian yang dilakukan oleh (Liao, Wang, Yu, Sato, & Cheng, 2016) yang membahas tentang sentiment analysis untuk klasifikasi latar situasi pada data Twitter yang membuahkan akurasi pengembangan sebesar 74.5%. Selanjutnya, (Abdelgwad, Soliman, Taloba, & Farghaly, 2021) melakukan penelitian tentang sentiment analysis berbasis aspek Bahasa Arab menggunakan model Bidirectional GRU yang mendapatkan akurasi optimal sebesar 83.98%

Untuk mempermudah dalam melihat dan membandingkan penelitian yang sudah diuraikan di atas, dapat melihat ke table 2.1. Dari referensi di atas, dapat dilihat bahwa sentiment analysis untuk Twitter menggunakan berbagai macam algoritma mendapatkan akurasi paling minimum adalah 82,91% dan hal ini bisa dikatakan sudah cukup baik.

|  |  |  |  |
| --- | --- | --- | --- |
| Pengarang dan Tahun | Judul | Metode | Keterangan |
| Mohammed M.Abdelgwad, Taysir Hassan A Soliman, Ahmed I.Taloba,  Mohamed Fawzy Farghaly  (2021) | Arabic Aspect Based Sentiment Analysis Using Bidirectional GRU Based Models | Bidirectional GRU | Mendapatkan akurasi optimal sebesar 83,98% |
| Veny Amilia Fitri, Rachmadita Andreswari, Muhammad Azani Hasibuan (2019) | Sentiment Analysis of Social Media Twitter with Case of Anti-LGBT Campaign in Indonesia using Naïve Bayes, Decision Tree, and Random Forest Algorithm | Naïve Bayes | Mendapatkan akurasi sebesar 83,43% |
| Decision Tree | Mendapatkan akurasi sebesar 82,91% |
| Random Forest | Mendapatkan akurasi sebesar 82,91% |
| Muhammad Ali Fauzi (2018) | Random Forest Approach for Sentiment Analysis in Indonesia Language | Random Forest | Melakukan nilai OOB/akurasi sebesar 82,9% |
| Zhao Jianqiang, Gui Xiaolin, Zhang Xuejun (2018) | Deep Convolution Neural Networks for Twitter Sentiment Analysis | Deep Learning CNN (Convolutional Neural Network) | Mendapatkan akurasi sebesar 87,62% |
| Mohammad Rezwanul Huq, Ahmad Ali, Anika Rahman (2017) | Sentiment Analysis on Twitter Data using KNN and SVM | KNN Classifier | Mendapatkan akurasi terbaik 84,32% dengan 5-fold cross validation |
| Support Vector Machine (SVM) | Mendapatkan akurasi terbaik 77,97% dengan 5-fold cross validation |
| Munir Ahmad, Shabib Aftab, Iftikhar Ali (2017) | Sentiment Analysis of Tweets using SVM | Support Vector Machine (SVM) | Mendapatkan akurasi sebesar 85% |
| Shiyang Liaoa, Junbo Wang, Ruiyun Yua  ,Koichi Satob  ,Zixue Cheng | CNN for situations understanding based on  sentiment analysis of twitter data | CNN | Mendapatkan akurasi tahap pengembangan sebesar 74,5% |



Tabel 2.1 Tinjauan Pustaka Penelitian.

Menurut (Singhal & Bhattacharyya, 2016) deep learning mempunyai potensi untuk mengatasi kekurangan pada machine learning, khususnya support vector machine dalam melakukan sentiment analysis dengan memberikan beberapa potensi keuntungan diantaranya adalah kemampuan untuk beradaptasi dengan variasi tugas dengan perubahan yang sangat kecil dalam sistem itu sendiri , memungkinkan pembelajaran representasi yang baik dan tidak memerlukan *feature extraction* tetapi menggunakan *words embedding* sebagai input yang menyimpan informasi.

Berangkat dari hal ini, saya ingin melakukan penelitian sentiment analysis ujaran kebencian Twitter di Indonesia menggunakan *deep learning* dengan Teknik LSTM dan Bidirectional LSTM.

## Landasan Teori

### Sentiment Analysis

Sentiment analysis adalah studi berbasis komputer tentang pendapat, keyakinan, dan emosi orang tentang entitas tertentu. Entitas memiliki kemampuan untuk mempengaruhi orang, gerakan, atau objek (Medhat, Hassan, & Korashy, 2014) . Analisis sentimen dengan kata lain digunakan untuk mengetahui keadaan suatu data, baik yang mengandung sentimen positif, netral, maupun negatif. Dimungkinkan juga untuk menggunakan sentiment analysis sebagai klasifikasi. Sentiment analysis bekerja menggunakan parameter berikut:

Diagram

Description automatically generated

Gambar 2.1 Alur Kerja Sentiment Analysis (Medhat, Hassan, & Korashy, 2014) Jaringan Syaraf Tiruan(Artificial Neural Network)

### Jaringan Syaraf Tiruan

Jaringan saraf tiruan terinspirasi oleh model awal pemrosesan sensorik oleh otak. Jaringan saraf tiruan dapat dibuat dengan mensimulasikan jaringan memodelkan neuron di komputer. Dengan menerapkan algoritma yang meniru proses nyata neuron, dapat membuat jaringan 'belajar' untuk memecahkan banyak jenis masalah (A)

Jaringan saraf tiruan (JST) merupakan jaringan yang terdiri dari unit pemrosesan kecil yang dimodelkan seperti sistem saraf manusia. JST adalah sistem adaptif yang dapat mengubah strukturnya untuk memecahkan masalah berdasarkan informasi eksternal dan internal yang mengalir melalui jaringan. Karena sifatnya yang adaptif, JST sering disebut sebagai jaringan adaptif. Sederhananya, JST adalah alat pemodelan data statistik non-linear. JST dapat digunakan untuk memodelkan hubungan yang kompleks antara input dan output untuk menemukan pola dalam data. Menurut teorema yang disebut "teorema estimasi universal", JST dengan setidaknya satu lapisan tersembunyi dengan fungsi aktivasi nonlinier dapat memodelkan setiap fungsi terukur boreal dari satu dimensi ke dimensi lain.

Model pada JST pada dasarnya merupakan fungsi model matematika yang mendefinisikan fungsi {\displaystyle f:X\rightarrow Y}dari X ke Y. Istilah "jaringan" pada JST merujuk pada interkoneksi dari beberapa *neuron* yang diletakkan pada lapisan yang berbeda. Secara umum, lapisan pada JST dibagi menjadi tiga bagian:

* Lapis masukan (*input layer*) terdiri dari *neuron* yang menerima data masukan dari variabel X. Semua *neuron* pada lapis ini dapat terhubung ke *neuron* pada lapisan tersembunyi atau langsung ke lapisan luaran jika jaringan tidak menggunakan lapisan tersembunyi.
* Lapisan tersembunyi (*hidden layer*) terdiri dari *neuron* yang menerima data dari lapisan masukan.
* Lapisan luaran (*output layer*) terdiri dari *neuron* yang menerima data dari lapisan tersembunyi atau langsung dari lapisan masukan yang nilai luarannya melambangkan hasil kalkulasi dari X menjadi nilai Y.

#### Dense Layer

Dense layer merupakan layer yang simple dimana setiap neuron menerima input dari layer sebelumnya ataupun bekerja sebagai input layer (Dumane). Layer ini akan melakukan operasi perkalian vector. Hasil dari setiap neuron dari lapisan sebelumnya menuju ke setiap neuron tunggal dari lapisan padat. Dapat dikatakan bahwa jika lapisan sebelumnya mengeluarkan matriks (M x N) dengan menggabungkan hasil dari setiap neuron, keluaran ini melewati lapisan padat di mana jumlah neuron pada lapisan padat harus N.

Diagram

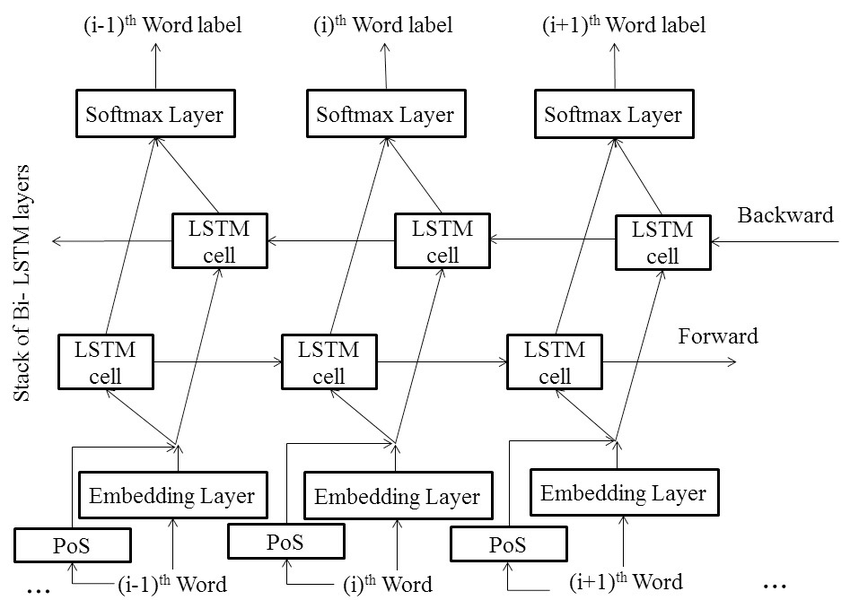
Description automatically generated

Gambar dense layer (Verma, 2021)

Gambar 2.2. Dense layer

#### Embedding Layer

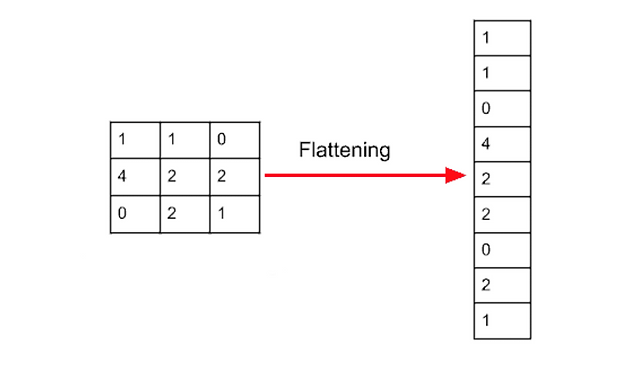
Embedding layer merupakan layer dalam deep learning yang merepresentasikan kosa kata dalam sebuah dokumen yang paling popular yang direpresentasikan dalam sebuah vector (Karani, 2018). Setelah itu, word embedding dapat menangkap konteks, kesamaan semantic dan sintaksis dalam suatu dokumen.



Gambar 2.3 Embedding layer di dalam model BiLSTM (ReaserchGate)

#### Flatten Layer

Flatten layer berfungsi untuk melakukan transformasi neurons dari multi dimensi menjadi dimensi tunggal. Dalam penelitian ini, flatten layer digunakan untuk melakukan transformasi output dari layer Bidirectional/Embedding layer yang tergolong multi dimensi (3 dimensi) menjadi dimensi tunggal agar bisa diterima oleh hidden layer dense.



Gambar 2.4. Flatten layer

### Deep Learning

Menurut (LeCun, Bengio, & Hinton, 2015), deep learning memungkinkan pemodelan komputasi yang memungkinkan beberapa lapisan/layer pemrosesan untuk mempelajari representasi data dengan berbagai tingkat abstraksi. Metode ini dapat menemukan struktur rumit dalam kumpulan data besar dengan menggunakan algoritma backpropagation untuk menunjukkan bagaimana mesin harus mengubah parameter internalnya yang digunakan untuk menghitung representasi di setiap lapisan dari representasi di lapisan sebelumnya.

Selain itu, menurut (MathWorks, 2022) deep learning adalah cara menggunakan komputer untuk melakukan tugas-tugas yang biasanya dilakukan oleh manusia atau dapat digambarkan seperti yang dilakukan dengan metode tertentu. Model komputer digunakan dalam deep learning untuk melakukan tes klasifikasi video, audio atau teks. Tujuan model pembelajaran adalah yang terbaru, meskipun mengurangi aktivitas manusia. Model ini didasarkan pada sejumlah besar data berlabel dan struktur jaringan multilayer. Istilah "deep" biasanya digunakan untuk sisa jaringan yang bersangkutan. Jaringan syaraf tradisional terdiri dari dua sampai tiga lapisan, sedangkan jaringan syaraf dengan 150 lapisan dapat ditemukan di deep learning. Di sisi lain, model deep learning menggunakan sejumlah besar data untuk menamainya dan juga menggunakan analisis jaringan saraf yang berasal dari data untuk membuat fungsi yang berbeda. Berikut adalah contoh deep learning dalam jaringan syaraf tiruan:Diagram

Description automatically generated

Gambar 2.5 contoh jaringan syaraf tiruan pada *deep learning*  (MathWorks, 2022)

Di dalam *deep learning* sendiri, terdapat beberapa komponen pendukung seperti optimizer, activation function, regularization dan loss function.

#### Optimizer

Optimizer adalah algoritma atau metode yang digunakan untuk mengubah atribut jaringan saraf, seperti bobot tingkat pembelajaran, untuk mengurangi loss (Musstafa, 2021). Algoritma atau strategi optimizer bertanggung jawab untuk mengurangi loss dan mencapai hasil yang paling akurat.

##### RMSProp

RMSprop menggunakan sinyal gradien yang menyesuaikan ukuran langkah untuk setiap bobot secara terpisah (Musstafa, 2021). Dalam algoritma ini, dua gradien pertama dibandingkan untuk tanda. Jika mereka memiliki tanda yang sama, pergi ke arah yang benar dan karena itu tingkatkan ukuran langkah dengan sebagian kecil. Di sisi lain, jika tandanya dibalik, maka harus mengurangi ukuran langkah dan kemudian membatasi ukuran langkah dan melakukan pembaruan bobot. Masalah dengan RMSProp adalah tidak bekerja dengan baik dengan kumpulan data besar atau selama pembaruan mini-lokal. Propagator RMS juga dapat dianggap sebagai peningkatan dari pengoptimal AdaGrad karena meminimalkan tingkat pembelajaran yang menurun secara monoton. Algoritma ini terutama berfokus pada mempercepat proses optimasi dengan mengurangi jumlah evaluasi fungsi untuk mencapai minimum lokal. Algoritme mengkuadratkan rata-rata pergerakan gradien untuk setiap bobot dan membagi gradien dengan akar kuadrat dari kuadrat rata-rata.

RMS Prop(Root Mean Square) Deep Learning Optimizer

Gambar 2.6. persamaan optimizer RMSProp (Musstafa, 2021)

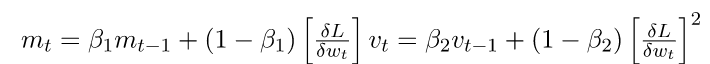
di mana gamma adalah faktor pelupa. Bobot diperbarui dengan rumus di bawah ini

RMS Prop(Root Mean Square) Deep Learning Optimizer

Gambar 2.7 persamaan optimizer RMSProp (Musstafa, 2021)

Sederhananya, jika ada parameter yang menyebabkan fungsi cost banyak berosilasi, itu akan menghukum memperbarui parameter itu. Algoritma ini memiliki beberapa keunggulan dibandingkan versi sebelumnya dari algoritma gradient descent. Algoritme konvergen dengan cepat dan membutuhkan lebih sedikit penyetelan daripada algoritme penurunan gradien dan variannya. Masalah dengan RMS Prop adalah kecepatan pembelajaran harus diatur secara manual dan nilai yang disarankan tidak berfungsi untuk setiap aplikasi.

##### Adam

Algoritma optimizer ini merupakan perpanjangan lebih lanjut dari penurunan gradien stokastik untuk memperbarui bobot jaringan selama pelatihan. Pengoptimal Adam memperbarui pembelajaran setiap bobot jaringan secara terpisah (Musstafa, 2021). Alih-alih menyesuaikan learning rate berdasarkan momen pertama (mean) seperti RMS Prop, Adamis juga menggunakan momen gradien kedua. Algoritme ini mudah diimplementasikan, memiliki runtime yang lebih cepat, kebutuhan memori yang lebih rendah, dan memerlukan konfigurasi yang lebih sedikit daripada algoritme pengoptimalan lainnya.

Gambar 2.8 persamaan optimizer adam (Musstafa, 2021)

Rumus di atas merupakan cara kerja pengoptimal adam. Di sini B1 dan B2 mewakili laju peluruhan rata-rata gradien.

Kelemahan dari optimizer ini biasanya berfokus pada waktu komputasi yang lebih cepat, sedangkan algoritma seperti penurunan gradien stokastik fokus pada titik data. Oleh karena itu, algoritme seperti SGD menggeneralisasi data dengan lebih baik dengan mengorbankan kecepatan komputer yang rendah. Oleh karena itu, algoritma optimasi dapat dipilih sesuai dengan kebutuhan dan tipe data.

#### Activation Function

(Marimuthu, 2022) mengatakan bahwa activation function bertanggung jawab atas apakah neuron dalam jaringan saraf deep learning harus diaktifkan atau tidak. Artinya, ia menggunakan beberapa operasi matematika sederhana untuk menentukan apakah neuron input jaringan relevan atau tidak relevan dengan proses prediksi. Tujuan dari fungsi aktivasi adalah kemampuan untuk memasukkan nonlinier ke dalam jaringan saraf dan menghasilkan output dari sekumpulan nilai input yang dimasukkan ke dalam lapisan.

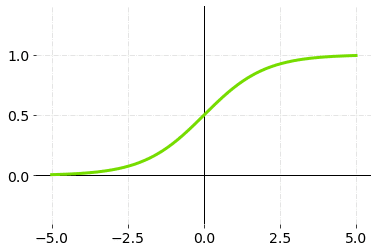
##### Sigmoid

Activation function sigmoid menerima nilai input dan mengembalikan 0 atau 1 (Marimuthu, 2022). Sigmoid sering digunakan untuk pemrosesan klasifikasi biner. Sigmoid memberikan probabilitas bahwa kelas tertentu ada. Ketika sigmoid diwakili dalam matematika, persamaan itu adalah:

Diagram, schematic

Description automatically generated

Gambar 2.9 persamaan activation function sigmoid (Marimuthu, 2022)



Gambar 2.10 grafik activation function sigmoid (Marimuthu, 2022)

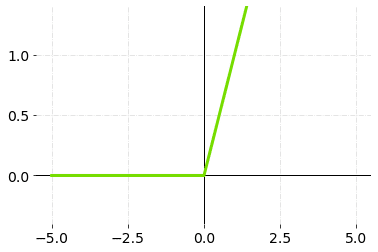
##### Relu

ReLU adalah singkatan dari Rectified Linear Unit dan merupakan salah satu activation function yang paling umum digunakan dalam aplikasi. Re-Lu memecahkan masalah gradien yang hilang karena fungsi re-lu memiliki gradien maksimum 1 (Marimuthu, 2022). Ini juga memecahkan masalah saturasi neuron karena kemiringan fungsi ReLU tidak pernah nol. ReLU berkisar dari 0 hingga tak terhingga. Ketika direpresentasikan dalam matematika, Re-Lu memiliki persamaan ini:

Text

Description automatically generated

Gambar 2.11 persamaan activation function ReLu (Marimuthu, 2022)



Gambar 2.12 grafik activation function ReLu (Marimuthu, 2022)

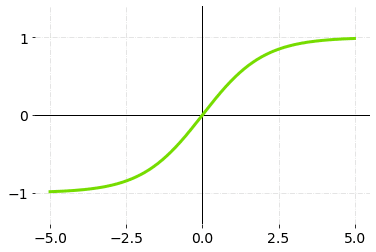
##### Softmax

Kombinasi dari banyak sigmoid disebut fungsi Softmax. Ini menentukan probabilitas relatif. Serupa dengan activation function sigmoid, fungsi Softmax mengembalikan probabilitas setiap kelas/label. Dalam klasifikasi multi-kelas, activation function softmax paling sering digunakan untuk lapisan terakhir dari jaringan saraf (Marimuthu, 2022). Fungsi softmax memberikan probabilitas kelas saat ini relatif terhadap yang lain. Ini berarti bahwa dia juga mempertimbangkan kemungkinan kelas lain. Ketika direpresentasikan dalam matematika, softmax adalah persamaan seperti ini:

Text

Description automatically generated

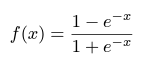
Gambar 2.13 persamaan activation function softmax (Marimuthu, 2022)



Gambar 2.13. grafik activation function softmax (Marimuthu, 2022)

##### TanH

Fungsi hiperbolik tangen atau tanh cocok digunakan untuk jaringan syaraf tiruan multi-layer (Datta, 2020). Fungsi tanh didefinisikan sebagai:



Gambar 2.14. persamaan activation function tanh (Datta, 2020)

Activation function tanh merupakan modifikasi dari activation function sigmoid karena persamaan tanh didefinisikan sebagai berikut

Gambar 2.15. persamaan activation function tanh

Tanh merupakan fungsi kontinu dimana outputnya berkisar dari negatif satu sampai dengan satu. Dari hal itu, outputnya bisa dikatakan negatif, nol, ataupun positif. Oleh karena itu, fungsi ini terpusat nol dan menyelesaikan masalah 'fungsi aktivasi bukan berpusat nol' dari fungsi sigmoid.

#### Regularization

Regularisasi adalah serangkaian teknik yang mengurangi kompleksitas model jaringan saraf selama pelatihan untuk menghindari overfitting. Salah satu metode regularisasi yang umum digunakan adalah Dropout. Istilah "dropout" mengacu pada putusnya node (input dan lapisan tersembunyi) dalam jaringan saraf. Semua koneksi maju dan mundur dengan node yang dijatuhkan dihapus sementara, sehingga menciptakan arsitektur jaringan baru dari jaringan induk (Yadav).

Dropout berarti selama proses training dengan beberapa probabilitas P neuron dari jaringan syaraf dimatikan.

A picture containing watch

Description automatically generated

Gambar 2.16. contoh implementasi dari Dropout

Contoh dapat diambil dari gambar di atas, di mana di sebelah kiri adalah jaringan saraf feedforward tanpa Dropout. Menggunakan penurunan dengan probabilitas P = 0,5 selama proses pelatihan, secara acak mematikan neuron dan membentuk jaringan saraf seperti yang ditunjukkan di sebelah kanan. Ini berarti bahwa setengah dari neuron mati dan tidak dihitung, yang membuat jaringan saraf lebih sederhana dan tidak terlalu kompleks, yang mengurangi terjadinya over-tuning. Penonaktifan neuron dengan probabilitas tertentu P diterapkan pada setiap langkah maju dan pembaruan bobot.

#### Loss Function

Loss function adalah fungsi yang membandingkan target dan nilai keluaran yang diprediksi; mengukur seberapa baik jaringan saraf memodelkan data pelatihan. Selama pelatihan, tujuannya adalah untuk meminimalkan loss antara kekuatan yang diprediksi dan target (Yathish, 2022). Ada dua jenis loss function dalam jaringan syaraf tiruan, yaitu fungsi regresi dan fungsi kerugian klasifikasi (Yathish, 2022)

##### Regression Loss Function

Saat digunakan dalam jaringan saraf regresi, model memprediksi nilai keluaran yang benar dengan nilai masukan (bukan judul yang dipilih sebelumnya, misalnya, mean squared error (MSE) dan mean absolute error (MAE) (Yathish, 2022).

1. Mean Squared Error (MSE)

Text

Description automatically generated

Gambar 2.17 Persamaan MSE (Yathish, 2022)

Fungsi ini memiliki banyak properti yang membuatnya sangat cocok untuk menghitung loss. Selisihnya kuadrat, artinya tidak masalah apakah nilai prediksi lebih besar atau lebih kecil dari nilai target; Namun, skor dengan kesalahan besar dihukum. MSE juga merupakan fungsi cembung dengan minimum global yang terdefinisi dengan baik. Ini memudahkan penggunaan pengoptimal penurunan gradien untuk menentukan nilai bobot. Namun, salah satu kelemahan dari fungsi kerugian ini adalah sangat sensitif terhadap penyimpangan, artinya jika nilai prediksi secara signifikan lebih tinggi atau lebih rendah dari nilai target, maka akan meningkatkan kerugian secara signifikan.

1. Mean Absolute Error (MAE)

Text

Description automatically generated with medium confidence

Gambar 2.18 Persamaan MAE (Yathish, 2022)

Loss function ini digunakan dalam beberapa kasus sebagai alternatif untuk MSE. MAE digunakan dalam kasus di mana ada banyak outlier dalam data pelatihan. Fungsi kerugian ini juga memiliki beberapa kelemahan; karena jarak rata-rata mendekati nol, optimasi penurunan gradien tidak akan bekerja karena turunan dari fungsi pada 0 tidak terdefinisi (yang menyebabkan kesalahan karena tidak mungkin untuk membagi dengan 0).

##### Classification Loss Function

Digunakan untuk klasifikasi dalam jaringan saraf, di mana jaringan saraf menghasilkan vektor probabilitas input yang dimiliki oleh beberapa kelas yang telah ditentukan, dan kemudian dapat memilih kelas dengan probabilitas kepemilikan tertinggi (Yathish, 2022). Sebagai contoh, entropi silang biner dan entropi silang kategoris dari fungsi kerugian.

1. Binary Crossentropy

Jaringan saraf klasifikasi menghasilkan vektor probabilitas dari kemungkinan bahwa input yang diberikan cocok dengan setiap kelas yang telah ditentukan, dan kemudian memilih kelas yang paling mungkin sebagai hasilnya. Dalam klasifikasi biner, hanya ada dua nilai yang mungkin, nilai aktual y adalah 0 atau 1. Untuk menentukan secara akurat kerugian antara nilai aktual dan prediksi, perlu membandingkan nilai aktual (0 atau 1) dengan probabilitas bahwa input termasuk dalam kategori itu (p(i ) ) = probabilitas, kelas itu adalah 1; 1 — p(i) = probabilitas kelas itu

Text

Description automatically generated with low confidence

Gambar 2.19 Persamaan loss function binary crossentropy

1. Categorical crossentropy

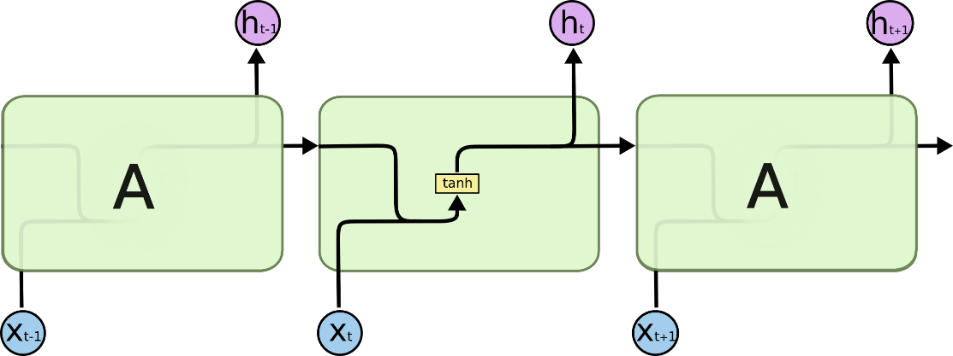
Dalam kasus di mana jumlah kategori lebih dari dua, menggunakan categorical crossentropy. Categorical crossentropy mengikuti proses yang sangat mirip dengan binary crossentropy, satu-satunya perbedaan adalah jumlah kategori yang tersedia.

Diagram, text

Description automatically generated

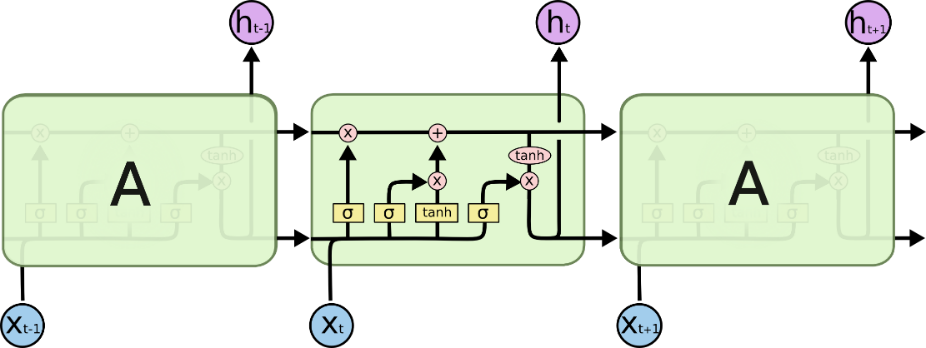
Gambar 2.20 Persamaan loss function categorical crossentropy

### Long Short-Term Memory (LSTM)

Long Short-Term Memory atau biasa dikenal dengan LSTM merupakan sebuah RNN (jaringan syaraf berulang) yang dapat mengatasi permasalahan ketergantungan jangka Panjang (Yu, Si, Hu, & Zhang, 2019). LSTM meningkatkan kapasitas mengingat sel rekuren standar (RNN) dengan mengenalkan “gate” atau gerbang ke dalam cell state. LSTM dapat mengingat informasi untuk waktu yang lama (Colah, 2015). Semua jenis RNN mengambil bentuk rantai modul jaringan saraf berulang.

Gambar 2.21 Contoh RNN (Colah, 2015)

LSTM juga memiliki struktur seperti rantai ini, tetapi modul yang diulang memiliki struktur yang berbeda. Alih-alih satu lapisan jaringan saraf, ada empat yang berkomunikasi dengan cara yang sangat spesifik.



Gambar 2.22 Struktur LSTM (Colah, 2015)

Kunci LSTM adalah cell state, garis horizontal yang melintasi bagian atas grafik. Keadaan sel itu bergerak langsung di sepanjang rantai dengan hanya beberapa interaksi linier kecil. Sangat mudah untuk informasi mengalir melalui tidak berubah. LSTM dapat menghapus atau menambahkan data ke keadaan sel yang dikendalikan oleh struktur yang disebut Gate. Gate adalah cara opsional untuk mentransfer informasi. Mereka terdiri dari lapisan jaringan saraf sigmoid dan operasi perkalian titik. Alur kerja dari LSTM sendiri akan berjalan seperti berikut:

Diagram

Description automatically generated

Gambar 2.23 Cell state melalui forget gate (Colah, 2015)

Langkah pertama adalah menentukan informasi apa yang akan dibuang dari cell state yang dilakukan oleh forget gate.

Diagram

Description automatically generated

Gambar 2.24 Output gerbang sigmoid (Colah, 2015)

Menentukan informasi baru yang akan disimpan dalam cell state.

Diagram

Description automatically generated with medium confidence

Gambar 2.25 Mengupdate cell state lama (Colah, 2015)

Melakukan pembaharuan informasi terhadap cell state lama dengan cell state yang baru, atau bisa dikatakan melakukan kolaborasi informasi antara informasi di cell state lama dengan informasi yang ada di cell state yang terbaru

A picture containing text, clock

Description automatically generated

Gambar 2.26 Menentukan output terbaru dari cell state. (Colah, 2015)

### Bidirectional Long Sshort-Term Memory (LSTM)

Bidirectional LSTM atau LSTM dua arah adalah perpanjangan dari LSTM tradisional yang dapat meningkatkan kinerja model untuk masalah klasifikasi urutan. Untuk masalah di mana semua langkah waktu dari urutan input tersedia, LSTM dua arah melatih 2 urutan input. Yang pertama dalam urutan input apa adanya dan yang kedua dalam salinan terbalik dari urutan input. Ini dapat memberikan konteks tambahan ke jaringan dan mengarah pada pembelajaran masalah yang lebih cepat dan lebih lengkap (Mungalpara, 2021).

Diagram

Description automatically generated

Gambar 2.27 Bidirectional LSTM (Mungalpara, 2021)

### Preprocessing

Preprocessing merupakan sebuah proses untuk membuat input untuk analisis yang diberikan kurang kompleks dengan cara yang tidak mempengaruhi interpretabilitas atau kesimpulan substantif dari model selanjutnya (Denny & Spirling, 2017). Preprocessing pada dasarnya bertujuan untuk melakukan transformasi data tanpa mengubah isi dari data tersebut agar data dapat lebih mudah digunakan. Preprocessing yang biasanya ada dalam melakukan klasifikasi teks ialah lowercasing, stemming, stopword removal, pembersihan karakter khusus, pembersihan karakter non-alphanumeric dan masih banyak lagi.

#### Lowercasing

Lowercasing merupakan salah satu tahap preprocessing yang sering diaplikasikan jika menghadapi data teks. Hal ini bertujuan untuk mengatasi masalah suatu karakter yang sebenernya sama, akan tetapi dapat terdeteksi beda oleh computer karena perbedaan huruf kapital dan kecil (misal Gajah dengan gajah) (Denny & Spirling, 2017).

#### Stemming

Stemming merupakan tahapan dalam teks preprocessing yang akan mereduksi sebuah kata menjadi bentuk dasar dari kata tersebut (Denny & Spirling, 2017). Stemming sendiri sering diartikan sebagai Teknik pengurangan kosakata. Stemming dapat dimisalkan dengan reduksi kata “menyapu” menjadi “sapu”.

#### Stopword Removal

Terdapat beberapa kata yang memiliki makna yang berarti, dan terdapat beberapa kata yang tidak memiliki kata berarti. Kata-kata yang tidak memberikan informasi/makna berarti sering disebut dengan “stop word”. Stopword removal berfungsi untuk menghilangkan karakter yang kurang memiliki makna (Denny & Spirling, 2017). Kata-kata yang kurang memiliki makna biasanya merupakan kata konjungsi, kata-kata fungsi seperti “itu”, “dan”, dan “dia” sebagai contoh.

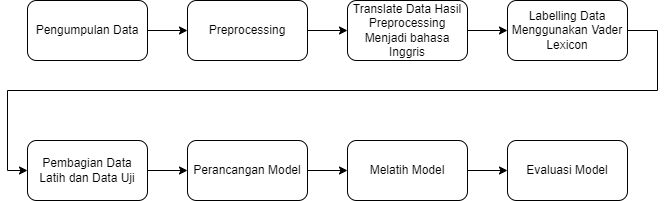
# BAB III METODE PENELITIAN

Bagian ini menguraikan mengenai rencana langkah-langkah yang akan dilakukan dalam penelitian. Termasuk bagaimana cara mendapatkan data, cara mengolah data, cara membuat alat uji jika diperlukan, cara analisis data, cara pengujian, dan di bagian akhir diberikan desain alat uji serta user interface dari alat uji yang akan dibangun.



## Tahapan Penelitian

Pada BAB ini, akan dijelaskan tahapan metode penelitian yang terdiri dari pengumpulan data, pengolahan data, pembuatan model deep learning, evaluasi model hingga skenario pengujian. Skema penelitian bisa ditinjau dalam gambar 3.1.



Gambar 3.1 Tahapan Penelitian

## Pengumpulan Data

Data yang digunakan dalam penelitian ini adalah hasil dari *web scrapping* di situs Twitter dengan mencari 20.000 total tweet yang mengandung kata “bjorka”. Peneliti melakukan *web scrapping* untuk mengambil tweet yang tidak mengandung link serta mention/reply dari pengguna lain sehingga tweet tersebut murni sebuah tweet yang berisi ide/gagasan/pendapat yang ingin dituliskan oleh pengguna. Setelah itu, data disatukan menjadi sebuah dataframe/table yang berisi username pengunggah tweet, waktu dibuatnya tweet tersebut serta isi tweet/teks tweet tersebut. Setelah itu, dataframe disimpan dalam format excel (.xlsx) agar bisa dilihat dengan mudah dan dapat digunakan kembali ketika akan digunakan dalam penelitian. Berikut beberapa sampel dataset hasil web scrapping dapat dilihat pada table 3.1.

|  |  |  |
| --- | --- | --- |
| created at | username | text |
| 2022-11-02 | rudiartraa | yaah , udah selesai nih Bjorka Bjorkaan nya ? |
| 2022-11-02 | oi\_filix315 | Kadang gua mikir sekilas pp gua mirip bjorka, tapi kan gua udh pake pp ini lama :((( |
| 2022-11-01 | falahsanjaya\_ | Knp ga ada yg jadi Bjorka Halloween party kemaren? :(( |
| 2022-11-01 | namanyaajess | Beritanya taik, mau ferdi sambo, mau bjorka, mau apalah yang kelaperan ga mungkin jadi tranding topik, yang tertindas sulit untuk berdiri, yang melawan akan tetap dibungkam. |
| 2022-11-01 | namakubaim | Apa kabar Bjorka? |

Tabel 3.1 Sample dataset

Selain itu, peneliti mengumpulkan dataset yang berisi kata-kata alay yang nantinya digunakan untuk melakukan transformasi kata-kata alay yang mungkin saja di dapatkan di dalam tweet menjadi bentuk normalnya. Dataset alay ini didapatkan dari Kaggle.com dalam format csv (comma separated value). Berikut beberapa sample dari kata-kata alay yang dapat dilihat pada table 3.2.

|  |  |
| --- | --- |
| Kata Alay | Pembenaran |
| aje | aja |
| ajg | anjing |
| diem | diam |
| ente | kamu |
| gr | gede rasa |
| ijo | hijau |
| knp | kenapa |

Tabel 3.2 Sample kata-kata alay dan pembenarannya

Kata-kata alay ini nantinya akan berguna Ketika memasuki tahap preprocessing data untuk mentransformasi kata alay menjadi bentuk dasar/bentuk benar.

Untuk memudahkan dalam memahami data, maka dapat melihat pada tabel 3.3 yang berisi tentang metadata dataset dan juga kamus alay.

|  |  |
| --- | --- |
| **Variable** | **Keterangan** |
| created at | Tanggal tweet diunggah |
| username | Nama pengguna akun Twitter |
| text | Tweet/teks yang diunggah oleh pengguna Twitter |

Tabel 3.3 Metadata dataset

|  |  |
| --- | --- |
| **Variable** | **Keterangan** |
| Kamus Alay | Kata dalam bentuk alay |
| Pembenaran | Kata setelah diperbaiki/dikembalikan ke bentuk semulal |

Tabel 3.4 Metadata Kamus Alay

## Preprocessing

Tahapan ini bertujuan untuk membuat kualitas data yang akan digunakan dalam training/analisis menjadi baik. Dalam kasus sentiment analysis ujaran kebencian ini, untuk membuat kualitas dataset teks menjadi baik, maka perlu dilakukan beberapa tahapan preprocessing diantaranya adalah lowercasing, pembersihan karakter selain alphanumeric, pembersihan karakter khusus/tidak biasa, normalisasi kata-kata alay, stemming, membersihkan stopword, tokenizing serta tranformasi data.

### Lowercasing

Lowercasing merupakan tahapan transformasi huruf menjadi huruf kecil. Input dari tahapan ini adalah sebuah kalimat yang terdiri dari huruf besar dan kecil yang akan diproses menjadi sebuah output kalimat yang terdiri dari huruf kecil saja. Berikut input dan output setelah dilakukan lowercasing dapat dilihat pada table 3.5.

|  |  |
| --- | --- |
| **Sebelum** | **Sesudah** |
| yaah , udah selesai nih Bjorka Bjorkaan nya ? | yaah , udah selesai nih bjorka bjorkaan nya ? |
| Apa kabar Bjorka? | apa kabar bjorka? |
| Kadang gua mikir sekilas pp gua mirip bjorka, tapi kan gua udh pake pp ini lama :((( | kadang gua mikir sekilas pp gua mirip bjorka, tapi kan gua udh pake pp ini lama :((( |

Tabel 3.5 Lowercasing

### Pembersihan Karakter Non-Alphanumeric

Alphanumeric merupakan karakter yang teridiri dari huruf dan angka. Pembersihan Non-Alphanumeric bertujuan untuk membersihkan karakter selain huruf dan angka, seperti tanda baca dan karakter khusus lainnya. Tahapan ini dilakukan agar data yang akan digunakan dalam penelitian dapat bekerja secara maksimal karena dalam membuat model deep learning, nantinya setiap kata akan dijadikan sebuah token yang di transformasi menjadi sebuah angka. Berikut input dan output untuk proses pembersihan Non-Alphanumeric karakter yang dapat dilihat pada table 3.6.

|  |  |
| --- | --- |
| **Sebelum** | **Sesudah** |
| yaah , udah selesai nih bjorka bjorkaan nya ? | yaah udah selesai nih bjorka bjorkaan nya |
| apa kabar bjorka? | apa kabar bjorka |
| kadang gua mikir sekilas pp gua mirip bjorka, tapi kan gua udh pake pp ini lama :((( | kadang gua mikir sekilas pp gua mirip bjorka tapi kan gua udh pake pp ini lama |

Tabel 3.6 Pembersihan Non-Alphanumeric

### Pembersihan Karakter Khusus

Pembersihan karakter khusus ini bertujuan untuk menghapus beberapa karakter khusus yang ada di dalam kalimat/teks, seperti kalimat yang mengandung kata “\n”, url, serta emoji. Hal ini akan membuat dataset yang nantinya akan digunakan kualitasnya menjadi baik karena kata-kata khusus tersebut yang berasal dari media sosial twitter cenderung tercantum di dalam kalimat/dataset tersebut sehingga akan membuat model machine learning kurang baik dalam melakukan klasifikasi. Berikut contoh input dan output untuk pembersihan karakter khusus yang dapat dilihat pada table 3.7.

|  |  |
| --- | --- |
| **Sebelum** | **Sesudah** |
| rame rame pada hapus twit yg menuduh msf adalah sosok bjorka 🤣🤣 | rame rame pada hapus twit yg menuduh msf adalah sosok bjorka |
| bjorka kembali tebar ancaman, kecam semua akun palsuhttp://inet.detik.com/security/d-6303238/bjorka-kembali-tebar-ancaman-kecam-semua-akun-palsu?bytedance=true | bjorka kembali tebar ancaman, kecam semua akun palsu |
| bjorka, aku padamu🫶🏻🖤 | bjorka, aku padamu |

Tabel 3.7 Pembersihan karakter khusus

### Normalisasi Kata-Kata “Alay”

Normalisasi kata-kata alay bertujuan untuk mengubah kata-kata yang dikategorikan menjadi kata-kata alay menjadi bentuk dasarnya. Hal ini bertujuan untuk mengurangi variansi kata yang ada di dalam dataset yang akan membuat kualitas dataset menjadi lebih baik dan optimal dalam klasifikasi nantinya. Tahapan ini akan menghilangkan kata alay yang sudah terdaftar pada dataset kata-kata alay yang sudah didapatkan melalui Kaggle. Berikut beberapa contoh normalisasi kata-kata alay menjadi bentuk dasarnya yang dapat dilihat pada table 3.8.

|  |  |
| --- | --- |
| **Sebelum** | **Sesudah** |
| anjirrr kok bisa adek gw percaya kalo bjorka itu gus syamsudin | anjirrr kok bisa adik gue percaya kalau bjorka itu gus syamsudin |
| knp ga ada yg jadi bjorka halloween party kemaren | kenapa tidak ada yang jadi bjorka halloween party kemarin |
| pengen gitu bikin sih bjorka biar gak bikin orang gak gagal fokus rakyat karena saat banyak kasus yg gagal fokus karena hacker bjorka bayaran pemerintah | ingin begitu bikin sih bjorka biar tidak bikin orang tidak gagal fokus rakyat karena saat banyak kasus yang gagal fokus karena hacker bjorka bayaran pemerintah |

Tabel 3.8 Normalisasi kata alay

### Stemming

Stemming merupakan Teknik menguraikan bentuk suatu kata menjadi bentuk asalnya/dasarnya. Hal ini dilakukan untuk mengubah kata berimbuhan menjadi bentuk dasarnya agar mempermudah model dalam melakukan klasifikasi, mengurangi variansi serta meningkatkan kualitas dataset menjadi lebih optimal. Sebagai contoh, untuk kata membenci akan ditransformasi menjadi benci, menyukai menjadi suka dan lebih banyak lagi. Berikut beberapa contoh stemming yang dapat dilihat pada table 3.9

|  |  |
| --- | --- |
| **Sebelum** | **Sesudah** |
| anjirrr kok bisa adik gue percaya kalau bjorka itu gus syamsudin | anjirrr kok bisa adik gue percaya kalo bjorka itu gus syamsudin |
| kenapa tidak ada yang jadi bjorka halloween party kemarin | kenapa tidak ada yang jadi bjorka halloween party kemarin |
| ingin begitu bikin sih bjorka biar tidak bikin orang tidak gagal fokus rakyat karena saat banyak kasus yang gagal fokus karena hacker bjorka bayaran pemerintah | ingin gitu bikin sih bjorka biar tidak bikin orang tidak gagal fokus rakyat karena saat banyak kasus yang gagal fokus karena hacker bjorka bayar pemerintah |

Tabel 3.9 Stemming word

### Stop Word Removal

Tahapan pembersihan stopword berfungsi untuk menghilangkan kata umum (common words) yang biasanya muncul dalam jumlah besar dan dianggap tidak memiliki makna. Tujuan dari menghilangkan stop words adalah mengurangi jumlah kata dalam sebuah dokumen yang akan berpengaruh terhadap kecepatan dan performa model. Stop word yang akan dijadikan sebagai acuan didapatkan dari libraries python Sastrawi (Sastrawi, 2017). Berikut beberapa contoh data sebelum dan sesudah dilakukannya tahapan pembersihan stopword yang dapat dilihat pada tabel 3.10

|  |  |
| --- | --- |
| **Sebelum** | **Sesudah** |
| anjirrr kok bisa adik gue percaya kalo bjorka itu gus syamsudin | anjirrr kok bisa adik gue percaya bjorka gus syamsudin |
| kenapa tidak ada yang jadi bjorka halloween party kemarin | bjorka halloween party kemarin |
| ingin gitu bikin sih bjorka biar tidak bikin orang tidak gagal fokus rakyat karena saat banyak kasus yang gagal fokus karena hacker bjorka bayar pemerintah | gitu bikin sih bjorka biar bikin gagal fokus rakyat gagal fokus hacker bjorka bayar pemerintah |

Tabel 3.10 Stopword Removal

## Penerjemahan Data

Tahapan ini bertujuan untuk menerjemahkan data tweet dari Bahasa Indonesia menjadi Bahasa Inggris. Hal ini perlu dilakukan karena dalam pelabelan menggunakan Vader, dibutuhkan dataset dengan Bahasa inggris sehingga dapat diketahui kutub/kecenderungan sebuah kalimat, apakah positif, negative, atau bahkan netral. Dataset diterjemahkan melalui program python menggunakan libraries translate. Penggunaan libraries ini memungkinkan untuk melakukan translate dengan memanggil API libraries tersebut dan melakukan translate di dalam program python, kemudian menambahkan hasil translate menjadi kolom baru. Berikut beberapa sample hasil penerjemahan dataset Bahasa Indonesia menjadi Bahasa Inggris yang dapat dilihat pada table 3.5.

|  |  |
| --- | --- |
| Indonesia | Inggris |
| yaah , udah selesai nih Bjorka Bjorkaan nya ? | yeah, have you finished Bjorka Bjorkaan? |
| Apa kabar Bjorka? | How are Bjorka? |
| Bjorka kemana ya | Where is Bjorka |
| Andai bjorka bisa tolong,,, | If bjorka can help ,,, |
| Pagi semua, kecuali bjorka | Morning all, except Bjorka |

Tabel 3.11 Sample dataset setelah diterjemahkan menjadi Bahasa Inggris menggunakan libraries translator dari python. Setelah diterjemahkan, maka akan melakukan pencarian kutub kecenderungan dataset, baik bermuatan positif, negative ataupun netral.

## Labelling Data

Pelabelan data berfungsi untuk memberikan label yang terdiri dari positif, negative maupun netral terhadap dataset. Hal ini perlu dilakukan karena model deep learning yang akan dibangun membutuhkan sebuah label untuk dapat berlatih dan mengklasifikasikan ujaran kebencian twitter. Untuk melakukan pelabelan terhadap dataset, menggunakan Vader Lexicon. Vader (Valence Aware Dictionary and Sentiment Reasoner) Lexicon merupakan sebuah tools/libraries yang berfungsi untuk melakukan sentiment analysis berbasis aturan yang secara khusus disesuaikan dengan sentiment yang diungkapkan di media sosial (MIT, 2014). Tools ini bersifat open-source dan dibawah lisensi MIT. Vader dapat menangani beberapa kasus khusus dalam sentiment analysis, seperti memahami negasi, memahami arti emoticon/emoji, memahami arti sebuah akronim, penggunaan tanda baca konvensional menandakan peningkatan intensitas sentiment, dan masih banyak lagi.

Dengan menggunakan Vader, data teks akan diproses dan dicari nilai sentimennya. Output yang dihasilkan oleh Vader merupakan sebuah probabilitas dari 3 kemungkinan yaitu positif, negative dan netral. Dari probabilitas tersebut, untuk melakukan labelling terhadap suatu teks di dalam dataset, maka diambil nilai compound dari ketiga probabilitas tersebut. Nilai compound adalah metric yang menghitung semua peringkat leksikon yang telah dinormalisasi antara -1 (paling negative) sampai dengan 1(paling positif). Untuk penelitian ini sentiment positif nilai compound nya dalam rentang lebih dari sama dengan 0,05, sentiment netral diantara -0,05 sampai dengan 0,05 serta untuk sentiment negative kurang dari sama dengan -0,05. Setelah menemukan nilai compound dan melakukan pelabelan data berdasarkan sentiment hasil nilai compound, maka dataset hasil pelabelan siap digunakan untuk diproses ditahap selanjutnya yaitu tahap preprocessing. Berikut contoh output dari Vader sentiment analysis.

|  |  |  |
| --- | --- | --- |
| **Teks** | **Vader Output** | **Label** |
| Bjorka is fun | {'neg': 0.0, 'neu': 0.694, 'pos': 0.306, 'compound': 0.296} | Positive |
| Bjorka please leak the secret document of suspicious plot funds | {'neg': 0.189, 'neu': 0.775, 'pos': 0.036, 'compound': -0.6728} | Negative |
| Talk to Bjorka Drinking Coffee Together | {'neg': 0.0, 'neu': 0.954, 'pos': 0.046, 'compound': 0.0387} | Neutral |

Tabel 3.12 Labeling Data menggunakan Vader Sentiment Analysis

Setelah mendapatkan label, maka label yang masih berupa text perlu dilakukan pengubahan menjadi kategori numeris. Karena terdapat tiga label, maka akan dikonversi menjadi 0, 1 dan 2. Hal ini bertujuan untuk dapat mengimplementasikan data kedalam model yang membutuhkan label numeris.

## Pemisahan Data

Pemisahan data bertujuan untuk membagi dataset menjadi dua bagian, yaitu bagian data latih dan data uji. Hal ini dilakukan untuk memisahkan data yang akan digunakan untuk melatih model dan data yang digunakan untuk uji data tunggal/testing model. Dataset akan dibagi menjadi data latih dan data uji dengan rasio 0,8:0,2 dimana data latih sebesar 16.000 dan data uji sebesar 4.000. Setelah dipisahkan, label pada data latih dan data uji akan dipisahkan untuk siap digunakan pada tahap melatih model.

Setelah data dipisahkan, maka label harus dilakukan transformasi menjadi bentuk one hot encoder. Label akan ditransformasi dari data tunggal menjadi sebuah array yang berisi 3 komponen angka. Hal ini perlu dilakukan untuk menyesuaikan penerapan deep learning menggunakan keras libraries. Berikut contoh penerapan one hot encoder yang dapat dilihat pada tabel 3.13.

|  |  |
| --- | --- |
| Label | Label One Hot Encoder |
| 0 | [1,0,0] |
| 1 | [0,1,0] |
| 2 | [0,0,1] |

Tabel 3.13. Label one hot encoder.

### Tokenizing

Tahapan tokenizing bertujuan untuk mengubah kata menjadi sebuah token, dan dalam penelitian ini kata akan diubah menjadi sebuah token angka. Tokenizing dapat dilakukan dengan cara berikut:

1. Mengimport library keras Tokenizer
2. Menginisialisasi tokenizer
3. Melakukan fit tokenizer dengan dataset. Hal ini bertujuan untuk mendeteksi semua kata yang ada di dalam dataset dan mengubah kata tersebut menjadi sebuah token angka.
4. Mengaplikasikan tokenizer ke setiap data yang ada di dalam data latih dan data uji dan menyimpannya sebagai sebuah sequences.
5. Memberikan padding ke setiap sequences sesuai dengan sequences terpanjang dengan menambahkan token “0” sampai ke element terakhir padding.

Untuk mempermudah melihat hasilnya, dapat dilihat pada tabel 3.14

|  |  |  |
| --- | --- | --- |
| Kalimat | Tokenizer | Padded |
| selesai nih bjorka bjorkaan | [131, 17, 2, 1] | [131, 17, 2, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| bjorka mending data kominfo data wifiku | [2, 106, 5, 60, 5, 1] | [ 2, 106, 5, 60, 5, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| sejarah bjorka spill data bunuh munir tau | [947, 2, 86, 5, 114, 83, 14] | [947, 2, 86, 5, 114, 83, 14, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| kabar bjorka | [89, 2] | [89, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |

Tabel 3.14 Tokenizing kalimat

## Perancangan Model

Model yang akan dirancang dalam penelitian ini akan terdiri dari beberapa hidden layer yang akan disusun menjadi sebuah sequences yang terdiri dari beberapa layers. Model yang akan dirancang akan mengimplementasikan model Sequentials dari libraries keras yang berarti mengelompokkan tumpukan lapisan linier. Dalam penelitian ini, akan lebih fokus pada susunan layer model deep learning dibandingkan parameter lain karena layer lebih berpengaruh dalam kasus sentiment analysis.

### Embedding Layer

Embedding layer merupakan input layer atau bisa dikatakan dengan layer pertama dalam model deep learning yang akan dirancang. Layer ini berfungsi untuk mengelompokan kata menjadi dua kutub yang berbeda. Parameter yang digunakan dalam layer ini adalah sebagai berikut:

Input\_length: Panjang input sequence

Output\_dim: dimensi output

Input\_dim: ukuran vocabulary

Hasil dari embbeding layer dapat dilihat pada tabel 3.15

|  |  |
| --- | --- |
| Sebelum | Sesudah |
| [131, 17, 2, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] | [[[-0.03162958 0.04269071 0.01509007 0.00124536]  [-0.04351017 0.000172 -0.03358928 0.0472861 ]  [-0.01396496 0.02855982 0.03887559 -0.01323872]  [-0.00201822 0.03641895 0.0171499 -0.02244518]  [-0.02064002 0.02616386 -0.03546425 -0.03617959]  …………………………….  [-0.02064002 0.02616386 -0.03546425 -0.03617959]]] |

Gambar 3.15 Hasil embedding layer

### Dense Layer

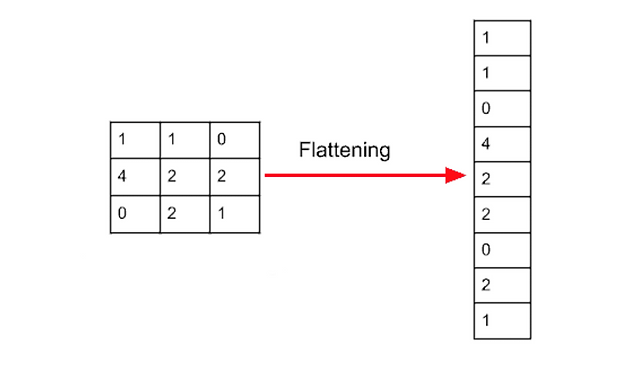
Dense layer akan memetakan input dari hidden layer sebelumnya menjadi sebuah output yang jumlahnya sesuai dengan jumlah unit di layer tersebut. Berikut contoh dari dense layer dari input hidden layer sebelumnya yang kemudian dipetakan sesuai jumlah output unit di layer tersebut.

|  |  |
| --- | --- |
| [[[-0.03162958 0.04269071 0.01509007 0.00124536]  [-0.04351017 0.000172 -0.03358928 0.0472861 ]  [-0.01396496 0.02855982 0.03887559 -0.01323872]  [-0.00201822 0.03641895 0.0171499 -0.02244518]  [-0.02064002 0.02616386 -0.03546425 -0.03617959]  …………………………….  [-0.02064002 0.02616386 -0.03546425 -0.03617959]]] | [[[-0.03966843]  [ 0.04057634]  [-0.01823187]  [ 0.05386413]  [-0.05570652]  [-0.05570652]  [-0.05570652]  ………………..  [-0.05570652]]] |

Tabel 3.16 Hasil dense layer dengan output unit=1

### Flatten Layer

Flatten layer berfungsi untuk melakukan transformasi neurons dari multi dimensi menjadi dimensi tunggal. Dalam penelitian ini, flatten layer digunakan untuk melakukan transformasi output dari layer Bidirectional/Embedding layer yang tergolong multi dimensi (3 dimensi) menjadi dimensi tunggal agar bisa diterima oleh hidden layer dense.



Gambar 3.2. Flatten layer

### LSTM Layer

Di dalam layer LSTM, terdapat beberapa Langkah perhitungan. Berikut ilustrasi pemodelan perhitungan LSTM pada suatu layer model deep learning:

Akan terdapat sebuah cell state dengan nilai , maka akan melakukan perhitungan dengan Langkah seperti di bawah ini.

* 1. Forget gate

Forget gate berfungsi untuk memilih informasi mana yang akan dilupakan/dihilangkan dari cell state.

Diagram

Description automatically generated

Gambar 3.3. Forget gate lstm (Colah, 2015)

* Menghitung nilai bobot forget gate dikalian dengan gabungan output dari waktu ke t-1 dan nilai input pada waktu ke t
* Menambahkan nilai hasil kalkulasi di atas dengan nilai bias forget gate
* Menghitung nilai dari forget gate

|  |  |
| --- | --- |
|  | Nilai forget gate |
|  | Nilai bobot untuk forget gate |
|  | Nilai output dari waktu ke t−1 |
|  | Nilai input pada waktu ke t |
|  | Bias pada forget gate |
|  | Fungsi sigmoid |

Tabel 3.17 Keterangan perhitungan pada formula forget gate

* 1. Input Gate

Input gate berfungsi untuk menyediakan informasi baru yang akan diteruskan ke dalam cell state.

Diagram

Description automatically generated

Gambar 3.4 Input gate lstm (Colah, 2015)

* Menghitung nilai dari input gate
* Menghitung nilai dari

|  |  |
| --- | --- |
|  | Nilai input gate |
|  | Nilai bobot untuk input gate |
|  | Nilai output dari waktu ke t−1 |
|  | Nilai input pada waktu ke t |
|  | Bias pada input gate |
|  | Nilai kandidat cell state |
|  | Fungsi sigmoid |
|  | Fungsi tanh |

Tabel 3.18 keterangan perhitungan pada formula input gate

* 1. Cell State

Menampung hasil operasi forget gate dan input gate yang nantinya akan digunakan untuk memperbarui nilai dari cell state.

Diagram

Description automatically generated with medium confidence

Gambar 3.5 Cell state lstm

|  |  |
| --- | --- |
|  | Nilai forget gate |
|  | Nilai input gate |
|  | Nilai memory cell state |
|  | Nilai memory pada cell state sebelumnya |
|  | Nilai kandidat cell state |

Tabel 3.19 Keterangan perhitungan pada formula cell state

* 1. Output gate

Menentukan informasi apa yang akan dijadikan sebuah output dari layer tersebut (melakukan filter terhadap cell state)

A picture containing text, clock

Description automatically generated

Gambar 3.6 Output gate lstm

* Menghitung nilai input gate
* Menghitung nilai output gate

|  |  |
| --- | --- |
|  | Nilai input gate |
|  | Nilai bobot untuk output gate |
|  | Nilai output dari waktu ke t−1 |
|  | Nilai input pada waktu ke t |
|  | Bias pada output gate |
|  | Nilai output gate |
|  | Fungsi sigmoid |
|  | Fungsi tanh |

Tabel 3.20 Keterangan perhitungan pada formula output gate

### Bidirectional LSTM Layer

## Evaluasi Model

Evaluasi model dilaukan dengan melakukan prediksi terhadap data uji, dan kemudian menyocokan kesamaan antara label asli dengan label hasil prediksi. Evaluasi akan ditinjau dari segi akurasi serta metric f1-score. Berikut rumus untuk f1-score yang dapat dilihat pada persamaan dibawah ini:

Keterangan :

TP = True Positive (data prediksi benar dan data actual benar)

TN = True Negative (data yang diprediksi benar dan data actual salah)

FP = False Positive (data prediksi salah dan data actual benar)

FN = False Negative (data prediksi salah dan data actual salah)

## Skenario Pengujian

Tahapan ini bertujuan untuk menjawab rumusan masalah yang ada pada Bab I yaitu sentiment analysis terhadap ujaran kebencian twitter. pada tahapan ini, data akan dilakukan preprocessing terlebih dahulu. Setelah itu, data akan dipisahkan antara data latih dan data uji. Setelah dipisahkan dengan rasio 0.8 untuk data latih dan 0.2 untuk data uji. Setelah itu, masing-masing label dari data uji dan data latih akan ditransformasi menggunakan one hot encoder agar menjadi sebuah array yang berisi 3 angka. Setelah melalui tahap pemisahan data, lalu akan mendefinisikan model deep learning.

Input model berupa array yang berisi nomor index kata yang telah diset pada tokenizer yang menampung 17586 indeks kata, dengan Panjang maksimum tiap token adalah 100 kata. Proses latih model akan berlangsung selama 20 epoch/iterasi dan dilakukan validasi dengan besaran ukuran data validasi adalah 0.2 dari data latih. Untuk pengukuran performa, akan melihat pada akurasi, validasi akurasi, loss, validasi loss, f1\_m dan validasi f1\_m.

Setelah itu, akan ada beberapa skenario pengujian yang akan diuraikan sebagai dengan fokus utama layer, optimizer dan loss function. Berikut rincian pembagian skenario pengujian:

### Layer Model

Layer model yang akan disusun terdiri dari beberapa kombinasi layer diantaranya embedding layer, lstm layer, bidirectional lstm layer, dense layer serta flatten layer.

### Activation Function

Activation function yang akan di dalam layer model terdiri dari beberapa susunan dan ketentuan. Untuk activation function tanh akan digunakan di dalam layer lstm dan bidirectional lstm. Untuk activation function relu akan digunakan di dalam layer dense. Untuk activation function softmax, akan digunakan pada output layer yaitu pada layer dense terakhir dalam model

### Optimizer

Optimizer yang akan digunakan dalam beberapa scenario pengujian diantaranya adalah rmsprop serta adam. Untuk learning rate optimizer diset sebesar 0,0003.

### Loss Function

Loss function yang akan digunakan dalam model ini terdiri dari dua, yaitu categorical crossentropy karena baik dalam mengatasi klasifikasi berkategori serta mean squared error karena nilai selisih error dikuadratkan sehingga cocok untuk memberi hukuman/terhadap suatu nilai agar pembelajaran berjalan lebih baik.

### Skenario Simpel Model 1

|  |  |  |  |
| --- | --- | --- | --- |
| Layer | Unit | Total Param | Activation Layer |
| Embedding | 100 | 1758600 | - |
| Flatten | 0 | 0 | - |
| Dense | 512 | 5120512 | Relu |
| Dense | 3 | 1539 | Softmax |

Tabel 3.21 Skenario simple model 1

1. Optimizer : Adam (learning rate=0.003)
2. Loss Functtion : Categorical Crossentropy
3. Epochs : 5

### Skenario Simpel Model 2

|  |  |  |  |
| --- | --- | --- | --- |
| Layer | Unit | Total Param | Activation Layer |
| Embedding | 100 | 1758600 | - |
| Flatten | 0 | 0 | - |
| Dense | 512 | 5120512 | Relu |
| Dense | 3 | 1539 | Softmax |

Tabel 3.22 Skenario simple model 2

1. Optimizer : RMSProp (learning rate=0.001)
2. Loss Functtion : Meaan Squared Error
3. Epochs : 5

### Skenario Kompleks Model 1

|  |  |  |  |
| --- | --- | --- | --- |
| Layer | Unit | Total Param | Activation Layer |
| Embedding | 100 | 1758600 | - |
| Bidirectional LSTM | 64 | 84480 | tanh |
| Bidirectional LSTM | 64 | 98816 | tanh |
| Flatten | 0 | 0 | - |
| Dense | 512 | 66048 | Relu |
| Dense | 3 | 1539 | Softmax |

Tabel 3.23 Skenario kompleks model 1

* + 1. Optimizer : Adam (learning rate=0.003)
    2. Loss Functtion : categorical crossentropy
    3. Epochs : 5

### Skenario Kompleks Model 2

|  |  |  |  |
| --- | --- | --- | --- |
| Layer | Unit | Total Param | Activation Layer |
| Embedding | 100 | 1758600 | - |
| Bidirectional LSTM | 64 | 84480 | tanh |
| Bidirectional LSTM | 64 | 98816 | tanh |
| Flatten | 0 | 0 | - |
| Dense | 512 | 66048 | Relu |
| Dense | 3 | 1539 | Softmax |

Tabel 3.24 Skenario kompleks model 2

* + 1. Optimizer : Adam (learning rate=0.003)
    2. Loss Functtion : categorical crossentropy
    3. Epochs : 5

### Pengujian Skenario

Setelah membuat beberapa skenario pengujian model deep learning, maka akan melakukan training untuk setiap skenario pengujian untuk mencari model yang paling optimal dengan menggunakan metrics evaluasi yaitu akurasi dan f1 score. Setelah melakukan training, maka akan menampilkan grafik akurasi, f1 score serta loss dari proses training terhadap tiap epochs yang berjalan. Setelah dilakukan training, akan beralih ke tahap evaluasi model dengan menggunakan data uji yang telah dipisahkan.

# BAB IV IMPLEMENTASI DAN HASIL PENELITIAN



## Implementasi Perangkat Lunak

### Pengumpulan Data

Tahapan ini merupakan tahapan untuk mendapatkan data tweet yang terdapat di platform Twitter dengan cara scrapping laman Twitter menggunakan library Python tweepy dan membutuhkan akun serta akses developer twitter. Data didapatkan dengan membuat sebuah script Python bernama “data.ipynb”. Data yang dicari diantaranya adalah data tweet dibuat, data username, serta text dari tweet tersebut. Setelah didapatkan, maka data tweet tersebut akan disimpan dalam sebuah dataframe dan kemudian diexport menjadi sebuah file excel. Saat implementasi, didapatkan data sebanyak 11.646 sample data dari Twitter dengan bahasa inggris dan tidak disertakan link di dalam tweetnya. Listing program dalam mengumpulkan data dapat dilihat pada gambar 4.1 sampai 4.6.

Text

Description automatically generated

Gambar 4.1 import library Python dan inisialisasi token untuk mendapat akses ke akun developer twitter

Text

Description automatically generated

Gambar 4.2 Inisialisasi API developer twitter account

Text

Description automatically generated

Gambar 4.3 Membuat function untuk melakukan iterasisearching tweet

Text

Description automatically generated

Gambar 4.4 Membuat function untuk membuat dataframe data hasil scraping

A screenshot of a computer

Description automatically generated with medium confidence

Gambar 4.5 Searching tweets menggunakan function sebelumnya dan inisialisasi query untuk mencari data twitter



Gambar 4.6 Implementasi membuat dataframe dan menyimpan ke file excel.

Graphical user interface, text

Description automatically generated

Gambar 4.7 Data hasil scraping Twitter

### Preprocessing Data

Pada tahap ini, akan dijelaskan secara rinci bagaimana tahapan preprocessing data yang ada agar bisa digunakan untuk membangun model deep learning sentiment analysis ujaran kebencian Twitter. Preprocessing yang akan dilakukan diantaranya adalah seleksi data, lowercasing, pembersihan karakter selain alphanumeric, pembersihan karakter khusus/tidak biasa, membersihkan stopword, stemming, dan lemmatization.

#### Seleksi Data

Karena data yang didapatkan adalah data teks dari Twitter, maka aka nada banyak peluang bahwa data tersebut merupakan data retweet, dimana user dapat memposting ulang tweet orang lain, dan hal ini akan menimbulkan banyak duplikasi data. Oleh karena itu, dilakukan seleksi data dengan mengambil data yang tidak mengandung kata “RT” di dalam teks tersebut. Untuk implementasi hal tersebut, dapat dilihat pada table 4.8.

Text

Description automatically generated

Gambar 4.8 Implementasi seleksi data

Setelah dilakukan implementasi, didapatkan data hasil seleksi sebanyak 5206 kalimat.

#### Lowercasing

Pada tahapan ini akan diimplementasikan mengubah karakter huruf yang ada di dalam dataset menjadi bentuk huruf kecil. Untuk melakukan implementasi ini, memanggil function “lower()” dari modul String pada python yang dapat dilihat implementasinya pada gambar 4.9 dan untuk hasilnya dapat dilihat pada gambar 4.10.

A picture containing text

Description automatically generated

Gambar 4.9 Lowercasing.

Text

Description automatically generated

Gambar 4.10 Hasil lowercasing.

#### Pembersihan Karakter Non-Alphanumeric

Pembersihan Non-Alphanumeric bertujuan untuk membersihkan karakter selain huruf dan angka, seperti tanda baca dan karakter khusus lainnya. Tahapan ini dilakukan dengan menggunakan libraries re (regular expression) dengan fungsi sub yang berfungsi untuk mengganti kata yang sesuai dengan pola yang ada, yaitu pola karakter selain alphanumeric dan kemudian diganti dengan whitespace. Setelah itu, digabungkan menjadi satu kalimat lagi dengan memfilter apakah kata yang digabungkan merupakan digit atau bukan yang dapat dilihat implementasinya pada gambar 4.11 dan untuk hasilnya dapat dilihat pada gambar 4.12.

Text

Description automatically generated

Gambar 4.11 Remove Non-Alphanumeric

Text

Description automatically generated

Gambar 4.12 Hasil pembersihan karakter non alpha numeric.

#### Pembersihan Karakter Khusus

Pembersihan karakter khusus ini bertujuan untuk menghapus beberapa karakter khusus yang ada di dalam kalimat/teks, seperti kalimat yang mengandung kata “\n”, url, serta emoji. Untuk implementasi, menggunakan libraries re(regular expression) dengan fungsi sub untuk mengganti kata denga pola yang sesuai dengan pola yang dicari digantikan dengan string kosongdan dapat dilihat implementasinya pada gambar 4.13 dan untuk hasilnya dapat dilihat pada gambar 4.14.

Text

Description automatically generated

Gambar 4.13 Pembersihan karakter khusus

Text

Description automatically generated

Gambar 4.14 Hasil pembersihan karakter khusus

#### Stopword Removal

Tahapan pembersihan stopword berfungsi untuk menghilangkan kata umum (common words) yang biasanya muncul dalam jumlah besar dan dianggap tidak memiliki makna. Tahapan ini diawali dengan mengimport modul stopwords dari libraries nltk.corpus. Setelah itu, mengambil set stopword inggris dari modul stopwords. Setelah itu, melakukan iterasi terhadap stopword dan melakukan seleksi tiap kata yang ada di dalam kalimat. Setelah iterasi selesai dilakukan, maka akan mengembalikan teks tanpa stopwords. Untuk implementasi, dapat dilihat pada gambar 4.15 sampai dengan gambar 4.17.



Gambar 4.15 Import modul stopwords dari libraries nltk.corpus

Text

Description automatically generated

Gambar 4.16 Implementasi penghapusan stopwords

Text

Description automatically generated

Gambar 4.17 Hasil implementasi penghapusan stopwords

#### Stemming

Stemming merupakan teknik menguraikan bentuk suatu kata menjadi bentuk asalnya/dasarnya. Untuk melakukan tahapan ini, diperlukan import modul PorterStemmer dari libraries nltk.stem.porter. Setelah itu, menginisialisasi PorterStemmer. Setelah diinisialisasi, maka memanggil fungsi stem dari PorterStemmer untuk setiap kata yang ada di dalam kalimat/data. Untuk detail implementasi dapat dilihat pada gambar 4.18 sampai dengan 4.20.

Text

Description automatically generated

Gambar 4.18 Import dan menginisialisasi PorterStemmer

Text

Description automatically generated

Gambar 4.19 Fungsi stem dari PorterStemmer

Text

Description automatically generated

Gambar 4.20 Hasil dari stemming data.

#### Lemmatization

Seperti stemming, lemmatization juga mengubah kata menjadi bentuk dasarnya. Satu-satunya perbedaan adalah lemmatization memastikan bahwa kata dasar milik bahasa. Untuk melakukan tahapan ini, diawali dengan melakukan import modul WordLemmatizer dari libraries nltk.stem dan kemudian menginisialisasi WordNetLemmatizer. Setelah diinisialisasi, maka memanggil fungsi lemmatize dari WordNetLemmatizer untuk setiap kata yang ada di dalam kalimat/data. Untuk detail implementasi dapat dilihat pada gambar 4.21 sampai dengan 4.23.

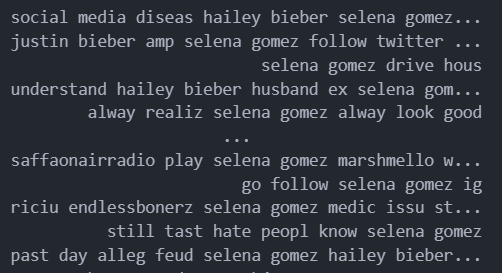


Gambar 4.21 Import dan inisialisasi WordNetLemmatizer

Text

Description automatically generated

Gambar 4.22 Implementasi lemmatization



Gambar 4.23 Hasil implementasi lemmatization

### Labelling Data

Proses labelling data merupakan tahapan memberikan label terhadap data yang dimiliki. Hal ini diperlukan karena untuk melakukan klasifikasi ujaran kebencian, diperlukan sebuah label untuk dapat melakukan training model deep learning nantinya. Label yang akan dibentuk diantaranya adalah positive, negative dan netral karena sentiment biasanya terdiri dari tiga label tersebut. Untuk melakukan labelling, digunakan libraries python untuk labelling sentiment analysis yaitu Vader. Cara kerja Vader yaitu setiap data teks yang ada di dalam dataset akan diimplementasikan ke dalam function polarity\_scores dari SentimentIntensityAnalyzer milik Vader dan kemudian akan mengembalikan berupa sebuah data bertipe data dictionary Python yang di dalamnya terdiri skor dari hasil polarity\_score terhadap masing-masing teks. Setelah didapatkan nilai tersebut, kemudian dikelompokkan/diberikan label pada niali “compound” yang ada di dalam data dictionary. Jika compound >=0.05 maka label yang diberikan adalah positive, jika diantara -0.05 dan 0.05 maka label yang diberikan netral, selain itu diberikan label negative. Untuk implementasi dan hasilnya dapat dilihat pada gambar 4.24 dan gambar 4.25.



Gambar 4.24 Implementasi Vader sentiment analysis

Text

Description automatically generated

Gambar 4.25 Hasil labelling menggunakan Vader

Setelah mendapatkan label data, kemudian label dipindahkan ke data yang sudah dilakukan preprocessing karena data yang diberikan label merupakan data yang murni, hal ini dilakukan agar ketika dilakukan penerjemahan dan labelling, tidak ada makna yang tereduksi. Setelah dijadikan satu antara data hasil preprocessing dan juga labelnya, maka melakukan transformasi label menjadi kategorikal angka dengan label 0,1 dan 2. Hal itu dilakukan untuk mempermudah dalam melakukan pemodelan data pada saat melakukan pemnbuatan model deep learning.

### Tokenizing Data

Tahapan ini akan mengubah data teks menjadi sebuah token yang merepresentasikan sebuah kata. Dalam tahapan ini, ada beberapa komponen penting yang harus diperhatikan yang dirangkum dalam table 4.1 sebagai berikut:

|  |  |
| --- | --- |
| Element | Keterangan |
| Vocab Size | Ukuran kata yang akan diubah menjadi index kata/kamus kata yang mewakili semua kata yang ada di dalam dataset |
| max\_len | Ukuran Panjang maksimal dari sebuah sequences/array yang menampung kumpulin token/index kata yang mewakili kata asli |
| Truncating type | Cara untuk memotong token dalam sequences/array yang ukurannya melebihi max\_len. |
| Padding type | Cara untuk memenuhi/menggenapi sequences yang ukurannya kurang dari max\_len agar mempunyai ukuran yang sama. |
| Oov token | Oov adalah singkatan dari Out of Vocabulary Token, yang berarti komponen yang akan menggantikan kata yang muncul di dalam dataset yang tidak terepresentasikan oleh word index/kamus kata. |

Tabel 4.1 Komponen penting dalam tokenizing

Pada tahapan ini, vocab size diset dengan nilai 6995, atau bisa dikatakan membuat kamus index dengan ukuran 6995 yang setara dengan Jumlah kata unik dalam dataset. Hal ini bearti semua kata di dalam dataset sudah diwakilkan oleh sebuah token yang disimpan dalam word index. Setelah mendapatkan banyaknya jumlah variasi kata, maka selanjutnya diperlukan mengetahui variasi panjang kalimat di dalam dataset tersebut. Untuk itu, dapat dilakukan dengan melakukan iterasi terhadap kalimat yang ada di dalam dataset dan kemudian menghitung panjang setiap kalimat. Untuk implementasi, dapat dilihat pada gambar 4.26 dan gambar 4.27.

Text

Description automatically generated

Gambar 4.26 Implementasi menghitung panjang setiap kalimat

A picture containing chart

Description automatically generated

Gambar 4.27 Hasil implementasi dan visualisasinya.

Pada gambar 4.27, dapat dilihat bahwa 75% data memiliki panjang kalimat 15 kata, dan oleh karena itu, akan lebih baik jika data panjang sequences/token diset menjadi 15 token. Setelah itu, truncating type yang digunakan ialah “post” yang berarti jika terdapat sebuah sequences/array yang memiliki Panjang lebih dari 15, maka yang akan dipotong/tidak diambil tokennya adalah bagian setelah token ke-15. Padding type pada tahapan ini menggunakan padding “post” yang berarti jika terdapat sebuah sequences/array yang panjangnya kurang dari 15, maka akan otomatis diisi oleh “0” setelah token terakhir yang ada di dalam sequences sampai dengan 100. Dan yang terakhir, oov token dari data ini menggunakan “<OOV>” yang berarti jika terdapat suatu kata yang tidak teridentifikasi di dalam index kata, maka akan digantikan oleh oov token yang telah dipasang. Untuk mengetahui lebih detail, dapat melihat lebih detail pada gambar 4.28 sampai dengan 4.30.

A screenshot of a computer

Description automatically generated with medium confidence

Gambar 4.28 Import libraries untuk melakukan tokenizing.

Text

Description automatically generated

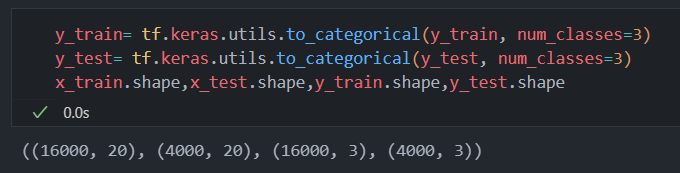
Gambar 4.29 Mempersiapkan element komponen penting tokenizing.



Gambar 4.30 Contoh hasil implementasi tokenizing.

### Pemisahan Data

Pada tahapan ini akan dilakukan pemisahan dataset menjadi dua bagian, yaitu data untuk dilatih dan data untuk menguji model yang telah dilatih. Dua puluh ribu data teks tweet akan dibagi dengan rasio 90% untuk training (16000 sampel data) dan 20% untuk testing (4000 sampel data). Pada tahapan ini menggunakan libraries train\_test\_split dari libraries sklearn yang nantinya akan otomatis membagi data dan label sesuai dengan rasio yang diinginkan. Setelah itu karena model membutuhkan input label berupa *one hot encode* label, maka label harus ditransformasi dari bentuk numeris menjadi array/list berisi tiga angka numeris yang merespresentasikan tiap label, dimana angka yang bernilai 1 di indeks label merupakan label tersebut. Untuk melakukan itu, menggunakan libraries dari keras yaitu function to\_categorical, dengan class yang diinginkan yaitu 3. Untuk detail implementasi, dapat dilihat pada gambar 4.28.



Gambar 4.28 Implementasi one hot encoding

Setelah proses tersebut, maka data siap digunakan dalam tahap training.

### Modelling

Dalam penelitian ini, ada beberapa model yang akan dirancang sesuai dengan skenario yang ada. Untuk membuat model deep learning, perlu melakukan import libraries tensorflow yaitu diantaranya. Sequentials, Embedding, Bidirectional, LSTM, Flatten, Dense, Adam, dan RMSProp dari modul keras. Setelah melakukan import, maka membuat sebuah function untuk membangun model sesuai dengan skenario yang diinginkan. Untuk implementasi model, dapat dilihat pada gambar 4.29.

Text

Description automatically generated

Gambar 4.29 Implementasi penyusunan model bagian 1.

Setelah model dibuat, maka dapat melakukan tahapan training model dimana model akan dilatih selama 20 epochs, dan rasio validasi data training sebesar 0.2. Dalam training, diinisialisasi sebuah callbacks yang berfungsi untuk menghentikan training ketika matrix yang menjadi acuan, baik akurasi maupun loss tidak menunjukan peningkatan signifikan. Callback yang digunakan yaitu EarlyStopping dengan patience sebesar 5, yang berarti training akan dihentikan ketika lima epoch setelah matrix acuan terbaik, dan akan mengembalikan bobot model terbaik jika nilai restore\_best\_weight bernilai True. Listing program yang dapat dilihat pada gambar 4.30 dan 4.31.

Text

Description automatically generated

Gambar 4.30 Implementasi callback



Gambar 4.30 Implementasi training model

### Evaluasi

Evaluasi model dilihat dari hasil akurasi akurasi, loss dan nilai f1\_m dari data training dan validasi serta hasil evaluasi model dengan memanggil function evaluate dari model keras. Selain itu, evaluasi dapat ditinjau jujga melalui confusion matrix yang dapat diperoleh dengan menggunakan function confusion\_matrix dari libraries sklearn, setelah itu untuk menampilkan visualisasinya dapat menggunakan ConfusionMatrixDisplay. Untuk mendapatkan confusion matrix, maka model harus melakukan prediksi data terlebih dahulu, setelah itu didapatkanlah label hasil prediksi. Hasil prediksi dan juga data testing kemudian dicocokan menggunakan confusion matrix. Akan tetapi, karena keduanya masih berupa array/list untuk setiap datanya, maka perlu dilakukan transformasi menggunakan function argmax dari libraries numpy untuk mengambil kelas yang memiliki nilai probabilitas prediksi paling besar. Setelah mendapatkan confusion matrixnya, maka dapat melakukan visualisasi menggunakan ConfusionMatrix Display. Untuk implementasi program, dapat dilihat pada gambar 4.32 sampai dengan gambar 4.3

Graphical user interface, text, application

Description automatically generated

Gambar 4.32 Implementasi function evaluate oleh model keras.

Text

Description automatically generated

Gambar 4.33 Implementasi Confusion Matrix.

Chart, treemap chart

Description automatically generated

Gambar 4.34 Output confusion matrix.

Setelah itu, untuk dapat menganalisa lebih detail terkait hasil training model, maka dibuatlah sebuah function untuk melakukan visualisasi hasil training yang terdiri dari element akurasi, loss dan f1 score terhadap iterasi epochs. Untuk implementasi, dapat dilihat pada gambar 4.35 dan gambar 4.36.

Text

Description automatically generated

Gambar 4.35 Implementasi visualisasi hasil training.

A picture containing text

Description automatically generated

Gambar 4.36 Implementasi visualisasi hasil training model.

## Analisa dan Pembahasan

Pada bagian ini, akan membahas tentang hasil training dan juga Analisa hasil training yang terjadi di setiap skenario. Skenario dibagi menjadi beberapa focus, yaitu model tanpa layer LSTM, model dengan layer LSTM serta model dengan layer Bidirectional LSTM.

### Pengujian Model Sederhana

Model sederhana yang disusun terdiri dari beberapa komponen penting diantaranya adalah layer dan acativation functionnya, optimizer, loss function, learning rate dan epoch. Layer yang akan disusun terdiri dari Embedding layer, LSTM layer, BiLSTM layer, Flatten layer dan Dense layer. Optimizer yang akan digunakan diantaranya adalah Adam dan RMSProp. Activation function yang akan digunakan untuk tiap layer diantaranya ialah untuk hidden layer berupa relu, dan untuk output layer berupa softmax. Loss function yang digunakan diataranya adalah categorical\_crossentropy dan mse (mean squared error).

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | Layer/Activation | Optimizer | Loss Function | Early stopping | Loss | Accuracy | F1 Score |
| 1 | * Embedding (16) * Flatten * Dense(3/softmax) | Adam | Categorical | 11 | 0.5954 | 0.7962 | 0.7914 |
| 2 | * Embedding (16) * Flatten * Dense(3/softmax) | Adam | MSE | 10 | 0.102 | 0.8 | 0.7916 |
| 3 | * Embedding (16) * Flatten * Dense(3/softmax) | RMSProp | Categorical | 20 | 0.6403 | 0.7972 | 0.7937 |
| 4 | * Embedding (16) * Flatten * Dense(3/softmax) | RMSProp | MSE | 14 | 0.1013 | 0.7995 | 0.7917 |
| 5 | * Embedding (16) * LSTM (64/tanh) * Flatten * Dense(3/softmax) | Adam | Categorical | 7 | 0.596 | 0.7908 | 0.7824 |
| 6 | * Embedding (16) * LSTM (64/tanh) * Flatten * Dense(3/softmax) | Adam | MSE | 10 | 0.1153 | 0.7885 | 0.7883 |
| 7 | * Embedding (16) * LSTM (64/tanh) * Flatten * Dense(3/softmax) | RMSProp | Categorical | 11 | 0.5781 | 0.7968 | 0.796 |
| 8 | * Embedding (16) * LSTM(64/tanh) * Flatten * Dense(3/softmax) | RMSProp | MSE | 10 | 0.1025 | 0.8002 | 0.7972 |
| 9 | * Embedding(16) * BiLSTM(64/tanh) * Flatten * Dense(3/softmax) | Adam | Categorical | 7 | 0.5901 | 0.7897 | 0.7834 |
| 10 | * Embedding(16) * BiLSTM(64/tanh) * Flatten * Dense(3/softmax) | Adam | MSE | 8 | 0.1052 | 0.799 | 0.7964 |
| 11 | * Embedding(16) * BiLSTM(64/tanh) * Flatten * Dense(3/softmax) | RMSProp | Categorical | 12 | 0.5877 | 0.7977 | 0.7942 |
| 12 | * Embedding(16) * BiLSTM(64/tanh) * Flatten * Dense(3/softmax) | RMSProp | MSE | 13 | 0.1071 | 0.795 | 0.7947 |

Tabel 4.2 Hasil percobaan model sederhana

Hasil percobaan diatas merupakan hasil training kombinasi antara arsitektur layer, optimizer serta loss function yang berbeda. Didapatkan dua belas kombinasi hasil training di model sederhana. Dari dua belas kombinasi, diambil tiga kombinasi model yaitu yang pertama model tanpa LSTM dan BiLSTM layer, menggunakan optimizer rmsprop serta loss function mse didapatkan akurasi setelah evaluasi model sebsar 0.7995; model kedua yaitu model dengan LSTM layer, menggunakan optimizer rmsprop serta loss function mse didapatkan akurasi setelah evaluasi sebesar 0.8002; model ketiga yaitu model dengan BiLSTM layer, menggunakan optimizer adam serta loss function mse didapatkan akurasi setelah evaluasi sebesar 0.799. Dari ketiga model terbaik, skor yang paling tinggi secara akurasi adalah model LSTM untuk akurasi dan f1 score tertinggi, diikuti oleh BiLSTM dan model tanpa LSTM maupun BiLSTM. Di dalam model 4 dan 8, menggunakan optimizer dan loss function yang sama yaitu rmsprop dan mse, dan pada model nomor 10 dan 12 hanya terpaut 0.004 di bagian akurasi dimana optimizer dan loss function yang digunakan sama dengan model nomor 4 dan 8 sehingga dapat dikatakan bahwa optimizer rmsprop dan loss function mse merupakan yang paling optimal di dalam scenario model sederhana ini. Setelah itu, dapat ditinjau lebih detail melalui confusion matrix dan juga grafik training tiap model yang dapat dilihat pada gambar

Chart, treemap chart

Description automatically generated

Gambar 4.37 Confusion matrix model nomor 4

Chart, line chart

Description automatically generated

Gambar 4.38 Grafik akurasi, validasi akurasi serta f1 score dari model nomor 4

Chart, treemap chart

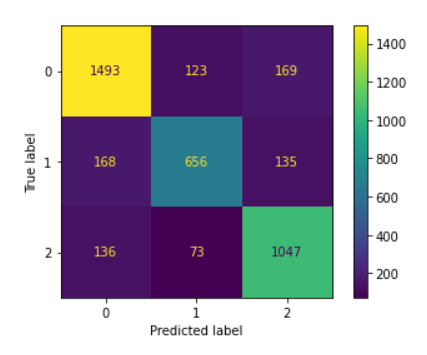
Description automatically generated

Gambar 4.37 Confusion matrix model nomor 8

Chart, line chart

Description automatically generated

Gambar 4.38 Grafik akurasi, validasi akurasi serta f1 score dari model nomor 8



Gambar 4.37 Confusion matrix model nomor 10

Chart, line chart

Description automatically generated

Gambar 4.38 Grafik akurasi, validasi akurasi serta f1 score dari model nomor 10.

Setelah diamati, ketiga model tersebut masih mengalami *overfitting* yang lumayan jauh jaraknya, antara akurasi dan f1 score dengan validasi akurasi dan validasi f1 score. Selain itu, untuk model nomor 4 dan 8 ditemukan noisenya lebih sedikit di semua matrix yang ada dibandingkan dengan model nomor 10 yang terlihat memiliki sedikit noise naik turun yang cenderung curam.

# Daftar Pustaka

# References

A, K. (n.d.). What are artificial neural network? *2008*. Retrieved from https://www.nature.com/articles/nbt1386

Abdelgwad, M. M., Soliman, T. H., Taloba, A. I., & Farghaly, M. F. (2021). Arabic aspect based sentiment analysis using bidirectional GRU based models. Retrieved from https://www.sciencedirect.com/science/article/pii/S1319157821002482

Ahmad, M., Ali, I., & Ahtab, S. (2017). *Sentiment Analysis of Tweets using SVM*. Retrieved from https://www.researchgate.net/profile/Shabib-Aftab-2/publication/321084834\_Sentiment\_Analysis\_of\_Tweets\_using\_SVM/links/5a1497b90f7e9b925cd514b0/Sentiment-Analysis-of-Tweets-using-SVM.pdf

algorit.ma. (2022). Apa Itu Long Short Term Memory Network (LSTM)? Retrieved from https://algorit.ma/blog/lstm-network-adalah-2022/

algorit.ma. (2022). Apa Itu Sentiment Analysis. Retrieved from https://algorit.ma/blog/sentiment-analysis-adalah-2022/

Colah. (2015). Understanding LSTM Networks. Retrieved from https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Datta, L. (2020). A Survey on Activation Functions and their relation with Xavier and He Normal Initialization.

Denny, M. J., & Spirling, A. (2017). Text Preprocessing For Unsupervised Learning: Why It. Retrieved from https://www.cambridge.org/core/journals/political-analysis/article/abs/text-preprocessing-for-unsupervised-learning-why-it-matters-when-it-misleads-and-what-to-do-about-it/AA7D4DE0AA6AB208502515AE3EC6989E

Dumane, G. (n.d.). Introduction to Convolutional Neural Network (CNN) using Tensorflow. *2020*. Retrieved from https://towardsdatascience.com/introduction-to-convolutional-neural-network-cnn-de73f69c5b83#:~:text=Dense%20Layer%20is%20simple%20layer,multiple%20number%20of%20such%20neurons.

Fauzi, M. A. (2018). *Random Forest Approach for Sentiment Analysis in Indonesian Language*. Retrieved from https://www.researchgate.net/profile/Muhammad-Fauzi-6/publication/327060733\_Random\_Forest\_Approach\_for\_Sentiment\_Analysis\_in\_Indonesian\_Language/links/5d305ff3458515c11c39adfd/Random-Forest-Approach-for-Sentiment-Analysis-in-Indonesian-Language.pdf

Fitri, V. A., Andreswari, R., & Hasibuan, M. A. (2019). *Sentiment Analysis of Social Media Twitter with Case of Anti-LGBT Campaign in Indonesia using Naïve Bayes, Decision Tree, and Random Forest Algorithm*. Retrieved from https://www.researchgate.net/profile/Muhammad-Fauzi-6/publication/327060733\_Random\_Forest\_Approach\_for\_Sentiment\_Analysis\_in\_Indonesian\_Language/links/5d305ff3458515c11c39adfd/Random-Forest-Approach-for-Sentiment-Analysis-in-Indonesian-Language.pdf

Huq, M. R., Ali, A., & Rahman, A. (2017). Sentiment Analysis on Twitter Data using KNN and SVM. Retrieved from https://pdfs.semanticscholar.org/05a8/78000170abcd0c6f8208080470858422e17c.pdf

Jianqiang, Z., Xiaolin, G., & Xuejun, Z. (2018). *Deep Convolution Neural Networks for Twitter Sentiment Analysi*. Retrieved from https://ieeexplore.ieee.org/abstract/document/8244338

Karani, D. (2018). Introduction to Word Embedding and Word2Vec. Retrieved from https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep Learning. Retrieved from https://www.nature.com/articles/nature14539

Liao, S., Wang, J., Yu, R., Sato, K., & Cheng, Z. (2016). CNN for situations understanding based on. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050917312103

Marimuthu, P. (2022). How Activation Functions Work in Deep Learning. Retrieved from https://www.kdnuggets.com/2022/06/activation-functions-work-deep-learning.html

MathWorks. (2022). What Is Deep Learning? Retrieved from https://www.mathworks.com/discovery/deep-learning.html#:~:text=Deep%20learning%20is%20a%20machine,a%20pedestrian%20from%20a%20lamppost

Medhat, W., Hassan, A., & Korashy, H. (2014). *Sentiment analysis algorithms and applications: A survey*. Retrieved from https://www.sciencedirect.com/science/article/pii/S2090447914000550

MIT. (2014). VADER-Sentiment-Analysis. Retrieved from https://github.com/cjhutto/vaderSentiment

Mungalpara, J. (2021). What does it mean by Bidirectional LSTM? Retrieved from https://medium.com/analytics-vidhya/what-does-it-mean-by-bidirectional-lstm-63d6838e34d9

Musstafa. (2021). Optimizers in Deep Learning. Retrieved from https://medium.com/mlearning-ai/optimizers-in-deep-learning-7bf81fed78a0

Permatasari, D. I., & Subyantoro. (2020). UJARAN KEBENCIAN FACEBOOK TAHUN 2017-2019. Retrieved from https://journal.unnes.ac.id/sju/index.php/jsi/article/view/33020

ReaserchGate. (n.d.). Gambar Embedding Layer BiLSTM. Retrieved from https://www.researchgate.net/publication/340716994/figure/fig3/AS:881352496918530@1587142218942/Network-Architecture-using-Bi-LSTM-model-with-input-being-Embedding-layer-and-PoS-vector.ppm

Rizaty, M. A. (2022). Pengguna Twitter di Indonesia Capai 18,45 Juta pada 2022. Retrieved from https://dataindonesia.id/digital/detail/pengguna-twitter-di-indonesia-capai-1845-juta-pada-2022

Rosyid. (2022). Hasil Survei Mengungkapkan Media Sosial Paling Digemari di Indonesia. Retrieved from https://www.gatra.com/news-548811-nasional-hasil-survei-mengungkapkan-media-sosial-paling-digemari-di-indonesia-.html#:~:text=Laporan%20We%20Are%20Social%20mengungkapkan,2021%20sebesar%20170%20juta%20orang.

Sastrawi. (2017). Sastrawi. Retrieved from https://github.com/sastrawi/sastrawi

Singhal, P., & Bhattacharyya, P. (2016). Sentiment Analysis and Deep Learning: A Survey. Retrieved from https://www.cfilt.iitb.ac.in/resources/surveys/sentiment-deeplearning-2016-prerna.pdf

Verma, Y. (2021). A Complete Understanding of Dense Layers in Neural Networks. Retrieved from https://analyticsindiamag.com/a-complete-understanding-of-dense-layers-in-neural-networks/

Wikipedia. (2022). Ucapan Kebencian. Retrieved from https://id.wikipedia.org/wiki/Ucapan\_kebencian

Wikipedia. (n.d.). Jaringan Syaraf Tiruan. *2022*. Retrieved from https://id.wikipedia.org/wiki/Jaringan\_saraf\_tiruan

Yadav, H. (n.d.). Dropout in Neural Networks. *2022*. Retrieved from https://towardsdatascience.com/dropout-in-neural-networks-47a162d621d9

Yathish, V. (2022). Loss Functions and Their Use In Neural Networks. Retrieved from https://towardsdatascience.com/loss-functions-and-their-use-in-neural-networks-a470e703f1e9

Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. Retrieved from https://direct.mit.edu/neco/article-abstract/31/7/1235/8500/A-Review-of-Recurrent-Neural-Networks-LSTM-Cells

# Jadwal Kerja

Isi jadwal kerja dari minggu ke minggu sesuai dengan contoh berikut ini.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Nama Kegiatan | Waktu / Tahun | | | | | | | | | | | |
| Nama Bulan | | | | Nama Bulan | | | | Nama Bulan dst | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |