ANALISIS PERBANDINGAN KINERJA METODE *WORD EMBEDDING GLOVE* DAN *WORD2VEC* DALAM ANALISIS SENTIMEN

SKRIPSI

Diajukan sebagai Salah Satu Syarat untuk Mendapatkan Gelar
Sarjana Komputer (S.Kom) Program Studi Informatika

AYU ANDIRA

105841108420

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS MUHAMMDIYAH MAKASSAR

2024

UNIVERSITAS MUHAMMADIYAH MAKASSAR

FAKULTAS TEKNIK

GEDUNG MENARA IQRA LT. 3

Jl. Sultan Alauddin No. 259 Telp. (0411) 866 972 Fax (0411) 865 588 Makassar 90221
Website: www.unismuh.ac.id, e_mail: unismuh@gmail.com
Website: http://teknik.unismuh.makassar.ac.id

بسم الله الرحمن الرحيم

PENGESAHAN

Skripsi atas nama Ayu Andira dengan nomor induk Mahasiswa 105 84 11084 20, dinyatakan diterima dan disahkan oleh Panitia Ujian Tugas Akhir/Skripsi sesuai dengan Surat Keputusan Dekan Fakultas Teknik Universitas Muhammadiyah Makassar Nomor : 221/05/A.5-VI/VII/45/2024, sebagai salah satu syarat guna memperoleh gelar Sarjana Komputer pada Program Studi Informatika Fakultas Teknik Universitas Muhammadiyah Makassar pada hari Jum'at tanggal 30 Agustus 2024.

Panitia Ujian :

1. Pengawas Umum — Makassar, 26 Safar 1446 H / 30 Agustus 2024 M
   a. Rektor Universitas Muhammadiyah Makassar
      Dr. Ir. H. Abd. Rakhim Nanda, ST., MT., IPU. : .....
   b. Dekan Fakultas Teknik Universitas Hasanuddin
      Prof. Dr. Eng. Muhammad Isran Ramli, ST., MT. : .....
2. Penguji
   a. Ketua : Dr. Ir. Zahir Zainuddin, M.Sc. .....
   b. Sekertaris : Muhyiddin A.M. Hayat, S.Kom., M.T. .....
3. Anggota : 1. Titin Wahyuni, S.Pd., MT. .....
   2. Rizki Yusliana Bakti ST., MT. .....
   3. Lukman, S.Kom., MT. : .....

Mengetahui :

Pembimbing I	Pembimbing II
Desi Anggreani, S.Kom., MT.	**Fahrim Irhamna Rahman S.Kom., MT**

Dekan

Dr. Ir. Hj. Nurnawaty, ST., MT., IPM.
NBM. 795 108

UNIVERSITAS MUHAMMADIYAH MAKASSAR
FAKULTAS TEKNIK
GEDUNG MENARA IQRA LT. 3
Jl. Sultan Alauddin No. 259 Telp. (0411) 866 972 Fax (0411) 865 588 Makassar 90221
Website: www.unismuh.ac.id, e_mail: unismuh@gmail.com
Website: http://teknik.unismuh.makassar.ac.id

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

HALAMAN PENGESAHAN

Tugas Akhir ini diajukan untuk memenuhi syarat ujian guna memperoleh gelar Sarjana Komputer (S.Kom) Program Studi Informatika Fakultas Teknik Universitas Muhammadiyah Makassar.

Judul Skripsi : **ANALISIS PERBANDINGAN KINERJA METODE *WORD EMBEDDING GLOVE* DAN *WORD2VEC* DALAM ANALISIS SENTIMEN**

Nama : Ayu Andira
Stambuk : 105 84 11084 20

Makassar, 30 Agustus 2024

Telah Diperiksa dan Disetujui
Oleh Dosen Pembimbing;

Pembimbing I	Pembimbing II
Desi Anggreani S, S.Kom., MT.	**Fahrim Irhamna Rahman S.Kom., MT.**

Mengetahui,
Ketua Program Studi Informatika

Muhyiddin A M Hayat, S.Kom., MT.
NBM : 1504 577

ABSTRAK

AYU ANDIRA, Analisis Perbandingan Kinerja Metode *Word Embedding GloVe* dan *Word2vec* Dalam Analisis Sentimen. (dibimbing oleh Desi Anggraeni, S,Kom.,M.T, dan Fachrim Irhamna Rachman, S.Kom., M.T).

Penelitian ini bertujuan untuk membandingkan kinerja metode *word embedding GloVe* dan *Word2Vec* dalam analisis sentimen menggunakan model *Convolutional Neural Network* (CNN). Data yang digunakan berupa saran dan kritik dari mahasiswa Universitas Muhammadiyah Makassar yang diambil melalui sistem SIMAK. Langkah awal dalam penelitian ini meliputi proses *preprocessing* data, penerapan embedding *GloVe* dan *Word2Vec*, serta implementasi model CNN untuk melakukan klasifikasi sentimen. Pada tahap evaluasi, hasil penelitian menunjukkan bahwa model CNN dengan embedding *Word2Vec* mampu mencapai akurasi validasi tertinggi sebesar 93.16%, sedangkan embedding *GloVe* hanya mencapai akurasi tertinggi sebesar 90.48%. Selain itu, *Word2Vec* juga menunjukkan performa yang lebih baik dalam metrik evaluasi lainnya, seperti presisi, recall, dan *F1-Score*. Dari hasil ini, dapat disimpulkan bahwa *Word2Vec* lebih efektif dalam menangkap fitur-fitur penting pada data teks yang digunakan, sehingga mampu menghasilkan prediksi yang lebih akurat dalam analisis sentimen dibandingkan dengan *GloVe*. Penelitian ini diharapkan dapat memberikan kontribusi bagi pengembangan teknik-teknik analisis teks serta pemrosesan bahasa alami, khususnya dalam penerapan metode embedding dalam klasifikasi sentimen.

Kata Kunci: Analisis Sentimen, *Word Embedding, GloVe, Word2Vec, Convolutional Neural Network (CNN)*

ABSTRACT

AYU ANDIRA, *Performance Analysis Comparison of GloVe and Word2Vec Word Embedding Methods in Sentiment Analysis. (Supervised by Desi Anggraeni, S.Kom., M.T., and Fachrim Irhamna Rachman, S.Kom., M.T.).*

This study aims to compare the performance of GloVe and Word2Vec word embedding methods in sentiment analysis using a Convolutional Neural Network (CNN) model. The data used consists of feedback and criticisms from students of Universitas Muhammadiyah Makassar, retrieved from the SIMAK system. The initial steps in this study include data preprocessing, the application of GloVe and Word2Vec embeddings, and the implementation of the CNN model to perform sentiment classification. In the evaluation phase, the results show that the CNN model with Word2Vec embedding achieved the highest validation accuracy of 93.16%, while GloVe embedding only reached a maximum accuracy of 90.48%. Additionally, Word2Vec demonstrated better performance in other evaluation metrics, such as precision, recall, and F1-Score. From these results, it can be concluded that Word2Vec is more effective in capturing important features from the text data, leading to more accurate sentiment analysis predictions compared to GloVe. This research is expected to contribute to the development of text analysis techniques and natural language processing, especially in the application of embedding methods in sentiment classification.

Keywords: *Sentiment Analysis, Word Embedding, GloVe, Word2Vec, Convolutional Neural Network (CNN)*

KATA PENGANTAR

Dengan mengucapkan puji Syukur kehadirat Allah SWT yang telah memberikan petunjuk, pertolongan serta seluruh nikmat-nya kepada kita, dengan izin dan ridho-nya lah sehingga skripsi dengan judul "**ANALISIS PERBANDINGAN KINERJA METODE *WORD EMBEDDING GLOVE* DAN *WORD2VEC* DALAM ANALISIS SENTIMEN"** ini dapat penulis selesaikan sebagaimana salah satu syarat untuk penyusunan skripsi program studi informatika. Shalawat serta salam tak lupa kami ucapkan kepada baginda Rasulullah, Nabi Muhammad Shallallahu alaihi wassallam.

Penulis menyadari bahwa dalam proses penyusunan proposal skripsi ini banyak pihak yang telah memberikan bantuan, bimbingan, dan dukungan, baik secara langsung. Oleh karena itu,dengan segala kerendahan hati, penulis ingin menyampaikan terima kasih sebesar – besarnya kepada :

1. Allah SWT, yang telah memberikan petunjuk, kekuatan, dan rahmat-Nya selama proses penulisan.
2. Kedua orang tua tercinta yang telah memberikan kasih sayang, doa, serta dukungan yang tiada henti. Terimakasih atas segala pengorbanan, semangat, dan cinta yang senantiasa mengiringi setiap Langkah penulis. Tanpa kehadiran dan restu kalian penulis tidak mampu mencapai titik ini. Semoga Allah SWT senantiasa melimpahkan rahmat dan keberkahan kepada Bapak dan Ibu.
3. Bapak Dr. Ir. H. Abd. Rakhim Nanda, MT., IPU., sebagai Rektor Perguruan Tinggi Universitas Muhammadiyah Makassar.
4. Ibu Dr. Ir. Hj. Nurnawaty, ST., MT., Selaku Dekan Fakultas Teknik Universitas Muhammadiyah Makassar.
5. Bapak Muhyiddin A M Hayat, S.Kom. M.T., Selaku Ketua Prodi Informatika, Fakultas Teknik Universitas Muhammadiyah Makassar.
6. Ibu Desi Anggraeni S, S.Kom., M.T, Selaku Dosen Pembimbing I yang telah banyak memberikan bimbingan dan bantuannya.

7. Bapak Fahrim Irhamna Rachman,S.Kom.,M.T., Selaku Dosen Pembimbing II dan Pendamping Akademik yang telah banyak memberikan bimbingan dan bantuannya.
8. Seluruh Dosen dan Staf Fakultas Teknik Universitas Muhammadiyah Makassar
9. Teman-teman seperjuangan kelas C Program Studi Informatika Angkatan 2020, Khususnya kepada Rizka Adrianingsih, Lis Indriani, Rosalinda Aprilia Sari, Arya Wibawa.Ar, David Arian Virgiawan. yang telah menjadi keluarga kedua bagi penulis. Terima kasih atas segala dukungan, bantuan, saran, dan kebersamaan yang telah kalian berikan kepada penulis. Ketulusan, semangat dan kebersamaan yang kalian berikan telah mewarnai perjalanan penulis dalam menyelesaikan skripsi ini.
10. Kepada sahabat- sahabat penulis Regita Nur oktaviani, dina julianti, dan vingky Anggraeni Terima kasih atas kesetiaan, kepedulian, dan ketulusan yang kalian berikan kepada penulis.

Untuk itu penulis menyadari bahwa skripsi ini masih jauh dari sempurna dan masih memerlukan saran serta kritik yang membangun dari berbagai pihak untuk penyempurnaan proposal ini. Semoga proposal skripsi ini dapat bermanfaat bagi semua pihak yang membutuhkan.

Makassar, 16 Juni 2024

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

DAFTAR ISTILAH

Analisis	Analisis adalah proses untuk memeriksa atau membagi sesuatu menjadi bagian-bagian yang lebih kecil agar lebih mudah dipahami, dipelajari, atau dievaluasi.
Word embedding	*Word embedding* adalah teknik dalam pemrosesan bahasa alami yang mengubah kata-kata menjadi representasi numerik dalam bentuk vektor
GloVe	*GloVe Embedding* merupakan jenis penyisipan kata yang mengodekan rasio probabilitas kemunculan bersamaan antara dua kata sebagai perbedaan vektor.
Word2vec	*Word2vec* mengubah teks menjadi vektor yang menangkap semantik dan hubungan antarkata.
CNN	CNN (*Convolutional Neural Network*) adalah jenis jaringan saraf tiruan yang dirancang secara khusus untuk mengolah data yang memiliki struktur grid, seperti gambar.
NLP	(*Natural Language Processing*) Cabang kecerdasan buatan yang berfokus pada interaksi antara komputer dan bahasa manusia, termasuk generasi bahasa.
Layer	*Layer* (lapisan) merupakan suatu konsep yang digunakan untuk mengorganisasi sistem yang kompleks menjadi bagian - bagian lebih kecil dan terdefenisi secara jelas.
Preprocessing	Langkah-langkah awal untuk membersihkan dan menyiapkan data sebelum digunakan dalam analisis atau pelatihan model, seperti normalisasi dan pembersihan data.

Tensorflow	*Tensorflow* adalah *library* dan *framework open-source* untuk pemrograman numerik dan kumputasi sains yang dikembangkan oleh google.
Flowchart	Visualisasi langkah-langkah dalam sebuah proses menggunakan simbol dan panah untuk menunjukkan alur dan keputusan. Berguna untuk merencanakan dan menganalisis proses.
Epoch	Satu siklus lengkap di mana model telah melihat seluruh dataset sekali. Pelatihan biasanya dilakukan dalam beberapa epoch untuk meningkatkan akurasi model.

BAB I

PENDAHULUAN

A. Latar Belakang

Analisis sentimen adalah proses untuk mengidentifikasi dan mengekstrak sentimen atau opini dari teks (Listyarini & Anggoro,2021). Dalam konteks akademik, analisis sentimen dapat memberikan wawasan tentang kepuasan dan ketidakpuasan mahasiswa terhadap layanan yang diberikan oleh institusi pendidikan (Informatika et al., 2022). Salah satu penerapannya adalah dalam memproses saran dan kritik mahasiswa untuk mengevaluasi kualitas pendidikan dan layanan. Di era digital, perguruan tinggi menghadapi tantangan dalam mengelola data yang besar dan beragam, termasuk umpan balik mahasiswa. Untuk itu, Sistem Informasi Akademik (SIMAK) digunakan oleh institusi pendidikan untuk meningkatkan pelayanan kepada mahasiswa secara lebih efektif (Riani et al., 2021).

Untuk melakukan analisis sentimen, diperlukan metode yang dapat merepresentasikan kata-kata dalam bentuk yang dapat diproses oleh mesin. *Word embedding* telah menjadi salah satu pendekatan yang paling efektif dalam representasi teks untuk analisis sentimen. *Word embedding* adalah teknik yang digunakan untuk memetakan kata-kata menjadi vektor-vektor dalam ruang vektor berdimensi tinggi di mana hubungan semantik antar kata-kata dapat terjaga (Dyantono & Putra, 2023). Dua metode *word embedding* yang paling populer adalah *GloVe (Global Vectors for Word Representation)* dan *Word2vec*. Kedua metode ini memiliki pendekatan yang berbeda dalam pembelajaran representasi kata, dan masing-masing memiliki kelebihan dan kekurangan dalam berbagai aplikasi (Nurdin et al., 2020).

GloVe adalah metode *word embedding* yang mengandalkan statistik global dari *co-occurrence* (kejadian bersama) kata-kata dalam korpus teks. Metode ini berfokus pada pemodelan matriks kejadian bersama dan melakukan dekomposisi matriks untuk menghasilkan representasi vektor kata. Sebaliknya, *Word2vec* menggunakan jaringan saraf dalam dua model utamanya, *Continuous*

Bag of Words (CBOW) dan *Skip-gram*, untuk memprediksi kata-kata berdasarkan konteks lokal di sekitar kata target (PRADANA, 2023).

Dalam penelitian ini, model *Convolutional Neural Network* (CNN) digunakan untuk mengolah representasi *word embedding* dalam analisis sentimen. CNN, meskipun lebih dikenal dalam pengolahan gambar, juga efektif dalam menganalisis teks karena kemampuannya untuk menangkap fitur lokal dan pola dalam teks.(Wulansari et al., n.d.)

Meskipun banyak penelitian telah dilakukan untuk mengevaluasi kinerja kedua metode ini dalam berbagai tugas NLP, masih diperlukan analisis mendalam dan komparatif untuk menentukan metode mana yang lebih efektif dalam analisis sentimen (Indrayana & Solikhin, 2020). Dalam konteks saran mahasiswa, analisis sentimen dapat digunakan untuk mengidentifikasi perasaan dan opini mahasiswa terhadap kurikulum, fasilitas kampus, dan pengalaman belajar mereka. Dengan memahami sentimen mahasiswa, institusi pendidikan dapat mengambil langkah-langkah yang tepat untuk meningkatkan kualitas pendidikan dan kesejahteraan mahasiswa.(Yuliska et al., 2021)

Oleh karena itu, penelitian ini bertujuan untuk melakukan analisis perbandingan kinerja metode *GloVe* dan *Word2vec* dalam analisis sentimen, dengan fokus pada saran dan opini mahasiswa. Hasil dari penelitian ini diharapkan dapat memberikan panduan bagi institusi pendidikan dalam memilih metode yang paling sesuai untuk menganalisis sentimen mahasiswa, serta memberikan kontribusi bagi pengembangan lebih lanjut dalam bidang analisis teks dan NLP.

B. Rumusan Masalah

Berdasarkan latar belakang di atas dapat dirumuskan permasalahan yaitu sebagai berikut:

1. Bagaimana model CNN diimplementasikan dalam analisis sentimen ketika menggunakan *word embedding GloVe* dan *Word2vec*?
2. Bagaimana perbandingan kinerja antara metode *word embedding GloVe* dan *Word2vec* dalam analisis sentimen menggunakan model CNN?

C. Tujuan Penelitian

Adapun Tujuan dari penelitian ini berdasarkan rumusan masalah yang sudah dipaparkan sebagai beriku:

1. Mengetahui bagaimana model CNN diimplementasikan dalam analisis sentimen Ketika menggunaakan *word embedding Glove* dan *Word2vec*.
2. Menganalisis dan membandingkan kinerja metode *word embedding GloVe* dan *Word2vec* dalam analisis sentimen dengan menggunakan model CNN

D. Manfaat Penelitian

Adapun manfaat yang diharapkan dari hasil penelitian ini adalah sebagai berikut:

A. Bagi Peneliti
   1. Peneliti dapat memberikan wawasan yang lebih dalam mengenai perbandingan kinerja kedua metode *word embedding* yaitu *GloVe* dan *Word2vec* dalam konteks analisis sentimen.
   2. Hasil penelitian dapat digunakan sebagai referensi untuk penelitian selanjutnya yang berfokus pada pengembangan model analisis sentimen dengan Teknik yang lebih baik.

B. Bagi Akademis
   1. Penelitian Ini akan menambah literatur dibidang pemrosesan Bahasa alami (NLP) dan pembelajaran mendalam (*Deep learning*), khususnya dalam penggunaan metode *word embedding GloVe* dan *Word2vec* dalam analisis sentimen.
   2. Dapat menjadi referensi bagi peneliti selanjutnya.

C. Bagi Universitas
   1. Peneliti dapat meningkatkan reputasi universitas sebagai Lembaga yang aktif dalam penelitian teknologi, khususnya bidang NLP dan analisis sentimen.
   2. Dapat digunkan sebagai bahan acuan dalam penyusunan kurikulum yang relevan dengan perkembangan ilmu pengetahuan dan teknologi, serta mendorong kalaborasi penelitian antara universitas dan industri.

E. Ruang Lingkup Penelitian

Perlunya ada Batasan dalam setiap penelitian agar menjadi terarah dengan baik, maka ruang lingkup pada penelitian ini adalah sebagai berikut:

1. Metode klasifikasi yang digunakan yaitu *convolutional Neural Network* (CNN).
2. Penelitian ini menggunakan data saran dan kritik mahasiswa Universitas Muhammadiyah Makassar yang diambil dari sistem SIMAK.
3. Parameter yang digunakan untuk membandingkan kinerja metode adalah nilai akurasi masing-masing model.

F. Sistematika Penulisan

Pada bagian ini diuraikan sistematika penulisan yang disajikan secara sistematis sebagai berikut:

BAB I PENDAHULUAN

Bab ini membahas terkait latar belakang masalah, rumusan masalah, tujuan penelitian, manfaat penelitian, ruang lingkup penelitian serta sistematik penulisan

BAB II TINJAUAN PUSTAKA

Bab ini membahas landasan teori yang meliputi teori-teori terkait dan kerangka pikir.

BAB III METODE PENELITIAN

Bab ini membahas tentang metode penelitian yang akan digunakan dalam melakukan penelitian beserta Langkah-langkah dalam penelitian.

BAB IV HASIL DAN PEMBAHASAN

Bab ini membahas tenttang hasil dan pembahasan dalam mengimplementasi algoritma *word embedding GloVe* dan *Word2vec*.

BAB V PENUTUP

Bab ini membahas tentang Kesimpulan dari penelitian dan saran untuk penelitian berikutnya.

BAB II
TINJAUAN PUSTAKA

A. Landasan Teori

1. Analisis Sentimen

Analisis sentimen merupakan salah satu Teknik *Natural Language Processing* (NLP) yang menganalisis pendapat, sikap, dan emosi terhadap suatu entitas yang berupa teks. Analisis sentimen diperlukan sebagai bahan evaluasi yang selanjutnya menjadi dasar dalam pengambilan Keputusan (Arsi & Waluyo, 2021). Analisis sentimen ini digunakan untuk menemukan pendapat yang bermuatan positif, negative. Analisis sentimen ini memiliki tugas untuk membuat sebuah teks tersebut menjadi terstruktur sehingga dapat dilakukan klasifikasi (Yuniarossy et al., 2024). Tujuan dari analisis sentimen adalah untuk memahami dan mengekstraksi informasi subjektif yang terkandung dalam teks, seperti ulasan produk, komentar pengguna, atau konten media sosial. Metode analisis sentimen melibatkan penggunaan teknik-teknik pemrosesan bahasa alami dan model statistik untuk mengolah data teks dan mengenali pola-pola sentimen yang terkandung di dalamnya. Hal ini memungkinkan analisis sentimen untuk memberikan wawasan yang berharga tentang persepsi dan pandangan pengguna terhadap suatu topik atau produk tertentu (Ardian Pradana et al., 2023).

Analisis sentimen dapat digunakan salah satunya untuk melakukan monitoring terhadap kualitas atau performa sebuah produk dan pelayanan sebuah Lembaga. Dengan menerapkan analisis sentimen pengembang produk dan pemilik layanan dapat dengan mudah mengetahui apakah sebuah produk dan layanan diterima secara positif oleh pelanggan atau malah sebaliknya (Nurdin et al., 2020).

2. Word embedding

Pada tahun 2003, Bengio dkk memperkenalkan istilah *word embedding. Word embedding* adalah sebuah fungsi parameter yang memetakan setiap kata kedalam vector berdimensi tinggi. Teknik ini bertujuan untuk mengubah kata-kata yang merupakan data kualitatif menjadi data kuantitatif yang dapat digunakan oleh algoritma *machine learning*. Representasi *vecto*r ini memungkinkan model untuk memahami dan menangkap hubungan s*emantic* antara kata-kata, Dimana kata-kata dengan makna yang mirip memiliki representasi vector yang dekat satu sama lain dalam ruang *vector* (Nurdin et al., 2020).

Word embedding dapat dibuat langsung dari dataset yang dimiliki atau menggunakan *pre-trained word embedding* yang telah tersedia. *Pre-trained word embedding* ini adalah *word embedding* yang telah dilatih menggunakan dataset yang besar pada domain permasalahan tertentu yang dapat digunakan untuk menyelesaikan permasalahan lain yang serupa. Pada penelitian yang dilakukan oleh Liang-Chih Yu membahas tentang *word embedding* seperti, *GloVe* dan *Word2vec* pada analisis sentimen untuk menangkap sintaks dan semantik kata (Fitriansyah et al., 2023).

3. GloVe

Global vector for word Representation (*GloVe*) adalah Teknik *word embedding* atau representasikan vector dari kata dari setiap dokumen. tujuan dari proses representasi kata menggunakan metode *GloVe* ini adalah bisa mendapatkan hubungan semnatik antara kata-kata dari matriks kemunculan Bersama *word embedding GloVe* membentuk corpus dengan membuat matriks kemunculan (*co-occurrence matrix*) yang menunjukkan hubungan antara kata dan seberapa sering kata tersebut muncul (Faiq et al., 2022)

Algoritma *GloVe* juga menggabungkan informasi *co-occurance* kata atau *statistic* global untuk memperoleh hubungan semantik antar kata dalam korpus. *GloVe* mempelajari hubungan kata kata muncul secara Bersama satu sama lain dalam sebuah korpus yang diberikan. Rasio propabilitas kemunculan kata-kata memiliki potensi untuk mengkodekan beberapa bentuk makna kata serta

membantu meningkatkan kinerja pada permasalahan analogi kata.(Nurdin et al., 2020).

Algoritma *GloVe* terdiri dari Langkah-langkah berikut :

1) Mengumpulkan statistic word *co-occurance* dalam bentuk sebuah matriks word *co-occurance* X. Setiap elemen X_{ij} mempresentasikan beberapa kali kata *i* muncul dalam kata *j*. Jika kata j muncul dalam jendela konteks kata *i*, X_{ij} ditingkatkan dengan satu. Matriks ini simetris denggan berukuran V x V, Dimana V adalah jumlah total kata unik dalam korpus.
2) Menentukan *soft constraints* untuk setiap pasangan kata :

$$W_i^T X_j + b_i + b_j = \log(X_{ij}) \quad (1)$$

Dimana W_i – vector kata utama, X_j vector kata konteks, b_i , b_j bias skalar untuk kata-kata utama dan kata-kata konteks.

3) Menentukan Sebuah *Cost Function*.

$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} f(X_{ij}) (w_i^T \tilde{w}_j + b_i + \tilde{b}_j - logX_{ij})^2 \quad (2)$$

Dimana disini, w_i dan $\tilde{w}_j$ adalah vector embedding untuk kata *i* dan kata konteks *j*, sementara b_i dan $\tilde{b}_j$ adalah bias. Fungsi $f(X_{ij})$ adalah fungsi pembobotan yang memberikan bobot lebih pada pasangan kata dengan frekuensi co-occurance yang lebih tinggi. Fungsi tersebut didefenisikan sebagai berikut :

$$f(x) = \begin{cases} (\frac{X_{ij}}{X_{max}})^{\alpha} & if X_{ij} < X_{max} \\ 1; lainnya \end{cases} \quad (3)$$

Di sisni X_{max} adalah ambang batas dan α biasanya diatur sekitar 0.75

4. Word2vec

Word2vec adalah sebuah metode atau algoritma dalam bidang pemrosesan bahasa alami (*natural language processing*) yang digunakan untuk menciptakan representasi vektor kata (*word embedding*) dari teks. Algoritma

ini dikembangkan oleh Tomas Mikolov dan timnya pada tahun 2013. Tujuan dari *Word2vec* adalah untuk menghubungkan setiap kata dalam teks ke dalam ruang vektor, di mana kata-kata dengan makna yang mirip atau sering kali muncul bersama-sama akan memiliki representasi vektor yang berdekatan satu sama lain. Representasi vektor ini memungkinkan komputer untuk mengenali dan memahami hubungan semantik antara kata-kata (Ardian Pradana et al., 2023)

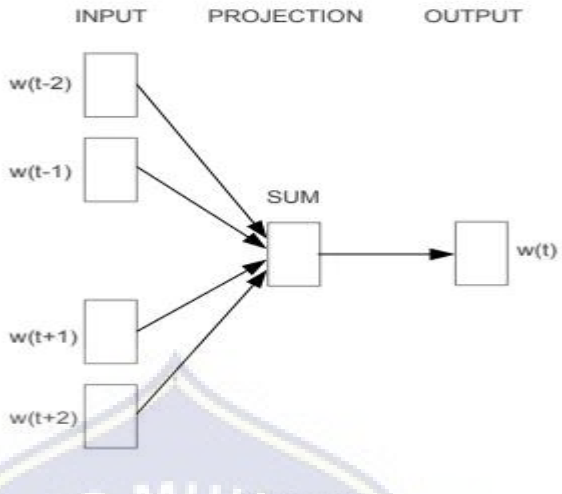
Model *word embedding* ini adalah salah satu aplikasi unsupervised learning menggunakan jaringan saraf (*neural network*) yang terdiri dari sebuah *hidden layer* dan *fully connected layer* (Maulana et al., 2023). Dimensi bobot setiap lapisan adalah jumlah kata dalam korpus (Kumpulan teks) dikalikan dengan jumlah sel/neuron tersembunyi pada lapisan tersembunyi. Matriks bobot pada *hidden layer* dari model yang telah dilatih digunakan untuk mentransformasikan kata kedalam vector. Matriks bobot ini seperti *lookup table*, Dimana setiap baris mewakili setiap kata dan kolom mewakili vector dari kata tersebut (Nurdin et al., 2020)

Word2vec mentransformasikan operasi dokumen menjadi perhitungan vektor dalam ruang vektor kata. Relasi semantic pada dokumen dapat dikarakterisasi berdasarkan kesamaan kata dalam ruang vektor. Tahap awal pada proses *Word2vec*, yaitu membangun kosakata dari data teks pelatihan dan kemudian mempelajari representasi vektor dari Kumpulan kata. Vektor yang dihasilkan dapat digunakan sebagai fitur untuk penerapan dalam kasus *natural laguangae processing dan machine learning* (Informatika et al., 2022).

Terdapat dua algoritma *Word2vec* yaitu *Continous Bag of-word* (CBOW) dan *Skip -gram*.

a. CBOW

CBOW (*Continous bags of words*) merupakan sebuah model arsitektur yang didasarkan pada *neural network* Dimana non-linear *hidden layer* dihilangkan dan *projection layer* dibagikan ke semua kata. CBOW memprediksi suatu kata berdasrkan kata-kata lain yang berada dalam suatu kalimat.



Gambar 1 Model Arsitektur CBOW (Mikolov et al., 2013)

Gambar di atas menunjukkan arsitektur model *Continuous Bag of Words* (CBOW), salah satu metode yang digunakan dalam *Word2vec* untuk menghasilkan representasi vektor dari kata-kata dalam sebuah teks.

1. Input

   Gambar menunjukkan bahwa input model CBOW adalah beberapa kata di sekitar (context words) kata target. Di sini, kata target w(t) dikelilingi oleh dua kata sebelumnya w(t−2) dan w(t−1)), serta dua kata sesudahnya w(t+1) dan w(t+2). Input ini adalah kata-kata konteks yang digunakan untuk memprediksi kata target.

2. Projection (projeksi)

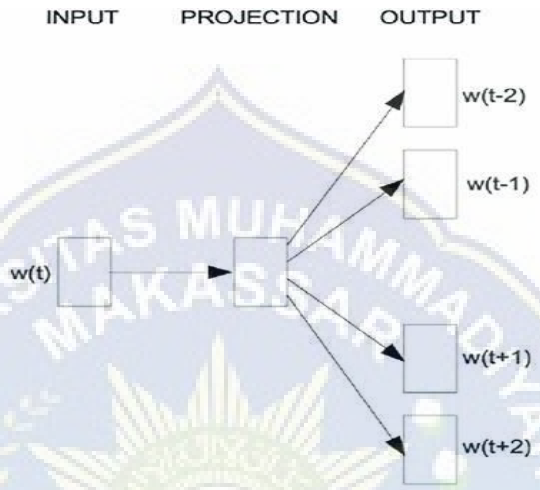
   Kata-kata konteks diwakili dalam bentuk vektor dan digabungkan (biasanya melalui penjumlahan atau rata-rata) untuk membentuk vektor proyeksi. Proses ini bertujuan untuk menghasilkan satu vektor yang merangkum informasi dari semua kata konteks.

3. Output

   Vektor proyeksi kemudian digunakan untuk memprediksi kata target w(t). Output dari model ini adalah representasi vektor dari kata target yang paling sesuai dengan vektor proyeksi yang telah dibentuk dari kata-kata konteks.

b. Skip- Gram

Model ini menggunakan sebuah kata untuk memperediksi target konteks. Skip-gram bertugas memprediksi konteks berdasarkan kata target yang dimiliki.



Gambar 2 Model Arsitektur Skip-Gram (Mikolov et al., 2013)

Gambar di atas menunjukkan arsitektur model *Skip-Gram*, yang merupakan bagian dari teknik *Word2vec* untuk representasi kata (*word embedding*).

1. Input

   Pada tahap ini, *input* adalah kata pusat, yang dilambangkan sebagai w(t). Kata ini digunakan untuk memprediksi konteks sekitarnya.

2. Projection Layer

   Di lapisan ini, kata pusat diubah menjadi representasi vektor melalui proses embedding. Vektor ini kemudian digunakan untuk menentukan distribusi kata-kata di sekitar kata pusat.

3. Output

   Lapisan *output* menghasilkan beberapa kata konteks berdasarkan kata pusat w(t)w(t)w(t). Pada gambar tersebut, kata-kata konteks yang dihasilkan adalah w(t−2), w(t−1), w(t+1) dan

w(t+2) Ini menunjukkan kata-kata yang ada di sekitar kata pusat dalam jangkauan tertentu.

Model arsitektur *skip-gram* mirip seperti *CBOW*. Pelatihan model *skip-gram* yaitu untuk representasi kata yang berguna dalam memprediksi kata-kata yang ada di dekatnya pada suatu kalimat atau dokumen. *Skip-gram* melakukan proses prediksi dengan melihat yang ada di sebelum dan sesudah *current word.* Model skip-gram memaksimalkan rata-rata *log probability* yang dapat dilihat pada persamaan berikut :

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log P\left(w_{t+j}|w_t\right) \quad (4)$$

w_t = kata *center*

w_{t+j} = kata setelah kata center

C = ukuran training context

Nilai vector dalam penelitian ini merupakan nilai vector cuitan (tweet). dilai vektor dokumen di dapatkan dengan menghitung nilai rata-rata vektor dari semua kata dalam satu cuitan.

$$vec_{c,j}\ \ \frac{1}{n_j}\sum_{i=1}^{m} w_i V(w_i) \quad (5)$$

$vec_{c,j}$ = Vektor dokumen j pada class c

n_j = jumlah fitur dalam dokumen j

$V_{(Wi}$ = vektor kata dari fitur w

w_i = nilai vektor dari fitur w

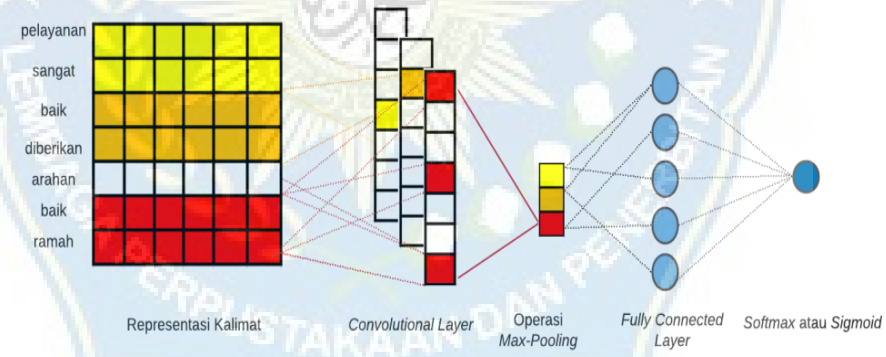
5. *Convolutional Neural Network* (CNN)

CNN adalah salah satu jenis arsitektur jaringan saraf tiruan yang paling umum digunakan untuk tugas pengolahan gambar dan teks. CNN memiliki

kemampuan otomatis untuk mengekstraksi dan mempelajari fitur-fitur penting berdasarkan data masukan, sehingga model ini sangat efektif dalam mengatasi permasalahan klasifikasi, deteksi, dan segmentasi.

CNN (*Convolutional neural network*) pada awalnya digunakan pada bidang *image processing* atau *computer vision*. Kemudian pada tahun2015, CNN digunakan pada bidang natural Language processing (NLP), yaitu untuk melakukan klasifikasi teks (Kim,2014). Inti dari CNN dalam mengklasifikasi teks yaitu dengan menerapkan teknik *convolution* terhadap kalimat, paragraf, atau keseluruhan dokumen teks. *Convolution* pada dasarnya adalah *sliding window* yang disebut juga dengan filter, yaitu membagi representasi teks yang berupa matriks menjadi beberapa window, kemudian menjumlahkannya. Hasil penjumlahan inilah yang menjadi representasi baru dari teks yang disebut dengan *feature maps* (Yuliska et al., 2021).

5.1 Arsitektur model CNN



Gambar 3 Model Arsitektur CNN (Yuliska et al., 2021)

Pada lapisan pertama merupakan input layer yang menyimpan teks yang akan diubah menjadi matriks dari gabungan vektor representasi kata. Dimensi matriks dari teks adalah n x k, Dimana n adalah Panjang dari sebuah teks dan k adalah dimensi dari *word vector*.

Kemudian lapisan kedua adalah *Convolutional Layer*, layer ini akan melakukan konvolusi terhadap data input dengan menggeser sebuah filter agar dapat menghasilkan output.

Lapisan ketiga pada CNN adalah *pooling Layer,* lapisan yang mengurangi dimensi dari output hasil konvolusi menjadi data dengan ukuran yang lebih kecil sehingga mudah mengontrol *overfitting*. proses poling dapat dilakukan menggunakan *max poling* atau *average pooling.*

Lapisan terakhir pada CNN adalah *Fully Connected Layer*, pada lapisan ini setiap neuron terhubung kesemua aktivasi dari lapisan sebelumnya. Pada lapisan ini merupakan klasifikasi teks yang diekstark fiturnya pada lapisan - lapisan sebeumnya.

Sigmoid adalah Hasil akhir dari fully connected layer terhubung pada sigmoid layer yang hasil akhirnya berupa 2 kelas. *softmax* Pada sub *multiclass classification,* hasil dari *fully connected layer* terhubung ke softmax layer yang memberikan hasil akhir berupa K buah kelas.

B. Penelitian Terkait

1. Intan Wulansari, Rifiana Arief (2023)

   Pada penelitian Intan Wulansari, Rifiana Arief yang berjudul "Analisis Performa Metode *Convolutional Neural Network* (CNN) Dengan *Word embedding GloVe* Pada Klasifikasi Sentimen dari Twitter". Tujuan penelitian ini adalah menentukan metode terbaik dalam melakukan analisis sentimen pada data *tweets* yang diambil dari media social Twitter. Metode analisis sentimen yang digunakan adalah model CNN dengan *word embedding GloVe* dan tanpa *GloVe.* Hasil penelitian menunjukkan bahwa model CNN menggunakan *word embedding GloVe* 100 dimensi mendapatkan nilai akurasi yang cukup tinggi yaitu 82.86 %, *presisi* 71%, *F1-socre* 70% dan waktu yang dibutuhkan untuk melakukan training selama 2 jam 47 menit 59 detik. Model CNN menggunakan *word embedding* tanpa *GloVe* memiliki nilai akurasi lebih kecil yaitu 77.46%, *presisi* 69% dan *F1-Score* 69.4% dan waktu yang dibutuhkan untuk melakukan training model CNN tanpa *GloVe* membutuhkan waktu selama 5 jam 29 menit 15 detik. Berdasarkan performa model klasifikasi maka model terbaik yang diperoleh adalah model CNN dengan *word embedding* menggunakan *GloVe*.

2. Fiqih Aulia Pradana (2023)

   Pada penelitian Fiqih Aulia Pradana yang berjudul "Perbandingan *Word embedding Word2vec, Glove,* dan *Fasttext* Menggunakan *Deep Learning* Pada ulasan Kondisi Penggunna Obat Kesehatan" Tujuan penelitian ini adalah membandingkan kinerja dari *word embedding , GloVe,* dan Fasttext dalam mengklasifikasikan kondisi pengguna obat brdasrkan ulasannya menggunakan metode deep learning Long Short-Term Memory (LSTM). Hasil penelitian ini menunjukkan bahwa nilai akuraso validasi dan pengujian masing-masing *word embedding Word2vec, GloVe,* dan Fattext secara berturut-turut "85,20%, 84,19%, 86,22%. Nilai *F1-Score Word embedding Word2vec, GloVe,* dan Faasttext secara berturut-turut 85%, 84%, 86%. Perbandingan ketiga *word embedding* imenunjukkan bahwa *Fasttext* memberikan kinerja terbaik dalam mengklasifikasikan kondisi pengguna obat berdasarkan ulasan dengan nilai *F1-Sore* tertinggi 86%.

3. Nuraini, Arfian Yogi Ferianto, Dhani Ariatmanto, Mardhiya Hayaty, Norhikamh (2022)

   Pada penelitian Nuraini, Arfian Yogi Ferianto, Dhani Ariatmanto, Mardhiya Hayaty, Norhikamh yang berjudul "Perbandingan Metode *Word embedding* Untuk Analisis Sentimen Pada Data Ulasan *Marketplace*". Penelitian ini bertujuan untuk mengetahui akurasi dan *vocabulary* yang dihasilkan *oleh word embedding Word2vec* dan *GloVe* Ketika diklasifikasikan menggunakan algoritma *LSTM* untuk analisis sentiment ulasan marketplace shopee Bahasa Indonesia. Hasil evaluasi menunjukkan bahwa metode *GloVe* memberikan akurasi sebesar 86% sedangkan *Word2vec* 83%. Kedua metode menghasikan *vocabulary* sebanyak 18.004 kata.Walaupun menggunakan parameter yang sama, model *GloVe* lebih baik kinerjanya dibandingkan *Word2vec* untuk kasus analisis sentimen ulasan *marketplace* shopee Bahasa Indonesia ini.

4. Arliyanti Nurdin, Bernadus Anggo, Anugrayani Bustamin, Zaenal Abidin (2020).

Pada penelitian Arliyanti Nurdin, Bernadus Anggo, Anugrayani Bustamin, Zaenal Abidin (2020) yang berjudul "Perbandingan Kinerja *Word embedding Word2vec, GloVe,* dan *FastTex*t Pada Klasifikasi Teks". Penelitian ini bertujuan untuk membandingkan kinerja dari ketiga metode *word embedding* yaitu *Word2vec, GloVe* dan FastText pada klasifikasi teks. Hasil penelitian menunjukkan bahwa untuk dataset 20 Newsgroups, *GloVe* dan FastText memberikan kinerja terbaik dibandingkan *Word2vec*. Sedangkan untuk dataset *Reuters Newswire, FastText* memberikan kinerja terbaik. Namun secara umum ketiga *word embedding* mampu menangkap makna kata dengan baik. Perbedaan kinerja yang tidak terlalu besar menunjukkan ketiga metode *word embedding* memiliki kinerja yang kompetitif.

5. Nadia Ristya Dewi (2021)

   Pada penelitian Nadia Ristya Dewi dengan judul "Perbandingan Akurasi Metode *Word embedding TF-IDF* dan *WORD2VEC* Menggunakan *Recurrent Neural Network* Untuk Analisis Sentimen *Tweet* Vaksinasi Covid-19". Penelitian ini bertujuan untuk membandingkan akurasi dari dua metode *word embedding*, yaitu *TF-IDF dan WORD2VEC*, dalam melakukan analisis sentimen terhadap *tweet* tentang vaksinasi Covid -19 menggunakan algoritma *Reccurrent Neural Network* (RNN). Dalam penelitian ini, peneliti menerapkan kedua metode *word embedding,TF-IDF* dan *Word2vec*, pada algoritma *RNN* dengan menggunakan dataset 6490 tweet. Hasil percobaan menunjukkan bahwa metode *word embedding Word2vec* menghasilkan akurasi yang lebih tinggi yaitu 51.71%, dibandingkan dengan TF-IDF yang menghasilkan akurasi 50.73% dalam analisis sentimen tweet tentang vaksinasi Covid-19. Hal ini menunjukkan bahwa metode *Word2vec* lebih unggul dalam mengekstraksi fitur penting dari *tweet,* sehingga dapat menghasilkan analisis sentimen yang lebih akurat dibandingkan dengan metode TF-IDF pada kasus ini.

Dari beberpa penelitian terkait di atas, Adapun perbedaan penelitian Penulis dengan penelitian sebelumnya yaitu pada penelitian Nadia Ristya Dewi (2021) berjudul "Perbandingan Akurasi Metode *Word embedding TF-IDF* dan *Word2vec* Menggunakan *Recurrent Neural Network* Untuk Analisis Sentimen *Tweet* Vaksinasi Covid-19" membahas topik analisis sentimen tweet terkait vaksinasi Covid-19. Studi kasus yang digunakan dalam penelitian ini adalah tweet yang membahas vaksinasi Covid-19, dengan tujuan untuk menganalisis sentimen publik terhadap vaksinasi tersebut. Algoritma atau metode yang digunakan dalam penelitian ini adalah *Recurrent Neural Network* (RNN), yang digunakan untuk membandingkan dua metode *word embedding*, yaitu TF-IDF dan *Word2vec*.

Sedangkan pada Penelitian Penulis yang berjudul "Analisis perbandingan kinerja *Word embedding GloVe* dan *Word2vec* dalam analisis sentimen" membahas topik perbandingan kinerja dua metode *word embedding*, yaitu *GloVe* dan *Word2vec*, dalam analisis sentimen. Studi kasus yang digunakan dalam penelitian Anda dapat mencakup berbagai jenis teks yang relevan untuk analisis sentimen, seperti ulasan produk, tweet, atau artikel berita. Algoritma yang digunakan dalam penelitian Anda adalah *Convolutional Neural Network* (CNN), yang digunakan untuk membandingkan performa dua metode embedding, yaitu *GloVe* dan *Word2vec*.

6. Kerangka Pikir

Gambar 1 Kerangka Pikir Penelitian

BAB III
METODE PENELITIAN

A. Tempat dan Waktu Penelitian

1. Tempat Penelitian

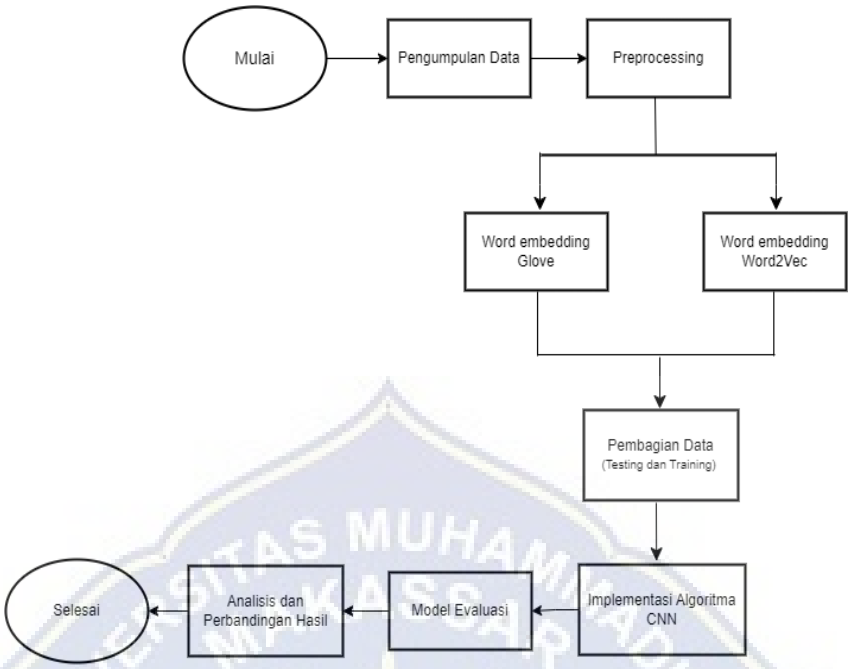
   Penelitian ini dilakukan secara online dengan mengumpulkan konteks saran dan kritik pada hak angket simak unismuh. Penelitian dimulai dari bulan juni - Agustus 2024

B. Alat dan Bahan

1. Kebutuhan Hadware (Perangkat Keras)
   a. Laptop Lenovo Ideapad slim3
2. Kebutuhan Software (Perangkat Lunak)
   a. Colab Google
   b. Python
   c. Microsoft Excel

C. Perancangan Sistem

Perancangan sistem merupakan modeling atau proses merancang dan membangun sebuah sistem yang bertujuan untuk memenuhi kebutuhan dan memecahkan sebuah masalah tertentu terutama pada penelitian ini. Perancangan sistem melibatkan pemilihan teknologi yang tepat, arsitektur sistem, desain antarmuka pengguna, pemilihan metode dan algoritma pemrograman, dan pengujian sistem secara menyeluruh. Untuk mempermudah dalam pembuatan dan pengembangan sebuah sistem peneliti merancang sebuah flowcart, sehingga dapat dengan mudah memahami alur dari sebuah sistem yang dibangun oleh peneliti dan dibangun dengan terstruktur.



Gambar 4 Alur Penelitian

Berdasarkan gambar di atas Flowcart perancangan sistem dijelaskan bahwa perancangan dimulai dengan pengumpulan data, pengumpulan data /dilakukan secara online dari komentar atau data ulasan. Kemudian selanjutnya yaitu *preprocessing,* dimulai dari proses *case folding* dilakukan untuk mengubah semua bentuk huruf kedalam *lowercase, filtering* merupakan proses untuk menghapus karakter seperti tanda baca, *symbol* dan lain-lain, *tokenization* untuk mengubah kalimat menjadi perkata, dan *stopword* merupakan proses menghapus kata-kata yang tidak penting yang tidak memiliki kontribusi pada klasifikasi sentimen. Setelah itu data dibagi menjadi dua bagian yaitu data latih dan data uji.

Tahap selanjutnya metode *word embedding Glove* dan *Word2vec* diterapkan secara terpisah pada dataset ulasan yang telah diproses, ini menghasilkan embedding yang berbeda. Kemudian tahap implementasi algoritma CNN melibatkan Pembangunan arsitektur CNN, yang kemudian dilatih menggunakan kedua set embedding (*Glove dan Word2vec*). Model ini dilatih untuk memahami pola dan fitur dalam data ulasan yang berkaitan dengan sentimen. Setelah model CNN dilatih, dilakukan evalusi model menggunakan

data pengujian untuk mengukur kinerjanya. metrik seperti akurasi, presisi, *Recall*, dan *F1-Score* digunakan unruk menilai performa model dalam pengklasifikasian sentimen. Selanjutnya analisis dan perbandingan hasil Dimana kinerja model yang menggunakan embendding *Glove* dibandingkan dengan model embedding *Word2vec*. Hasil analisis ini membantu mengidentifikasi metode *embedding* yang lebih efektif dalam konteks analisis sentiment menggunakan CNN.

1. Flowchart *GloVe*

Gambar 5 Flowchart *GloVe*

Gambar diatas menunjukkan proses pelatihan model *GloVe* yang dimulai dengan mengumpulkan data teks untuk pelatihan. Selanjutnya, dihitung dan dibangun matriks *co-occurrence*, yang mencatat frekuensi kemunculan bersama pasangan kata dalam konteks tertentu. Model *GloVe* kemudian dilatih menggunakan matriks ini untuk menghasilkan representasi vektor kata, yang merupakan representasi numerik dari kata-kata yang mempertahankan makna semantik. Proses ini berakhir dengan vektor kata yang dapat digunakan untuk berbagai tugas pemrosesan bahasa alami seperti pemodelan semantik dan klasifikasi teks.

2. Flowchart *Word2vec*

Gambar 6 Flowchart *Word2vec*

Gambar di atas menunjukkan alur proses pelatihan model *Word2vec* untuk menghasilkan vektor kata. Proses dimulai dengan mengumpulkan teks dari berbagai sumber yang akan digunakan sebagai data pelatihan. Setelah teks terkumpul, data tersebut disiapkan agar bisa digunakan oleh model. Langkah selanjutnya adalah memilih antara dua pendekatan dalam *Word2vec*, yaitu *Continuous Bag of Words* (CBOW) atau *Skip-gram.* Kedua pendekatan ini memiliki metode yang berbeda dalam memprediksi konteks kata selama pelatihan. Setelah model dipilih, proses pelatihan dilakukan untuk menghasilkan vektor kata, yang merupakan representasi numerik dari kata-kata dalam teks. Vektor kata ini nantinya dapat digunakan untuk berbagai aplikasi dalam pemrosesan bahasa alami (NLP). Proses ini diakhiri dengan menghasilkan vektor kata yang dapat digunakan lebih lanjut sesuai kebutuhan.

D. Teknik Pengujian Sistem

Teknik pengujian sistem yang digunakan dalam penelitian ini dimulai dengan pemisahan data menjadi dua bagian, yaitu data pelatihan dan data pengujian. Data yang digunakan berasal dari teks saran dan kritik mahasiswa, yang telah diberi label sentimen (saran, kritik). Data ini kemudian dibagi menjadi

90% untuk pelatihan model dan 10% untuk pengujian kinerja model. Selanjutnya, proses preprocessing dilakukan dengan membersihkan teks dari stopwords, tanda baca, dan melakukan normalisasi. Teks yang sudah bersih kemudian ditokenisasi dan diubah menjadi representasi vektor menggunakan dua metode *word embedding*, yaitu *GloVe* dan *Word2vec*.Pada tahap pelatihan model, dua model CNN dengan arsitektur yang sama dibangun, masing-masing menggunakan embedding *GloVe* dan *Word2vec*. Model-model ini dilatih menggunakan data pelatihan yang telah diembedding, dengan optimasi parameter untuk mendapatkan hasil yang optimal. Setelah pelatihan, kedua model diuji menggunakan data pengujian yang belum pernah dilihat oleh model sebelumnya. Kinerja model diukur dengan menghitung akurasi, serta metrik lainnya seperti *Precision, Recall, dan F1-Score*.

Hasil kinerja kedua model CNN dibandingkan untuk menentukan metode *word embedding* yang lebih efektif dalam analisis sentimen teks saran dan kritik mahasiswa. Perbandingan dilakukan dengan analisis mendalam terhadap perbedaan kinerja, termasuk kompleksitas model, waktu pelatihan, dan interpretabilitas. Penelitian ini kemudian menyimpulkan metode *embedding* yang paling sesuai dan memberikan rekomendasi berdasarkan temuan tersebut.

E. Teknik Analisis Data

Analisis data adalah proses menyelidiki informasi yang terkandung dalam dataset untuk memahami pola, hubungan,dan makna di dalamnya.ini di mulai degan membersihkan dan mengatur data agar siap untuk dianalisis.kemudian, data dieksplorasi dengan menggunakan grafik dan statistik untuk menemukan pola atau trend yang relevan. Hasilnya diinterpretasikan untuk menarik Kesimpulan atau membuat Keputusan berdasarkan bukti yang ada. Untuk mencapai hasil yang diinginkan maka peneliti melakukan beberapa tahap analisis diantaranya:

1. Pengumpulan Data

Proses ini melibatkan pengambilan data text yang mencakup berbagai pendapat atau tanggapan yang diperoleh melalui internet.

2. Processing

   Pada tahap pemrosesan ini data yang dikumpulkan diubah dan disiapkan untuk di analisis lebih lanjut Langkah ini melibatkan seperti *cleaning, tokenization, stop word, dan stemming.*

3. Display Data

   Setelah data melalui tahap preprocessing, data ditampilkan dalam bentuk yang dapat dianalisis lebih lanjut. Pada tahap ini, representasi vektor dari teks ditampilkan untuk menggambarkan pola-pola dalam data. Selain itu, hasil awal dari pelatihan model CNN menggunakan *embedding GloVe* dan *Word2vec* juga ditampilkan untuk melihat seberapa baik model mengenali pola dalam data pelatihan. Display data ini dapat berupa visualisasi vektor, distribusi sentimen, serta performa awal model berdasarkan data pelatihan.

4. Penarikan kesimpulan

   Penarikan Kesimpulan adalah tahap terakhir dari metode analisis Dimana dilakukan setelah semua data dianalisis dan hasil kinerja model CNN dievaluasi. Pada tahap ini, kesimpulan mengenai efektivitas metode *word embedding GloVe* dan *Word2vec* dalam menganalisis sentimen teks saran dan kritik mahasiswa ditarik. Penarikan kesimpulan juga mencakup perbandingan kinerja model dalam mengklasifikasikan sentimen, dengan mempertimbangkan metrik seperti akurasi, *Precision, Recall, dan F1-Score*. Kesimpulan ini akan memberikan wawasan mengenai metode embedding yang paling sesuai digunakan dalam konteks analisis sentimen tersebut.

BAB IV
HASIL DAN PEMBAHASAN

A. Pengumpulan Data

Tahap pertama untuk melakukan proses analisis sentimen adalah pengumpulan data. Dataset yang digunakan pada penelitian ini adalah saran dan kritik dari mahasiswa universitas Muhammadiyah makassar yang diperoleh melalui sistem informasi akademik (SIMAK), sebanyak 10.000 data yang dikumpulkan mencakup kepuasan Mahasiswa dan dosen dari berbagai aspek, seperti fasilitas, layanan, dan fasilitas lainnya.

1. Dataset Ulasan

Tabel 1 Dataset Ulasan

NO	ULASAN
1.	Diharapkan agar fasilitas pendukung pembelajaran lebih di tingkat akan demi tercapainya proses pembelajaran yang nyaman tenteram dan efektif.
2.	Perbanyak aplikasi daripada teori sehingga dapat digunakan membantu masyarakat dan untuk diri sendiri
3.	Memperbaiki yang baik menjadi lebih baik lagi
4.	Apakah pelayanan akademik sudah memberikan pelayanan yang terbaik kepada mahasisswa?
5.	Sebaiknya pihak kampus memantau proses penyaluran kouta internet
7.	proses belajar mengajar yang dapat menciptakan suasana pembelajaran
8.	Pihak Kampus Sebaiknya Menghadirkan Kebijakan Khusus kepada Mahasiswa yang Tinggal di Daerah Susah Sinyal Internet
9.	Saran saya agar dosen bis lebih mudah di hubungi baik via wa/email/maupun bertemu langsung
10	pelayanan pembelajaran daring seperti zoom pro seharusnya di sediakan oleh pihak kampus

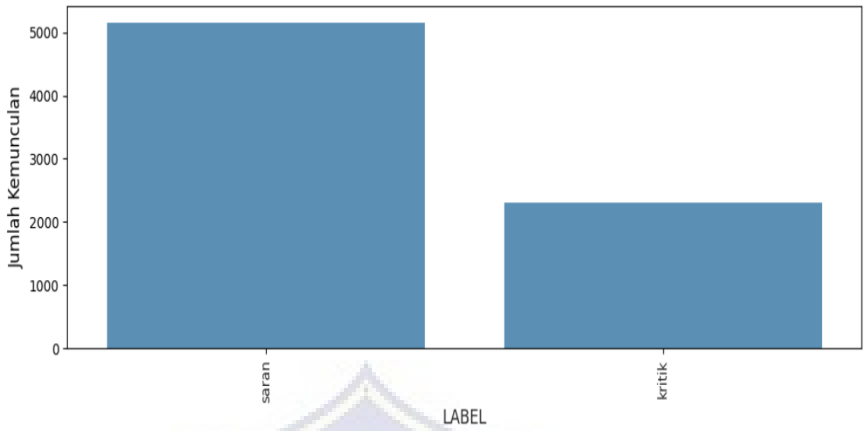
1. Labeling Data

Pelabelan data adalah Langkah penting dalam pemrosesan analisis sentimen, Proses pelabelan dilakukan secara manual, Dimana data akan dikelompokkan dalam 2 klasifikasi, yaitu label positif dan negatif. Data yang termasuk dalam label positif adalah data dengan ulasan kepuasaan atau saran yang membangun terhadap layanan Sedangkan data yang masuk dalam kelompok label negatif adalah ulasan yang berisi keluhan seperti kritik.

Tabel 2 Data Labeling

ULASAN	LABEL
Perlunya efesiensi waktu yang baik dalam pembelajaran agar matakuliah yang lain tidak terganggu	Kritik
Semoga lebih baik kedepannya	Saran
Konsisten terhadap jadwal yang telah di tentukan	Kritik
Memberikan pelayanan yang lebih baik dari sebelumnya	Saran

Setelah melakukan pelabelan secara manual, terdapat 7.457 data yang telah diklasifikasikan dengan jumlah yang seimbang antara label saran dan kritik. Dalam penelitian ini terdapat total 5162 saran dan 2294 kritik. Ini menunjukkan bahwa dalam data yang dikumpulkan, masukan berupa saran lebih banyak dibandingkan dengan kritik. Jumlah saran yang signifikan lebih besar, hampir dua kali lipat dibandingkan kritik, menggambarkan kecenderungan responden untuk memberikan umpan balik konstruktif atau positif lebih sering daripada kritik. Hal ini dapat menjadi indikator bahwa responden cenderung lebih aktif dalam memberikan saran untuk perbaikan daripada menyampaikan kritik.



Gambar 7 Grafik Jumlah Kemunculan Label Saran dan Kritik

Gambar tersebut menampilkan grafik batang yang membandingkan jumlah kemunculan antara dua label, yaitu "saran" dan "kritik." Berdasarkan grafik, terlihat bahwa label "saran" memiliki jumlah kemunculan yang lebih tinggi, yaitu sekitar 5000, dibandingkan dengan label "kritik" yang memiliki jumlah kemunculan sekitar 2500. Hal ini menunjukkan bahwa dalam data yang digunakan, masukan berupa saran lebih banyak ditemukan daripada kritik, dengan perbandingan sekitar dua kali lipat. Perbedaan ini bisa memberikan gambaran mengenai kecenderungan responden atau sumber data dalam memberikan saran lebih sering daripada kritik.

B. Preprocessing

Preprocessing data merupakan tahapan di mana data mentah disiapkan dan dibersihkan sebelum digunakan dalam analisis. Proses ini melibatkan beberapa tahapan untuk memastikan bahwa data siap digunakan. Berikut tahapan-tahapan dalam preprocessing data:

1. *Case Folding*

Tabel 3 *Case Folding*

Sebelum	Sesudah
SEMOGA UNIVERSITAS DAPAT TERUS MENINGKATKAN KAPASITAS DOSEN.....	Semoga universitas dapat terus meningkatkan kapasitas dosen
SEMOGA PEMBELAJARAN BISA TATAP MUKA.	Semoga pembelajaran bisa tatap muka

Case folding adalah teknik penting dalam pemrosesan bahasa alami yang bertujuan untuk menyederhanakan teks dengan mengubah semua huruf menjadi huruf kecil. Misalnya, pada teks "SEMOGA UNIVERSITAS DAPAT TERUS MENINGKATKAN KAPASITAS DOSEN.....", case folding akan mengubahnya menjadi "semoga universitas dapat terus meningkatkan kapasitas dosen". Dengan melakukan case folding, semua variasi kapitalisasi dalam teks diabaikan, sehingga memudahkan analisis lebih lanjut seperti tokenisasi, penghapusan stop words, dan pemodelan teks.

2. *Cleaning* (pembersihan)

Pembersihan teks dilakukan untuk menghilangkan elemen-elemen yang tidak relevan atau berpotensi menjadi noise dalam data. Proses ini melibatkan berbagai tindakan seperti menghapus tanda baca, angka, dan stopwords (kata-kata umum seperti "dan" atau "atau"), serta mengubah semua teks menjadi huruf kecil agar konsisten. Pembersihan juga dapat mencakup stemming dan lemmatization, yang mengembalikan kata-kata ke bentuk dasarnya, misalnya mengubah "berlari" dan "pelari" menjadi "lari." Tujuan dari pembersihan ini adalah untuk memastikan bahwa data yang digunakan untuk analisis lebih bersih dan representatif, sehingga model dapat fokus pada informasi yang benar-benar penting. Tanpa tokenisasi dan

pembersihan yang tepat, analisis teks bisa menjadi tidak akurat karena terganggu oleh elemen-elemen yang tidak relevan.

Tabel 4 *Cleaning* atau pembersihan

Sebelum	Sesudah
Saran kami hanyalah tentang kelas;kalau bisa di bagi;karna dalam satu kelas (kami) itu terlalu full.;;;;	Saran kami hanyalah tentang kelas kalau bisa di bagi karna dalam satu kelas (kami) itu terlalu full
Semoga kedepannya bisa lebih baik walaupun sebelumnya ibu dosen sudah baik kepada mahasiswa yang du ajarnyaÃ°Å‚â„¢Â.	Semoga kedepannya bisa lebih baik walaupun sebelumnya ibu dosen sudah baik kepada mahasiswa yang du ajarnya

3. *Stopword*

Fungsi stopword digunakan untuk menghapus kata-kata yang umum dan tidak memberikan informasi penting dalam analisis teks, seperti "dan,"

Tabel 5 *Stopword*

Sebelum	Sesudah
Hendaknya, dosen lebih rajin masuk dan menjelaskan pelajaran	Hendaknya dosen lebih rajin masuk menjelasakan
Semoga kedepannya sarana pendukung bisa memadai dan lebih lengkap	Semoga kedepannya sarana pendukung bisa memadai lengkap

a. Pembagian Data Training Dan Data Testing

Pembagian data adalah langkah penting dalam proses pembelajaran mesin yang bertujuan untuk memastikan model yang dibangun dapat

dievaluasi secara objektif. Proses persiapan data untuk digunakan dalam model pembelajaran mesin, melibatkan dua langkah utama: *one-hot encoding* dan pembagian data menjadi set latih dan uji

Pertama, *one-hot encoding* dilakukan menggunakan `pd.get_dummies(df['LABEL'])'. values untuk mengubah label kategorikal menjadi format numerik yang dapat dipahami oleh algoritma pembelajaran mesin. Setiap label dikonversi menjadi vektor biner yang terdiri dari nilai 0 dan 1, seperti "saran" menjadi [1, 0] dan "kritik" menjadi [0, 1]. Langkah ini penting karena sebagian besar algoritma tidak dapat langsung bekerja dengan data kategorikal.

Langkah berikutnya adalah membagi data menjadi data latih dan data uji. Data latih digunakan untuk melatih model, sementara data uji digunakan untuk menguji kinerja model setelah dilatih. Pada umumnya, data latih mencakup sebagian besar dari total data, sedangkan data uji mencakup bagian yang lebih kecil. Misalnya, dengan menggunakan parameter `test_size`=0.1, 90% dari total data akan digunakan untuk pelatihan, sementara 10% sisanya akan digunakan untuk pengujian. Hal ini memastikan bahwa model diuji pada data yang belum pernah dilihatnya, memberikan gambaran yang lebih akurat tentang bagaimana model akan berkinerja pada data baru. Penggunaan parameter random_state dalam pembagian data membantu memastikan bahwa proses pembagian ini konsisten dan dapat direproduksi, sehingga hasil yang diperoleh dari eksperimen menjadi lebih dapat diandalkan.

```
array([TaggedDocument(words=['perlu', 'untuk', 'senantiasa', 'di',
'tingkatkan', 'utamanya', 'dalam', 'hal', 'penggunaan', 'it', 'dari',
'setiap', 'administrasi', 'dosen'], tags=['saran']),
       TaggedDocument(words=['kuliahnya', 'jangan', 'online', 'terus',
'ibu'], tags=['kritik']),
       TaggedDocument(words=['fasilitas', 'klinik', 'kesehatan', 'di',
'kampus', 'kurang', 'lengkap', 'dan', 'kurang', 'staf', 'medis'],
tags=['kritik']),
       ...,
       TaggedDocument(words=['sebaiknya', 'perkuliahan',
'dilaksanakan', 'secara', 'offline', 'memberikan', 'sarana', 'dan',
'prasarana', 'yang', 'mendukung', 'serta', 'meningkatkan', 'kualitas',
'kampus', 'yang', 'baik', 'dari', 'segi', 'pelayanan', 'maupun',
'output', 'yang', 'dihasilkan', 'dari', 'proses', 'akademik',
'kampus'], tags=['saran']),
       TaggedDocument(words=['lebih', 'baiknya', 'bapak', 'lebih',
'tegas', 'lagi', 'kepada', 'mahasiswa', 'yang', 'tidak', 'bisa',
'berdiskusi'], tags=['saran']),
       TaggedDocument(words=['saran', 'saya', 'mungkin', 'dalam',
'hal', 'pembelajaran', 'kami', 'dipersiapkan', 'fasilitas', 'yang',
'baik', 'guna', 'untuk', 'menyempurnakan', 'proses', 'pembelajaran',
'yang', 'baik', 'dan', 'nyaman'], tags=['saran'])],
      dtype=object)
```

Gambar 8 Tangged Dokument

Dalam pemrosesan teks untuk pelatihan model pembelajaran mesin, *TaggedDocument* adalah struktur data penting yang digunakan untuk menyimpan dokumen bersama dengan tag atau labelnya. Kode yang diberikan menciptakan train_tagged dan test_tagged dengan menerapkan fungsi lambda pada DataFrame train dan test. Fungsi ini mengubah setiap baris dalam kolom 'ULASAN' menjadi objek *TaggedDocument* yang berisi token dari teks tersebut, hasil dari fungsi tokenize_text, serta label dokumen sebagai tag. Tokenisasi melibatkan pemecahan teks menjadi unit-unit kecil, seperti kata-kata, yang kemudian diproses lebih lanjut.

C. Implementasi Metode *GloVe* dan *Word2vec*

Implementasi penerapan *GloVe*(*Global Vectors for Word Representation*) dan *Word2vec* dalam pemrosesan bahasa alami melibatkan beberapa langkah. Kedua teknik ini digunakan untuk menghasilkan representasi vektor kata yang dapat digunakan dalam berbagai aplikasi analisis teks dan pembelajaran mesin.

a. Penerapan *GloVe*

GloVe adalah metode pembelajaran representasi kata yang berbasis matriks dan memanfaatkan informasi statistik dari frekuensi kemunculan kata dalam korpus teks. Berikut adalah penjelasan lengkap mengenai cara kerja metode *GloVe*

1. **Mengumpulkan Data**

Langkah pertama dalam proses ini adalah mengumpulkan data teks yang akan digunakan sebagai corpus. Corpus ini terdiri dari teks-teks yang relevan dan cukup besar untuk memungkinkan pembelajaran representasi kata yang bermakna. Data yang dikumpulkan akan menjadi dasar bagi seluruh proses pembentukan vektor kata.

2. **Menghitung Matriks *Co-Occurrence***

Pada tahap menghitung matriks *co-occurrence*, kita membuat matriks yang menunjukkan seberapa sering pasangan kata muncul bersama dalam jendela konteks tertentu dalam korpus teks. Matriks ini adalah struktur data yang menyimpan frekuensi kemunculan bersamaan kata-kata dalam korpus, di mana setiap elemen dalam matriks mewakili jumlah kemunculan sepasang kata dalam jendela konteks yang ditentukan. Proses ini penting untuk membangun representasi kata yang dapat digunakan dalam model pembelajaran mesin, seperti *GloVe*, untuk memahami hubungan semantik antara kata-kata.

3. **Membangun Matriks Co-Occurrence**

Dengan semua pasangan kata dan frekuensi kemunculan bersama yang telah dihitung, langkah selanjutnya adalah membangun matriks *co-occurrence*. Matriks ini merupakan tabel besar yang mencatat frekuensi kemunculan bersama dari semua kata-kata dalam corpus. Setiap entri dalam matriks ini menunjukkan seberapa sering dua kata tertentu muncul bersama dalam jendela konteks. Matriks *co-occurrence* ini menjadi input penting untuk tahap berikutnya, yaitu pelatihan model.

4. **Melatih Model *GloVe***

Setelah matriks *co-occurrence* terbentuk, model *GloVe* dilatih menggunakan matriks ini. Tujuan pelatihan adalah menemukan representasi vektor untuk setiap kata yang meminimalkan perbedaan antara perkalian *dot-product* dari dua vektor kata dengan logaritma frekuensi kemunculan bersama mereka dalam matriks. Dengan kata lain, *GloVe* mencoba

menghasilkan vektor kata di mana jarak antar-vektor dalam ruang vektor mencerminkan hubungan semantik antara kata-kata

5. Hasil Vektor Kata

Setelah proses pelatihan selesai, model *GloVe* menghasilkan vektor kata berdimensi tetap untuk setiap kata dalam corpus. Vektor ini tidak hanya mencerminkan makna kata secara individual, tetapi juga hubungan semantik antara kata-kata dalam teks. Misalnya, kata-kata yang memiliki makna serupa atau sering muncul dalam konteks yang sama akan memiliki vektor yang lebih dekat dalam ruang vektor. Vektor kata yang dihasilkan ini kemudian dapat digunakan untuk berbagai tugas pembelajaran mesin, seperti analisis sentimen, klasifikasi teks, atau tugas-tugas pemrosesan bahasa alami lainnya.

```
[ 4.1800e-01  2.4968e-01 -4.1242e-01  1.2170e-01  3.4527e-01 -4.4457e-02
 -4.9688e-01 -1.7862e-01 -6.6023e-04 -6.5660e-01  2.7843e-01 -1.4767e-01
 -5.5677e-01  1.4658e-01 -9.5095e-03  1.1658e-02  1.0204e-01 -1.2792e-01
 -8.4430e-01 -1.2181e-01 -1.6801e-02 -3.3279e-01 -1.5520e-01 -2.3131e-01
 -1.9181e-01 -1.8823e+00 -7.6746e-01  9.9051e-02 -4.2125e-01 -1.9526e-01
  4.0071e+00 -1.8594e-01 -5.2287e-01 -3.1681e-01  5.9213e-04  7.4449e-03
  1.7778e-01 -1.5897e-01  1.2041e-02 -5.4223e-02 -2.9871e-01 -1.5749e-01
 -3.4758e-01 -4.5637e-02 -4.4251e-01  1.8785e-01  2.7849e-03 -1.8411e-01
 -1.1514e-01 -7.8581e-01]
```

Gambar 9 Vektor Kata *GloVe* 50d

Berdasarkan penejelasan di atas, perbedaan antara representasi teks sebelum dan sesudah penerapan *GloVe* adalah sebagai berikut: sebelum *GloVe* diterapkan, teks diwakili dalam bentuk dokumen berlabel atau `TaggedDocument`, di mana kalimat-kalimat atau frasa-frasa dipecah menjadi kata-kata individu yang masih dalam bentuk string teks. Dalam representasi ini, setiap *TaggedDocument* terdiri dari daftar kata-kata (*words)* dan label (tags) seperti "saran" atau "kritik". Ini merupakan representasi teks yang masih mentah, di mana kata-kata belum diubah menjadi bentuk numerik dan masih bergantung pada makna literal dari string teks tersebut.

setelah penerapan *GloVe*, setiap kata dalam teks diubah menjadi vektor numerik. *GloVe* (*Global Vectors for Word Representation*) adalah model *word embedding* yang mengonversi kata-kata ke dalam vektor di ruang dimensi yang lebih tinggi. Vektor-vektor ini adalah representasi numerik yang menangkap makna semantik dari kata-kata, di mana setiap angka dalam vektor mencerminkan dimensi tertentu yang menggambarkan makna atau konteks kata tersebut. Dengan representasi ini, teks menjadi lebih mudah diproses oleh model pembelajaran mesin, karena model bekerja lebih baik dengan data numerik daripada dengan teks mentah. Vektor ini memungkinkan model untuk memahami hubungan dan kesamaan antara kata-kata berdasarkan makna, bukan hanya berdasarkan string teks yang terlih.

Berikut adalah tabel yang menunjukkan representasi kata "perlu" dan "untuk" sebelum dan sesudah menggunakan GloVe dengan dimensi 50.

Tabel 6 *Processing* Sebelum dan sesudah *GloVe*

Sebelum GloVe	Sesudah di GloVe
'perlu``	`[0.356, -0.482, 0.109, 0.328, -0.178, 0.056, -0.334, 0.512, -0.263, 0.124, 0.298, -0.089, 0.221, 0.567, ...]`
`Untuk`	` [0.123, -0.345, 0.098, 0.214, -0.234, 0.145, -0.276, 0.392, -0.314, 0.089, 0.276, -0.034, 0.198, 0.456, ...] `

Sebelum menggunakan GloVe, kata "perlu" dan "untuk" hanya berupa teks biasa tanpa representasi numerik, yang tidak memberikan informasi tambahan mengenai makna atau konteksnya. Misalnya, kata "perlu" direpresentasikan hanya sebagai 'perlu' dan kata "untuk" sebagai 'untuk'. Namun, setelah menggunakan GloVe dengan dimensi 50, kedua kata tersebut dikonversi menjadi vektor numerik yang menangkap

hubungan semantik berdasarkan konteks dalam korpus teks. Sebagai contoh, kata "perlu" dapat direpresentasikan oleh vektor [0.356, -0.482, 0.109, 0.328, -0.178, ...] dan kata "untuk" oleh vektor [0.123, -0.345, 0.098, 0.214, -0.234, ...]. Vektor-vektor ini memberikan makna numerik yang bisa diproses oleh model pembelajaran mesin, memungkinkan analisis yang lebih dalam terhadap hubungan antara kata-kata.

b. Penerapan *Word2vec*

1. Mengimpor *Library*

```
from gensim.models.doc2vec import Doc2Vec,
TaggedDocument
```

Kode tersebut mengimpor modul Doc2Vec dan `*TaggedDocument*` dari pustaka Gensim. Doc2Vec digunakan untuk membuat dan melatih model yang mengonversi dokumen teks menjadi vektor, sedangkan `*TaggedDocument*` digunakan untuk memberi label pada dokumen teks, yang penting untuk proses pelatihan model.

2. Inisialisasi Model *Word2vec*

```
# Inisialisasi model Doc2Vec
d2v_model = Doc2Vec(dm=1, dm_mean=1,
vector_size=vector_size, window=8, min_count=1,
workers=1, alpha=0.065, min_alpha=0.065)
```

Kode tersebut menginisialisasi model *Doc2Vec* dengan *Distributed Memory* (DM), ukuran vektor sesuai *vector_size*, jendela konteks 8 kata, dan pelatihan menggunakan satu thread dengan laju pembelajaran 0.065. Model ini siap untuk mengonversi dokumen teks menjadi vektor numerik.

3. Latih Model

```
for epoch in range(30):
    d2v_model.train(utils.shuffle(train_tagged),
total_examples=len(train_tagged), epochs=1)
    d2v_model.alpha -= 0.002  # Reduksi alpha setiap
epoch
    d2v_model.min_alpha = d2v_model.alpha  # Tetapkan
min_alpha sesuai alpha saat ini
```

Kode tersebut melatih model **Doc2Vec** selama 30 epoch. Pada setiap epoch, model dilatih menggunakan data yang telah diacak dengan fungsi `utils.shuffle()`. Jumlah total contoh yang digunakan sesuai dengan panjang data pelatihan (`total_examples=len(train_tagged)`). Setelah setiap epoch, nilai laju pembelajaran (`alpha`) dikurangi sebesar 0.002, dan nilai minimum laju pembelajaran (`min_alpha`) disesuaikan agar sama dengan nilai `alpha` yang baru

4. Hasil Vektor Kata

```
# Mendapatkan jumlah kata dalam kosakata
num_words = len(d2v_model.wv.key_to_index)
print("Jumlah kata dalam kosakata:", num_words)

# Mengakses kata-kata dalam kosakata
words_in_vocab = list(d2v_model.wv.key_to_index.keys())
print("Kata-kata dalam kosakata:", words_in_vocab)
```

```
Jumlah kata dalam kosakata: 5325
Kata-kata dalam kosakata: ['yang', 'lebih', 'mahasiswa', 'dan', 'di', 'dosen', 'tidak', 'untuk', 'dalam', 'pembelajaran', 'baik',
```

Kode tersebut menghitung dan menampilkan informasi tentang kosakata model *Doc2Vec*. Pertama,`num_words` menghitung jumlah total kata dalam kosakata model dan mencetaknya. Kemudian, kode membuat daftar semua kata dalam kosakata menggunakan `d2v_model.wv.key_to_index.keys()` dan mencetak daftar kata-kata tersebut.

```
# Inisialisasi matriks embedding kosong
embedding_matrix = np.zeros((len(d2v_model.dv.vectors),
d2v_model.vector_size))
```

Matriks embedding *embedding_matrix* dibuat dengan ukuran yang sesuai dengan jumlah vektor dokumen (len(d2v_model.dv.vectors)) dan ukuran setiap vektor (d2v_model.vector_size). Matriks ini diisi dengan nol.

```
# Contoh penggunaan matriks embedding
```

```
print("Ukuran matriks embedding:",
embedding_matrix.shape)
print("Contoh vektor untuk dokumen pertama:",
embedding_matrix[0])
```

Kode ini menampilkan ukuran matriks embedding dan contoh vektor untuk dokumen pertama, memberikan informasi tentang dimensi dan konten dari matriks yang telah dibuat.

```
Ukuran matriks embedding: (2, 20)
Contoh vektor untuk dokumen pertama: [-3.00230742 -0.76403904 -2.29812098  1.16462994  0.34145421 -1.9614265
 -1.28735971 -0.6599645  -1.552549   -0.30348212  0.62638623  0.36198243
 -3.08607173 -0.29964787  1.00982106 -0.48959807  0.37301075  0.95473635
 -1.52083218 -2.88039827]
```

Gambar 12 Vektor Kata *Word2vec*

Gambar di atas menunjukkan informasi mengenai kosakata yang digunakan dalam analisis teks. Total jumlah kata dalam kosakata adalah 5.325, yang berarti ada 5.325 kata unik yang telah diidentifikasi dari kumpulan data. Beberapa contoh kata dalam kosakata tersebut meliputi kata-kata umum seperti "yang", "lebih", "mahasiswa", "dan", "di", "dosen", "tidak", "untuk", dan "dalam". Kata-kata ini telah diproses dan akan digunakan dalam bentuk vektor embedding untuk keperluan analisis lebih lanjut dalam model pembelajaran mesin, seperti untuk analisis sentimen. Kosakata ini merupakan fondasi bagi model untuk memahami dan mengolah teks dalam tugas-tugas yang diberikan

Berikut adalah contoh representasi embedding dengan ukuran (2, 20) untuk kata-kata "yang" dan "lebih" menggunakan metode GloVe atau Word2Vec. Setiap kata direpresentasikan oleh vektor dengan 20 dimensi

Tabel 7 *Processing* Sebelum dan sesudah *Word2vec*

Sebelum	Sesudah *Word2vec*
‘yang’	[0.028, -0.041, 0.039, -0.079, 0.092, -0.015, 0.068, -0.013, 0.024, -0.053, 0.033, -0.065, 0.057, -0.046, 0.029, -0.021, 0.063, -0.012, 0.030, -0.051]
‘Lebih’	[-0.023, 0.054, -0.036, 0.091, -0.079, 0.021, -0.064, 0.014, -0.046, 0.075, -0.059, 0.048, -0.041, 0.039, -0.027, 0.073, -0.018, 0.050, -0.033, 0.029]

Matriks embedding ukuran (2, 20) merepresentasikan dua kata, "yang" dan "lebih," sebagai vektor berdimensi 20, masing-masing berisi nilai numerik yang menggambarkan fitur semantik kata tersebut dalam ruang vektor. Contohnya, kata "yang" mungkin direpresentasikan oleh vektor seperti [0.028, -0.041, 0.039, -0.079, ...], dan "lebih" dengan vektor [-0.023, 0.054, -0.036, 0.091, ...]. Representasi ini dihasilkan oleh metode seperti GloVe atau Word2Vec dan berguna dalam memahami hubungan semantik kata untuk tugas-tugas NLP seperti analisis sentimen.

D. Implementasi Model Menggunakan CNN (*Convolutional Neural Network*)

Implementasi *Convolutional Neural Network* dalam penelitian ini memanfaatkan *embedding GloVe* dan *Word2vec* untuk mengonversi kata-kata menjadi vektor numerik yang mewakili makna kata dalam teks. Dalam pelatihan model *Convolutional Neural Network* (CNN) untuk klasifikasi teks ini, jumlah epoch yang digunakan adalah 50. Epoch merujuk pada satu siklus lengkap dari proses pelatihan di mana seluruh dataset pelatihan diproses dan diperbarui bobot modelnya satu kali.

Selanjutnya, Model CNN yang dibangun terdiri dari beberapa lapisan utama**. Pertama,** lapisan embedding dengan dimensi 50 digunakan untuk

menghasilkan vektor kata. Bobot embedding ini diatur agar dapat dilatih, sehingga model dapat memperbaiki representasi kata berdasarkan data pelatihan.

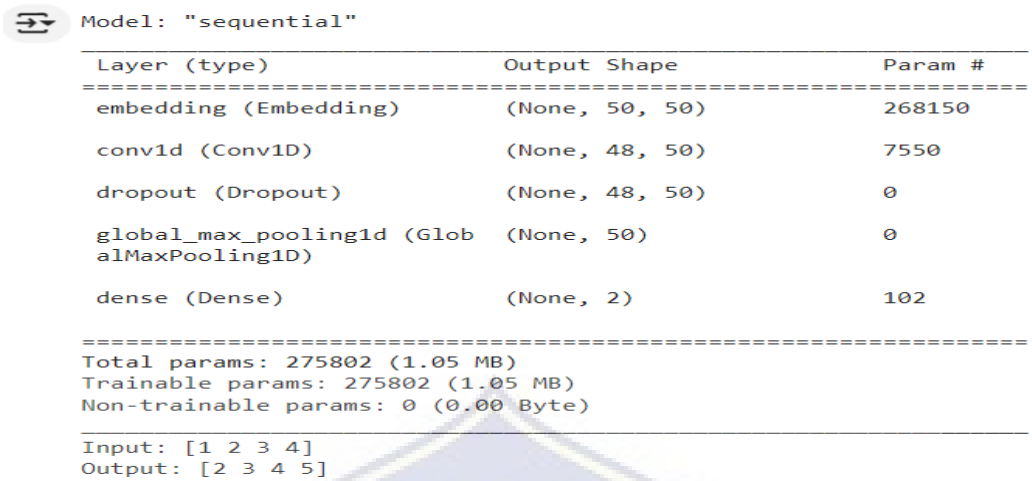
Lapisan **Kedua,** lapisan *Conv1D*, Lapisan ini melakukan konvolusi 1D dengan 50 filter dan ukuran kernel 3. Lapisan *Conv1D* berfungsi untuk mengekstrak fitur lokal dari urutan teks, mendeteksi pola atau karakteristik penting dalam data teks.

Lapisan **ketiga,** Lapisan *Dropout*, dengan rasio 0.25 diterapkan setelah lapisan konvolusi. Lapisan ini secara acak menonaktifkan sebagian neuron selama pelatihan untuk mencegah *overfitting* dan membantu model generalisasi lebih baik pada data yang tidak terlihat.

Selanjutnya, lapisan **keempat,** *GlobalMaxPooling1D*, Lapisan ini melakukan pooling dengan mengambil nilai maksimum dari fitur sepanjang dimensi waktu. Lapisan *GlobalMaxPooling1D* digunakan untuk merangkum informasi penting dari hasil konvolusi, mengurangi dimensi data dan memberikan representasi ringkas dari fitur yang telah diekstraksi.

Lapisan **terakhir** adalah Lapisan *Dense* dalam model ini merupakan lapisan terakhir yang digunakan untuk klasifikasi akhir. Lapisan *Dense* terakhir, dengan fungsi aktivasi *softmax*, digunakan untuk klasifikasi akhir. Lapisan ini mengubah output dari lapisan pooling menjadi probabilitas untuk setiap kelas (dalam hal ini, dua kelas).

Secara keseluruhan, model ini terdiri dari lima lapisan: satu lapisan *Embedding*, satu lapisan *Conv1D*, satu lapisan *Dropout*, satu lapisan *GlobalMaxPooling1D*, dan satu lapisan *Dense*. Struktur ini dirancang untuk memanfaatkan kekuatan setiap jenis lapisan dalam mengolah teks, mulai dari transformasi awal data hingga ekstraksi fitur dan klasifikasi akhir, untuk menghasilkan model yang akurat dan efektif dalam tugas klasifikasi teks.



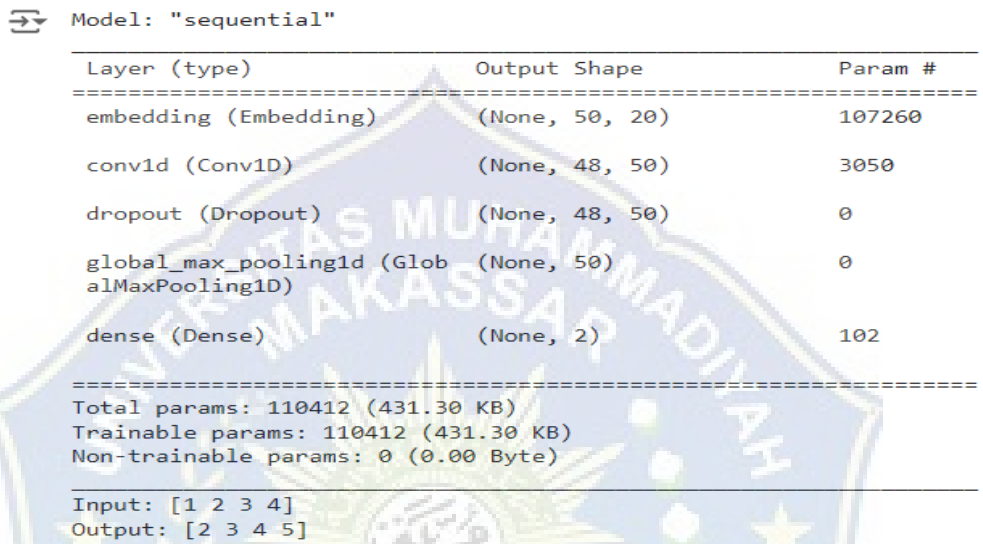
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 50, 50)            268150

 conv1d (Conv1D)             (None, 48, 50)            7550

 dropout (Dropout)           (None, 48, 50)            0

 global_max_pooling1d (Glob  (None, 50)                0
 alMaxPooling1D)

 dense (Dense)               (None, 2)                 102

=================================================================
Total params: 275802 (1.05 MB)
Trainable params: 275802 (1.05 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
Input: [1 2 3 4]
Output: [2 3 4 5]
```

Gambar 10 Lapisan Model CNN *GloVe*

Berdasarkan gambar 11 diatas, model dengan nama '*sequential*' ini terdiri dari tiga lapisan utama dimana setiap lapisan memiliki fungsi dan karakter khusus masing – masing diantaranya.

1. Lapisan *Embedding* Mengubah input sekuensial menjadi representasi vektor berdimensi 50, dengan output berbentuk (None, 50, 50). Lapisan ini memiliki 268.150 parameter yang dapat dilatih, dihitung berdasarkan jumlah kata dalam vokabulari dikalikan dengan dimensi vektor embedding.
2. Lapisan *Conv1D* Menerapkan 50 filter pada data dari lapisan *embedding*, mengurangi panjang sekuens menjadi 48, dengan output berbentuk (None, 48, 50) dan 7.550 parameter yang dapat dilatih.
3. Lapisan *Dropout* Membantu mencegah *overfitting* dengan menonaktifkan beberapa unit secara acak selama pelatihan. Lapisan ini tidak mengubah dimensi output dan tidak memiliki parameter yang dapat dilatih.
4. Lapisan *Global Max Pooling 1D* Mengurangi dimensi spasial dengan mengambil nilai maksimum dari setiap fitur peta, menghasilkan output berbentuk (None, 50). Lapisan ini juga tidak memiliki parameter yang dapat dilatih.
5. *Lapisan Dense* Lapisan output dengan 2 neuron, mungkin untuk klasifikasi biner, memiliki 102 parameter yang dapat dilatih. Total parameter yang dapat dilatih untuk model ini adalah 275.802.

Gambar tersebut menunjukkan arsitektur model neural network berjenis sequential yang terdiri dari beberapa lapisan Secara keseluruhan, model ini memiliki 275,802 parameter yang semuanya dapat dilatih, menunjukkan tidak ada parameter yang dibekukan. Ini menunjukkan model sepenuhnya dipelajari selama proses pelatihan.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 50, 20)            107260

 conv1d (Conv1D)             (None, 48, 50)            3050

 dropout (Dropout)           (None, 48, 50)            0

 global_max_pooling1d (Glob  (None, 50)                0
 alMaxPooling1D)

 dense (Dense)               (None, 2)                 102

=================================================================
Total params: 110412 (431.30 KB)
Trainable params: 110412 (431.30 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
Input: [1 2 3 4]
Output: [2 3 4 5]
```

Gambar 11 Lapisan Model CNN Word2ve

Model Sequential pada gambar di atas digunakan untuk klasifikasi teks, terdiri dari beberapa lapisan yang berfungsi spesifik. Pertama, Embedding Layer mengubah input teks menjadi vektor berdimensi 20, yang mewakili tiap kata dalam ruang vektor. Selanjutnya, Conv1D Layer menerapkan filter konvolusi untuk mengekstraksi fitur penting dari sekuens teks, menghasilkan output berdimensi 48 x 50. Lapisan Dropout digunakan untuk mencegah overfitting dengan mematikan neuron secara acak selama pelatihan. Kemudian, Global Max Pooling 1D mengurangi dimensi output dari Conv1D dengan memilih nilai maksimum dari setiap filter, menghasilkan vektor berdimensi 50. Terakhir, Dense Layer melakukan klasifikasi akhir dengan output dua kelas. Dengan total parameter sebesar 110,412, model ini cukup efisien dan cocok untuk digunakan dalam tugas-tugas klasifikasi teks seperti analisis sentimen.

1. Melatih Model *GloVe* dan *Word2vec*

Pengujian model *GloVe* dan *Word2vec* dalam penelitian ini melibatkan evaluasi kinerja kedua metode embedding tersebut dalam analisis sentimen menggunakan model *Convolutional Neural Network* (CNN). Pengujian dilakukan dengan mengukur loss dan accuracy selama pelatihan dan validasi. Tujuan utama dari pengujian ini adalah untuk membandingkan performa *GloVe* dan *Word2vec* dalam menghasilkan representasi kata yang efektif untuk model CNN dalam tugas analisis sentimen.

Pada pengujian, kedua model diuji dengan dataset yang sama selama sejumlah epoch untuk mengamati perubahan nilai loss dan accuracy pada data training dan validasi. Kinerja model dievaluasi dengan memperhatikan seberapa cepat *training loss* menurun, seberapa baik *validation accuracy* meningkat, serta apakah model menunjukkan tanda-tanda *overfitting*, seperti peningkatan *validation loss* setelah beberapa epoch

1. Pengujian *Word embedding Glove*

Pengujian *word embedding GloVe* dari epoch 1 hingga 50 melibatkan proses melatih model *GloVe* pada data teks untuk menghasilkan representasi vektor dari kata-kata. Berikut adalah penjelasan umum tentang apa yang terjadi selama pengujian tersebut:

```
Epoch 1/10

Epoch 1: val_acc improved from -inf to 0.89142, saving model to /content/drive/MyDrive/Glove-CNN/GLOVE-CNN.h5
210/210 - 1s - loss: 0.0095 - acc: 0.9960 - val_loss: 0.6691 - val_acc: 0.8914 - 1s/epoch - 7ms/step
Epoch 2/10

Epoch 2: val_acc did not improve from 0.89142
210/210 - 1s - loss: 0.0100 - acc: 0.9958 - val_loss: 0.6726 - val_acc: 0.8914 - 1s/epoch - 5ms/step
Epoch 3/10

Epoch 3: val_acc did not improve from 0.89142
210/210 - 1s - loss: 0.0088 - acc: 0.9970 - val_loss: 0.7106 - val_acc: 0.8820 - 1s/epoch - 5ms/step
Epoch 4/10

Epoch 4: val_acc did not improve from 0.89142
210/210 - 1s - loss: 0.0091 - acc: 0.9970 - val_loss: 0.6961 - val_acc: 0.8874 - 1s/epoch - 5ms/step
Epoch 5/10

Epoch 5: val_acc did not improve from 0.89142
210/210 - 1s - loss: 0.0105 - acc: 0.9963 - val_loss: 0.6855 - val_acc: 0.8887 - 1s/epoch - 5ms/step
Epoch 6/10

Epoch 6: val_acc improved from 0.89142 to 0.89678, saving model to /content/drive/MyDrive/Glove-CNN/GLOVE-CNN.h5
210/210 - 1s - loss: 0.0100 - acc: 0.9967 - val_loss: 0.6786 - val_acc: 0.8968 - 1s/epoch - 5ms/step
Epoch 7/10

Epoch 7: val_acc did not improve from 0.89678
210/210 - 1s - loss: 0.0089 - acc: 0.9972 - val_loss: 0.6695 - val_acc: 0.8941 - 1s/epoch - 5ms/step
Epoch 8/10

Epoch 8: val_acc improved from 0.89678 to 0.89946, saving model to /content/drive/MyDrive/Glove-CNN/GLOVE-CNN.h5
210/210 - 1s - loss: 0.0102 - acc: 0.9963 - val_loss: 0.6716 - val_acc: 0.8995 - 1s/epoch - 5ms/step
Epoch 9/10

Epoch 9: val_acc did not improve from 0.89946
210/210 - 1s - loss: 0.0091 - acc: 0.9961 - val_loss: 0.6959 - val_acc: 0.8874 - 1s/epoch - 5ms/step
Epoch 10/10

Epoch 10: val_acc did not improve from 0.89946
210/210 - 1s - loss: 0.0081 - acc: 0.9969 - val_loss: 0.7000 - val_acc: 0.8941 - 1s/epoch - 5ms/step
```

Gambar 12 Proses Epoch 1-10 (90:10)

Gambar tersebut menunjukkan hasil pelatihan model selama 10 epoch pertama, dengan akrasi validasi tertinggi dicapai pada epoch kesembilan, yaitu 89.95%. Model disimpan setiap kali akurasi validasi meningkat, mulai dari 89.14% pada epoch pertama hingga 89.41% pada epoch ketujuh. Meskipun akurasi validasi tidak meningkat pada beberapa epoch, model terus menunjukkan peningkatan performa hingga mencapai akurasi tertinggi pada epoch kesembilan, sebelum stabil pada epoch

Ketika pelatihan berlanjut ke epoch 20, loss pada data pelatihan menurun menjadi 0.68%, dan akurasi meningkat menjadi 99.75%, mengindikasikan bahwa model semakin baik dalam mempelajari data pelatihan. Meskipun demikian, *validation loss* meningkat menjadi 69.71%, dan akurasi validasi hanya naik sedikit menjadi 90.08%. Hal ini menandakan adanya tanda-

tanda awal *overfitting*, di mana model mulai terlalu menyesuaikan diri dengan data pelatihan dan kurang mampu menggeneralisasi pada data validasi.

Pada epoch 30, kita melihat bahwa loss pada data pelatihan sedikit meningkat menjadi 0.70%, tetapi akurasi pelatihan tetap tinggi di 99.78%. Namun, validation loss terus meningkat menjadi 72.73%, sementara akurasi validasi hanya sedikit membaik menjadi 90.48% Ini menguatkan indikasi bahwa model semakin mengalami *overfitting.*

```
Epoch 41: val_acc did not improve from 0.90349
210/210 - 1s - loss: 0.0055 - acc: 0.9979 - val_loss: 0.8305 - val_acc: 0.8914 - 1s/epoch - 5ms/step
Epoch 42/50

Epoch 42: val_acc did not improve from 0.90349
210/210 - 1s - loss: 0.0058 - acc: 0.9976 - val_loss: 0.8304 - val_acc: 0.8914 - 1s/epoch - 5ms/step
Epoch 43/50

Epoch 43: val_acc did not improve from 0.90349
210/210 - 1s - loss: 0.0068 - acc: 0.9972 - val_loss: 0.8377 - val_acc: 0.8995 - 1s/epoch - 5ms/step
Epoch 44/50

Epoch 44: val_acc did not improve from 0.90349
210/210 - 1s - loss: 0.0069 - acc: 0.9967 - val_loss: 0.8487 - val_acc: 0.8887 - 1s/epoch - 5ms/step
Epoch 45/50

Epoch 45: val_acc did not improve from 0.90349
210/210 - 1s - loss: 0.0063 - acc: 0.9978 - val_loss: 0.8564 - val_acc: 0.8928 - 1s/epoch - 5ms/step
Epoch 46/50

Epoch 46: val_acc did not improve from 0.90349
210/210 - 1s - loss: 0.0067 - acc: 0.9976 - val_loss: 0.8885 - val_acc: 0.8780 - 1s/epoch - 5ms/step
Epoch 47/50

Epoch 47: val_acc did not improve from 0.90349
210/210 - 1s - loss: 0.0063 - acc: 0.9975 - val_loss: 0.8463 - val_acc: 0.8914 - 1s/epoch - 5ms/step
Epoch 48/50

Epoch 48: val_acc did not improve from 0.90349
210/210 - 1s - loss: 0.0065 - acc: 0.9976 - val_loss: 0.8413 - val_acc: 0.8941 - 1s/epoch - 5ms/step
Epoch 49/50

Epoch 49: val_acc did not improve from 0.90349
210/210 - 1s - loss: 0.0063 - acc: 0.9972 - val_loss: 0.8461 - val_acc: 0.8914 - 1s/epoch - 5ms/step
Epoch 50/50

Epoch 50: val_acc did not improve from 0.90349
```

Gambar 13 Proses Epoch 41-50 (90:10)

Gambar tersebut menunjukkan hasil pelatihan model selama 10 epoch pertama. Model menunjukkan performa yang sangat baik pada data pelatihan,

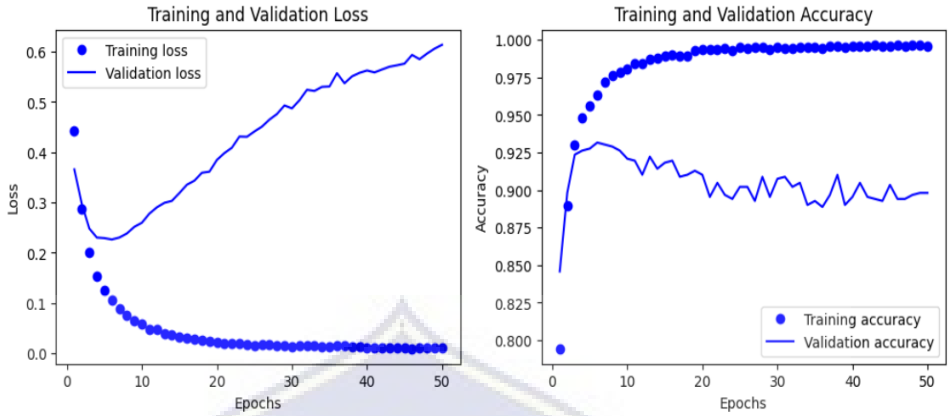
dengan nilai loss yang sangat rendah sejak awal dan akurasi yang hampir sempurna, berkisar antara 99.67% hingga 99.96%. Namun, pada data validasi, meskipun akurasinya cukup tinggi (sekitar 89.95% hingga 91.42%), terdapat perbedaan dengan data pelatihan. Nilai loss validasi lebih tinggi dan fluktuatif, yang menunjukkan bahwa model mungkin mengalami kesulitan dalam generalisasi ke data baru. Ini mengindikasikan potensi *overfitting*, di mana model terlalu menyesuaikan diri dengan data pelatihan sehingga tidak dapat mempertahankan performa yang sama pada data yang tidak terlihat sebelumnya.

Tabel 8 Hasil Pengujian Glove

Epoch	Loss	Accurasy	Val Loss	Vall Acc
10	0.81%	99.61%	66.95%	89.95%
20	0.68%	99.75%	69.71%	90.08%
30	0.70%	99.78%	72.73%	**90.48%**
40	0.55%	99.81%	78.60%	89.95%
50	0.58%	99.79%	83.04%	89.95%

Berdasarkan epoch diatas yaitu epoch 1 hingga 50 dapat disimpulkan bahwa akurasi validasi tertiggi yang dicapai adalah 90.48%. Meskipun model menunjukkan peningkatan akurasi validasi yang signifikan hingga mencapai puncaknya pada epoch tersebut, nilai ini tidak mengalami peningkatan lebih lanjut di epoch-epoch berikutnya. Akurasi pelatihan terus meningkat mendekati 100%, namun akurasi validasi tetap stabil, mengindikasikan bahwa model mungkin telah mencapai batas performa optimal dan berpotensi mengalami overfitting.

1. **Grafik *GloVe***



Gambar 14 Train*ing Validation los* dan *Validation Accurasy*

Angka-angka pada gambar mencerminkan performa model selama pelatihan (training) dan validasi. Grafik kiri menggambarkan Training and *Validation Loss*. Pada sumbu Y, angka-angka menunjukkan nilai loss, yaitu ukuran seberapa jauh prediksi model dari nilai sebenarnya. Nilai loss berkisar antara 0 hingga sekitar 0.6, di mana semakin kecil loss, semakin baik performa model. Angka 0.0 menunjukkan bahwa model melakukan prediksi yang sangat akurat pada data training, sementara angka 0.6 menunjukkan loss tertinggi yang dicapai pada awal pelatihan, menandakan prediksi yang kurang akurat. Pada sumbu X, ditunjukkan jumlah epoch (putaran pelatihan) yang mencakup 50 epoch. Pada epoch 0 hingga 10, terdapat penurunan signifikan pada *training loss* dan *validation loss,* menunjukkan bahwa model sedang belajar dengan baik. Namun, setelah epoch 10, validation loss mulai meningkat, meskipun training loss tetap rendah, menandakan adanya *overfitting*.

Grafik kanan memperlihatkan *Training* and *Validation Accuracy*. Sumbu Y menunjukkan akurasi model, yang berkisar antara 0.8 (80%) hingga 1.0 (100%). Angka 0.8 pada awal pelatihan menunjukkan bahwa akurasi model masih rendah, menandakan banyak prediksi yang salah. Namun, akurasi model meningkat dengan cepat dan mendekati 1.0, menunjukkan bahwa model berhasil memprediksi hampir semua sampel dengan benar pada data training.

Sama seperti grafik kiri, sumbu X menunjukkan jumlah epoch. Pada epoch 0 hingga 10, akurasi meningkat pesat, mencapai nilai mendekati 1.0 untuk training accuracy. Namun, setelah epoch 10, akurasi validasi mulai fluktuatif, menunjukkan bahwa performa model menjadi tidak stabil pada data validasi, yang juga menandakan adanya i

2. Pengujian *Word embedding Word2vec*

Pengujian *Word2vec* epoch 1 hingga 50 melibatkan proses melatih model *Word2vec* pada data teks untuk menghasilkan representasi vektor dari kata-kata. Berikut adalah penjelasan umum yang terjadi selama pengujian tersebut:

```
Epoch 1/50
420/420 - 3s - loss: 0.4789 - acc: 0.7820 - val_loss: 0.3648 - val_acc: 0.8780 - 3s/epoch - 6ms/step
Epoch 2/50
420/420 - 2s - loss: 0.2845 - acc: 0.8906 - val_loss: 0.2693 - val_acc: 0.9129 - 2s/epoch - 4ms/step
Epoch 3/50
420/420 - 1s - loss: 0.2192 - acc: 0.9209 - val_loss: 0.2371 - val_acc: 0.9290 - 1s/epoch - 4ms/step
Epoch 4/50
420/420 - 1s - loss: 0.1834 - acc: 0.9362 - val_loss: 0.2225 - val_acc: 0.9316 - 1s/epoch - 3ms/step
Epoch 5/50
420/420 - 1s - loss: 0.1586 - acc: 0.9455 - val_loss: 0.2186 - val_acc: 0.9236 - 1s/epoch - 3ms/step
Epoch 6/50
420/420 - 1s - loss: 0.1484 - acc: 0.9523 - val_loss: 0.2099 - val_acc: 0.9276 - 1s/epoch - 3ms/step
Epoch 7/50
420/420 - 1s - loss: 0.1309 - acc: 0.9532 - val_loss: 0.2171 - val_acc: 0.9155 - 1s/epoch - 3ms/step
Epoch 8/50
420/420 - 1s - loss: 0.1184 - acc: 0.9569 - val_loss: 0.2157 - val_acc: 0.9196 - 1s/epoch - 3ms/step
Epoch 9/50
420/420 - 1s - loss: 0.1084 - acc: 0.9615 - val_loss: 0.2100 - val_acc: 0.9182 - 1s/epoch - 3ms/step
Epoch 10/50
420/420 - 1s - loss: 0.0994 - acc: 0.9671 - val_loss: 0.2097 - val_acc: 0.9236 - 1s/epoch - 4ms/step
```

Gambar 17. Proses epoch 1-10 (90:10)

Pada epoch pertama, model mencapai nilai loss sebesar 0.4789 dengan akurasi 78.20%, sementara pada data validasi, nilai loss adalah 0.3648 dengan akurasi 87.80%. Pada epoch kedua, loss menurun drastis menjadi 0.2845 dengan akurasi meningkat menjadi 89.06%. Pada data validasi, loss juga menurun menjadi 0.2693 dengan akurasi 91.29%, menunjukkan peningkatan kinerja model.

Seiring bertambahnya epoch, model terus menunjukkan penurunan loss dan peningkatan akurasi pada data pelatihan. Pada epoch ketiga hingga kelima,

loss pelatihan menurun dari 0.2192 menjadi 0.1586 dengan akurasi meningkat dari 92.09% menjadi 94.55%. Pada data validasi, loss juga terus menurun hingga mencapai 0.2186 pada epoch kelima dengan akurasi yang sedikit menurun menjadi 92.36%.

Pada epoch keenam hingga kesepuluh, model menunjukkan penurunan loss yang konsisten pada data pelatihan dari 0.1484 menjadi 0.0994. Akurasi pelatihan juga terus meningkat mencapai 96.71%. Pada data validasi, meskipun nilai loss menurun hingga 0.2097, akurasi cenderung stabil, berada di kisaran 91.82% hingga 92.36%. Penurunan pada nilai loss pada data validasi tidak selalu diikuti oleh peningkatan signifikan dalam akurasi.

Namun, setelah epoch kesepuluh, model mulai menunjukkan tanda-tanda *overfitting.* Meskipun loss pelatihan terus menurun dan akurasi pelatihan meningkat mencapai 98.15% pada epoch kedua puluh satu, loss pada data validasi mulai meningkat dan akurasi pada data validasi mulai menurun, menjadi 90.75% pada epoch yang sama. Fenomena ini semakin jelas pada epoch-epoch berikutnya.

Pada epoch dua puluh lima hingga tiga puluh, nilai loss pada data pelatihan semakin rendah, menunjukkan bahwa model belajar dengan sangat baik dari data pelatihan. Namun, pada data validasi, nilai loss terus meningkat hingga mencapai 0.3736 pada epoch ketiga puluh, dengan akurasi validasi yang menurun hingga 89.41%. Hal ini mengindikasikan bahwa model mulai mengalami *overfitting*, di mana performa pada data validasi tidak sebaik performa pada data pelatihan.

```
Epoch 41/50
420/420 - 1s - loss: 0.0273 - acc: 0.9906 - val_loss: 0.4204 - val_acc: 0.9142 - 1s/epoch - 3ms/step
Epoch 42/50
420/420 - 1s - loss: 0.0226 - acc: 0.9918 - val_loss: 0.4309 - val_acc: 0.9021 - 1s/epoch - 3ms/step
Epoch 43/50
420/420 - 1s - loss: 0.0242 - acc: 0.9909 - val_loss: 0.4408 - val_acc: 0.9062 - 1s/epoch - 3ms/step
Epoch 44/50
420/420 - 1s - loss: 0.0227 - acc: 0.9918 - val_loss: 0.4371 - val_acc: 0.9021 - 1s/epoch - 3ms/step
Epoch 45/50
420/420 - 1s - loss: 0.0193 - acc: 0.9934 - val_loss: 0.4743 - val_acc: 0.9075 - 1s/epoch - 3ms/step
Epoch 46/50
420/420 - 1s - loss: 0.0180 - acc: 0.9934 - val_loss: 0.4891 - val_acc: 0.9115 - 1s/epoch - 3ms/step
Epoch 47/50
420/420 - 1s - loss: 0.0195 - acc: 0.9921 - val_loss: 0.4794 - val_acc: 0.9075 - 1s/epoch - 3ms/step
Epoch 48/50
420/420 - 1s - loss: 0.0210 - acc: 0.9921 - val_loss: 0.4813 - val_acc: 0.9115 - 1s/epoch - 3ms/step
Epoch 49/50
420/420 - 1s - loss: 0.0203 - acc: 0.9917 - val_loss: 0.4862 - val_acc: 0.9062 - 1s/epoch - 3ms/step
Epoch 50/50
420/420 - 1s - loss: 0.0210 - acc: 0.9921 - val_loss: 0.4948 - val_acc: 0.9075 - 1s/epoch - 3ms/step
```

Gambar 21. Epoch 41-50 (90:10)

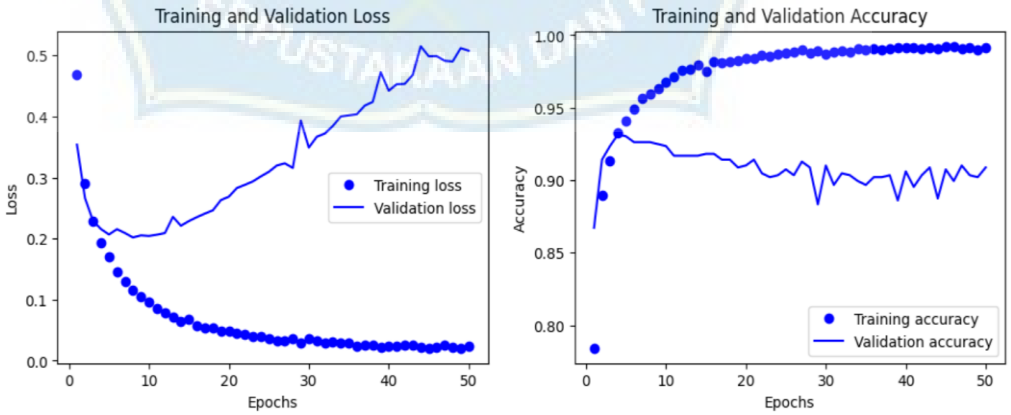
Log hasil pelatihan di atas menunjukkan kinerja model selama 10 epoch terakhir dari total 50 epoch. Pada periode ini, nilai loss pada data pelatihan terus menurun dari 0,0273 pada epoch 41 menjadi 0,0210 pada epoch 50, menandakan bahwa model semakin baik dalam menyesuaikan diri dengan data pelatihan. Akurasi (acc) pada data pelatihan tetap stabil pada tingkat yang sangat tinggi, sekitar 99%. Namun, meskipun performa pada data pelatihan meningkat, nilai *validation loss* (val_loss) justru mengalami sedikit peningkatan, dari 0,4204 pada epoch 41 menjadi 0,4948 pada epoch 50. Hal ini menunjukkan potensi *overfitting*, di mana model bekerja sangat baik pada data pelatihan tetapi kurang optimal pada data validasi. *Validation accuracy* (val_acc) selama periode ini relatif stabil, berkisar antara 90,62% hingga 91,15%, menunjukkan bahwa kemampuan model untuk menggeneralisasi ke data baru tidak meningkat secara signifikan meskipun jumlah epoch bertambah.

Tabel 9 Hasil Pengujian *Word2vec*

Epoch	Loss	Accurasy	Val Loss	Vall Acc
10	0.95%	99.68%	0.2002	93.16%
20	0.47%	99.29%	0.2009	92.90%
30	0.30%	99.75%	0.2643	92.09%
40	0.24%	99.18%	0.3334	91.29%
50	0.18%	99.34%	0.4376	90.62%

Tabel di atas menunjukkan hasil pengujian model *Word2vec* pada berbagai epoch untuk tugas analisis sentimen. Pada epoch 10, model mencapai akurasi pelatihan yang sangat tinggi sebesar 99,68% dengan nilai loss sebesar 0,95% dan nilai val loss sebesar 0,2002. *Val accuracy* pada tahap ini mencapai 93,16%, menunjukkan model dapat memprediksi dengan cukup baik pada data validasi. Namun, saat pelatihan dilanjutkan hingga epoch 50, meskipun akurasi pelatihan tetap tinggi dengan nilai 99,34%, terdapat penurunan pada *val accuracy* hingga 90,62%, dan val loss meningkat menjadi 0,4376. Hal ini menunjukkan bahwa model mengalami *overfitting*, di mana performa pada data pelatihan tetap tinggi, tetapi kemampuan untuk menggeneralisasi pada data baru menurun.

1. Grafik *Word2vec*



Gambar 15 Training Validation Loss dan Accurasy *Word2vec*

Gambar menunjukkan dua grafik yang berkaitan dengan pelatihan dan validasi model pembelajaran mesin (machine learning), yang masing-masing menunjukkan Loss dan *Accuracy* selama sejumlah epochs.

Pada grafik *Training* and *Validation Loss* (Grafik Kiri), sumbu X menunjukkan jumlah epoch yang telah dilalui selama pelatihan, dengan angka seperti 0, 10, 20, 30, 40, dan 50. Epoch merupakan satu iterasi penuh dari seluruh dataset melalui algoritma pelatihan. Sumbu Y pada grafik ini menggambarkan nilai loss, dengan angka seperti 0.0, 0.1, 0.2, 0.3, 0.4, dan 0.5, yang mencerminkan seberapa buruk model dalam melakukan prediksi—semakin kecil nilainya, semakin baik performa model. Training Loss pada grafik ini menurun dari sekitar 0,5 di epoch awal hingga mendekati 0,0 di akhir epoch ke-50, menandakan bahwa model semakin baik dalam pelatihan. Namun, Validation Loss menunjukkan pola berbeda; setelah awalnya menurun dari sekitar 0,3 menjadi 0,1, nilai ini mulai meningkat kembali hingga mencapai sekitar 0,4 setelah 50 epoch. Hal ini mengindikasikan bahwa model mengalami overfitting, di mana performa pada data validasi menurun setelah titik tertentu.

Pada grafik Training and Validation Accuracy (Grafik Kanan), sumbu X juga menunjukkan jumlah epoch yang telah dilalui, dengan angka yang sama seperti pada grafik loss. Sumbu Y menggambarkan tingkat akurasi model, dengan angka seperti 0.80, 0.85, 0.90, 0.95, dan 1.00, yang mencerminkan persentase prediksi yang benar dibandingkan dengan total prediksi. *Training Accuracy* meningkat cepat dari sekitar 0,8 (80%) di awal hingga hampir 1,0 (100%) mendekati epoch ke-50, menunjukkan bahwa model sangat efektif dalam memprediksi data pelatihan. Sebaliknya, *Validation Accuracy* mencapai puncaknya sekitar 0,95 (95%) di sekitar epoch ke-10, kemudian mengalami fluktuasi dan sedikit penurunan di akhir epoch ke-50, menunjukkan bahwa model tidak sebaik itu dalam memprediksi data yang belum pernah dilihat setelah beberapa epoch tertentu Grafik *Word2vec*

Secara keseluruhan, angka-angka dalam grafik ini menunjukkan bahwa meskipun model memiliki kinerja yang sangat baik pada data pelatihan, kemampuan model untuk menggeneralisasi ke data baru menurun setelah sekitar 20-30 epoch, yang terlihat dari peningkatan validation loss dan fluktuasi pada validation accuracy. Ini mengindikasikan bahwa model perlu disesuaikan untuk mengurangi overfitting, misalnya dengan menggunakan teknik regularisasi atau *early stopping.*

E. Evaluasi Kinerja Model

Evaluasi kinerja adalah proses untuk menilai seberapa baik suatu model atau sistem bekerja dalam menyelesaikan tugas tertentu, berdasarkan metrik-metrik yang telah ditetapkan. Dalam konteks pembelajaran mesin atau pemrosesan bahasa alami, evaluasi kinerja digunakan untuk mengukur efektivitas model dalam melakukan prediksi atau klasifikasi berdasarkan data yang ada.

Ada beberapa metrik yang umum digunakan dalam evaluasi kinerja model:

1. *Accuracy* (Akurasi) adalah persentase prediksi yang benar dari total prediksi yang dibuat oleh model. Meskipun akurasi adalah metrik yang paling sederhana dan intuitif, ia memiliki keterbatasan, terutama jika data tidak seimbang (misalnya, jika satu kelas jauh lebih dominan dibandingkan kelas lain).

2. *Precision* (Presisi) Presisi mengukur akurasi dari prediksi positif model, yaitu berapa banyak prediksi yang dinyatakan positif benar-benar positif. Presisi penting ketika biaya kesalahan positif palsu (false positives) tinggi, seperti dalam kasus deteksi penipuan atau diagnosis penyakit.

3. *Recall* atau sensitivitas, mengukur kemampuan model untuk mendeteksi semua sampel positif sebenarnya. Ini sangat penting dalam situasi di mana melewatkan prediksi positif (false negatives) memiliki konsekuensi serius, seperti dalam pendeteksian kanker.

4. *F1-Score* adalah rata-rata harmonis dari presisi dan *Recall*, memberikan gambaran yang seimbang ketika ada trade-off antara keduanya. Ini berguna ketika kita perlu mempertimbangkan keseimbangan antara presisi dan *Recall*, terutama dalam kasus data yang tidak seimbang.

5. *Loss Function* (Fungsi Kehilangan) Loss function mengukur seberapa jauh prediksi model dari nilai sebenarnya. Selama pelatihan, model mencoba meminimalkan loss untuk meningkatkan akurasinya. Nilai loss yang rendah menunjukkan bahwa model telah mempelajari pola data dengan baik.

6. *Confusion Matrix* adalah tabel yang menunjukkan distribusi prediksi model ke dalam kategori yang benar dan salah. Dari confusion matrix, kita bisa mendapatkan lebih banyak wawasan tentang presisi, *Recall*, dan kesalahan model, serta mengidentifikasi kelas mana yang paling sulit diprediksi.

Evaluasi kinerja model biasanya dilakukan pada data validasi atau data tes yang tidak digunakan dalam pelatihan model. Ini untuk memastikan bahwa model dapat menggeneralisasi dengan baik ke data yang belum pernah dilihat sebelumnya dan tidak hanya menghafal data pelatihan. Selain itu, beberapa teknik seperti cross-validation digunakan untuk memastikan bahwa evaluasi kinerja model lebih robust dan tidak bergantung pada satu set data validasi tertentu.

Berikut adalah hasil evaluasi dari sebuah model prediksi dalam bentuk metrik evaluasi seperti *F1 Score*, *Precision*, dan *Recall*, serta output array hasil prediksi dan Klasifikasi

a. Hasil Prediksi

   1. *F1-Score, Precision,* dan *Recall*

      F1-Score (0.8976), Ini adalah rata-rata harmonis dari *Precision* dan *Recall*. Skor ini menunjukkan keseimbangan antara kemampuan model dalam mendeteksi data positif yang benar dan menghindari kesalahan dalam mendeteksi data negatif.

Precision (0.8793), Ini mengindikasikan bahwa dari semua prediksi positif yang dibuat oleh model, 89.84% benar-benar positif.

Recall (0.9168), Ini menunjukkan bahwa model mampu mendeteksi 89.94% dari semua data positif yang sebenarnya

Berikut adalah tabel yang merepresentasikan hasil dari evaluasi model *Glove* berdasarkan metrik *F1 Score, Precision,* dan *Recall.*

Tabel 10 Hasil Prediksi model *Glove*

Kelas	*Precision*	*Recall*	*F1-Score*	Support
0	0.83	0.83	0.83	227
1	0.92	0.92	0.92	519
Micro avg	0.90	0.90	0.90	746
macroavg	0.88	0.88	0.88	746
Weighted avg	0.90	0.90	0.90	746
Samples avg	0.90	0.90	0.90	746

Tabel 6 di atas menyajikan hasil prediksi model *GloVe* dalam analisis sentimen dengan memaparkan berbagai metrik evaluasi. Pada kelas 0, yang mewakili sentimen negatif, data yang termasuk dalam kelas ini seperti ulasan buruk, komentar yang tidak puas, atau *feedback* yang mencerminkan ketidakpuasan. *Precision* mencapai 0.83, yang berarti 83% dari prediksi model untuk kelas negatif adalah benar. *Recall* untuk kelas ini juga 0.83, mengindikasikan bahwa 83% dari data yang sebenarnya negatif berhasil diidentifikasi dengan benar oleh model. *F1-Score* untuk kelas 0 adalah 0.83, yang merupakan rata-rata harmonis dari *Precision* dan *Recall*, sedangkan *support* menunjukkan bahwa ada 227 data yang benar-benar termasuk dalam kelas negatif.

Untuk kelas 1, yang mencerminkan sentimen positif, seperti ulasan baik, komentar yang memuji, atau feedback yang mencerminkan kepuasan. *Precision* mencapai 0.92, yang berarti 92% dari prediksi model untuk kelas

positif adalah benar. *Recall* pada kelas ini juga berada di angka 0.92, yang menunjukkan bahwa model berhasil mengidentifikasi 92% dari semua data yang sebenarnya positif. *F1-Score* untuk kelas 1 juga 0.92, yang menandakan keseimbangan yang baik antara *Precision* dan *Recall. Support* pada kelas ini adalah 519, menunjukkan jumlah data yang sebenarnya termasuk dalam kategori positif.

Mikro rata-rata (*Micro avg*) dari *Precision, Recall,* dan *F1-Score* adalah 0.90, yang merupakan perhitungan rata-rata dari seluruh data tanpa mempertimbangkan perbedaan antar kelas. Mikro rata-rata ini menunjukkan performa umum model terhadap keseluruhan dataset yang terdiri dari 746 data (227 untuk kelas 0 dan 519 untuk kelas 1).

Makro rata-rata (Macro avg) memiliki nilai *Precision, Recall,* dan *F1-Score* sebesar 0.88. Ini adalah rata-rata dari metrik-metrik tersebut untuk masing-masing kelas, di mana setiap kelas dianggap memiliki bobot yang sama. Total jumlah data yang digunakan juga sebanyak 746.

Rata-rata tertimbang *(Weighted avg)* untuk *Precision, Recall,* dan *F1-Score* juga berada di angka 0.90. Perhitungan ini memperhitungkan proporsi data di setiap kelas, memberikan bobot lebih pada kelas dengan lebih banyak data. Total jumlah data yang dipertimbangkan sama, yaitu 746.

Rata-rata sampel (Samples avg) juga menghasilkan *Precision, Recall,* dan *F1-Score* sebesar 0.90, yang serupa dengan rata-rata tertimbang, menegaskan konsistensi performa model di seluruh data yang digunakan.

2. *F1-Score, Precision,* dan *Recall*

   F1 Score (0.9113), mengukur keseimbangan antara *Precision* dan *Recall.*

   Precisio (0.9111), Persentase prediksi positif yang benar.

   Recall (0.9115), Persentase data positif yang benar teridentifikasi.

Berikut adalah tabel yang merepresentasikan hasil dari evaluasi model *Word2vec* berdasarkan *metrik F1 Score, Precision,* dan *Recall:*

Tabel 11 Hasil Prediksi Model *Word2vec*

Kelas	*Precision*	*Recall*	*F1-Score*	Support
0	0.86	0.83	0.85	229
1	0.93	0.94	0.93	517
Micro avg	0.91	0.91	0.91	746
macroavg	0.90	0.89	0.90	746
Weighted avg	0.91	0.91	0.91	746
Samples avg	0.91	0.91	0.91	746

Tabel yang ditampilkan adalah hasil evaluasi dari model *Word2vec* yang digunakan dalam analisis sentimen, mengukur kinerja model dengan metrik seperti *Precision, Recall, F1-Score,* dan Support untuk dua kelas sentimen, yaitu negatif (Kelas 0) dan positif (Kelas 1). Untuk Kelas 0, yang mewakili sentimen negatif, model mencapai *Precision* sebesar 0.86, menunjukkan bahwa 86% dari prediksi negatif oleh model adalah benar-benar negatif, dengan *Recall* 0.83 yang mengindikasikan bahwa model berhasil mengidentifikasi 83% dari semua contoh negatif yang ada. *F1-Score* untuk kelas ini adalah 0.85, menunjukkan keseimbangan yang baik antara *Precision* dan *Recall*.

Di sisi lain, Kelas 1, yang mewakili sentimen positif, menunjukkan kinerja yang lebih tinggi dengan *Precision* 0.93 dan *Recall* 0.94, artinya model tidak hanya akurat dalam prediksinya (93% akurat) tetapi juga sangat efektif dalam mengidentifikasi sentimen positif yang ada (94% ditemukan). *F1-Score* yang tinggi, 0.93, memperkuat efisiensi model dalam menangani sentimen positif. Support untuk Kelas 0 adalah 229 sementara untuk Kelas 1 adalah 517, menunjukkan distribusi jumlah sampel yang ditangani oleh model.

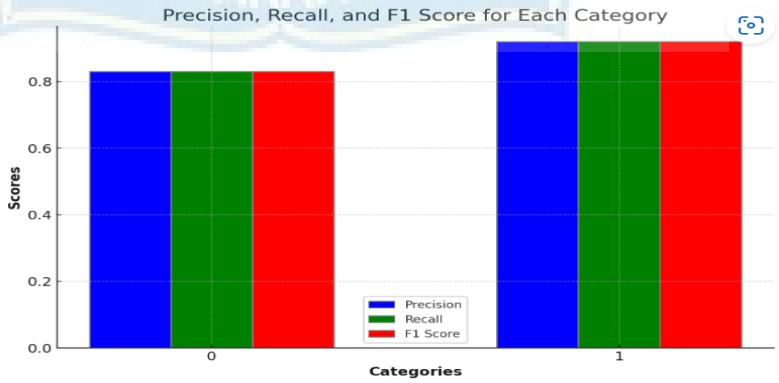
Pada evaluasi lebih luas, nilai *Micro avg* dari *Precision*, *Recall*, dan *F1-Score* semua berada pada 0.91, menggambarkan kinerja model secara keseluruhan tanpa membedakan antar kelas. Sementara itu, nilai *Macro avg* adalah 0.90 untuk semua metrik, menunjukkan rata-rata performa model di semua kelas tanpa mempertimbangkan ukuran kelas. Terakhir, Weighted avg juga mencatat nilai 0.91 untuk ketiga metrik, yang menyesuaikan perhitungan dengan ukuran relatif masing-masing kelas, memberikan gambaran yang akurat tentang kinerja model berdasarkan distribusi sampel. Keseluruhan, hasil ini menunjukkan bahwa model *Word2vec* melakukan pekerjaan yang sangat baik dalam mengklasifikasikan sentimen, terutama lebih cenderung akurat dalam mendeteksi dan menilai sentimen positif.

b. Klasifikasi

1. Klasifikasi *GloVe*

```
24/24 [==============================] - 0s 2ms/step
F1 Score: 0.8954423592493298
Precision: 0.8954423592493298
Recall: 0.8954423592493298
              precision    recall  f1-score   support

           0       0.83      0.83      0.83       227
           1       0.92      0.92      0.92       519

    accuracy                           0.90       746
   macro avg       0.88      0.88      0.88       746
weighted avg       0.90      0.90      0.90       746
```

Gambar 16 Hasil Klasifikasi *GloVe*



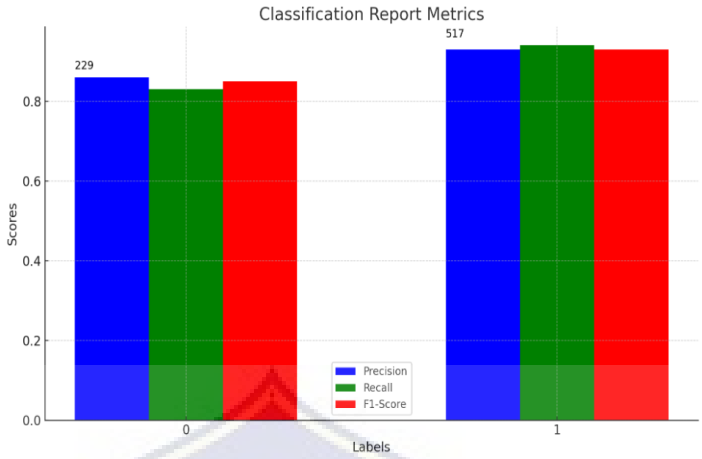
Gambar 17 Grafik klasifikasi *GloVe*

Grafik di atas menggambarkan hasil metrik evaluasi model klasifikasi, meliputi *Precision*, *Recall*, dan *F1-Score* untuk kedua label, yaitu 0 dan 1. Pada label 0, *Precision* tercatat sebesar 0.86, yang berarti 86% dari prediksi model untuk label 0 adalah benar. *Recall* untuk label ini sebesar 0.83 menunjukkan bahwa model berhasil mengidentifikasi 83% dari seluruh data aktual yang memang berlabel 0. *F1-Score* sebesar 0.85 menggambarkan keseimbangan antara *Precision* dan *Recall* untuk label 0. Sementara itu, pada label 1, *Precision* mencapai 0.93, yang berarti 93% dari prediksi model untuk label 1 adalah akurat. *Recall* untuk label ini sebesar 0.94, menunjukkan bahwa model berhasil mengidentifikasi 94% dari data aktual yang berlabel 1. *F1-Score* untuk label 1 berada di angka 0.93, menandakan performa yang seimbang antara *Precision* dan *Recall*. Selain itu, angka support yang terlihat pada grafik menunjukkan jumlah sampel pada setiap kategori, dengan 229 untuk label 0 dan 517 untuk label 1. Secara keseluruhan, grafik ini menunjukkan bahwa model memiliki performa yang baik, dengan kinerja yang lebih kuat pada label 1 dibandingkan label 0.

2. Klasifikasi *Word2vec*

```
24/24 [==============================] - 0s 2ms/step
F1 Score: 0.9069787697010997
Precision: 0.9067560692398866
Recall: 0.9075067024128687
              precision    recall  f1-score   support

           0       0.86      0.83      0.85       229
           1       0.93      0.94      0.93       517

    accuracy                           0.91       746
   macro avg       0.89      0.89      0.89       746
weighted avg       0.91      0.91      0.91       746
```

Gambar 18 Hasil Klasifikasi *Word2vec*



Gambar 19 Grafik Klasifikasi *Word2vec*

Grafik di atas menggambarkan hasil metrik evaluasi model klasifikasi, meliputi *Precision*, *Recall*, dan *F1-Score* untuk kedua label, yaitu 0 dan 1. Pada label 0, *Precision* tercatat sebesar 0.86, yang berarti 86% dari prediksi model untuk label 0 adalah benar. *Recall* untuk label ini sebesar 0.83 menunjukkan bahwa model berhasil mengidentifikasi 83% dari seluruh data aktual yang memang berlabel 0. *F1-Score* sebesar 0.85 menggambarkan keseimbangan antara *Precision* dan *Recall* untuk label 0. Sementara itu, pada label 1, *Precision* mencapai 0.93, yang berarti 93% dari prediksi model untuk label 1 adalah akurat. *Recall* untuk label ini sebesar 0.94, menunjukkan bahwa model berhasil mengidentifikasi 94% dari data aktual yang berlabel 1. *F1-Score* untuk label 1 berada di angka 0.93, menandakan performa yang seimbang antara *Precision* dan *Recall*. Selain itu, angka support yang terlihat pada grafik menunjukkan jumlah sampel pada setiap kategori, dengan 229 untuk label 0 dan 517 untuk label 1. Secara keseluruhan, grafik ini menunjukkan bahwa model memiliki performa yang baik, dengan kinerja yang lebih kuat pada label 1 dibandingkan label 0.

F. Perbandigan Hasil

Dari hasil pengujian yang telah dilakukan, dapat disimpulkan bahwa kedua metode embedding, *Word2vec* dan *GloVe*, memiliki kinerja yang sangat baik dalam konteks analisis sentimen. Berikut adalah hasil

perbandingan yang diperoleh dari akurasi, Presisi,*Recall*,*F1 Score*, dan *support*.

Tabel 12 Hasil Perbandingan Model *Word2vec* dan *GloVe*

Metrik	Kelas	*Word2vec*	*GloVe*
Akurasi		93 %	90%
Presisi	Positif	86%	83%
	Negatif	93%	92%
Recall	Positif	83%	83%
	Negatif	94%	92%
F1Score	Positif	85%	83%
	Negatif	93%	92%

Berdasarkan tabel perbandingan di atas dapat simpulkan bahwa perbandingan kinerja dua model pembelajaran mesin, *Word2vec* dan *GloVe*, menggunakan beberapa metrik evaluas seperti akurasi, presisi, *Recall*, *F1Score*, dan *support*, yang diterapkan pada dua kelas, yaitu kelas positif dan kelas negatif. Secara keseluruhan, *Word2vec* menunjukkan performa yang lebih baik dibandingkan *GloVe* pada semua metrik yang diukur. *Word2vec* memiliki akurasi sedikit lebih tinggi, yaitu 93% dibandingkan dengan 90% pada *GloVe*. Dari segi presisi, *Word2vec* mencatat 86% untuk kelas positif dan 93% untuk kelas negatif, mengungguli *GloVe* yang mencatat 83% dan 92% untuk kelas yang sama. Pada metrik *Recall*, *Word2vec* juga lebih baik dengan skor 83% untuk kelas positif dan 94% untuk kelas negatif, sementara *GloVe* memiliki skor *Recall* 83% dan 92%. *F1Score*, yang merupakan rata-rata harmonis dari presisi dan *Recall*, menunjukkan *Word2vec* unggul dengan skor 85% untuk kelas positif dan 93% untuk kelas negatif, dibandingkan dengan *GloVe* yang mencatat 83% dan 92%. *Support*, yang menunjukkan jumlah data dalam masing-masing kelas, hampir sama antara kedua model, dengan sedikit perbedaan dalam jumlah data positif dan negatif. Keseluruhan hasil ini menunjukkan bahwa

Word2vec lebih unggul dalam klasifikasi teks dibandingkan *GloVe* pada konteks yang diuji dalam tabel ini.

UNIVERSITAS MUHAMMADIYAH MAKASSAR
LEMBAGA PERPUSTAKAAN DAN PENERBITAN

BAB V
KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan hasil pengujian yang telah dilakukan pada penelitian ini, dapat disimpulkan bahwa :

1. Dalam penelitian ini, dapat disimpulkan bahwa model *Convolutional Neural Network* (CNN) dapat diimplementasikan dalam analisis sentimen dengan menggunakan dua *teknik word embedding*, yaitu *GloVe* dan *Word2vec*. Kedua metode embedding ini memungkinkan CNN untuk memproses dan menganalisis teks secara efisien, meskipun mereka memiliki pendekatan yang berbeda dalam representasi kata. Dalam perbandingan kinerja, *Word2vec* menunjukkan hasil yang sedikit lebih baik daripada *GloVe* dalam analisis sentimen pada dataset saran dan kritik mahasiswa Universitas Muhammadiyah Makassar dari sistem SIMAK, menunjukkan kemampuannya dalam menangkap konteks spesifik yang relevan.
2. Berdasarkan hasil penelitian perbandingan Kinerja Metode *Word embedding GloVe* dan *Word2vec* menunjukkan bahwa *Word2vec* secara keseluruhan memiliki performa yang lebih baik dibandingkan dengan *GloVe* dalam konteks analisis sentimen. *Word2vec* mencapai akurasi 93%, sedikit lebih tinggi dibandingkan dengan 90% dari *GloVe*. Dalam hal presisi, *Word2vec* menunjukkan keunggulan dengan presisi positif sebesar 86% dan presisi negatif sebesar 93%, dibandingkan dengan 83% dan 92% pada *GloVe*. Begitu pula dengan *Recall*, *Word2vec* mencatatkan nilai yang lebih tinggi, yaitu 83% untuk kelas positif dan 94% untuk kelas negatif, sementara *GloVe* mencatatkan nilai 83% dan 92%. *F1-Score* juga mengindikasikan bahwa *Word2vec* lebih unggul, dengan nilai 85% untuk kelas positif dan 93% untuk kelas negatif, dibandingkan dengan 83% dan 92% pada *GloVe*

B. Saran

Berdasarkan Kesimpulan yang diperoleh dari penelitian ini Disarankan untuk menggunakan *Word2vec* sebagai *metode word embedding* dalam analisis sentimen karena kinerjanya yang lebih baik dibandingkan *GloVe*. Pengoptimalan parameter model CNN, seperti ukuran jendela dan dimensi *embedding*, dapat lebih meningkatkan performa. Selanjutnya Penelitian lanjutan dapat mengeksplorasi dataset yang berbeda atau menggunakan metode *embedding* yang lebih baru seperti BERT atau ELMo. Selain itu, penerapan model ini pada berbagai domain lain dapat memperkuat hasil penelitian. Jika memungkinkan, penggunaan perangkat keras yang lebih kuat dapat mempercepat pelatihan model, terutama untuk embedding dengan dimensi tinggi.

DAFTAR PUSTAKA

Ardian Pradana, Y., Cholissodin, I., & Kurnianingtyas, D. (2023). Analisis Sentimen Pemindahan Ibu Kota Indonesia pada Media Sosial Twitter menggunakan Metode LSTM dan Word2Vec. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *7*(5), 2389–2397. http://j-ptiik.ub.ac.id

Arsi, P., & Waluyo, R. (2021). Analisis Sentimen Wacana Pemindahan Ibu Kota Indonesia Menggunakan Algoritma Support Vector Machine (SVM). *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *8*(1), 147. https://doi.org/10.25126/jtiik.0813944

Dyantono, A. M. D., & Putra, R. E. (2023). Perbandingan Sent2vec TF-IDF Logistic Regression dan Word2vec CNN pada hasil Sentiment Analysis Youtube Comment. *Journal of Informatics and Computer Science (JINACS)*, *5*(01), 63–72.

Faiq, M., Putro, A., & Setiawan, E. B. (2022). Analisis Sentimen Terhadap Kebijakan Pemerintah dengan Feature Expansion Metode GloVe pada Media sosial Twitter. *E-Proceeding of Engineering*, *9*(1), 54–66.

Fitriansyah, A. R., Informatika, F., Telkom, U., Sibaroni, Y., Informatika, F., & Telkom, U. (2023). *Analisis Sentimen Terhadap Pembangunan Kereta Cepat Jakarta - Bandung Pada Media Sosial Twitter Menggunakan Metode SVM dan GloVe Word Embedding*. *10*(2), 1713–1723.

Indrayana, R., & Solikhin, M. (2020). Analisis Sentimen Pada Media Sosial Twitter. *Seminar Nasional Pendidikanipa Danmatematika*, *12*(1), 79–86.

Informatika, J., Jik, K., & Juli, V. N. (2022). *Analisis Sentimen Twitter Bahasa Indonesia Menggunakan*. *6*(2), 7821–7829.

Listyarini, S. N., & Anggoro, D. A. (2021). Analisis Sentimen Pilkada di Tengah Pandemi Covid-19 Menggunakan Convolution Neural Network (CNN).

Jurnal Pendidikan Dan Teknologi Indonesia, *1*(7), 261–268. https://doi.org/10.52436/1.jpti.60

Maulana, A. R., Wijoyo, S. H., & Mursityo, Y. T. (2023). *Sekolah Dasar Dan Sekolah Menengah Pada Media Sosial Twitter Dengan Menggunakan Metode Word Embedding Dan Long Short-Term Memory Networks (Lstm) Sentiment Analysis of Implementation Independent Curriculum Policy Elementary and Secondary School on Twitte*. *10*(3), 523–530. https://doi.org/10.25126/jtiik.2023106977

Nurdin, A., Anggo Seno Aji, B., Bustamin, A., & Abidin, Z. (2020). Perbandingan Kinerja Word Embedding Word2Vec, Glove, Dan Fasttext Pada Klasifikasi Teks. *Jurnal Tekno Kompak*, *14*(2), 74. https://doi.org/10.33365/jtk.v14i2.732

PRADANA, F. A. (2023). *Perbandingan Word Embedding Word2Vec, Glove, Dan Fasttext Menggunakan Deep Learning Pada Ulasan Kondisi Pengguna Obat Kesehatan*. 7–65.

Riani, E., Yonathan, J., & Oliver, L. (2021). Audit Sistem Informasi Akademik (SIMAK) Menggunakan Framework COBIT 5 di Universitas Universal, Journal of Digital Ecosystem for Natural Sustainability (JoDENS). *Journal of Digital Ecosystem for Natural Sustainability (JoDENS)*, *1*(2), 2798–6179.

Wulansari, I., Arief, R., Manajemen, M., Informasi, S., Gunadarma, U., Barat, J., & Network, C. N. (n.d.). *ANALISIS PERFORMA METODE CONVOLUTIONAL NEURAL NETWORK (CNN) DENGAN WORD EMBEDDING*. *28*(3), 252–264.

Yuliska, Y., Qudsi, D. H., Lubis, J. H., Syaliman, K. U., & Najwa, N. F. (2021). Analisis Sentimen pada Data Saran Mahasiswa Terhadap Kinerja Departemen di Perguruan Tinggi Menggunakan Convolutional Neural Network. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *8*(5), 1067. https://doi.org/10.25126/jtiik.2021854842

Yuniarossy, B. A., Maulida Hindrayani, K., Damaliana, A. T., Studi, P., Data, S.,

Pembangunan, U., Veteran, N. ", & Timur, J. (2024). *Analisis Sentimen Terhadap Isu Feminisme Di Twitter Menggunakan Model Convolutional Neural Network (Cnn)*. *5*(1), 477–491. http://lebesgue.lppmbinabangsa.id/index.php/home

UNIVERSITAS MUHAMMADIYAH MAKASSAR
LEMBAGA PERPUSTAKAAN DAN PENERBITAN

LAMPIRAN

Lampiran 1 Pengumpulan data

1	id;kategori;saran;tgl_insert;;;
2	2;MHS;Kedepanya kami bisa diberikan dosen yang lebih produktif dan mampu beoerja sama dengan mahasiswa menyelesaikan permasalahan dalam pertemuan covid ini. Tapi untuk dosen saat ini sudah luar biasa kok;15/02/2022 07.01;;;
3	3;MHS;Saya berharap pelayanan akademik mampu memberikan pelayanan yang baik kepada mahasiswa dan seluruh masyarakat Unismuh.Terutama untuk para bapak/ibu dosen sekiranya mampu memberikan metode yang baik bagi mahasiswa sehingga
4	5;MHS;Memperbaiki yang baik menjadi lebih baik lagi.;15/02/2022 07.08;;;
5	8;MHS;Sel pihak kampus juga harus membuat rancangan kesepakatan perkuliahan online yang disetujui dosen dan mahasiswa serta meningkatkan kualitas layanan akedemik seperti portal online untuk kebutuhan kuliah online;15/02/2022 07.13;;;
6	9;MHS;Perlunya efesiensi waktu yang baik dalam pembelajaran agar matakuliah yabg lain tidak terganggu ;15/02/2022 07.13;;;
7	10;MHS;Apakah pelayanan akademik sudah memberikan pelayanan yang terbaik kepada mahasisswa ?;15/02/2022 07.19;;;
8	12;MHS;Saran saya agar proses pembelajaran secara online harus juga di selingi dengan pembelajaran secara offline agar para mahasiswa lebih mengerti lagi materi perkuliahan;15/02/2022 07.19;;;
9	14;MHS;Tetap pertahankan pembelajaran yang baik dan tingkat terus-menerus kedepannya;15/02/2022 07.21;;;
10	23;MHS;Semoga sarana dan prasarana kampus dapat digunakan secepatnya dan semoga kuliah online tidak lagi dilaksanakan;15/02/2022 07.24;;;
11	29;MHS;Semogah pembeljaranya semakin merdeka dan bisa offlinr trutama pada saat praktikum;15/02/2022 07.28;;;
12	;15/02/2022 07.28;;;;;
13	;;;;;;
14	;;;;;;
15	;;;;;;
16	;15/02/2022 07.32;;;;;
17	1) diharapkan agar Dosen mengajar pada waktu yang tepat; jam dan hari yang sesuai dengan jadwal dan tidak mengambil waktu istirahat kami untuk kuliah;;;;;
18	2) diharapkan agar dosen bisa menegur dan membimbing kita secara baik-baik apabila kita melakukan kesalahan ";15/02/2022 07.39;;;;;
19	50;MHS;Terima kasih;15/02/2022 07.44;;;
20	51;MHS;D Pembela| Karena Bi Sakit Mat Dan Pembelian Kuota Tiap Bulan. Saya Berharap Kedepannya Sudah Bisa Melakukan Pembelajaran Tatap Muka.;15/02/2022 07.44;;;
21	53;MHS;sa pembelajaran ataupun dari pelayanan akademik akan menjadi jauh lebih baik untuk kedepannya.;15/02/2022 07.45;;;
22	54;MHS;Peningkatan fasilitas kampus;15/02/2022 07.45;;;
23	56;MHS;-;15/02/2022 07.47;;;
24	57;MHS;Saran saya agar lebih ditingkatkan lagi proses pembelajaran daring yang lebik baik lagi;15/02/2022 07.50;;;
25	59;MHS;Ki diharapkan agar sering mengingatkan kepada mahasiswa masalah koneksi ketika ada kuis/UTS/UAS agar mahasiswa bisa mempersiapkan kemungkinan buruk yang terjadi terutama masalah koneksi internet.;15/02/2022 07.52;;;
26	60;MHS;Semoga para dosen bisa menjelaskan materi dengan kreatif agar mahasiswa mudah memahami;15/02/2022 07.54;;;
27	62;MHS;Semoga lebih baik kedepannya;15/02/2022 07.59;;;
28	64;MHS;Saran sya semoga bisah offline;15/02/2022 08.03;;;
29	66;MHS;Pendidikan agama Islam harus di tingkatkan;15/02/2022 08.10;;;
30	70;MHS;Si sebaiknya kampus segera melaksanakan perkuliahan offline agar mahasiswa dapat menikmati fasilitas kampus kembali;15/02/2022 08.18;;;
31	72;MHS;Semoga kedepannya lebih baik lagi;15/02/2022 08.21;;;

survey_saran

Lampiran 2 Pelabelan Data

	A	B
1	**ULASAN**	**LABEL**
2	Kedepanya kami bisa diberikan dosen yang lebih produktif dan mampu beoerja sama dengan mahasiswa menyelesaikan permasalahan dalam pertemuan covid ini. Tapi untuk dosen saat ini sudah luar biasa kok	kritik
3	Saya berharap pelayanan akademik mampu memberikan pelayanan yang baik kepada mahasiswa dan seluruh masyarakat Unismuh.Terutama untuk para bapak/ibu dosen sekiranya mampu memberikan metode yang baik bagi mahasiswa sehingga mahasiswa lebih aktif dar	kritik
4	Memperbaiki yang baik menjadi lebih baik lagi.	saran
5	Sebaiknya pihak kampus memantau proses penyaluran kouta internet , pihak kampus juga harus membuat rancangan kesepakatan perkuliahan online yang disetujui dosen dan mahasiswa serta meningkatkan kualitas layanan akedemik seperti portal online untuk kebutuhar	kritik
6	Perlunya efesiensi waktu yang baik dalam pembelajaran agar matakuliah yabg lain tidak terganggu	kritik
7	Apakah pelayanan akademik sudah memberikan pelayanan yang terbaik kepada mahasisswa ?	kritik
8	Saran saya agar proses pembelajaran secara online harus juga di selingi dengan pembelajaran secara offline agar para mahasiswa lebih mengerti lagi materi perkuliahan	saran
9	Tetap pertahankan pembelajaran yang baik dan tingkat terus-menerus kedepannya	saran
10	Semoga sarana dan prasarana kampus dapat digunakan secepatnya dan semoga kuliah online tidak lagi dilaksanakan	saran
11	Semogah pembeljaranya semakin merdeka dan bisa offlinr trutama pada saat praktikum	saran
12	1) diharapkan agar Dosen mengajar pada waktu yang tepat. 2) diharapkan agar dosen bisa menegur dan membimbing kita secara baik-baik apabila kita melakukan kesalahan " Terima kasih	saran
13	Dengan Melihat Keadaan Dalam Pembelajaran Daring (Online), Pembelajaran Online (Daring) Sangat Tidak Baik Bagi Kami Untuk Fokus belajar, Karena Banyak-Nya Kendela2 Yang Kami Alamin Seperti : Jelek Jaringan, Sakit Mata, Dan Pembelian Kuota Tiap Bulan. Saya B	kritik
14	semoga dengan adanya peningkatan pelayanan baik itu dalam proses perkuliahan, pembelajaran ataupun dari pelayanan akademik akan menjadi jauh lebih baik untuk kedepannya.	saran
15	Peningkatan fasilitas kampus	kritik
16	Saran saya agar lebih ditingkatkan lagi proses pembelajaran daring yang lebik baik lagi	saran
17	Karena proses perkuliahan masih daring, diharapkan agar sering mengingatkan kepada mahasiswa masalah koneksi ketika ada kuis/UTS/UAS agar mahasiswa bisa mempersiapkan kemungkinan buruk yang terjadi terutama masalah koneksi internet.	saran
18	Semoga para dosen bisa menjelaskan materi dengan kreatif agar mahasisv a mudah memahami	saran
19	Semoga lebih baik kedepannya	saran
20	Saran sya semoga bisah offline	saran
21	Pendidikan agama Islam harus di tingkatkan	kritik
22	Saran saya, sebaiknya kampus segera melaksanakan perkuliahan offline agar mahasiswa dapat menikmati fasilitas kampus kembali	saran
23	Semoga kedepannya lebih baik lagi	saran
24	Agar kedepannya buku referensi di perpustakaan diperbanyak lagi sesuai dengan buku yang dipakai pada tahun ajaran yang sedang digunakan	kritik
25	membuat susana belajar yang menyenangkan	saran
26	Saran saya semoga semester depan bisa kuliah offline	saran
27	Kepada dosen sekiranya dalam metode pembelajarannya haris di tingkatkan lagi keberagaman nya agar mahasisv a nya lebih tertarik apalagi dalam hal kuliah online ini	kritik
28	Semogah layanan dalam proses pembelajaran kedepannya bisa meningkat kualitasnya di kampus unismuh makassar. Sekian terima kasih	saran
29	SEMOGA UNIVERSITAS DAPAT TERUS MENINGKATKAN KAPASITAS DOSEN.....	saran
30	Jangan mempersulit mahasisv a(i)ketika dalam pelayanan	kritik
31	Konsisten terhadap jadv al yang telah di tentukan	kritik
32	Saran saya, Dosen dapat memberikan materi dan bimbingan kepada siswanya dengan menggunakan video, siaran langsung, atau rekaman. Agar memungkinkan kami dapat lebih memahami pelajaran dan bisa mengulangi apa yang diajarkan kepada kami.	saran
33	Tetap lebih efektif dalam melakukan pelayanan saat proses pembelajara supaya lebih maju lagi.	saran
34	Semoga dosen bisa memberikan kebijakan yang lebih baik kepada mahasisv a dalam menghadapi masalah.	saran
35	Menurut saya yang harus dilakukan untuk meningkatkan pelayanan akademik di program studi adalah dengan cara membuat ruang belajar dengan nyaman. Seperti dilengkapi penyejuk udara serta pengharum udara. Agar udaranya menjadi segar.	kritik
36	Sebaiknya dalam pembelajaran daring dosen dapat mengerti tidak semua mahasisv a dapat mengakses internet yang baik, dan dosen dapat mengerti jika ada kendala dalam keterlambatan mengirim tugas dan hal yang diakibatkan oleh jaringan yang bermasalah.	kritik
37	Diupayakan dosen masuk di jam mata kuliah dengan tepat waktu dan tidak menganti jadwal jam kuliah	kritik
38	Saya hanya berharap semoga kedepannya jauh lebih baik, Aamiin...	saran
39	saran terkait pelayanan akademik perlu dpertahankan, karna pelayanan yang diberikan sudah sangat baik	saran
40	Semoga kecepatan dosen menanggapi pertanyaan dari mahasisv a kedepannya bisa lebih cepat respon	saran
41	Kami harap dosen pengampu disetiap matakuliah dapat demokratis dalam pembelajaran. Mengedepankan diskusi dan tidak menyalahkan ataupun menyudutkan mahasiswa disetiap diskusi!!! Terima kasih	saran
42	Pihak akademik lebih fokus pada peningkatan kualitas mahasisv a	kritik
43	Semoga dosen dapat lebih mudah dihubungi	saran
44	semoga dosen lebih mudah di hubungi	saran
45	Pelayanan perlu di tingkatkan	kritik

Lampiran 3 Source Code Prepocessing

```
[1] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
!pip install openpyxl
```

```
Collecting openpyxl
  Downloading openpyxl-3.1.5-py2.py3-none-any.whl.metadata (2.5 kB)
Collecting et-xmlfile (from openpyxl)
  Downloading et_xmlfile-1.1.0-py3-none-any.whl.metadata (1.8 kB)
Downloading openpyxl-3.1.5-py2.py3-none-any.whl (250 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 250.9/250.9 kB 2.1 MB/s eta 0:00:00
Downloading et_xmlfile-1.1.0-py3-none-any.whl (4.7 kB)
Installing collected packages: et-xmlfile, openpyxl
Successfully installed et-xmlfile-1.1.0 openpyxl-3.1.5
```

```
import pandas as pd
import numpy as np
from tqdm import tqdm
from keras.preprocessing.text import Tokenizer
tqdm.pandas(desc="progress-bar")
from gensim.models import Doc2Vec
from sklearn import utils
from sklearn.model_selection import train_test_split
from keras.preprocessing.sequence import pad_sequences
import gensim
from sklearn.linear_model import LogisticRegression
from gensim.models.doc2vec import TaggedDocument
import re
import seaborn as sns
import matplotlib.pyplot as plt
from gensim.test.utils import common_texts
from gensim.models import Word2Vec
from random import shuffle # Import shuffle from random
```

```
[5] # Load data from Excel file
    df = pd.read_excel('/content/drive/MyDrive/Glove-CNN/saran.xlsx',sheet_name="Sheet1")  # Replace 'path_to_your_excel_file.xlsx' with your actual file
    df = df[['ULASAN', 'LABEL']]  # Selecting relevant columns
    df = df[pd.notnull(df['ULASAN'])]  # Dropping rows with null 'ULASAN' values
    df.rename(columns={'ULASAN': 'ULASAN'}, inplace=True)  # Rename 'ULASAN' to 'ULASAN' for consistency
```

```
df.head()
```

	ULASAN	LABEL
0	Kedepanya kami bisa diberikan dosen yang lebih...	kritik
1	Saya berharap pelayanan akademik mampu memberi...	kritik
2	Memperbaiki yang baik menjadi lebih baik lagi.	saran
3	Sebaiknya pihak kampus memantau proses penyalu...	kritik
4	Perlunya efesiensi waktu yang baik dalam pembe...	kritik

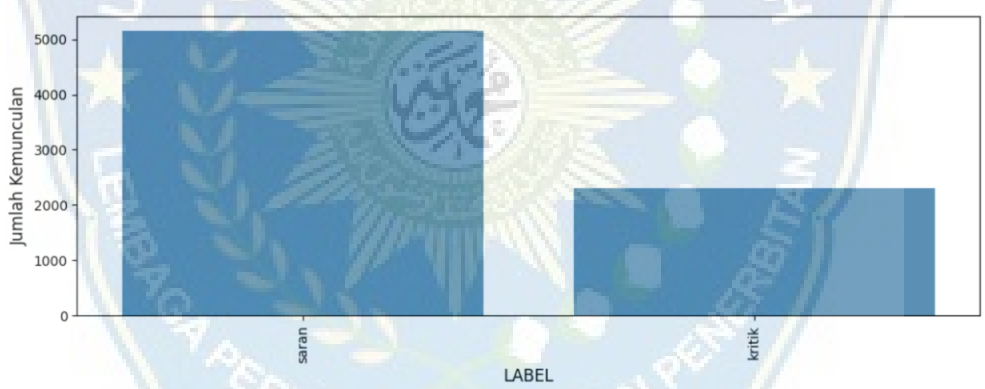
```
[6] df.shape
```

```
(7456, 2)
```

```
[7] # Mengatur ulang indeks baris
    df.index = range(len(df))

    # Menghitung total jumlah kata dalam kolom 'ULASAN'
    total_words = df['ULASAN'].apply(lambda x: len(x.split(' '))).sum()

    print("Total jumlah kata dalam semua ulasan:", total_words)
```

```
Total jumlah kata dalam semua ulasan: 82685
```

```
# Menghitung jumlah kemunculan setiap nilai dalam kolom 'LABEL'
cnt_pro = df['LABEL'].value_counts()

# Menggambar diagram batang menggunakan Seaborn
plt.figure(figsize=(12, 4))
sns.barplot(x=cnt_pro.index, y=cnt_pro.values, alpha=0.8)
plt.ylabel('Jumlah Kemunculan', fontsize=12)
plt.xlabel('LABEL', fontsize=12)
plt.xticks(rotation=90)
plt.show()
```




```
def print_message(index):
    example = df.iloc[index][['ULASAN', 'LABEL']].values
    if len(example) > 0:
        print('ULASAN:', example[0])
        print('LABEL:', example[1])

# Menggunakan fungsi print_message() dengan indeks tertentu
print_message(12)
```

```
ULASAN: semoga dengan adanya peningkatan pelayanan  baik itu dalam proses perkuliahan, pembelajaran ataupun dari pelayanan akademik  akan menjadi jauh
LABEL: saran
```

```
[10] import string
     def remove_punctuation(text):
         return text.translate(str.maketrans('', '', string.punctuation))

     # Menghapus tanda baca dari kolom ULASAN
     df['ULASAN'] = df['ULASAN'].apply(remove_punctuation)
```

```
import nltk
# Download the 'punkt' resource
nltk.download('punkt')

# Tokenisasi teks menggunakan nltk
def tokenize_text(text):
    tokens = []
    for sent in nltk.sent_tokenize(text):
        for word in nltk.word_tokenize(sent):
            if len(word) <= 0:
                continue
            tokens.append(word.lower())
    return tokens

# Memisahkan data menjadi train dan test
train, test = train_test_split(df, test_size=0.1, random_state=0)

# TaggedDocument untuk train dan test set
train_tagged = train.apply(
    lambda r: TaggedDocument(words=tokenize_text(r['ULASAN']), tags=[r.LABEL]), axis=1)
test_tagged = test.apply(
    lambda r: TaggedDocument(words=tokenize_text(r['ULASAN']), tags=[r.LABEL]), axis=1)

# Pengaturan tokenizer
max_features = 500000  # Jumlah maksimum kata yang akan digunakan
max_sequence_length = 50  # Panjang maksimum setiap teks

tokenizer = Tokenizer(num_words=max_features, split=' ', filters='!"#$%&()*+,-./:;<=>?@[\]^_`{|}~', lower=True)
tokenizer.fit_on_texts(df['ULASAN'].values)

# Konversi teks ke dalam urutan angka (sequences)
X_train = tokenizer.texts_to_sequences(train['ULASAN'].values)
X_train = pad_sequences(X_train, maxlen=max_sequence_length)

X_test = tokenizer.texts_to_sequences(test['ULASAN'].values)
X_test = pad_sequences(X_test, maxlen=max_sequence_length)

print('Found %s unique tokens.' % len(tokenizer.word_index))
```

```
[12] # Konversi teks ke dalam urutan angka (sequences)
     X = tokenizer.texts_to_sequences(df['ULASAN'].values)
     X = pad_sequences(X, maxlen=max_sequence_length)

     print('Shape dari data tensor:', X.shape)
```

```
#train_tagged.values
train_tagged.values
```

```
array([TaggedDocument(words=['perlu', 'untuk', 'senantiasa', 'di', 'tingkatkan',
'utamanya', 'dalam', 'hal', 'penggunaan', 'it', 'dari', 'setiap', 'administrasi',
'dosen'], tags=['saran']),
       TaggedDocument(words=['kuliahnya', 'jangan', 'online', 'terus', 'ibu'], tags=
['kritik']),
       TaggedDocument(words=['fasilitas', 'klinik', 'kesehatan', 'di', 'kampus',
'kurang', 'lengkap', 'dan', 'kurang', 'staf', 'medis'], tags=['kritik']),
       ...,
       TaggedDocument(words=['sebaiknya', 'perkuliahan', 'dilaksanakan', 'secara',
'offline', 'memberikan', 'sarana', 'dan', 'prasarana', 'yang', 'mendukung', 'serta',
'meningkatkan', 'kualitas', 'kampus', 'yang', 'baik', 'dari', 'segi', 'pelayanan',
'maupun', 'output', 'yang', 'dihasilkan', 'dari', 'proses', 'akademik', 'kampus'],
tags=['saran']),
       TaggedDocument(words=['lebih', 'baiknya', 'bapak', 'lebih', 'tegas', 'lagi',
'kepada', 'mahasiswa', 'yang', 'tidak', 'bisa', 'berdiskusi'], tags=['saran']),
       TaggedDocument(words=['saran', 'saya', 'mungkin', 'dalam', 'hal',
'pembelajaran', 'kami', 'dipersiapkan', 'fasilitas', 'yang', 'baik', 'guna', 'untuk',
'menyempurnakan', 'proses', 'pembelajaran', 'yang', 'baik', 'dan', 'nyaman'], tags=
['saran'])],
      dtype=object)
```

LAMPIRAN 4 SOURCE CODE GLOVE

```
from gensim.models import KeyedVectors

# Ganti path dengan lokasi file GloVe embeddings Anda
glove_file = '/content/drive/MyDrive/Glove-CNN/glove.6B.50d.txt'

# Muat embeddings GloVe secara manual
glove_model = {}
with open(glove_file, 'r', encoding='utf-8') as f:
    for line in f:
        values = line.split()
        word = values[0]
        coefs = np.asarray(values[1:], dtype='float32')
        glove_model[word] = coefs

# Contoh penggunaan: mendapatkan vektor kata 'the'
print(glove_model['the'])
```

```
[ 4.1800e-01  2.4968e-01 -4.1242e-01  1.2170e-01  3.4527e-01 -4.4457e-02
 -4.9688e-01 -1.7862e-01 -6.6023e-04 -6.5660e-01  2.7843e-01 -1.4767e-01
 -5.5677e-01  1.4658e-01 -9.5095e-03  1.1658e-02  1.0204e-01 -1.2792e-01
 -8.4430e-01 -1.2181e-01 -1.6801e-02 -3.3279e-01 -1.5520e-01 -2.3131e-01
 -1.9181e-01 -1.8823e+00 -7.6746e-01  9.9051e-02 -4.2125e-01 -1.9526e-01
  4.0071e+00 -1.8594e-01 -5.2287e-01 -3.1681e-01  5.9213e-04  7.4449e-03
  1.7778e-01 -1.5897e-01  1.2041e-02 -5.4223e-02 -2.9871e-01 -1.5749e-01
 -3.4758e-01 -4.5637e-02 -4.4251e-01  1.8785e-01  2.7849e-03 -1.8411e-01
 -1.1514e-01 -7.8581e-01]
```

```
[17] from keras.models import Sequential
     from keras.layers import Conv1D, GlobalMaxPooling1D, Dense, Embedding, Dropout
     import numpy as np

     # Definisikan panjang maksimum urutan
     MAX_SEQUENCE_LENGTH = 50

     # Definisikan jumlah kata unik
     num_unique_words = len(tokenizer.word_index) + 1

     # Pastikan bahwa embedding_matrix memiliki bentuk yang sesuai dengan jumlah kata unik dan dimensi embedding
     embedding_dim = 50  # Sesuaikan dengan dimensi GloVe yang Anda gunakan
     embedding_matrix = np.zeros((num_unique_words, embedding_dim))

     # Isi embedding_matrix dengan embeddings GloVe yang sesuai
     for word, i in tokenizer.word_index.items():
         embedding_vector = glove_model.get(word)
         if embedding_vector is not None:
             embedding_matrix[i] = embedding_vector

     # Inisialisasi model Sequential
     model = Sequential()

     # Menambahkan lapisan Embedding dengan bobot yang sesuai
     model.add(Embedding(num_unique_words, embedding_dim, input_length=MAX_SEQUENCE_LENGTH, weights=[embedding_matrix], trainable=True))

     # Menambahkan lapisan Conv1D
     model.add(Conv1D(50, 3, activation='relu'))

     # Menambahkan lapisan Dropout
     model.add(Dropout(0.25))

     # Menambahkan lapisan GlobalMaxPooling1D
     model.add(GlobalMaxPooling1D())

     # Menambahkan lapisan Dense untuk output
     model.add(Dense(2, activation="softmax"))

     # Menampilkan ringkasan model
     model.summary()

     # Kompilasi model
     model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=['acc'])

     # Contoh pemanggilan fungsi split_input
     def split_input(sequence):
         return sequence[:-1], sequence[1:]

     # Contoh penggunaan split_input
     sequence_example = np.array([1, 2, 3, 4, 5])
     x, y = split_input(sequence_example)
     print("Input:", x)
     print("Output:", y)
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 50, 50)            268150

 conv1d_1 (Conv1D)           (None, 48, 50)            7550

 dropout_1 (Dropout)         (None, 48, 50)            0

 global_max_pooling1d_1 (Gl  (None, 50)                0
 obalMaxPooling1D)

 dense_1 (Dense)             (None, 2)                 102

=================================================================
Total params: 275802 (1.05 MB)
Trainable params: 275802 (1.05 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
Input: [1 2 3 4]
Output: [2 3 4 5]
```

```
Y = pd.get_dummies(df['LABEL']).values
from sklearn.model_selection import train_test_split

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.1, random_state=0)

print("Shape of X_train:", X_train.shape)
print("Shape of Y_train:", Y_train.shape)
print("Shape of X_test:", X_test.shape)
print("Shape of Y_test:", Y_test.shape)
```

```
Shape of X_train: (6710, 50)
Shape of Y_train: (6710, 2)
Shape of X_test: (746, 50)
Shape of Y_test: (746, 2)
```

```
poch 1/50

poch 1: val_acc improved from -inf to 0.85523, saving model to /content/drive/MyDrive/Glove-CNN/GLOVE-CNN.h5
usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model
 saving_api.save_model(
10/210 - 2s - loss: 0.4393 - acc: 0.7975 - val_loss: 0.3572 - val_acc: 0.8552 - 2s/epoch - 10ms/step
poch 2/50

poch 2: val_acc improved from 0.85523 to 0.89678, saving model to /content/drive/MyDrive/Glove-CNN/GLOVE-CNN.h5
10/210 - 1s - loss: 0.2776 - acc: 0.8967 - val_loss: 0.2907 - val_acc: 0.8968 - 1s/epoch - 5ms/step
poch 3/50

poch 3: val_acc improved from 0.89678 to 0.90483, saving model to /content/drive/MyDrive/Glove-CNN/GLOVE-CNN.h5
10/210 - 1s - loss: 0.1884 - acc: 0.9350 - val_loss: 0.2500 - val_acc: 0.9048 - 1s/epoch - 5ms/step
poch 4/50

poch 4: val_acc improved from 0.90483 to 0.92225, saving model to /content/drive/MyDrive/Glove-CNN/GLOVE-CNN.h5
10/210 - 1s - loss: 0.1440 - acc: 0.9520 - val_loss: 0.2305 - val_acc: 0.9223 - 1s/epoch - 5ms/step
poch 5/50

poch 5: val_acc did not improve from 0.92225
10/210 - 1s - loss: 0.1145 - acc: 0.9614 - val_loss: 0.2345 - val_acc: 0.9088 - 1s/epoch - 5ms/step
poch 6/50

poch 6: val_acc did not improve from 0.92225
10/210 - 1s - loss: 0.0960 - acc: 0.9680 - val_loss: 0.2345 - val_acc: 0.9169 - 1s/epoch - 5ms/step
poch 7/50

poch 7: val_acc did not improve from 0.92225
10/210 - 1s - loss: 0.0806 - acc: 0.9753 - val_loss: 0.2433 - val_acc: 0.9182 - 1s/epoch - 5ms/step
poch 8/50

poch 8: val_acc did not improve from 0.92225
10/210 - 1s - loss: 0.0687 - acc: 0.9790 - val_loss: 0.2453 - val_acc: 0.9169 - 1s/epoch - 5ms/step
poch 9/50

poch 9: val_acc did not improve from 0.92225
10/210 - 1s - loss: 0.0579 - acc: 0.9817 - val_loss: 0.2585 - val_acc: 0.9115 - 1s/epoch - 5ms/step
poch 10/50

poch 10: val_acc did not improve from 0.92225
10/210 - 1s - loss: 0.0518 - acc: 0.9830 - val_loss: 0.2824 - val_acc: 0.9048 - 1s/epoch - 5ms/step
```

```
Epoch 11: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0432 - acc: 0.9872 - val_loss: 0.2894 - val_acc: 0.9102 - 1s/epoch - 5ms/step
Epoch 12/50

Epoch 12: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0391 - acc: 0.9867 - val_loss: 0.3090 - val_acc: 0.9142 - 1s/epoch - 5ms/step
Epoch 13/50

Epoch 13: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0354 - acc: 0.9890 - val_loss: 0.3220 - val_acc: 0.9048 - 1s/epoch - 5ms/step
Epoch 14/50

Epoch 14: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0307 - acc: 0.9896 - val_loss: 0.3279 - val_acc: 0.9075 - 1s/epoch - 5ms/step
Epoch 15/50

Epoch 15: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0287 - acc: 0.9900 - val_loss: 0.3452 - val_acc: 0.9048 - 1s/epoch - 5ms/step
Epoch 16/50

Epoch 16: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0257 - acc: 0.9914 - val_loss: 0.3618 - val_acc: 0.9062 - 1s/epoch - 5ms/step
Epoch 17/50

Epoch 17: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0249 - acc: 0.9921 - val_loss: 0.3640 - val_acc: 0.9075 - 1s/epoch - 5ms/step
Epoch 18/50

Epoch 18: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0220 - acc: 0.9933 - val_loss: 0.3783 - val_acc: 0.9035 - 1s/epoch - 5ms/step
Epoch 19/50

Epoch 19: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0196 - acc: 0.9930 - val_loss: 0.3953 - val_acc: 0.9035 - 1s/epoch - 5ms/step
Epoch 20/50

Epoch 20: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0221 - acc: 0.9915 - val_loss: 0.3972 - val_acc: 0.9102 - 1s/epoch - 5ms/step
Epoch 21/50

Epoch 21: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0201 - acc: 0.9940 - val_loss: 0.4070 - val_acc: 0.9062 - 1s/epoch - 5ms/step
Epoch 22/50

Epoch 22: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0183 - acc: 0.9942 - val_loss: 0.4153 - val_acc: 0.8928 - 1s/epoch - 5ms/step
Epoch 23/50

Epoch 23: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0175 - acc: 0.9934 - val_loss: 0.4403 - val_acc: 0.9062 - 1s/epoch - 5ms/step
Epoch 24/50
```

```
Epoch 24: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0179 - acc: 0.9939 - val_loss: 0.4455 - val_acc: 0.9035 - 1s/epoch - 5ms/step
Epoch 25/50

Epoch 25: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0179 - acc: 0.9934 - val_loss: 0.4562 - val_acc: 0.8995 - 1s/epoch - 5ms/step
Epoch 26/50

Epoch 26: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0154 - acc: 0.9946 - val_loss: 0.4510 - val_acc: 0.8928 - 1s/epoch - 5ms/step
Epoch 27/50

Epoch 27: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0167 - acc: 0.9943 - val_loss: 0.4516 - val_acc: 0.9021 - 1s/epoch - 5ms/step
Epoch 28/50

Epoch 28: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0127 - acc: 0.9955 - val_loss: 0.4778 - val_acc: 0.8941 - 1s/epoch - 5ms/step
Epoch 29/50

Epoch 29: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0134 - acc: 0.9958 - val_loss: 0.4879 - val_acc: 0.8981 - 1s/epoch - 5ms/step
Epoch 30/50

Epoch 30: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0140 - acc: 0.9957 - val_loss: 0.4964 - val_acc: 0.9008 - 1s/epoch - 5ms/step
Epoch 31/50

Epoch 31: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0125 - acc: 0.9961 - val_loss: 0.4846 - val_acc: 0.8981 - 1s/epoch - 5ms/step
Epoch 32/50

Epoch 32: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0131 - acc: 0.9960 - val_loss: 0.5213 - val_acc: 0.8968 - 1s/epoch - 5ms/step
Epoch 33/50

Epoch 33: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0124 - acc: 0.9957 - val_loss: 0.5209 - val_acc: 0.9035 - 1s/epoch - 5ms/step
Epoch 34/50

Epoch 34: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0123 - acc: 0.9957 - val_loss: 0.5416 - val_acc: 0.8968 - 1s/epoch - 5ms/step
Epoch 35/50

Epoch 35: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0142 - acc: 0.9948 - val_loss: 0.5502 - val_acc: 0.8861 - 1s/epoch - 5ms/step
Epoch 36/50

Epoch 36: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0124 - acc: 0.9952 - val_loss: 0.5584 - val_acc: 0.9008 - 1s/epoch - 5ms/step
Epoch 37/50

Epoch 37: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0137 - acc: 0.9960 - val_loss: 0.5479 - val_acc: 0.8968 - 1s/epoch - 5ms/step
```

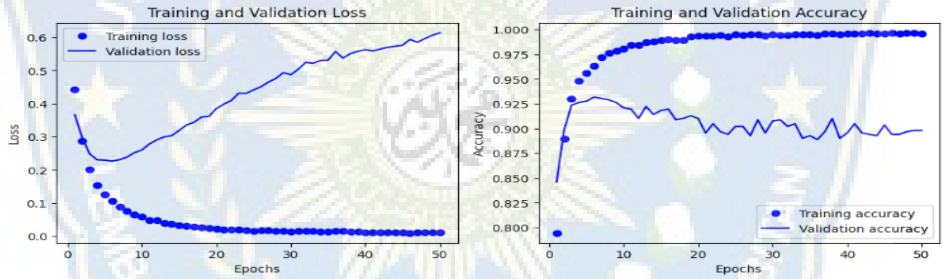
```
Epoch 38: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0106 - acc: 0.9966 - val_loss: 0.5521 - val_acc: 0.8968 - 1s/epoch - 5ms/step
Epoch 39/50

Epoch 39: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0104 - acc: 0.9964 - val_loss: 0.5695 - val_acc: 0.8968 - 1s/epoch - 5ms/step
Epoch 40/50

Epoch 40: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0118 - acc: 0.9966 - val_loss: 0.5540 - val_acc: 0.8941 - 1s/epoch - 5ms/step
Epoch 41/50

Epoch 41: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0113 - acc: 0.9966 - val_loss: 0.5630 - val_acc: 0.9008 - 1s/epoch - 5ms/step
Epoch 42/50

Epoch 42: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0112 - acc: 0.9963 - val_loss: 0.5678 - val_acc: 0.9008 - 1s/epoch - 5ms/step
Epoch 43/50

Epoch 43: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0119 - acc: 0.9960 - val_loss: 0.5790 - val_acc: 0.8981 - 1s/epoch - 5ms/step
Epoch 44/50

Epoch 44: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0096 - acc: 0.9969 - val_loss: 0.5758 - val_acc: 0.8847 - 1s/epoch - 5ms/step
Epoch 45/50

Epoch 45: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0122 - acc: 0.9958 - val_loss: 0.5936 - val_acc: 0.8981 - 1s/epoch - 5ms/step
Epoch 46/50

Epoch 46: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0092 - acc: 0.9967 - val_loss: 0.6048 - val_acc: 0.9008 - 1s/epoch - 5ms/step
Epoch 47/50

Epoch 47: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0111 - acc: 0.9967 - val_loss: 0.6046 - val_acc: 0.8968 - 1s/epoch - 5ms/step
Epoch 48/50

Epoch 48: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0112 - acc: 0.9960 - val_loss: 0.6262 - val_acc: 0.8794 - 1s/epoch - 5ms/step
Epoch 49/50

Epoch 49: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0134 - acc: 0.9951 - val_loss: 0.6070 - val_acc: 0.8901 - 1s/epoch - 5ms/step
Epoch 50/50

Epoch 50: val_acc did not improve from 0.92225
210/210 - 1s - loss: 0.0118 - acc: 0.9966 - val_loss: 0.6100 - val_acc: 0.8954 - 1s/epoch - 5ms/step
```

```python
from keras.callbacks import EarlyStopping, ModelCheckpoint
# Membuat callback EarlyStopping dan ModelCheckpoint
model_checkpoint = ModelCheckpoint('GLOVE-CNN.h5', monitor='val_acc', save_best_only=True, verbose=2)

# Pelatihan model dengan callback
history = model.fit(X_train, Y_train,
                    epochs=50,
                    batch_size=32,
                    validation_data=(X_test, Y_test),
                    callbacks=[model_checkpoint],
                    verbose=2)
# Mendapatkan histori pelatihan
print(history.history.keys())

# Menampilkan val_loss dan val_accuracy
val_loss = history.history['val_loss']
val_acc = history.history['val_acc']
print("Validation Loss:", val_loss)
print("Validation Accuracy:", val_acc)
```

```
[20] # Save the model
     #model.save('/content/drive/MyDrive/Glove-CNN/GLOVE-CNN.h5')
```

```
# Mendapatkan histori pelatihan
history_dict = history.history

# Ekstrak nilai untuk setiap metrik
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
acc_values = history_dict['acc']
val_acc_values = history_dict['val_acc']

# Buat range untuk jumlah epoch
epochs = range(1, len(loss_values) + 1)

# Plot Loss
plt.figure(figsize=(12, 4))

plt.subplot(1, 2, 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

# Plot Accuracy
plt.subplot(1, 2, 2)
plt.plot(epochs, acc_values, 'bo', label='Training accuracy')
plt.plot(epochs, val_acc_values, 'b', label='Validation accuracy')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7bbdb834a7d0>
```



LAMPIRAN 5 SOURCE CODE *WORD2VEC*

```
from gensim.models.doc2vec import Doc2Vec, TaggedDocument

# Ubah ukuran vektor (vector_size) sesuai kebutuhan Anda
vector_size = 20

# Inisialisasi model Doc2Vec
d2v_model = Doc2Vec(dm=1, dm_mean=1, vector_size=vector_size,
window=8, min_count=1, workers=1, alpha=0.065,
min_alpha=0.065)

# Membangun kosakata dari tagged documents pada data pelatihan
train_tagged =
[TaggedDocument(words=tokenize_text(row['ULASAN']),
tags=[row['LABEL']]) for index, row in train.iterrows()]
d2v_model.build_vocab(train_tagged)
```

```python
# Bangun vocab dari train_tagged
d2v_model.build_vocab(train_tagged)

# Latih model
for epoch in range(30):
    d2v_model.train(utils.shuffle(train_tagged),
total_examples=len(train_tagged), epochs=1)
    d2v_model.alpha -= 0.002  # Reduksi alpha setiap epoch
    d2v_model.min_alpha = d2v_model.alpha  # Tetapkan
min_alpha sesuai alpha saat ini
```

```python
print(d2v_model)
```

```
Doc2Vec<dm/m,d20,n5,w8,s0.001>
```

```python
# Mendapatkan jumlah kata dalam kosakata
num_words = len(d2v_model.wv.key_to_index)
print("Jumlah kata dalam kosakata:", num_words)

# Mengakses kata-kata dalam kosakata
words_in_vocab = list(d2v_model.wv.key_to_index.keys())
print("Kata-kata dalam kosakata:", words_in_vocab)
```

```
Jumlah kata dalam kosakata: 5325
Kata-kata dalam kosakata: ['yang', 'lebih', 'mahasiswa', 'dan', 'di', 'dosen', 'tidak', 'untuk', 'dalam', 'pembelajaran', 'baik', 'semoga', 'bisa', 'lagi', 'agar', 'dengan',
```

```python
from keras.models import Sequential
from keras.layers import Conv1D, GlobalMaxPooling1D, Dense, Embedding, Dropout

# Definisikan panjang maksimum urutan
MAX_SEQUENCE_LENGTH = 50

# Definisikan jumlah kata unik
num_unique_words = len(tokenizer.word_index) + 1

# Pastikan bahwa embedding_matrix memiliki bentuk yang sesuai
embedding_matrix = np.random.rand(num_unique_words, 20)

# Inisialisasi model Sequential
model = Sequential()

# Menambahkan lapisan Embedding dengan bobot yang sesuai
model.add(Embedding(num_unique_words, 20, input_length=MAX_SEQUENCE_LENGTH, weights=[embedding_matrix], trainable=True))

# Menambahkan lapisan Conv1D
model.add(Conv1D(50, 3, activation='relu'))

model.add(Dropout(0.5))

# Menambahkan lapisan GlobalMaxPooling1D
model.add(GlobalMaxPooling1D())


# Menambahkan lapisan Dense untuk output
model.add(Dense(2, activation="softmax"))

# Menampilkan ringkasan model
model.summary()

# Kompilasi model
model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=['acc'])

# Contoh pemanggilan fungsi split_input
def split_input(sequence):
    return sequence[:-1], sequence[1:]

# Contoh penggunaan split_input
sequence_example = np.array([1, 2, 3, 4, 5])
x, y = split_input(sequence_example)
print("Input:", x)
print("Output:", y)
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 50, 20)            107260

 conv1d (Conv1D)             (None, 48, 50)            3050

 dropout (Dropout)           (None, 48, 50)            0

 global_max_pooling1d (Glob  (None, 50)                0
 alMaxPooling1D)

 dense (Dense)               (None, 2)                 102

=================================================================
Total params: 110412 (431.30 KB)
Trainable params: 110412 (431.30 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
Input: [1 2 3 4]
Output: [2 3 4 5]
```

```
[ ] Y = pd.get_dummies(df['LABEL']).values
    from sklearn.model_selection import train_test_split

    X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.1, random_state=42)

    print("Shape of X_train:", X_train.shape)
    print("Shape of Y_train:", Y_train.shape)
    print("Shape of X_test:", X_test.shape)
    print("Shape of Y_test:", Y_test.shape)
```

```
Shape of X_train: (6710, 50)
Shape of Y_train: (6710, 2)
Shape of X_test: (746, 50)
Shape of Y_test: (746, 2)
```

```
# Melatih model dengan data validasi
history = model.fit(X_train, Y_train, epochs=50, batch_size=16, verbose=2, validation_data=(X_test, Y_test))

# Mendapatkan histori pelatihan
print(history.history.keys())

# Menampilkan val_loss dan val_accuracy
val_loss = history.history['val_loss']
val_acc = history.history['val_acc']
print("Validation Loss:", val_loss)
print("Validation Accuracy:", val_acc)
```

```
Epoch 1/50
420/420 - 3s - loss: 0.4789 - acc: 0.7820 - val_loss: 0.3648 - val_acc: 0.8780 - 3s/epoch - 6ms/step
Epoch 2/50
420/420 - 2s - loss: 0.2845 - acc: 0.8906 - val_loss: 0.2693 - val_acc: 0.9129 - 2s/epoch - 4ms/step
Epoch 3/50
420/420 - 1s - loss: 0.2192 - acc: 0.9209 - val_loss: 0.2371 - val_acc: 0.9290 - 1s/epoch - 4ms/step
Epoch 4/50
420/420 - 1s - loss: 0.1834 - acc: 0.9362 - val_loss: 0.2225 - val_acc: 0.9316 - 1s/epoch - 3ms/step
Epoch 5/50
420/420 - 1s - loss: 0.1586 - acc: 0.9455 - val_loss: 0.2186 - val_acc: 0.9236 - 1s/epoch - 3ms/step
Epoch 6/50
420/420 - 1s - loss: 0.1484 - acc: 0.9523 - val_loss: 0.2099 - val_acc: 0.9276 - 1s/epoch - 3ms/step
Epoch 7/50
420/420 - 1s - loss: 0.1309 - acc: 0.9532 - val_loss: 0.2171 - val_acc: 0.9155 - 1s/epoch - 3ms/step
Epoch 8/50
420/420 - 1s - loss: 0.1184 - acc: 0.9569 - val_loss: 0.2157 - val_acc: 0.9196 - 1s/epoch - 3ms/step
Epoch 9/50
420/420 - 1s - loss: 0.1084 - acc: 0.9615 - val_loss: 0.2100 - val_acc: 0.9182 - 1s/epoch - 3ms/step
Epoch 10/50
420/420 - 1s - loss: 0.0994 - acc: 0.9671 - val_loss: 0.2097 - val_acc: 0.9236 - 1s/epoch - 4ms/step
Epoch 11/50
420/420 - 1s - loss: 0.0937 - acc: 0.9674 - val_loss: 0.2198 - val_acc: 0.9249 - 1s/epoch - 4ms/step
Epoch 12/50
420/420 - 1s - loss: 0.0808 - acc: 0.9714 - val_loss: 0.2177 - val_acc: 0.9236 - 1s/epoch - 4ms/step
Epoch 13/50
420/420 - 1s - loss: 0.0761 - acc: 0.9735 - val_loss: 0.2290 - val_acc: 0.9223 - 1s/epoch - 3ms/step
Epoch 14/50
420/420 - 1s - loss: 0.0693 - acc: 0.9747 - val_loss: 0.2278 - val_acc: 0.9169 - 1s/epoch - 3ms/step
Epoch 15/50
420/420 - 1s - loss: 0.0684 - acc: 0.9750 - val_loss: 0.2302 - val_acc: 0.9142 - 1s/epoch - 3ms/step
Epoch 16/50
420/420 - 1s - loss: 0.0593 - acc: 0.9796 - val_loss: 0.2353 - val_acc: 0.9196 - 1s/epoch - 3ms/step
Epoch 17/50
420/420 - 1s - loss: 0.0590 - acc: 0.9797 - val_loss: 0.2430 - val_acc: 0.9196 - 1s/epoch - 3ms/step
Epoch 18/50
420/420 - 1s - loss: 0.0570 - acc: 0.9808 - val_loss: 0.2496 - val_acc: 0.9182 - 1s/epoch - 3ms/step
Epoch 19/50
420/420 - 1s - loss: 0.0541 - acc: 0.9794 - val_loss: 0.2477 - val_acc: 0.9155 - 1s/epoch - 4ms/step
Epoch 20/50
420/420 - 1s - loss: 0.0499 - acc: 0.9806 - val_loss: 0.2647 - val_acc: 0.9075 - 1s/epoch - 3ms/step
```

```
Epoch 21/50
420/420 - 1s - loss: 0.0507 - acc: 0.9815 - val_loss: 0.2667 - val_acc: 0.9115 - 1s/epoch - 3ms/step
Epoch 22/50
420/420 - 1s - loss: 0.0429 - acc: 0.9848 - val_loss: 0.2729 - val_acc: 0.9075 - 1s/epoch - 3ms/step
Epoch 23/50
420/420 - 1s - loss: 0.0429 - acc: 0.9833 - val_loss: 0.2841 - val_acc: 0.9088 - 1s/epoch - 3ms/step
Epoch 24/50
420/420 - 1s - loss: 0.0381 - acc: 0.9869 - val_loss: 0.2910 - val_acc: 0.9129 - 1s/epoch - 3ms/step
Epoch 25/50
420/420 - 1s - loss: 0.0406 - acc: 0.9854 - val_loss: 0.3008 - val_acc: 0.9115 - 1s/epoch - 3ms/step
Epoch 26/50
420/420 - 1s - loss: 0.0383 - acc: 0.9858 - val_loss: 0.3104 - val_acc: 0.9115 - 1s/epoch - 3ms/step
Epoch 27/50
420/420 - 1s - loss: 0.0337 - acc: 0.9863 - val_loss: 0.3242 - val_acc: 0.9088 - 1s/epoch - 3ms/step
Epoch 28/50
420/420 - 1s - loss: 0.0346 - acc: 0.9875 - val_loss: 0.3329 - val_acc: 0.9263 - 1s/epoch - 3ms/step
Epoch 29/50
420/420 - 1s - loss: 0.0335 - acc: 0.9867 - val_loss: 0.3272 - val_acc: 0.9155 - 1s/epoch - 3ms/step
Epoch 30/50
420/420 - 1s - loss: 0.0347 - acc: 0.9864 - val_loss: 0.3736 - val_acc: 0.8941 - 1s/epoch - 3ms/step
Epoch 31/50
420/420 - 1s - loss: 0.0330 - acc: 0.9876 - val_loss: 0.3486 - val_acc: 0.9048 - 1s/epoch - 3ms/step
Epoch 32/50
420/420 - 1s - loss: 0.0323 - acc: 0.9890 - val_loss: 0.3612 - val_acc: 0.9088 - 1s/epoch - 3ms/step
Epoch 33/50
420/420 - 1s - loss: 0.0326 - acc: 0.9890 - val_loss: 0.3933 - val_acc: 0.8968 - 1s/epoch - 3ms/step
Epoch 34/50
420/420 - 1s - loss: 0.0280 - acc: 0.9908 - val_loss: 0.3800 - val_acc: 0.9048 - 1s/epoch - 3ms/step
Epoch 35/50
420/420 - 1s - loss: 0.0261 - acc: 0.9917 - val_loss: 0.3807 - val_acc: 0.9142 - 1s/epoch - 3ms/step

Epoch 36/50
420/420 - 1s - loss: 0.0283 - acc: 0.9897 - val_loss: 0.3949 - val_acc: 0.9129 - 1s/epoch - 3ms/step
Epoch 37/50
420/420 - 1s - loss: 0.0269 - acc: 0.9903 - val_loss: 0.4211 - val_acc: 0.9142 - 1s/epoch - 3ms/step
Epoch 38/50
420/420 - 1s - loss: 0.0293 - acc: 0.9887 - val_loss: 0.4023 - val_acc: 0.9088 - 1s/epoch - 3ms/step
Epoch 39/50
420/420 - 1s - loss: 0.0259 - acc: 0.9905 - val_loss: 0.4145 - val_acc: 0.9169 - 1s/epoch - 3ms/step
Epoch 40/50
420/420 - 1s - loss: 0.0247 - acc: 0.9920 - val_loss: 0.4209 - val_acc: 0.9102 - 1s/epoch - 3ms/step
Epoch 41/50
420/420 - 1s - loss: 0.0273 - acc: 0.9906 - val_loss: 0.4204 - val_acc: 0.9142 - 1s/epoch - 3ms/step
Epoch 42/50
420/420 - 1s - loss: 0.0226 - acc: 0.9918 - val_loss: 0.4309 - val_acc: 0.9021 - 1s/epoch - 3ms/step
Epoch 43/50
420/420 - 1s - loss: 0.0242 - acc: 0.9909 - val_loss: 0.4408 - val_acc: 0.9062 - 1s/epoch - 3ms/step
Epoch 44/50
420/420 - 1s - loss: 0.0227 - acc: 0.9918 - val_loss: 0.4371 - val_acc: 0.9021 - 1s/epoch - 3ms/step
Epoch 45/50
420/420 - 1s - loss: 0.0193 - acc: 0.9934 - val_loss: 0.4743 - val_acc: 0.9075 - 1s/epoch - 3ms/step
Epoch 46/50
420/420 - 1s - loss: 0.0180 - acc: 0.9934 - val_loss: 0.4891 - val_acc: 0.9115 - 1s/epoch - 3ms/step
Epoch 47/50
420/420 - 1s - loss: 0.0195 - acc: 0.9921 - val_loss: 0.4794 - val_acc: 0.9075 - 1s/epoch - 3ms/step
Epoch 48/50
420/420 - 1s - loss: 0.0210 - acc: 0.9921 - val_loss: 0.4813 - val_acc: 0.9115 - 1s/epoch - 3ms/step
Epoch 49/50
420/420 - 1s - loss: 0.0203 - acc: 0.9917 - val_loss: 0.4862 - val_acc: 0.9062 - 1s/epoch - 3ms/step
Epoch 50/50
420/420 - 1s - loss: 0.0210 - acc: 0.9921 - val_loss: 0.4948 - val_acc: 0.9075 - 1s/epoch - 3ms/step
```

Lampiran 6 Source code evaluasi model GloVe dan Word2vec

```python
from sklearn.metrics import classification_report, f1_score, precision_score, recall_score

# Melakukan prediksi
predictions = model.predict(X_test)
predicted_labels = (predictions > 0.5).astype(int)  # Konversi probabilitas menjadi label biner (0 atau 1)
true_labels = Y_test

# Menghitung metrik evaluasi tambahan
f1 = f1_score(true_labels, predicted_labels, average='weighted')
precision = precision_score(true_labels, predicted_labels, average='weighted')
recall = recall_score(true_labels, predicted_labels, average='weighted')
print(f"F1 Score: {f1}")
print(f"Precision: {precision}")
print(f"Recall: {recall}")
print(classification_report(true_labels, predicted_labels))

# Tampilkan hasil prediksi dalam array
print("Array hasil prediksi:")
print(true_labels)
print(predicted_labels)
```

Lampiran 7 Hasil Prediksi GloVe dan Word2vec

```
24/24 [==============================] - 0s 2ms/step
F1 Score: 0.8954423592493298
Precision: 0.8954423592493298
Recall: 0.8954423592493298
              precision    recall  f1-score   support

           0       0.83      0.83      0.83       227
           1       0.92      0.92      0.92       519

   micro avg       0.90      0.90      0.90       746
   macro avg       0.88      0.88      0.88       746
weighted avg       0.90      0.90      0.90       746
 samples avg       0.90      0.90      0.90       746

Array hasil prediksi:
[[False  True]
 [False  True]
 [ True False]
 ...
 [ True False]
 [False  True]
 [False  True]]
[[0 1]
 [1 0]
 [1 0]
 ...
 [1 0]
 [0 1]
 [0 1]]
```

```
24/24 [==============================] - 0s 3ms/step
F1 Score: 0.9064990697737997
Precision: 0.9069631084594794
Recall: 0.9061662198391421
              precision    recall  f1-score   support

           0       0.84      0.86      0.85       229
           1       0.94      0.93      0.93       517

   micro avg       0.91      0.91      0.91       746
   macro avg       0.89      0.89      0.89       746
weighted avg       0.91      0.91      0.91       746
 samples avg       0.91      0.91      0.91       746

Array hasil prediksi:
[[ True False]
 [False  True]
 [ True False]
 ...
 [False  True]
 [False  True]
 [ True False]]
[[1 0]
 [0 1]
 [0 1]
 ...
 [0 1]
 [0 1]
 [1 0]]
```

Lampiran 8 Klasifikasi Model GloVe dan Word2vec

```
Panjang Tes Ulasan: 746
Panjang X_test: 746
Panjang Y_test: 746
Panjang true_labels: 746
Panjang predicted_labels: 746
```

```python
import pandas as pd
from sklearn.metrics import classification_report, f1_score, precision_score, recall_score

# Melakukan prediksi
predictions = model.predict(X_test)
predicted_labels = (predictions > 0.5).astype(int)  # Konversi probabilitas menjadi label biner (0 atau 1)
true_labels = Y_test

# Menghitung metrik evaluasi tambahan
f1 = f1_score(true_labels.argmax(axis=1), predicted_labels.argmax(axis=1), average='weighted')
precision = precision_score(true_labels.argmax(axis=1), predicted_labels.argmax(axis=1), average='weighted')
recall = recall_score(true_labels.argmax(axis=1), predicted_labels.argmax(axis=1), average='weighted')
print(f"F1 Score: {f1}")
print(f"Precision: {precision}")
print(f"Recall: {recall}")
print(classification_report(true_labels.argmax(axis=1), predicted_labels.argmax(axis=1)))

# Ambil ulasan, label sebenarnya, dan prediksi label
test_results = pd.DataFrame({
    'Ulasan': test['ULASAN'].values,
    'Label Sebenarnya': true_labels.argmax(axis=1),
    'Prediksi': predicted_labels.argmax(axis=1)
})

# Klasifikasi label 'saran' dan 'kritik' berdasarkan nilai
def classify_label(label):
    return 'saran' if label == 1 else 'kritik'

# Menambahkan kolom klasifikasi label
test_results['Label Sebenarnya'] = test_results['Label Sebenarnya'].apply(classify_label)
test_results['Prediksi'] = test_results['Prediksi'].apply(classify_label)

# Export ke Excel
test_results.to_excel('Hasil Prediksi Glove.xlsx', index=False)

# Tampilkan hasil
print("\nHasil Prediksi:\n", test_results)
print("Data berhasil diekspor ke 'Hasil Prediksi Glove.xlsx'.")
```

```
24/24 [==============================] - 0s 2ms/step
F1 Score: 0.8954423592493298
Precision: 0.8954423592493298
Recall: 0.8954423592493298
              precision    recall  f1-score   support

           0       0.83      0.83      0.83       227
           1       0.92      0.92      0.92       519

    accuracy                           0.90       746
   macro avg       0.88      0.88      0.88       746
weighted avg       0.90      0.90      0.90       746


Hasil Prediksi:
                                                Ulasan Label Sebenarnya  \
0     Lebih maksimal lagi dalam menjelaskan materi a...              saran
1     Terapkan kebijakan ketat mengenai kehadiran te...              saran
2     Prosedur pengurusan beasiswa sangat rumit dan ...             kritik
3     Kurangnya fasilitas untuk kegiatan seni dan bu...             kritik
4     Sarannya dosen diharapkan lebih mengerti terha...              saran
..                                                  ...                ...
741   Dosen sebaiknya mendiversifikasi metode penila...              saran
742   Saran saya sebaiknya dosen di beri arahan untu...              saran
743      Kurangnya fasilitas bagi mahasiswa disabilitas             kritik
744                   Melakukan perkuliahan tatap muka              saran
745   saran saya yaitu lebih dikembangkan lagi dalam...              saran

     Prediksi
0       saran
1      kritik
2      kritik
3      kritik
4       saran
..        ...
741    kritik
742     saran
743    kritik
744     saran
745     saran


24/24 [==============================] - 0s 2ms/step
F1 Score: 0.9069787697010997
Precision: 0.9067560692398866
Recall: 0.9075067024128687
              precision    recall  f1-score   support

           0       0.86      0.83      0.85       229
           1       0.93      0.94      0.93       517

    accuracy                           0.91       746
   macro avg       0.89      0.89      0.89       746
weighted avg       0.91      0.91      0.91       746


Hasil Prediksi:
                                                Ulasan Label Sebenarnya  \
0    Pengurusan berkas sering kali terhambat karena...             kritik
1                          Tetap melayani dengan baik              saran
2                Hadirkan dosen dan siap dalam mengajar             kritik
3    Menyediakan materi kuliah dalam berbagai forma...              saran
4    tabe dok mungkin lebih meningkatkan lagi prose...              saran
..                                                  ...                ...
70   Di berikan arahan mengenai buku yang cocok seb...              saran
71           Semoga lebih ditingkatkan lagi kinerjanya              saran
72   Saran saya agar kedepanya materi di paparkan d...              saran
73   Mahasiswa sering kali harus menunggu lama untu...             kritik
74   Banyak mahasiswa yang mengeluh tentang sulitny...             kritik

    Prediksi
0     kritik
1      saran
2      saran
3      saran
4      saran
..       ...
70     saran
71     saran
72     saran
73    kritik
74    kritik
```

MAJELIS PENDIDIKAN TINGGI PIMPINAN PUSAT MUHAMMADIYAH
UNIVERSITAS MUHAMMADIYAH MAKASSAR
UPT PERPUSTAKAAN DAN PENERBITAN
Alamat kantor: Jl.Sultan Alauddin N0.259 Makassar 90221 Tlp.(0411) 866972,881593, Fax.(0411) 865588

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

SURAT KETERANGAN BEBAS PLAGIAT

UPT Perpustakaan dan Penerbitan Universitas Muhammadiyah Makassar, Menerangkan bahwa mahasiswa yang tersebut namanya di bawah ini:

Nama : Ayu Andira

Nim : 105841108420

Program Studi : Teknik Informatika

Dengan nilai:

No	Bab	Nilai	Ambang Batas
1	Bab 1	10 %	10 %
2	Bab 2	25 %	25 %
3	Bab 3	6 %	10 %
4	Bab 4	9 %	10 %
5	Bab 5	5 %	5 %

Dinyatakan telah lulus cek plagiat yang diadakan oleh UPT- Perpustakaan dan Penerbitan Universitas Muhammadiyah Makassar Menggunakan Aplikasi Turnitin.

Demikian surat keterangan ini diberikan kepada yang bersangkutan untuk dipergunakan seperlunya.

Makassar, 29 Agustus 2024
Mengetahui,

Kepala UPT- Perpustakaan dan Pernerbitan,

Nersiah, S.Hum., M.I.P
NBM. 964 591

Jl. Sultan Alauddin no 259 makassar 90222
Telepon (0411)866972,881 593,fax (0411)865 588
Website: www.library.unismuh.ac.id
E-mail : perpustakaan@unismuh.ac.id

BAB I Ayu Andira 105841108420

by Tahap Tutup

Submission date: 28-Aug-2024 03:26PM (UTC+0700)
Submission ID: 2439628505
File name: BAB_1.doc (31K)
Word count: 777
Character count: 5207

BAB I Ayu Andira 105841108420

ORIGINALITY REPORT

10%	10%	8%	8%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	repository.upi.edu Internet Source	2%
2	siat.ung.ac.id Internet Source	2%
3	repository.ub.ac.id Internet Source	2%
4	ariefhasanudin.blogspot.com Internet Source	2%
5	moam.info Internet Source	2%

Exclude quotes On

Exclude bibliography On

Exclude matches < 2%

UNIVERSITAS MUHAMMADIYAH MAKASSAR LULUS UPT PERPUSTAKAAN turnitin

BAB II Ayu Andira 105841108420

by Tahap Tutup

Submission date: 28-Aug-2024 03:27PM (UTC+0700)
Submission ID: 2439628721
File name: BAB_2_5.doc (145.5K)
Word count: 2593
Character count: 16316

BAB II Ayu Andira 105841108420

ORIGINALITY REPORT

25%	29%	5%	12%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	ejurnal.teknokrat.ac.id Internet Source	5%
2	ejournal.gunadarma.ac.id Internet Source	5%
3	docplayer.info Internet Source	5%
4	Submitted to Sriwijaya University Student Paper	3%
5	jtiik.ub.ac.id Internet Source	3%
6	digilib.unila.ac.id Internet Source	3%
7	Submitted to Universitas Muhammadiyah Makassar Student Paper	3%
8	Submitted to Universitas Brawijaya Student Paper	2%

UNIVERSITAS MUHAMMADIYAH MAKASSAR
LULUS
UPT PERPUSTAKAAN DAN PENERBITAN
turnitin

BAB III Ayu Andira 105841108420

by Tahap Tutup

Submission date: 28-Aug-2024 03:28PM (UTC+0700)
Submission ID: 2439628859
File name: BAB_3_6.doc (93.5K)
Word count: 1033
Character count: 6885

BAB III Ayu Andira 105841108420

ORIGINALITY REPORT

6%	6%	2%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	ejournal.poltektegal.ac.id Internet Source	4%
2	eprintslib.ummgl.ac.id Internet Source	2%

UNIVERSITAS MUHAMMADIYAH MAKASSAR
LULUS
UPT PERPUSTAKAAN DAN PENERBITAN
turnitin

Exclude quotes	On	Exclude matches	< 2%
Exclude bibliography	On		

UNIVERSITAS MUHAMMADIYAH MAKASSAR
LEMBAGA PERPUSTAKAAN DAN PENERBITAN

BAB IV Ayu Andira 105841108420

by Tahap Tutup

Submission date: 28-Aug-2024 03:29PM (UTC+0700)
Submission ID: 2439629063
File name: BAB_4_6.doc (646K)
Word count: 5775
Character count: 36715

BAB IV Ayu Andira 105841108420

ORIGINALITY REPORT

9%	6%	3%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	sofiadutta.github.io Internet Source	1%
2	digilibadmin.unismuh.ac.id Internet Source	1%
3	publikasi.dinus.ac.id Internet Source	1%
4	ejournal.itn.ac.id Internet Source	1%
5	etheses.uin-malang.ac.id Internet Source	<1%
6	python.hotexamples.com Internet Source	<1%
7	Submitted to SUNY, Binghamton Student Paper	<1%
8	repository.its.ac.id Internet Source	<1%
9	Iwa Ovyawan Herlistiono, Sriyani Violina. "Model Prediksi Risiko Stroke Menggunakan Machine Learning", INTECOMS: Journal of	<1%

	Information Technology and Computer Science, 2024 Publication	
10	Angga Herlangga. "PENERAPAN TRANSFER LEARNING EFFICIENTNETB3 UNTUK PENGENALAN SENJATA TRADISIONAL SUMATERA BARAT MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK (CNN)", Jurnal Informatika dan Teknik Elektro Terapan, 2024 Publication	<1 %
11	Riska Amelia, Dwi Budi Santoso. "Prediksi Genre Film Dengan Klasifikasi Multi Kelas Sinopsis Menggunakan Jaringan LSTM", INTECOMS: Journal of Information Technology and Computer Science, 2023 Publication	<1 %
12	Ismail Setiawan, Renata Fina Antika Cahyani, Irfan Sadida. "EXPLORING COMPLEX DECISION TREES: UNVEILING DATA PATTERNS AND OPTIMAL PREDICTIVE POWER", Journal of Innovation And Future Technology (IFTECH), 2023 Publication	<1 %
13	www.gurupenyemangat.com Internet Source	<1 %
14	Submitted to Padjadjaran University Student Paper	<1 %

15	Submitted to Universitas Muhammadiyah Makassar Student Paper	<1 %
16	Submitted to University of Muhammadiyah Malang Student Paper	<1 %
17	cmsab.org Internet Source	<1 %
18	Submitted to Universitas Pertamina Student Paper	<1 %
19	Submitted to Unika Soegijapranata Student Paper	<1 %
20	Submitted to Universiti Kebangsaan Malaysia Student Paper	<1 %
21	fti.uajy.ac.id Internet Source	<1 %
22	Fajar Ferdiawan, Budi Hartono. "DETEKSI SUARA CHORD PIANO MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK", Jurnal Informatika dan Rekayasa Elektronik, 2022 Publication	<1 %
23	journal.lembagakita.org Internet Source	<1 %
24	www.researchgate.net Internet Source	<1 %

25	Hafiz Irsyad, Akhsani Taqwiym. "Community Analysis Sentiment Against Palestinian People with Naive Bayes Classification", JTECS : Jurnal Sistem Telekomunikasi Elektronika Sistem Kontrol Power Sistem dan Komputer, 2021 Publication	<1 %
26	ichi.pro Internet Source	<1 %
27	repository.ub.ac.id Internet Source	<1 %
28	www.scribd.com Internet Source	<1 %

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

UNIVERSITAS MUHAMMADIYAH MAKASSAR
LEMBAGA PERPUSTAKAAN DAN PENERBITAN

BAB V Ayu Andira 105841108420

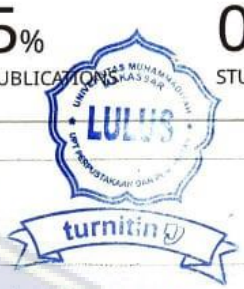
by Tahap Tutup

Submission date: 28-Aug-2024 03:29PM (UTC+0700)
Submission ID: 2439629274
File name: BAB_5_5.doc (18K)
Word count: 313
Character count: 2041

BAB V Ayu Andira 105841108420

ORIGINALITY REPORT

5%	3%	5%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS



PRIMARY SOURCES

1	repository.trisakti.ac.id Internet Source	3%
2	Arif Zainudin. "Analisis Indeks Kepuasan Masyarakat Badan Pelayanan Perizinan Terpadu Kota Tegal", Jurnal Ilmu Pemerintahan : Kajian Ilmu Pemerintahan dan Politik Daerah, 2016 Publication	2%

Exclude quotes
Exclude bibliography

Exclude matches