PENERAPAN *WORD EMBEDDING FASTTEXT* DALAM ANALISIS SENTIMEN REVIEW APLIKASI JAKI MENGGUNAKAN METODE *CNN*

SKRIPSI

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

OLEH :

ARYO DININGRAT SALEA
105841108820

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH MAKASSAR
2024

PENERAPAN *WORD EMBEDDING FASTTEXT* DALAM ANALISIS SENTIMEN REVIEW APLIKASI JAKI MENGGUNAKAN METODE *CNN*

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

Disusun Dan Diajukan Oleh :

ARYO DININGRAT SALEA
105841108820

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH 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 Aryo Diningrat Salea dengan nomor induk Mahasiswa 105 84 11088 20, dinyatakan diterima dan disahkan oleh Panitia Ujian Tugas Akhir/Skripsi sesuai dengan Surat Keputusan Dekan Fakultas Teknik Universitas Muhammadiyah Makassar Nomor : 227/05/A.5-VI/VII/46/2024, sebagai salah satu syarat guna memperoleh gelar Sarjana Komputer pada Program Studi Informatika Fakultas Teknik Universitas Muhammadiyah Makassar pada hari Sabtu tanggal 31 Agustus 2024.

Makassar, 26 Safar 1446 H / 31 Agustus 2024 M

Panitia Ujian :

1. Pengawas Umum
   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 : Titin Wahyuni, S.Pd., M.T. .....
3. Anggota : 1. Rizki Yusliana Bakti, ST., M.T. : .....
   2. Lukman Anas, S.Kom., M.T. : .....
   3. Desi Anggreani, S.Kom., MT. : .....

Mengetahui :

Pembimbing I

Muhyiddin A.M Hayat, S.Kom., M.T

Pembimbing II

Fahrim Irhamna Rachman, S.Kom., M.T

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 : **PENERAPAN WORD EMBEDDING FAST TEXT DALAM ANALISIS SENTIMEN REVIEW APLIKASI JAKI MENGGUNAKAN METODE CNN**

Nama : Aryo Diningrat Salea
Stambuk : 105 84 11088 20

Makassar, 31 Agustus 2024

Telah Diperiksa dan Disetujui
Oleh Dosen Pembimbing;

Pembimbing I

Muhyiddin A.M Hayat, S.Kom., M.T

Pembimbing II

Fahrim Irhamna Rachman, S.Kom., M.T

Mengetahui,
Ketua Program Studi Arsitektur

Muhyiddin A.M Hayat, S.Kom., M.T
NBM : 604 577

MOTTO DAN PERSEMBAHAN

Motto

"Kerjakanlah hari ini sebaik mungkin, karena ia tidak akan terulang."

Persembahan

Karya ini merupakan wujud rasa syukur kepada Allah SWT. atas segala nikmat yang tak terhingga, sehingga penulis dapat meneyelesaikan skripsi ini.

Terciptanya skripsi ini tidak lain dan tidak bukan merupakan bantuan dan dorongan penyemangat oleh orang-orang yang terkasih. Kepada kedua orang tua dan saudari saya, Ayah saya Marsel Y. Salea, yang senantiasa bekerja keras dan mendukung agar memastikan anaknya mendapatkan pendidikan yang terbaik. Ibu saya Hasna, yang senantiasa melangitkan setiap do'a dan dukungan agar tercapainya cita-cita, menjadikan saya sebagai seorang sarjana pertama dalam keluarga. Saudari saya Marsya M. Salea, yang mendukung dan memberikan semangat disetiap proses penyelesaian skripsi ini.

Ucapan terima kasih saya ucapkan kepada semua pihak yang telah membantu keberhasilan pencapaian ini, kepada Kepala Prodi Informatika dan para dosen Prodi Informatika, serta seluruh staf dan segenap civitas akademika Fakultas Teknik Universitas Muhammadiyah Makassar saya ucapkan banyak terima kasih.

Rasa bangga yang tak terhingga kepada diri sendiri yang telah kuat bertahan ditengah rasa ketidakyakinan serta kecemasan akan ketertinggalan membuat saya harus percaya bahwa saya mampu untuk menyelesaikan tugas akhir ini. Kepada seluruh teman-teman seperjuangan, Informatika Angkatan 2020 saya ucapkan banyak terima kasih karena telah saling mendukung dan membantu sehingga pada kesempatan ini kita semua dapat bersama-sama menyelesaikan studi di tahun ini. Teruntuk orang-orang yang berada di barisan belakang, segenap keluarga, kerabat, teman maupun sahabat dan seluruh pihak yang tidak dapat saya sebutkan satu persatu saya ucapkan banyak-banyak terima kasih karena telah memberi semangat, do'a dan dukungannya.

ABSTRAK

ARYO DININGRAT SALEA, Penerapan *Word Embedding Fasttext* Dalam Analisis Sentimen Review Aplikasi Jaki Menggunakan Metode *CNN (dibimbing oleh* Muhydin A M Hayat, S.Kom.,MT dan Fachrim Irhamna Rachman, S.Kom., M.T).

Penelitian ini bertujuan untuk menguji tingkat akurasi kinerja model teknik *FastText* sebagai *Word Embedding* terhadap metode *CNN* dalam memproses data teks, khususnya pada ulasan aplikasi JAKI. Penelitian ini juga ingin mengetahui sejauh mana pengaruh metode CNN dalam menganalisis sentimen. Dataset yang digunakan berjumlah 4.455 data, namun dilakukan penghapusan atribut seperti *reviewId, userName, userImage, score, reviewCreatedVersion, replyContent, dan appVersion*. Setelah penghapusan atribut, data yang tersisa sebanyak 3.199 ulasan. Proses *preprocessing* data dilakukan dengan melibatkan penghapusan tanda baca *(punctuation)* dan tokenisasi *(tokenizing)*. Dataset sebanyak 3.199 ulasan ini diuji dengan tiga kategori sentimen yaitu positif, negatif, dan netral. Hasil pengujian menunjukkan bahwa kombinasi "*Word Embedding FastText – CNN*" efektif dalam menganalisis sentimen. Model yang menggunakan kombinasi ini mencapai nilai akurasi yang tinggi antara 91% hingga 100%, serta menunjukkan nilai *loss* yang lebih rendah dan stabil secara konsisten.

Kata Kunci: Analisis Sentimen, Aplikasi Jaki, *Convolutional Neural Network*, *Word Embedding Fasttext*.

ABSTRACT

ARYO DININGRAT SALEA, *Application of Word Embedding Fasttext in Sentiment Analysis Review of the Jaki Application Using the CNN Method (supervised by Muhydin A M Hayat, S.Kom., MT and Fachrim Irhamna Rachman, S.Kom., M.T).*

This research aims to test the level of accuracy of the performance of the FastText engineering model as Word Embedding against the CNN method in processing text data, especially in JAKI application reviews. This study also wants to know the extent of the influence of the CNN method in analyzing sentiment. The dataset used was 4,455 data, but attributes such as reviewId, userName, userImage, score, reviewCreatedVersion, replyContent, and appVersion were removed. After attribute removal, the remaining data was 3,199 reviews. The data preprocessing process involves the removal of punctuation and tokenization. This dataset of 3,199 reviews was tested with three sentiment categories, namely positive, negative, and neutral. The test results show that the combination of Word Embedding FastText – CNN is effective in analyzing sentiment. Models that use this combination achieve high accuracy values between 91% and 100%, and show consistently lower and stable loss values.

Keywords: *Sentiment Analysis, Jaki Application, Convolutional Neural Network, Word Embedding Fasttext.*

KATA PENGANTAR

Segala puji bagi Allah Subhanahu Wa Ta'ala yang telah melimpahkan rahmat dan petunjuk-Nya kepada penulis. Sholawat dan salam semoga tercurahkan kepada baginda Nabi Muhammad Shallallahu`alaihi Wa Sallam, sosok revolusioner sejati yang menjadi teladan bagi seluruh umat, menyebarkan Islam hingga saat ini kita masih merasakan berkahnya sebagai seorang Muslim. Dengan berkat-Nya, penulis berhasil menyelesaikan skripsi dengan judul. **"Penerapan *Word Embedding FastText* Dalam Analisis Sentimen Review Aplikasi JAKI Menggunakan Metode *CNN*"**.

Penulisan skripsi ini disusun oleh penulis sebagai bagian dari persyaratan untuk menyelesaikan Program Sarjana (S1) di Program Studi Informatika Fakultas Teknik Universitas Muhammadiyah Makassar. Penulis berharap bahwa dengan adanya skripsi ini, dapat memberikan tambahan referensi bagi para pembaca, terutama mahasiswa informatika, dan secara umum, bagi kalangan masyarakat pada umumnya. Penulis menyadari bahwa dalam proses penyusunan skripsi ini melibatkan banyak pihak. Oleh karena itu, pada kesempatan ini, penulis ingin menyampaikan rasa terima kasih yang besar kepada:

1. Ibunda **Hasna**, cinta pertama sekaligus pintu surgaku. Serta kepada Ayahanda **Marsel Y Salea**, beliau yang menjadi panutan dalam memimpin keluarga. Walaupun keduanya tidak sempat merasakan pendidikan hingga bangku perkuliahan mereka senantiasa memberikan kasih sayang, bimbingan, dukungan materi, dan doa yang selalu dilangitkan untuk kesuksesan penulis. Pengorbanan yang tak ternilai dari keduanya telah menjadi pendorong utama sehingga penulis dapat menyelesaikan perjalanan studi hingga tingkat sarjana.
2. Bapak **Dr. Ir. H. Abd. Rakhim Nanda, S.T., M.T., IPU**, sebagai Rektor Perguruan Tinggi Universitas Muhammadiyah Makassar.
3. Ibu **Dr. Hj. Ir. Nurnawaty, ST., MT**, selaku Dekan Fakultas Teknik Universitas Muhammadiyah Makassar.

4. Bapak **Muhydin A. M Hayat, S.Kom., M.T**, Selaku Ketua Prodi Informatika, Teknik Universitas Muhammadiyah Makassar.
5. Bapak **Muhydin A. M Hayat, S.Kom., M.T**, selaku Dosen Pembimbing I dan Bapak **Fahrim Irhamna Rahman, S.Kom., M.T**, selaku Dosen Pembimbing II yang senantiasa meluangkan waktu dan pikirannya untuk membimbing dan mengarahkan penulis dalam penyusunan skripsi ini.
6. Seluruh Dosen dan Staf Fakultas Teknik Universitas Muhammadiyah Makassar.

Semoga Tuhan Yang Maha Esa memberikan ganjaran yang lebih besar kepada beliau, sebagai akhir dari segala ucapan. Harapannya, skripsi ini dapat memberikan manfaat bagi pembaca secara umum dan khususnya bagi penulis.

Makassar, 10 September 2024

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

DAFTAR ISTILAH

Epoch	*Epoch* adalah periode atau titik waktu tertentu yang digunakan sebagai referensi dalam berbagai konteks, seperti dalam pembelajaran mesin atau astronomi.
API	*API* (Application Programming Interface) adalah serangkaian aturan dan alat yang memungkinkan satu aplikasi berkomunikasi dengan aplikasi lain. Ini memungkinkan perangkat lunak yang berbeda untuk berinteraksi dan saling bertukar data.
Feed Forward Neural Network	*Feed Forward Neural Network (FFNN)* adalah jenis jaringan saraf tiruan di mana informasi bergerak dalam satu arah dari *input*, melalui lapisan tersembunyi, hingga *output* tanpa *looping* atau umpan balik.
Accuracy	*Accuracy* adalah ukuran kinerja model yang menunjukkan persentase prediksi yang benar dari total prediksi yang dibuat. Ini mengindikasikan seberapa baik model tersebut dalam mengklasifikasikan data dengan benar.
Loss	*Loss* adalah ukuran seberapa jauh prediksi model dari nilai yang sebenarnya. Ini digunakan untuk menilai kinerja model dalam pembelajaran mesin; semakin kecil nilai loss, semakin baik model tersebut dalam memprediksi hasil yang benar.
Validation Accuracy	*Validation Accuracy* adalah ukuran kinerja model yang menunjukkan persentase prediksi yang benar pada dataset validasi, yang digunakan untuk mengevaluasi model selama proses pelatihan tanpa

	mempengaruhi model itu sendiri.
Validation Loss	*Validation Loss* adalah ukuran kesalahan model pada dataset validasi yang digunakan untuk mengevaluasi kinerja model selama pelatihan. Ini membantu menentukan seberapa baik model dapat memprediksi data yang tidak terlihat dan digunakan untuk mendeteksi *overfitting*.
Overfitting	*Overfitting* adalah kondisi di mana model *machine learning* terlalu menyesuaikan diri dengan data latihannya sehingga berkinerja buruk pada data baru atau tidak terlihat. Model ini menangkap *noise* atau detail spesifik dari data latihan, yang tidak berlaku untuk data lain.
Underfitting	*Underfitting* adalah kondisi di mana model *machine learning* terlalu sederhana sehingga gagal menangkap pola dasar dalam data latihan, yang mengakibatkan kinerja yang buruk baik pada data latihan maupun data baru.
Bag Of Words	*Bag of Words (BoW)* adalah metode representasi teks yang mengubah dokumen menjadi vektor berdasarkan frekuensi kemunculan kata-kata tanpa memperhatikan urutan kata.
NLP	*NLP (Natural Language Processing)* adalah cabang kecerdasan buatan yang fokus pada interaksi antara komputer dan bahasa manusia.
Word2Vec	*Word2Vec* adalah metode untuk mengubah kata-kata menjadi vektor numerik dalam ruang berdimensi tinggi, sehingga kata-kata yang memiliki makna serupa berada dekat satu sama lain dalam ruang vektor.

Glove	*GloVe (Global Vectors for Word Representation)* adalah metode untuk mengubah kata-kata menjadi vektor numerik berdasarkan statistik global dari korpus teks berdasarkan frekuensi kemunculan kata-kata dalam konteksnya.
Opinion Mining	*Opinion Mining*, juga dikenal sebagai analisis sentimen, adalah proses untuk mengidentifikasi dan mengevaluasi opini, perasaan, atau sikap dalam teks, seperti ulasan atau komentar.
Python	*Python* adalah bahasa pemrograman tingkat tinggi yang mudah dibaca dan digunakan, dikenal karena sintaksisnya yang sederhana dan fleksibilitasnya.
OOV	*OOV (Out Of Vocabulary)* merujuk pada kata-kata atau istilah yang tidak ada dalam kosakata yang telah ditetapkan atau dikenal oleh model atau sistem pemrosesan bahasa alami.
Deep Learning	*Deep Learning* adalah cabang dari pembelajaran mesin yang menggunakan jaringan saraf tiruan dengan banyak lapisan untuk menganalisis data dan belajar dari pola kompleks seperti gambar, suara, atau teks.
n-gram	*n-gram* adalah sebuah teknik dalam pemrosesan bahasa alami yang memecah teks menjadi urutan kata atau karakter berukuran (n). Misalnya, dalam *bigram* (n=2), teks dipecah menjadi pasangan kata, sementara dalam *trigram* (n=3), dipecah menjadi *triplet* kata.
Skip-gram	*Skip-gram* adalah model dalam *Word2Vec* yang digunakan untuk mempelajari representasi kata dengan memprediksi kata-kata konteks (kata-kata di sekitar) dari kata target.

Softmax	*Softmax* adalah fungsi matematika yang mengubah vektor nilai menjadi probabilitas distribusi. Ini sering digunakan di jaringan saraf untuk mengubah output lapisan terakhir menjadi probabilitas yang menjumlahkan hingga 1, memungkinkan model untuk membuat prediksi kelas dalam klasifikasi multi-kelas.
ReLu	*ReLU (Rectified Linear Unit)* adalah fungsi aktivasi yang digunakan dalam jaringan saraf yang mengubah input menjadi output dengan mengabaikan nilai negatif dan hanya meneruskan nilai positif.
Scraping	*Scraping* adalah proses otomatis untuk mengambil data dari situs web. Ini melibatkan pengambilan konten halaman web dan ekstraksi informasi yang diinginkan, sering kali menggunakan alat atau skrip khusus.
Confussion Matrix	*Confusion matrix* adalah tabel yang digunakan untuk mengevaluasi kinerja model klasifikasi dengan menunjukkan jumlah prediksi yang benar dan salah untuk setiap kelas dalam membantu menganalisis kesalahan model.

BAB I
PENDAHULUAN

A. Latar Belakang

Analisis sentimen muncul sebagai alat penting untuk memahami dan menginterpretasi perasaan atau sikap yang digunakan dalam tulisan. Dalam era digital saat ini, volume data tekstual yang dihasilkan oleh pengguna melalui media sosial, ulasan produk, blog dan *platform* lainnya terus meningkat.

Analisis sentimen telah menarik perhatian dalam beberapa dekade terakhir dalam penelitian di bidang *Natural Language Processing*. Analisis sentimen menjadi proses untuk mengidentifikasi atau mengkategorikan opini pengguna dalam bentuk teks terhadap berbagai hal seperti film, produk, acara, dan lainnya, baik itu positif, negatif, maupun netral. Analisis sentimen sangat berguna untuk mengidentifikasi, mengekstraksi, dan mempelajari informasi subjektif tentang produk suatu perusahaan. Terkadang, perusahaan membutuhkan wawasan yang lebih rinci mengenai sentimen terhadap produk mereka, seperti aspek apa saja dari produk yang perlu dievaluasi**.** (Muhammad Afif Raihan & Erwin Budi Setiawan, 2022)

Pemerintah provinsi DKI Jakarta melalui Badan Layanan Umum Daerah (BLUD) Jakarta *Smart City*, telah meluncurkan sebuah inovasi dalam bentuk aplikasi super yang dikenal dengan sebutan JAKI (Jakarta Kini). Aplikasi JAKI ini berfungsi sebagai pusat resmi layanan informasi pemerintah Provinsi DKI Jakarta yang dirancang guna memenuhi kebutuhan harian masyarakat. (Putu Sawitra Danda Prasetia et al., 2024). Aplikasi JAKI memiliki rating 3,1-3,8 pada skala 1 hingga 5 pada *platform AppStore* dan *PlayStore*. Meskipun aplikasi ini mendapatkan sejumlah ulasan dan peringkat yang cukup baik, namun masih ada beberapa pengguna yang memberikan ulasan buruk. (Rodhi et al., 2022). Maka dengan penulisan skripsi ini diharapkan mampu menjadi rujukan evaluasi terhadap sikap pengguna aplikasi JAKI.

Sekitar tahun 2000, teknik *Word Embedding* mulai dikembangkan dimana *Word Embedding* itu sendiri memetakan setiap kata dalam dokumen kedalam vektor padat, dimana setiap vektor mewakili proyeksi kata dalam dimensi vektor. Metode *Word Embedding* ini menjadi sangat penting dalam proses analisis sentimen karena menyediakan representasi kata yang lebih baik. *Word Embedding* terbagi atas beberapa metode diantaranya *Word2Vec, Glove* dan *FastText*. Salah satu kelebihan dari *Word Embedding* adalah tidak memerlukan anotasi/label dan dapat dihasilkan langsung dari korpus yang tidak teranotasi sehingga waktu pelatihan data dapat lebih cepat, (Nurdin et al., 2020). *Convolutional Neural Network* termasuk salah satu jenis jaringan saraf yang ditemukan dalam deep learning. *CNN* dikenal sebagai *Feed Forward Neural Network* yang dirancang khusus untuk menangani data yang dipecah menjadi bagian-bagian terpisah. (Yuniarossy et al., 2024).

Penelitian ini diharapkan mampu menerapkan dan menguji efektivitas kombinasi antara *Word Embedding FastText* dan *CNN* dalam analisis sentimen review aplikasi JAKI. Maka dengan dilakukannya pendekatan tersebut dapat meningkatkan *accuracy* klasifikasi sentimen dan memberikan dasar yang kuat kepada pengembang untuk perbaikan layanan aplikasi.

B. Rumusan Masalah

Sehubungan dengan latar belakang yang telah dijelaskan diatas, rumusan masalah yang diangkat dalam penelitian ini adalah terbagi atas dua poin yaitu :

1. Menguji tingkat akurasi kinerja model kombinasi antara *Word Embedding FastText* dan *CNN* dalam memproses data teks khususnya pada ulasan aplikasi JAKI.
2. Bagaimana pengaruh metode *CNN* dalam memproses dan mengkategorikan analisis sentimen.

C. Tujuan Penelitian

1. Adapun tujuan dari penelitian ini ialah untuk menguji tingkat keakuratan teknik *FastText* sebagai *Word Embedding* terhadap *accuracy* model *CNN* dalam memproses data teks, khususnya pada ulasan aplikasi JAKI.
2. Seberapa baik pengaruh *CNN* dalam berkontribusi meningkatkan kinerja model pada proses mengkategorikan sentimen ulasan.

D. Manfaat Penelitian

Diharapkan penelitian ini mampu memberikan kontribusi manfaat baik secara teoritis maupun praktis adalah :

1. Secara Teoritis
   a. Untuk pengembangan ilmu pengetahuan terutama dalam bidang teknik informatika.
   b. Memberikan wawasan dan literatur ilmiah dalam bidang analisis sentimen dan pengolahan bahasa alami, khususnya pada penggunaan *Word Embedding*.
2. Secara Praktis
   a. Bagi Peneliti
      1) Salah satu persyaratan yang harus dipenuhi untuk menyelesaikan program S1.
      2) Mendapatkan pengalaman praktis dalam memahami penggunaan teknik *Word Embedding* dan pengembangan model *CNN* terhadap analisis sentimen.
   b. Bagi Universitas
      1) Sebagai bahan referensi untuk penelitian yang akan dilakukan di masa mendatang.
      2) Sebagai bahan evaluasi bagi universitas dalam mengembangkan keilmuan, dalam hal ini yang berkaitan dengan penerapan *Word Embedding* dan pengembangan *CNN* dalam memproses data teks dan pengolahan bahasa alami.

E. Ruang Lingkup Penelitian

1) Studi kasus pada penelitian ini terbatas pada analisis sentimen terkait ulasan pengguna aplikasi JAKI.
2) Dataset yang digunakan hanya diperoleh dari ulasan pada *platform* Google *PlayStore.*
3) Penerapan *Word Embedding* terbatas pada penggunaan model *FastText* saja tanpa mempertimbangkan model lain.
4) Terbatas hanya menggunakan metode *Convolutional Neural Network* (*CNN*).

F. Sistematika Penulisan

Secara garis besar penulisan laporan tugas akhir ini terbagi menjadi beberapa bab yaitu :

BAB I PENDAHULUAN

Bab ini menjelaskan tentang latar belakang masalah, rumusan masalah, tujuan, manfaat, batasan masalah dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini menjelaskan tentang teori-teori yang melandasi penulisan dalam melaksanakan skripsi.

BAB III METODE PENELITIAN

Membahas tentang metode penelitian dan alat yang digunakan untuk pembuatan sistem.

BAB II
TINJAUAN PUSTAKA

A. Landasan Teori

1. Analisis Sentimen

Sentimen analisis juga dikenal sebagai *opinion mining*, adalah studi komputasi yang bertujuan untuk mengidentifikasi dan mengungkapkan opini, sentimen, evaluasi, sikap, emosi, subjektivitas, penilaian atau pandangan terhadap sebuah kata/teks, (Manggopa et al., 2024). Dimana hasil dari opini tersebut bisa berupa sentimen positif, negatif ataupun netral. Maka analisis sentimen dapat disimpulkan sebagai pengambil keputusan terhadap pendapat dari satu individu atau lebih mengenai objek atau topik tertentu, (Mustasaruddin et al., 2023).

Analisis sentimen mulai menjadi isu yang penting seiring dengan meningkatnya interaksi pengguna di media sosial, penggunaan berbagai forum dan blog, serta komentar dan ulasan/*review* di berbagai aplikasi atau situs *e-commerce*, (Jihad et al., 2021). Analisis sentimen juga sangat terkait dengan *Natural Language Processing*, yang merupakan kemampuan mesin untuk memahami bahasa manusia melalui penggunaan algoritma komputer, matematika dan linguistik komputasi, (Vidya Chandradev et al., 2023).

2. Aplikasi JAKI (Jakarta Kini)

Aplikasi merupakan perangkat lunak yang dirancang untuk membantu dan mempermudah kinerja manusia dalam menyelesaikan tugas-tugas tertentu serta dalam berinteraksi dengan orang lainnya, (Mustasaruddin et al., 2023). Pada tahun 2019, melalui unit pengelola Jakarta *Smart City* meluncurkan sebuah aplikasi layanan informasi resmi pemprov DKI Jakarta yang diberi nama Jakarta Kini (JAKI), (Rodhi et al., 2022).

JAKI adalah aplikasi kota pertama yang menerapkan konsep kota pintar dengan memanfaatkan teknologi informasi dan komunikasi untuk mengidentifikasi, menganalisis dan mengelola berbagai data secara efisien dan efektif, (Andriyanto et al., 2021). Seperti aplikasi bisnis *start-up*, JAKI mengitegrasikan berbagai layanan dari OPD dan BUMD kedalam satu aplikasi melalui beberapa fitur, sehingga memudahkan masyarakat dalam memenuhi kebutuhan sehari-hari, (Putu Sawitra Danda Prasetia et al., 2024). Fitur-fitur tersebut diantaranya ialah JakWarta, JakRespons, JakPangan, JakPantau, JakSiaga, JakWifi, Jejak, JakPenda, JakSurvei, JAKISPU, dan LaporVideo, (Sofiana, 2023).

3. Word Embedding

Word Embedding di artikan sebagai metode penyisipan kata yang mengubah kata menjadi vektor kontinu dengan panjang yang sudah ditentukan, sehingga tidak dibatasi oleh jumlah kosakata yang besar, (Rahman et al., 2021). Metode ini mendapatkan sorotan khusus dalam klasifikasi teks dan analisis sentimen karena kemampuannya menangkap keterkaitan sintaksis dan semantik antar kata, (Daniati & Utama, 2023).

Dari berbagai metode dalam *Word Embedding* seperti *Word2Vec*, *Glove* dan *FastText*. Metode *FastText* menjadi salah satu yang memiliki kinerja sangat baik dalam merepresentasikan kata dan mengklasifikan kalimat lalu mengubahnya menjadi vektor, (Nanda et al., 2023). *FastText* merupakan pengembangan dari *Word2Vec* dimana metode ini mampu mengatasi masalah *Out Of Vocabulary* (OOV) yang tidak dapat ditangani oleh *Word2Vec* dan *Glove*. Dengan menggunakan *n-gram* dalam model *skip-gram*, *FastText* menerapkan representasi kata melalui informasi sub-kata, memungkinkan penangkapan kata-kata pendek serta memahami awalan dan akhiran kata, (Riza & Charibaldi, 2021).

4. Deep Learning

Deep Learning dipandang sebagai kombinasi antara machine learning dengan AI (*Artifical Neural Network*) yang mengembangkan jaringan saraf *multiple layer*. Beberapa algoritma termasuk dalam kategori *Deep Learning* antara lain, (Nugroho et al., 2020) :

a. *Convolutional Networks*
b. *Restricted Boltzmann Machinee (RBM)*
c. *Deep Beliefe Networks (DBN)*
d. *Stacked Authocodersx*

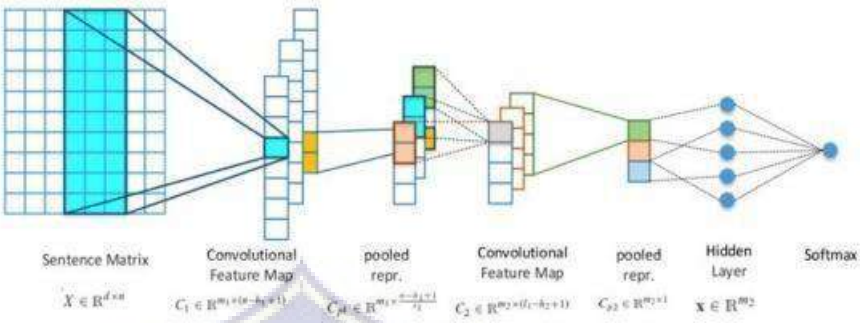
Deep Learning juga berfungsi sebagai metode yang bisa digunakan untuk menerapkan analisis sentimen. Proses kerja *Deep Learning* melibatkan ekstraksi data menggunakan *neural network*, kemudian model tersebut belajar dari nilai *error* yang dihasilkan, (Alghifari et al., 2022).

5. Convolutional Neural Network

Kunihiko Fukushima, seorang peneliti dari NHK *Broadcasting Science Research Laboratories* menjadi orang pertama yang mengembangkan konsep *CNN* kemudian konsep tersebut diperkuat oleh Yaan LeChun dalam penelitiannya, (Azhar et al., 2021). *CNN* adalah neural network yang dirancang untuk mengolah data dua dimensi dan merupakan pengembangan dari *Multilayer Perceptron* (MLP) yang termasuk dalam *neural network* bertipe *feed forward* (bukan berulang). Karena kedalaman jaringannya yang tinggi, *CNN* banyak digunakan pada data citra dan termasuk dalam kategori *Deep Neural Networks*, (Nugroho et al., 2020).

Pada awalnya, *CNN* digunakan dalam bidang pemrosesan gambar atau visi komputer. Namun, pada tahun 2015, *CNN* mulai digunakan dalam bidang pemrosesan bahasa natural *(NLP)*, yaitu untuk mengklasifikasikan teks, dimana *CNN* mengklasifikasikan teks dengan

menggunakan teknik *convolution* pada kalimat, paragraf, atau dokumen secara keseluruhan, (Yuliska et al., 2021).



Gambar 1. Arsiterktur CNN, (Hermanto et al., 2021)

6. Supervised Learning

Supervised Learning adalah algoritma pembelajaran mesin yang menggunakan data *input* berlabel untuk mempelajari fungsi yang dapat menghasilkan *output* sesuai saat diberikan data baru tanpa label. Algoritma ini melatih model dengan data yang sudah diketahui hasilnya, sehingga bisa memprediksi atau mengklasifikasikan data yang belum memiliki label. Contoh penerapannya termasuk klasifikasi email spam dan prediksi harga rumah, (Kristiawan et al., 2020).

Supervised Learning memerlukan validasi dan pengujian untuk memastikan *accuracy* dan kinerja model. Teknik seperti *cross-validation* membagi data pelatihan menjadi beberapa subset untuk menghindari *overfitting* atau *underfitting*. *Supervised Learning* sering digunakan dalam membuat model *Machine Learning* untuk dua jenis masalah utama diantaranya :

a. Klasifikasi, untuk memetakan input ke dalam salah satu dari beberapa kategori. Contohnya termasuk klasifikasi email sebagai spam atau bukan spam, dan pengenalan tulisan tangan dan angka.
b. Regresi, untuk memprediksi nilai kontinu. Contohnya termasuk memprediksi harga rumah berdasarkan fitur-fiturnya seperti luas bangunan, jumlah kamar tidur, dan lokasi.

7. TensorFlow

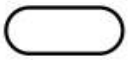
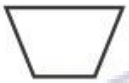
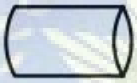
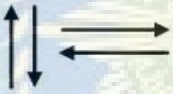
TensorFlow adalah sebuah *open source library* untuk *machine learning* yang dirilis oleh Google dan mendukung berbagai bahasa pemrograman. *TensorFlow* digunakan dalam proses *transfer learning* untuk memproses *Model Inception-v3* untuk dilatih ulang dengan data baru. Kemudian, itu menghasilkan *classifier* dengan komputasi yang cepat dan akurat. *TensorFlow* berfungsi dengan semua sistem operasi, (Nugroho et al., 2020).

TensorFlow dapat dijalankan pada berbagai *platform* (CPU, GPU, TPU) dan mendukung berbagai sistem operasi. API fleksibel yang disediakan termasuk Keras untuk penggunaan tingkat tinggi dan *low-level* API untuk kontrol lebih detail. *TensorFlow* juga memiliki ekosistem luas dengan alat bantu seperti *TensorBoard* dan *TensorFlow Lite*, serta komunitas yang kuat dan dokumentasi lengkap. Cocok untuk pengembangan dan implementasi model pembelajaran mesin di berbagai skala dan lingkungan.

8. Flowchart

Bagian alir juga dikenal sebagai *Flowchart*, adalah metode analitis bergambar yang digunakan untuk menjelaskan secara ringkas, jelas, dan logis beberapa komponen sistem informasi. Tujuan dari *Flowchart* adalah untuk menggambarkan prosedur program secara grafik. Biasanya digunakan untuk memudahkan penyelesaian masalah yang terutama

memerlukan penyelidikan dan evaluasi lanjut, (Tuasamu et al., 2023).

Simbol	Nama	Fungsi
	Terminal	Digunakan untuk memulai atau mengakhiri program.
	Input/Output	Digunakan untuk menyatakan input atau output tanpa melihat jenisnya.
	Manual Opertion	Digunakan untuk menunjukkan pengolahan yang tidak dilakukan oleh komputer.
	Decision	Digunakan untuk memilih proses yang akan dilakukan berdasarkan kondisi tertentu.
	Processing	Digunakan untuk menunjukkan pengolahan data yang dilakukan oleh komputer.
	Disk Storage	Digunakan untuk menyatakan masukan dan keluaran yang berasal dari disk.
	Flow Direction Symbol/Connecting line	Berfungsi untuk menghubungkan simbol yang satu dengan yang lainnya, menyatakan arus suatu proses.

Gambar 2. Flowchart, (Yulianeu & Oktamala, 2022).

9. Scikit Learn

Scikit-Learn merupakan modul yang memungkinkan pengembangan *machine learning* menggunakan *Python* dan tersedia di bawah lisensi *3-Clause* BSD, (Nafisah Nurul Hakim, 2020). Modul ini menawarkan berbagai alat untuk analisis data dan pembuatan model prediktif, termasuk algoritma untuk klasifikasi, regresi, klastering, pengurangan dimensi, validasi model, dan seleksi fitur. *Scikit-Learn* juga terintegrasi dengan pustaka *Python* lain seperti *NumPy*, *SciPy*, dan *matplotlib*, sehingga mempermudah pengguna dalam *ppreprocessing* data, membangun dan mengevaluasi model, serta visualisasi hasil.

B. Penelitian Terkait

1. Rona Guines Purnasiwi , Kusrini, Muhammad Hanafi (2023)

   Pada penelitian yang berjudul "Analisis Sentimen Pada Review Produk *Skincare* Menggunakan *Word Embedding* dan Metode *Long Short-Term Memory* (LSTM)" Penelitian ini bertujuan untuk memahami persepsi konsumen terhadap produk perawatan kulit *Female Daily* dengan mengembangkan model untuk klasifikasi sentimen. Masalah yang diidentifikasi dalam penelitian tersebut adanya ketidakseimbangan jumlah dataset positif, negatif, dan netral, yang dapat mengakibatkan *overfitting* pada hasil analisis sentimen. Hasil penelitian menunjukkan bahwa konsumen cenderung memberikan ulasan negatif pada produk tersebut, dengan menunjukkan penggunaan *Word2Vec* meningkatkan *accuracy* model LSTM menjadi 91%, sedangkan tanpa *Word2Vec accuracy* mencapai 71%, (Purnasiwi et al., 2023).

2. Mustasaruddin, Elvia Budianita, M Fikry, Febi Yanto (2023)

   Penelitian yang berjudul "Klasifikasi Sentimen Review Aplikasi *MyPertamina* Menggunakan *Word Embedding FastText* dan SVM (*Support Vector Machine*)" bertujuan mengidentifikasi sentimen ulasan aplikasi *MyPertamina* di Google *Play* menggunakan model SVM dan pembobotan fitur *FastText*. Hasil penelitian menunjukkan mayoritas ulasan adalah negatif. Selain itu, metode *Lexicon-Based* dengan *VADER* digunakan untuk klasifikasi sentimen menunjukkan metode SVM menjadi model terbaik dengan data 90:10, menghasilkan *accuracy* 80%, *recall* 50%, dan *precission* 84%, serta mengklasifikasikan 1.405 ulasan positif, 1.698 netral dan 4.897 negatif dari 8.000 ulasan, (Mustasaruddin et al., 2023).

3. Mohammed Hafizh Al-Areef1, Kana Saputra S (2023)

Pada penelitian yang berjudul "Analisis Sentimen Pengguna Twitter Mengenai Calon Presiden Indonesia Tahun 2024 Menggunakan Algoritma LSTM". Penelitian ini bertujuan mengklasifikasikan sentimen masyarakat Indonesia tentang calon presiden 2024 (Ganjar Pranowo, Prabowo Subianto, Ridwan Kamil) di Twitter menggunakan algoritma *Long Short Term Memory* (LSTM). Prosesnya meliputi *ppreprocessing* data, pelabelan dengan *TextBlob, Word Embedding* dengan *FastText,* dan *Balancing* data dengan SMOTE, serta penentuan *Hyperparameter* untuk melatih model LSTM. Hasil menunjukkan model LSTM mencapai *accuracy* dan performa tinggi: Ganjar Pranowo dengan *accuracy* 82%, *precission* 86%, *recall* 92%, dan *f1-score* 89%; Prabowo Subianto dengan *accuracy* 82%, *precission* 82%, *recall* 96%, dan *f1-score* 89%; Ridwan Kamil dengan *accuracy* 87%, *precission* 91%, *recall* 95%, dan *f1-score* 93%. Penelitian ini membuktikan bahwa LSTM dengan hyperparameter yang tepat efektif dalam klasifikasi sentimen di Twitter, (Al-Areef & Saputra S, 2023).

4. Putri Rizki Amalia (2021)

Penelitian yang berjudul "Analisis Sentimen Berdasarkan Aspek Pada Ulasan Restoran Berbahasa Indonesia Menggunakan *CNN* Dan *Contextualized Word Embedding*" bertujuan membandingkan kinerja *Word2Vec, Glove*, dan *FastText* dalam klasifikasi teks menggunakan *CNN*. Ketiga metode ini dipilih karena kemampuannya menangkap makna semantik, sintaktik, dan konteks kata, yang lebih unggul dibandingkan *Bag of Words*. Eksperimen dilakukan pada dua dataset berita: 20 *Newsgroup* dan Reuters Newswire, dengan kinerja diukur menggunakan *F-measure*. Hasilnya, *FastText* menunjukkan performa terbaik dengan *F-Measure* 0.979 untuk 20 *Newsgroup* dan 0.715 untuk *Reuters*,

meskipun perbedaan kinerja antara ketiganya tidak signifikan, (Amalia & Winarko, 2021).

5. Novrindah Alvi Hasanah, Nanik Suciati, Diana Purwitasari (2021).

   Pada penelitian yang berjudul "Pemantauan Perhatian Publik terhadap Pandemi COVID-19 melalui Klasifikasi Teks dengan *Deep Learning*". Tujuan dari penelitian ini adalah untuk memantau perhatian publik terhadap pandemi COVID-19 melalui klasifikasi teks Twitter menggunakan variasi *Deep Learning* (*CNN*, RNN, dan LSTM) dan variasi *Word Embedding* (*Word2Vec* dan *FastText*). Penelitian ini bertujuan untuk mengetahui seberapa tinggi perhatian publik terhadap pandemi COVID-19 berdasarkan data Twitter di area Surabaya. Masalah yang ditangani dalam penelitian ini yaitu adanya variasi topik pada data teks Twitter yang sulit diidentifikasi dan diklasifikasi pada topik tertentu. Hasil penelitian ini menunjukkan bahwa kombinasi metode *FastText* dan LSTM memberikan *accuracy* terbaik dalam klasifikasi teks Twitter yang berhubungan dengan COVID-19. *Accuracy* terbaik yang diperoleh adalah 97.86%, yang menunjukkan bahwa metode ini dapat dengan baik mengklasifikasikan teks dari data yang bervariasi, (Alvi Hasanah et al., 2021).

6. Yuliska, Dini Hidayatul Qudsi, Juanda Hakim Lubis, Khairul Umam Syaliman, Nina Fadilah Najwa (2021).

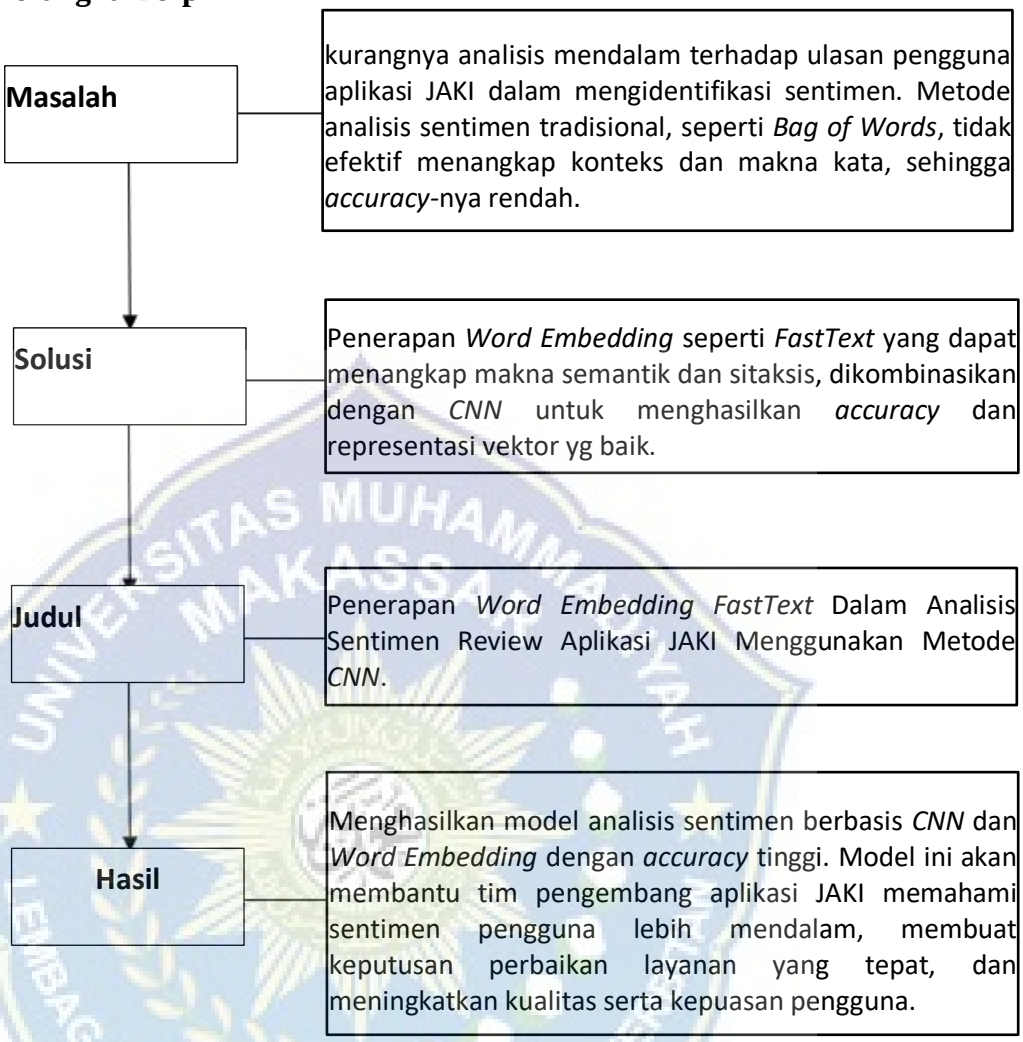
   Pada penelitian yang berjudul "Analisis Sentimen Pada Data Saran Mahasiswa Terhadap Kinerja Departemen Di Perguruan Tinggi Menggunakan *CNN*". Penelitian ini bertujuan menganalisis sentimen data saran mahasiswa terhadap kinerja departemen di Politeknik Caltex Riau menggunakan *CNN* dan *Word2Vec*. Masalah yang diidentifikasi adalah pengolahan data saran belum menerapkan analisis sentimen, sehingga saran belum terfilter menjadi positif dan

negatif untuk pengambilan keputusan. Hasil penelitian menunjukkan bahwa metode DoubleMax *CNN* mencapai *accuracy* 98%, *recall* 97%, *precision* 98%, dan *F1-score* 98%, dengan menggunakan *Word2Vec* untuk representasi kata dan *CNN* untuk representasi kalimat panjang, (Yuliska et al., 2021).

7. Aliyanti Nurdin, Bernadus Anggo Seno Aji, Anugrayani Bustamin, Zaenal Abidin (2020).

   Pada penelitian yang berjudul "Perbandingan Kinerja *Word Embedding Word2Vec*, *Glove* dan *FastText* Pada Klasifikasi Teks" bertujuan membandingkan kinerja *Word2Vec*, *Glove*, dan *FastText* dalam klasifikasi teks menggunakan *CNN*. Metode ini lebih unggul dibandingkan *Bag of Words* yang tidak mempertimbangkan konteks. Eksperimen dilakukan pada dataset berita 20 *Newsgroup* dan *Reuters Newswire*, dengan kinerja diukur menggunakan *F-measure*. Hasilnya, *FastText* menunjukkan performa terbaik dengan *F-Measure* 0.979 untuk 20 *Newsgroup* dan 0.715 untuk *Reuters*, meskipun perbedaan kinerja antara ketiga metode tidak signifikan, (Nurdin et al., 2020).

C. Kerangka Berpikir



Gambar 3. Kerangka Pikir

BAB III
METODE PENELITIAN

A. Tempat dan Waktu Penelitian

1. Tempat Penelitian

Penelitian ini dilakukan secara online dengan mengambil beberapa *ulasan* aplikasi JAKI pada *platform* Google *PlayStore.*

2. Waktu Penelitian

Adapun pelaksanaan penelitian ini dilakukan mulai pada Juni 2024 dan akan berlangsung hingga seluruh proses pengumpulan data selesai.

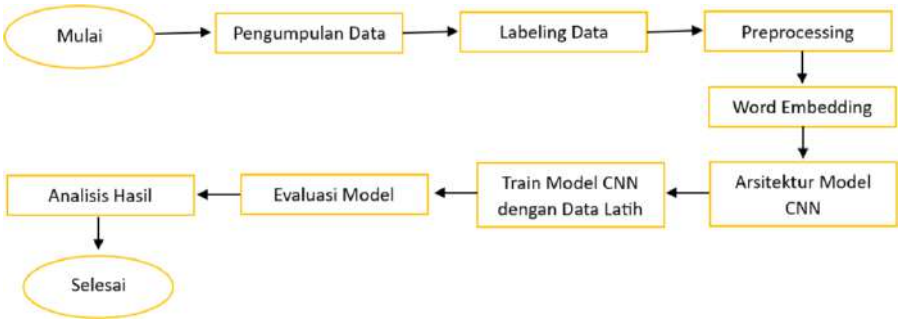
B. Alat dan Bahan

1. Kebutuhan *Hardware*
   a. Laptop Asus VivoBook X415JAB
2. Kebutuhan *Software*
   a. Google Colaboratory
   b. Google Drive
   c. Excel
   d. *Python*
   e. *Scikit-Learn* dan *TensorFlow*

C. Perancangan Sistem

Perancangan sistem adalah aspek penting dalam pengembangan suatu sistem karena menjelaskan proses dari tahap perencanaan hingga implementasi fungsi-fungsi yang diperlukan untuk operasional. Tujuan utama perancangan sistem adalah memastikan bahwa sistem yang akan dibangun mampu mencapai hasil yang diharapkan.

Flowchart atau diagram alur adalah representasi grafis yang menampilkan urutan langkah-langkah dan keputusan yang diperlukan untuk menjalankan suatu proses dalam suatu program. Setiap langkah direpresentasikan dalam bentuk diagram dan dihubungkan oleh garis atau panah untuk menunjukkan arah alur proses.

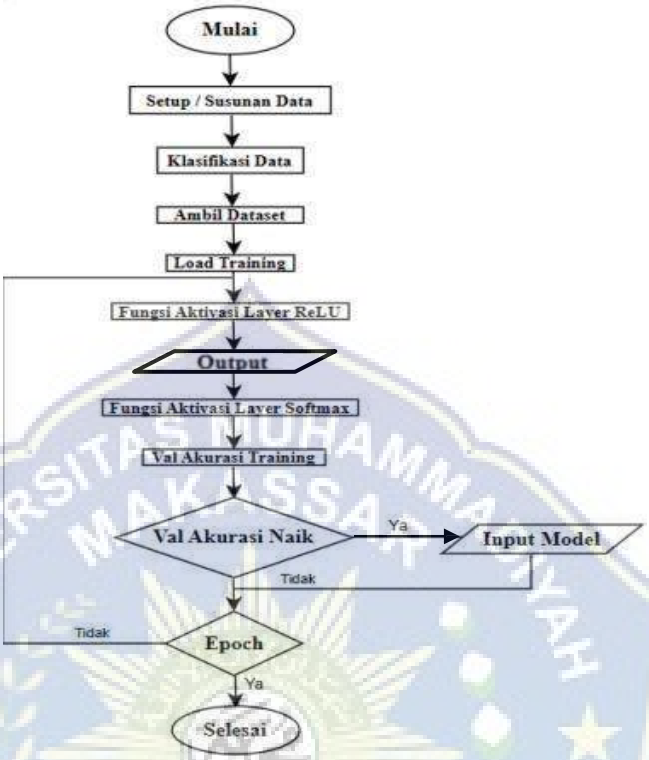


Gambar 4. Perancangan Sistem

Diagram tersebut menggambarkan proses dalam mengevaluasi penggunaan *Word Embedding* dengan *FastText* untuk meningkatkan kinerja model *CNN* dalam analisis sentimen terhadap ulasan aplikasi JAKI. Inisiasi proyek dan pengumpulan data dibatasi hanya ulasan aplikasi JAKI pada *platform* Google *PlayStore* menggunakan teknik *scraping*. Setelah data terkumpul, selanjutnya adalah pelabelan data, di mana ulasan diberikan label sentimen positif atau negatif sebagai target untuk pelatihan model diikuti oleh proses pra-pemrosesan seperti pembersihan, tokenisasi, dan normalisasi teks. Berikutnya ialah mengubah teks ulasan menjadi representasi vektor numerik menggunakan teknik *Word Embedding* dengan *FastText*, yang menangkap makna semantik dari setiap kata. Setelah mendapatkan *embedding* dari *FastText*, representasi vektor kata ini digunakan sebagai input untuk model yang dibangun menggunakan *framework* seperti *TensorFlow.*

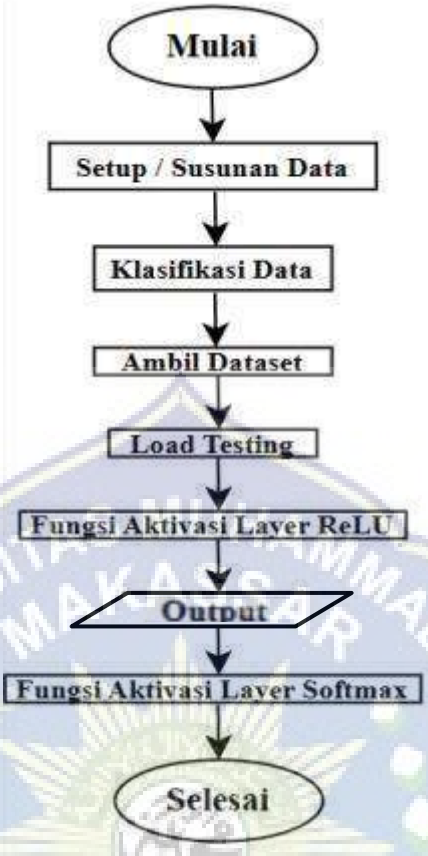
Dengan data yang telah di-embed, desain arsitektur model *CNN* kemudian dilatih menggunakan data latih yang telah melalui proses *embedding*. Setelah pelatihan selesai, model dievaluasi untuk menilai kinerjanya menggunakan metrik seperti *accuracy, precision, recall* dan *F1-score*. Hasil evaluasi kemudian dianalisis untuk memahami seberapa baik model bekerja dalam mengklasifikasi sentimen ulasan. Setelah proses pelatihan dan evaluasi, setiap langkah penelitian di dokumentasikan dengan baik, diikuti oleh penyusunan laporan skripsi yang mencakup latar belakang, metodologi, hasil, dan kesimpulan.

Dalam perancangan sistem atau diagram sistem yang akan dibuat yaitu sebagai berikut :



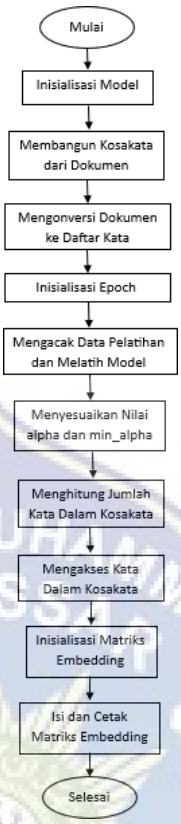
Gambar 5. Perancangan Sistem Training

Dalam diagram di atas, alur kerja dimulai dengan persiapan data, pengklasifikasian, dan pemuatan data untuk pelatihan model. Beberapa lapisan menerapkan fungsi aktivasi *ReLU*, diikuti oleh lapisan output yang menggunakan fungsi aktivasi *Softmax* untuk menghasilkan probabilitas kelas. Selama pelatihan, *accuracy* data pelatihan dievaluasi secara berkala, dan jika *accuracy* terus meningkat, proses dilanjutkan ke langkah berikutnya. Model dimasukkan ke dalam sistem untuk proses pelatihan yang melibatkan iterasi (*epoch*) guna memperbaiki kinerja model. Setelah pelatihan selesai, proses pun berakhir.



Gambar 6. Perancangan Sistem Testing

Sistem yang dirancang memiliki dua alur utama: perancangan sistem pelatihan dan perancangan sistem pengujian. Pada perancangan sistem pelatihan, fokus utamanya adalah memantau kurva *accuracy* untuk mengevaluasi kinerja model selama proses pelatihan. Sementara itu, perancangan sistem pengujian difokuskan pada tahap pengujian, dengan tujuan utama untuk menguji kemampuan sistem dalam mengidentifikasi dan menguji aspek-aspek yang ditargetkan.



Gambar 7. Flowchart FastText

Flowchart di atas menjelaskan proses pelatihan model FastText untuk pemrosesan teks. Proses dimulai dengan inisialisasi model FastText, diikuti oleh pembentukan kosakata dari dokumen yang telah dikonversi menjadi daftar kata. Kemudian, model diinisialisasi dengan epoch sebanyak 30. Selanjutnya, model memeriksa apakah epoch kurang dari 30. Jika ya, data pelatihan diacak dan model dilatih, serta nilai alpha dan min_alpha disesuaikan. Jika tidak, model melanjutkan untuk menghitung jumlah kata dalam kosakata dan mengakses kata-kata tersebut. Setelah itu, dilakukan inisialisasi matriks embedding, diikuti dengan pengisian dan pencetakan matriks embedding. Proses berakhir setelah seluruh langkah ini selesai.

D. Teknik Pengujian Sistem

Teknik pengujian sistem yang akan digunakan dalam pengujian ini melibatkan pemisahan data menjadi dua bagian, yaitu *data training* dan *data testing*. Dengan demikian, teknik pengujian ini memungkinkan

evaluasi yang akurat terhadap kinerja model dalam mengklasifikasikan sentimen pada teks ulasan aplikasi JAKI.

Penelitian ini menggunakan teknik pengujian *Confusion Matrix* untuk mendapatkan pemahaman yang lebih baik tentang bagaimana model berfungsi untuk menganalisis sentimen dengan tiga variabel yaitu, positif, negatif, dan netral. *Confusion Matrix* memungkinkan analisis menyeluruh terhadap kemampuan model untuk mengklasifikasikan setiap sentimen.

Pengujian *accuracy* bertujuan untuk mengukur keberhasilan model dalam mengklasifikasikan sentimen dengan tepat. *Accuracy* model dihitung dengan membandingkan hasil klasifikasi sentimen dari model dengan label sentimen yang sebenarnya pada data pengujian, menggunakan persamaan tertentu untuk menghitung proporsi prediksi yang benar dari keseluruhan data uji.

Di bawah ini adalah rumus perhitungan *Confusion Matrix* untuk menghitung *precision*, *recall*, dan nilai *accuracy*, (Dinata et al., 2020) :

a. Accuracy

Berfungsi untuk mengukur sejauh mana kinerja model dalam mengkalsifikasikan kelas sentimen dalam dataset

$$AC = \frac{TP + TN}{(TP+TN+FP+FN)} \times 100\% \quad (1)$$

b. Recall

Berdasarkan persamaan berikut, berfungsi untuk mengukur tingkat keberhasilan sistem dalam menemukan kembali data

$$Rec = \frac{TP}{(TP+FN)} \times 100\% \quad (2)$$

c. Precision

Berfungsi untuk mengevaluasi ketepatan hubungan antara informasi yang diminta pengguna dan jawaban yang diberikan sistem.

$$Pre = \frac{TP}{(TP+FP)} \times 100\% \quad (3)$$

Keterangan :

TP : *True Positif*

TN : *True Negatif*

FP : *False Positif*

FN : *False Negatif*

E. Teknik Analisis Data

Analisis data adalah upaya mencari dan menata hasil observasi, wawancara, dan hasil lainnya secara sistematis untuk meningkatkan pemahaman peneliti tentang kasus yang diteliti serta menyajikannya sebagai temuan bagi orang lain. Ini juga mencakup proses mengurutkan dan mengorganisir data ke dalam pola, kategori, dan uraian dasar sejak awal pengumpulan data di lapangan untuk memastikan data terkumpul secara lengkap dan terstruktur, (Nurdewi, 2022). Untuk mencapai hasil yang diinginkan maka peneliti melakukan beberapa tahapan analisis sebagai berikut :

1. Reduksi Data (*Data Reduction*)

   Reduksi data adalah proses menyederhanakan, memilah, dan mengubah data kasar menjadi bentuk yang lebih mudah dikelola dan dipahami. Dalam tahap ini, data yang relevan dipilih dan dirangkum sementara data yang tidak relevan dibuang. Proses ini melibatkan kegiatan seperti *coding*, pengelompokan, dan pembuatan tema atau kategori. Tujuannya adalah untuk mengidentifikasi pola dan tema utama yang muncul dari data sehingga memudahkan analisis lebih lanjut.

2. Penyajian Data (*Display Data*)

   Penyajian data adalah proses menata data yang telah direduksi dalam bentuk yang terorganisir dan mudah dipahami. Data dapat disajikan dalam berbagai bentuk seperti teks naratif, tabel, grafik,

atau diagram. Penyajian data bertujuan untuk membantu peneliti memahami informasi yang kompleks dengan lebih baik dan untuk mendukung proses penarikan kesimpulan. Ini juga memudahkan dalam mengkomunikasikan temuan kepada audiens lain.

3. Penarikan Kesimpulan (*Conclucing Drawing Verification*)

Penarikan kesimpulan adalah tahap di mana peneliti mengevaluasi data yang telah direduksi dan disajikan untuk membuat interpretasi dan kesimpulan yang *valid*. Proses ini melibatkan mencari hubungan, pola, dan makna dalam data serta mengaitkan temuan dengan pertanyaan penelitian atau hipotesis yang telah ditetapkan. Kesimpulan yang ditarik harus didasarkan pada bukti yang kuat dan dapat diandalkan dari data yang telah dianalisis.

BAB IV
HASIL DAN PEMBAHASAN

A. Pengumpulan Data

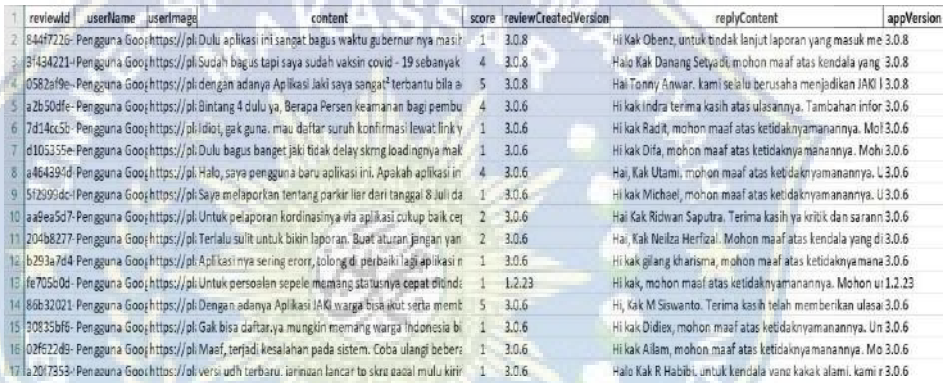
Data ulasan diambil menggunakan teknik *scraping* pada *python*, yang mengumpulkan data dari *platform* Google *PlayStore* pada Aplikasi JAKI. Dengan jumlah total data yang berhasil dikumpulkan ialah sebanyak 4.455 ulasan. Berikut proses pengambilan data menggunakan teknik scraping pada pyhton :

```
from google_play_scraper import Sort, reviews_all
import pandas as pd

# Mengumpulkan semua ulasan untuk aplikasi Jakarta Kini
scrapreview = reviews_all(
    'id.go.jakarta.smartcity.jaki',
    sleep_milliseconds=0,  # Anda dapat menyesuaikan
jeda waktu jika diperlukan
    lang='id',
    country='id',
    sort=Sort.MOST_RELEVANT
)
# Menampilkan hasil scrapping
 print(scrapreview)

# Membuat DataFrame dari hasil scrapping
app_reviews_df = pd.DataFrame(scrapreview)

# Menyimpan DataFrame ke dalam file Excel
app_reviews_df.to_excel('C:/Users/asus/Downloads/DataRa
ting.xlsx', index=False, header=True)
```

Kode ini dimulai dengan mengimpor dua *library*, yaitu `google_play_scraper` dan `pandas`. Fungsi `reviews_all` digunakan untuk mengumpulkan semua ulasan aplikasi "Jakarta Kini" dari Google *PlayStore*. Parameter `sleep_milliseconds`, `lang`, `country`, dan `sort` menentukan pengaturan untuk pengambilan data seperti tidak ada jeda waktu, bahasa Indonesia, ulasan dari Indonesia, dan pengurutan berdasarkan relevansi. Hasil ulasan ini disimpan dalam variabel `scrapreview` dan ditampilkan menggunakan `print`. Kemudian, data

ulasan tersebut diubah menjadi sebuah *DataFrame* menggunakan `pandas.*DataFrame*`. Terakhir, *DataFrame* tersebut disimpan ke dalam file Excel bernama `DataRating.xlsx` di direktori yang ditentukan, tanpa menyertakan indeks dan dengan menyertakan nama kolom. Pesan konfirmasi dicetak untuk menunjukkan bahwa data telah berhasil disimpan.

Setelah mengumpulkan data ulasan dari Google *PlayStore* untuk aplikasi JAKI menggunakan *Scraping* Data, hasilnya disimpan dalam format excel. Berikut adalah hasil pengambilan dataset ulasan yang kemudian di simpan dalam format excel



	reviewId	userName	userImage	content	score	reviewCreatedVersion	replyContent	appVersion
2	844f7226-	Pengguna Goo	https://pl	Dulu aplikasi ini sangat bagus waktu gubernur nya masih	1	3.0.8	Hi Kak Obenz, untuk tindak lanjut laporan yang masuk me	3.0.8
3	3f434221-	Pengguna Goo	https://pl	Sudah bagus tapi saya sudah vaksin covid - 19 sebanyak	4	3.0.8	Halo Kak Danang Setyadi, mohon maaf atas kendala yang	3.0.8
4	0582af9e-	Pengguna Goo	https://pl	dengan adanya Aplikasi Jaki saya sangat² terbantu bila a	5	3.0.8	Hai Tonny Anwar. kami selalu berusaha menjadikan JAKI l	3.0.8
5	a2b50df6-	Pengguna Goo	https://pl	Bintang 4 dulu ya. Berapa Persen keamanan bagi pembu	4	3.0.6	Hi kak Indra terima kasih atas ulasannya. Tambahan infor	3.0.6
6	7d14cc5b-	Pengguna Goo	https://pl	Idiot, gak guna. mau daftar suruh konfirmasi lewat link y	1	3.0.6	Hi kak Radit, mohon maaf atas ketidaknyamanannya. Mol	3.0.6
7	d105355e-	Pengguna Goo	https://pl	Dulu bagus banget jaki tidak delay skrng loadingnya mak	1	3.0.6	Hi kak Difa, mohon maaf atas ketidaknyamanannya. Mohi	3.0.6
8	a46439d	Pengguna Goo	https://pl	Halo, saya pengguna baru aplikasi ini. Apakah aplikasi in	4	3.0.6	Hai, Kak Utami. mohon maaf atas ketidaknyamanannya. U	3.0.6
9	5f2899dc-	Pengguna Goo	https://pl	Saya melaporkan tentang parkir liar dari tanggal 8 Juli da	1	3.0.6	Hi kak Michael, mohon maaf atas ketidaknyamanannya. U	3.0.6
10	aa9ea5d7-	Pengguna Goo	https://pl	Untuk pelaporan kordinasinya via aplikasi cukup baik cep	2	3.0.6	Hai Kak Ridwan Saputra. Terima kasih ya kritik dan saran	3.0.6
11	204b8277-	Pengguna Goo	https://pl	Terlalu sulit untuk bikin laporan. Buat aturan jangan yan	2	3.0.6	Hai, Kak Neilza Herfizal. Mohon maaf atas kendala yang di	3.0.6
12	b293a7d4	Pengguna Goo	https://pl	Aplikasi nya sering erorr, tolong di perbaiki lagi aplikasi n	1	3.0.6	Hi kak gilang kharisma, mohon maaf atas ketidaknyamana	3.0.6
13	fe705c0d-	Pengguna Goo	https://pl	Untuk persoalan sepele memang statusnya cepat ditinda	1	1.2.23	Hi kak, mohon maaf atas ketidaknyamanannya. Mohon ur	1.2.23
14	86b32021-	Pengguna Goo	https://pl	Dengan adanya Aplikasi JAKI warga bisa ikut serta memb	5	3.0.6	Hi, Kak M Siswanto. Terima kasih telah memberikan ulasa	3.0.6
15	30835bf6-	Pengguna Goo	https://pl	Gak bisa daftar,ya mungkin memang warga Indonesia bi	1	3.0.6	Hi kak Didiex, mohon maaf atas ketidaknyamanannya. Un	3.0.6
16	02f622d9-	Pengguna Goo	https://pl	Maaf, terjadi kesalahan pada sistem. Coba ulangi beber	1	3.0.6	Hi kak Ailam, mohon maaf atas ketidaknyamanannya. Mo	3.0.6
17	a20f7353-	Pengguna Goo	https://pl	versi udh terbaru, jaringan lancar tp skrg gagal mulu kirir	1	3.0.6	Halo Kak R Habibi, untuk kendala yang kakak alami, kami r	3.0.6

Gambar 8. Dataset Ulasan Aplikasi JAKI

Setelah dataset ulasan pada aplikasi JAKI telah diperoleh dengan mendaptkan total keseluruhan data sebanyak 4.455 ulasan, selanjutnya dilakukan penghapusan atribut seperti *reviewId, userName, userIamage, score, reviewCreatedVersion, replyContent, appVersion* dan hanya menyisahkan *"content"* saja yang nantinya akan dijadikan sebagai atribut ulasan. Setelah itu, dataset diurutkan berdasarkan abjad A-Z untuk memudahkan dalam proses pembersihan data yang memiliki kemiripan ulasan serta menghapus ulasan-ulasan yang hanya menggunakan *emoji* tanpa teks, yang kemudian menyisahkan total data secara keseluruhan ialah sebanyak 3.199 data ulasan.

B. Pelabelan Data

Pelabelan sentimen dilakukan secara manual untuk menemukan pola dan karakteristik dalam teks yang menunjukkan sentimen positif, negatif, maupun netral. Ulasan yang dikumpulkan berasal dari *platform* Google *PlayStore* pada Aplikasi JAKI yang kemudian disimpan dalam atribut ulasan, sedangkan nilai klasifikasi seperti positif, negatif, atau netral disimpan dalam atribut label. Di bawah ini adalah tabel pelabelan data:

Tabel 1. Tahap Pelabelan Data

ULASAN	LABEL
Sangat membantu warga DKI Jakarta, terutama yang akan vaksin. Yuk instal, dan langsung daftar vaksin sekarang.	Positif
Aplikasi ga jelas, saya dalam kondisi sehat, tidak pernah kontak, Suhu tubuh stabil, kenapa hasilnya tidak aman??	Negatif
Respon ok untuk laporan / aduan di bidang kebersihan, tetapi untuk di bidang lain perlu ditingkatkan...	Netral

C. *Preprocessing Data*

Pra-pemrosesan data, juga dikenal sebagai *preprocessing* data, adalah proses yang dilakukan untuk mempersiapkan data mentah agar dapat digunakan oleh algoritma pengajaran mesin atau analisis data. Tujuan dari pra-pemrosesan data adalah untuk membersihkan, mengubah, dan menyusun data sehingga lebih mudah dianalisis dan diolah. Langkah-langkah pra-pemrosesan data biasanya mencakup beberapa atau semua dari langkah-langkah yang disebutkan di atas, tergantung pada jenis data dan tujuan analisis.

1. *Punctuation*

   Punctuation atau dikenal sebagai tanda baca adalah simbol yang digunakan dalam tulisan untuk membedakan bagian-bagian yang berbeda dari teks, seperti titik (.), koma (,), tanda tanya (?), tanda seru (!), dan lainnya. Penanganan tanda baca *(punctuation)* membantu

membersihkan dan menormalisasi data dalam pra-pemrosesan teks untuk mempersiapkan data sebelum digunakan dalam analisis atau model *machine learning*.

```
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)
```

Ini bertujuan untuk menghapus tanda baca dari teks dalam kolom `ULASAN` pada *DataFrame* `df`, fungsi `remove_punctuation(text)` di definisikan untuk menghilangkan tanda baca dari sebuah teks dengan menggunakan metode `translate()` dan modul `string.punctuation`, yang berisi semua karakter tanda baca. Fungsi `str.maketrans('', '', string.punctuation)` membuat tabel translasi yang akan menggantikan semua tanda baca dengan string kosong, sehingga menghapusnya dari teks. `df['ULASAN'] = df['ULASAN'].apply(remove_punctuation)` menerapkan fungsi `remove_punctuation` pada setiap elemen dalam kolom `ULASAN`, memproses setiap teks untuk menghilangkan semua tanda baca yang ada.

Tabel 2. Tahap Punctuation Data

SEBELUM	SESUDAH
Sangat membantu warga DKI Jakarta, terutama yang akan vaksin. Yuk instal, dan langsung daftar vaksin sekarang.	Sangat membantu warga DKI Jakarta terutama yang akan vaksin Yuk instal dan langsung daftar vaksin sekarang
Aplikasi ga jelas, saya dalam kondisi sehat, tidak pernah kontak, Suhu tubuh stabil, kenapa hasilnya tidak aman??	Aplikasi ga jelas saya dalam kondisi sehat tidak pernah kontak Suhu tubuh stabil kenapa hasilnya tidak aman

Respon ok untuk laporan / aduan di bidang kebersihan, tetapi untuk di bidang lain perlu ditingkatkan...	Respon ok untuk laporan aduan di bidang kebersihan tetapi untuk di bidang lain perlu ditingkatkan

2. *Tokenizing*

Tokenizing adalah proses pemrosesan teks di mana teks dibagi menjadi bagian kecil yang disebut "token", yang biasanya terdiri dari kata-kata, frasa, atau bahkan karakter tergantung pada tujuan analisis. Tokenisasi adalah langkah penting dalam pra-pemrosesan data teks karena memungkinkan model pengajaran mesin atau algoritma lainnya untuk mengubah teks yang panjang atau kompleks menjadi bentuk yang lebih sederhana yang lebih mudah dianalisis.

```
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 token
# Memisahkan data menjadi train dan test
train, test = train_test_split(df, test_size=0.1,
random_state=42)
```

Kode ini menggunakan pustaka *Natural Language Toolkit* (nltk) untuk melakukan tokenisasi teks dan memisahkan data menjadi set *train* dan *test*. `nltk.download('punkt')` digunakan untuk mengunduh modul punkt, yang berisi algoritma pemisahan teks menjadi kalimat dan kata. Fungsi `tokenize_text(text)` kemudian didefinisikan untuk mengubah teks menjadi token (kata-kata yang terpisah). Di dalam

fungsi ini, `nltk.sent_tokenize(text)` memisahkan teks menjadi kalimat, dan `nltk.word_tokenize(sent)` memisahkan setiap kalimat menjadi kata. Kata-kata yang tidak kosong diubah menjadi huruf kecil dan disimpan dalam daftar `tokens`. Setelah itu, `train, test = train_test_split(df, test_size=0.1, random_state=42)` membagi *DataFrame* `df` menjadi dua subset: `train` dan `test`, dengan ukuran 10% dari data digunakan sebagai data test. Parameter `random_state=42` memastikan pemisahan yang konsisten untuk reproduktibilitas.

Tabel 3. Tahap Tokenizing Data

SEBELUM	SESUDAH
Sangat membantu warga DKI Jakarta, terutama yang akan vaksin. Yuk instal, dan langsung daftar vaksin sekarang.	['sangat ', 'membantu', 'warga', 'dki', 'jakarta', 'terutama', 'yang' 'akan', 'vaksin', 'yuk', 'instal', 'dan', 'langsung', 'daftar', 'vaksin', 'sekarang']
Aplikasi ga jelas, saya dalam kondisi sehat, tidak pernah kontak, Suhu tubuh stabil, kenapa hasilnya tidak aman??	['aplikasi', 'ga', 'jelas', 'saya', 'dalam', 'kondisi', 'sehat', 'tidak', 'pernah', 'kontak', 'suhu', 'tubuh', 'stabil', 'kenapa', 'hasilnya', 'tidak', 'aman']
Respon ok untuk laporan / aduan di bidang kebersihan, tetapi untuk di bidang lain perlu ditingkatkan...	['respon', 'ok', 'untuk', 'laporan', 'aduan', 'di', 'bidang','kebersihan', 'tetapi', 'untuk', 'di', 'bidang', 'lain', 'perlu', 'ditingkatkan']

D. Penerapan Metode

```
df = pd.read_excel('/content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/DATAJAKI.xlsx',sheet_name="Sheet1")  # Replace 'path_to_your_excel_file.xlsx' with your actual file path
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()
df.shape
```

Dimulai dengan membaca file Excel yang berlokasi di direktori tertentu menggunakan fungsi `pd.read_excel`. File Excel tersebut diakses dari jalur yang ditentukan, dan data diambil dari sheet bernama "Sheet1". Setelah data dimuat ke dalam *DataFrame* `df`, kode ini memilih kolom yang relevan, yaitu kolom "ULASAN" dan "LABEL", dan menyimpannya kembali di dalam `df`. Langkah berikutnya adalah menghapus baris-baris yang memiliki nilai kosong (*null*) pada kolom "ULASAN" dengan menggunakan fungsi `pd.notnull`. Setelah itu, kode ini mengganti nama kolom "ULASAN" menjadi "ULASAN" dengan `df.rename` untuk menjaga konsistensi penamaan kolom, meskipun dalam hal ini tidak terjadi perubahan karena nama awal dan akhir sama. Akhirnya, kode ini menampilkan beberapa baris pertama dari *DataFrame* `df` dengan menggunakan `df.head()`, yang berguna untuk melihat pratinjau data yang telah dibersihkan dan dipilih, serta `df.shape` yang memberikan informasi tentang dimensi dari *DataFrame* berisi jumlah baris dan jumlah kolom.

```
# 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))
```

Kode ini menyiapkan data teks untuk pemodelan dengan langkah-langkah sistematis. Kode membuat instance `TaggedDocument` untuk dataset pelatihan dan pengujian, di mana setiap `TaggedDocument` berisi kata-kata yang telah ditokenisasi dari kolom 'ULASAN' serta tag yang sesuai dengan label ulasan. Ini mempersiapkan data untuk model yang memerlukan format teks tertentu, seperti model dalam pustaka Gensim. Selanjutnya, sebuah `Tokenizer` diinisialisasi dengan batas kosakata maksimum 500.000 kata dan diatur untuk memproses teks dengan memisahkan kata berdasarkan spasi dan mengabaikan tanda baca.

Tokenizer ini juga mengonversi semua teks menjadi huruf kecil untuk menjaga konsistensi. *Tokenizer* kemudian dilatih menggunakan data teks dari kolom 'ULASAN' dalam *DataFrame* `df`, sehingga mempelajari kosakata yang ada. Setelah itu, teks dari dataset pelatihan dan pengujian dikonversi menjadi urutan angka, di mana setiap angka mewakili kata dalam kosakata *tokenizer*, dan urutan ini dipadatkan menjadi panjang yang seragam sebanyak 50 token untuk memastikan konsistensi ukuran input bagi model. Terakhir, kode mencetak jumlah token unik yang ditemukan dalam kosakata *tokenizer*, memberikan

gambaran mengenai ukuran kosakata data teks. Langkah-langkah ini secara efektif menyiapkan data teks dengan mengonversinya ke format yang sesuai untuk model pembelajaran mesin.

```
from gensim.models import FastText
from gensim.models.FastText import TaggedDocument
# Ubah ukuran vektor (vector_size) sesuai kebutuhan
Anda
vector_size = 500
# Inisialisasi model FastText
FastText_model = FastText(vector_size=vector_size,
window=8, min_count=1, workers=1, sg=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()]
# Ubah train_tagged menjadi list kata-kata
train_corpus = [doc.words for doc in train_tagged]
# Bangun vocab dari train_corpus
FastText_model.build_vocab(train_corpus)
```

Kode ini mengatur dan membangun model *FastText* menggunakan pustaka Gensim. Ukuran vektor (`vector_size`) diatur menjadi 500 untuk model *FastText*. Model kemudian diinisialisasi dengan berbagai parameter, termasuk ukuran jendela konteks (`window`), jumlah minimum kemunculan kata (`min_count`), jumlah *thread* (`workers`), tipe model (`sg` untuk Skip-gram), serta tingkat pembelajaran awal (`alpha`) dan minimum (`min_alpha`). Setelah inisialisasi, kosakata dibangun dari data pelatihan. Ini dilakukan dengan membuat daftar `TaggedDocument` dari data pelatihan, di mana setiap `TaggedDocument` berisi kata-kata yang telah ditokenisasi dari kolom 'ULASAN' dan tag yang sesuai dengan label. Daftar kata-kata ini kemudian disiapkan sebagai `train_corpus`, dan kosakata dibangun untuk model *FastText* menggunakan `build_vocab` berdasarkan `train_corpus`.

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

Kode ini melatih model *FastText* selama 30 *epoch* dengan data pelatihan. Proses pelatihan dilakukan dengan mengacak urutan data pelatihan menggunakan fungsi `shuffle` dari `sklearn.utils`, untuk memastikan bahwa data yang digunakan dalam setiap *epoch* bervariasi. Model dilatih dengan data yang sudah diacak, dan parameter `total_examples` diatur sesuai dengan jumlah contoh dalam `train_corpus`. Setelah setiap *epoch*, tingkat pembelajaran awal (`alpha`) dikurangi sebesar 0.002, dan nilai `min_alpha` diperbarui agar sesuai dengan nilai `alpha` saat ini. Ini membantu mengurangi laju pembelajaran seiring berjalannya pelatihan. Setelah semua *epoch* selesai, model *FastText* yang telah dilatih dicetak untuk menampilkan hasil akhir dari pelatihan.

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


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

Kode ini bertujuan untuk memperoleh dan menampilkan informasi tentang kosakata yang dibangun oleh model *FastText*. Pertama, jumlah

kata dalam kosakata dihitung dengan menggunakan `len(*FastText*_model.wv.key_to_index)`, yang mengembalikan jumlah total kata yang ada dalam kosakata model. Hasilnya dicetak untuk menunjukkan berapa banyak kata yang telah dipelajari oleh model. Selanjutnya, kode mengakses semua kata dalam kosakata dengan mengambil kunci dari kamus `key_to_index` dan mengubahnya menjadi daftar menggunakan `list()`. Daftar kata-kata ini kemudian dicetak untuk menampilkan semua kata yang ada dalam kosakata model.

```
# Inisialisasi matriks embedding kosong
embedding_matrix =
np.zeros((len(FastText_model.wv.key_to_index),
FastText_model.vector_size))
# Mengisi matriks embedding dengan vektor-vektor kata
dari model FastText
for i, word in
enumerate(FastText_model.wv.index_to_key):
    embedding_vector =
FastText_model.wv.get_vector(word)
    if embedding_vector is not None:
        embedding_matrix[i] = embedding_vector
# Contoh penggunaan matriks embedding
print("Ukuran matriks embedding:",
embedding_matrix.shape)
print("Contoh vektor untuk kata pertama:",
embedding_matrix[0])
```

Kode ini menginisialisasi dan mengisi matriks *embedding* berdasarkan vektor kata dari model *FastText*. Sebuah matriks *embedding* kosong dengan ukuran yang sesuai dibuat menggunakan `np.zeros`, dimana jumlah barisnya adalah jumlah kata dalam kosakata model (`len(*FastText*_model.wv.key_to_index)`) dan jumlah kolomnya adalah ukuran vektor *embedding* (`*FastText*_model.vector_size`). Kemudian, kode mengisi matriks *embedding* dengan vektor-vektor kata dari model *FastText*. Untuk setiap kata dalam kosakata model (`*FastText*_model.wv.index_to_key`), vektor kata diambil menggunakan `*FastText*_model.wv.get_vector(word)`. Jika vektor kata tidak kosong,

vektor tersebut dimasukkan ke dalam baris yang sesuai dalam matriks *embedding*.

Tabel 4. Tahap Vektorisasi Data

SEBELUM	SESUDAH
di	-2.08027959e-01
aplikasi	-5.76429367e-02
saya	3.31954211e-02
vaksin	-5.45904189e-02
bisa	-1.66589215e-01
dan	-4.63835895e-03
tidak	1.95747614e-02
ini	7.12913871e-02
daftar	-6.00991994e-02
ada	-1.21092416e-01
jaki	-7.68785179e-02

Tabel di atas menampilkan proses perubahan data teks menjadi representasi numerik atau vektor, menunjukkan bagaimana kata-kata dalam teks diubah menjadi angka sehingga dapat digunakan dalam model pembelajaran mesin

```
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'))
```

Kode ini mendefinisikan dan menginisialisasi model jaringan saraf konvolusional (*CNN*) untuk pemrosesan teks menggunakan pustaka Keras. Panjang maksimum urutan (`MAX_SEQUENCE_LENGTH`) diatur ke 50, yang menentukan panjang setiap input teks. Jumlah kata unik dalam kosakata ditentukan dengan menambahkan 1 pada jumlah kata yang ada dalam `tokenizer.word_index`.

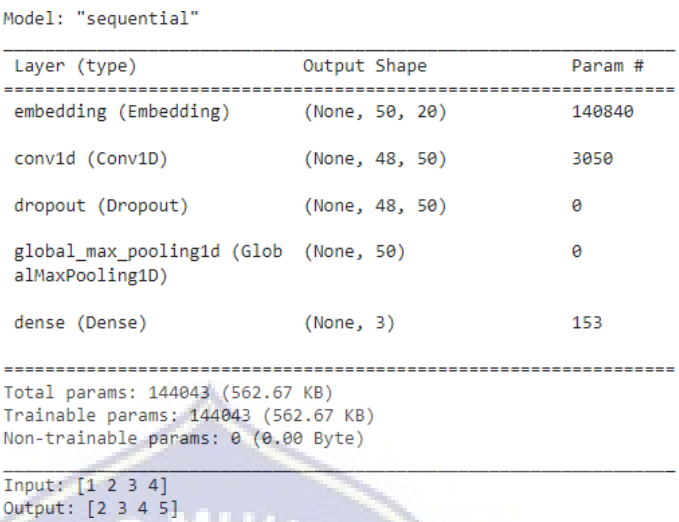
Selanjutnya, sebuah matriks *embedding* diinisialisasi dengan ukuran `num_unique_words` x 20, di mana 20 adalah dimensi dari vektor *embedding*. Ini adalah matriks acak untuk sementara waktu, yang seharusnya digantikan dengan matriks *embedding* yang sebenarnya.

Model *Sequential* diinisialisasi, dan lapisan `Embedding` ditambahkan sebagai lapisan pertama. Lapisan ini menggunakan `embedding_matrix` yang telah diinisialisasi sebelumnya untuk memetakan indeks kata menjadi vektor berdimensi 20, dengan panjang input yang ditentukan oleh `MAX_SEQUENCE_LENGTH`. Lapisan ini dapat dilatih (`trainable=True`). Lapisan `Conv1D` ditambahkan dengan 50 filter dan ukuran jendela 3, menggunakan fungsi aktivasi ReLU. Lapisan ini akan mengaplikasikan konvolusi pada input teks untuk menangkap fitur spasial dalam data.

```
model.add(Dropout(0.25))
# Menambahkan lapisan GlobalMaxPooling1D
model.add(GlobalMaxPooling1D())
# Menambahkan lapisan Dense untuk output
model.add(Dense(3, 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)
```

Kode ini menyelesaikan definisi dan konfigurasi model jaringan saraf konvolusional (*CNN*) untuk pemrosesan teks serta menyertakan contoh fungsi untuk manipulasi data. Lapisan `Dropout` dengan rasio 0.25 ditambahkan setelah lapisan konvolusi untuk mengurangi risiko *overfitting* dengan menonaktifkan 25% neuron secara acak selama pelatihan. Selanjutnya, lapisan `GlobalMaxPooling1D` ditambahkan untuk merangkum fitur penting dari output lapisan konvolusi dengan mengambil nilai maksimum sepanjang dimensi temporal. Lapisan `Dense` dengan 3 unit dan fungsi aktivasi `softmax` ditambahkan sebagai lapisan output untuk klasifikasi, di mana 3 unit mewakili jumlah kelas dan `softmax` mengonversi output menjadi probabilitas kelas. Model dirangkum dengan `model.summary()` untuk memberikan ringkasan struktur model. Model kemudian dikompilasi menggunakan *optimizer* Adam dan fungsi kehilangan `categorical_crossentropy`, dengan akurasi sebagai metrik evaluasi. Terakhir, kode menyediakan contoh fungsi `split_input`, yang membagi urutan data menjadi pasangan input dan output, misalnya mengubah urutan `[1, 2, 3, 4, 5]` menjadi input `[1, 2, 3, 4]` dan output `[2, 3, 4, 5]`, yang berguna untuk pelatihan model berbasis sekuens.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 50, 20)            140840

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

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

 global_max_pooling1d (Glob  (None, 50)                0
 alMaxPooling1D)

 dense (Dense)               (None, 3)                 153

=================================================================
Total params: 144043 (562.67 KB)
Trainable params: 144043 (562.67 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
Input: [1 2 3 4]
Output: [2 3 4 5]
```

Gambar 9. Struktur Model CNN

Gambar di atas menunjukkan struktur model *CNN* yang terdiri dari beberapa lapisan:

1. ***Embedding Layer*** : Mengubah kata menjadi vektor dengan dimensi (50, 20), yang berarti panjang urutan 50 dan *embedding* berukuran 20.
2. ***Conv1D*** : Menggunakan filter konvolusi untuk menangkap pola dalam data sekuensial, menghasilkan *output* (48, 50).
3. ***Dropout*** : Mencegah *overfitting* dengan secara acak mengabaikan beberapa unit selama pelatihan.
4. ***Global Max Pooling 1D*** : Mengambil nilai maksimum dari tiap filter, mengurangi dimensi menjadi (50).
5. ***Dense Layer*** : Lapisan akhir untuk klasifikasi dengan 3 kelas.

Model ini menggunakan total 144,043 parameter untuk mendeteksi fitur penting dari teks dan mengklasifikasikannya. Input `[1,2,3,4]` dan Output `[2,3,4,5]` ini mungkin adalah contoh tokenized data yang menunjukkan pergeseran satu posisi, sering kali digunakan dalam pengaturan pemrosesan sekuens.

```
from keras.callbacks import EarlyStopping,
ModelCheckpoint
# Membuat callback EarlyStopping dan ModelCheckpoint
```

```
model_checkpoint =
ModelCheckpoint('/content/drive/Othercomputers/My
Laptop/Documents/SKRIPSI/Aryo - CNN -
FastText/CNN_FastText.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)
```

Kode ini digunakan untuk melatih model dengan dua *callback* untuk memantau dan menyimpan model selama proses pelatihan. Pertama, sebuah callback `ModelCheckpoint` diinisialisasi untuk menyimpan model yang memiliki kinerja terbaik pada data validasi dengan nama file yang ditentukan. Parameter `monitor` diatur untuk memantau akurasi validasi (`val_acc`), `save_best_only=True` memastikan hanya model terbaik yang disimpan, dan `verbose=2` memberikan informasi detail selama pelatihan. Model kemudian dilatih menggunakan `model.fit`, dengan data pelatihan dan data validasi yang telah ditentukan, berlangsung selama 50 *epoch* dengan ukuran *batch* 32. *Callback* `ModelCheckpoint` diterapkan untuk menyimpan model terbaik. Setelah pelatihan selesai, riwayat pelatihan diperiksa untuk metrik yang dilacak, dan nilai `val_loss` dan `val_acc` dicetak untuk menunjukkan kerugian dan akurasi model pada data validasi.

```
# 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()
```

Kode ini digunakan untuk memvisualisasikan hasil pelatihan model dengan grafik kerugian (*loss*) dan akurasi. Pertama, riwayat pelatihan diambil dari objek `history` dan disimpan dalam `history_dict`, yang menyimpan nilai-nilai untuk setiap metrik pelatihan seperti `loss`, `val_loss`, `acc`, dan `val_acc`. Nilai-nilai ini diekstrak untuk digunakan dalam *plot*. Rentang *epoch* dibuat berdasarkan jumlah *epoch* pelatihan. Dengan menggunakan `matplotlib`, dua subplot dalam satu gambar dibuat: *subplot* pertama menunjukkan grafik kerugian pelatihan dan validasi terhadap *epoch*, dengan kerugian pelatihan ditampilkan sebagai titik biru dan kerugian validasi sebagai garis biru; *subplot* kedua menunjukkan grafik akurasi pelatihan dan validasi terhadap *epoch*, dengan akurasi pelatihan sebagai titik biru dan akurasi validasi sebagai

garis biru. Grafik ini memberikan gambaran visual tentang bagaimana kerugian dan akurasi model berubah selama proses pelatihan.

```
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)
```

Kode ini digunakan untuk mengevaluasi performa model klasifikasi menggunakan beberapa metrik evaluasi, seperti *F1 Score*, *Precision*, dan *Recall*, serta menampilkan laporan klasifikasi terperinci. Pertama, prediksi dilakukan pada data pengujian (`X_test`) menggunakan model terlatih, menghasilkan probabilitas untuk setiap sampel. Probabilitas ini dikonversi menjadi label biner (0 atau 1) dengan menerapkan *threshold* 0.5. Proses ini dilakukan dengan memeriksa apakah nilai probabilitas lebih besar dari 0.5, dan hasilnya disimpan dalam `predicted_labels`. Label sebenarnya (`true_labels`) diambil dari `Y_test`.

Setelah itu, metrik evaluasi dihitung untuk menilai kinerja model. *F1 Score* dihitung menggunakan `f1_score()` dari `sklearn.metrics`, dengan `average='weighted'` yang memperhitungkan ketidakseimbangan antar kelas. *Precision* dan *Recall* dihitung menggunakan `*precision*_score()`

dan `*recall*_score()` dengan cara yang sama. Hasil dari setiap metrik dicetak untuk memberikan wawasan tentang kualitas prediksi model.

Selanjutnya, laporan klasifikasi lengkap dicetak menggunakan `classification_report()`, yang menyediakan metrik evaluasi untuk setiap kelas dalam data, termasuk *precision*, *recall*, *F1 Score*, dan *support* (jumlah sampel per kelas). Terakhir, hasil prediksi dalam bentuk *array* dicetak, yang memungkinkan pengguna untuk membandingkan label sebenarnya (`true_labels`) dengan label yang diprediksi (`predicted_labels`). Ini membantu dalam memahami performa model dan mengidentifikasi area yang memerlukan perbaikan.

```
print("Panjang Tes Ulasan:", len(test['ULASAN']))
print("Panjang X_test:", len(X_test))
print("Panjang Y_test:", len(Y_test))
print("Panjang true_labels:", len(true_labels))
print("Panjang predicted_labels:",
len(predicted_labels))
```

Kode ini digunakan untuk memverifikasi konsistensi ukuran data pengujian dalam proses evaluasi model. Pertama, panjang ulasan teks dalam dataset pengujian asli dicetak menggunakan `print("Panjang Tes Ulasan:", len(test['ULASAN']))`, memberikan gambaran tentang jumlah sampel dalam data pengujian. Selanjutnya, panjang array `X_test`, yang berisi fitur-fitur ulasan setelah melalui pra-pemrosesan seperti tokenisasi dan *padding*, dicetak dengan `print("Panjang X_test:", len(X_test))`. Kemudian, panjang `Y_test`, yang berisi label asli untuk setiap ulasan dalam `X_test`, dicetak menggunakan `print("Panjang Y_test:", len(Y_test))`.

Setelah itu, panjang `true_labels`, yang diambil langsung dari `Y_test`, dicetak dengan `print("Panjang true_labels:", len(true_labels))`. Terakhir, panjang `predicted_labels`, yang dihasilkan setelah model melakukan prediksi pada `X_test`, dicetak menggunakan `print("Panjang predicted_labels:", len(predicted_labels))`. Pernyataan-pernyataan ini memastikan bahwa panjang `X_test`, `Y_test`, `true_labels`, dan

`predicted_labels` sesuai satu sama lain, yang penting untuk memastikan integritas dan konsistensi data sebelum melakukan evaluasi atau analisis lebih lanjut terhadap hasil prediksi model.

```
import pandas as pd
from sklearn.metrics import
classification_report,f1_score, precision_score,
recall_score
# Melakukan prediksi dengan softmax (misalnya, jika
menggunakan TensorFlow/Keras)
predictions = model.predict(X_test)
predicted_labels = predictions.argmax(axis=1)  #
Mengambil kelas dengan probabilitas tertinggi sebagai
prediksi
# Pastikan true_labels adalah dalam bentuk indeks kelas
yang sama dengan predicted_labels
true_labels = Y_test.argmax(axis=1)  # Jika Y_test
adalah dalam bentuk one-hot encoded, konversi ke indeks
kelas
# 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))
# Ambil ulasan, label sebenarnya, dan prediksi label
# Pastikan panjang semua array sama
min_length = min(len(test['ULASAN']), len(true_labels),
len(predicted_labels))
test_results = pd.DataFrame({
    'Ulasan': test['ULASAN'].values[:min_length],
#Gunakan min_length untuk memastikan panjang yang sama
    'Label Sebenarnya': true_labels[:min_length],
    'Prediksi': predicted_labels[:min_length]
})
# Klasifikasi label 'Negatif', 'Positif', dan 'Netral'
berdasarkan nilai
```

```
def classify_label(label):
    if label == 0:
        return 'Negatif'
    elif label == 1:
       return 'Netral'
    else:
        return 'Positif'
# 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('/content/drive/Othercomputers/My
Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/HASIL
PREDIKSI.xlsx', index=False)
# Tampilkan hasil
print("\nHasil Prediksi:\n", test_results)
print("Data berhasil diekspor ke
'hasil_prediksi2.xlsx'.")
```

Kode ini bertujuan untuk mengevaluasi kinerja model klasifikasi yang telah dilatih dan mengekspor hasil prediksi ke dalam file Excel. Pada awalnya, prediksi dibuat dengan model menggunakan data pengujian `X_test`, yang memberikan probabilitas untuk setiap kelas. Fungsi `argmax(axis=1)` digunakan pada output prediksi ini untuk menentukan indeks kelas dengan probabilitas tertinggi dan menyimpannya dalam variabel `predicted_labels`. Langkah ini mengkonversi probabilitas *softmax* menjadi label kelas yang dapat digunakan untuk perbandingan langsung dengan label sebenarnya. Label sebenarnya (`true_labels`), yang diambil dari `Y_test`, juga dikonversi dari bentuk *one-hot encoding* ke indeks kelas yang sesuai menggunakan `argmax(axis=1)`, sehingga keduanya memiliki format yang sama untuk evaluasi.

Selanjutnya, kode ini menghitung tiga metrik evaluasi: *F1 Score*, *Precision*, dan *Recall*. Ketiga metrik ini dihitung dengan metode `f1_score`, `*precision*_score`, dan `*recall*_score` dari `sklearn.metrics`, masing-masing dengan opsi `average='weighted'`, yang digunakan untuk menangani kemungkinan ketidakseimbangan antar kelas dalam data.

Nilai-nilai ini dicetak untuk memberikan wawasan tentang performa model, dan laporan klasifikasi yang lebih rinci dihasilkan dengan menggunakan `classification_report()`, yang memberikan metrik untuk setiap kelas, termasuk *precision*, *recall*, *F1 Score*, dan *support*.

Kemudian, kode ini membangun *DataFrame* `test_results` yang menyimpan informasi ulasan dari `test['ULASAN']`, serta label sebenarnya dan prediksi label. Untuk memastikan bahwa panjang dari semua elemen yang digunakan dalam *DataFrame* konsisten, `min_length` dihitung sebagai nilai minimum dari panjang `test['ULASAN']`, `true_labels`, dan `predicted_labels`. Hal ini untuk memastikan bahwa jumlah data yang diambil sama pada semua variabel tersebut dan mencegah adanya kesalahan data.

Selain itu, fungsi `classify_label` di definisikan untuk mengubah indeks kelas menjadi label teks yang lebih mudah dipahami, seperti 'Negatif', 'Netral', dan 'Positif'. Fungsi ini diterapkan pada kolom 'Label Sebenarnya' dan 'Prediksi' dari *DataFrame* `test_results` untuk mengganti indeks numerik dengan label yang lebih intuitif. Setelah *DataFrame* diisi dengan data yang diformat ulang ini, kode menggunakan metode `to_excel()` untuk mengekspor `test_results` ke file Excel dengan nama file yang ditentukan, yaitu `'HASIL PREDIKSI.xlsx'`. Akhirnya, hasil prediksi dicetak ke konsol, dan pesan sukses ditampilkan untuk menginformasikan bahwa data telah berhasil diekspor ke file Excel. Kode ini tidak hanya memfasilitasi evaluasi performa model tetapi juga memudahkan penyimpanan dan interpretasi hasil prediksi klasifikasi, sehingga pengguna dapat menganalisis data dengan lebih efektif dan menyajikan hasil dalam format yang dapat diakses dan mudah dibaca.

E. Pengujian dan Hasil Metode

Pengujian data terdiri dari 3.199 ulasan dari aplikasi JAKI (Jakarta Kini). Data latih dan uji dibagi dengan perbandingan 90:10, 80:20, dan 70:30, untuk menunjukkan bahwa data latih akan digunakan untuk

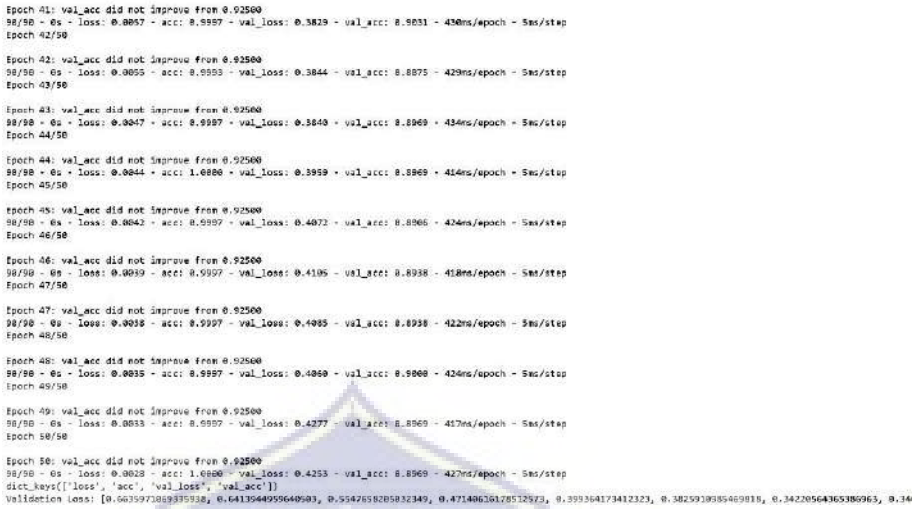
melatih model dan data uji akan digunakan untuk menguji kinerja model yang dilatih. Berikut tampilan hasil akurasi terhadap pengujian data berdasarkan pembagian dengan masing-masing perbandingan baik yang menggunakan kombinasi antara *Word Embedding FastText* dan *CNN* maupun yang hanya menggunakan *CNN* saja :

1. Pengujian *Word Embedding FastText* dan *CNN*
    a. Pembagian data 90:10
        1) *Epoch*

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

Serangkaian baris teks yang berasal dari proses pelatihan model pembelajaran mesin digambarkan di sini. Setiap baris dimulai dengan *"epoch"*, diikuti oleh pecahan yang menunjukkan kemajuan pelatihan (misalnya, 1/50, 2/50, dst.) dan mencakup berbagai metrik untuk setiap *epoch*, termasuk *loss*, akurasi (*acc*), *val_loss*, dan *val_acc*. Angka-angka yang mengikuti metrik ini menunjukkan nilai pada titik tertentu dalam proses pelatihan. Di akhir setiap baris, jumlah sampel yang diproses per langkah dan total langkah per epoch disebutkan.

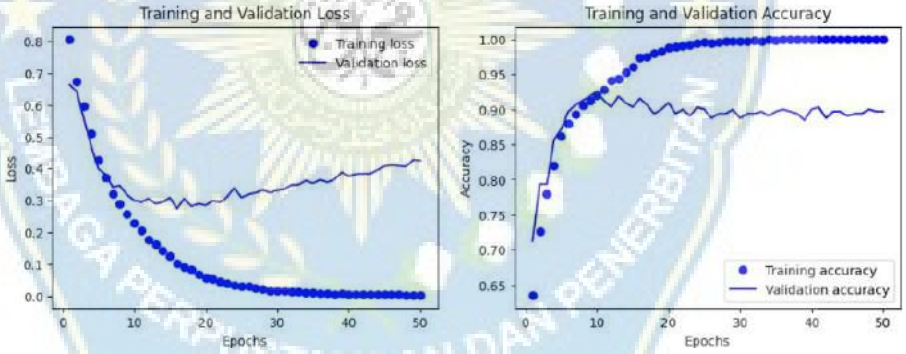
Pada gambar diatas menunjukkan bahwa nilai pada iterasi atau epoch ke-10 di dapatkan *accuracy* tertinggi yaitu sebesar 92%.



Gambar 11. Proses Epoch 41-50 (90:10)

Sedangkan pada gambar diatas menunjukkan bahwa nilai pada iterasi akhir atau epoch 41-50 di dapatkan *accuracy* yang konsisten di sebesar 92%.

2) Grafik



Gambar 12. Grafik 90:10

Gambar tersebut menampilkan dua grafik yang memvisualisasikan *loss* dan akurasi selama pelatihan model. Grafik kiri menunjukkan bahwa *training loss* terus menurun, sementara *validation loss* mulai meningkat setelah beberapa *epoch*, menandakan kemungkinan *overfitting*. Di grafik kanan, *training accuracy* meningkat mendekati 1.0, tetapi *validation accuracy* stagnan dan sedikit menurun setelah titik tertentu. Ini menunjukkan bahwa model semakin baik pada data pelatihan namun kehilangan kemampuan untuk menggeneralisasi pada data validasi.

3) Prediksi

```
F1 Score: 0.8943664702148084
Precision: 0.8923978442728442
Recall: 0.896875
              precision    recall  f1-score   support

           0       0.92      0.94      0.93       217
           1       0.14      0.10      0.12        10
           2       0.91      0.88      0.90        93

   micro avg       0.90      0.90      0.90       320
   macro avg       0.66      0.64      0.65       320
weighted avg       0.89      0.90      0.89       320
 samples avg       0.90      0.90      0.90       320
```

Gambar 13. Hasil Prediksi 90:10

Gambar tersebut menampilkan evaluasi model klasifikasi dengan metrik *precision*, *recall*, dan *f1-score*:

Metrik Evaluasi Model :

- **Kelas 0 (Negatif)** : Kinerja sangat baik dengan *f1-score* 93%.
- **Kelas 1 (Netral)** : Kinerja buruk dengan *f1-score* 12%, kemungkinan karena sampel sedikit.
- **Kelas 2 (Positif)** : Kinerja baik dengan *f1-score* 90%.

Rata-rata Keseluruhan :

- **Micro Avg**: 0.90 untuk *precision*, *recall*, dan *f1-score*, menghitung keseluruhan rata-rata berdasarkan jumlah total prediksi yang benar.
- **Macro Avg**: 0.66 untuk *precision*, 0.64 untuk *recall*, dan 0.65 untuk *f1-score*, menghitung rata-rata untuk setiap kelas tanpa mempertimbangkan jumlah sampel.
- **Weighted Avg**: Menghitung rata-rata dengan mempertimbangkan jumlah sampel di setiap kelas, menunjukkan kinerja model yang lebih representatif dari distribusi kelas.

4) Klasifikasi

F1 Score: 0.8941622218487953
Precision: 0.8918398085585586
Recall: 0.896875

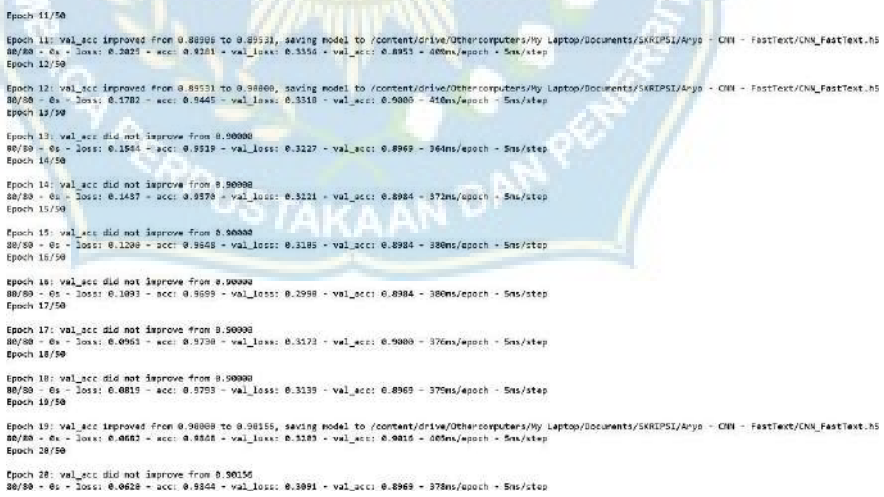
	precision	recall	f1-score	support
0	0.92	0.94	0.93	217
1	0.12	0.10	0.11	10
2	0.91	0.88	0.90	93
accuracy			0.90	320
macro avg	0.65	0.64	0.65	320
weighted avg	0.89	0.90	0.89	320

Gambar 14. Hasil Klasifikasi Label 90:10

Hasil evaluasi menunjukkan model bekerja baik untuk kelas 0 dan 2 dengan *f1-score* masing-masing 93% dan 90%, namun kinerja sangat buruk pada kelas 1 dengan *f1-score* hanya 11%, kemungkinan disebabkan oleh jumlah sampel yang sedikit. Secara keseluruhan, model memiliki akurasi 90%, tetapi performa tidak merata di semua kelas.

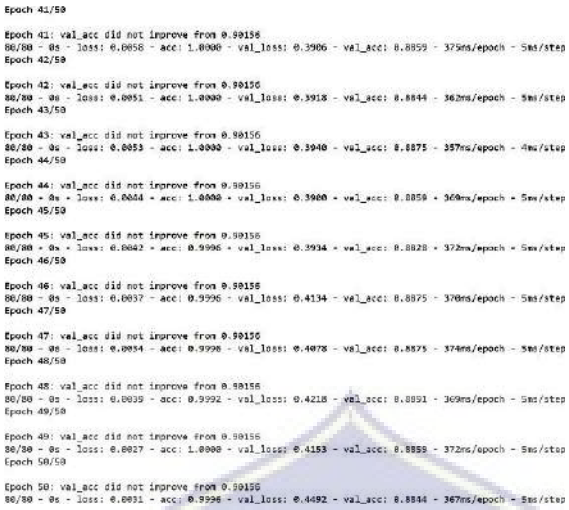
b. Pembagian data 80:20

1) *Epoch*



Gambar 15. Proses Epoch 11-20 (80:20)

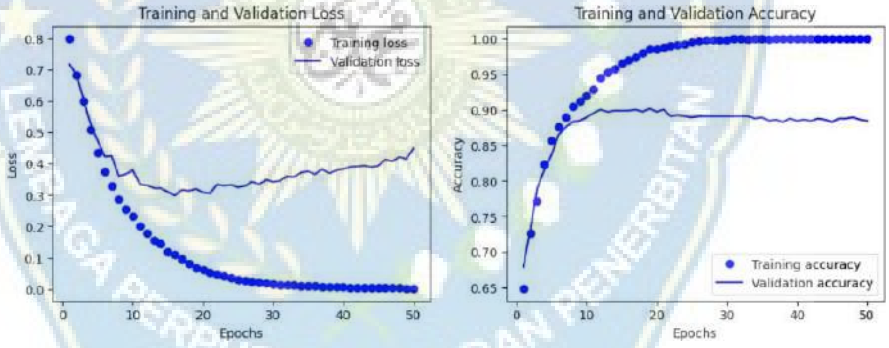
Pada gambar diatas menunjukkan bahwa nilai pada iterasi awal atau epoch ke-12 di dapatkan *accuracy* tertinggi yaitu sebesar 90%.



Gambar 16. Proses Epoch 41-50 (80:20)

Sedangkan pada gambar diatas menunjukkan bahwa nilai pada iterasi akhir atau epoch 41-50 di dapatkan *accuracy* yang sedikit menurun dikisaran 88%.

2) Grafik



Gambar 17. Grafik 80:20

Dalam grafik *loss*, kita melihat bahwa *training loss* terus menurun karena model belajar dari 80% data pelatihan, sementara *validation loss* (untuk 20% data validasi) mulai meningkat seiring waktu. Karena kemampuan model untuk memprediksi data pelatihan yang lebih baik maka *training accuracy* terus meningkat, seperti yang ditunjukkan pada grafik *accuracy*. Namun, *validation accuracy* mencapai puncaknya sekitar *epoch* ke-12, dan kemudian tetap stabil atau sedikit menurun. Ini menunjukkan bahwa model mungkin terlalu berkonsentrasi pada data

pelatihan dan mulai kehilangan kemampuan untuk menggeneralisasi pada data validasi. Kondisi ini disebut *overfitting*.

3) Prediksi

```
F1 Score: 0.8812742197151471
Precision: 0.8882456893172694
Recall: 0.8796875
              precision    recall  f1-score   support

           0       0.90      0.96      0.93       415
           1       0.32      0.29      0.31        31
           2       0.96      0.80      0.87       194

   micro avg       0.89      0.88      0.88       640
   macro avg       0.73      0.68      0.70       640
weighted avg       0.89      0.88      0.88       640
 samples avg       0.88      0.88      0.88       640
```

Gambar 18. Hasil Prediksi 80:20

Gambar tersebut menampilkan evaluasi model klasifikasi dengan metrik *precision*, *recall*, dan *f1-score*:

Metrik Evaluasi Model :

- **Kelas 0 (Negatif)** : Kinerja sangat baik dengan *f1-score* 93%.
- **Kelas 1 (Netral)** : Kinerja buruk dengan *f1-score* 31%, kemungkinan karena sampel sedikit.
- **Kelas 2 (Positif)** : Kinerja baik dengan *f1-score* 87%.

Rata-rata Keseluruhan :

- **Micro Avg**: 0.89 untuk *precision*, 0.88 untuk *recall*, dan 0.88 untuk *f1-score*, menghitung keseluruhan rata-rata berdasarkan jumlah total prediksi yang benar.
- **Macro Avg**: 0.73 untuk *precision*, 0.68 untuk *recall*, dan 0.70 untuk *f1-score*, menghitung rata-rata untuk setiap kelas tanpa mempertimbangkan jumlah sampel.
- **Weighted Avg**: Menghitung rata-rata dengan mempertimbangkan jumlah sampel di setiap kelas, menunjukkan kinerja model yang lebih representatif dari distribusi kelas.

4) Klasifikasi

F1 Score: 0.8828379205975093
Precision: 0.8859194378866817
Recall: 0.884375

	precision	recall	f1-score	support
0	0.90	0.96	0.93	415
1	0.33	0.32	0.33	31
2	0.95	0.81	0.88	194
accuracy			0.88	640
macro avg	0.73	0.70	0.71	640
weighted avg	0.89	0.88	0.88	640

Gambar 19. Hasil Klasifikasi Label 80:20

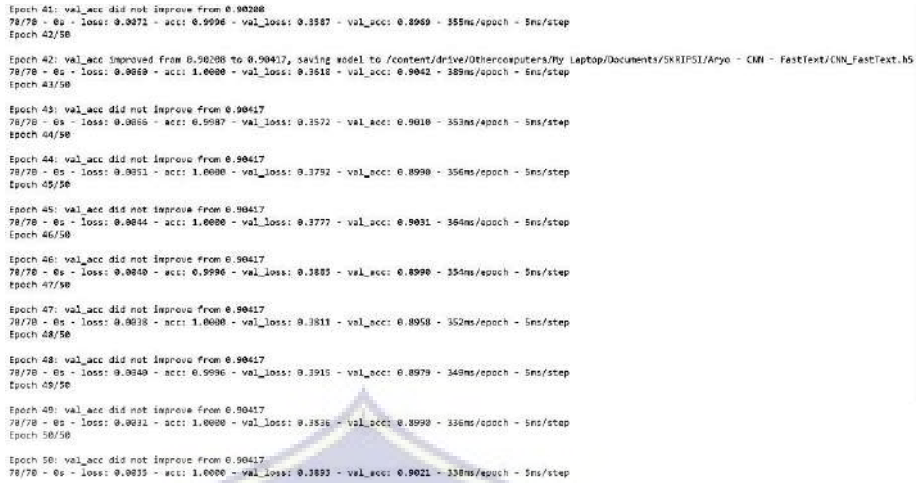
Model ini bekerja sangat baik pada kelas 0 dan 2 dengan F1-score masing-masing 93% dan 88%, namun kinerja pada kelas 1 sangat rendah (F1-score 33%) karena jumlah sampel yang sedikit. Secara keseluruhan, akurasi model adalah 88%, menunjukkan performa yang cukup baik meskipun ada ketidakseimbangan antara kelas.

c. Pembagian data 70:30

1) *Epoch*

Gambar 20. Proses Epoch 1-10 (70:30)

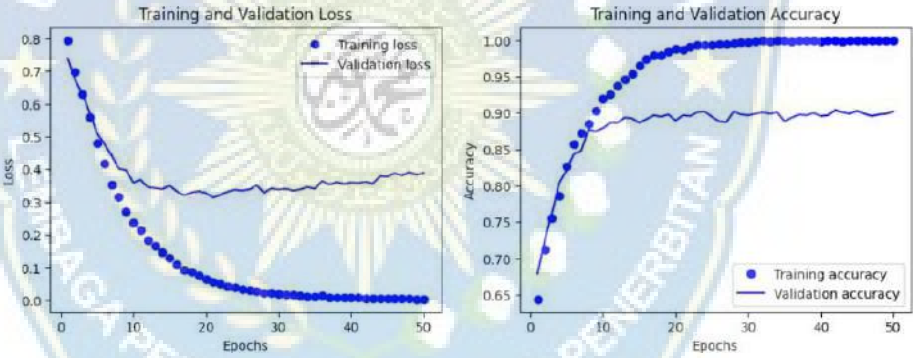
Pada gambar diatas menunjukkan bahwa nilai pada iterasi awal atau epoch 1-10 di dapatkan *accuracy* tertinggi yaitu sebesar 87%.



Gambar 21. Proses Epoch 41-50 (70:30)

Sedangkan pada gambar diatas menunjukkan bahwa nilai pada iterasi akhir atau epoch ke-42 di dapatkan *accuracy* meningkat yaitu sebesar 90%.

2) Grafik



Gambar 22. Grafik 70:30

Tidak berbeda jauh dengan dua grafik sebelumnya yang membandingkan kinerja model pada data pelatihan dan data validasi selama 50 *epoch*. terlihat bahwa *training loss* terus menurun, dan *validation loss* juga meningkat setelah beberapa *epoch*. Namun pada perbandingan 70:30 memperlihatkan bahwa *training accuracy* terus meningkat hingga mendekati 100%, sedangkan *validation accuracy* menunjukkan kondisi yang sama seperti kedua perbandingan sebelumnya.

3) Prediksi

```
F1 Score: 0.8886694151495232
Precision: 0.8936015342692969
Recall: 0.8895833333333333
              precision    recall  f1-score   support

           0       0.91      0.97      0.94       622
           1       0.29      0.15      0.19        48
           2       0.95      0.84      0.89       290

   micro avg       0.91      0.89      0.90       960
   macro avg       0.72      0.65      0.68       960
weighted avg       0.89      0.89      0.89       960
 samples avg       0.89      0.89      0.89       960
```

Gambar 23. Hasil Prediksi 70:30

Gambar tersebut menampilkan evaluasi model klasifikasi dengan metrik *precision*, *recall*, dan *f1-score*:

Metrik Evaluasi Model :

- **Kelas 0 (Negatif)** : Kinerja sangat baik dengan *f1-score* 94%.
- **Kelas 1 (Netral)** : Kinerja buruk dengan *f1-score* 19%, kemungkinan karena sampel sedikit.
- **Kelas 2 (Positif)** : Kinerja baik dengan *f1-score* 89%.

Rata-rata Keseluruhan :

- **Micro Avg**: 0.91 untuk *precision*, 0.89 untuk *recall*, dan 0.90 untuk *f1-score*, menghitung keseluruhan rata-rata berdasarkan jumlah total prediksi yang benar.
- **Macro Avg**: 0.72 untuk *precision*, 0.65 untuk *recall*, dan 0.68 untuk *f1-score*, menghitung rata-rata untuk setiap kelas tanpa mempertimbangkan jumlah sampel.
- **Weighted Avg**: Menghitung rata-rata dengan mempertimbangkan jumlah sampel di setiap kelas, menunjukkan kinerja model yang lebih representatif dari distribusi kelas.

4) Klasifikasi

```
F1 Score: 0.8933458321826132
Precision: 0.8889789186430977
Recall: 0.9020833333333333
              precision    recall  f1-score   support

           0       0.91      0.97      0.94       622
           1       0.31      0.17      0.22        48
           2       0.94      0.87      0.90       290

    accuracy                           0.90       960
   macro avg       0.72      0.67      0.69       960
weighted avg       0.89      0.90      0.89       960
```

Gambar 24. Hasil Klasifikasi Label 70:30

Model ini memiliki performa yang sangat baik pada kelas 0 dan 2 dengan F1-score masing-masing 0.94 dan 0.90, menunjukkan bahwa model mampu memprediksi kedua kelas ini dengan baik. Namun, performa pada kelas 1 sangat rendah (F1-score 0.22) karena jumlah sampel yang sedikit dan deteksi yang buruk. Secara keseluruhan, model memiliki akurasi 90%, dengan precision dan recall yang tinggi, tetapi ketidakseimbangan kelas mempengaruhi kinerja pada kelas minoritas.

2. Pengujian *Convolutional Neural Network* (*CNN*)

a. Pembagian data 90:10

```
Epoch 1/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 7s 23ms/step - accuracy: 0.6165 - loss: 0.8490 - val_accuracy: 0.8500 - val_loss: 0.4324
Epoch 2/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 4s 23ms/step - accuracy: 0.8917 - loss: 0.3647 - val_accuracy: 0.8813 - val_loss: 0.3540
Epoch 3/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 6s 32ms/step - accuracy: 0.9237 - loss: 0.2419 - val_accuracy: 0.9000 - val_loss: 0.3782
Epoch 4/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 9s 26ms/step - accuracy: 0.9428 - loss: 0.1676 - val_accuracy: 0.8813 - val_loss: 0.4321
Epoch 5/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 7s 38ms/step - accuracy: 0.9525 - loss: 0.1173 - val_accuracy: 0.8906 - val_loss: 0.4469
Epoch 6/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 8s 24ms/step - accuracy: 0.9703 - loss: 0.0889 - val_accuracy: 0.8844 - val_loss: 0.5157
Epoch 7/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 6s 34ms/step - accuracy: 0.9745 - loss: 0.0762 - val_accuracy: 0.8844 - val_loss: 0.5539
Epoch 8/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 9s 24ms/step - accuracy: 0.9876 - loss: 0.0436 - val_accuracy: 0.8938 - val_loss: 0.5621
Epoch 9/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 7s 36ms/step - accuracy: 0.9882 - loss: 0.0360 - val_accuracy: 0.8875 - val_loss: 0.5628
Epoch 10/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 8s 24ms/step - accuracy: 0.9917 - loss: 0.0331 - val_accuracy: 0.8969 - val_loss: 0.6431
<keras.src.callbacks.history.History at 0x79a0f02bdb40>
```

Gambar 25. Proses Epoch 1-10 (180/180)

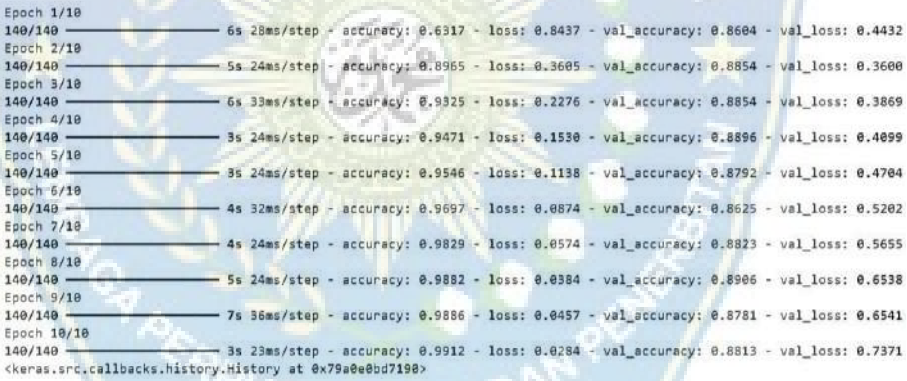
Pada gambar diatas menunjukkan bahwa nilai pada iterasi atau epoch ke-3 di dapatkan *accuracy* tertinggi yaitu sebesar 90%.

b. Pembagian data 80:20

Gambar 26. Proses Epoch 1-10 (160/160)

Pada gambar diatas menunjukkan bahwa nilai pada iterasi atau epoch ke-3 di dapatkan *accuracy* tertinggi yaitu sebesar 88%.

c. Pembagian data 70:30



Gambar 27. Proses Epoch 1-10 (140/140)

Pada gambar diatas menunjukkan bahwa nilai pada iterasi atau epoch ke-8 di dapatkan *accuracy* atau hasil training tertinggi yaitu sebesar 89%.

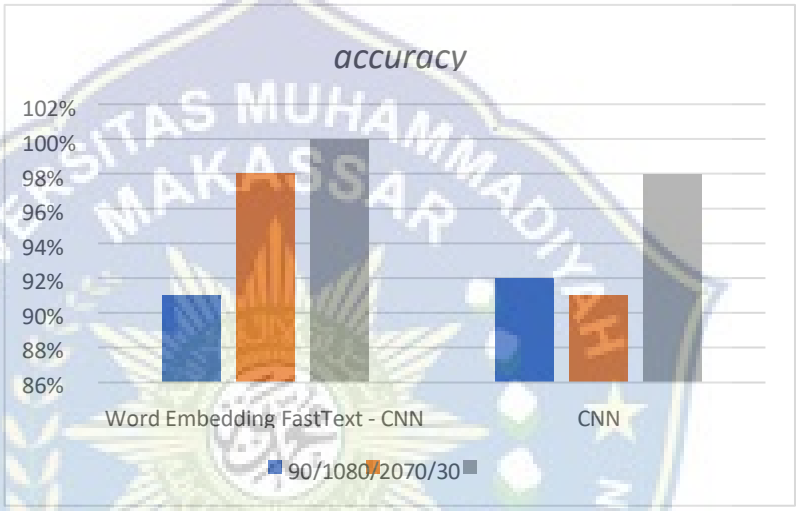
3. Tabel Hasil Pengujian Metode

Berikut tabel yang membandingkan hasil akurasi dan *loss* model pada data validasi dengan menggunakan dua pendekatan yang berbeda yaitu kombinasi ***Word Embedding FastText - CNN*** dan yang hanya

CNN saja, untuk tiga skenario pembagian data (90:10, 80:20, dan 70:30):

Tabel 5. Hasil Perbandingan Accuracy Model

Pembagian Data	*Accuracy*	
	Word Embedding FastText - CNN	*CNN*
90:10	91%	92%
80:20	98%	91%
70:30	100%	98%

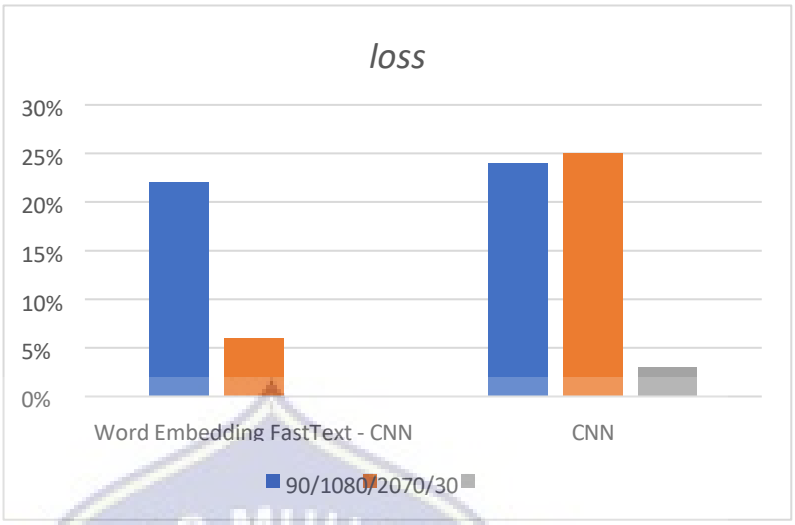


Gambar 28. Grafik Diagram Accuracy

Tabel dan grafik diagram diatas menunjukkan hasil perbandingan kinerja antara kedua model berdasarkan nilai *Accuracy* dengan tiga skenario pembagian data yang berbeda.

Tabel 6. Hasil Perbandingan Loss Model

Pembagian Data	*Loss*	
	Word Embedding FastText - CNN	*CNN*
90:10	22%	24%
80:20	6%	25%
70:30	0,06%	3%

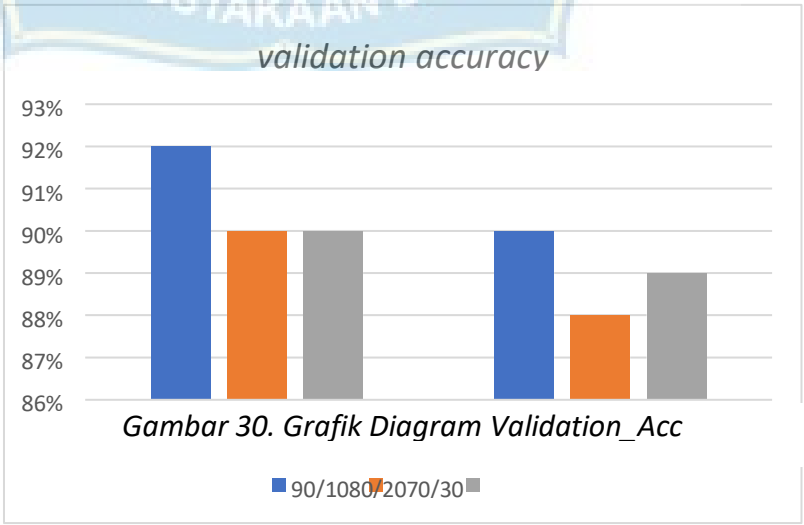


Gambar 29. Grafik Diagram Loss

Sedangkan tabel dan grafik diagram diatas menunjukkan hasil perbandingan kinerja antara kedua model berdasarkan nilai *Loss* dengan tiga skenario pembagian data yang berbeda.

Tabel 7. Hasil Perbandingan Validation Accuracy Model

Pembagian Data	*Validation Accuracy*	
	Word Embedding FastText - CNN	*CNN*
90:10	92%	90%
80:20	90%	88%
70:30	90%	89%

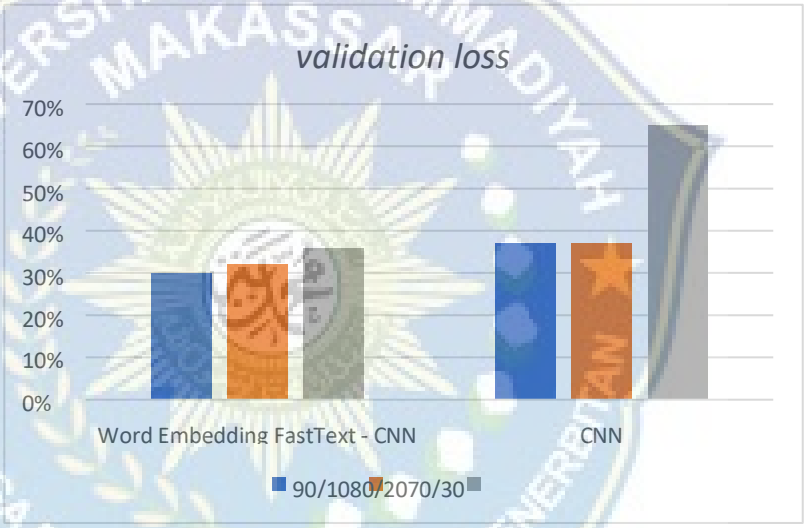


Gambar 30. Grafik Diagram Validation_Acc

Tabel dan grafik diagram diatas menunjukkan hasil perbandingan kinerja antara kedua model berdasarkan nilai *Validation Accuracy* dengan tiga skenario pembagian data yang berbeda.

Tabel 8. Hasil Perbandingan Validation Loss Model

Pembagian Data	*Validation Loss*	
	Word Embedding FastText - CNN	*CNN*
90:10	30%	37%
80:20	32%	37%
70:30	36%	65%



Gambar 31. Grafik Diagram Validation_Los

Sedangkan tabel dan grafik diagram diatas menunjukkan hasil perbandingan kinerja antara kedua model berdasarkan nilai *Validation Loss* dengan tiga skenario pembagian data yang berbeda.

Berdasarkan semua hasil pengujian yang telah dilakukan, perbandingan antara model kombinasi "*Word Embedding FastText – CNN"* dan model "*CNN"* saja menunjukkan bahwa model kombinasi cenderung memiliki kinerja yang lebih baik dengan rata-rata 91% - 100% pada nilai *accuracy* dan 90% - 92% pada nilai *validation accuracy*. Sedangkan yang hanya menggunakan model "*CNN"* memiliki rata-rata

92% - 98% pada nilai *accuracy* dan 88% - 90% pada nilai *validation accuracy*. Meskipun accuracy pada data pelatihan bisa sangat tinggi, *validation accuracy* memberikan gambaran yang lebih realistis tentang seberapa baik model akan bekerja pada data yang tidak terlihat sebelumnya. Perbedaan besar antara *accuracy* dan *validation accuracy* bisa menjadi tanda *overfitting*, di mana model bekerja sangat baik pada data pelatihan tetapi kurang baik pada data baru.

Namun ketika melihat nilai *loss*, model "*Word Embedding* ***FastText*** *– CNN*" konsisten menunjukkan nilai *loss* yang lebih rendah dibandingkan dengan model "*CNN*" saja. Begitupun pada *validation loss*, model kombinasi "*Word Embedding FastText – CNN*" lebih efisien dan stabil, dengan *error* prediksi yang lebih rendah. Perbedaan yang cukup signifikan pada pembagian data 70:30 (36% vs 65%).

4. Hasil Prediksi Model

Tabel 9. Hasil Prediksi

Ulasan	Label Sebenarnya	Label Prediksi	Label Klasifikasi
sangat mempermudah kerja RT RW dan dengan cepat mengetahui keadaan jakarta saat ini	Positif	Positif	True
aplikasi yang sangat bermanfaat bagi warga jakarta semoga lancar penggunaannya	Positif	Positif	True
mau daftar selalu gagal terus bilangnya captcha is not valid lemot parah	Negatif	Negatif	True

ini apk kenapa sih sinyal bagus tapi sistem bilang kesalahan pada sinyal	Negatif	Negatif	True
saya lihat dulu kalau bagus saya kasih bintang lima	Netral	Netral	True
anak sekolah jkt disuruh ngisi rafidtest online di apk ini saya bingung	Netral	Negatif	False
aplikasi ini sangatttttt burukkkkkk untuk pendaftaran vaksin	Negatif	Positif	False
aplikasi jaki sangat membantu saya berterimakasih	Positif	Negatif	False

Secara keseluruhan, model yang menggabungkan Word Embedding FastText dengan CNN menunjukkan kinerja yang baik dalam mengklasifikasikan sentimen ulasan pengguna. Namun, kesalahan dalam beberapa kasus mengindikasikan adanya ruang untuk peningkatan, terutama dalam menangani bahasa informal dan ulasan yang lebih ambigu. Pembahasan ini juga menyoroti pentingnya validasi model pada dataset yang lebih beragam untuk mengurangi kesalahan prediksi dan meningkatkan akurasi klasifikasi sentimen.

BAB V
PENUTUP

A. Kesimpulan

Berdasarkan pengujian yang telah dilakukan dalam penelitian ini, beberapa kesimpulan dapat ditarik sebagai berikut:

1. Dataset dengan 3199 ulasan pada aplikasi JAKI diuji dengan tiga kategori yaitu positif, negatif dan netral. Hasil pengujian menunjukkan bahwa kombinasi "*Word Embedding FastText – CNN*" efektif dalam menganalisis sentimen. Dimana model yang menggunakan kombinasi ini mencapai nilai *accuracy* tinggi antara 91% - 100% serta degan nilai *loss* yang lebih rendah dan stabil secara konsisten ditunjukkan oleh model kombinasi.
2. Penggunaan model CNN memberikan pengaruh positif dalam menganalisis sentimen yang mampu memproses dan mengidentifikasi pola dalam teks dengan baik terutama dalam hal akurasi. Namun juga memiliki kecenderungan overfitting, yang ditandai dengan nilai *validation loss* yang lebih tinggi antara 37% - 65% dibandingkan dengan *training loss* terutama ketika tidak dikombinasikan dengan teknik *embedding* seperti *FastText*. Hal ini menunjukkan bahwa meskipun *CNN* efektif dalam mengenali pola, tetapi kurang optimal dalam meminimalkan *error* prediksi tanpa dukungan representasi kata yang lebih kaya.

Dengan demikian, dapat disimpulkan bahwa penerapan *Word Embedding FastText* dalam analisis sentimen terhadap pengaruh model *CNN* terbukti efektif dan memberikan manfaat dalam analisis sentimen, terutama dalam meningkatkan stabilitas akurasi model serta meminimalkan kesalahan dalam memprediksi sentimen. ini menunjukkan kemampuannya dalam generalisasi yang lebih baik pada data baru.

B. Saran

Meskipun kombinasi *Word Embedding FastText* dan *CNN* telah terbukti efektif. Berdasarkan hasil penelitian yang telah dilakukan, penelitian selanjutnya disarankan untuk dapat mengeksplorasi penggunaan kombinasi metode lain seperti *Long Short-Term Memory* (LSTM) ataupun *Transformer-based models* seperti BERT yang lebih baik dalam memperbaiki pemahaman konteks dan mengurangi *overfitting* dan meminimalkan *error* prediksi (*loss*) dalam analisis sentimen.

DAFTAR PUSTAKA

Al-Areef, M. H., & Saputra S, K. (2023). Analisis Sentimen Pengguna Twitter Mengenai Calon Presiden Indonesia Tahun 2024 Menggunakan Algoritma LSTM. *Jurnal SAINTIKOM (Jurnal Sains Manajemen Informatika Dan Komputer)*, *22*(2), 270. https://doi.org/10.53513/jis.v22i2.8680

Alghifari, D. R., Edi, M., & Firmansyah, L. (2022). Implementasi Bidirectional LSTM untuk Analisis Sentimen Terhadap Layanan Grab Indonesia. *Jurnal Manajemen Informatika (JAMIKA)*, *12*(2), 89–99. https://doi.org/10.34010/jamika.v12i2.7764

Alvi Hasanah, N., Nanik Suciati, & Diana Purwitasari. (2021). Pemantauan Perhatian Publik terhadap Pandemi COVID-19 melalui Klasifikasi Teks dengan Deep Learning. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *5*(1), 193–202. https://doi.org/10.29207/resti.v5i1.2927

Amalia, P. R., & Winarko, E. (2021). Aspect-Based Sentimen Analysis on Indonesian Restaurant Review Using a Combination of *Convolutional Neural Network* and Contextualized *Word Embedding*. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, *15*(3), 285. https://doi.org/10.22146/ijccs.67306

Andriyanto, D., Said, F., Titiani, F., & Erni, E. (2021). Analisis Kesuksesan Aplikasi Jakarta Kini (JAKI) Menggunakan Model Delone and McLean. *Paradigma - Jurnal Komputer Dan Informatika*, *23*(1), 43–48. https://doi.org/10.31294/p.v23i1.10018

Azhar, K. M., Santoso, I., & Adi, A. (2021). Implementasi Deep Learning Menggunakan Metode *Convolutional Neural Network* Dan Algoritma Yolo Dalam Low Vision. *Transient: Jurnal Ilmiah Teknik Elektro*, *10*(3), 502–509.

Daniati, E., & Utama, H. (2023). Analisis Sentimen Dengan Pendekatan Ensemble Learning Dan *Word Embedding* Pada Twitter. *Journal of Information System Management (JOISM)*, *4*(2), 125–131. https://doi.org/10.24076/joism.2023v4i2.973

Dinata, R. K., Safwandi, S., Hasdyna, N., & Azizah, N. (2020). Analisis K-Means Clustering pada Data Sepeda Motor. *INFORMAL: Informatics Journal*, *5*(1),

10. https://doi.org/10.19184/isj.v5i1.17071

Hermanto, D. T., Setyanto, A., & Luthfi, E. T. (2021). Algoritma LSTM-*CNN* untuk Binary Klasifikasi dengan *Word2Vec* pada Media Online. *Creative Information Technology Journal*, *8*(1), 64. https://doi.org/10.24076/citec.2021v8i1.264

Jihad, M. A. A., Adiwijaya, & Astuti, W. (2021). Analisis Sentimen Terhadap Ulasan Film Menggunakan *Word2Vec* dan SVM. *E-Proceeding of Engineering*, *8*(4), 4136–4144.

Kristiawan, K., Somali, D. D., Linggan jaya, T. A., & Widjaja, A. (2020). Deteksi Buah Menggunakan Supervised Learning dan Ekstraksi Fitur untuk Pemeriksa Harga. *Jurnal Teknik Informatika Dan Sistem Informasi*, *6*(3), 541–548. https://doi.org/10.28932/jutisi.v6i3.3029

Manggopa, R., Rantung, V. P., Kembuan, O., Informatika, T., Teknik, F., & Manado, U. N. (2024). *Aplikasi Analisis Sentimen Terhadap Kebijakan MBKM Menggunakan Algoritma Convolutional Neural Network (CNN) Berbasis Web*. 45–53.

Muhammad Afif Raihan, & Erwin Budi Setiawan. (2022). Aspect Based Sentimen Analysis with *FastText* Feature Expansion and Support Vector Machine Method on Twitter. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *6*(4), 591–598. https://doi.org/10.29207/resti.v6i4.4187

Mustasaruddin, M., Budianita, E., Fikry, M., & Yanto, F. (2023). Klasifikasi Sentimen Review Aplikasi MyPertamina Menggunakan *Word Embedding FastText* dan SVM (Support Vector Machine). *Jurnal Sistem Komputer Dan Informatika (JSON)*, *4*(3), 526. https://doi.org/10.30865/json.v4i3.5695

Nafisah Nurul Hakim. (2020). Implementasi Machine Learning pada Sistem Prediksi Kejadian dan Lokasi Patah Rel Kereta Api di Indonesia. *Jurnal Sistem Cerdas*, *3*(1), 25–35. https://doi.org/10.37396/jsc.v3i1.58

Nanda, S., Elvia, B., Fikry, M., & Pizaini. (2023). Analisis Sentimen Review Aplikasi MyPertamina Menggunakan *Word Embedding FastText* dan Algoritma K-Nearest Neighbor. *INFORMASI (Jurnal Informatika Dan Sistem Informasi)*, *15*(1), 91.

Nugroho, P. A., Fenriana, I., & Arijanto, R. (2020). Implementasi Deep Learning Menggunakan *Convolutional Neural Network* (*CNN*) Pada Ekspresi Manusia. *Algor*, *2*(1), 12–21.

Nurdewi, N. (2022). Implementasi Personal Branding Smart Asn Perwujudan Bangga Melayani Di Provinsi Maluku Utara. *SENTRI: Jurnal Riset Ilmiah*, *1*(2), 297–303. https://doi.org/10.55681/sentri.v1i2.235

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

Purnasiwi, R. G., Kusrini, & Hanafi, M. (2023). Analisis Sentimen Pada Review Produk Skincare Menggunakan *Word Embedding* dan Metode Long Short-Term Memory (LSTM). *Innovative: Journal Of Social Science Research*, *3*(2), 11433–11448.

Putu Sawitra Danda Prasetia, I., Adi Sastra Wijaya, K., Putu Dharmanu Yudartha, I., & Savitri, R. (2024). Analisis Persepsi Masyarakat Terhadap Aplikasi JAKI (Jakarta Kini) di Provinsi DKI Jakarta. *Jurnal Ilmu Sosial Dan Humaniora*, *7*(1), 1–12. https://jayapanguspress.penerbit.org/index.php/ganaya

Rahman, M. D., Djunaidy, A., & Mahananto, F. (2021). Penerapan Weighted *Word Embedding* pada Pengklasifikasian Teks Berbasis Recurrent Neural Network untuk Layanan Pengaduan Perusahaan Transportasi. *Jurnal Sains Dan Seni ITS*, *10*(1). https://doi.org/10.12962/j23373520.v10i1.56145

Riza, M. A., & Charibaldi, N. (2021). Emotion Detection in Twitter Social Media Using Long Short-Term Memory (LSTM) and Fast Text. *International Journal of Artificial Intelligence & Robotics (IJAIR)*, *3*(1), 15–26. https://doi.org/10.25139/ijair.v3i1.3827

Rodhi, M., Fadhlurrahman, R., & Aknuranda, I. (2022). Evaluasi Usability pada Aplikasi JAKI menggunakan Pengujian Usability. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *6*(8), 3857–3863. https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/11464

Sofiana, A. (2023). Analisis Implementasi Open Government Melalui Aplikasi Jakarta Kini (Jaki) Pada Fitur Jak-Respon Dalam Penyebarluasan Informasi

Publik. In *Journal of Social Contemplativa* (Vol. 1, Issue 1, pp. 45–53). https://doi.org/10.61183/jsc.v1i1.10

Tuasamu, Z., M. Lewaru, N. A. I., Idris, M. R., Syafaat, A. B. N., Faradilla, F., Fadlan, M., Nadiva, P., & Efendi, R. (2023). Analisis Sistem Informasi Akuntansi Siklus Pendapatan Menggunakan DFD Dan Flowchart Pada Bisnis Porobico. *Jurnal Bisnis Manajemen*, *1*(2), 495–510.

Vidya Chandradev, I Made Agus Dwi Suarjaya, & I Putu Agung Bayupati. (2023). Analisis Sentimen Review Hotel Menggunakan Metode Deep Learning BERT. *Jurnal Buana Informatika*, *14*(02), 107–116. https://doi.org/10.24002/jbi.v14i02.7244

Yulianeu, A., & Oktamala, R. (2022). Sistem Informasi Geografis Trayek Angkutan Umum Di Kota Tasikmalaya Berbasis Web. *JUTEKIN (Jurnal Teknik Informatika)*, *10*(2). https://doi.org/10.51530/jutekin.v10i2.669

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., Hindrayani, K. M., & Damaliana, A. T. (2024). Analisis Sentimen Terhadap Isu Feminisme Di Twitter Menggunakan Model *Convolutional Neural Network* (*CNN*). *Jurnal Lebesgue : Jurnal Ilmiah Pendidikan Matematika, Matematika Dan Statistika*, *5*(1), 477–491. https://doi.org/10.46306/lb.v5i1.585

LAMPIRAN

Lampiran 1. Source Code

```
import pandas as pd
import numpy as np
from tqdm import tqdm
from keras.ppreprocessing.text import Tokenizer
tqdm.pandas(desc="progress-bar")
from sklearn import utils
from sklearn.model_selection import train_test_split
from keras.ppreprocessing.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 random import shuffle # Import shuffle from random

# Load data from Excel file
df = pd.read_excel('/content/drive/Othercomputers/My
Laptop/Documents/SKRIPSI/Aryo - CNN -
FastText/DATAJAKI.xlsx',sheet_name="Sheet1")  # Replace
'path_to_your_excel_file.xlsx' with your actual file path
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()
df.shape

df.index = range(len(df))
total_words = df['ULASAN'].apply(lambda x: len(x.split('
'))).sum()
print("Total jumlah kata dalam semua ulasan:", total_words)

# 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])
print_message(12)

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 token
# Memisahkan data menjadi train dan test
train, test = train_test_split(df, test_size=0.1,
random_state=42)

# 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))

# 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

from gensim.models import FastText
from gensim.models.FastText import TaggedDocument

# Ubah ukuran vektor (vector_size) sesuai kebutuhan Anda
vector_size = 500

# Inisialisasi model FastText
FastText_model = FastText(vector_size=vector_size, window=8,
min_count=1, workers=1, sg=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()]

# Ubah train_tagged menjadi list kata-kata
train_corpus = [doc.words for doc in train_tagged]

# Bangun vocab dari train_corpus
FastText_model.build_vocab(train_corpus)

from sklearn.utils import shuffle
```

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

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

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

# Inisialisasi matriks embedding kosong
embedding_matrix =
np.zeros((len(FastText_model.wv.key_to_index),
FastText_model.vector_size))

# Mengisi matriks embedding dengan vektor-vektor kata dari
model FastText
for i, word in enumerate(FastText_model.wv.index_to_key):
    embedding_vector = FastText_model.wv.get_vector(word)
    if embedding_vector is not None:
        embedding_matrix[i] = embedding_vector

# Contoh penggunaan matriks embedding
print("Ukuran matriks embedding:", embedding_matrix.shape)
print("Contoh vektor untuk kata pertama:",
embedding_matrix[0])

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(0.25))

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

# Menambahkan lapisan Dense untuk output
model.add(Dense(3, 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)

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)

from keras.callbacks import EarlyStopping, ModelCheckpoint
# Membuat callback EarlyStopping dan ModelCheckpoint
model_checkpoint =
ModelCheckpoint('/content/drive/Othercomputers/My
Laptop/Documents/SKRIPSI/Aryo - CNN -
FastText/CNN_FastText.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)

# 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()

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)

print("Panjang Tes Ulasan:", len(test['ULASAN']))
```

```
print("Panjang X_test:", len(X_test))
print("Panjang Y_test:", len(Y_test))
print("Panjang true_labels:", len(true_labels))
print("Panjang predicted_labels:", len(predicted_labels))

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

# Melakukan prediksi dengan softmax (misalnya, jika
menggunakan TensorFlow/Keras)
predictions = model.predict(X_test)
predicted_labels = predictions.argmax(axis=1)  # Mengambil
kelas dengan probabilitas tertinggi sebagai prediksi

# Pastikan true_labels adalah dalam bentuk indeks kelas yang
sama dengan predicted_labels
true_labels = Y_test.argmax(axis=1)  # Jika Y_test adalah
dalam bentuk one-hot encoded, konversi ke indeks kelas

# 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))

# Ambil ulasan, label sebenarnya, dan prediksi label
# Pastikan panjang semua array sama
min_length = min(len(test['ULASAN']), len(true_labels),
len(predicted_labels))
test_results = pd.DataFrame({
    'Ulasan': test['ULASAN'].values[:min_length],

#Gunakan min_length untuk memastikan panjang yang sama
    'Label Sebenarnya': true_labels[:min_length],
    'Prediksi': predicted_labels[:min_length]
})
```

```
# Klasifikasi label 'Negatif', 'Positif', dan 'Netral'
berdasarkan nilai
def classify_label(label):
    if label == 0:
        return 'Negatif'
    elif label == 1:
       return 'Netral'
    else:
        return 'Positif'

# 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('/content/drive/Othercomputers/My
Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/HASIL
PREDIKSI.xlsx', index=False)

# Tampilkan hasil
print("\nHasil Prediksi:\n", test_results)
print("Data berhasil diekspor ke 'hasil_prediksi2.xlsx'.")
```

Lampiran 2. Dataset Ulasan Positif

	ULASAN	LABEL
4	1 kata, kerennnnn	Positif
11	Absensi mobile yang cepat.	Positif
12	Ada banyak info berita di Jaki	Positif
13	Ada jaki ,kita makin berkolaborasi	Positif
21	Aduan sangat cepat di respon 1x24 jam petugas langsung datang dan langsung ditindak lanjuti, terimakasih pemprov DKI Jakarta	Positif
31	Ajib	Positif
36	Aku suka	Positif
39	Akurasi dan penanganan sesuai dengan real-time based	Positif
40	Akurat terpecaya mantap Tq. Tetap jujur dan amanah.	Positif
41	Alhamdulillah bisa tau keadaan jakarta terkini	Positif
42	Alhamdulillah	Positif
43	Alhamdulillah apk ini membantu saya	Positif
44	Alhamdulillah aplikasinya juara nasional	Positif
45	Alhamdulillah bang anis gubernurnya nyaman warganya	Positif
46	Alhamdulillah berkat aplikasi ini saya bisa divaksin pertama dan next vaksin ke dua,persyaratan gk ribet dan simple,makasih	Positif
47	Alhamdulillah cepet bgt doong ditanganinnya, seandainya didalam ada aplikasi bintang lima keren kl nh atas pelayanan mantap gercep... Seandainya laporan anda lama terkirim mungkin ada k	Positif
1512	Kenyataan yg di alami teman dgn mengunakan aplikasi jaki menjadi lebih peduli dgn lingkungan utk menjadi dki lebih baik,BRAVO Jakarta ku	Positif
1519	Kereennnnn sangat bergunaaa	Positif
1520	Keren	Positif
1521	Keren aplikasi nya sangat membantu, saya vaksinasi ga sampe 1 jam, sudh beres..trimakasih ya..	Positif
1522	Keren aplikasinya	Positif
1523	keren aplikasinya, memudahkan warga DKI	Positif
1524	keren app nya, infonya up to date banget	Positif
1525	Keren banget dah diingatkan	Positif
1526	Keren banget nih aplikasi, banyak feature dalam satu aplikasi, membangun Jakarta dalam genggaman..	Positif
1527	Keren banget pemprov DKI	Positif
1528	Keren bgt memudahkan urusan daftar vaksin mudah dan praktis	Positif
1529	Keren bisa jaga rahasia	Positif
1530	Keren bisa mengetahui dan mengambil pembelajaran dari para warga Indonesia	Positif
1531	Keren dah ah	Positif
1532	Keren DKI	Positif
1533	Keren euy Pemda Jakarta	Positif
1534	Keren euy, memberikan informasi dan pelayanan menjadi lebih baik	Positif

3094	Udah oke keluhan selalu di tangani dengan baik. Semoga pandemik cepat berakhir	Positif
3133	Up to date dan praktis	Positif
3140	User friendly, easy to use... Keep on the good work...	Positif
3144	Uwow keren...makin yahud jaki aplikasi dengan berbagai informasi bagi warga jekartah yg akuh cintah...	Positif
3160	Verry good is the best	Positif
3163	very good	Positif
3164	Very good app and JakCLM is useful app for COVID19 self assessment	Positif
3165	Very great	Positif
3166	Very helpful	Positif
3167	very helpful application. easy to use and all services are here.	Positif
3171	Wah Aplikasinya membantu banget. Cuma Daftar Vaksin di Aplikasi Jaki tinggal Pilih tanggal dan lokasi. Sudah tersedia semua. Padahal Aku mah orang Bogor. Gak perlu repot2 ngatri di feskes2	Positif
3172	wah bagus sekali, wifi diaerah pejaten barat 2 rw 8, sinyalnya kenceng ngebantu sekali anak2 pada belajar online	Positif
3176	Wahh bagus sekalii. Aku terbantu skali. Makasih bnget.	Positif
3188	Wowww di aplikasi vaksin sinovac tp pas dateng Astrea, thanks alot yaw	Positif
3190	Wuihh aplikasi makin mantap aja nih Sangat di butuhkan Sekali, sebagai jembatan informasi Antara masyarakat dan Pemprov DKI	Positif
3191	ya alhamdulillah terimakasih	Positif
3200	Yg komentar jelek menurutku ngga smart pakai aplikasi, soal Pelayanan puskesmas yg krg baik tdk ada hubungannya sma aplikasi. Aplikasi dibuat untuk memudahkan database agr pelaksanaan	Positif

Lampiran 3. Dataset Ulasan Negatif

	ULASAN	LABEL
2	"Login error saat ini server sedang sibuk" selalu seperti itu saat login, terkadang log out dengan sendirinya.	Negatif
3	1 bintang karena aplikasi jaki kagak sinkron sama aplikasi peduli lindungi! Data vaksin beda tanggal, bikin ribet. Apps pedulilindungi cacad, ditambah harus sinkron sama aplikasi laen, jadi payah	Negatif
5	2 kali berturut-turut laporan yg saya buat, tutup oleh pihak JAKI sebelum problem selesai.. Kecewa banget... Tidak recommended	Negatif
6	2 kali install 2 kali jg ngeuninstall App buruk mau vaksin mesti ngprint sendiri kertas screeningnya, ga modal ya pemprovnya? Mending gw vaksin d sentra vaksin ui modal bawa diri dikasih hadiah	Negatif
10	Abis vaksin pertama di tanggal 30 Juli kok belum dapat sertifikat nya ya?... Aku cek nik di aplikasi jaki aku malah anda belum di vaksin aku daftar vaksin di aplikasi jaki	Negatif
14	Ada kendala saat pengisian CLM pada tahap identitas diri. Mengapa pada kolom yang dapat ter expand (dalam hal ini jenis kelamin, country, dll) namun di hp saya tidak bisa ? Sudah mencoba b	Negatif
15	ada pertanyaan tapi ga di respon	Negatif
22	Aduh... tujuan dounlod app buat daftar jakclm.. eh.. udh dounlod jakclm nya gak ada.. gimana ini. Bikin bingung	Negatif
23	Aduhhh susah banget buat daftat atau login .. buang' kouta kalo kaya giniii.. Padahal cuma mau ngunjungin taman aja login susah bangeettt- -	Negatif
24	After updating the Jaki application, it often has problems and cannot send reports, unlike the previous Jaki application which could send reports!!!	Negatif
25	Aga sulit mempergunakan jaki untuk partai banjir dan jaksurvai tidak mau valid	Negatif
26	Agak bingung utk jadwal vaksin dosis 2 krn di form Nov, tp di JAKI nya Okt yg benar yg mana donk? Knp kok China cepat sembuhnya?	Negatif
27	Agak lemot buka aplikasi nya	Negatif
28	Agak pusing makainya..	Negatif
29	agak sedikit susah	Negatif
30	Ah ga bener ni yg CLM masa iya gw dah di rumah ga kemana selama 4 bulan ini di bilang ga aman hasil test 38 sekian ga pernah ngumpul2 ketemu org yg pdp coba tetep tidak aman perbaiki ha	Negatif

1553	Keren, waktu mau daftar loading trs gk sudah2	Negatif
1554	Keren,tapi ngedownlodnya lama bgt	Negatif
1561	kesel, ngelag bangettt padahal butuh banget	Negatif
1562	Kesel, pdhl blm pernah pergi ke luar Jakarta mlh hasil tesnya rendah sih. Pdhl g pernah kontak sm ODP, PDP knp tetap sm sih. Padahal udh isi tes dengan jujur, bingung isi yg alamat lalnnya. Ini	Negatif
1563	Keseringan eror nya, gajelas	Negatif
1564	ketika mengisi CLM, hanya bisa sampai tanggal lahir, mulai jenis kelamin dan selanjutnya tidak bisa. padahal hp saya android 5.1, mohon penjelasan	Negatif
1570	Kirain satu aplikasi bisa mencakup semua hal.. Ternyata isi nya kaya playstore juga.. Banyak aplikasi yang perlu di install juga.. Sirukim Jaksiape Alpukat betawi Jakarta Jakone mobil MRT-J Tijek	Negatif
1571	Kirim kode verifikasi email gak masuk2 sampe saat ini gak ada email nya , aplikasi pemerintah kenapa selalu ampas !!! Yang kerja orang bawaan sih ,tolong di perbaiki lah !	Negatif
1573	Kita ngisis formulirnya gak sesuai hasil tes broo... Masa sya ngisi suhu 36, 1 hasil cek nya jadi 38 ..aneh	Negatif
1574	Klo blh cctv terminal pulogebang klo dimasukan juga di aplikasi jaki Jakarta kini	Negatif
1575	Klo blh di tambah kyk di masukan cctv	Negatif
1576	Kmarin isi data padahal suhu badan 36.5 „ngk pernah kontak langsung sama odp,apalagi pasien yg bener" Positif covid. Tp di situ tertulis tidak aman,,sebenernya batas suhu badan yang di perb	Negatif
1577	Knp aplikasi mrt,tje masih terpisah dg aplikasinya Lebih baik ini sudah included semuanya tanpa harus beda aplikasi,klo memang sudah smart	Negatif
1578	Knp aplikasi peduli lindungi sdh 2 minggu ini tdk bisa check in/out ??	Negatif
1579	Knp di aplikasi ini tidak busa buat daftar vaksin kedua !! Udah dicoba brulang kali tetap tidak bisa dgn tulisan kuota sudah penuh pdhal daftar nya buat besok	Negatif
1580	Knp di downloadnya gak bisa yaa??	Negatif
1581	Knp ga ada fitur jakclmnya setelah sy mengunduhnya..jadi bingung ini	Negatif

3094	Udah oke keluhan selalu di tangani dengan baik. Semoga pandemik cepat berakhir	Positif
3133	Up to date dan praktis	Positif
3140	User friendly, easy to use... Keep on the good work...	Positif
3144	Uwow keren...makin yahud jaki aplikasi dengan berbagai informasi bagi warga jekartah yg akuh cintah...	Positif
3160	Verry good is the best	Positif
3163	very good	Positif
3164	Very good app and JakCLM is useful app for COVID19 self assessment	Positif
3165	Very great	Positif
3166	Very helpful	Positif
3167	very helpful application. easy to use and all services are here.	Positif
3171	Wah Aplikasinya membantu banget. Cuma Daftar Vaksin di Aplikasi Jaki tinggal Pilih tanggal dan lokasi. Sudah tersedia semua. Padahal Aku mah orang Bogor. Gak perlu repot2 ngatri di feskes2	Positif
3172	wah bagus sekali, wifi diaerah pejaten barat 2 rw 8, sinyalnya kenceng ngebantu sekali anak2 pada belajar online	Positif
3176	Wahh bagus sekalii. Aku terbantu skali. Makasih bnget.	Positif
3188	Wowww di aplikasi vaksin sinovac tp pas dateng Astrea, thanks alot yaw	Positif
3190	Wuihh aplikasi makin mantap aja nih Sangat di butuhkan Sekali, sebagai jembatan informasi Antara masyarakat dan Pemprov DKI	Positif
3191	ya alhamdulillah terimakasih	Positif
3200	Yg komentar jelek menurutku ngga smart pakai aplikasi, soal Pelayanan puskesmas yg krg baik tdk ada hubungannya sma aplikasi. Aplikasi dibuat untuk memudahkan database agr pelaksanaan	Positif

Lampiran 4. Dataset Ulasan Netral

	ULASAN	LABEL
7	3 dulu nnti klo good tmbah lgi 1	Netral
8	3 dulu ya, terimakasih	Netral
9	3 nanti tambahan lagi	Netral
16	Adain aplikasi buat lowongan pekerjaan dong	Netral
17	Admin kenapa jakCLM di aplikasi saya ga ada ya padahal belum pernah saya gunakan.. mohon bantuannya	Netral
18	Admin saya perlu bantuan cara mengakses CLM di aplikasi ini. Saya bolak-balik kok nggak ketemu ya?	Netral
19	Admin tolong dong masa buat jenis kelamin nya ga bisa di klik jadinya ga bisa buat mengikuti tes, soalnya saya mau keluar kota	Netral
20	Admin,kenapa saya daftar Jawabannya Maaf Nama anda tidak sesuai dengan NIK yang terdaftar,padahal sy sudah cek update ke kemendagri Nama dan NIK saya sama.Jadi gimana solusinya? T	Netral
32	Aktivasi akun tidak berjalan lancar entah aplikasinya yang masih blm sempurna atau..... Jadi saya blm bisa memberi penilaian kinerja aparatur pemprov dki	Netral
33	aku blm tau bngt si cara pakai apss ini gmn.... soalnya aku disuh skola aku download apss ini..... ywd deh trpksa aku download walopun gtau cara pakai nya	Netral
34	Aku mau daftar vaksin tapi kuota sudah penuh semua, bagaimana ya?	Netral
35	Aku pake ini ke tebet eco park lancar cuman gitu kadang nge lag	Netral
37	Aku sukaaaa apk ini tapi pas aku buka JAKI dan aku tekan Lain nya trus aku pncet JAKCO nyako ga ada JAKCLM nya ya??	Netral
55	Alhamdulillah sebentar lagi anis selesai menjabat!!! #hiphiphore nanti id sdh selesai jabatannya si anis saya akan edit dan saya kasih 5*	Netral
68	Anak sekolah di jkt disuruh ngisi rafidtest online di apk ini. Saya bingung!	Netral
81	Apa benar ditindak sih kalo lapor kesini	Netral

1144	Harus oke setiap saat .. Trimakasih ..	Netral
1168	Hmm ya bolehlah aplikasi ini, banyak info. Andai sekalian berafiliasi sama aplikasi Trafi pasti keren	Netral
1172	I want the Jakarta to be the best city	Netral
1178	Info mudik gratis dari Pemprov Jakarta yang valid? Apa bisa dapatkan disini..?	Netral
1206	ini aplikasi khusus warga Jakarta doang	Netral
1267	izin bertanya sudah daftar vaksin online melalui jaki tetapi ke hapus data nya,apakah masih bisa mengikuti vaksin terima kasih	Netral
1270	Jadi gini gaes perihal daftar Vaksinasi yak.JAKI hanya buat database + cetak dokumen saja.Perihal antrian Penuh/gak dapat nomor antrian di TKP,ya karena petugas di lapangan patokannya non	Netral
1318	Jaki kalo lapor biasanya ditangani berapa lame prosesnya? Semoga jaki bisa bertindak cepat tanggap ya..	Netral
1371	Jika kita pengguna lupa mendownload hasil tes clm itu harus bagaimana	Netral
1383	Kalau ada yang eror/masalahny saya akan laporkan	Netral
1384	Kalau aplikasinya berguna dan bagus, Diem gak mau komentar. Kalau ada bug sedikit langsung bintang 1 dan komentar jelek². Semoga programermya bersabar menghadapi warga 62.	Netral
1387	Kalau kalian kesulitan daftar di aplikasi, langsung aja buka corona.jakarta.go.id lebih cepat	Netral
1388	Kalau lapor kerumunan bisa lewat jaki?	Netral
1401	Karena di suruh ngisi clm jadi download abis itu ngisi clm uninstall lagi	Netral
1404	Kasih 3 bintang dulu krn baru masuk ke ke akun Jaki	Netral
1405	Kasih 4 bintang dulu ya soalnya masih percobaan belom liat semuanya	Netral
1406	Kasih bintang 3 dl krna saya baru instal, dilihat dl fungsinya.. 🙄	Netral

2557	Saya mau vaksin ke 2,dan udah dapat no tiket,tpi knapa nggak bisa masuk pdhal mau vaksin buat tgl 27 ini	Netral
2561	Saya melakukan Vaksinasi Tahap 1 Tgl 11 Juli 2021 di GBK, Mendaftar melalui Jaki ini. Ada bukti foto dan selembarannya. Namun di cek di peduli lindungi, Data saya blm di vaksinasi sampai saat	Netral
2589	Saya sudah divaksin 1, terus vaksin kedua sudah terjadwal tapi lokasi faskesnya gak ada cuma kecamatan doang, apakah vaksin 2 harus daftar ulang lagi? Terima kasih	Netral
2591	Saya sudah install aplikasinya, tapi menu JakCLM nya tidak ada	Netral
2592	Saya sudah melakukan vaksin tgl 15/7/2021 hasil sertifikasi vaksin saya kenapa belum terbit ya	Netral
2595	Saya sudah mengisi data, tetapi untuk pengisian jenis kelamin mengalami kendala, tidak bisa mengisi atau mengetik sampai berulang-ulang .bahkan saya mendaftar ke akun jaki....mohon bantu	Netral
2643	Sedang mencoba..... Mudah2an bisa bermanfaat	Netral
2647	Segalanya akan lebih cepat dan adanya komunikasi antara bawah dan atas	Netral
2738	Setelah di update, tidak dapat cek pajak kendaraan motor roda dua. Catatan : Sudah fix, permasalahan ada pada jaringan internet Handphone. Terimakasih, Tim Developer sudah membuatkan	Netral
2791	Sudah lancar di pertahankan.. Tp sayang cek bansos tdk terdaftar😭😭😭😭	Netral
2792	Sudah lumayan baik tanggapan atas pelaporam yg valid Semoga makin cepat tanggap. Insentif lebih di perlukan untuk pegawi yg benar benar bekerja.	Netral
2842	Sy beri 4 bintang biar makin semangat IT JAKI. Dha mau gunakan JAKI daftar vaksinasi booster ke-2 muncul jawaban "Maaf, saat ini fitur tidak bisa digunakan". Terus bgmi cara sy daftar vaksinas	Netral
3108	Untuk cek diri dari covid buka JakCLM" Mana di jaki gua ga ada JakCLM nya ini buat urusan sekolah lho... Pls lah jaki.. Gua kasih bintang satu aja ya...	Netral
3131	Untuk TL (selesai) yg mendapat bintang 1 atau 2 harusnya kembali ke laporan/proses kembali, sehingga bisa di TL sampai akhirnya dapat minimal bintang 4. Ini pernah di lakukan aplikasi qlue	Netral
3143	Utk menu clm tolong di maximum kan, setiap ini data tiba tiba langsung kembali ke menu awal	Netral
3168	Very useful and friendly apps	Netral
3187	Woke ok lah	Netral

Lampiran 5. Proses Epoch

1. Model Word Embedding FastText – CNN

a. Pembagian Data 90:10

```
Epoch 1: val_acc improved from -inf to 0.71250, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as an HDF5 file via `model.save()`. This file form
  saving_api.save_model(
90/90 - 2s - loss: 0.8866 - acc: 0.6336 - val_loss: 0.6636 - val_acc: 0.7125 - 2s/epoch - 24ms/step
Epoch 2/50

Epoch 2: val_acc improved from 0.71250 to 0.79375, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
90/90 - 0s - loss: 0.6747 - acc: 0.7263 - val_loss: 0.6414 - val_acc: 0.7937 - 442ms/epoch - 5ms/step
Epoch 3/50

Epoch 3: val_acc did not improve from 0.79375
90/90 - 0s - loss: 0.5965 - acc: 0.7794 - val_loss: 0.5548 - val_acc: 0.7937 - 404ms/epoch - 4ms/step
Epoch 4/50

Epoch 4: val_acc improved from 0.79375 to 0.85625, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
90/90 - 0s - loss: 0.5091 - acc: 0.8204 - val_loss: 0.4714 - val_acc: 0.8562 - 438ms/epoch - 5ms/step
Epoch 5/50

Epoch 5: val_acc improved from 0.85625 to 0.87187, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
90/90 - 0s - loss: 0.4262 - acc: 0.8618 - val_loss: 0.3994 - val_acc: 0.8719 - 440ms/epoch - 5ms/step
Epoch 6/50

Epoch 6: val_acc improved from 0.87187 to 0.89688, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
90/90 - 0s - loss: 0.3711 - acc: 0.8795 - val_loss: 0.3826 - val_acc: 0.8969 - 442ms/epoch - 5ms/step
Epoch 7/50

Epoch 7: val_acc improved from 0.89688 to 0.90625, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
90/90 - 0s - loss: 0.3201 - acc: 0.8934 - val_loss: 0.3422 - val_acc: 0.9062 - 474ms/epoch - 5ms/step
Epoch 8/50

Epoch 8: val_acc improved from 0.90625 to 0.91250, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
90/90 - 0s - loss: 0.2880 - acc: 0.9055 - val_loss: 0.3478 - val_acc: 0.9125 - 494ms/epoch - 5ms/step
Epoch 9/50

Epoch 9: val_acc improved from 0.91250 to 0.91875, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
90/90 - 0s - loss: 0.2580 - acc: 0.9125 - val_loss: 0.3181 - val_acc: 0.9187 - 448ms/epoch - 5ms/step
Epoch 10/50

Epoch 10: val_acc improved from 0.91875 to 0.92500, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
90/90 - 0s - loss: 0.2283 - acc: 0.9198 - val_loss: 0.3003 - val_acc: 0.9250 - 448ms/epoch - 5ms/step
```

```
Epoch 11: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.2052 - acc: 0.9278 - val_loss: 0.2945 - val_acc: 0.9125 - 417ms/epoch - 5ms/step
Epoch 12/50

Epoch 12: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.1782 - acc: 0.9410 - val_loss: 0.3064 - val_acc: 0.9031 - 413ms/epoch - 5ms/step
Epoch 13/50

Epoch 13: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.1618 - acc: 0.9434 - val_loss: 0.2896 - val_acc: 0.9187 - 417ms/epoch - 5ms/step
Epoch 14/50

Epoch 14: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.1430 - acc: 0.9528 - val_loss: 0.2946 - val_acc: 0.9094 - 413ms/epoch - 5ms/step
Epoch 15/50

Epoch 15: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.1244 - acc: 0.9604 - val_loss: 0.3092 - val_acc: 0.9031 - 435ms/epoch - 5ms/step
Epoch 16/50

Epoch 16: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.1036 - acc: 0.9733 - val_loss: 0.2744 - val_acc: 0.9156 - 409ms/epoch - 5ms/step
Epoch 17/50

Epoch 17: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0913 - acc: 0.9750 - val_loss: 0.3058 - val_acc: 0.9062 - 413ms/epoch - 5ms/step
Epoch 18/50

Epoch 18: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0816 - acc: 0.9799 - val_loss: 0.2819 - val_acc: 0.8938 - 419ms/epoch - 5ms/step
Epoch 19/50

Epoch 19: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0692 - acc: 0.9823 - val_loss: 0.2897 - val_acc: 0.9000 - 431ms/epoch - 5ms/step
Epoch 20/50

Epoch 20: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0554 - acc: 0.9875 - val_loss: 0.2848 - val_acc: 0.9094 - 411ms/epoch - 5ms/step
Epoch 21/50

Epoch 21: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0531 - acc: 0.9889 - val_loss: 0.2989 - val_acc: 0.8938 - 429ms/epoch - 5ms/step
Epoch 22/50

Epoch 22: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0454 - acc: 0.9899 - val_loss: 0.2946 - val_acc: 0.9000 - 426ms/epoch - 5ms/step
Epoch 23/50

Epoch 23: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0395 - acc: 0.9920 - val_loss: 0.3110 - val_acc: 0.8906 - 423ms/epoch - 5ms/step
Epoch 24/50

Epoch 24: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0331 - acc: 0.9948 - val_loss: 0.3401 - val_acc: 0.9031 - 418ms/epoch - 5ms/step
Epoch 25/50

Epoch 25: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0301 - acc: 0.9958 - val_loss: 0.3092 - val_acc: 0.9000 - 421ms/epoch - 5ms/step
Epoch 26/50

Epoch 26: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0297 - acc: 0.9941 - val_loss: 0.3196 - val_acc: 0.8875 - 420ms/epoch - 5ms/step
Epoch 27/50

Epoch 27: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0257 - acc: 0.9958 - val_loss: 0.3243 - val_acc: 0.8938 - 424ms/epoch - 5ms/step
Epoch 28/50

Epoch 28: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0208 - acc: 0.9972 - val_loss: 0.3331 - val_acc: 0.8938 - 422ms/epoch - 5ms/step
Epoch 29/50

Epoch 29: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0178 - acc: 0.9976 - val_loss: 0.3248 - val_acc: 0.9000 - 427ms/epoch - 5ms/step
Epoch 30/50

Epoch 30: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0161 - acc: 0.9976 - val_loss: 0.3324 - val_acc: 0.8875 - 426ms/epoch - 5ms/step
Epoch 31/50

Epoch 31: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0158 - acc: 0.9976 - val_loss: 0.3359 - val_acc: 0.8938 - 422ms/epoch - 5ms/step
Epoch 32/50

Epoch 32: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0137 - acc: 0.9983 - val_loss: 0.3490 - val_acc: 0.8938 - 434ms/epoch - 5ms/step
Epoch 33/50

Epoch 33: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0125 - acc: 0.9972 - val_loss: 0.3499 - val_acc: 0.8969 - 425ms/epoch - 5ms/step
Epoch 34/50

Epoch 34: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0101 - acc: 0.9993 - val_loss: 0.3625 - val_acc: 0.8906 - 426ms/epoch - 5ms/step
Epoch 35/50

Epoch 35: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0105 - acc: 0.9986 - val_loss: 0.3539 - val_acc: 0.8969 - 431ms/epoch - 5ms/step
Epoch 36/50

Epoch 36: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0087 - acc: 0.9997 - val_loss: 0.3652 - val_acc: 0.9000 - 437ms/epoch - 5ms/step
Epoch 37/50

Epoch 37: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0073 - acc: 0.9993 - val_loss: 0.3571 - val_acc: 0.8969 - 434ms/epoch - 5ms/step
Epoch 38/50

Epoch 38: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0062 - acc: 0.9993 - val_loss: 0.3659 - val_acc: 0.8938 - 436ms/epoch - 5ms/step
Epoch 39/50

Epoch 39: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0064 - acc: 0.9997 - val_loss: 0.3889 - val_acc: 0.8844 - 434ms/epoch - 5ms/step
Epoch 40/50

Epoch 40: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0061 - acc: 1.0000 - val_loss: 0.3772 - val_acc: 0.9000 - 425ms/epoch - 5ms/step
Epoch 41/50
```

```
Epoch 41: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0057 - acc: 0.9997 - val_loss: 0.3823 - val_acc: 0.9031 - 430ms/epoch - 5ms/step
Epoch 42/50

Epoch 42: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0055 - acc: 0.9993 - val_loss: 0.3844 - val_acc: 0.8875 - 429ms/epoch - 5ms/step
Epoch 43/50

Epoch 43: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0047 - acc: 0.9997 - val_loss: 0.3840 - val_acc: 0.8969 - 434ms/epoch - 5ms/step
Epoch 44/50

Epoch 44: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0044 - acc: 1.0000 - val_loss: 0.3959 - val_acc: 0.8969 - 414ms/epoch - 5ms/step
Epoch 45/50

Epoch 45: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0042 - acc: 0.9997 - val_loss: 0.4072 - val_acc: 0.8906 - 424ms/epoch - 5ms/step
Epoch 46/50

Epoch 46: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0039 - acc: 0.9997 - val_loss: 0.4105 - val_acc: 0.8938 - 418ms/epoch - 5ms/step
Epoch 47/50

Epoch 47: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0038 - acc: 0.9997 - val_loss: 0.4085 - val_acc: 0.8938 - 422ms/epoch - 5ms/step
Epoch 48/50

Epoch 48: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0035 - acc: 0.9997 - val_loss: 0.4060 - val_acc: 0.9000 - 424ms/epoch - 5ms/step
Epoch 49/50

Epoch 49: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0033 - acc: 0.9997 - val_loss: 0.4277 - val_acc: 0.8969 - 417ms/epoch - 5ms/step
Epoch 50/50

Epoch 50: val_acc did not improve from 0.92500
90/90 - 0s - loss: 0.0028 - acc: 1.0000 - val_loss: 0.4253 - val_acc: 0.8969 - 427ms/epoch - 5ms/step
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
Validation Loss: [0.6635971069335938, 0.6413944959640503, 0.5547658205032349, 0.4714061617851257, 0.3993641734123230, 0.3825910985469818, 0.3422056436538696, 0.34
```

b. Pembagian Data 80:20

```
Epoch 1: val_acc improved from -inf to 0.67969, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as an HDF5 file via `model.save()`. This file form
  saving_api.save_model(
80/80 - 2s - loss: 0.8007 - acc: 0.6495 - val_loss: 0.7176 - val_acc: 0.6797 - 2s/epoch - 21ms/step
Epoch 2/50

Epoch 2: val_acc improved from 0.67969 to 0.73281, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
80/80 - 0s - loss: 0.6843 - acc: 0.7272 - val_loss: 0.6821 - val_acc: 0.7328 - 401ms/epoch - 5ms/step
Epoch 3/50

Epoch 3: val_acc improved from 0.73281 to 0.79062, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
80/80 - 0s - loss: 0.5996 - acc: 0.7714 - val_loss: 0.6028 - val_acc: 0.7906 - 409ms/epoch - 5ms/step
Epoch 4/50

Epoch 4: val_acc improved from 0.79062 to 0.81563, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
80/80 - 0s - loss: 0.5087 - acc: 0.8238 - val_loss: 0.5253 - val_acc: 0.8156 - 412ms/epoch - 5ms/step
Epoch 5/50

Epoch 5: val_acc improved from 0.81563 to 0.83594, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
80/80 - 0s - loss: 0.4358 - acc: 0.8570 - val_loss: 0.4730 - val_acc: 0.8359 - 409ms/epoch - 5ms/step
Epoch 6/50

Epoch 6: val_acc improved from 0.83594 to 0.86406, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
80/80 - 0s - loss: 0.3755 - acc: 0.8753 - val_loss: 0.4229 - val_acc: 0.8641 - 442ms/epoch - 6ms/step
Epoch 7/50

Epoch 7: val_acc improved from 0.86406 to 0.87500, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
80/80 - 0s - loss: 0.3299 - acc: 0.8894 - val_loss: 0.4243 - val_acc: 0.8750 - 403ms/epoch - 5ms/step
Epoch 8/50

Epoch 8: val_acc improved from 0.87500 to 0.88281, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
80/80 - 0s - loss: 0.2864 - acc: 0.9047 - val_loss: 0.3612 - val_acc: 0.8828 - 427ms/epoch - 5ms/step
Epoch 9/50

Epoch 9: val_acc improved from 0.88281 to 0.88437, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
80/80 - 0s - loss: 0.2566 - acc: 0.9117 - val_loss: 0.3669 - val_acc: 0.8844 - 410ms/epoch - 5ms/step
Epoch 10/50

Epoch 10: val_acc improved from 0.88437 to 0.88906, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
80/80 - 0s - loss: 0.2332 - acc: 0.9191 - val_loss: 0.3805 - val_acc: 0.8891 - 416ms/epoch - 5ms/step

Epoch 11: val_acc improved from 0.88906 to 0.89531, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
80/80 - 0s - loss: 0.2025 - acc: 0.9281 - val_loss: 0.3354 - val_acc: 0.8953 - 409ms/epoch - 5ms/step
Epoch 12/50

Epoch 12: val_acc improved from 0.89531 to 0.90000, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
80/80 - 0s - loss: 0.1782 - acc: 0.9445 - val_loss: 0.3318 - val_acc: 0.9000 - 410ms/epoch - 5ms/step
Epoch 13/50

Epoch 13: val_acc did not improve from 0.90000
80/80 - 0s - loss: 0.1544 - acc: 0.9519 - val_loss: 0.3227 - val_acc: 0.8969 - 364ms/epoch - 5ms/step
Epoch 14/50

Epoch 14: val_acc did not improve from 0.90000
80/80 - 0s - loss: 0.1437 - acc: 0.9570 - val_loss: 0.3221 - val_acc: 0.8984 - 372ms/epoch - 5ms/step
Epoch 15/50

Epoch 15: val_acc did not improve from 0.90000
80/80 - 0s - loss: 0.1200 - acc: 0.9648 - val_loss: 0.3105 - val_acc: 0.8984 - 380ms/epoch - 5ms/step
Epoch 16/50

Epoch 16: val_acc did not improve from 0.90000
80/80 - 0s - loss: 0.1093 - acc: 0.9699 - val_loss: 0.2990 - val_acc: 0.8984 - 380ms/epoch - 5ms/step
Epoch 17/50

Epoch 17: val_acc did not improve from 0.90000
80/80 - 0s - loss: 0.0961 - acc: 0.9730 - val_loss: 0.3173 - val_acc: 0.9000 - 376ms/epoch - 5ms/step
Epoch 18/50

Epoch 18: val_acc did not improve from 0.90000
80/80 - 0s - loss: 0.0819 - acc: 0.9793 - val_loss: 0.3139 - val_acc: 0.8969 - 379ms/epoch - 5ms/step
Epoch 19/50

Epoch 19: val_acc improved from 0.90000 to 0.90156, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
80/80 - 0s - loss: 0.0682 - acc: 0.9848 - val_loss: 0.3203 - val_acc: 0.9016 - 405ms/epoch - 5ms/step
Epoch 20/50

Epoch 20: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0620 - acc: 0.9844 - val_loss: 0.3091 - val_acc: 0.8969 - 378ms/epoch - 5ms/step
```

```
Epoch 21/50

Epoch 21: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0545 - acc: 0.9867 - val_loss: 0.3061 - val_acc: 0.9000 - 369ms/epoch - 5ms/step
Epoch 22/50

Epoch 22: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0477 - acc: 0.9894 - val_loss: 0.3346 - val_acc: 0.8906 - 378ms/epoch - 5ms/step
Epoch 23/50

Epoch 23: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0422 - acc: 0.9910 - val_loss: 0.3304 - val_acc: 0.8922 - 376ms/epoch - 5ms/step
Epoch 24/50

Epoch 24: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0362 - acc: 0.9926 - val_loss: 0.3333 - val_acc: 0.8906 - 381ms/epoch - 5ms/step
Epoch 25/50

Epoch 25: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0297 - acc: 0.9949 - val_loss: 0.3252 - val_acc: 0.8891 - 381ms/epoch - 5ms/step
Epoch 26/50

Epoch 26: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0275 - acc: 0.9961 - val_loss: 0.3294 - val_acc: 0.8906 - 383ms/epoch - 5ms/step
Epoch 27/50

Epoch 27: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0245 - acc: 0.9973 - val_loss: 0.3432 - val_acc: 0.8906 - 382ms/epoch - 5ms/step
Epoch 28/50

Epoch 28: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0226 - acc: 0.9969 - val_loss: 0.3350 - val_acc: 0.8906 - 375ms/epoch - 5ms/step
Epoch 29/50

Epoch 29: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0186 - acc: 0.9977 - val_loss: 0.3495 - val_acc: 0.8906 - 381ms/epoch - 5ms/step
Epoch 30/50

Epoch 30: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0169 - acc: 0.9977 - val_loss: 0.3426 - val_acc: 0.8906 - 383ms/epoch - 5ms/step
Epoch 31/50

Epoch 31: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0133 - acc: 0.9996 - val_loss: 0.3462 - val_acc: 0.8906 - 380ms/epoch - 5ms/step
Epoch 32/50

Epoch 32: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0122 - acc: 0.9996 - val_loss: 0.3607 - val_acc: 0.8906 - 372ms/epoch - 5ms/step
Epoch 33/50

Epoch 33: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0134 - acc: 0.9980 - val_loss: 0.3591 - val_acc: 0.8906 - 371ms/epoch - 5ms/step
Epoch 34/50

Epoch 34: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0108 - acc: 0.9992 - val_loss: 0.3722 - val_acc: 0.8875 - 391ms/epoch - 5ms/step
Epoch 35/50

Epoch 35: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0096 - acc: 0.9996 - val_loss: 0.3758 - val_acc: 0.8891 - 387ms/epoch - 5ms/step
Epoch 36/50

Epoch 36: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0106 - acc: 0.9988 - val_loss: 0.3672 - val_acc: 0.8844 - 386ms/epoch - 5ms/step
Epoch 37/50

Epoch 37: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0085 - acc: 0.9996 - val_loss: 0.3824 - val_acc: 0.8859 - 378ms/epoch - 5ms/step
Epoch 38/50

Epoch 38: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0066 - acc: 1.0000 - val_loss: 0.3693 - val_acc: 0.8828 - 377ms/epoch - 5ms/step
Epoch 39/50

Epoch 39: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0065 - acc: 0.9996 - val_loss: 0.3791 - val_acc: 0.8875 - 388ms/epoch - 5ms/step
Epoch 40/50

Epoch 40: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0066 - acc: 0.9996 - val_loss: 0.3832 - val_acc: 0.8844 - 385ms/epoch - 5ms/step
Epoch 41/50

Epoch 41: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0058 - acc: 1.0000 - val_loss: 0.3906 - val_acc: 0.8859 - 375ms/epoch - 5ms/step
Epoch 42/50

Epoch 42: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0051 - acc: 1.0000 - val_loss: 0.3918 - val_acc: 0.8844 - 362ms/epoch - 5ms/step
Epoch 43/50

Epoch 43: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0053 - acc: 1.0000 - val_loss: 0.3940 - val_acc: 0.8875 - 357ms/epoch - 4ms/step
Epoch 44/50

Epoch 44: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0044 - acc: 1.0000 - val_loss: 0.3900 - val_acc: 0.8859 - 369ms/epoch - 5ms/step
Epoch 45/50

Epoch 45: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0042 - acc: 0.9996 - val_loss: 0.3934 - val_acc: 0.8828 - 372ms/epoch - 5ms/step
Epoch 46/50

Epoch 46: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0037 - acc: 0.9996 - val_loss: 0.4134 - val_acc: 0.8875 - 370ms/epoch - 5ms/step
Epoch 47/50

Epoch 47: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0034 - acc: 0.9996 - val_loss: 0.4078 - val_acc: 0.8875 - 374ms/epoch - 5ms/step
Epoch 48/50

Epoch 48: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0039 - acc: 0.9992 - val_loss: 0.4218 - val_acc: 0.8891 - 369ms/epoch - 5ms/step
Epoch 49/50

Epoch 49: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0027 - acc: 1.0000 - val_loss: 0.4153 - val_acc: 0.8859 - 372ms/epoch - 5ms/step
Epoch 50/50

Epoch 50: val_acc did not improve from 0.90156
80/80 - 0s - loss: 0.0031 - acc: 0.9996 - val_loss: 0.4492 - val_acc: 0.8844 - 367ms/epoch - 5ms/step
```

c. Pembagian Data 70:30

```
Epoch 1: val_acc improved from -inf to 0.67917, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as an HDF5 file via `model.save()`. This file for
  saving_api.save_model(
70/70 - 1s - loss: 0.7946 - acc: 0.6440 - val_loss: 0.7389 - val_acc: 0.6792 - 1s/epoch - 21ms/step
Epoch 2/50

Epoch 2: val_acc improved from 0.67917 to 0.72500, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.6980 - acc: 0.7119 - val_loss: 0.6725 - val_acc: 0.7250 - 408ms/epoch - 6ms/step
Epoch 3/50

Epoch 3: val_acc improved from 0.72500 to 0.76354, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.6302 - acc: 0.7552 - val_loss: 0.6308 - val_acc: 0.7635 - 385ms/epoch - 6ms/step
Epoch 4/50

Epoch 4: val_acc improved from 0.76354 to 0.80417, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.5605 - acc: 0.7861 - val_loss: 0.5737 - val_acc: 0.8042 - 383ms/epoch - 5ms/step
Epoch 5/50

Epoch 5: val_acc improved from 0.80417 to 0.82187, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.4809 - acc: 0.8267 - val_loss: 0.5108 - val_acc: 0.8219 - 390ms/epoch - 6ms/step
Epoch 6/50

Epoch 6: val_acc improved from 0.82187 to 0.84375, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.4193 - acc: 0.8566 - val_loss: 0.4779 - val_acc: 0.8438 - 385ms/epoch - 5ms/step
Epoch 7/50

Epoch 7: val_acc improved from 0.84375 to 0.84792, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.3540 - acc: 0.8723 - val_loss: 0.4432 - val_acc: 0.8479 - 391ms/epoch - 6ms/step
Epoch 8/50

Epoch 8: val_acc improved from 0.84792 to 0.87708, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.3165 - acc: 0.8852 - val_loss: 0.4026 - val_acc: 0.8771 - 382ms/epoch - 5ms/step
Epoch 9/50

Epoch 9: val_acc did not improve from 0.87708
70/70 - 0s - loss: 0.2722 - acc: 0.9031 - val_loss: 0.3973 - val_acc: 0.8750 - 360ms/epoch - 5ms/step
Epoch 10/50

Epoch 10: val_acc improved from 0.87708 to 0.87917, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.2368 - acc: 0.9196 - val_loss: 0.3598 - val_acc: 0.8792 - 384ms/epoch - 5ms/step
Epoch 11/50

Epoch 11: val_acc improved from 0.87917 to 0.88646, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.2140 - acc: 0.9254 - val_loss: 0.3682 - val_acc: 0.8865 - 389ms/epoch - 6ms/step
Epoch 12/50

Epoch 12: val_acc did not improve from 0.88646
70/70 - 0s - loss: 0.1846 - acc: 0.9375 - val_loss: 0.3456 - val_acc: 0.8865 - 357ms/epoch - 5ms/step
Epoch 13/50

Epoch 13: val_acc improved from 0.88646 to 0.89375, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.1660 - acc: 0.9469 - val_loss: 0.3427 - val_acc: 0.8938 - 380ms/epoch - 5ms/step
Epoch 14/50

Epoch 14: val_acc did not improve from 0.89375
70/70 - 0s - loss: 0.1479 - acc: 0.9540 - val_loss: 0.3394 - val_acc: 0.8927 - 350ms/epoch - 5ms/step
Epoch 15/50

Epoch 15: val_acc did not improve from 0.89375
70/70 - 0s - loss: 0.1283 - acc: 0.9652 - val_loss: 0.3505 - val_acc: 0.8865 - 369ms/epoch - 5ms/step
Epoch 16/50

Epoch 16: val_acc did not improve from 0.89375
70/70 - 0s - loss: 0.1094 - acc: 0.9741 - val_loss: 0.3323 - val_acc: 0.8917 - 357ms/epoch - 5ms/step
Epoch 17/50

Epoch 17: val_acc improved from 0.89375 to 0.89688, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.0923 - acc: 0.9799 - val_loss: 0.3206 - val_acc: 0.8969 - 394ms/epoch - 6ms/step
Epoch 18/50

Epoch 18: val_acc did not improve from 0.89688
70/70 - 0s - loss: 0.0859 - acc: 0.9795 - val_loss: 0.3295 - val_acc: 0.8948 - 358ms/epoch - 5ms/step
Epoch 19/50

Epoch 19: val_acc improved from 0.89688 to 0.89792, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.0769 - acc: 0.9848 - val_loss: 0.3334 - val_acc: 0.8979 - 388ms/epoch - 6ms/step
Epoch 20/50

Epoch 20: val_acc did not improve from 0.89792
70/70 - 0s - loss: 0.0643 - acc: 0.9879 - val_loss: 0.3278 - val_acc: 0.8896 - 371ms/epoch - 5ms/step
Epoch 21/50

Epoch 21: val_acc did not improve from 0.89792
70/70 - 0s - loss: 0.0558 - acc: 0.9870 - val_loss: 0.3134 - val_acc: 0.8969 - 362ms/epoch - 5ms/step
Epoch 22/50

Epoch 22: val_acc did not improve from 0.89792
70/70 - 0s - loss: 0.0509 - acc: 0.9906 - val_loss: 0.3216 - val_acc: 0.8958 - 366ms/epoch - 5ms/step
Epoch 23/50

Epoch 23: val_acc improved from 0.89792 to 0.90104, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.0424 - acc: 0.9937 - val_loss: 0.3320 - val_acc: 0.9010 - 397ms/epoch - 6ms/step
Epoch 24/50

Epoch 24: val_acc improved from 0.90104 to 0.90208, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.0385 - acc: 0.9937 - val_loss: 0.3375 - val_acc: 0.9021 - 417ms/epoch - 6ms/step
Epoch 25/50

Epoch 25: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0337 - acc: 0.9937 - val_loss: 0.3349 - val_acc: 0.8948 - 360ms/epoch - 5ms/step
Epoch 26/50

Epoch 26: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0302 - acc: 0.9951 - val_loss: 0.3386 - val_acc: 0.8885 - 364ms/epoch - 5ms/step
Epoch 27/50

Epoch 27: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0280 - acc: 0.9946 - val_loss: 0.3512 - val_acc: 0.8875 - 371ms/epoch - 5ms/step
Epoch 28/50

Epoch 28: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0231 - acc: 0.9964 - val_loss: 0.3278 - val_acc: 0.9021 - 367ms/epoch - 5ms/step
Epoch 29/50

Epoch 29: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0217 - acc: 0.9969 - val_loss: 0.3415 - val_acc: 0.8979 - 366ms/epoch - 5ms/step
Epoch 30/50

Epoch 30: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0183 - acc: 0.9973 - val_loss: 0.3379 - val_acc: 0.8969 - 362ms/epoch - 5ms/step
Epoch 31/50
```

```
Epoch 31: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0168 - acc: 0.9978 - val_loss: 0.3406 - val_acc: 0.8990 - 360ms/epoch - 5ms/step
Epoch 32/50

Epoch 32: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0153 - acc: 0.9991 - val_loss: 0.3350 - val_acc: 0.9010 - 354ms/epoch - 5ms/step
Epoch 33/50

Epoch 33: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0130 - acc: 0.9982 - val_loss: 0.3383 - val_acc: 0.8990 - 354ms/epoch - 5ms/step
Epoch 34/50

Epoch 34: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0117 - acc: 0.9991 - val_loss: 0.3459 - val_acc: 0.9010 - 361ms/epoch - 5ms/step
Epoch 35/50

Epoch 35: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0102 - acc: 0.9996 - val_loss: 0.3404 - val_acc: 0.8885 - 358ms/epoch - 5ms/step
Epoch 36/50

Epoch 36: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0124 - acc: 0.9982 - val_loss: 0.3644 - val_acc: 0.8938 - 361ms/epoch - 5ms/step
Epoch 37/50

Epoch 37: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0089 - acc: 0.9996 - val_loss: 0.3539 - val_acc: 0.8979 - 353ms/epoch - 5ms/step
Epoch 38/50

Epoch 38: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0078 - acc: 0.9996 - val_loss: 0.3613 - val_acc: 0.8969 - 356ms/epoch - 5ms/step
Epoch 39/50

Epoch 39: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0071 - acc: 0.9996 - val_loss: 0.3559 - val_acc: 0.9000 - 351ms/epoch - 5ms/step
Epoch 40/50

Epoch 40: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0081 - acc: 0.9987 - val_loss: 0.3602 - val_acc: 0.8958 - 358ms/epoch - 5ms/step
Epoch 41/50

Epoch 41: val_acc did not improve from 0.90208
70/70 - 0s - loss: 0.0072 - acc: 0.9996 - val_loss: 0.3587 - val_acc: 0.8969 - 355ms/epoch - 5ms/step
Epoch 42/50

Epoch 42: val_acc improved from 0.90208 to 0.90417, saving model to /content/drive/Othercomputers/My Laptop/Documents/SKRIPSI/Aryo - CNN - FastText/CNN_FastText.h5
70/70 - 0s - loss: 0.0060 - acc: 1.0000 - val_loss: 0.3618 - val_acc: 0.9042 - 389ms/epoch - 6ms/step
Epoch 43/50

Epoch 43: val_acc did not improve from 0.90417
70/70 - 0s - loss: 0.0066 - acc: 0.9987 - val_loss: 0.3572 - val_acc: 0.9010 - 353ms/epoch - 5ms/step
Epoch 44/50

Epoch 44: val_acc did not improve from 0.90417
70/70 - 0s - loss: 0.0051 - acc: 1.0000 - val_loss: 0.3792 - val_acc: 0.8990 - 356ms/epoch - 5ms/step
Epoch 45/50

Epoch 45: val_acc did not improve from 0.90417
70/70 - 0s - loss: 0.0044 - acc: 1.0000 - val_loss: 0.3777 - val_acc: 0.9031 - 364ms/epoch - 5ms/step
Epoch 46/50

Epoch 46: val_acc did not improve from 0.90417
70/70 - 0s - loss: 0.0040 - acc: 0.9996 - val_loss: 0.3885 - val_acc: 0.8990 - 354ms/epoch - 5ms/step
Epoch 47/50

Epoch 47: val_acc did not improve from 0.90417
70/70 - 0s - loss: 0.0038 - acc: 1.0000 - val_loss: 0.3811 - val_acc: 0.8958 - 352ms/epoch - 5ms/step
Epoch 48/50

Epoch 48: val_acc did not improve from 0.90417
70/70 - 0s - loss: 0.0040 - acc: 0.9996 - val_loss: 0.3915 - val_acc: 0.8979 - 349ms/epoch - 5ms/step
Epoch 49/50

Epoch 49: val_acc did not improve from 0.90417
70/70 - 0s - loss: 0.0032 - acc: 1.0000 - val_loss: 0.3836 - val_acc: 0.8990 - 336ms/epoch - 5ms/step
Epoch 50/50

Epoch 50: val_acc did not improve from 0.90417
70/70 - 0s - loss: 0.0035 - acc: 1.0000 - val_loss: 0.3893 - val_acc: 0.9021 - 338ms/epoch - 5ms/step
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
Validation Loss: [0.7388646006584167, 0.672521710395813, 0.6307746171951294, 0.5736809372901917, 0.5108854280281067, 0.47789791226387024, 0.4431992173194885, 0.402
```

2. Model Convolutional Neural Network

a. Pembagian Data 90:10

```
Epoch 1/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 7s 23ms/step - accuracy: 0.6165 - loss: 0.6490 - val_accuracy: 0.8500 - val_loss: 0.4324
Epoch 2/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 4s 23ms/step - accuracy: 0.8917 - loss: 0.3647 - val_accuracy: 0.8813 - val_loss: 0.3540
Epoch 3/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 6s 32ms/step - accuracy: 0.9237 - loss: 0.2419 - val_accuracy: 0.9000 - val_loss: 0.3782
Epoch 4/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 9s 26ms/step - accuracy: 0.9428 - loss: 0.1676 - val_accuracy: 0.8813 - val_loss: 0.4321
Epoch 5/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 7s 38ms/step - accuracy: 0.9525 - loss: 0.1173 - val_accuracy: 0.8906 - val_loss: 0.4469
Epoch 6/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 8s 24ms/step - accuracy: 0.9703 - loss: 0.0889 - val_accuracy: 0.8844 - val_loss: 0.5157
Epoch 7/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 6s 34ms/step - accuracy: 0.9745 - loss: 0.0762 - val_accuracy: 0.8844 - val_loss: 0.5539
Epoch 8/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 9s 24ms/step - accuracy: 0.9876 - loss: 0.0436 - val_accuracy: 0.8938 - val_loss: 0.5621
Epoch 9/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 7s 36ms/step - accuracy: 0.9882 - loss: 0.0360 - val_accuracy: 0.8875 - val_loss: 0.5628
Epoch 10/10
180/180 ━━━━━━━━━━━━━━━━━━━━ 8s 24ms/step - accuracy: 0.9917 - loss: 0.0331 - val_accuracy: 0.8969 - val_loss: 0.6431
<keras.src.callbacks.history.History at 0x79a0f02bd540>
```

b. Pembagian Data 80:20

```
Epoch 1/10
160/160 ━━━━━━━━━━━━━━━━━━━━ 8s 36ms/step - accuracy: 0.6202 - loss: 0.8456 - val_accuracy: 0.8594 - val_loss: 0.4379
Epoch 2/10
160/160 ━━━━━━━━━━━━━━━━━━━━ 4s 22ms/step - accuracy: 0.8884 - loss: 0.3854 - val_accuracy: 0.8766 - val_loss: 0.3596
Epoch 3/10
160/160 ━━━━━━━━━━━━━━━━━━━━ 5s 23ms/step - accuracy: 0.9179 - loss: 0.2592 - val_accuracy: 0.8844 - val_loss: 0.3761
Epoch 4/10
160/160 ━━━━━━━━━━━━━━━━━━━━ 5s 33ms/step - accuracy: 0.9383 - loss: 0.1759 - val_accuracy: 0.8797 - val_loss: 0.4497
Epoch 5/10
160/160 ━━━━━━━━━━━━━━━━━━━━ 9s 24ms/step - accuracy: 0.9546 - loss: 0.1171 - val_accuracy: 0.8781 - val_loss: 0.5058
Epoch 6/10
160/160 ━━━━━━━━━━━━━━━━━━━━ 5s 34ms/step - accuracy: 0.9613 - loss: 0.1056 - val_accuracy: 0.8750 - val_loss: 0.6089
Epoch 7/10
160/160 ━━━━━━━━━━━━━━━━━━━━ 8s 23ms/step - accuracy: 0.9801 - loss: 0.0647 - val_accuracy: 0.8797 - val_loss: 0.6469
Epoch 8/10
160/160 ━━━━━━━━━━━━━━━━━━━━ 5s 33ms/step - accuracy: 0.9896 - loss: 0.0412 - val_accuracy: 0.8781 - val_loss: 0.6693
Epoch 9/10
160/160 ━━━━━━━━━━━━━━━━━━━━ 4s 24ms/step - accuracy: 0.9939 - loss: 0.0242 - val_accuracy: 0.8813 - val_loss: 0.7251
Epoch 10/10
160/160 ━━━━━━━━━━━━━━━━━━━━ 4s 23ms/step - accuracy: 0.9957 - loss: 0.0190 - val_accuracy: 0.8750 - val_loss: 0.8211
<keras.src.callbacks.history.History at 0x79a8e26e8d90>
```

c. Pembagian Data 70:30

```
Epoch 1/10
140/140 ━━━━━━━━━━━━━━━━━━━━ 6s 28ms/step - accuracy: 0.6317 - loss: 0.8437 - val_accuracy: 0.8604 - val_loss: 0.4402
Epoch 2/10
140/140 ━━━━━━━━━━━━━━━━━━━━ 5s 24ms/step - accuracy: 0.8965 - loss: 0.3605 - val_accuracy: 0.8854 - val_loss: 0.3600
Epoch 3/10
140/140 ━━━━━━━━━━━━━━━━━━━━ 6s 33ms/step - accuracy: 0.9325 - loss: 0.2276 - val_accuracy: 0.8854 - val_loss: 0.3869
Epoch 4/10
140/140 ━━━━━━━━━━━━━━━━━━━━ 3s 24ms/step - accuracy: 0.9471 - loss: 0.1530 - val_accuracy: 0.8896 - val_loss: 0.4099
Epoch 5/10
140/140 ━━━━━━━━━━━━━━━━━━━━ 3s 24ms/step - accuracy: 0.9548 - loss: 0.1138 - val_accuracy: 0.8782 - val_loss: 0.4784
Epoch 6/10
140/140 ━━━━━━━━━━━━━━━━━━━━ 4s 32ms/step - accuracy: 0.9697 - loss: 0.0874 - val_accuracy: 0.8625 - val_loss: 0.5202
Epoch 7/10
140/140 ━━━━━━━━━━━━━━━━━━━━ 4s 24ms/step - accuracy: 0.9829 - loss: 0.0574 - val_accuracy: 0.8823 - val_loss: 0.5655
Epoch 8/10
140/140 ━━━━━━━━━━━━━━━━━━━━ 5s 24ms/step - accuracy: 0.9882 - loss: 0.0384 - val_accuracy: 0.8906 - val_loss: 0.6588
Epoch 9/10
140/140 ━━━━━━━━━━━━━━━━━━━━ 7s 38ms/step - accuracy: 0.9885 - loss: 0.0457 - val_accuracy: 0.8781 - val_loss: 0.6541
Epoch 10/10
140/140 ━━━━━━━━━━━━━━━━━━━━ 3s 23ms/step - accuracy: 0.9912 - loss: 0.0284 - val_accuracy: 0.8813 - val_loss: 0.7371
<keras.src.callbacks.history.History at 0x79a060bd7190>
```

Lampiran 6. Dataset Uji Hasil Prediksi

	Ulasan	Label Sebenarnya	Prediksi	
2	sangat mempermudah kerja RT RW dan dengan cepat mengetahui keadaan Jakarta saat ini	Positif	Positif	1
3	Emang sudah tutup ya ⭐ satu saja tidak bisa daftar Oy kerja kerja jangan duduk cuman bilang maaf saja Emang sedang lebaran apa ? ?	Negatif	Negatif	1
4	Bagus dan memudahkan warga DKI pada saat pandemi covid 19 untuk berbagai urusan dgn Pemprov DKI	Positif	Positif	1
5	Sangat tepat untuk inforamsi para warga masyarakat	Positif	Positif	1
6	Saran Alangkah baiknya di aplikasi JAKI ini ada menu CCTV online live jakarta baik itu cctv binamarga cctv etle cctv traffic line dan CCtv yg lain lain Trimss	Netral	Negatif	0
7	Semoga terus dikembangkan dan diperbaiki kekurangan yg ada Thanks	Positif	Positif	1
8	Gabisa daftar vaksin karena waktu nya ga keluar	Negatif	Negatif	1
9	ci tl nya lama harusnya jika laporan dengan detail adanya transaksi barang haram jam sekian jam sekian harusnya ditindak cepat karena meresahkan warga	Negatif	Negatif	1
10	aplikasi yang tidak bagus	Negatif	Negatif	1
11	Aplikasi bobrok selalu ga bisa buat dftr vaksin ke 2	Negatif	Negatif	1
12	Ini kenapa ga bisa login yaudah di coba berkali2	Negatif	Negatif	1
13	Biar bintang bicara	Netral	Negatif	0
14	Sangat terbantu buat pendaftran vaksin tanpa kontakthanks jaki semoga applikasinya jauh lebih baik	Positif	Positif	1
15	Aplikasi yang sangat bermanfaat bagi warga Jakarta Semoga lancar penggunaannya	Positif	Positif	1
16	Maaf terjadi kesalahan pada sistem Coba ulangi beberapa saat lagi ya Sistem tolong di benerin	Negatif	Negatif	1
17	ada pertanyaan tapi ga di respon	Negatif	Negatif	1

456	Sangat berguna dan fast Respon	Positif	Positif	1
457	Maaf saya kasih bitang 1 Kami sekeluarga aga kecewa Jadwal vaksin pun tidak jelas Sudah tertera di apl ini dan di kartu vaksin tanggalnya Lalu berubah Dar	Negatif	Negatif	1
458	Sepertinya saya ada kesalahan cantumin no tlp pada saat pendaftaran vaksin kmrn tgl 11 Juli 2021 di stadion GBK mk nya sampai sekarang saya blm dapat not	Negatif	Negatif	1
459	Aplikasi baguuuuuus buruan download	Positif	Negatif	0
460	luar biasasangat membantuterimakasih	Positif	Positif	1
461	Jakclm di aplikasi kenapa ga ada	Negatif	Negatif	1
462	Berita isinya coviid mulu ga kreatif	Negatif	Negatif	1
463	Bagus si tapi sayang aplikasi ini online	Positif	Netral	0
464	Laporan menjadi mudah informasi jadi terbuka	Positif	Positif	1
465	Sangat membantu untuk melaporkan hal hal yang melanggar peraturan	Positif	Positif	1
466	Berfaedah atau bermanfaat	Positif	Netral	0
467	Aplikasi yang very Useful buat warga Jakarta banyak informasi update didalamnya termasuk info pandemi Covid19 untuk yang belum vaksin covid19 bisa dat	Positif	Positif	1
468	Ini knapa pdhal email user sama password nya benar tapi ga bisa daftar katanya ga sesuai	Negatif	Negatif	1
469	Baru update mudah2an tambah bagus	Netral	Netral	1
470	Sedang mencoba Mudah2 an bisa bermanfaat	Netral	Negatif	0
471	Aplikasi Pempov yg sangat bagus LaporResponPantau Semoga Kepolisian di Jakarta Punya aplikasi seperti ini	Positif	Positif	1
472	Tidak bisa di gunakan	Negatif	Negatif	1

945	Gak bisa dipake daftar vaksin Mending datang langsung	Negatif	Negatif	1
946	Koq menu JAKCLM hilang dari menu Kenapa ini	Negatif	Negatif	1
947	Aplikasi apaan ini ada aja gk pernah kontak fisik sama yg Positif gk pernah nanganin yg pdp malah di bilang anda tidak aman jgn di jadikan mainan yg kaya gi	Negatif	Negatif	1
948	Cukup bagus menambah wawasan untuk DKI Jajarta	Positif	Positif	1
949	Notifikasi berita update kekinian	Positif	Negatif	0
950	Masih lama loading awal pas masuk bukannya gak sabar tapi emang lama	Negatif	Negatif	1
951	Baru telusuri baru donlode terima kasih semoga bisa membantu	Netral	Positif	0
952	Mau login ke jaki susah bgt gk msuk2 mohon di perbaiki secepatnya	Negatif	Negatif	1
953	Gausah bnyak gaya bikin apl lah Berikan fasilitas pelayanan lapangan aja dlu yg bner Apl ga guna bnyak bug luding ga kelar2 pdhal sinyal dah 4G udh pke wifi	Negatif	Negatif	1
954	Aplikasi gajelas Bilangnya jakarta smart Bikin app aja kualitas buruk Bilang smart	Negatif	Negatif	1
955	Gak bisa di pencet pilihannya padahal waktu itu bisa kenapa jadi gak bisa Jadi gak bisa isi jakclm nih	Negatif	Negatif	1
956	Ini ngapa dah aplikasisaya udah vaksin keduaterus malah udah pengen k3 eh liat status vaksinasi malah belom sama sekaliudah ghitu Mao login juga kaga bi	Negatif	Negatif	1
957	Aplikasi gak berguna verifikasi by email gak ada masuk apa yg mau di verifikasi	Negatif	Negatif	1
958	Apakah kita tulis verifikasi nik KTP Di profil jaki gak Takutnya identitas saya kebaca	Netral	Negatif	0
959	Aplikasi jaki Kalau kasih info vaksin 12 yg lengkap memang jenus vaksinnya bener tp karna ga semua tempat ad vaksin 1contoh kemaren info vaksin jaki di LI	Negatif	Negatif	1
960	Gak jelas Daftar vaksin gak bisa bisa Sudah masukin semua data tapi keteranganya faskes tidak tersedia terus	Negatif	Negatif	1
961	Sangat membantu proses informasi ekonomi dan absensi	Positif	Positif	1

Lampiran 7. Hasil Uji Plagiat

MAJELIS PENDIDIKAN TINGGI PIMPINAN PUSAT MUHAMMADIYAH
UNIVERSITAS MUHAMMADIYAH MAKASSAR
UPT PERPUSTAKAAN DAN PENERBITAN
Alamat kantor: Jl Sultan Alauddin NO 259 Makassar 90221 Tlp (0411) 866972,881593, Fax (0411) 865588

بسم الله الرحمن الرحيم

SURAT KETERANGAN BEBAS PLAGIAT

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

Nama : Aryo Diningrat Salea
Nim : 105841108820
Program Studi : Teknik Informatika

Dengan nilai:

No	Bab	Nilai	Ambang Batas
1	Bab 1	8 %	10 %
2	Bab 2	23 %	25 %
3	Bab 3	8 %	10 %
4	Bab 4	3 %	10 %
5	Bab 5	3 %	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, 26 Agustus 2024
Mengetahui,

Kepala UPT- Perpustakaan dan Penerbitan,

Nursiah, 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 Aryo Diningrat Salea 105841108820

by TahapTutup

Submission date: 26-Aug-2024 02:10PM (UTC+0700)
Submission ID: 2438249323
File name: BAB_I_SKRIPSI_ARYO_DININGRAT.docx (22.85K)
Word count: 771
Character count: 4995

3 I Aryo Diningrat Salea 105841108820

.GINALITY REPORT

8%	6%	0%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	mafiadoc.com Internet Source	3%
2	eprints.umm.ac.id Internet Source	2%
3	Submitted to Vietnam Buddhist Institute Student Paper	2%
4	repository.radenintan.ac.id Internet Source	2%

Exclude quotes Off
Exclude bibliography Off
Exclude matches

BAB II Aryo Diningrat Salea
105841108820
by Tahap Tutup

Submission date: 26-Aug-2024 02:10PM (UTC+0700)
Submission ID: 2438249665
File name: BAB_II_SKRIPSI_ARYO_DININGRAT.docx (177.87K)
Word count: 1852
Character count: 12453

ɜ II Aryo Diningrat Salea 105841108820

GINALITY REPORT

23%	20%	2%	12%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Universitas Muhammadiyah Makassar Student Paper	7%
2	jurnal.buddhidharma.ac.id Internet Source	5%
3	www.jurnal.iaii.or.id Internet Source	3%
4	garuda.kemdikbud.go.id Internet Source	2%
5	jtiik.ub.ac.id Internet Source	2%
6	repository.teknokrat.ac.id Internet Source	2%
7	repository.uin-suska.ac.id Internet Source	2%

Exclude quotes Off
Exclude bibliography Off
Exclude matches < 2%

BAB III Aryo Diningrat Salea 105841108820

by TahapTutup

Submission date: 26-Aug-2024 02:11PM (UTC+0700)
Submission ID: 2438249955
File name: BAB_III_SKRIPSI_ARYO_DININGRAT.docx (113.85K)
Word count: 1007
Character count: 6653

AB III Aryo Diningrat Salea 105841108820

ORIGINALITY REPORT

8%	8%	8%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Binus University International Student Paper	3%
2	core.ac.uk Internet Source	3%
3	ejournal.itn.ac.id Internet Source	2%

BAB IV Aryo Diningrat Salea
105841108820
by TahapTutup

Submission date: 26-Aug-2024 02:12PM (UTC+0700)
Submission ID: 2438250396
File name: BAB_IV_SKRIPSI_ARYO_DININGRAT.docx (6.34M)
Word count: 5853
Character count: 42802

BAB IV Aryo Diningrat Salea 105841108820

ORIGINALITY REPORT

3%	5%	2%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	digilibadmin.unismuh.ac.id Internet Source	3%

LULUS

turnitin

Exclude quotes Off

Exclude bibliography Off

Exclude matches < 2%

BAB V Aryo Diningrat Salea 105841108820

by TahapTutup

Submission date: 26-Aug-2024 02:52PM (UTC+0700)
Submission ID: 2438265560
File name: BAB_V_SKRIPSI_ARYO_DININGRAT.docx (16.18K)
Word count: 260
Character count: 1732

.B V Aryo Diningrat Salea 105841108820

IGINALITY REPORT

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

PRIMARY SOURCES

1	pt.scribd.com Internet Source	3%

LULUS

turnitin

Exclude quotes Off

Exclude bibliography Off

Exclude matches < 2%