EVALUASI KINERJA HIBRIDA BERBASIS CNN UNTUK TUGAS KLASIFIKASI SENTIMEN BERITA ONLINE

Skripsi

Oleh

GADING ARYA DWI CAHYO
NPM. 2157031013

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

ABSTRACT

CNN-BASED HYBRID PERFORMANCE EVALUATION TOWARDS ONLINE NEWS SENTIMENT CLASSIFICATION TASK

By

Gading Arya Dwi Cahyo

CNN is a deep learning model that is effective in extracting features in text data. However, CNN has shortcomings in understanding long-term context and lacks sensitivity to word order. In this study, a CNN model was hybridized with a machine learning model to improve the model's performance. The LSTM model and BiLSTM model were chosen because they have long-term memory features and sensitivity to word order. Apart from that, other machine learning models such as TF-IDF, Word2Vec, and BERT are used to add features that are useful in extracting text data. In model testing, the accuracy, precision, recall, F1-score, and AUC-ROC matrices are used. The evaluation results of all models are compared to determine the influence of vector dimensions and obtain the best model. From the evaluation results, it was found that the RoBERTa-CNN-BiLSTM model had the best performance with a matrix accuracy of 98.18%, precision 98.19%, recall 98.18%, F1-score 98.18%, and AUC-ROC 99.86%. Apart from that, it is known that the vector 38 dimension provides the most superior performance. These results indicate that hybridized deep learning models, especially CNNs, are effective in improving model performance.

Keywords: CNN, LSTM, BiLSTM, Word2Vec, TF-IDF, RoBERTa, Online News, NLP.

ABSTRAK

EVALUASI KINERJA HIBRIDA BERBASIS CNN UNTUK TUGAS KLASIFIKASI SENTIMEN BERITA ONLINE

Oleh

Gading Arya Dwi Cahyo

CNN merupakan model deep learning yang efektif dalam mengekstraksi fitur pada data teks. Namun, CNN memiliki kelemahan dalam memahami konteks jangka panjang dan kurang sensitif terhadap urutan kata. Dalam penelitian ini, model CNN dihybridisasi dengan model machine learning untuk meningkatkan performa model. Model LSTM dan BiLSTM dipilih karena memiliki kemampuan memori jangka panjang serta sensitivitas terhadap urutan kata. Selain itu, model machine learning lainnya seperti TF-IDF, Word2Vec, dan BERT digunakan untuk menambahkan fitur yang berguna dalam ekstraksi data teks. Pada pengujian model, matriks akurasi, presisi, recall, F1-score, dan AUC-ROC digunakan. Hasil evaluasi dari semua model dibandingkan untuk mengetahui pengaruh dimensi vektor dan memperoleh model terbaik. Dari hasil evaluasi, ditemukan bahwa model RoBERTa-CNN-BiLSTM memiliki performa terbaik dengan nilai akurasi 98,18%, presisi 98,19%, recall 98,18%, F1-score 98,18%, dan AUC-ROC 99,86%. Selain itu, diketahui bahwa dimensi vektor 38 memberikan performa paling unggul. Hasil ini menunjukkan bahwa model deep learning yang dihybridisasi, khususnya CNN, efektif dalam meningkatkan performa model.

Kata-kata kunci: CNN, LSTM, BiLSTM, Word2Vec, TF-IDF, RoBERTa, Berita Online, NLP.

EVALUASI KINERJA HIBRIDA BERBASIS CNN UNTUK TUGAS KLASIFIKASI SENTIMEN BERITA ONLINE

GADING ARYA DWI CAHYO

Skripsi

Sebagai Salah Satu Syarat untuk Memperoleh Gelar
SARJANA MATEMATIKA

Pada

Jurusan Matematika

Fakultas Matematika dan Ilmu Pengetahuan Alam

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

Judul Skripsi	:	**EVALUASI KINERJA HIBRIDA BERBASIS CNN UNTUK TUGAS KLASIFIKASI SENTIMEN BERITA ONLINE**
Nama Mahasiswa	:	Gading Arya Dwi Cahyo
Nomor Pokok Mahasiswa	:	**2157031013**
Program Studi	:	**Matematika**
Fakultas	:	**Matematika dan Ilmu Pengetahuan Alam**

MENYETUJUI

1. Komisi Pembimbing

Dr. Aang Nuryaman, S.Si., M.Si.	**Dr. Purnomo Husnul Khotimah, M.T.**
NIP.197403162005011001	NIP.198003232005022002

2. Ketua Jurusan Matematika

Dr. Aang Nuryaman, S.Si., M.Si.
NIP.197403162005011001

MENGESAHKAN

1. tim penguji

Ketua	:	**Dr. Aang Nuryaman, S.Si., M.Si.**	.....
Sekretaris	:	**Dr. Purnomo Husnul Khotimah, M.T.**	.....
Penguji Bukan Pembimbing	:	**Dr. Khoirin Nisa, S.Si., M.Si.**	.....

2. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

Dr. Eng. Heri Satria, S.Si., M.Si.
NIP. 197110012005011002

Tanggal Lulus Ujian Skripsi: **02 Juni 2025**

PERNYATAAN SKRIPSI MAHASISWA

Yang bertanda tangan di bawah ini:

Nama	:	**Gading Arya Dwi Cahyo**
Nomor Pokok Mahasiswa	:	**2157031013**
Jurusan	:	**Matematika**
Judul Skripsi	:	**Evaluasi Kinerja Hibrida Berbasis CNN untuk Tugas Klasifikasi Sentimen Berita Online**

Dengan ini menyatakan bahwa skripsi ini adalah hasil pekerjaan saya sendiri. Apabila kemudian hari terbukti bahwa skripsi inni merupakan hasil salinan atau dibuat oleh orang lain, maka saya bersedia menerima sanksi sesuai degnan ketentuan akademik yang berlaku.

Bandar Lampung,

Penulis,

Gading Arya Dwi Cahyo

RIWAYAT HIDUP

Penulis memiliki nama lengkap Gading Arya Dwi Cahyo yang lahir di Bandar Lampung pada tanggal 11 Januari 2002. Penulis merupakan anak kedua dari tiga bersaudara pasangan Bapak Edi Sucahyo dan Ibu Suparmi.

Penulis menempuh pendidikan dasar di SDN 2 Sumberrejo pada tahun 2008-2014. Penulis elanjutkan Sekolah Menengah Pertama di SMPN 4 Bandar Lampung pada tahun 2014-2017 dan Sekolah Menengah Atas di SMAN 2 Bandar Lampung pada tahun 2017-2020. Pada tahun 2021, penulis melanjutkan studi di Universitas Lampung Fakultas MIPA Jurusan Matemaika.

Pada tahun 2024, penulis melakasanakan Kerja Praktik Lapangan (PKL) di Badan Riset dan Inovasi Nasional (BRIN) KST Samaun Samadikun yang berlokasi di Bandung dan melanjutkan program magang riset Merdeka Belajar Kampus Merdeka (MBKM) ditempat yang sama.

KATA INSPIRASI

”If a God is exist, Tuhan tidak akan menempatkan mahluknya di suatu tempat tertentu tanpa adanya tujuan”
-Arvi Hasanah

”I have no enemies, that nobody has them, nobody in this entire world deserves to get HURT”
-Thors

”Realitas diciptakan oleh ketidakadilan, maka berdamailah ketidakadilan itu”
-ElephantSun

”Nampaknya tidak ada yang bisa mengalahkan kenyataan, dan kamu harus mencintai kenyataanmu”
-Bryan Furran

”Hidup dimulai ketika lu hampir mati”
-Indra Firmawan

”Kita sekarang adalah hasil dari perjalanan hidup kita”
-Indra Firmawan

”Hidup adalah masalah yang diselingi oleh kebahagian”
-Ferry Irwandi

”Jangan takut dengan orang menguasai 10.000 jenis tendangan, tapi takutlah dengan orang yang melatih satu tedangan selama 10.000 kali. Dalam prosesnya kita akan salah, keliru, terpukul, dan banyak hal hal lain yang harus dihadapi. Tapi ingat, tidak pernah ada sesuatu yang terbentuk tanpa pernah terbentur. Terbentur, terbentur, terbentuk.”
-Ferry Irwandi

”Seiring ilmu pengetahuan bertambah, seharusnya yang tumbuh itu kebijaksanaan kita, buka ego kita”
-Ferry Irwandi

”Walaupun terlihat mustahil, lebih baik mencoba dan gagal, dari pada diam dan gak melakukan apa-apa”
-Coki Pardede

”If you want a thing done well, do it yourself”
-Napoleon Bonaparte

”Tidak ada badai yang tidak usai, cepat atau lambat tiap luka akan pulih dan mengering, mungkin meninggalkan bekas tapi tidak lagi menyakitkan, selama lu percaya hari itu akan datang”
-Ferry Irwandi

”Hidup itu bukan kompetisi, tapi ada satu hal yang harus lu menangkan, yaitu dengan diri lu sendiri”
-Ferry Irwandi

”Gw ingin hidup bahkan ketika mati gw, tubuh organik ini tidak akan bisa mewujudkan mimpi tersebut, idea is a bulletproof. Gw berusaha mempertahankan hidup ini selama lamanya, melalui mata yang melihat kuping yang mnedengar dan otak yang berfikir, and then lets do it.”
-Ferry Irwandi

”If sometimes gw percaya dengan Tuhan, gw dengan penuh keyakinan akan mempetahankan hidup gw, karena Tuhan telah mempercayakan gw untuk membantunya menulis takdir diri gw sendiri dan orang disekitar gw.”
-ElephantSun

”Hidup adalah tentang perjalanan, jika satu waktu kamu menemukan hal buruk menimpah mu, maka kamu hanya harus tertaha, dan semesta akan membawamu melewatinya”
-ElephantSun

”Hal terbaik dari mengejar sesuatu adalah perjalanan dan pelajaran yang selalu didapat pada setiap langkahnya”
-ElephantSun

”You can’t control the wind, but you can adjust your sails, it’s not happens to you it’s how you react, to make a better choice”

”To make something special you just have to believe it’s special”
-Ping

”Yesterday is history, Tomorrow is mystery, and Today is a gift” -Oogway

”Make sure you save yourself before you save others”
-Eno Bening

”Tidak ada yang salah dari sebuah pilihan, yang salah adalah ketika lu memilih dan lu mengeluh, dan yang bodoh adalah ketika sudah mengeluh tidak mencoba pilihan lain”
-Dzawin Nur

”Sebijaknya kata-kata, tetap lebih bijak seseorang yang dapat mengimplementasikan setiap kalitmat menjadi menjadi perbuatan nyata”
-wira Negara

”Bajumu membuatmu terhormat sebelum duduk, Akalmu membuatmu terhormat setelah duduk”
-Dzawin Nur

PERSEMBAHAN

Dalam terselesaikannya skripsi ini, saya persembahkan rasa terimakasih saya kepada:

Ayah dan Ibu

Terimakasih kepada orang tuaku atas segala pengorbanan, motivasi, dan izin serta dukungannya selama ini. Terimakasih telah memberikan pelajaran berharga kepada anakmu ini tentang makna perjalanan hidup yang sebenarnya sehingga kelak bisa menjadi orang yang bermanfaat bagi banyak orang.

Dosen Pembimbing dan Pembahas

Terimakasih kepada dosen pembimbing dan pembahas yang sudah sangat membantu, memberikan motivasi, memberikan arahan serta ilmu yang berharga.

Orang-orang di sekitarku

Terimakasih kepada semua orang yang telah memberikan pengalaman, semangat, motivasinya, serta senantiasa memberikan dukungan dalam beberapa hal.

Almamater Tercinta
Universitas Lampung

SANWACANA

Terima kasih kepada Semesta yang telah mengizinkan penulis menyelesaikan skripsi ini yang berjudul ”Evaluasi Kinerja Hibrida Berbasis CNN untuk Tugas Klasifikasi Sentimen Berita Online” dengan baik dan lancar serta tepat pada waktu yang telah ditentukan.

Dalam proses penyusunan skripsi ini, banyak pihak yang telah membantu memberikan bimbingan, dukungan, arahan, motivasi serta saran sehingga skripsi ini dapat terselesaikan. Oleh karena itu, dalam kesempatan ini penulis mengucapkan terimakasih kepada:

1. Bapak Dr. Aang Nuryaman, S.Si.,M.Si. selaku Pembimbing 1 yang telah banyak meluangkan waktunya untuk memberikan arahan, bimbingan, motivasi, saran serta dukungan kepada penulis sehingga dapat menyelesaikan skripsi ini.

2. Ibu Dr. Purnomo Husnul Khotimah, M.T. selaku Pembimbing II yang telah memberikan arahan, bimbingan dan dukungan kepada penulis sehingga dapat menyelesaikan skripsi ini.

3. Ibu Dr. Khoirin Nisa, S.Si., M.Si. selaku Penguji yang telah bersedia memberikan kritik dan saran serta evaluasi kepada penulis sehingga dapat menjadi lebih baik lagi.

4. Bapak Dr. Aang Nuryaman, S.Si., M.Si. selaku Ketua Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung.

5. Bapak Andri Fachrur Rozie S.Kom., M.Eng. selaku salah satu pembimbing magang MBKM dari Pusat Riset Sains Data dan Informasi.

6. Bapak Drs. Nursyirwan, M.Si. selaku dosen pembimbing akademik.

7. Seluruh dosen, staff dan karyawan Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung.
8. Seluruh peneliti, staff dan karyawan BRIN KST Samaun Samadikun Bandung
9. Ayah dan ibu yang selalu memberi dukungan sepenuhnya dalam semua hal kepada penulis,serta kakak, adik, sepupu, paman, tante dan orang-orang disekitar penulis yang telah memberikan dukugan dalam bentuk apapun, sehingga skripsi ini dapat diselesaikan sebagaimana harusnya.

Semoga skripsi ini dapat bermanfaat bagi kita semua. Penulis menyadari bahwa skripsi ini masih jauh dari sempurna, sehingga penulis mengharapkan kritik dan saran yang membangun untuk menjadikan skripsi ini lebih baik lagi.

Bandar Lampung,

Gading Arya Dwi Cahyo

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Deep learning merupakan salah satu cabang artifial intelegant (AI) yang mengalami perkembangan pesat dalam beberapa dekade terakhir. Dominasi *Convolution Neural Network* (CNN) yang merupakan salah-satu model deep learning dalam kompetisi *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC) di tahun 2012 memantik revolusi deep learning (Serre & Lindsay, 2021). Saat itu, model jaringan yang digunakan adalah AlexNet yang dirancang oleh (Krizhevsky *et al.*, 2012). Model tersebut mencapai accuracy 5 teratas dengan 83,6%, mengungguli sistem terbaik kedua lebih dari 10% dan mengurangi kesalahan dibanding dengan tahun sebelumnya sebesar lebih dari 60% (Russakovsky *et al.*, 2015).

Kebutuhan manusia terhadap teknologi seperti AI mendorong inovasi untuk memperoleh model paling efektif (Chowdhury *et al.*, 2023). Hibridasi model deep learning merupakan salah-satu cara yang lazim dilakukan untuk meningkatkan performa model deep learning (Kapoor *et al.*, 2023). Hybridasi dapat meningkatkan performa model dengan memanfaatkan fitur yang ada pada masing-masing model. CNN memiliki kelemahan dalam memahami konteks jangka panjang, hal ini dapat diatasi oleh model LSTM yang memiliki fitur ingatan jangka panjang (Wu *et al.*, 2020). Pemanfaatan hybridasi model-model ini telah digunakan dalam beberapa penugasan seperti pengenalan citra atau gambar, peramalan, dan analisis sentimen (Tan *et al.*, 2022).

Analisis sentimen merupakan jenis klasifikasi teks yang mengatalogkan teks berdasarkan orientasi sentimen sebuah opini (Devika *et al.*, 2016). Analisis sentimen biasa digunakan dalam mengidentifikasi sentimen review produk tertentu, namun dalam studi ini, analisis sentimen akan digunakan sebagai alat pemantauan kejadi-

an. Pengklasifikasian data teks berdasar pada informasi perkembangan kejadian.

Berita online memiliki potensi sebagai sarana pemantau kejadian. Berita online memiliki keunggulan dalam kecepatan yang medekati real time, serta memiliki tingkat akurasi yang tinggi karena diterbitkan oleh media terakreditasi (Liliana dkk, 2021). Pemantauan kejadian seperti geopolitik, kejadian alam, ekomomi, serta penyebaran penyakit menular dapat dilakukan secara real time. Berkaca pada pandemi COVID-19 di beberapa tahun lalu, pemanfaatan berita online sebagai alat pemantauan penyebaran penyakit menular penting dilakukan dalam bentuk persiapan pemangku kepentingan.

Peningkatan performa deep learning model telah banyak dilakukan. Hibridasi *deep learning models* sering digunakan untuk meningkatkan performa models. Metode hibridasi model CNN dengan LSTM, BiLSTM, dan BERT terbukti meningkatkan performa model. Dalam paper (Zhou, 2022) hybridasi CNN dan LSTM digunakan dalam tugas NLP menghasilkan performa yang lebih baik dibanding original models. Dalam paper tersebut menggunakan dua dataset yang masing-masing sebesar 20000 data. Data berasal dari THUCNews dan crawled taobao review.

Hibridasi model CNN dengan Bi-LSTM akan menambahkan fitur ingatan jangka panjang. Studi dari (Rhanoui *et al.*, 2019) menggunakan model model CNN yang dihibridasi dengan Bi-LSTM dalam penugasan anilisis sentimen teks dengan level dokumen. (Rhanoui *et al.*, 2019), menggunakan dataset yang terdiri dari 2003 artikel prancis dan diperoleh dari nasional dan *internasional papernews* (TelQuel, Aujourd'hui, Le Figaro, and LeMonde, among others). Banyak kata dalam setiap artikel memiliki 4000 kata di rataannya. Dengan penyebaran sentimen 1247 artikel bersentimen netral, 474 bersentimen positif, dan 282 bersentimen negatif. Dalam peper lain, penghibridasian dengan model BERT juga dilakukan (Rahman *et al.*, 2024) dengan dengan hasil RoBERTa-BiLSTM accuracy dan f1-score unggul dibanding hibridasi model lainnya.

Pada penelitian sebelumnya (Nugraheni *et al.*, 2020) model CNN digunakan untuk mengklasifikasikan sentimen judul berita online. Dalam studi tersebut performa terbaik diperoleh oleh CNN tanpa *one-hot encoding* dan penyesuaian tuning dengan accuracy 87.585%, dan F1-score 76.00%. Bagaimanapun, CNN memiliki potensi dalam penugasan bahasa alami, karena mampu mengekstrak fitur penting dalam

sebuah kalimat. Peningkatan performa CNN untuk penugasan klasifikasi sentimen pada studi sebelumnya dengan metode hibridasi untuk mendapatkan model terbaik dalam penugasan ini (Minaee *et al.*, 2019).

Berbagai penelitian terdahulu menunjukkan metode *deep learning* seperti CNN, LSTM, BiLSTM, dan BERT dapat diandalkan dalam melakukan klasifikasi sentimen dengan mendeteksi pola teks. Dalam berbagai penelian, metode hibridasi model *deep learning* berhasil meningkatkan performa model. Metode Hibridasi dapat menggabungkan keunggulan yang dimiliki setiap model yang akan membuat model kaya akan fitur. Model CNN dengan fitur ekstraksi dan LSTM dengan fitur ingatan jangka pajang akan membuat model baik dalam menghadapi data teks, yang merupakan salah satu contoh kelebiahan metode hibridasi. Pendekatan ini terbukti efektif dalam dalam menangkap pola spasial dan temporan dalam data teks, hingga mempunyai potensi besar dalam meningkatkan kinerja model dalam pengolahan data teks.

Berdasar pada latar belakang dan penelitian terdahulu yang telah dipaparkan, penelitian ini berfokus pada peningkatan performa model deep learning dalam mengklasifikasikan data teks pendek terutama sentimen judul berita online. Kami berhipotesis bahwa dengan menghibridasikan model CNN yang memiliki fitur ekstraksi dengan beberapa model seperti LSTM, BiLSTM, dan BERT akan meningkatkan performa model CNN dalam pengolahan data teks pendek terutama judul berita online. CNN sebagai basis model dihibridasi dengan *long short-term memory* (LSTM) dan *bidirectional long short-term memory* (BiLSTM). Selain itu, model seperti TF-IDF dan *Word2Vec* digunakan sebagai teks *encoding*, serta penggunaan *embedding layer* dan BERT. Untuk mengevaluasi model digunakan metrik *accuracy, precision, recall, F1-Score*, dan AUC-ROC.

1.2 Tujuan Penelitian

Tujuan dari penelitian ini yaitu meningkatkan performa kinerja model CNN dengan membandingkan antara model CNN, dengan model CNN *hybrid*, yaitu CNN-LSTM, dan CNN-BiLSTM dan menggunakan beberapa model representasi kata yaitu *Word2Vec*, TF-IDF, dan RoBERTa untuk mendapatkan model yang lebih efektif

dan efisien dalam mengklasifikasi sentimen judul berita online.

1.3 Manfaat Penelitian

Manfaat yang diharapkan dari hasil penelitian ini sebagai berikut:

1) Menambah pengetahuan dan wawasan tentang metode hibrida model CNN, LSTM, BiLSTM, *Word2Vec*, Tf-IDF, dan RoBERTa dalam mengklasifikasi teks pendek terutama sentimen judul berita online.
2) Mengetahui model yang lebih efektif untuk mengklasifikasi judul berita online dari beberapa model yang dipilih.

BAB II

TINJAUAN PUSTAKA

2.1 Sentimen Analisis dan Dataset

Sentimen analisis merupakan salah satu topik penting dalam bidang *natural language processing* (NLP) yang telah mengalami perkembangan pesat dalam beberapa dekade terakhir (Pang & Lee, 2008). Secara umum, sentimen analisis bertujuan untuk memahami, mengekstrak, dan memproses data teks secara otomatis guna mengidentifikasi informasi sentimen yang terkandung dalam sebuah teks (Liu, 2019). Aplikasi awal dari metode ini banyak digunakan untuk menganalisis ulasan produk, konten media sosial, atau dokumen berbasis opini lainnya. Namun, pendekatan sentimen analisis tidak terbatas pada data teks subjektif. Metode ini juga dapat diterapkan pada teks objektif, seperti judul berita, untuk memberikan wawasan yang lebih mendalam mengenai dinamika suatu peristiwa, termasuk situasi penyebaran penyakit menular.

Dalam konteks pandemi COVID-19, sentimen analisis dapat digunakan untuk mengklasifikasikan judul berita ke dalam kategori positif, negatif, atau netral. Klasifikasi ini bertujuan untuk memberikan informasi apakah situasi penyakit menular yang dilaporkan menunjukkan perbaikan, penurunan, atau tetap stabil (Wang *et al.*, 2020). Pendekatan ini memungkinkan data kualitatif yang terdapat dalam teks berita dikonversi menjadi data kuantitatif yang dapat digunakan untuk berbagai keperluan, seperti memprediksi tren penyebaran penyakit atau mendeteksi potensi lonjakan kasus. Oleh karena itu, sentimen analisis menjadi alat penting yang dapat mendukung pengambilan keputusan berbasis data, khususnya dalam upaya mitigasi dampak penyakit menular.

Dataset yang digunakan dalam penelitian ini adalah *Indonesian Corpus for COVID-19 Event Detection* (InaCOVED) yang diperoleh dari Badan Riset dan Inovasi Na-

sional (BRIN) (Khotimah, 2023). Dataset InaCOVED ini berisi 16.821 data judul berita daring terkait COVID-19 yang dikumpulkan dari tujuh portal berita daring di Indonesia, yaitu Tirto, Tempo, Republika, Merdeka, Kompas, Detik, dan Antara, dalam rentang waktu 26 Januari hingga 24 Maret 2020.

Judul-judul berita yang dikumpulkan kemudian diberi label berdasarkan sentimen yang mencerminkan perkembangan situasi pandemi COVID-19. Proses pelabelan ini dibagi ke dalam tiga kategori, yaitu:

- Negatif (-1): Berita yang mengandung informasi tentang memburuknya situasi pandemi, seperti penguncian wilayah (lockdown), peningkatan jumlah kasus infeksi, atau kenaikan angka kematian.
- Netral (0): Berita yang tidak langsung berkaitan dengan situasi pandemi, tetapi lebih pada dampaknya, seperti penurunan nilai tukar mata uang, kenaikan harga kebutuhan pokok, atau larangan impor barang.
- Positif (1): Berita yang memuat informasi tentang perbaikan situasi pandemi, seperti peningkatan jumlah pasien sembuh, penurunan angka infeksi, atau pelonggaran kebijakan lockdown.

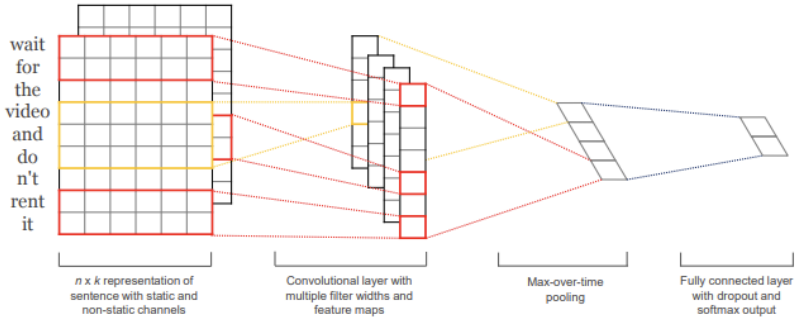
Keberhasilan implementasi sentimen analisis bergantung pada berbagai faktor, seperti tahapan *preprocessing* teks, pemilihan algoritma, serta kualitas dataset yang digunakan. Proses *preprocessing* mencakup berbagai langkah, termasuk tokenisasi, penghapusan stopwords, normalisasi teks, serta stemming atau lemmatization untuk memastikan data yang diolah memiliki kualitas yang baik (Cambria *et al.*, 2013). Setelah melalui tahap pre-processing, algoritma *machine learning* seperti Naive Bayes dan *Support Vector Machine* (SVM), serta model deep learning seperti *Long Short-Term Memory* (LSTM) dan Transformer-based models (misalnya, BERT), dapat digunakan untuk meningkatkan akurasi klasifikasi sentimen (Devlin *et al.*, 2019).

Penerapan sentimen analisis pada berita COVID-19 memiliki manfaat yang signifikan. Sebagai contoh, berita dengan sentimen negatif dapat menggambarkan situasi yang memburuk, seperti peningkatan jumlah kasus atau angka kematian, sedangkan berita dengan sentimen positif dapat mencerminkan adanya penurunan kasus atau keberhasilan pelaksanaan vaksinasi. Dengan demikian, sentimen analisis berperan

penting dalam menganalisis dinamika penyebaran penyakit menular serta memberikan kontribusi yang signifikan dalam mendukung pengambilan keputusan yang lebih efektif di bidang kesehatan masyarakat.

2.2 *Convolution Neural Network*

Convolutional Neural Network (CNN) adalah jenis arsitektur jaringan syaraf tiruan yang sering digunakan dalam pengolahan citra dan pengenalan pola. CNN memanfaatkan konsep konvolusi untuk mengekstrak fitur dari input, yang dapat berupa data gambar, data teks, atau data lainnya. Arsitektur CNN terdiri dari lapisan-lapisan konvolusi yang mengambil data input, mengekstraksi fitur dari data tersebut, dan akhirnya menghasilkan output yang sesuai dengan tugas yang diberikan, seperti klasifikasi gambar dan klasifikasi sentimen (Ouyang *et al.*, 2015).

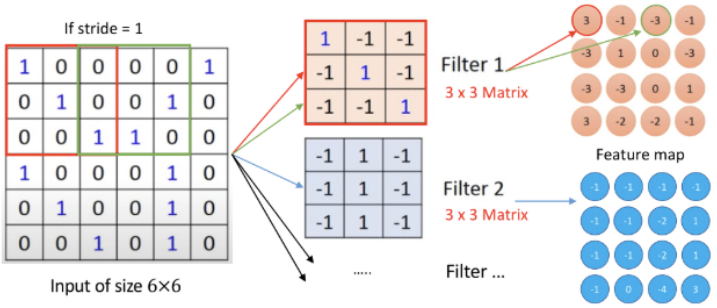


Gambar 2.1 Arsitektur CNN
Sumber: (Kim, 2014)

Secara umum, model CNN terdiri dari beberapa lapisan yaitu *convolutional layer*, *activation layer* , *pooling layer*, *fully-connected layer*, dan *output layer*, dapat dilihat pada Gambar 2.1. Lapisan-lapisan ini bertanggung jawab pada fungsinya masing-masing. *convolutional layer* berfungsi untuk mengekstraksi fitur dari data input, *activation layer* berfungsi untuk mengintroduksi non-linearitas ke jaringan, *pooling layer* berfungsi untuk mengurangi dimensi data untuk menurunkan kompleksitas komputasi, *fully-connected layer* berfungsi untuk menggabungkan fitur yang diekstraksi untuk menghasilkan prediksi akhir, dan *output layer* menghasilkan hasil akhir dari jaringan (Khan *et al.*, 2020).

2.2.1 *Covolutional layer*

Covolutional layer merupakan lapisan inti dari model CNN. Secara singkat, data input dengan bentuk tertentu yang melewati lapisan ini akan diubah menjadi peta fitur (*feature map*). kumpulan fiter atau kernel yang dapat dipelajari memainkan peran penting dalam proses ini. Gambar 2.2 memberikan penjelasan yang lebih intuitif tentang *convolutional layar* (Beaglehole *et al.*, 2023).



Gambar 2.2 Struktur operasional dasar lapisan konvolusional
Sumber: (Yang, 2020)

Input pada *Neural Network* diasumsikan sebagai matrix 6 x 6, di mana setiap elemen dapat direpresentasikan sebagai bilangan bulat '0' atau '1'. Dalam lapisan konvolusi terdapat sekumpulan filter yang dapat dipelajari, dan setiap filter dapat diasumsika sebagai matrix, mirip dengan *neuron* dalam lapisan *fully-connected*. Dalam contoh yang diberikan, filter berukuran 3 x 3 bergerak dengan langkah tertentu, dan setiap elemen matriks atau filter berfungsi sebagai parameter (berat dan bias) dari *neural network*.

Filter aktf berukuran 3 x 3 memiliki kemampuan untuk mendeteksi pola dengan ukuran yang sama pada posisi spesial tertentu dalam *input*. Operasi aljabar berikut menjelaskan proses transformasi dari input ke peta fitur (*feature map*).

$$X := \begin{pmatrix} X_{11} & X_{12} & \cdots & X_{16} \\ X_{21} & X_{22} & \cdots & X_{26} \\ \vdots & \vdots & \ddots & \vdots \\ X_{61} & X_{62} & \cdots & X_{66} \end{pmatrix}$$

Di mana X merupakan matriks input;

$$F := \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{pmatrix}$$

Di mana F merupakan filter berukuran 3×3;

$$\beta := \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{33} \end{pmatrix}$$

Di mana β merupakan matriks atau filter yang sudah di-*unrolled*;

$$A_{11} = (F \times X)_{11} := \beta \cdot \begin{pmatrix} X_{11} & X_{12} & X_{13} & X_{21} & X_{22} & X_{23} & X_{31} & X_{32} & X_{33} \end{pmatrix}^T \tag{2.2.1}$$

Elemen pertama pada peta fitur (*feature map*) adalah A_{11} yang dapat dihitung melalui operasai *dot product* seperti yang telah dijelasakan diatas. Secara berurutan, elemen kedua dari peta fitur ditentukan oleh operasi dot product sliding filter dan matriks input berikutnya dengan ukuran yang sama setelah mengatur nilai stride tertentu, yang dapat dianggap sebagai jarak ergeseran. Setelah seluruh proses, peta fitur berukuran 4×4 telah dihasilkan. Secara umum, terdapat lebih dari satu filter dalam lapisan konvolusi, dan setiap filter menghasilkan peta fitur dengan ukuran yang sama. Hasil dari lapisan konvolusi ini adalah beberapa peta fitur (juga disebut peta aktivasi), dan peta fitur ini, yang sesuai dengan filter yang berbeda, ditumpuk bersama sepanjang dimensi kedalaman (Putra, 2016).

2.2.2 ReLU *Layer*

Pada setiap jaringan saraf konvolusional, layer non-linear atau dikenal sebagai layer aktivas merupakan komponen penting yang selalu mengikuti layer konvolusi. Layer ini dirancang untuk memperkenalkan *non-linearitas* ke dalam model. *Non-linearitas* diperlukan karena operasi yang dilakukan pada layer konvolusi bersifat linear, yaitu hanya melibatkan perkalian elemen-elemen pada kernel dan input, serta penjumlahan hasil perkalian tersebut.

Secara matematis, kombinasi beberapa layer konvolusi yang hanya menjalankan operasi linear pada dasarnya setara dengan satu layer konvolusi besar. Jika hal ini

terjadi, jaringan hanya mampu mempelajari pola yang sederhana dan linear, sehingga kapasitas representasi dari model menjadi terbatas. Sebagai akibatnya, hubungan *non-linear* yang ada antar fitur tidak dapat direpresentasikan dengan baik. Padahal, dalam banyak kasus, hubungan antar fitur pada data dunia nyata cenderung bersifat non-linear (Zoumpourlis *et al.*, 2017).

Oleh karena itu, untuk mengatasi keterbatasan ini, fungsi aktivasi digunakan di antara layer-layer konvolusi. Fungsi aktivasi bertugas melakukan transformasi *non-linear* terhadap output layer konvolusi, sehingga model mampu mempelajari pola yang lebih kompleks dan abstrak. Dengan kata lain, fungsi aktivasi memungkinkan jaringan untuk meningkatkan daya representasi serta kemampuan generalisasi dalam menyelesaikan berbagai tugas, seperti pengenalan pola, klasifikasi, dan segmentasi.

Keberadaan layer aktivasi tidak hanya memperkaya struktur jaringan, tetapi juga memungkinkan model untuk memetakan hubungan yang lebih mendalam antara input dan output, yang merupakan elemen esensial dalam pengembangan jaringan saraf yang efektif.

Beragam fungsi aktivasi tersedia untuk digunakan setelah layer konvolusi, seperti fungsi sigmoid, fungsi hiperbolik, dan lainnya. Namun, fungsi aktivasi *Rectified Linear Unit* (ReLU) adalah yang paling sering digunakan, terutama dalam jaringan saraf modern (Krizhevsky *et al.*, 2012). Penggunaan ReLU secara luas didukung oleh dua sifat utamanya, yaitu *non-linearitas* dan *non-saturasi*.

1. *non-linearitas*

   ReLU merupakan singkatan dari *Rectified Linear Unit*, dan secara matematis didefinisikan sebagai:

$$R(z) = z^{+} = \max(0, z) = \frac{z + |z|}{2} = \begin{cases} z, & \text{jika } z > 0 \\ 0, & \text{jika } z \leq 0 \end{cases} \tag{2.2.2}$$

   Di mana z adalah elemen output dari layer konvolusi sebelumnya. Fungsi ini bekerja dengan mempertahankan nilai positif z, sementara nilai negatif z akan

diubah menjadi nol. Transformasi ini memungkinkan jaringan untuk memperkenalkan *non-linearitas*, yang sangat penting dalam mempelajari pola kompleks pada data.

Sebagai contoh, jika *feature* map yang dihasilkan dari layer konvolusi sebelumnya mengandung nilai-nilai negatif, fungsi ReLU akan menyaring nilai tersebut dengan menetapkannya menjadi nol, sehingga hanya fitur positif yang dipertahankan untuk proses selanjutnya.

2. non-saturasi

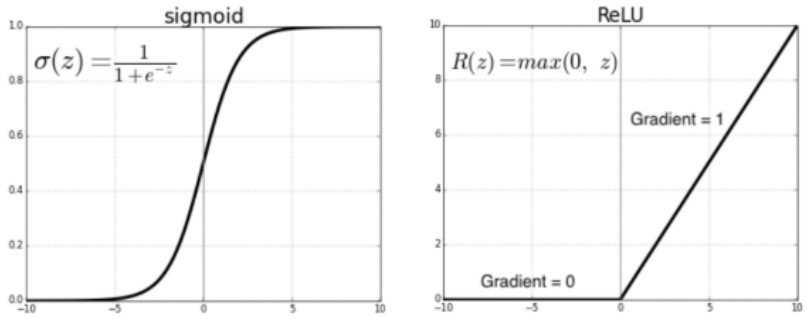
   Saturasi dalam konteks jaringan saraf mengacu pada kondisi di mana nilai output dari suatu fungsi aktivasi berada dalam rentang tetap, seperti yang terjadi pada fungsi *sigmoid* atau tanh. Fungsi yang tidak mengalami *saturasi* (*non-saturating*) memiliki properti sebagai berikut:

$$\lim_{z \to -\infty} f(z) = -\infty \quad \text{atau} \quad \lim_{z \to +\infty} f(z) = +\infty \tag{2.2.3}$$

   ReLU termasuk fungsi *non-saturasi* karena *output* nya tidak dibatasi pada rentang tertentu. Tidak seperti fungsi sigmoid, yang dapat jenuh (*saturasi*) pada nilai masukan yang sangat besar atau sangat kecil, ReLU tidak mengalami *saturasi*, yang berarti gradiennya tetap konstan untuk nilai positif z. Gradien ReLU adalah nol pada sumbu negatif dan satu pada sumbu positif.

Keunggulan ini memberikan manfaat besar dalam proses pembelajaran jaringan saraf. Dengan gradien yang konsisten, pembaruan bobot jaringan pada setiap iterasi akan tetap optimal. Sebaliknya, pada fungsi saturasi seperti sigmoid, gradien dapat menjadi sangat kecil pada nilai input yang besar atau kecil, yang menyebabkan masalah vanishing gradient. Dalam skenario ini, pembaruan bobot menjadi lambat atau bahkan terhenti sepenuhnya, yang dapat menghambat proses pelatihan jaringan.

Perbandingan Fungsi Aktivasi *Saturasi* dan *Non-Saturasi* sebagaimana ditunjukkan pada Gambar 2.3, ReLU menawarkan performa *superior* dibandingkan fungsi sigmoid yang jenuh pada nilai input yang besar. Dengan sifat *non-saturasi*, ReLU memungkinkan jaringan untuk menghindari masalah *vanishing gradient*, sehingga



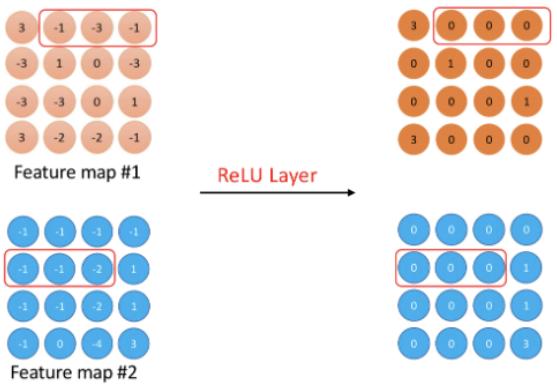
Gambar 2.3 Perbandingan antara fungsi aktivasi saturasi dan non-saturasi
Sumber: (Yang, 2020)

mempercepat konvergensi selama pelatihan.

Proses ReLU pada Layer CNN setelah layer konvolusi menghasilkan *feature map*, fungsi ReLU diterapkan pada setiap elemen dari *feature map* tersebut. Pada *layer* ini:

1. Semua nilai positif tetap dipertahankan.
2. Semua nilai negatif diatur menjadi nol.

Hasil dari fungsi aktivasi ReLU memiliki struktur jaringan yang identik dengan *feature map* dari *layer* konvolusi sebelumnya. Gambar 2.4 menggambarkan bagaimana ReLU bekerja pada *feature map*. *Output* dari *layer* ReLU ini kemudian digunakan sebagai input untuk *layer* konvolusi berikutnya, memperkuat proses ekstraksi fitur yang lebih kompleks pada data.



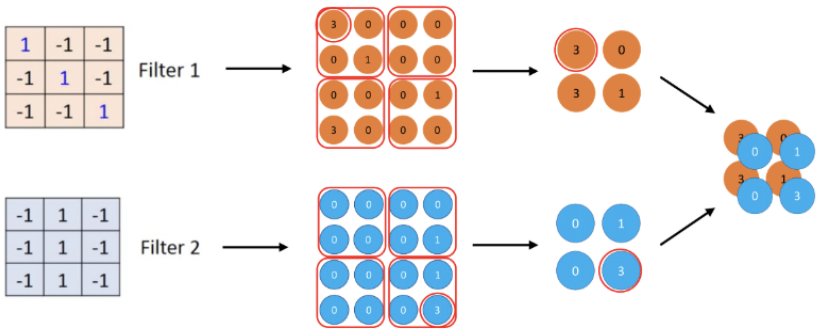
Gambar 2.4 Struktur operasional dasar dari lapisan ReLU
Sumber: (Yang, 2020)

Dengan demikian, fungsi ReLU tidak hanya meningkatkan efisiensi pelatihan jaringan tetapi juga memungkinkan jaringan untuk mempelajari pola yang lebih kaya dan bermakna dari data masukan.

2.2.3 *Pooling Layer*

Layer pooling merupakan komponen penting dalam jaringan *Convolutional Neural Network* (CNN) yang berfungsi untuk mengurangi ukuran spasial (*spatial size*) dari *feature map* yang dihasilkan oleh layer konvolusi sebelumnya, sambil mempertahankan informasi fitur utama. Proses ini berperan dalam mengidentifikasi fitur-fitur penting, mengurangi kompleksitas jaringan, serta membantu mencegah overfitting. Dalam CNN, terdapat beberapa jenis operasi pooling yang umum digunakan, antara lain *average pooling*, L2-*norm pooling*, dan *max pooling*. Dari berbagai jenis pooling tersebut, *max pooling* merupakan metode yang paling sering digunakan dalam berbagai penelitian (Ciresan *et al.*, 2011).

Konsep dasar dari max pooling adalah bahwa lokasi spesifik suatu fitur tidak terlalu penting dibandingkan dengan posisi relatifnya terhadap fitur lain (Scherer *et al.*, 2010). Selain itu, max pooling juga memiliki peran penting dalam mengurangi kemungkinan *overfitting* pada model dengan hanya mengekstraksi fitur utama dari setiap *subregion*. Penggunaan *max pooling* ini memungkinkan jaringan untuk mempertahankan informasi yang paling relevan, sembari mengurangi ukuran data yang diproses dan meningkatkan efisiensi komputasi. Gambar 2.5 memberikan sebuah contoh yang membangun struktur operasional dasar dari *max pooling*.



Gambar 2.5 Struktur operasional dasar dari *Pooling layer*
Sumber: (Yang, 2020)

Secara matematis, *max pooling* dapat dijelaskan sebagai berikut. Misalkan X adalah matriks input berukuran $H \times W$ (tinggi dan lebar), dengan ukuran *kernel pooling* $k \times k$ dan *stride* s. *Output* dari *max pooling* akan menghasilkan matriks Y dengan ukuran $H_{\text{out}} \times W_{\text{out}}$ yang dihitung menggunakan rumus:

$$H_{\text{out}} = \left\lfloor \frac{s_H - k}{1} \right\rfloor + 1, \quad W_{\text{out}} = \left\lfloor \frac{s_W - k}{1} \right\rfloor + 1 \tag{2.2.4}$$

Di mana H dan W adalah dimensi dari matriks input X, k adalah ukuran *kernel pooling*, dan s adalah *stride* yang menentukan pergeseran kernel. Setiap elemen $y_{i,j}$ dalam output Y dihitung sebagai nilai maksimum dari elemen-elemen dalam *sub-matriks* $k \times k$ pada posisi tertentu, yang dinyatakan dengan persamaan:

$$y_{i,j} = \max \{X_{m,n} \mid m \in [i \cdot s, i \cdot s + k), n \in [j \cdot s, j \cdot s + k)\} \tag{2.2.5}$$

Sebagai contoh, jika matriks input X adalah:

$$X = \begin{bmatrix} 1 & 4 & 7 & 3 \\ 3 & 6 & 9 & 5 \\ 2 & 5 & 8 & 4 \\ 1 & 2 & 4 & 2 \end{bmatrix}$$

Dengan *kernel pooling* $k = 2$ dan *stride* $s = 2$, maka output Y akan dihitung dengan mengambil nilai maksimum dari setiap submatriks 2×2, menghasilkan *output* sebagai berikut:

$$Y = \begin{bmatrix} 6 & 9 \\ 5 & 8 \end{bmatrix}$$

Proses *max pooling* ini tidak hanya mengurangi dimensi spasial matriks input, tetapi juga menonjolkan fitur utama dari setiap *subregion*.

Penerapan *max pooling* dalam CNN memiliki beberapa keuntungan, antara lain mengurangi kompleksitas komputasi dengan mengurangi dimensi *output*, sehingga mengurangi jumlah parameter dalam jaringan dan meningkatkan efisiensi komputasi. Selain itu, dengan mengekstraksi hanya fitur-fitur utama dari setiap *subregion*, risiko *overfitting* dapat diminimalkan. Meskipun *max pooling* sering digunakan,

terdapat juga metode *pooling* lain seperti *average pooling* yang sering digunakan dalam model yang berkaitan dengan topik atau tema tertentu, serta *dynamic pooling* yang dapat menyesuaikan jumlah fitur berdasarkan struktur jaringan, memungkinkan ekstraksi informasi semantik yang lebih kompleks dari data *input* (Kalchbrenner *et al.*, 2014).

2.2.4 *Fully-Connected Layer*

Fully-connected layer adalah salah satu komponen penting dalam arsitektur CNN yang bertujuan untuk mengintegrasikan informasi lokal yang diekstraksi oleh layer konvolusi dan *pooling*menjadi representasi global untuk klasifikasi. Pada *layer fully-connected*, setiap neuron terhubung secara penuh dengan semua neuron di layer sebelumnya, memungkinkan jaringan untuk mempelajari kombinasi fitur secara menyeluruh (Goodfellow, 2016) .

Secara matematis, output dari setiap *neuron* y_i dalam layer *fully-connected* didefinisikan sebagai:

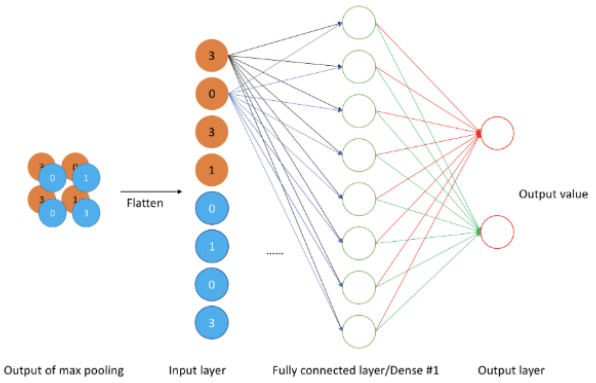
$$y_i = f\left(\sum_{j=1}^{N} w_{ij} \cdot x_j + b_i\right) \tag{2.2.6}$$

di mana x_j adalah input dari *neuron* j pada layer sebelumnya, w_ij adalah bobot koneksi antara *neuron* j dan i, b_i adalah bias *neuron* i, dan f adalah fungsi aktivasi.

Layer fully-connected berfungsi untuk menyusun fitur-fitur hasil ekstraksi dari layer sebelumnya menjadi representasi yang lebih bermakna secara semantik, sehingga dapat digunakan untuk prediksi akhir seperti klasifikasi atau regresi. *Layer fully-connected*, yang ditunjukkan pada Gambar 2.6, dapat mengintegrasikan informasi lokal dengan pembeda kelas di lapisan konvolusi atau lapisan *pooling*. Peran penting layer ini dalam CNN membuatnya menjadi komponen yang tak tergantikan dalam membangun model pembelajaran mendalam yang andal.

Contoh 2.2.1 ronaldo dilaporkan dirawat karena pneumonia

Langkah awal dalam klasifikasi yaitu dengan mengubah data teks menjadi data



Gambar 2.6 Struktur operasional dasar dari *Fully Connected layer*
Sumber: (Yang, 2020)

numerik. kalimat ronaldo dilaporkan dirawat karena pneumonia direpresentasikan dalam bentuk matriks embedding (5 kata × 4 dimensi):

$$X = \begin{bmatrix} 0.2 & -0.1 & 0.5 & 0.3 \\ 0.4 & 0.3 & -0.1 & -0.2 \\ -0.3 & 0.2 & 0.0 & 0.4 \\ 0.1 & -0.3 & 0.4 & 0.2 \\ -0.2 & 0.4 & -0.3 & 0.1 \end{bmatrix}$$

Misalkan filter berukuran 3×4:

$$W = \begin{bmatrix} 0.1 & -0.2 & 0.3 & 0.4 \\ -0.3 & 0.5 & -0.1 & 0.2 \\ 0.2 & 0.1 & -0.4 & -0.3 \end{bmatrix}$$

Jendela pertama (3 kata pertama):

$$X_1 = \begin{bmatrix} 0.2 & -0.1 & 0.5 & 0.3 \\ 0.4 & 0.3 & -0.1 & -0.2 \\ -0.3 & 0.2 & 0.0 & 0.4 \end{bmatrix}$$

Dot product dengan filter W:

$$Z_1 = (X_1 \cdot W) + b$$

Misalkan bias:

$$b = \begin{bmatrix} 0.1 \\ -0.2 \\ 0.3 \end{bmatrix}$$

Maka,

$$Z_1 = \begin{bmatrix} (0.2 \times 0.1) + (-0.1 \times -0.2) + (0.5 \times 0.3) + (0.3 \times 0.4) \\ (0.4 \times -0.3) + (0.3 \times 0.5) + (-0.1 \times -0.1) + (-0.2 \times 0.2) \\ (-0.3 \times 0.2) + (0.2 \times 0.1) + (0.0 \times -0.4) + (0.4 \times -0.3) \end{bmatrix} + \begin{bmatrix} 0.1 \\ -0.2 \\ 0.3 \end{bmatrix}$$

Hasil perhitungan:

$$Z_1 = \begin{bmatrix} 0.41 \\ -0.2 \\ 0.14 \end{bmatrix}$$

Fungsi aktivasi ReLU:

$$f(Z) = \max(0, Z)$$

$$f(Z_1) = \begin{bmatrix} \max(0, 0.41) \\ \max(0, -0.2) \\ \max(0, 0.14) \end{bmatrix}$$

$$= \begin{bmatrix} 0.41 \\ 0 \\ 0.14 \end{bmatrix}$$

Misalkan dilakukan max pooling (2 × 1):

Dari $[0.41, 0, 0.14]$, diambil nilai maksimum:

$$P = \max(0.41, 0, 0.14) = 0.41$$

Misalkan digunakan *weight fully connected layer*:

$$W' = \begin{bmatrix} 0.2 \\ -0.1 \end{bmatrix}$$

Output dari *fully connected layer*:

$$Y = W' \cdot P + b'$$

Misalkan bias $b' = 0.1$:

$$Y = (0.2 \times 0.41) + (-0.1 \times 0) + 0.1$$

$$Y = 0.082 + 0 + 0.1 = 0.182$$

Menggunakan fungsi *sigmoid*:

$$P(\text{positif}) = \frac{1}{1 + e^{-Y}}$$

$$P(\text{positif}) = \frac{1}{1 + e^{-0.182}}$$

$$P(\text{positif}) \approx 0.545$$

Jika threshold 0.5

- Jika $P > 0.5 \rightarrow$ Sentimen Positif (1)
- Jika $P < 0.5 \rightarrow$ Sentimen Negatif (0 atau -1)

Hasil dari klasifikasi ronaldo dikabarkan dirawat karena pneumonia iyalah sentimen positif (1).

Perhitungan *Loss* (Fungsi Biaya) Misalkan label sebenarnya $y = 1$ (sentimen positif). *Loss function* yang digunakan adalah *Binary Cross-Entropy*:

$$L = -\left(y \log(P) + (1 - y) \log(1 - P)\right)$$

Substitusi $P = 0.545$:

$$L = -\log(0.545)$$

$$L \approx 0.606$$

Backpropagation dan *Update* Parameter

Update Parameter *Fully Connected Layer* (W' dan b') *Gradient* dari *loss* terhadap Y:

$$\frac{\partial L}{\partial Y} = P - y = 0.545 - 1 = -0.455$$

Turunan terhadap W':

$$\frac{\partial L}{\partial W'} = \frac{\partial L}{\partial Y} \cdot P$$

$$= -0.455 \times 0.41 = -0.18655$$

Misalkan *learning rate* $\eta = 0.01$:

$$W' = W' - \eta \cdot \frac{\partial L}{\partial W'}$$

$$= W' + 0.0018655$$

Jika awalnya $W' = [0.2, -0.1]$:

$$W' = [0.20187, -0.09814]$$

Untuk bias b':

$$b' = b' - \eta \cdot \frac{\partial L}{\partial b'}$$

$$b' = b' + 0.00455$$

Jika awalnya $b' = 0.1$:

$$b' = 0.10455$$

Update Parameter Konvolusi (W dan b) *Gradient* sebelum ReLU:

$$\frac{\partial L}{\partial Z} = \frac{\partial L}{\partial Y} \times W'$$

$$= -0.455 \times [0.20187, -0.09814]$$

$$= [-0.09185, 0.04462]$$

Misalkan dilakukan update elemen pertama dari filter W:

$$\frac{\partial L}{\partial W_{1,1}} = \frac{\partial L}{\partial Z} \times X$$

$$= -0.09185 \times 0.2 = -0.01837$$

Update filter:

$$W = W - \eta \cdot \frac{\partial L}{\partial W}$$

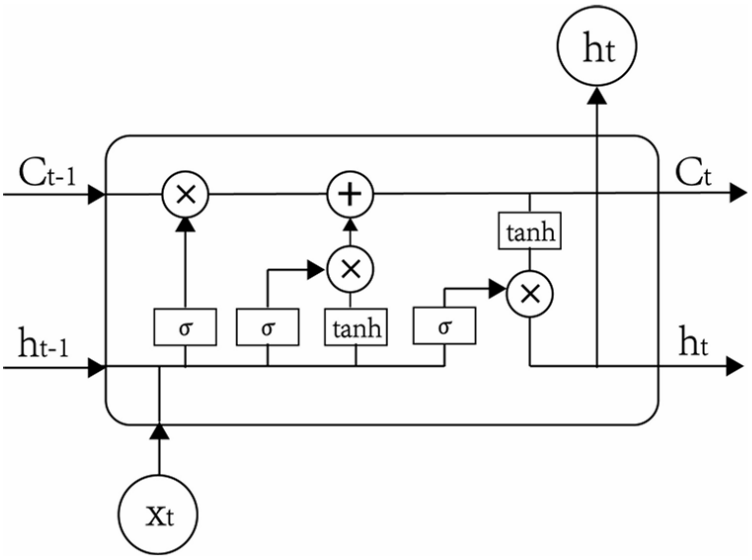
$$W_{1,1} = W_{1,1} + 0.0001837$$

Proses ini diulang sampai model konvergen.

2.3 *Long Short-Term Memory*

Model Long Short-Term Memory (LSTM) merupakan salah satu bentuk jaringan saraf dalam deep learning yang dibangun di atas kerangka RNN, dirancang secara khusus untuk mengatasi masalah ketergantungan jangka panjang pada data sekuensial. Keunggulan ini menjadikan LSTM sangat ideal untuk berbagai aplikasi, termasuk pemrosesan dan prediksi deret waktu (Qiu *et al.*, 2020). Dalam arsitekturnya, LSTM menggantikan fungsi neuron pada lapisan tersembunyi *(hidden layer)* RNN dengan serangkaian sel memori yang terintegrasi.

Seperti yang ditunjukkan Gambar 2.7, sel memori ini terdiri atas beberapa elemen kunci yang bekerja melalui mekanisme gerbang, yaitu input gate, forget gate, dan



Gambar 2.7 Arsitektur LSTM
Sumber: (Qiu *et al.*,2020)

output gate. Mekanisme gerbang ini menggunakan lapisan sigmoid untuk mengendalikan informasi yang diterima, dilupakan, atau diteruskan, sedangkan lapisan $tanh$ berfungsi menormalisasi nilai dalam memori. Fungsi tanh didefinisikan secara matematis sebagai:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{\sinh(x)}{\cosh(x)} \tag{2.3.7}$$

Secara matematis, proses dalam LSTM dapat dirumuskan dengan memberikan masukan deret waktu $x = (x_1, \ldots, x_T)$, di mana model menghitung vektor tersembunyi $h = (h_1, \ldots, h_T)$ dan keluaran $y = (y_1, \ldots, y_T)$ menggunakan iterasi:

$$h_t = \mathcal{H}(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{2.3.8}$$

$$y_t = W_{hy}h_t + b_y \tag{2.3.9}$$

Dalam persamaan tersebut, W adalah matriks bobot, b merupakan vektor bias, dan H adalah fungsi aktivasi tersembunyi. Lebih lanjut, setiap elemen inti LSTM yaitu *input gate* (i_t), *forget gate* (f_t), *output gate* (o_t), dan aktivasi memori (c_t) ditentukan melalui persamaan berikut:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{2.3.10}$$
$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{2.3.11}$$
$$c_t = f_c c_{t_1} + i_t tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{2.3.12}$$
$$0_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{c0}c_t + b_o) \tag{2.3.13}$$
$$h_t = o_t tanh(c_t) \tag{2.3.14}$$

Setiap gerbang dirancang untuk melakukan operasi yang selektif terhadap informasi, dengan σ sebagai fungsi sigmoid yang memetakan nilai antara 0 dan 1. Selain itu, matriks bobot dari memori ke gerbang bersifat diagonal, memastikan bahwa setiap elemen memori hanya memengaruhi elemen yang sesuai dalam gerbang. Hal ini memungkinkan LSTM secara efisien menangkap hubungan temporal kompleks dalam data, sebagaimana dijelaskan oleh (Graves *et al.*, 2013).

Contoh 2.3.1 ronaldo dilaporkan dirawat karena pneumonia

Misal, vektorisasi menggunakan TF-IDF, didapatkan representasi vektor:

$$X_t = [0.1398, 0.1398, 0.1398, 0.1398, 0.0796]$$

Hidden state awal (h_{t-1}) dan *cell state* awal (C_{t-1}) diinisialisasi dengan nol.

Tabel 2.1 Hasil Perhitungan LSTM untuk Setiap *Gate*

Parameter	**Bobot**	**Bias**	**Hasil**
Forget Gate (f_t)	[0.2, 0.3, 0.5, 0.1, 0.4]	0.1	[0.65, 0.70, 0.75, 0.68, 0.66]
Input Gate (i_t)	[0.4, 0.3, 0.2, 0.5, 0.1]	0.2	[0.55, 0.60, 0.65, 0.58, 0.50]
Candidate Cell ($\tilde{C}_t$)	[0.3, 0.4, 0.2, 0.6, 0.1]	0.1	[0.40, 0.42, 0.38, 0.35, 0.39]
Output Gate (o_t)	[0.1, 0.3, 0.4, 0.2, 0.5]	0.1	[0.60, 0.65, 0.70, 0.63, 0.58]

***Update Cell State*:**

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{2.3.15}$$

$$C_t = [0.22, 0.25, 0.28, 0.24, 0.19]$$

***Update Hidden State*:**

$$h_t = o_t \odot \tanh(C_t) \tag{2.3.16}$$

$$h_t = [0.12, 0.15, 0.18, 0.14, 0.11]$$

Misalkan hidden state terakhir dari LSTM:

$$h_t = \begin{bmatrix} 0.12 \\ 0.15 \\ 0.18 \\ 0.14 \\ 0.11 \end{bmatrix}$$

Bobot (W_h) dan bias (b_h) untuk lapisan *fully connected* sebelum *Softmax*:

$$W_h = \begin{bmatrix} 0.3 & 0.2 & 0.5 & 0.4 & 0.1 \\ 0.6 & 0.3 & 0.2 & 0.5 & 0.4 \\ 0.2 & 0.7 & 0.4 & 0.3 & 0.5 \end{bmatrix}$$

$$b_h = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.3 \end{bmatrix}$$

Menghitung z

$$z = W_h \cdot h_t + b_h \tag{2.3.17}$$

Hitung satu per satu:

$$\begin{aligned} z_1 &= (0.3 \times 0.12) + (0.2 \times 0.15) + (0.5 \times 0.18) + (0.4 \times 0.14) + (0.1 \times 0.11) + 0.1 \\ &= 0.036 + 0.03 + 0.09 + 0.056 + 0.011 + 0.1 = 0.323 \end{aligned}$$

$$\begin{aligned} z_2 &= (0.6 \times 0.12) + (0.3 \times 0.15) + (0.2 \times 0.18) + (0.5 \times 0.14) + (0.4 \times 0.11) + 0.2 \\ &= 0.072 + 0.045 + 0.036 + 0.07 + 0.044 + 0.2 = 0.467 \end{aligned}$$

$$\begin{aligned} z_3 &= (0.2 \times 0.12) + (0.7 \times 0.15) + (0.4 \times 0.18) + (0.3 \times 0.14) + (0.5 \times 0.11) + 0.3 \\ &= 0.024 + 0.105 + 0.072 + 0.042 + 0.055 + 0.3 = 0.598 \end{aligned}$$

Sehingga:

$$z = \begin{bmatrix} 0.323 \\ 0.467 \\ 0.598 \end{bmatrix}$$

Menghitung *Probabilitas Softmax*

$$P(y_1) = \frac{e^{0.323}}{e^{0.323} + e^{0.467} + e^{0.598}} = \frac{1.381}{1.381 + 1.595 + 1.818} = \frac{1.381}{4.794} \approx 0.288$$

$$P(y_2) = \frac{e^{0.467}}{e^{0.323} + e^{0.467} + e^{0.598}} = \frac{1.595}{4.794} \approx 0.333$$

$$P(y_3) = \frac{e^{0.598}}{e^{0.323} + e^{0.467} + e^{0.598}} = \frac{1.818}{4.794} \approx 0.379$$

Hasil Akhir Klasifikasi Dari hasil *Softmax*:

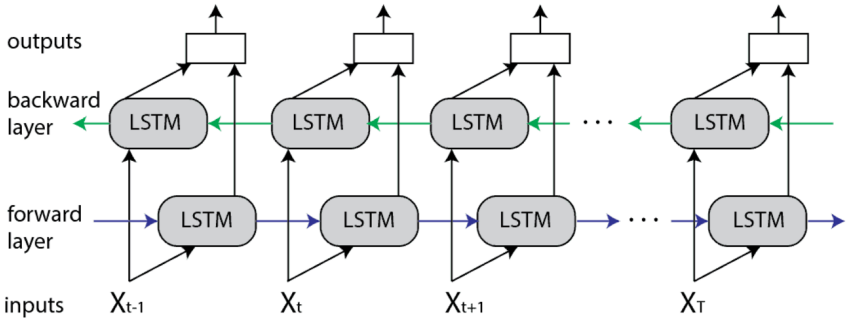
- $P(y_1) = 0.288$ (Sentimen Negatif)
- $P(y_2) = 0.333$ (Sentimen Netral)
- $P(y_3) = 0.379$ (Sentimen Positif)

Karena probabilitas tertinggi adalah pada kelas Positif ($P(y_3) = 0.379$), maka prediksi sentimen akhir adalah positif.

2.4 *Bidirectional Long Short-Term Memory*

Bidirectional Long Short-Term Memory (BiLSTM) merupakan pengembangan dari arsitektur jaringan *Long Short-Term Memory* (LSTM) yang dirancang untuk mengatasi keterbatasan LSTM dalam memproses data sekuensial hanya dalam satu arah.

Dengan memanfaatkan dua lapisan LSTM, BiLSTM memproses data secara *bidirectional*, yaitu menggunakan satu lapisan untuk memproses sekuens secara maju (*forward*) dan lapisan lainnya untuk memproses sekuens secara mundur (*backward*). Arsitektur ini memungkinkan pemanfaatan konteks dari masa lalu dan masa depan secara simultan, sehingga meningkatkan akurasi prediksi dalam tugas pemodelan sekuensial, sebagaimana yang dijelaskan oleh (Schuster & Paliwal, 1997).



Gambar 2.8 Arsitektur BiLSTM
Sumber: (Serghini *et al*.,2023)

Seperti yang ditunjukkan pada Gambar 2.8, arsitektur BiLSTM memanfaatkan keunggulan bidirectional untuk mengakses informasi kontekstual yang lebih lengkap dibandingkan model LSTM standar. Proses matematis yang mendasari BiLSTM tidak berbeda jauh dengan LSTM, namun *Bidirectional* RNN dalam BiLSTM memanfaatkan barisan tersembunyi maju $(\overrightarrow{h})$ dan mundur $(\overleftarrow{h})$, serta menghasilkan output barisan y.

Proses matematis ini dijelaskan melalui iterasi lapisan *backward* dari $t = T$ ke $t = 1$ dan lapisan forward dari $t = 1$ ke $t = T$. Rumus-rumus berikut menggambarkan operasinya:

$$\overrightarrow{h}_t = \mathcal{H}\left(W_{x\overrightarrow{h}} x_t + W_{\overrightarrow{h}\overrightarrow{h}} \overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}\right) \tag{2.4.18}$$

$$\overleftarrow{h}_t = \mathcal{H}\left(W_{x\overleftarrow{h}} x_t + W_{\overleftarrow{h}\overleftarrow{h}} \overleftarrow{h}_{t+1} + b_{\overleftarrow{h}}\right) \tag{2.4.19}$$

$$y_t = W_{\overrightarrow{h}y} \overrightarrow{h}_t + W_{\overleftarrow{h}y} \overleftarrow{h}_t + b_y \tag{2.4.20}$$

Pada persamaan di atas, $\overrightarrow{h}_t$ dan $\overleftarrow{h}_t$ masing-masing merujuk pada status tersembunyi dari arah maju dan mundur dalam proses BiLSTM. H adalah fungsi aktivasi ter-

sembunyi yang dapat berupa fungsi seperti ReLU atau tanh, yang digunakan untuk memproses informasi pada setiap langkah waktu. W melambangkan matriks bobot yang menghubungkan antar lapisan dalam jaringan, sedangkan b adalah vektor bias yang ditambahkan pada setiap arah pemrosesan, baik maju maupun mundur, untuk menyesuaikan output yang dihasilkan dari perhitungan sebelumnya.

Menurut (Graves *et al.*, 2013), penggabungan konsep *Bidirectional* RNN dengan LSTM memberikan kemampuan kepada BiLSTM untuk menangkap konteks jangka panjang dalam kedua arah input, menjadikannya alat yang sangat efektif dalam tugas-tugas seperti pemrosesan bahasa alami dan prediksi deret waktu.

Contoh 2.4.1 ronaldo dilaporkan dirawat karena pneumonia.

Tokenisasi:

['Ronaldo', 'dilaporkan', 'dirawat', 'karena', 'pneumonia']

Word Embedding (misalnya dengan Word2Vec):

$$\begin{bmatrix} 0.2 & 0.3 & 0.5 \\ 0.1 & 0.4 & 0.7 \\ 0.6 & 0.8 & 0.2 \\ 0.3 & 0.9 & 0.5 \\ 0.7 & 0.1 & 0.4 \end{bmatrix}$$

Forward LSTM memproses teks dari awal ke akhir.

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{2.4.21}$$
$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{2.4.22}$$
$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{2.4.23}$$
$$\tilde{C}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{2.4.24}$$
$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{2.4.25}$$
$$\overrightarrow{h_t} = o_t \odot \tanh(C_t) \tag{2.4.26}$$

Hasil *Forward* LSTM:

$$\overrightarrow{h} = \begin{bmatrix} 0.12 \\ 0.15 \\ 0.18 \\ 0.14 \\ 0.11 \end{bmatrix}$$

Backward LSTM memproses teks dari akhir ke awal.

$$f_t^{(b)} = \sigma(W_f^{(b)} x_t + U_f^{(b)} h_{t+1}^{(b)} + b_f^{(b)}) \tag{2.4.27}$$
$$i_t^{(b)} = \sigma(W_i^{(b)} x_t + U_i^{(b)} h_{t+1}^{(b)} + b_i^{(b)}) \tag{2.4.28}$$
$$o_t^{(b)} = \sigma(W_o^{(b)} x_t + U_o^{(b)} h_{t+1}^{(b)} + b_o^{(b)}) \tag{2.4.29}$$
$$\tilde{C_t}^{(b)} = \tanh(W_c^{(b)} x_t + U_c^{(b)} h_{t+1}^{(b)} + b_c^{(b)}) \tag{2.4.30}$$
$$C_t^{(b)} = f_t^{(b)} \odot C_{t+1}^{(b)} + i_t^{(b)} \odot \tilde{C_t}^{(b)} \tag{2.4.31}$$
$$\overleftarrow{h_t} = o_t^{(b)} \odot \tanh(C_t^{(b)}) \tag{2.4.32}$$

Hasil *Backward* LSTM:

$$\overleftarrow{h} = \begin{bmatrix} 0.10 \\ 0.13 \\ 0.16 \\ 0.12 \\ 0.09 \end{bmatrix}$$

Menggabungkan *Forward* dan *Backward* LSTM

$$h_t = \overrightarrow{h_t} \oplus \overleftarrow{h_t} \tag{2.4.33}$$

Hasil *hidden state* BiLSTM:

$$h_t = \begin{bmatrix} 0.12 \\ 0.15 \\ 0.18 \\ 0.14 \\ 0.11 \\ 0.10 \\ 0.13 \\ 0.16 \\ 0.12 \\ 0.09 \end{bmatrix}$$

Fully Connected Layer Hidden state BiLSTM dikalikan dengan bobot dan ditambahkan bias untuk mendapatkan logit z:

$$z = W_h \cdot h_t + b_h \tag{2.4.34}$$

Bobot:

$$W_h = \begin{bmatrix} 0.3 & 0.2 & 0.5 & 0.4 & 0.1 & 0.3 & 0.2 & 0.5 & 0.4 & 0.1 \\ 0.6 & 0.3 & 0.2 & 0.5 & 0.4 & 0.6 & 0.3 & 0.2 & 0.5 & 0.4 \\ 0.2 & 0.7 & 0.4 & 0.3 & 0.5 & 0.2 & 0.7 & 0.4 & 0.3 & 0.5 \end{bmatrix}$$

Bias:

$$b_h = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.3 \end{bmatrix}$$

Hasil:

$$z = \begin{bmatrix} 1.2 \\ 2.3 \\ 0.8 \end{bmatrix}$$

Softmax Activation:

$$P(y_1) = \frac{e^{z_1}}{e^{z_1} + e^{z_2} + e^{z_3}}$$
$$P(y_2) = \frac{e^{z_2}}{e^{z_1} + e^{z_2} + e^{z_3}}$$
$$P(y_3) = \frac{e^{z_3}}{e^{z_1} + e^{z_2} + e^{z_3}}$$

Hasil probabilitas:

- $P(y_1) = 0.25$ (Negatif)
- $P(y_2) = 0.30$ (Netral)
- $P(y_3) = 0.45$ (Positif)

Prediksi akhir kalimat ”ronaldo dikabarkan dirawat karena pneumonia” sentimen positif.

Model akan terus diupdate dengan adanya data yang masuk. untuk mengetahui cara model diupdate maka dimisal dimiliki data variabel input, *Hiden state* sebelumnya, *Cell state* sebelumnya, Bobot dan bias.

Diketahui:

- Input: $x_t = 0.5$
- *Hidden state* sebelumnya: $h_{t-1} = 0.1$
- *Cell state* sebelumnya: $C_{t-1} = 0.4$
- Bobot dan bias:

$$\begin{array}{lll} W_f = 0.8, & U_f = 0.3, & b_f = 0.2 \\ W_i = 0.6, & U_i = 0.5, & b_i = 0.1 \\ W_o = 0.9, & U_o = 0.4, & b_o = 0.3 \\ W_c = 0.7, & U_c = 0.6, & b_c = 0.2 \end{array}$$

Forget Gate:

$$\begin{aligned} f_t &= \sigma((W_f x_t) + (U_f h_{t-1}) + b_f) \\ &= \sigma((0.8 \times 0.5) + (0.3 \times 0.1) + 0.2) \\ &= \sigma(0.4 + 0.03 + 0.2) \\ &= \sigma(0.63) \\ &= \frac{1}{1 + e^{-0.63}} \approx 0.65 \end{aligned}$$

Input Gate:

$$\begin{aligned} i_t &= \sigma((W_i x_t) + (U_i h_{t-1}) + b_i) \\ &= \sigma((0.6 \times 0.5) + (0.5 \times 0.1) + 0.1) \\ &= \sigma(0.3 + 0.05 + 0.1) \\ &= \sigma(0.45) \\ &= \frac{1}{1 + e^{-0.45}} \approx 0.61 \end{aligned}$$

Cell Candidate:

$$\begin{aligned} \tilde{C}_t &= \tanh((W_c x_t) + (U_c h_{t-1}) + b_c) \\ &= \tanh((0.7 \times 0.5) + (0.6 \times 0.1) + 0.2) \\ &= \tanh(0.35 + 0.06 + 0.2) \\ &= \tanh(0.61) \approx 0.54 \end{aligned}$$

Cell State:

$$\begin{aligned} C_t &= (f_t \times C_{t-1}) + (i_t \times \tilde{C}_t) \\ &= (0.65 \times 0.4) + (0.61 \times 0.54) \\ &= 0.26 + 0.33 \\ &= 0.59 \end{aligned}$$

Output Gate:

$$\begin{aligned} o_t &= \sigma((W_o x_t) + (U_o h_{t-1}) + b_o) \\ &= \sigma((0.9 \times 0.5) + (0.4 \times 0.1) + 0.3) \\ &= \sigma(0.45 + 0.04 + 0.3) \\ &= \sigma(0.79) \\ &= \frac{1}{1 + e^{-0.79}} \approx 0.69 \end{aligned}$$

Hidden State:

$$\begin{aligned} h_t &= o_t \times \tanh(C_t) \\ &= 0.69 \times \tanh(0.59) \\ &= 0.69 \times 0.53 \\ &= 0.37 \end{aligned}$$

Dengan perhitungan ini, diperoleh:

- Forget Gate: $f_t \approx 0.65$
- Input Gate: $i_t \approx 0.61$
- Cell Candidate: $\tilde{C}_t \approx 0.54$
- Cell State: $C_t \approx 0.59$
- Output Gate: $o_t \approx 0.69$
- Hidden State: $h_t \approx 0.37$

Perhitungan Gradien untuk *Forget Gate*:

$$\begin{aligned} \frac{\partial L}{\partial W_f} &= \frac{\partial L}{\partial f_t} \cdot \frac{\partial f_t}{\partial W_f} \\ f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\ \sigma'(0.63) &= 0.65 \times (1 - 0.65) = 0.2275 \\ \frac{\partial f_t}{\partial W_f} &= x_t \cdot \sigma'(W_f x_t + U_f h_{t-1} + b_f) = 0.5 \times 0.2275 = 0.11375 \\ \frac{\partial L}{\partial W_f} &= 0.2 \times 0.11375 = 0.02275 \end{aligned}$$

Perhitungan Gradien untuk *Input Gate*:

$$\frac{\partial L}{\partial W_i} = \frac{\partial L}{\partial i_t} \cdot \frac{\partial i_t}{\partial W_i}$$
$$\sigma'(0.45) = 0.61 \times (1 - 0.61) = 0.2379$$
$$\frac{\partial i_t}{\partial W_i} = x_t \cdot \sigma'(W_i x_t + U_i h_{t-1} + b_i) = 0.5 \times 0.2379 = 0.11895$$
$$\frac{\partial L}{\partial W_i} = 0.2 \times 0.11895 = 0.02379$$

Perhitungan Gradien untuk *Cell Candidate*:

$$\frac{\partial L}{\partial W_c} = \frac{\partial L}{\partial \tilde{C}_t} \cdot \frac{\partial \tilde{C}_t}{\partial W_c}$$
$$\tanh'(0.61) = 1 - \tanh^2(0.61) = 1 - 0.3721 = 0.6279$$
$$\frac{\partial \tilde{C}_t}{\partial W_c} = x_t \cdot \tanh'(W_c x_t + U_c h_{t-1} + b_c) = 0.5 \times 0.6279 = 0.31395$$
$$\frac{\partial L}{\partial W_c} = 0.2 \times 0.31395 = 0.06279$$

Perhitungan Gradien untuk *Output Gate*:

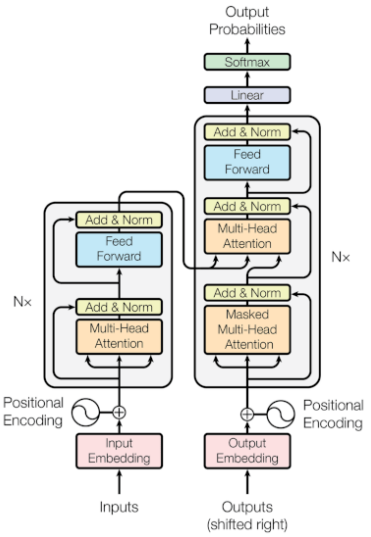
$$\frac{\partial L}{\partial W_o} = \frac{\partial L}{\partial o_t} \cdot \frac{\partial o_t}{\partial W_o}$$
$$\sigma'(0.79) = 0.69 \times (1 - 0.69) = 0.2139$$
$$\frac{\partial o_t}{\partial W_o} = x_t \cdot \sigma'(W_o x_t + U_o h_{t-1} + b_o) = 0.5 \times 0.2139 = 0.10695$$
$$\frac{\partial L}{\partial W_o} = 0.2 \times 0.10695 = 0.02139$$

Proses ini diulangi hingga model konvergen.

2.5 *Bidirectional Encoder Representations from Transformers*

Bidirectional encoder representations from transformers (BERT) adalah model bahasa berbasis transformer yang dikembangkan oleh Google untuk memahami representasi teks secara mendalam. Secara matematis, BERT memanfaatkan pendekatan pengkodean *bidirectional* dalam *transformer*, yang memungkinkan pemahaman konteks dari kedua arah teks (kiri ke kanan dan kanan ke kiri) secara simultan (Ken-

ton & Toutanova, 2019).



Gambar 2.9 Arsitektur BERT
Sumber: (Vaswani,2017)

BERT terdiri dari beberapa kompenen yaitu *representasi embedding, self-ittention, masked languege modeling* (MLM), *objective function*, dan *transformer layer*, arsitektur BERT dapat dilihat pada Gambar 2.9. *Representasi embedding* merupakan komponen yang bertujuan untuk mengubah setiap kata w dalam teks menjadi vektor *embedding* $E(w)$ yang berukuran tetap. Ini mencakup *embedding* posisi, token, dan *segmen*:

$$E(w) = E_{\text{token}}(w) + E_{\text{posisi}} + E_{\text{segmen}} \tag{2.5.35}$$

Mekanisme *self-attention* menghitung relevansi antar kata dalam sebuah urutan menggunakan skalar perhatian ($\alpha_i j$) yang dihitung dari skor perhatian Q, K, V:

$$\alpha_{ij} = \text{softmax}\left(\frac{Q_i \cdot K_j^\top}{\sqrt{d_k}}\right) \tag{2.5.36}$$

Dengan (Q_i) *Query* vektor dari token ke-i, (K_j) *Key vektor* dari token ke-j, (d_k) Dimensi *key/query*.

Representasi output dihitung sebagai:

$$\textit{Attention}(Q, K, V) = \sum_j \alpha_{ij} \cdot V_j \tag{2.5.37}$$

dengan bobot perhatian antara token ke-i dan token ke-j. dan vektor nilai (*value vector*).

BERT dilatih dengan Masked Language Modeling, yaitu menutupi beberapa token (w_mask) dalam input dan memprediksi token tersebut berdasarkan konteksnya. Probabilitas prediksi dihitung dengan:

$$P(w_{\text{mask}} \mid w_1, \ldots, w_n) = \text{softmax}(W \cdot h_{\text{mask}} + b) \tag{2.5.38}$$

Di mana h_{mask} adalah representasi tersembunyi dari *token* yang *dimask*, dan W, b adalah parameter.

Objective Function yaitu fungsi yang digunakan untuk mengukur dan memandu proses pembelajarandengan memengaruhi fungsi *loss* dalam MLM dan NSP. dengan Fungsi kehilangan (L) pada BERT adalah kombinasi dari kehilangan MLM (L_{MLM}) dan tugas prediksi hubungan antar kalimat (L_{NSP}):

$$L = L_{\text{MLM}} + L_{\text{NSP}} \tag{2.5.39}$$

BERT terdiri dari beberapa lapisan *transformer*, di mana setiap lapisan menggunakan *self-attention* dan *feed-forward neural network*:

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2.5.40}$$

$$\text{Output}_l = \text{FFN}(\text{Attention}(\text{Input}_l)) + \text{Input}_l \tag{2.5.41}$$

BERT mendasarkan diri pada optimalisasi parameter (θ) model untuk memaksimalkan probabilitas kondisi pada data:

$$\theta^* = \arg\max_\theta \prod_{i=1}^{N} P(w_i \mid \text{Context}) \tag{2.5.42}$$

Di mana Context meliputi informasi dari kedua arah teks. Pendekatan matematis ini memungkinkan BERT memahami hubungan semantik dan sintaksis yang kompleks dalam teks, menghasilkan model representasi yang canggih untuk berbagai tugas NLP.

Robustly Optimized Bidirectional Encoder Representations from Transformers Approach (RoBERTa) merupakan pengembangan yang signifikan dari arsitektur BERT yang bertujuan untuk meningkatkan performa pretraining serta hasil evaluasi tugas akhir. RoBERTa memperkenalkan serangkaian modifikasi penting, antara lain penggunaan masking dinamis, pelatihan dengan kalimat utuh tanpa NSP (*Next Sentence Prediction) loss, mini-batch* yang lebih besar, dan penerapan BPE (*Byte Pair Encoding*) tingkat byte yang lebih besar. Pendekatan ini memungkinkan RoBERTa untuk mengatasi keterbatasan BERT dalam memanfaatkan konteks data secara lebih luas (Liu, 2019).

Selain itu, RoBERTa juga mempertimbangkan dua faktor yang sering diabaikan dalam penelitian sebelumnya: (1) data yang digunakan untuk pretraining, dan (2) jumlah iterasi yang diperlukan untuk memproses data. RoBERTa dilatih pada dataset yang jauh lebih besar, termasuk gabungan dari BOOKCORPUS dan WIKIPEDIA, dibandingkan dengan BERTLARGE dan XLNetLARGE. Ini memungkinkan RoBERTa untuk mengakses lebih banyak data dan meningkatkan kemampuannya dalam memahami teks.

Contoh 2.5.1 ronaldo dilaporkan dirawat karena pneumonia

RoBERTa menggunakan tokenisasi *Byte-Pair Encoding* (BPE). Contoh tokenisasi:

['Ro', 'nal', 'do', ' di', 'lapor', 'kan', ' di', 'rawat', ' karena', ' pneu', 'monia']

Token ID:

[0, 1002, 4523, 2842, 250, 9842, 621, 250, 9524, 421, 7854, 9652, 2]

RoBERTa mengubah token ID menjadi representasi vektor:

$$E = W_E \cdot x$$

Contoh embedding untuk tiap token:

$$E = \begin{bmatrix} 0.15 & 0.23 & 0.51 \\ 0.18 & 0.32 & 0.45 \\ 0.22 & 0.40 & 0.60 \\ \vdots & \vdots & \vdots \end{bmatrix}$$

Transformer Encoder (Self-Attention) Self-attention score dihitung sebagai:

$$A = \textit{Softmax}\left(\frac{QK^T}{d_k}\right)$$

Dimana:

$$Q = W_Q \cdot E, \quad K = W_K \cdot E, \quad V = W_V \cdot E$$

dengan d_k adalah dimensi dari embedding.

Contoh hasil *attention* score:

$$A = \begin{bmatrix} 0.12 & 0.31 & 0.23 \\ 0.21 & 0.11 & 0.32 \\ 0.19 & 0.16 & 0.15 \end{bmatrix}$$

Kemudian, nilai keluaran dihitung:

$$Z = A \cdot V$$

Contoh *output* dari *self-attention*:

$$Z = \begin{bmatrix} 0.35 & 0.28 & 0.33 \\ 0.40 & 0.30 & 0.31 \\ 0.45 & 0.38 & 0.42 \end{bmatrix}$$

Feedforward Network (FFN) Setelah melewati *multi-head attention*, hasilnya diproses dengan FFN:

$$FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

Contoh *output* FFN:

$$\begin{bmatrix} 0.42 & 0.35 & 0.39 \\ 0.38 & 0.40 & 0.42 \\ 0.50 & 0.45 & 0.47 \end{bmatrix}$$

Klasifikasi dengan *Softmax* Vektor [CLS] dari *output* terakhir digunakan untuk klasifikasi sentimen:

$$y = \text{Softmax}(W_h \cdot h_{[CLS]} + b_h)$$

Bobot dan bias:

$$W_h = \begin{bmatrix} 0.3 & 0.6 & 0.2 \\ 0.2 & 0.3 & 0.7 \\ 0.5 & 0.2 & 0.4 \end{bmatrix}, \quad b_h = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.3 \end{bmatrix}$$

Hasil *logit*:

$$z = \begin{bmatrix} 1.8 \\ 2.4 \\ 0.9 \end{bmatrix}$$

Softmax probability:

$$P(y_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

Dengan hasil:

$$P(y_1) = 0.30, \quad P(y_2) = 0.50, \quad P(y_3) = 0.20$$

Prediksi sentimen netral karena $P(y_2)$ tertinggi.

Hasil Akhir:

$$P(y_1) = 0.30(Negatif)$$

$P(y_2) = 0.50$ (Netral)

$P(y_3) = 0.20$ (Positif)

Prediksi kalimat ”ronaldo dilaporkan dirawat karena pneumonia” adalah sentimen

netral.

2.6 *Word2Vec*

Word2Vec adalah model pembelajaran mesin yang dikembangkan oleh (Mikolov *et al.*, 2013) untuk merepresentasikan kata dalam bentuk vektor dalam ruang berdimensi tinggi. Model ini memungkinkan kata-kata dengan makna serupa memiliki representasi vektor yang berdekatan. *Word2Vec* memiliki arsitektur utama, yaitu *Continuous Bag of Words* (CBOW).

Dalam pendekatan CBOW, model berusaha memprediksi kata target berdasarkan kata-kata di sekitarnya dalam suatu jendela konteks. Probabilitas kata target diberikan oleh:

$$P(w_t|w_{t-m},\ldots,w_{t-1},w_{t+1},\ldots,w_{t+m}) = \frac{\exp(v_{w_t}\cdot v_c)}{\sum_{w\in V}\exp(v_w\cdot v_c)} \tag{2.6.43}$$

Dimana v_{w_t} adalah vektor kata target, v_s adalah rata-rata vektor kata-kata dalam konteks, dan V adalah ukuran kosakata, Fungsi loss yang digunakan dalam CBOW adalah:

$$J = -\sum_t \log P(w_t|\text{context}) \tag{2.6.44}$$

Salah satu tantangan utama dalam pelatihan *Word2Vec* adalah komputasi *softmax* penuh yang mahal secara komputasi ($\mathcal{O}(V)$). Untuk mengatasi hal ini, (Mikolov *et al.*, 2013) mengusulkan *Negative Sampling*, yang menggantikan perhitungan *softmax* dengan pendekatan probabilitas *biner*:

$$J = \log\sigma(v_{w_t}\cdot v_c) + \sum_{i=1}^{k}\mathbb{E}_{w_{\text{neg}}\sim P(w)}\log\sigma(-v_{w_{\text{neg}}}\cdot v_c) \tag{2.6.45}$$

dengan $\sigma(x) = \frac{1}{1+e^{-x}}$ sebagai fungsi aktivasi *sigmoid*, dan w_{neg} adalah sampel kata negatif dari distribusi probabilitas $P(w)$.

CBOW lebih unggul dalam analisis sentimen karena kemampuannya menangkap informasi dari beberapa kata dalam suatu kalimat dengan lebih stabil. Menurut (Jiang *et al.*, 2020), CBOW lebih efektif dibandingkan model lain dalam menangkap pola semantik dalam kalimat yang pendek. Selain itu, penelitian (Dos Santos & Maira Gatti, 2014) menunjukkan bahwa CBOW lebih baik dalam menangani data sentimen yang memiliki konteks luas karena memperhitungkan keseluruhan konteks dalam satu representasi vektor rata-rata.

Contoh 2.6.1 ronaldo dilaporkan dirawat karena pneumonia

Langkah 1, representasikan kata dalam vektor. Kata ronaldo, dilaporkan, dirawat, karena, dan pneumonia akan direpresentasikan dalam vektor dan membentuk matrix. Matriks embedding adalah W dengan dimensi $V \times D = 10 \times 10$ (V adalah jumlah kata unik, D adalah dimensi vektor).

$$W = \begin{bmatrix} 0.2 & -0.1 & 0.5 & 0.3 & -0.2 & 0.1 & -0.4 & 0.2 & 0.0 & 0.3 \\ 0.4 & 0.3 & -0.1 & -0.2 & 0.5 & -0.3 & 0.1 & -0.1 & 0.4 & -0.2 \\ -0.3 & 0.2 & 0.0 & 0.4 & -0.1 & 0.3 & -0.2 & 0.5 & -0.4 & 0.1 \\ 0.1 & -0.3 & 0.4 & 0.2 & -0.5 & 0.1 & 0.3 & -0.2 & 0.1 & -0.4 \\ -0.2 & 0.4 & -0.3 & 0.1 & 0.2 & -0.5 & 0.4 & -0.1 & 0.3 & 0.5 \end{bmatrix}$$

Langkah 2, Hitung rata-rata embedding konteks. Misal diambil embedding ronaldo dan dirawat, untuk memprediksi kata dilaporkan.

$$v_{\text{context}} = \frac{1}{2}(W_{\text{ronaldo}} + W_{\text{dirawat}})$$

$$v_{\text{context}} = \frac{1}{2} \begin{bmatrix} (0.2 + (-0.3)) \\ (-0.1 + 0.2) \\ (0.5 + 0.0) \\ (0.3 + 0.4) \\ (-0.2 + (-0.1)) \\ (0.1 + 0.3) \\ (-0.4 + (-0.2)) \\ (0.2 + 0.5) \\ (0.0 + (-0.4)) \\ (0.3 + 0.1) \end{bmatrix}$$

$$v_{\text{context}} = \frac{1}{2} \begin{bmatrix} -0.1 \\ 0.1 \\ 0.5 \\ 0.7 \\ -0.3 \\ 0.4 \\ -0.6 \\ 0.7 \\ -0.4 \\ 0.4 \end{bmatrix} = \begin{bmatrix} -0.05 \\ 0.05 \\ 0.25 \\ 0.35 \\ -0.15 \\ 0.2 \\ -0.3 \\ 0.35 \\ -0.2 \\ 0.2 \end{bmatrix}$$

misal W' memiliki dimensi 5×10

$$W' = \begin{bmatrix} 0.2 & -0.1 & 0.5 & 0.3 & -0.2 & 0.1 & -0.4 & 0.2 & 0.0 & 0.3 \\ 0.4 & 0.3 & -0.1 & -0.2 & 0.5 & -0.3 & 0.1 & -0.1 & 0.4 & -0.2 \\ -0.3 & 0.2 & 0.0 & 0.4 & -0.1 & 0.3 & -0.2 & 0.5 & -0.4 & 0.1 \\ 0.1 & -0.3 & 0.4 & 0.2 & -0.5 & 0.1 & 0.3 & -0.2 & 0.1 & -0.4 \\ -0.2 & 0.4 & -0.3 & 0.1 & 0.2 & -0.5 & 0.4 & -0.1 & 0.3 & 0.5 \end{bmatrix}$$

$$z = W' \cdot v_{\text{context}}$$

$$z = \begin{bmatrix} 0.2 & -0.1 & 0.5 & 0.3 & -0.2 & 0.1 & -0.4 & 0.2 & 0.0 & 0.3 \\ 0.4 & 0.3 & -0.1 & -0.2 & 0.5 & -0.3 & 0.1 & -0.1 & 0.4 & -0.2 \\ -0.3 & 0.2 & 0.0 & 0.4 & -0.1 & 0.3 & -0.2 & 0.5 & -0.4 & 0.1 \\ 0.1 & -0.3 & 0.4 & 0.2 & -0.5 & 0.1 & 0.3 & -0.2 & 0.1 & -0.4 \\ -0.2 & 0.4 & -0.3 & 0.1 & 0.2 & -0.5 & 0.4 & -0.1 & 0.3 & 0.5 \end{bmatrix} \cdot \begin{bmatrix} -0.05 \\ 0.05 \\ 0.25 \\ 0.35 \\ -0.15 \\ 0.2 \\ -0.3 \\ 0.35 \\ -0.2 \\ 0.2 \end{bmatrix}$$

$$z = \begin{bmatrix} 0.515 \\ -0.42 \\ 0.575 \\ -0.015 \\ -0.255 \end{bmatrix}$$

Probabilitas untuk semua kata:

$$P(w_j) = \frac{e^{z_j}}{\sum_{k=1}^{5} e^{z_k}}$$

$$e^{0.515} \approx 1.673$$
$$e^{-0.42} \approx 0.657$$
$$e^{0.575} \approx 1.777$$
$$e^{-0.015} \approx 0.985$$
$$e^{-0.255} \approx 0.775$$

$$\sum_{k=1}^{5} e^{z_k} = 1.673 + 0.657 + 1.777 + 0.985 + 0.775 = 5.867$$

Hasil probabilitas:

$$P(w_1) = \frac{1.673}{5.867} \approx 0.285$$
$$P(w_2) = \frac{0.657}{5.867} \approx 0.112$$
$$P(w_3) = \frac{1.777}{5.867} \approx 0.303$$
$$P(w_4) = \frac{0.985}{5.867} \approx 0.168$$
$$P(w_5) = \frac{0.775}{5.867} \approx 0.132$$

$$P = \begin{bmatrix} 0.285 \\ 0.112 \\ 0.303 \\ 0.168 \\ 0.132 \end{bmatrix}$$

$$L = -\log P(w_{\text{target}})$$

Diketahui bahwa kata target berada di indeks ke-2:

$$P(w_{\text{target}}) = P(w_3) = 0.303$$

Maka,

$$L = -\log(0.303)$$

Menggunakan logaritma natural:

$$L \approx -(-1.194) = 1.194$$

$$\frac{\partial L}{\partial z} = P - y$$

Diketahui bahwa *one-hot encoding* dari kata target adalah:

$$y = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Sehingga:

$$\frac{\partial L}{\partial z} = \begin{bmatrix} 0.285 \\ 0.112 \\ 0.303 \\ 0.168 \\ 0.132 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.285 \\ 0.112 \\ -0.697 \\ 0.168 \\ 0.132 \end{bmatrix}$$

Gradien terhadap bobot:

$$\frac{\partial L}{\partial W'} = \frac{\partial L}{\partial z} \cdot v_{\text{context}}^{T}$$

Dengan:

$$v_{\text{context}} = \begin{bmatrix} -0.05 \\ 0.05 \\ 0.25 \\ 0.35 \\ -0.15 \\ 0.2 \\ -0.3 \\ 0.35 \\ -0.2 \\ 0.2 \end{bmatrix}$$

Bobot diperbarui dengan *learning rate* $\eta = 0.001$:

$$W'_{\text{new}} = W'_{\text{old}} - \eta \cdot \frac{\partial L}{\partial W'}$$

$$W'^{T}_{\text{new}} = \begin{bmatrix} -1.0450\text{e}{-5} & -9.4000\text{e}{-6} & 3.4850\text{e}{-5} & -8.7500\text{e}{-6} & -6.2500\text{e}{-6} \\ 1.0450\text{e}{-5} & 9.4000\text{e}{-6} & -3.4850\text{e}{-5} & 8.7500\text{e}{-6} & 6.2500\text{e}{-6} \\ 5.2250\text{e}{-5} & 4.7000\text{e}{-5} & -1.7425\text{e}{-4} & 4.3750\text{e}{-5} & 3.1250\text{e}{-5} \\ 7.3150\text{e}{-5} & 6.5800\text{e}{-5} & -2.4395\text{e}{-4} & 6.1250\text{e}{-5} & 4.3750\text{e}{-5} \\ -3.1350\text{e}{-5} & -2.8200\text{e}{-5} & 1.0455\text{e}{-4} & -2.6250\text{e}{-5} & -1.8750\text{e}{-5} \\ 4.1800\text{e}{-5} & 3.7600\text{e}{-5} & -1.3940\text{e}{-4} & 3.5000\text{e}{-5} & 2.5000\text{e}{-5} \\ -6.2700\text{e}{-5} & -5.6400\text{e}{-5} & 2.0910\text{e}{-4} & -5.2500\text{e}{-5} & -3.7500\text{e}{-5} \\ 7.3150\text{e}{-5} & 6.5800\text{e}{-5} & -2.4395\text{e}{-4} & 6.1250\text{e}{-5} & 4.3750\text{e}{-5} \\ -4.1800\text{e}{-5} & -3.7600\text{e}{-5} & 1.3940\text{e}{-4} & -3.5000\text{e}{-5} & -2.5000\text{e}{-5} \\ 4.1800\text{e}{-5} & 3.7600\text{e}{-5} & -1.3940\text{e}{-4} & 3.5000\text{e}{-5} & 2.5000\text{e}{-5} \end{bmatrix}$$

Langkah pembaruan W:

$$\frac{\partial L}{\partial v_{\text{context}}} = \begin{bmatrix} -0.0325 \\ 0.0175 \\ -0.0625 \\ 0.0925 \\ -0.0150 \end{bmatrix}$$

$$\frac{\partial L}{\partial W} = \frac{1}{2} \frac{\partial L}{\partial v_{\text{context}}}$$

$$\frac{\partial L}{\partial W} = \begin{bmatrix} -0.01625 \\ 0.00875 \\ -0.03125 \\ 0.04625 \\ -0.0075 \end{bmatrix}$$

Pembaruan matriks W:

$$W_{\text{new}} = W_{\text{old}} - (0.001 \times \frac{\partial L}{\partial W})$$

$$W_{\text{new}} = \begin{bmatrix} w_1 - (0.001 \times (-0.01625)) \\ w_2 - (0.001 \times 0.00875) \\ w_3 - (0.001 \times (-0.03125)) \\ w_4 - (0.001 \times 0.04625) \\ w_5 - (0.001 \times (-0.0075)) \end{bmatrix}$$

$$W_{\text{new}} = \begin{bmatrix} w_1 + 0.00001625 \\ w_2 - 0.00000875 \\ w_3 + 0.00003125 \\ w_4 - 0.00004625 \\ w_5 + 0.0000075 \end{bmatrix}$$

$$W = \begin{bmatrix} -0.05 \\ 0.05 \\ 0.25 \\ 0.35 \\ -0.15 \end{bmatrix}$$

Setelah diperbarui:

$$W_{\text{new}} = \begin{bmatrix} -0.05 + 0.00001625 \\ 0.05 - 0.00000875 \\ 0.25 + 0.00003125 \\ 0.35 - 0.00004625 \\ -0.15 + 0.0000075 \end{bmatrix}$$

$$W_{\text{new}} = \begin{bmatrix} -0.04998375 \\ 0.04999125 \\ 0.25003125 \\ 0.34995375 \\ -0.1499925 \end{bmatrix}$$

Bobot matrix W intput akan terus diperbarui, sama halnya dengan bobot matrix W'. Model akan melakukan perbaruan bobot matrix secara berulang di semua konteks hubungan antar kata.

2.7 TF-IDF

TF-IDF (*Term Frequency - Inverse Document Frequency*) adalah metode numerik yang digunakan dalam pemrosesan teks untuk mengevaluasi seberapa penting suatu kata dalam sebuah dokumen relatif terhadap kumpulan dokumen (corpus). TF-IDF banyak digunakan dalam pencarian informasi dan pengklasifikasian teks.

TF mengukur seberapa sering sebuah kata muncul dalam sebuah dokumen. Rumus-

nya diberikan oleh:

$$TF(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \tag{2.7.46}$$

di mana:

- $f_{t,d}$ adalah jumlah kemunculan kata t dalam dokumen d,
- $\sum_{t' \in d} f_{t',d}$ adalah total jumlah kata dalam dokumen d.

IDF mengukur seberapa jarang suatu kata muncul di seluruh dokumen dalam korpus. Rumusnya adalah:

$$IDF(t, D) = \log \frac{|D|}{|d \in D : t \in d|} \tag{2.7.47}$$

di mana:

- $|D|$ adalah jumlah total dokumen dalam korpus,
- $|d \in D : t \in d|$ adalah jumlah dokumen yang mengandung kata t.

Nilai TF-IDF dihitung dengan mengalikan nilai TF dan IDF:

$$TF - IDF(t, d, D) = TF(t, d) \times IDF(t, D) \tag{2.7.48}$$

TF-IDF memiliki beberapa keunggulan dalam analisis teks, antara lain:

- Memberikan bobot lebih tinggi pada kata-kata penting yang jarang muncul di dokumen lain.
- Mengurangi pengaruh kata-kata umum (*stopwords*) yang sering muncul dalam seluruh dokumen.
- Efektif digunakan dalam pencarian informasi dan ekstraksi fitur dalam model *machine learning* berbasis teks.

Contoh 2.7.1 Misalkan kita memiliki 5 dokumen dengan tokenisasi sebagai berikut:

1. **D1**: ronaldo dilaporkan dirawat karena pneumonia
2. **D2**: hobi merokok seperti pelatih juventus risiko pneumonia 3 kali lebih tinggi
3. **D3**: sentimen wabah virus corona bisa tekan gerak rupiah hari ini
4. **D4**: dki tidak keluarkan izin keramaian akibat corona
5. **D5**: kemenkes ada 50 orang di acara dansa yang dihadiri pasien positif corona

Dari kelima dokumen ini, kita membentuk sebuah *vocabulary* yang berisi semua kata unik.

Perhitungan *Term Frequency* (TF) *Term Frequency* (*TF*) dihitung dengan rumus:

$$TF(t,d) = \frac{\text{jumlah kemunculan kata } t \text{ dalam dokumen } d}{\text{total kata dalam dokumen } d}$$

Sebagai contoh, untuk dokumen pertama (**D1**):

Tabel 2.2 *Term Frequency* untuk Dokumen 1 (D1)

Kata	**Frekuensi**	**TF**
ronaldo	1	$\frac{1}{5} = 0.2$
dilaporkan	1	$\frac{1}{5} = 0.2$
dirawat	1	$\frac{1}{5} = 0.2$
karena	1	$\frac{1}{5} = 0.2$
pneumonia	1	$\frac{1}{5} = 0.2$

Perhitungan *Inverse Document Frequency* (IDF) *Inverse Document Frequency* (*IDF*) dihitung dengan rumus:

$$IDF(t) = \log \frac{N}{DF(t)}$$

di mana:

- N adalah jumlah total dokumen ($N = 5$).
- $DF(t)$ adalah jumlah dokumen yang mengandung kata t.

Sebagai contoh, untuk kata ”pneumonia”:

$$IDF(”pneumonia”) = \log \frac{5}{2} = \log 2.5 \approx 0.398$$

Dan untuk kata ”corona” yang muncul di 3 dokumen:

$$IDF(”corona”) = \log \frac{5}{3} = \log 1.67 \approx 0.222$$

Perhitungan TF-IDF Nilai TF-IDF dihitung dengan rumus:

$$TFIDF(t, d) = TF(t, d) \times IDF(t)$$

Sebagai contoh, untuk kata ”pneumonia” di dokumen **D1**:

$$TFIDF(”pneumonia”, D1) = 0.2 \times 0.398 = 0.0796$$

Dan untuk kata ”ronaldo” di **D1**:

$$TFIDF(”ronaldo”, D1) = 0.2 \times 0.699 = 0.1398$$

Matriks TF-IDF Setelah semua kata dihitung, kita dapat menyusun matriks TF-IDF sebagai berikut:

Tabel 2.3 Matriks TF-IDF

Kata	**D1**	**D2**	**D3**	**D4**	**D5**
ronaldo	0.1398	0	0	0	0
pneumonia	0.0796	0.0796	0	0	0
corona	0	0	0.0444	0.0444	0.0444
virus	0	0	0.222	0	0

2.8 Optimalisasi Performa Model

Optimalisasi performa model merupakan langkah penting dalam pengembangan sistem berbasis kecerdasan buatan, khususnya model berbasis neural network. Optimalisasi ini bertujuan untuk meningkatkan akurasi, efisiensi komputasi, dan kemampuan generalisasi model. Beberapa pendekatan yang digunakan meliputi regularisasi, optimasi *hyperparameter*, augmentasi data, penggunaan algoritma optimi-

sasi lanjutan, dan penerapan model *hybrid*. Regularisasi, seperti L1 dan L2, digunakan untuk mengurangi *overfitting* dengan mengontrol nilai bobot model, sementara dropout membantu mengurangi ketergantungan antar neuron selama pelatihan (Zhang *et al.*, 2021). Selain itu, optimasi *hyperparameter*, seperti *grid search, random search*, dan *Bayesian optimization*, sangat penting untuk menemukan kombinasi parameter terbaik (Snoek *et al.*, 2012). Di sisi lain, augmentasi data digunakan untuk memperbanyak variasi data pelatihan melalui teknik seperti rotasi, *flipping*, atau penambahan *noise*, yang terbukti meningkatkan generalisasi model (Shorten & Khoshgoftaar, 2019). Penggunaan algoritma optimisasi lanjutan, seperti Adam yang menggabungkan keunggulan momentum dan RMSProp (Kingma, 2014), serta teknik *learning rate scheduling* (Loshchilov & Hutter, 2016), juga berkontribusi besar terhadap konvergensi model yang lebih cepat.

Salah satu pendekatan yang semakin populer adalah penerapan model *hybrid*, yang menggabungkan dua atau lebih arsitektur model untuk memanfaatkan keunggulan masing-masing. *Hybrid* model sering dipilih karena kemampuannya untuk memanfaatkan kekuatan spesifik dari setiap arsitektur dan mengatasi kekurangan model tunggal. Misalnya, CNN sangat baik dalam ekstraksi fitur lokal, sementara LSTM unggul dalam menangkap hubungan temporal dalam data berurutan (Zhou *et al.*, 2022). Kombinasi ini memungkinkan pengolahan data yang lebih mendalam dan kompleks. Kelebihan dari *hybrid models* mencakup fleksibilitas dalam menangani berbagai jenis data, kemampuan generalisasi yang lebih baik, serta hasil yang lebih stabil pada data validasi (Zhou *et al.*, 2022). Namun, hybrid model juga memiliki kelemahan, seperti kompleksitas komputasi yang lebih tinggi dan tantangan dalam desain serta integrasi arsitektur. Meskipun demikian, kelebihannya seringkali lebih dominan, terutama dalam menangani data yang kompleks seperti teks atau sinyal berurutan (Wang *et al.*, 2022).

Pendekatan dalam hibridasi dapat dibagi menjadi tiga jenis utama, yaitu hibridasi serial, hibridasi paralel, dan hibridasi ensemble. Pada hibridasi serial, model pertama memproses data, dan outputnya digunakan sebagai input untuk model kedua. Sebagai contoh, CNN dapat digunakan untuk ekstraksi fitur, kemudian LSTM memahami hubungan temporal berdasarkan fitur yang dihasilkan oleh CNN (Huang *et al.*, 2022). Sebaliknya, hibridasi paralel melibatkan dua atau lebih model yang bekerja secara bersamaan pada data yang sama, dengan hasil akhirnya digabungkan (Abdullah *et al.*, 2016). Hibridasi ensemble, di sisi lain, menggunakan beberapa model untuk menghasilkan prediksi akhir dengan teknik seperti *voting* atau *avera-*

ging (Dietterich, 2000.

Hibridasi serial dipilih dalam penelitian ini karena berbagai kelebihan yang dimilikinya. Data diproses secara bertahap sehingga setiap model fokus pada tugas spesifiknya. Model pertama, seperti CNN, menangani fitur lokal, sementara model kedua, seperti LSTM, menangkap konteks global. Pendekatan ini juga membantu model menangani data kompleks dengan lebih baik (Zhou *et al.*, 2022). Selain itu, hibridasi serial memungkinkan fokus hierarkis dalam pemrosesan, di mana setiap tahap memiliki fungsi spesifik yang saling melengkapi. Meskipun hibridasi serial memiliki kelemahan, seperti latensi akibat waktu pemrosesan yang lebih lama, kelebihannya dalam efisiensi pemrosesan dan kemampuan generalisasi menjadikannya pilihan yang unggul dalam banyak aplikasi (Wang *et al.*, 2022).

Sebagai implementasi spesifik pada data NLP, CNN digunakan untuk mengekstraksi pola lokal seperti *n-grams* dalam teks, sedangkan LSTM digunakan untuk menangkap hubungan temporal antara pola-pola tersebut. Contoh penerapannya mencakup analisis sentimen, di mana CNN menangkap fitur penting dari teks seperti kata-kata utama, dan LSTM memahami hubungan antara fitur tersebut dalam urutan kalimat (Kim & Jeong, 2019). Dalam pengenalan suara, CNN dapat digunakan untuk menangkap fitur akustik, sedangkan LSTM mengolah informasi temporal dari sinyal suara (Amodei *et al.*, 2016). Dengan pendekatan hybrid serial seperti CNN-LSTM, model mampu mencapai performa optimal dalam menangani data yang memiliki karakteristik kompleks. Pendekatan ini telah terbukti meningkatkan akurasi dan generalisasi dibandingkan metode konvensional, seperti yang dilaporkan oleh (Liang *et al.*, 2012).

2.9 Metrik Evaluasi

Dalam penelitian ini, kinerja model evaluasi diukur menggunakan beberapa metrik utama, yaitu *Accuracy, Precision, Recall, F1-Score*, dan ROC-AUC. Penjelasan dari masing-masing metrik tersebut adalah sebagai berikut:

2.9.1 *Accuracy*

Akurasi mengukur proporsi prediksi yang benar terhadap total jumlah prediksi yang dilakukan oleh model. Akurasi memberikan gambaran umum tentang kemampuan model dalam mengklasifikasikan data dengan benar (Powers, 2020). Rumus akurasi adalah:

$$Accuracy = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{2.9.49}$$

Di mana:

- TP: *True Positive* (prediksi positif yang benar)
- TN: *True Negative* (prediksi negatif yang benar)
- FP: *False Positive* (prediksi positif yang salah)
- FN: *False Negative* (prediksi negatif yang salah)

2.9.2 *Precision*

Presisi mengukur seberapa akurat model dalam memprediksi data positif dibandingkan dengan semua prediksi positif yang dilakukan. Rumusnya adalah:

$$Precision = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.9.50}$$

Nilai presisi yang tinggi menunjukkan bahwa model mampu menghindari memberikan prediksi positif yang salah (Bishop & Nasrabadi, 2020).

2.9.3 *Recall*

Recall mengukur sejauh mana model dapat mengidentifikasi seluruh data positif yang ada dalam dataset. Rumus *recall* adalah:

$$Recall = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.9.51}$$

Nilai *recall* yang tinggi menunjukkan bahwa model memiliki sensitivitas yang baik terhadap data positif (Fawcett, 2020).

2.9.4 *F1-Score*

F1-Score adalah rata-rata harmonik dari presisi dan *recall*, memberikan keseimbangan antara keduanya. Rumusnya adalah:

$$\textit{F1-Score} = 2 \times \frac{\textit{Precision} \times \textit{Recall}}{\textit{Precision} + \textit{Recall}} \tag{2.9.52}$$

F1-Score cocok digunakan ketika terdapat ketidakseimbangan antara jumlah data positif dan negatif dalam dataset (Bishop & Nasrabadi, 2020).

2.9.5 ROC-AUC

ROC-AUC mengukur kemampuan model untuk membedakan antara kelas positif dan negatif. ROC menunjukkan hubungan antara *true positive rate* (TPR) atau *recall* dan *false positive rate* (FPR), sedangkan AUC adalah luas area di bawah kurva tersebut (Fawcett, 2020).

$$Recall = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.9.53}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{2.9.54}$$

Nilai AUC mendekati 1 menunjukkan performa model yang baik.

BAB III

METODE PENELITIAN

3.1 Waktu dan Tempat Penelitian

Penelitian ini dilakukan pada semester genap pada tahun pelajaran 2023/2024 dan semester ganjil pada tahun pelajaran 2024/2025 di BRIN KST Samaun Samadikun Bandung dan Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung.

3.2 Metode Penelitian

Penelitian ini menggunakan metode studi literatur yang berfokus pada skripsi terdahulu, jurnal-jurnal yang relevan, serta buku-buku yang terdapat di perpustakaan Universitas Lampung dan ruang baca Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung. Dengan demikian, penelitian ini memanfaatkan berbagai sumber yang mendukung dan memperkaya kajian yang dilakukan.

Adapun langkah yang digunakan dalam menyelesaikan penelitian ini antara lain:

1. Mempelajari arsitektur algoritma yang menjadi topik penelitian.
2. Menentukan dan mengidentifikasi kasus permasalahan yang akan diselesaikan menggunakan metode *deep learning*.
3. Merancang algoritma berdasarkan konsep yang telah dipelajari, kemudian mengimplementasikannya dalam bentuk program komputer menggunakan Google Colab.
4. Menggunakan program yang telah dibuat untuk menyelesaikan kasus permasalahan pada masing-masing metode.

5. Menganalisis hasil metrik evaluasi dari setiap metode, serta membandingkan performa masing-masing metode dalam menyelesaikan kasus yang telah ditentukan.

6. Menentukan kesimpulan.

Penelitian ini berfokus pada mengembangan model hibrida *deep learning* dalam menghadapi klasifikasi sentimen judul berita online. Dalam peneliatian ini digunakan data judul berita online mengenai covid-19 tahun 2020, yang disebut InaCOVED. Penelitian ini menggunakan hardware dengan spesifikasi processor intel core i5-1135G7 processor 2.4 GHz, memori 4GB DDR4 on board + 4GB DDR4, storage 256GB SSD, graphics intel iris Xe graphics. Selain itu, software yang digunakan yaitu Google Colab dan beberapa library yang digunakan iyalah pandas, numpy, tensorflow, sklearn, keras, dan matplotlib. Adapun langkah penelitian sebagai berikut:

1. Mempersiapkan data yang akan digunakan dalam penelitian (*preprocessing* data).

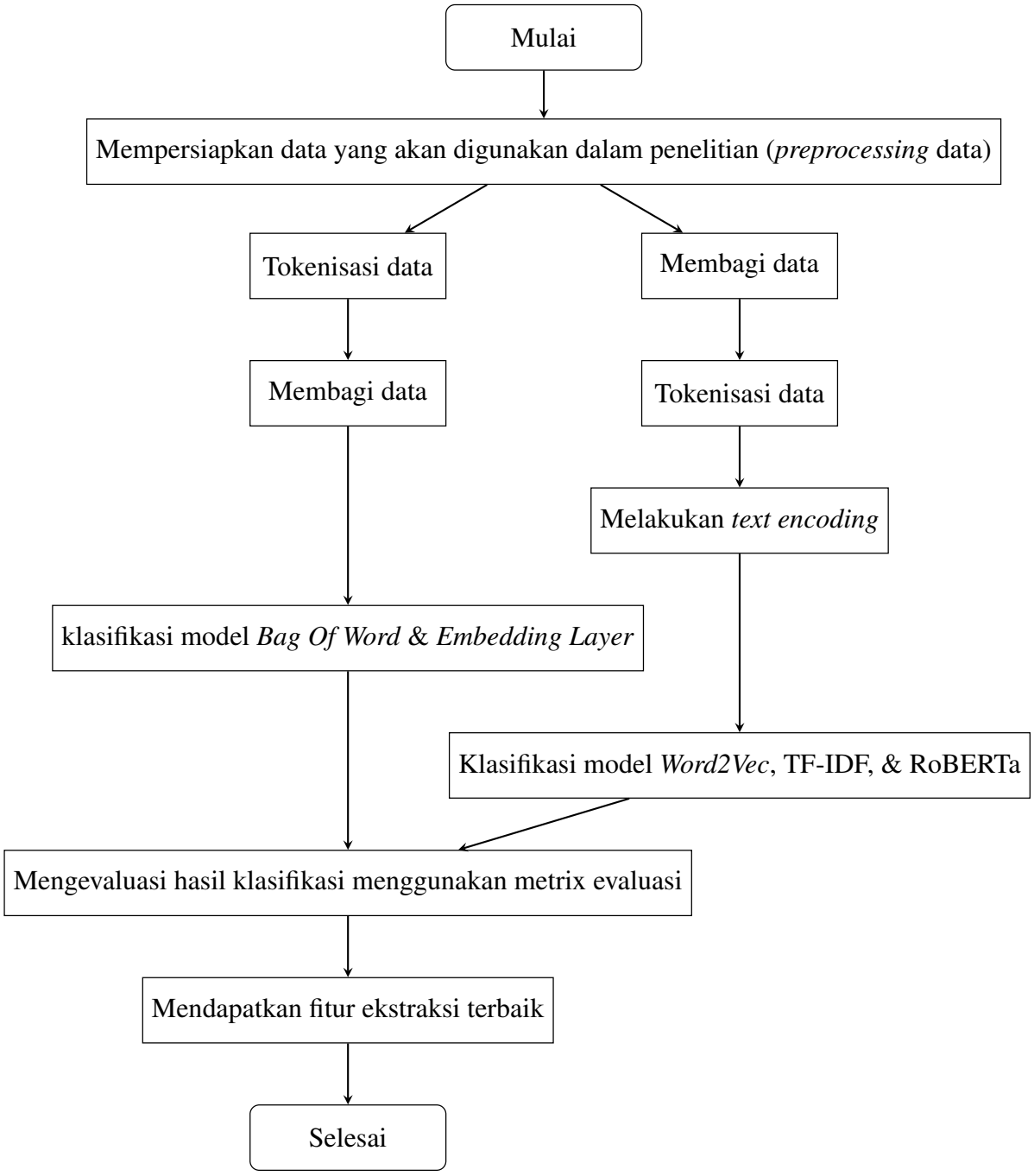
2. Melakukan tokenisasi pada data dan pembagian data menjadi *training data*, *testing data*, dan *validation data*.

3. Melakukan *text encoding* menggunakan *Embedding Layer*, TF-IDF, *Word2Vec*, dan RoBERTa.

4. Melakukan klasifikasi sentimen pada data menggunakan model-model CNN hibrida (LSTM, BiLSTM).

5. Mengevaluasi hasil kinerja algoritma.

Supaya memberikan gambaran yang lebih jelas mengenai proses penelitian ini, maka disiapkan *flowchart* yang disusun untuk mengilustrasikan aliran kerja penelitian ini, yang ditunjukkan pada gambar 3.1.



Gambar 3.1 *flowchart* alur proses klasifikasi

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Penelitian ini mengevaluasi kinerja model CNN yang dihibridasi dengan model LSTM dan BiLSTM, serta model encoding seperti embedding layer, Word2Vec, TF-IDF, dan RoBERTa. Perbandingan model munggunakan metrik evaluasi dilakukan untuk mendapatkan model terbaik. Metrik evaluasi yang digunakan untuk menentukan kinerja model yaitu *accuracy, Precision, Recall, F1-Score*, dan AUC-ROC, serta waktu latih untuk menentukan model paling efisien.

Dari hasil penelitian dapat ditarik beberapa kesimpulan, sebagai berikut:

- Hibridasi model CNN dengan model LSTM dan BiLSTM dapat meningkatkan kinerja model.
- Model *Encoding* yang digunakan sangat berpengaruh pada kinerja model *deep learning*.
- Besar dimensi vektor berpengaruh pada kinerja model *deep learning*.
- Frekuensi data di setiap kelas berbanding lurus dengan hasil kinerja model pada setiap kelas data.

Berdasarka hasil penelitian, model dengan hasil evaluasi kinerja terbaik adalah model RoBERTa-CNN-BiLSTM dimensi vektor 38 dengan nilai *accuracy* 0.9818, nilai *precision* 0.9819, nilai *recall* 0.9818, nilai *F1-Score* 0.9779, dan nilai ROC-AUC 0.9968, dengan waktu latih 8.100 detik atau 2 jam 15 menit menggunkan GPU. Model dengan waktu latih terifisien dan kinerja yang cukup baik adalah model *embedding layer*-CNN-BiLSTM dimensi vektor 100 dengan nilai *accuracy* 0.9327, nilai *precision* 0.9314, nilai *recall* 0.9327, nilai *F1-Score* 0.9318, dan nilai ROC-AUC

0.9759, dengan waktu latih 810 detik atau 13 menit 30 detik menggunakan CPU.

5.2 Saran

Penelitian ini berkontribusi dalam pengembangan model *deep learning* dalam penyelesaian penugasan klasifikasi teks pendek, khususnya model *convolutional neural network* (CNN) dalam penyelesaian penugasan klasifikasi judul berita online. Pengembangan model dengan kinerja terbaik berdasarkan metrik evaluasi seperti *accuraccy, precision, recall, F1-Score,* dan AUC-ROC, serta keefisienan waktu latih model.

Keterbatasan penetian ini yaitu pada *dataset* yang terbatas dan tidak seimbang. Diharapkan penelitian selajutnya akan berfokus pada data teks pendek dalam berbagai bidang seperti ekonomi, pendidikan, dan politik. Selain itu, pengelolaan data yang tidak seimbang akan dilakukan dalam rangka peningkatan efektifitas kinerja model. Serta pengembangan model yang memberikan kinerja lebih baik dan waktu latih yang lebih efisien.

DAFTAR PUSTAKA

Abdullah, S. M. S. A., Ameen, S. Y. A., Sadeeq, M. A., & Zeebaree, S. (2021). Multimodal emotion recognition using deep learning. *Journal of Applied Science and Technology Trends*, 2(01), 73-79.

Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., ... & Zhu, Z. (2016, June). Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning* (pp. 173-182). PMLR.

Beaglehole, D., Radhakrishnan, A., Pandit, P., & Belkin, M. (2023). Mechanism of feature learning in convolutional neural networks. *arXiv* preprint arXiv:2309.00570.

Bishop, C. M., & Nasrabadi, N. M. (2006). Pattern recognition and machine learning (Vol. 4, No. 4, p. 738). New York: *springer*.

Bistritz, I., Kahana, D., Bambos, N., Ben-Gal, I., & Yamin, D. (2019, December). Controlling contact network topology to prevent measles outbreaks. In 2019 *IEEE Global Communications Conference (GLOBECOM)* (pp. 1-6). IEEE.

Cambria, E., Schuller, B., Xia, Y., & Havasi, C. (2013). New avenues in opinion mining and sentiment analysis. *IEEE Intelligent Systems*, 28(2), 15–21.

Chowdhury, S., Dey, P., Joel-Edgar, S., Bhattacharya, S., Rodriguez-Espindola, O., Abadie, A., & Truong, L. (2023). Unlocking the value of artificial intelligence in human resource management through AI capability framework. *Human resource management review*, 33(1), 100899.

Ciresan, D. C., Meier, U., Masci, J., Gambardella, L. M., & Schmidhuber, J. (2011, June). Flexible, high performance convolutional neural networks for image classification. In *Twenty-second international joint conference on artificial intelligence*.

Devika, M. D., Sunitha, C., & Ganesh, A. (2016). Sentiment analysis: a comparative study on different approaches. *Procedia Computer Science*, 87, 44-49.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Volume 1 (Long and Short Papers), 4171–4186.

Dietterich, T. G. (2000, June). Ensemble methods in machine learning. In International workshop on multiple classifier systems (pp. 1-15). *Berlin, Heidelberg: Springer Berlin Heidelberg*.

Dos Santos, Cicero, and Maira Gatti. "Deep convolutional neural networks for sentiment analysis of short texts." Proceedings of COLING 2014, the 25th international conference on computational linguistics: technical papers. 2014.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern recognition letters*, 27(8), 861-874.

Goodfellow, I. (2016). *Deep learning*.

Graves, A., Jaitly, N., & Mohamed, A. R. (2013, December). Hybrid speech recognition with deep bidirectional LSTM. In *2013 IEEE workshop on automatic speech recognition and understanding* (pp. 273-278). IEEE.

Huang, T., Zhang, Q., Tang, X., Zhao, S., & Lu, X. (2022). A novel fault diagnosis method based on CNN and LSTM and its application in fault diagnosis for complex systems. *Artificial Intelligence Review*, 55(2), 1289-1315.

Jiang, Zilong, Shu Gao, and Liangchen Chen. "Study on text representation method based on deep learning and topic information." Computing 102.3 (2020): 623-642.

Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. *arXiv* preprint arXiv:1404.2188.

Kapoor, A., Pathiraja, S., Marshall, L., & Chandra, R. (2023). DeepGR4J: A deep learning hybridization approach for conceptual rainfall-runoff modelling. *Environmental Modelling & Software*, 169, 105831.

Kenton, J. D. M. W. C., & Toutanova, L. K. (2019, June). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT* (Vol. 1, p. 2).

Khan, A., Sohail, A., Zahoora, U., & Qureshi, A. S. (2020). A survey of the recent architectures of deep convolutional neural networks. *Artificial intelligence review*, 53, 5455-5516.

Khotimah, P. H., Arisal, A., Rozie, A. F., Nugraheni, E., Riswantini, D., Suwarningsih, W., ... Purwarianti, A. (2023). Monitoring Indonesian online news for COVID-19 event detection using deep learning. International Journal of Electrical Computer Engineering (2088-8708), 13(1).

Kim, Yoon. 2014. Convolutional Neural Networks for Sentence Classification.

Kim, H., & Jeong, Y. S. (2019). Sentiment classification using convolutional neural networks. *Applied Sciences*, 9(11), 2347.

Kingma, D. P. (2014). Adam: A method for stochastic optimization. *arXiv preprint* arXiv:1412.6980.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

Liang, S., Chen, X., Ma, J., Du, W., & Ma, H. (2021). An Improved Double Channel Long Short-Term Memory Model for Medical Text Classification. *Journal of healthcare engineering*, 2021(1), 6664893.

Liliana, D. Y., Hikmah, N. N., & Harjono, M. (2021). PENGEMBANGAN SISTEM PEMANTAUAN SENTIMEN BERITA BERBAHASA INDONESIA BERDASARKAN KONTEN DENGAN LONG SHORT-TERM MEMORY. *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, 8(5).

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1), 1–167.

Lindsay, G. W., & Serre, T. (2021). Deep Learning Networks and Visual Perception. *In Oxford Research Encyclopedia of Psychology*.

Liu, Y. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv* preprint arXiv:1907.11692, 364.

Loshchilov, I., & Hutter, F. (2016). Sgdr: Stochastic gradient descent with warm restarts. *arXiv* preprint arXiv:1608.03983.

Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." Advances in *neural information processing systems* 26 (2013).

Minaee, S., Azimi, E., & Abdolrashidi, A. (2019). Deep-sentiment: Sentiment analysis using ensemble of cnn and bi-lstm models. *arXiv* preprint arXiv:1904.04206.

Nugraheni, E., Khotimah, P. H., Arisal, A., Rozie, A. F., Riswantini, D., & Purwarianti, A. (2020, November). Classifying aggravation status of COVID-19 event from short-text using CNN. In 2020 *International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET)* (pp. 240-245). IEEE.

Ouyang, X., Zhou, P., Li, C. H., & Liu, L. (2015, October). Sentiment analysis using convolutional neural network. In 2015 *IEEE international conference on computer and information technology; ubiquitous computing and communications; dependable, autonomic and secure computing; pervasive intelligence and computing* (pp. 2359-2364). IEEE.

P. H. Khotimah, "InaCOVED (Indonesian Corpus for COVID-19 Event Detection) on Online News," 2023. [Online]. Available: https://hdl.handle.net/20.500.12690/RIN/8K14KW

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. Foundations and Trends® in *Information Retrieval*, 2(1–2), 1–135.

Powers, D. M. (2020). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv* preprint arXiv:2010.16061.

Putra, W. S. E. (2016). Klasifikasi citra menggunakan convolutional neural network (CNN) pada caltech 101. *Jurnal Teknik ITS*, 5(1).

Qiu, J., Wang, B., & Zhou, C. (2020). Forecasting stock prices with long-short term memory neural network based on attention mechanism. *PloS one*, 15(1), e0227222.

Rahman, M. M., Shiplu, A. I., Watanobe, Y., & Alam, M. A. (2024). RoBERTa-BiLSTM: A Context-Aware Hybrid Model for Sentiment Analysis. *arXiv* preprint arXiv:2406.00367.

Rhanoui, M., Mikram, M., Yousfi, S., & Barzali, S. (2019). A CNN-BiLSTM model for document-level sentiment analysis. Machine Learning and Knowledge Extraction, 1(3), 832-847.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115, 211-252.

Scherer, D., Müller, A., & Behnke, S. (2010, September). Evaluation of pooling operations in convolutional architectures for object recognition. *In International conference on artificial neural networks* (pp. 92-101). Berlin, Heidelberg: Springer Berlin Heidelberg.

Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11), 2673-2681.

Serghini, O., Semlali, H., Maali, A., Ghammaz, A., & Serrano, S. (2023). 1-D Convolutional Neural Network-Based Models for Cooperative Spectrum Sensing. *Future Internet*, 16(1), 1

She, X., & Zhang, D. (2018, December). Text classification based on hybrid CNN-LSTM hybrid model. In 2018 11th *International symposium on computational intelligence and design* (ISCID) (Vol. 2, pp. 185-189). IEEE.

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of big data*, 6(1), 1-48.

Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25.

Tan, K. L., Lee, C. P., Lim, K. M., & Anbananthen, K. S. M. (2022). Sentiment analysis with ensemble hybrid deep learning model. *IEEE Access*, 10, 103694-103704.

Vaswani, A. (2017). Attention is all you need. Advances in *Neural Information Processing Systems*.

Wang, X., Han, Y., Leung, V. C., Niyato, D., Yan, X., & Chen, X. (2020). Convergence of edge computing and deep learning: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(2), 869-904.

Wang, Y., Lo, K., Chandrasekhar, Y., Reas, R., Yang, J., Eide, D., ... & McCallum, A. (2020). CORD-19: The COVID-19 open research dataset. *arXiv* preprint, arXiv:2004.10706.

Wu, J. L., He, Y., Yu, L. C., & Lai, K. R. (2020). Identifying emotion labels from psychiatric social texts using a bi-directional LSTM-CNN model. *IEEE Access*, 8, 66638-66646.

Yang, R. (2020). Convolutional neural networks and their applications in NLP. Seminar NLP SS20. https://slds-lmu.github.io/seminar_nlp_ss20/convolutional-neural-networks-and-their-applications-in-nlp.html

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2021). Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3), 107-115.

Zhou, H. (2022). Research of text classification based on TF-IDF and CNN-LSTM. In *Journal of Physics: Conference Series* (Vol. 2171, No. 1, p. 012021). IOP Publishing.

Zhou, Y., Zheng, S., & Zhang, G. (2019). Artificial neural network based multivariable optimization of a hybrid system integrated with phase change materials, active cooling and hybrid ventilations. *Energy Conversion and Management*, 197, 111859.

Zhao, R., Yang, Z., Zheng, H., Wu, Y., Liu, F., Wu, Z., ... & Shi, L. (2022). A framework for the general design and computation of hybrid neural networks. *Nature communications*, 13(1), 3427.

Zoumpourlis, G., Doumanoglou, A., Vretos, N., & Daras, P. (2017). Non-linear convolution filters for cnn-based learning. In *Proceedings of the IEEE international conference on computer vision* (pp. 4761-4769).