KLASIFIKASI MASALAH PADA KOMUNITAS MARAH-MARAH DI TWITTER MENGGUNAKAN *BIDIRECTIONAL LONG SHORT-TERM MEMORY*

Disusun Oleh:

N a m a : Dian Sukma Hani

NIM : 19523181

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2023

HALAMAN PENGESAHAN DOSEN PEMBIMBING

KLASIFIKASI MASALAH PADA KOMUNITAS MARAH-MARAH DI TWITTER MENGGUNAKAN *BIDIRECTIONAL LONG SHORT-TERM MEMORY*

TUGAS AKHIR

Disusun Oleh:

N a m a : Dian Sukma Hani

NIM : 19523181

Yogyakarta, 15 Oktober 2023

Pembimbing,

(Chanifah Indah Ratnasari, S.Kom., M.Kom.)

HALAMAN PENGESAHAN DOSEN PENGUJI

KLASIFIKASI MASALAH PADA KOMUNITAS MARAH-MARAH DI TWITTER MENGGUNAKAN *BIDIRECTIONAL LONG SHORT-TERM MEMORY*

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 15 Oktober 2023

Tim Penguji

Chanifah Indah Ratnasari, S.Kom., M.Kom. _____

Anggota 1

Arrie Kurniawardhani, S.Si., M.Kom. _____

Anggota 2

Sheila Nurul Huda, S.Kom., M.Cs. _____

Mengetahui,

Ketua Program Studi Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia

(Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Dian Sukma Hani

NIM : 19523181

Tugas akhir dengan judul:

KLASIFIKASI MASALAH PADA KOMUNITAS MARAH-MARAH DI TWITTER MENGGUNAKAN *BIDIRECTIONAL LONG SHORT-TERM MEMORY*

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 15 Oktober 2023

(Dian Sukma Hani)

HALAMAN PERSEMBAHAN

Teruntuk Abi dan Mama yang sudah membawa saya ke dunia. Terima kasih untuk semua bantuan berupa materi maupun nonmateri. Untuk setiap doa-doa yang selalu dipanjatkan dalam setiap helaan napas.

Teruntuk Ibu (Nenek) dan Alm. Abah (Kakek) yang telah berpulang kepada sebenar-benarnya pemilik kehidupan. Terima kasih sudah menjadi orang tua kedua yang mengasuh saya sejak kecil, yang mengajari bagaimana cara berjalan hingga bagaimana cara menunaikan salat.

Teruntuk Ade Kurnia Dewi dan Baita Rohmana. Terima kasih untuk semua waktu dan pengertiannya. Terima kasih sudah menjadi tempat pulang.

Teruntuk Dian Sukma Hani, terima kasih untuk tidak berhenti mencari alasan untuk tetap hidup.

HALAMAN MOTO

‘Apakah manusia mengira bahwa mereka akan dibiarkan hanya dengan mengatakan, “Kami telah beriman,” dan mereka tidak diuji?’

(QS. Al-Ankabut:2)

‘Apa yang melewatkanku tidak akan pernah menjadi takdirku, dan apa yang ditakdirkan untukku tidak akan pernah melewatkanku.’

(Umar bin Khattab)

‘Apa yang menjadi takdirmu akan mencari jalannya sendiri untuk menemukanmu.’

(Ali bin Abi Thalib)

KATA PENGANTAR

Assalamu alaikum wa rahmatullahi wa barakaatuh,

Alhamdulillahirabbil'alamin, puji dan syukur atas kehadirat Allah SWT yang telah melimpahkan segala rahmat dan karunia-Nya, sehingga penulisan skripsi dengan judul 'Klasifikasi Masalah pada Komunitas Marah-Marah di Twitter Menggunakan *Bidirectional Long Short-Term Memory'* ini dapat diselesaikan. Skripsi ini ditulis sebagai salah satu persyaratan untuk menyelesaikan program pendidikan jenjang Sarjana di Program Studi Informatika Universitas Islam Indonesia. Selawat serta salam selalu senantiasa tercurahkan kepada Nabi Muhammad SAW beserta keluarga dan para sahabat.

Selama proses penyusunan tugas akhir ini, berbagai tantangan dan permasalahan selalu ada. Namun, penulis sangat menghargai semua yang telah memberikan kontribusi dalam bentuk dukungan, bimbingan, arahan, semangat, dan doa. Oleh karena itu, penulis ingin mengucapkan rasa terima kasih kepada:

1. Dr. Raden Teduh Dirgahayu, S.T., M.Sc. selaku Ketua Jurusan Informatika beserta seluruh jajarannya.
2. Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D. selaku Ketua Program Studi Informatika Program Sarjana Fakultas Teknologi Industri, Universitas Islam Indonesia.
3. Chanifah Indah Ratnasari, S. Kom., M. Kom. selaku dosen pembimbing yang selalu sabar dalam mendampingi, membimbing, dan mengarahkan selama proses penyusunan tugas akhir ini.
4. Seluruh dosen Jurusan Informatika UII yang telah memberikan banyak ilmu bagi penulis.
5. Semua pihak yang sudah menemani dan membantu penulis dalam menyusun tugas akhir ini.

Yogyakarta, 15 Oktober 2023

(Dian Sukma Hani)

SARI

Twitter adalah salah satu platform media sosial yang memiliki basis pengguna yang besar di Indonesia. Pada Twitter, pengguna diberi keleluasaan untuk berbagi momen dan pemikiran pribadi mereka tanpa ada pembatasan yang signifikan. Banyak dari mereka yang menjadikan platform ini sebagai wadah untuk mengungkapkan amarah, seperti yang terlihat pada Komunitas Marah-Marah. Bergabung dengan komunitas ini hanya memerlukan persetujuan admin, tanpa persyaratan khusus yang harus dipenuhi oleh pengguna. Dalam Komunitas Marah-Marah, para anggota memiliki kesempatan untuk secara bebas mengeluarkan dan berbagi amarah mereka. Penelitian ini bertujuan untuk mengklasifikasikan twit dari Komunitas Marah-Marah ke dalam jenis atau kategori permasalahan yang sesuai. Data teks diambil dengan menggunakan teknik web *scraping*, kemudian melewati sejumlah tahapan *preprocessing*, termasuk penghapusan karakter yang tidak relevan, normalisasi, tokenisasi, dan *stemming*. Melalui pemanfaatan algoritma *Bidirectional Long Short-Term Memory* (Bi-LSTM), teks berhasil diklasifikasikan ke dalam enam kategori permasalahan yang berbeda, seperti Studi, Percintaan, Keluarga, Karier/Pekerjaan, Person/Personal, dan Tidak Diketahui Masalahnya. Hasil evaluasi menunjukkan keberhasilan model mencapai tingkat akurasi data latih sebesar 0,8033 % dan akurasi data uji sebesar 0,8005%.

Kata kunci: klasifikasi, klasifikasi masalah, Twitter, Bi-LSTM, *deep learning*

GLOSARIUM

Algoritma	Panduan sistematis yang digunakan dalam komputasi, matematika, dan ilmu komputer untuk memecahkan masalah atau mencapai tujuan dengan mengikuti serangkaian langkah yang jelas dan terdefinisi.
Deep Learning	Subbidang dalam pembelajaran mesin (*machine learning*) yang berfokus pada pengembangan jaringan saraf tiruan (*neural networks*) yang sangat dalam untuk mengekstraksi pola dan fitur dari data.
Klasifikasi	Proses pengelompokan atau pemisahan objek atau data ke dalam beberapa kategori atau kelas yang telah ditentukan sebelumnya berdasarkan karakteristik atau atribut tertentu.
Machine Learning	Cabang dari kecerdasan buatan yang fokus pada pengembangan algoritma komputer yang memungkinkan sistem untuk belajar dari data dan pengalaman tanpa perlu diprogram secara eksplisit.
Preprocessing	Serangkaian langkah atau proses yang dilakukan pada data mentah (*raw data*) sebelum data tersebut digunakan dalam analisis atau pemodelan lebih lanjut.
Web Scraping	Teknik pengambilan data dari halaman web secara otomatis menggunakan perangkat lunak atau skrip komputer.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pada abad ini, revolusi media sosial berbarengan dengan pesatnya perkembangan teknologi informasi (Antinasari et al., 2017). Selain itu, berkembangnya media sosial berbanding lurus dengan meningkatnya masyarakat pengguna internet (Syauqi et al., 2020). Dalam periode tahun 2019-2020, internet digunakan oleh sekitar 73,3% atau sekitar 196,71 juta orang dari total populasi Indonesia yang mencapai 266,91 juta individu (Febrianti et al., 2021). Media sosial yang memiliki banyak pengguna di Indonesia di antaranya adalah Twitter dan Facebook (Gupta et al., 2022). Media sosial Twitter telah menjadi integral dalam komunikasi masyarakat Indonesia. Pertumbuhannya didorong oleh jumlah pengguna yang besar dan lonjakan jumlah *tweet* yang diposting setiap harinya (Rahutomo et al., 2018; Suharso, 2019). Pada bulan Januari 2019, terdapat sekitar 6,43 juta orang yang aktif menggunakan Twitter (Tineges et al., 2020). Kebebasan pada Twitter sebagai platform media sosial mendorong pengguna untuk lebih mudah mengekspresikan diri (Nugraha & Azhar, 2022). Pengguna Twitter memiliki beragam cara untuk mengekspresikan diri, salah satunya adalah dengan membuat postingan berupa teks (Budiman et al., 2021). Postingan teks tersebut memungkinkan pengguna di Twitter untuk berbagi pikiran, pendapat, cerita, atau pesan dengan pengguna lain. Dari berbagai postingan tersebut, *tweeps*, atau pengguna Twitter, secara tidak langsung memberikan informasi mengenai diri mereka, keluhan, emosi, serta masalah yang tengah mereka hadapi (Gaind et al., 2019; Mardiana & Zi'ni, 2020).

Twitter menjadi tempat bagi pengguna untuk saling bertukar informasi, baik melalui *timeline* pribadi maupun dalam berbagai komunitas yang sudah terbentuk (Maulani & Priyambodo, 2021). Twitter mulai memperkenalkan fitur komunitas kepada pengguna pada 9 September 2021. Komunitas menggambarkan tempat bagi sekelompok individu dengan minat dan ketertarikan yang serupa (Nurhaliza & Fauziah, 2020). Cara kerja komunitas di platform Twitter hampir sama dengan fitur grup yang ditemukan di Facebook (Artika, 2021). Salah satu contoh komunitas di Twitter adalah Komunitas Marah-Marah (url: https://bit.ly/KomunitasMarah-Marah). Komunitas Marah-Marah bersifat tertutup sehingga jika ingin bergabung harus menunggu persetujuan dari admin komunitas. Komunitas Marah-Marah didirikan pada tanggal 24 Agustus 2022. Hingga saat ini, jumlah anggota dalam komunitas ini telah mencapai 114 ribu orang. Dalam lingkungan Komunitas Marah-Marah,

para pengguna platform Twitter memiliki kesempatan secara terbuka untuk mengekspresikan perasaan kemarahan, frustrasi, dan emosi negatif lainnya terkait berbagai aspek kehidupan mereka. Melalui ruang dialog yang dihasilkan oleh Komunitas Marah-Marah, para anggotanya memiliki kebebasan untuk berbicara tentang masalah mereka, termasuk situasi yang memicu kemarahan atau bahkan peristiwa yang menimbulkan dampak emosional yang mendalam. Namun, anggota Komunitas Marah-Marah harus mematuhi beberapa peraturan jika tidak ingin dikeluarkan. Peraturan tersebut antara lain dilarang memposting mengenai barang jualan, politik, kekerasan, dan *self harm*. Tidak diperkenankan men-*tweet* yang mengandung SARA, wajib menghargai *privacy* setiap anggota dengan tidak menyebarkan postingan *tweet* anggota lain. Jika ada anggota yang menyalahi aturan tersebut maka admin komunitas akan mengeluarkan anggota yang melanggar aturan dari Komunitas Marah-Marah.

Menurut Kamus Besar Bahasa Indonesia, marah dapat diartikan sebagai reaksi ketidakpuasan terhadap perlakuan yang dirasa tidak pantas (Munawir, 2023). Permasalahan merupakan suatu keadaan yang menghambat seseorang untuk mencapai berbagai tujuan yang telah ditetapkan (Nugraha, 2018). Adanya masalah adalah sesuatu yang tidak diinginkan dan tidak diharapkan oleh semua orang (Indrawati, 2020). Oleh karena itu, masalah merupakan sesuatu yang harus diselesaikan, agar tidak menjadi penyebab terjadinya suatu amarah yang berkepanjangan (Tasik et al., 2022). Twit yang berada pada Komunitas Marah-Marah perlu dipetakan agar mengetahui apa saja penyebab atau masalah apa saja yang memicu anggota Komunitas Marah-Marah merasakan amarah. Masalah yang dialami manusia dapat dikategorikan menjadi beberapa kategori, seperti permasalahan Studi, Percintaan, Keluarga, Karier/Pekerjaan, Person/Personal, dan Tidak Diketahui Masalahnya.

Pada masa kini, permasalahan seseorang dapat diketahui hanya dengan melihat postingan media sosialnya (Fajar, 2018). Teks twit mengenai permasalahan dapat diolah lebih lanjut dengan beberapa pendekatan, seperti deteksi (Marta et al., 2022), analisis (Nandwani & Verma, 2021), dan klasifikasi (Hasanah et al., 2021). Klasifikasi merupakan salah satu metode untuk mengetahui permasalahan apa yang sedang dialami seseorang (Yuliana & Supriyanto, 2019). Metode klasifikasi dapat digunakan untuk mendapatkan informasi mengenai pengelompokan sesuatu sesuai kategorinya (Indrayuni, 2019). Metode klasifikasi dapat dilakukan dengan beberapa macam masukan, seperti audio atau suara (Waasiu et al., 2021), gambar (Putri, 2020), video (Kristian et al., 2018), dan teks (Widhiyasana et al., 2021). Klasifikasi menggunakan masukan teks dilakukan dengan mengekstrak informasi yang didapatkan, sehingga hasilnya dapat memberikan sebuah informasi.

Klasifikasi merujuk pada tindakan mengategorikan atau mengelompokkan objek ke dalam kategori yang telah disepakati di awal (Ardiani et al., 2020). Klasifikasi dapat dilakukan dengan berbagai macam metode. Penelitian mengenai penggunaan berbagai metode untuk proses klasifikasi telah banyak dilakukan pada penelitian-penelitian sebelumnya. Meskipun demikian, penerapan metode *deep learning* dinilai lebih unggul daripada pendekatan *machine learning* (Hasanah et al., 2021; Ihsan et al., 2022). Kemampuan *deep learning* dianggap sangat efektif dalam melakukan pemodelan pada berbagai jenis data yang kompleks, seperti suara, gambar, dan teks (Ihsan et al., 2022). Solusi untuk mengatasi kerumitan dalam pemrosesan teks telah lama ditemukan, salah satunya dengan menggunakan pengembangan algoritma *Long Short-Term Memory* (LSTM) yang berasal dari *Recurrent Neural Network* (RNN) (Nugroho et al., 2021). Namun, terdapat batasan dalam kemampuan algoritma *Long Short-Term Memory* (LSTM), yaitu hanya mampu melakukan pembacaan informasi secara satu arah (Mukhlis et al., 2021). Karenanya, sebagai alternatif, konsep *Bidirectional Long Short-Term Memory* (BiLSTM) diajukan untuk mengatasi isu hilangnya *gradient* dengan memperhitungkan pengambilan informasi dari dua arah (Nugroho et al., 2021).

Berdasarkan pemaparan yang disajikan di atas, penelitian ini akan melakukan klasifikasi masalah berupa teks pada Komunitas Marah–Marah di Twitter menggunakan metode *Bidirectional Long-Short Term Memory* (BiLSTM). Data teks akan dikelompokkan ke dalam beberapa kategori permasalahan, seperti Studi, Percintaan, Keluarga, Karier/Pekerjaan, Person/Personal, dan Tidak Diketahui Masalahnya. Algoritma *Bidirectional Long Short-Term Memory* (BiLSTM) dipilih karena kemampuannya untuk memproses informasi dalam dua arah, sehingga dapat mengatasi masalah hilangnya gradien dengan lebih efektif (Karyadi & Santoso, 2022). Penelitian ini dilakukan agar mengetahui masalah apa yang menyebabkan anggota Komunitas Marah-Marah merasakan amarah, sehingga hasil dari penelitian ini dapat digunakan sebagai bahan penelitian lain.

1.2 Rumusan Masalah

Rumusan masalah pada penelitian ini yaitu:

a. Bagaimana cara melakukan klasifikasi masalah pada Komunitas Marah-Marah menggunakan *Bidirectional Long Short-Term Memory* (BiLSTM)?
b. Berapa persentase akurasi ketepatan model dalam mengklasifikasikan masalah pada Komunitas Marah-Marah di Twitter menggunakan algoritma *Bidirectional Long Short-Term Memory* (BiLSTM)?

1.3 Batasan Masalah

Batasan masalah pada penelitian ini adalah:

a. Data yang diambil berasal dari Komunitas Marah-Marah di Twitter.
b. Rentang waktu pengambilan data dimulai dari 11 Maret 2023 – 17 Juli 2023.
c. Postingan twit yang diambil adalah twit yang berbahasa Indonesia.

1.4 Tujuan Penelitian

Tujuan penelitian ini adalah:

a. Mengetahui cara melakukan klasifikasi masalah pada twit Komunitas Marah-Marah di Twitter.
b. Mengetahui seberapa tepat klasifikasi masalah menggunakan algoritma *Bidirectional Long Short-Term Memory* (BiLSTM) dengan menggunakan data dari Komunitas Marah-Marah di Twitter.

1.5 Manfaat Penelitian

Manfaat penelitian ini adalah:

a. Mengetahui permasalahan apa saja yang menjadi masalah dari anggota Komunitas Marah-Marah di Twitter.
b. Hasil dari penelitian ini dapat digunakan sebagai bahan penelitian lain, sehingga dapat memberikan manfaat yang lebih luas.

1.6 Metodologi Penelitian

Tahapan-tahapan yang diterapkan dalam penelitian ini agar mencapai tujuan yang diinginkan adalah sebagai berikut:

a. Pengumpulan Data

Pengumpulan data dilakukan dengan menggunakan teknik *scraping* menggunakan tools *visual code studio* dengan bahasa pemrograman Python. Data yang dikumpulkan berasal dari media sosial Twitter pada Komunitas Marah-Marah. Data yang diambil disesuaikan dengan jumlah kebutuhan dan rentang waktu yang telah ditentukan.

b. Pelabelan Data

Setelah mendapatkan data yang berasal dari Komunitas Marah-Marah yang ada di Twitter, data tersebut dilakukan pemeriksaan apakah sesuai atau relevan dengan yang dibutuhkan atau

yang dicari. Kemudian dilanjutkan dengan proses penghapusan duplikasi data agar tidak terdapat banyak *noise* ketika pelatihan model. Setelah data siap, maka dilanjutkan untuk proses pelabelan data.

c. Pengolahan Data

Data yang sudah dilakukan pemeriksaan kemudian harus diolah sesuai dengan tujuan penelitian yaitu klasifikasi, tetapi sebelum itu data perlu dibersihkan agar tidak mengganggu pada saat dilakukan pelatihan. Proses pengelolaan data dimulai dengan tahapan berikut:

1. Melakukan *case folding.*
2. Menghapus tanda baca (!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~) dan spasi yang berlebih.
3. Melakukan proses tokenisasi.
4. Melakukan normalisasi teks.
5. Melakukan *stemming.*

d. Klasifikasi

Membangun arsitektur algoritma pemodelan yang akan digunakan pada data yang sudah dilakukan pembersihan dan *preprocessing*.

e. Evaluasi Model

Evaluasi dilakukan dengan tujuan memverifikasi bahwa arsitektur model yang sudah dibangun memenuhi tujuan dari penelitian yaitu untuk mengklasifikasi masalah.

1.7 Sistematika Penulisan

Sistematika penulisan dalam penyusunan laporan tugas akhir ini terdiri dari beberapa bab. Berikut merupakan sistematika penulisan yang tertuang dalam lima bab:

a. BAB I PENDAHULUAN

Pada bab ini berisi pembahasan latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian, dan sistematika penulisan. Setiap bab tersebut akan membahas mengenai klasifikasi masalah studi kasus: "Komunitas Marah-Marah" di Twitter.

b. BAB II LANDASAN TEORI

Pada bab ini berisi penjelasan mengenai studi literatur terhadap penelitian yang pernah ada berhubungan dengan apa yang akan dirancang dan diimplementasikan serta teori dasar yang digunakan berhubungan dengan klasifikasi masalah studi kasus: "Komunitas Marah-Marah" di Twitter.

c. BAB III METODOLOGI

Bab ini berisi uraian tentang algoritma model yang akan digunakan untuk membangun sistem klasifikasi masalah studi kasus: “Komunitas Marah–Marah” di Twitter.

d. BAB IV HASIL DAN PEMBAHASAN

Bab ini akan membahas tentang hasil dari pembangunan algoritma model yang diimplementasikan untuk klasifikasi masalah menggunakan postingan teks Twitter di “Komunitas Marah-Marah”. Pemaparan hasil evaluasi dengan menunjukkan hasil dari *Precision, Recall*, dan *F1-Score.*

e. BAB V KESIMPULAN DAN SARAN

Bab ini merupakan bab terakhir yang akan membahas kesimpulan dan saran terhadap penelitian mengenai sistem klasifikasi masalah (Studi: “Komunitas Marah-Marah” di Twitter).

BAB II
LANDASAN TEORI

2.1 Media Sosial

Pengguna aktif media sosial di Indonesia berkisar di angka 49% atau setara dengan 130 juta jiwa dari total populasi di Indonesia sebesar 256,4 juta jiwa (Puspitarini & Nuraeni, 2019). Media sosial terbentuk dari dua unsur kata, yakni media dan sosial. Secara etimologi media mengandung makna sebagai alat komunikasi, sedangkan sosial diartikan sebagai kehidupan masyarakat (Astajaya, 2020). Dalam pengertian lain, media sosial dapat diartikan sebagai perantara di internet yang memberikan kesempatan kepada penggunanya untuk merepresentasikan diri, bersosialisasi, melakukan interaksi, bekerja sama, dan melakukan komunikasi dengan pengguna lain (Puspitarini & Nuraeni, 2019). Bersosialisasi di media sosial merujuk pada nilai pengenalan (*cognition*), komunikasi (*communicate*), dan kerjasama (*cooperation*) (Anggraeni & Khasan, 2018). Media sosial memiliki tiga jenis yang berbeda, yaitu *Social Networks*, seperti Instagram, Facebook, Twitter. Komunitas *online* atau forum contohnya seperti Brainly, Kaskus, Indowebster, dan yang terakhir situs blog seperti juragancipir.com, BloggerBorneo.com, sugeng.id (Cahyono, 2018). Beragamnya jenis media sosial telah memutus jarak kelas sosial, hingga dapat menghubungkan orang untuk berkomunikasi lintas negara. Selain itu, media sosial tidak hanya berkutat dan ramai pada pembahasan permasalahan sosial dan politik tetapi sudah masuk ke dalam ranah menceritakan atau berbagi mengenai masalah pribadi (Astajaya, 2020).

2.2 Twitter

Media sosial yang memiliki banyak pengguna di Indonesia salah satunya adalah Twitter. Selain sebagai media sosial, Twitter digunakan sebagai media komunikasi oleh masyarakat Indonesia (Fikri et al., 2020). Badan Statistik Republik Indonesia menuturkan dari hasil penelitiannya bahwa pada Januari 2022, Twitter memiliki 342,75 juta aktivitas setiap harinya yang dapat dimonetisasi oleh pengguna dari seluruh dunia. Oleh karena itu, pengguna Twitter di Indonesia masuk dalam urutan kelima terbesar di dunia (Hidayatullah & Maharani, 2022). Selain itu, Indonesia berada di peringkat ketiga negara yang paling aktif membuat *tweet* setiap hari nya (Yuniasti, 2021). Twitter merupakan layanan media sosial dan mikroblog daring yang memberikan kesempatan penggunanya untuk mengirim dan membaca pesan dalam format teks atau sering disebut dengan kicauan atau *tweet* (Zukhrufillah, 2018). Twitter memungkinkan

penggunanya untuk membagikan opini, kehidupan privat pengguna, dan emosi pribadi mereka (Yu et al., n.d.). Emosional individu tersebut dapat berupa emosi kebahagiaan, sedih, kecewa, marah, dan lain sebagainya (Pachouly et al., 2021). Selain itu, Twitter juga menyediakan wadah bagi para pengguna yang memiliki ketertarikan dan minat yang sama. Pada 9 September 2021, Twitter memperkenalkan fitur baru yaitu fitur komunitas. Komunitas adalah tempat di mana sekelompok individu yang memiliki minat dan ketertarikan yang serupa berkumpul dan berinteraksi bersama (Nurhaliza & Fauziah, 2020). Cara komunitas beroperasi di platform Twitter hampir sama dengan cara grup berfungsi di Facebook (Artika, 2021). Beberapa komunitas memiliki aturan yang mengikat para anggotanya. Pada komunitas di Twitter terdiri dari admin yang mendirikan komunitas dan anggota yang memiliki ketertarikan terhadap topik komunitas.

2.3 *Scraping*

Scraping adalah metode untuk mengambil informasi dari beragam dokumen yang berada di sebuah *website* secara otomatis (Gunawan et al., 2019). Dalam prosesnya, dilakukan pengambilan materi atau konten yang sesuai dengan kueri, kemudian menggabungkan serta mengkonversi data dari format yang tidak memiliki struktur menjadi bentuk yang terstruktur (Suarez et al., 2018). Web *scraping* juga disebut dengan web *crawling* (Khder, 2021). Proses web *scraping* dilakukan dengan mengekstraksi data dari sebuah situs menggunakan *protocol* HTTP yang digunakan oleh *web browser*, prosedur ini bisa dijalankan baik secara manual maupun secara otomatis dengan memanfaatkan *web crawling* (Mauro et al., 2018; Zhao, 2017). Bahasa pemrograman yang biasa digunakan untuk *scraping* adalah *Python*, karena selain populer juga mudah untuk dipahami semua kalangan (Mauro et al., 2018). Web *scraping* melibatkan dua proses yaitu *scraper* dan *crawler*. Proses *crawler* merupakan proses mengunduh data dari internet, sedangkan proses *scraper* merupakan proses mengekstrak informasi penting dari web tujuan kemudian menyimpannya dalam *file* berdasarkan struktur dan format yang sudah ditetapkan sebelumnya (Khder, 2021).

2.4 Masalah

Manusia erat sekali kaitannya dalam hal permasalahan sehari-hari. Setiap individu manusia pasti memiliki masalah dalam hidupnya (Putri, 2019). Masalah dapat diartikan sebagai kesenjangan antara realita yang terjadi dengan harapan yang diinginkan. Dalam pengertian lain masalah dapat juga diterjemahkan sebagai tidak terpenuhinnya kebutuhan seseorang (Bastomi, 2020). Masalah terjadi karena erat keterikatannya dengan harapan atau keinginan individu yang

tidak terwujud (Putri, 2019). Menurut definisi dalam Kamus Besar Bahasa Indonesia, masalah merupakan sesuatu yang harus diselesaikan dan harus dicarikan jalan keluar atau pemecahanya (Ompusunggu, 2022). Seorang individu dianggap sedang menghadapi masalah ketika mengalami kondisi sebagai berikut: (1) memahami secara mendalam mengenai keadaan yang sedang terjadi; (2) memahami dengan jelas perkara yang menjadi tujuan; (3) mengetahui sumber daya yang dapat dimanfaatkan untuk menyelesaikan atau mengatasi permasalahan tersebut; (4) mempunyai kecakapan untuk menggunakan sumber daya tersebut untuk mencapai tujuan (Ginanjar, 2019; Sastria et al., 2022).

2.5 Marah

Marah dapat didefinisikan sebagai reaksi yang timbul akibat ketidakpuasan terhadap perlakuan yang dianggap tidak sesuai (Munawir, 2023). Pada beberapa kasus, munculnya amarah merupakan akibat dari adanya pikiran negatif terhadap sesuatu hal. Pikiran negatif ini berlangsung terus-menerus dan sulit untuk dikelola sendiri. Sering kali, hal ini menjadi pemicu masalah (Fajriyanti, 2022). Marah merupakan reaksi emosional atau afektif yang menyebabkan munculnya kesiapan psikologis untuk bertindak agresif sebagai respons terhadap situasi yang memicu emosi tersebut (Syahra, 2021). Dalam konteks psikologis, emosi marah dapat memicu berbagai reaksi fisik, kognitif, dan perilaku, termasuk peningkatan detak jantung, perubahan persepsi terhadap dunia sekitar, dan ekspresi verbal atau tindakan yang mengekspresikan ketidakpuasan (Setyawati & Sumekto, 2022).

2.6 Klasifikasi Teks

Klasifikasi teks merupakan salah satu teknologi dalam pengolahan bahasa alami yang paling sering digunakan (Cai et al., 2018). Klasifikasi teks tersusun atas dua kata yaitu klasifikasi dan teks. Teks dimaknai sebagai suatu naskah atau dokumen yang disampaikan dalam bentuk tulisan (Budiman, 2018), sedangkan klasifikasi diartikan sebagai prosedur pengelompokan atau pengkategorian objek ke dalam suatu kategori yang sudah didefinisikan sebelumnya (Bagus, 2022). Klasifikasi teks dapat diimplemtasikan pada beberapa tugas pemrosesan bahasa alami, seperti analisis sentimen, kategorisasi berita, klasifikasi topik, *question answering*, dan *natural language inference* (Minaee et al., 2020). Dalam pemrosesan bahasa alami, klasifikasi teks juga dikenal sebagai kategorisasi teks (Elnagar et al., 2020). Klasifikasi teks mempunyai tujuan untuk menganalisis, mengolah serta mengekstrak informasi yang terdapat dalam teks (Bagus, 2022). Dalam proses pengolahanya, klasifikasi teks memiliki

beberapa metode yang dapat diterapkan pada data, baik menggunakan pemodelan *machine learning* maupun pemodelan *deep learning*.

2.6.1 *K-Nearest Neighbor*

K-Nearest Neighbor atau KNN (Pamungkas & Kharisudin, 2021) merupakan bagian algoritma dari *machine learning* sederhana yang dapat digunakan untuk menyelesaikan masalah klasifikasi atau regresi (Pusporani et al., 2019). Algoritma ini termasuk ke dalam *supervised learning* yang berarti membutuhkan data latih. *K-Nearest Neighbor* bekerja dengan prinsip mencari jarak terdekat antara data yang sedang dievaluasi dan sejumlah k tetangga dalam data pelatihan (Devita et al., 2018). Besar atau kecilnya jarak antar lokasi dapat dihitung menggunakan salah satu metrik jarak yang telah ditentukan (Pusporani et al., 2019). Metode KNN bersifat *non-parametric* yang tidak membuat asumsi tentang distribusi data yang mendasarinya. Artinya, dalam model ini tidak ada parameter yang tetap, dan ini berlaku baik untuk data berukuran kecil maupun besar (Ahluna et al., 2023). KNN bersifat *lazy learning,* hal ini berarti bahwa suatu pendekatan pembelajaran di mana tidak ada proses pelatihan yang dilakukan, hanya nilai dari data pelatihan yang disimpan dan digunakan ketika melakukan prediksi (Putra et al., 2022). KNN dikenal memiliki tingkat efisiensi yang tinggi dan dalam beberapa situasi, serta tingkat akurasi yang tinggi dalam pengklasifikasian. Selain itu KNN juga termasuk model yang mudah diterapkan dan mudah beradaptasi (Pusporani et al., 2019). Namun, metode KNN memiliki beberapa kelemahan, seperti penggunaan seluruh fitur untuk menghitung jarak yang dapat menyebabkan beban komputasi menjadi berat dalam penerapannya, kemudian algoritma ini tidak bekerja dengan baik pada *dataset* berukuran besar, rentan terhadap gangguan data, nilai yang hilang, dan *outliers* (Suharno et al., 2017).

2.6.2 *Naïve Bayes*

Naïve Bayes merupakan algoritma klasifikasi statistik yang digunakan untuk memproyeksikan probabilitas sebuah objek masuk ke dalam suatu kelas tertentu (Ridwan, 2020). Algoritma *naïve bayes* memiliki dua tahapan utama, yakni tahap pelatihan dan tahap klasifikasi. Pada tahap pelatihan, dilakukan analisis pada sampel dokumen untuk memilih kata-kata yang mungkin muncul dalam koleksi dokumen tersebut. Selain itu, probabilitas untuk setiap kategori ditentukan berdasarkan sampel dokumen yang ada. Dalam proses klasifikasi, dokumen akan diberikan label kategori berdasarkan kata-kata yang ada dalam dokumen itu sendiri (Hemanto et al., 2020). Tipe data yang mendukung untuk pemodelan *naïve bayes* adalah

tipe data diskrit dan metode ini tidak cocok untuk atribut dengan nilai berkelanjutan numerik (*continuous*) (Ridwan, 2020). Meskipun termasuk algoritma yang sederhana, metode *naïve bayes* memiliki keunggulan dalam mencapai tingkat akurasi yang tinggi dalam perhitunganya dengan waktu cepat dan efisien (Pusporani et al., 2019; Sihombing, 2021). Kekurangan dari *naïve bayes* terjadi ketika probabilitas kondisional mencapai nilai nol, hasil prediksi juga akan menjadi nol. Kemudian *naïve bayes* memiliki anggapan bahwa setiap variabel independen dapat mempengaruhi penurunan akurasi, dikarenakan umumnya terdapat korelasi antara satu variabel dengan variabel lainnya (Sihombing, 2021).

2.6.3 *Support Vector Machine*

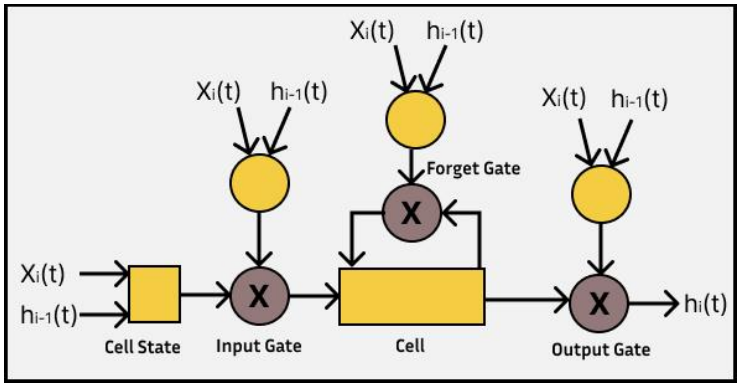
Support vector machine atau SVM merupakan metode yang berlaku baik untuk data yang memiliki hubungan linier maupun nonlinier. Algoritma SVM termasuk dalam kategori *supervised learning* karena memiliki tujuan pembelajaran yang ditargetkan (Hemanto et al., 2020). Tujuan utama dari metode ini adalah untuk membuat *Optimal Separating Hyperplane* (OSH), yang akan menghasilkan fungsi pemisahan yang optimal untuk keperluan klasifikasi (Pusporani et al., 2019). *Hyperplane* optimum merujuk pada *hyperplane* yang terletak di tengah antara kedua kelas sehingga memiliki jarak maksimal dengan data terluar dari kedua kelas tersebut (Pamungkas & Kharisudin, 2021). SVM merupakan algoritma yang relative mudah diimplementasikan. Selain itu, meskipun dilatih dengan *dataset* yang relatif kecil dan menggunakan pengaturan parameter yang sederhana, SVM mampu menghasilkan model klasifikasi yang baik. SVM memiliki konsep dan formulasi yang jelas dengan sedikit parameter yang perlu diatur (Hemanto et al., 2020). Namun, dibalik kelebihan yang dimiliki SVM, algoritma ini memiliki kekurangan, seperti sulit diterapkan pada *dataset* yang memiliki jumlah sampel dan dimensi yang sangat besar. Kemudian pada umumnya, SVM dirancang untuk menangani masalah klasifikasi dua kelas. Meskipun ada upaya untuk mengadaptasi SVM untuk masalah klasifikasi multi kelas, setiap strategi SVM yang digunakan untuk klasifikasi multi kelas memiliki kelemahan tersendiri (Handayani, 2021).

2.6.4 *Long Short-Term Memory*

LSTM (*Long Short-Term Memory*) merupakan salah satu algoritma *deep learning* yang terkenal dan sangat efektif dalam membuat prediksi dan melakukan klasifikasi (Prasetyanwar & Jondri, 2018). Algoritma LSTM adalah hasil perkembangan dari algoritma RNN (*Recurrent Neural Network*) (Zahara et al., 2019). Struktur algoritma LSTM terdiri dari jaringan saraf dan

beberapa komponen memori yang berbeda, yang sering disebut sebagai sel (Adiarso et al., 2019). Pada Gambar 2.1 yang berbentuk persegi panjang berwarna abu-abu muda merupakan ilustrasi sel pada LSTM. Algoritma LSTM mengumpulkan informasi yang akan disimpan dalam sel dan mengelolanya melalui komponen yang disebut *gate*. LSTM memiliki tiga *gate*, yaitu *forget gate, input gate*, dan *output gate* (Putra et al., 2021). Kelebihan dari algoritma *Long Short-Term Memory* (LSTM) adalah memiliki memori sel yang panjang dan dapat melakukan pembacaan informasi satu arah. Algoritma ini lebih baik digunakan untuk kasus memori pendek karena di dalam LSTM terdapat modifikasi formula pada memori internal (Maliki et al., 2022). LSTM memiliki kekurangan seperti komputasi yang lebih rumit dibutuhkan dibandingkan dengan RNN karena jumlah parameter yang lebih besar yang harus dipelajari. Selain itu, memori yang dibutuhkan juga lebih tinggi akibat keberadaan beberapa memori sel (Talita & Wiguna, 2019).

Dalam LSTM, sel berfungsi sebagai wadah untuk menyalurkan informasi dari satu lokasi ke lokasi lainnya, sebagaimana ditunjukkan pada Gambar 2.1. Selanjutnya, komponen sel memori melibatkan *input gate* yang berfungsi sebagai pengontrol untuk menentukan informasi yang harus diarahkan ke dalam sel memori untuk diperbarui (Dirgantara, 2022). *Forget gate* mengatur penolakan informasi dengan mengendalikan sinyal berulang dari sel, diikuti oleh *output gate* yang menentukan hasil keluaran yang akan dihasilkan (Arief et al., 2023).

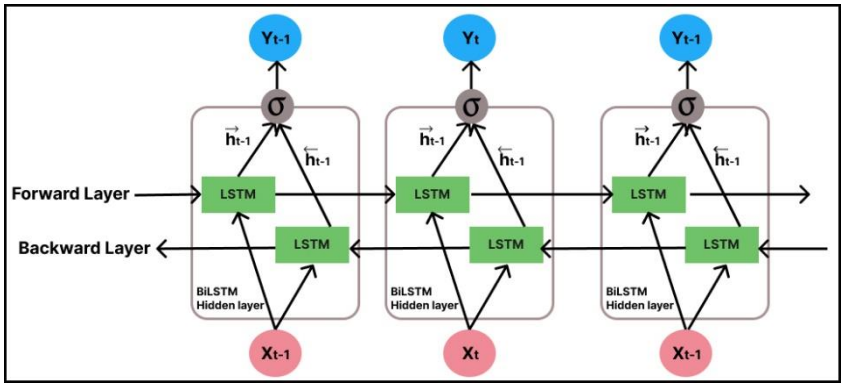


Gambar 2.1 Arsitektur LSTM

2.6.5 *Bidirectional Long-Short Term Memory*

Algoritma *Bidirectional Long Short-Term Memory* (BiLSTM) merupakan pengembangan dari algoritma *Long Short-Term Memory* (LSTM) (Wiharto et al., 2021). Pada *Bidirectional Long Short-Term Memory* (BiLSTM) dilakukan dua kali pembacaan informasi, hal ini akan memperkaya pemahaman dan konteks terhadap teks yang sedang diproses (Yunanto et al.,

2021). Pada proses BiLSTM, Satu aliran adalah urutan input konvensional, sedangkan aliran kedua adalah urutan input yang dihadapkan secara terbalik (Naquitasia, 2021). Informasi dari dua arah pemrosesan digabungkan untuk menghasilkan representasi data sekuensial yang lebih mendalam dan kontekstual. Biasanya, penggabungan ini terjadi pada tahap akhir sebelum mengambil *output* (Fitrinanda & Djunaidy, 2022). Penjelasan tersebut diilustrasikan pada Gambar 2.2.



Gambar 2.2 Arsitektur BiLSTM

Oleh karena itu, hal ini akan meningkatkan konteks dalam jaringan, menghasilkan peningkatan kecepatan dan efektivitas pembelajaran (Nugroho et al., 2021). Algoritma ini memiliki kelebihan untuk mengatasi masalah menghilangnya atau meledaknya gradien, memahami konteks ke depan dan ke belakang sehingga mampu meningkatkan akurasi. Namun Bi-LSTM juga memiliki kekurangan, seperti kompleksitas komputasi, ini dapat mengakibatkan waktu pelatihan yang lebih lama dan lebih banyak sumber daya komputasi yang dibutuhkan. Untuk menjalankan model Bi-LSTM dibutuhkan beberapa lapisan yang membangun arsitektur model seperti:

1. ***Embedding Layer***

   Lapisan ini merupakan komponen penting untuk mengolah data yang berbentuk teks atau kata (Sani & Sarwani, 2022). Lapisan *embedding* memiliki fungsi untuk mengonversi representasi diskrit kata-kata menjadi vektor kontinu berdimensi rendah, di mana setiap dimensi vektor merepresentasikan aspek-aspek semantik atau makna kata-kata tersebut (Prabowo et al., 2019). *Embedding layer* memungkinkan jaringan saraf untuk memiliki pemahaman yang lebih mendalam mengenai hubungan antara kata-kata dalam suatu teks, sehingga mampu meningkatkan kemampuan proses pembelajaran dan generalisasi dalam berbagai tugas berorientasi teks.

2. *BiLSTM Layer*

Proses klasifikasi pada penelitian ini menggunakan model *Bidirectional Long Short-Term Memory* (BiLSTM). Model BiLSTM merupakan model yang mengkombinasikan perhitungan LSTM *forward* dan LSTM *backward*. Algoritma *Long Short-Term Memory* (LSTM) merupakan pengembangan dari arsitektur *Recurrent Neural Network* (RNN) (Nugroho et al., 2021). Hal ini terjadi karena arsitektur *Recurrent Neural Network* (RNN) memiliki kekurangan dalam mengatasi *vanishing gradient,* di mana ketika data sekuensial dengan panjang yang signifikan diproses, gradien fungsi kerugian menurun secara eksponensial. Situasi ini mengakibatkan RNN tidak mampu menangkap ketergantungan jangka panjang dan akhirnya mempengaruhi hasil pelatihan yang akan berdampak pada rendahnya akurasi dan performa model dalam memprediksi. LSTM memperbarui pendekatan lapisan RNN dengan memasukkan modul memori berupa sel memori yang memiliki elemen gerbang, mencakup *forget gate, input gate*, dan *output gate.*

Forget gate berperan sebagai langkah awal dalam *Long Short-Term Memory* (LSTM) untuk mengatur keputusan mengenai informasi yang harus diingat atau dihilangkan dari *cell state* (Nugroho et al., 2021). Pada gerbang ini proses menerima masukan atau *input* h_{t-1} dan X_t untuk menghasilkan nilai 0 atau nilai 1 pada C_{t-1} seperti yang dijelaskan pada persamaan (2.1). Ketika *forget gate* bernilai 1, maka otomatis *cell state* akan menyimpan informasi, tetapi ketika bernilai 0 maka informasi yang didapatkan akan dibuang dari *cell state*. Oleh karena itu, meningkatkan bias b_f pada *forget gate* dapat meningkatkan kualitas kinerja pada model *Long Short-Term Memory* (LSTM).

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, X_t] + b_f\right) \tag{2.1}$$

Input gate merupakan tahap kedua dalam LSTM yang bertugas memutuskan elemen mana yang akan dianggap penting dan disimpan dalam *cell state*. Komponen ini terdiri dari lapisan s*igmoid* dan *lapisan tanh*. Lapisan *sigmoid* bertugas untuk mengatur dan memutuskan nilai-nilai mana yang akan mengalami pembaruan, sebagaimana dijelaskan dalam persamaan (2.2) yang telah diuraikan. Pada lapisan tanh dilakukan penambahan nilai baru yaitu C_t yang guna nya untuk ditambahkan ke *cell state* seperti yang dituliskan di persamaan (2.3). Setelah itu keluaran dari kedua lapisan ini akan digabungkan yang fungsinya untuk memperbarui *cell state*.

$$i_t = \sigma\,(W_i \cdot [h_{t-1}, X_t] + b_i) \tag{2.2}$$

$$C_t = tanh\,(W_c \cdot [h_{t-1}, X_t] + b_c) \tag{2.3}$$

Setelah kedua keluaran digabungkan menjadi *cell state*, langkah berikutnya adalah melakukan pembaruan pada *cell state* tersebut. Pembaruan tersebut dilakukan pada nilai *cell state* lama C_{t-1} menjadi C_t dengan mengalikan *cell state* lama dengan f_t yang bertujuan untuk menghapus nilai pada *forget gate* sebelumnya. Proses selanjutnya adalah dengan menambahkan nilai baru yang ditulis dengan $i_t C_t$, penjelasan tersebut tertulis pada persamaan (2.4).

$$C_t = f_t \cdot C_{t-1} + i_t \cdot C_t \tag{2.4}$$

Output gate merupakan tahap akhir dalam arsitektur LSTM yang memiliki peran penting dalam menentukan bagian mana dari *cell state* yang akan dijadikan *output*. Dengan menggunakan lapisan *sigmoid* seperti pada persamaan (2.5), *output gate* secara efektif mengontrol bagaimana *cell state* akan dihasilkan sebagai *output* akhir. Langkah berikutnya melibatkan penggunaan *output* tersebut sebagai masukan ke dalam lapisan *tanh* seperti pada persamaan (2.6). Kemudian, lapisan *sigmoid* digunakan kembali untuk mengatur sejauh mana *output* ini akan dipertahankan, sehingga *output* akhir yang sesuai dengan keputusan sebelumnya terbentuk, sesuai dengan penjelasan yang telah diberikan sebelumnya.

$$O_t = \sigma\,(W_o \cdot [h_{t-1}, X_t] + b_o) \tag{2.5}$$

$$h_t = O_t \cdot tanh(C_t) \tag{2.6}$$

Keterbatasan yang ada pada LSTM adalah kurangnya kemampuan untuk secara efektif menangkap dan mempertimbangkan informasi kontekstual dari kata-kata terakhir, karena fokusnya hanya pada pemrosesan sekuensial dari awal hingga akhir (Pramunendar et al., 2022). Berdasarkan pemaparan penjelasan diatas, penelitian ini akan menggunakan algoritma *Bidirectional Long Short-Term Memory* (BiLSTM). Pada algoritma *Bidirectional Long Short-Term Memory* (BiLSTM) pembacaan informasi akan dilakukan secara dua arah. Dalam hal teknis, implementasi BiLSTM melibatkan penggunaan dua unit LSTM yang bekerja terpisah, satu untuk arah ke depan dan yang lainnya untuk arah ke belakang. Penjelasan tersebut tertulis pada persamaan (2.7).

$$h_t^{BiLSTM} = h_t^{forward} + h_t^{backward} \quad (2.7)$$

3. *Dense Layer*

Dense layer menjadi sangat penting karena kemampuannya dalam menjalankan transformasi yang kompleks pada data. Apabila suatu model memiliki beberapa *dense layer*, model tersebut memiliki kapasitas untuk mempelajari representasi data yang lebih kompleks dan abstrak seiring dengan kedalaman *layer* tersebut. Dalam konteks tugas klasifikasi teks atau bidang lainnya, *dense layer* seringkali berperan sebagai lapisan terakhir sebelum *output layer*, yang menghasilkan prediksi akhir dari model. Dalam *dense layer*, terdapat sejumlah besar unit atau *neuron*, dengan setiap neuron dihubungkan baik dengan *neuron* di *layer* sebelumnya maupun *layer* selanjutnya. Fungsi utama dari *dense layer* adalah mengintegrasikan informasi yang berasal dari *layer* sebelumnya dan menjalankannya melalui serangkaian operasi linier maupun nonlinier. Hal ini memungkinkan model untuk mengembangkan pemahaman yang lebih mendalam terhadap hubungan yang kompleks dalam data.

4. *Dropout Layer*

Dropout layer adalah elemen kunci dalam struktur jaringan saraf yang diciptakan untuk mengatasi *overfitting* dalam model yang kompleks. *Overfitting* terjadi ketika model memiliki performa yang baik pada data pelatihan namun gagal dalam generalisasi yang memadai untuk data yang belum pernah terlihat sebelumnya. *Dropout layer* bertujuan untuk mengurangi *overfitting* dengan cara acak mengabaikan sejumlah unit atau *neuron* di lapisan tertentu selama proses pelatihan. Cara kerja *dropout* melibatkan penghentian sementara sebagian unit atau *neuron* di lapisan yang ditentukan pada setiap iterasi pelatihan secara acak. Ini diterapkan melalui probabilitas *dropout*, yang mengatur seberapa sering *neuron* tertentu akan dinonaktifkan selama pelatihan. Pada setiap iterasi, unit yang dinonaktifkan tidak menerima masukan atau berkontribusi dalam perhitungan dan pembelajaran, sehingga jaringan belajar untuk tidak terlalu bergantung pada setiap *neuron* secara spesifik.

Selain menggunakan lapisan *dropout* model klasifikasi juga menggunakan lapisan *spatial dropout. Spatial dropout* merupakan strategi pengaturan yang diterapkan dalam jaringan saraf konvolusional (CNN) dengan tujuan mengatasi masalah *overfitting*. *Overfitting* terjadi ketika model sangat memahami data pelatihan, sehingga mengalami

kesulitan dalam menggeneralisasi dengan baik pada data baru yang belum pernah dilihat sebelumnya. *Spatial dropout* adalah variasi dari metode *dropout* yang lebih umum. Penggunaan *spatial dropout* membantu dalam mengurangi *overfitting* pada jaringan saraf konvolusional dengan cara yang lebih efektif daripada *dropout* konvensional. Ini dapat meningkatkan kinerja model dan membantu dalam pencapaian generalisasi yang lebih baik pada data uji (Marinda & Al Amin, 2023).

2.7 Evaluasi

Pengujian model dilaksanakan dengan penerapan matriks konfusi yang bertujuan untuk mengkaji kinerja dan tingkat akurasi dari model yang telah dikembangkan sebelumnya. Dengan memanfaatkan matriks konfusi, evaluasi dilakukan untuk menganalisis sejauh mana model mampu mengklasifikasikan data dengan tepat dan membandingkan hasil prediksi dengan keadaan sebenarnya. Dengan demikian, dapat diperoleh gambaran yang lebih terperinci mengenai kemampuan serta efektivitas model dalam mengatasi tugas klasifikasi. Performa model dapat dievaluasi melalui pengamatan nilai variabel TP (*True Positive*) dan TN (*True Negative*), yang mewakili jumlah prediksi yang benar yang dihasilkan oleh model. Sebagai kontras, variabel FP (*False Positive*) dan FN (*False Negative*) mencerminkan total prediksi yang keliru yang telah dihasilkan oleh model. Dengan mengacu pada variabel-variabel ini, dapat diketahui sejauh mana model mampu mengidentifikasi dan memisahkan kelas target dengan akurat serta mengukur seberapa baik model menghindari kesalahan dalam klasifikasi data.

2.7.1 *Accuracy*

Accuracy merupakan ukuran seberapa dekat nilai prediksi dengan nilai sebenarnya, dan digunakan sebagai metrik untuk mengevaluasi kinerja model. *Accuracy* merupakan salah satu metrik evaluasi yang digunakan untuk mengukur kinerja suatu model atau sistem prediksi. Metrik ini menghitung sejauh mana model berhasil dalam memprediksi dengan benar dibandingkan dengan total jumlah data yang ada. Secara matematis, *accuracy* dihitung dengan membagi jumlah prediksi benar (*True Positive* dan *True Negative*) dengan total jumlah data. Penjelasan di atas dapat dituliskan dalam persamaan (2.8).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.8}$$

2.7.2 *Precision*

Precision adalah matrik evaluasi yang mengukur sejauh mana prediksi positif yang dibuat oleh suatu model adalah benar. Dalam konteks klasifikasi, *precision* mengukur proporsi dari prediksi positif yang benar terhadap total prediksi positif yang dilakukan oleh model. Secara lebih rinci, *precision* dihitung dengan membagi jumlah *True Positive* (prediksi positif yang benar) dengan total jumlah prediksi positif yang dibuat oleh model, yakni *True Positive* ditambah *False Positive* (prediksi positif yang salah). Penjelasan di atas dapat dituliskan dalam persamaan (2.9).

$$Precision \ = \frac{TP}{TP + FP} \tag{2.9}$$

2.7.3 *Recall*

Recall, juga dikenal sebagai sensitivitas, adalah metrik evaluasi yang mengukur sejauh mana model mampu mengidentifikasi dan mengambil sampel dari *instance* positif yang sebenarnya. Dalam konteks klasifikasi, *recall* mengukur proporsi dari *instance* positif yang benar teridentifikasi oleh model terhadap total jumlah *instance* positif yang sebenarnya. Secara lebih rinci, *recall* dihitung dengan membagi jumlah *True Positive* (prediksi positif yang benar) dengan total jumlah *instance* positif yang sebenarnya, yakni *True Positive* ditambah *False Negative* (*instance* positif yang tidak teridentifikasi). Penjelasan di atas dapat dituliskan dalam persamaan (2.10).

$$Recall \ = \frac{TP}{TP+FN} \tag{2.10}$$

2.7.4 *F1- Score*

F1-Score adalah sebuah metrik evaluasi yang menggabungkan informasi dari *precision* dan *recall* menjadi satu angka yang merefleksikan keseimbangan antara keduanya. Metrik ini berguna ketika ingin mencari keseimbangan antara mengoptimalkan presisi dan mengoptimalkan *recall* dalam model klasifikasi. Penjelasan di atas dapat dituliskan dalam persamaan (2.11).

$$F1 - Score \ = \frac{2 \ x \ Precision \ x \ Recall}{Precision+Recall} \tag{2.11}$$

Keterangan:

True positive (TP): Memprediksi kelas positif dengan benar.

True Negative (TN): Memprediksi kelas negatif dengan benar.

False Positive (FP): Salah memprediksi kelas positif.
False Negative (FN): Salah memprediksi kelas negatif.

2.8 Penelitian Terdahulu

Pencarian literatur pada penelitian terdahulu ini dilakukan dengan menggunakan Google Scholar, Research Gate, Science Direct, IEEE Xplore, dan Academia Edu. Rentang tahun yang digunakan dalam pencarian ini antara tahun 2017-2023. Pencarian literatur ini menggunakan kata kunci 'klasifikasi teks', 'klasifikasi teks + *machine learning*', 'klasifikasi teks + *deep learning'*, 'klasifikasi masalah', '*human problem classification*', '*problem classification*', dan 'kalsifikasi teks + BiLSTM'. Pencarian literatur digunakan untuk mengidentifikasi tingkat keakuratan klasifikasi teks menggunakan metode yang berbeda-beda. Hasil dari identifikasi tersebut akan digunakan sebagai acuan dalam penelitian.

Penelitian pertama dari Palma et al., (2021) melakukan peneltian mengenai klasifikasi teks pada artikel berita hoaks covid-19. Penelitian ini menggunakan algoritma *machine learning* KNN. Data yang digunakan untuk pemodelan berjumlah 9.517 data. Penelitian ini memiliki empat label utama, yaitu *fact, hoax, unknown*, dan *unimportant*. Kemudian terdapat empat label pembanding, yaitu *support, oppose, NEI*, dan *irrelevan.* Setelah dilakukan pemodelan, didapatkan akurasi sebesar 48% dengan komposisi pembagian data 80% data *train* dan 20% data *test* dengan nilai k = 5.

Mustofa et al., (2019) melakukan penelitian mengenai klasifikasi teks pada berita hoaks. Metode yang diterapkan menggunakan *Naïve Bayes*. Data berita hoaks penelitian diambil dari turnbackhoax.com dengan rentang pengambilan data dimulai dari November 2018 hingga Februari 2019. Total data yang diambil sebanyak 360 data yang selanjutnya dilabeli oleh pakar. Sebanyak 297 data berlabel hoaks dan 63 data berlabel fakta. Pemodelan menggunakan *Naïve Bayes* menghasilkan akurasi sebesar 85,28%.

Hemanto et al., (2020) melakukan penelitian mengenai kalsifikasi teks perbandingan metode *Support Vector Machine* dan *Naïve Bayes*. Penelitian ini menggunakan data yang diambil dari *database* sisfo akademik (*students.bsi.ac.id*). Jumlah keseluruhan data pada penelitian tersebut sebanyak 8.699 data. Kemudian data dibagi menjadi dua label, yaitu label *complaine* dan *notcomplaine*. Setelah dilakukan pemodelan, model SVM mendapatkan akurasi sebesar 84,45%, sedangkan metode *Naïve Bayes* mendapatkan akurasi sebesar 69,75%. Metode SVM mendaparkan akurasi lebih tinggi daripada metode *Naïve Bayes* dengan selisih akurasi sebesar 14,7%.

Pamungkas et al., (2021) melakukan penelitian mengenai sentimen analisis terhadap tanggapan masyarakat indonesia tentang pandemi Covid-19 dengan membandingkan tiga metode *machine learning*, yaitu SVM, *Naïve Bayes*, dan KNN. Penelitian ini menggunakan data sebanyak 10.000 data yang terbagi menjadi dua kelas, yaitu kelas sentimen positif sebanyak 6128 data dan sentimen negatif sebanyak 3872 data. Berdasarkan hasil pemodelan, didapatkan akurasi metode SVM sebesar 90,1%. Akurasi menggunakan metode *Naïve Bayes* sebesar 79,2%. Kemudian pada metode terakhir, yaitu KNN mendapatkan akurasi sebesar 62,1%. Pada penelitian ini metode SVM lebih unggul daripada model Naïve Bayes dan KNN.

Fadli et al., (2021) melakukan penelitian mengenai identifikasi *cyber bullying* pada media sosial Twitter. Metode yang digunakan untuk klasifikasi pada penelitian ini adalah algoritma *Bidirectional Long Short-Term Memory* (BiLSTM) dan algoritma *Long Short-Term Memory* (LSTM). Pada penelitian ini menggunakan data sebanyak 6835 cuitan twitter yang terbagi menjadi dua kelas yaitu *non-cyber bullying* dan *cyberbullying*. Setelah dilakukan pemodelan didapatkan hasil akurasi model LSTM sebesar 93,77 dan model BiLSTM sebesar 95,24. Berdasarkan hasil dari penelitian ini, model BiLSTM memiliki nilai akurasi yang lebih tinggi dari LSTM.

Nugroho et al., (2021) melakukan penelitian mengenai deteksi depresi dan kecemasan pengguna pada Twitter. Metode klasifikasi teks yang digunakan adalah *Bidirectional Long Short-Term Memory* (BiLSTM). Data yang digunakan berbahasa Indonesia dengan jumlah 2.751 kicauan. Pada data tersebut dibagi menjadi dua kelas yaitu kelas label 1 mengindikasikan jika *tweet* memiliki potensi kecemasan, kegelisahan, atau depresi dengan jumlah 894 kicauan. Kemudian Label 0 untuk keadaan sebaliknya dengan jumlah 1.857 kicauan. Penelitian ini melakukan perbandingan antara dua metode yaitu *Bidirectional Long Short-Term Memory* (BiLSTM) dan *Long Short-Term Memory* (LSTM). Parameter yang digunakan antara lain *embedding_size* 200, *activation sigmoid*, *optimizer adam*, *batch size* 64, dan *learning rate* sebesar 1e-3. Berdasarkan hasil pelatihan didapatkan hasil akurasi *Long Short-Term Memory* (LSTM) sebesar 0,8491 dan akurasi *Bidirectional Long Short-Term Memory* (BiLSTM) sebesar 0,9412. berdasarkan hasil pemodelan model *Bidirectional Long Short-Term Memory* (BiLSTM) mendapatkan akurasi lebih tinggi dari model *Long Short-Term Memory* (LSTM).

Amalia et al., (2022) melakukan penelitian mengenai klasifikasi berita palsu Menggunakan *Bidirectional Long Short-Term Memory* (BiLSTM). Data yang digunakan pada penelitian ini menggunakan data penelitian dari Sheng How Kong et al., (2020). Pada data tersebut terdapat dua *dataset* yang berisi mengenai teks berita berbahasa nggris dengan dua kategori kelas yaitu

real dan *fake*. Penelitian ini melakukan tiga kali percobaan dengan *embedding layer* dan vektorisasi yang berbeda. Akurasi tertinggi didapatkan sebesar 92,80 dengan 200 *output dimension*, 68/1000 *input length*, 256 *units* Bi-LSTM *layer*, serta menggunakan aktivasi sigmoid.

Af'idah et al., (2022) melakukan penelitian mengenai sentimen analisis pada destinasi wisata Pulau Bali. Metode klasifikasi teks yang digunakan adalah *Bidirectional Long Short-Term Memory* (BiLSTM). Data yang digunakan pada penelitian diambil melalui web tripadvisor.com dengan cara *scraping*. Proses pengambilan dilakukan pada bulan Februari 2020. Data terbagi menjadi dua kelas yaitu kelas positif sebanyak 5000 data dan kelas negatif sebanyak 5000 data. Pengujian dilakukan dengan tiga skenario. Akurasi terbaik diperoleh sebesar 96,86 dengan skenario pengujian menggunakan 200 dimensi *Word2Vec*, 0,5 nilai *dropout*, dan 0,0001 nilai *learning rate*.

Dirgantara, (2022) melakukan penelitian mengenai deteksi kalimat menggunakan metode *Bidirectional Long Short-Term Memory* (BiLSTM). Penelitian ini mengambil studi kasus bahasa Indonesia dan Malaysia. Penelitian ini menggunakan *dataset* sebesar 41.317 kalimat yang didapatkan dari surat kabar yang berasal dari Indonesia dan Malaysia. Sebelum memasuki proses pemodelan, *dataset* dibagi menjadi dua yaitu data latih sebesar 75% dan data uji sebesar 25%. Setelah dilakukan pelatihan model didapatkan hasil akurasi sebesar 98%. Adapun rekap dari penelitian terdahulu yang dibahas di atas ditunjukkan pada Tabel 2.1.

Tabel 2.1 Penelitian Terdahulu

Peneliti	**Bahasan**	**Dataset**	**Jumlah Data**	**Metode**	**Akurasi**
(Palma et al., 2021b)	Klasifikasi Teks Artikel Berita Hoaks Covid-19 dengan Menggunakan Algotrima KNearest Neighbor	Artikel berita	9.517	KNN	48%
(Mustofa & Mahfudh, 2019)	Klasifikasi Berita Hoax Dengan Menggunakan Metode Naive Bayes	turnbackhoax.com	367	*Naïve Bayes*	85.28 %
(Hemanto et al., 2020b)	Algoritma Klasifikasi *Naïve Bayes* dan *Support Vector Machine* dalam Layanan Komplain Mahasiswa	students.bsi.ac.id	8.699	*Naïve Bayes*	62,1%
				SVM	84,45%
(Pamungkas &	Analisis Sentimen dengan SVM, NAIVE	Twitter	10.000	SVM	90,1%

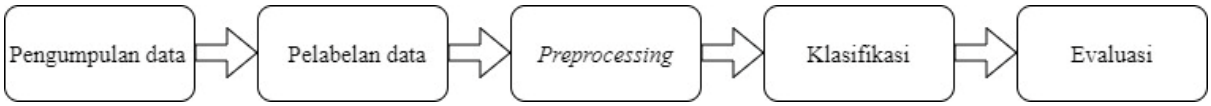
Kharisudin, 2021)	BAYES dan KNN untuk Studi Tanggapan Masyarakat Indonesia Terhadap Pandemi Covid-19 pada Media Sosial Twitter			*Naïve Bayes*	79,2%
				KNN	62,1%
(Fadli & Hidayatullah, 2021)	Identifikasi Cyberbullying pada Media Sosial Twitter Menggunakan Metode LSTM dan BiLSTM	Twitter	6.835	LSTM	93,77%
				BiLSTM	95,24%
(Nugroho et al., 2021)	Deteksi Depresi dan Kecemasan Pengguna Twitter Menggunakan Bidirectional LSTM	Twitter	2.751	LSTM	0,849%
				BiLSTM	0.941%
(Amalia et al., 2022)	Model Klasifikasi Berita Palsu Menggunakan *Bidirectional LSTM* dan *Word2Vec* Sebagai Vektorisasi	Dataset Sheng How Kong et al.	-	BiLSTM	92,80%
(Af'idah et al., 2022)	Sentimen Ulasan Destinasi Wisata Pulau Bali Menggunakan *Bidirectional Long Short-Term Memory*	tripadvisor.com	5.000	BiLSTM	96,86%
(Dirgantara, 2022)	Deteksi Kalimat Menggunakan Metode Bidirectional LSTM Studi Kasus: Indoensia dan Malaysia	Surat Kabar Indonesia dan Malayasia	41.317	BiLSTM	98%

Hasil pemaparan pada Tabel 2.1 menunjukkan bahwa, akurasi yang dihasilkan dari pemodelan menggunakan *deep learning* seperti pada LSTM dan BiLSTM. Nilai akurasi yang didapat yaitu diatas 0,849%. Nilai akurasi tersebut lebih tinggi jika dibandingkan dengan nilai akurasi yang didapatkan oleh *mechine learning* seperti pada metode KNN, SVM, dan *Naïve Bayes*. Sehingga, penerapan metode *deep learning* dinilai lebih unggul daripada pendekatan *machine learning* (Hasanah et al., 2021). Metode BiLSTM meraih nilai akurasi tertinggi diantara metode lain, yaitu diatas 92,80%. Berdasarkan hasil perbandingan motode tersebut, penelitian ini akan menggunakan menggunakan metode *deep learning*, BiLSTM. Metode BiLSTM dipilih karena kemampuannya dalam memahami konteks dari kata berurutan. Selain itu, pada BiLSTM dilakukan dua kali pembacaan informasi, satu aliran mengikuti urutan input konvensional, sementara aliran kedua mengikuti urutan input yang dibalik. Hal ini akan memperkaya pemahaman dan konteks terhadap teks yang sedang diproses (Yunanto et al., 2021).

BAB III
METODOLOGI PENELITIAN

3.1 Tahapan Penelitian

Penelitian ini memiliki beberapa tahapan yang harus dilalui sebagaimana ditunjukkan pada Gambar 3.1. Tahapan pertama dimulai dari proses pengumpulan data. Setelah data terkumpul kemudian dilanjutkan dengan proses pelabelan data yang di dalam proses tersebut juga terdapat proses penghapusan duplikasi data. Kemudian setelah data selesai dilabeli, data akan dikelompokkan menjadi enam kategori. Data yang sudah melewati proses pelabelan akan melanjutkan ke proses selanjutnya yaitu, *preprocessing* atau proses mengubah data yang berantakan menjadi data siap pakai. Selanjutnya dilakukan proses klasifikasi atau pemodelan menggunakan model *Bidirectional Long Short-Term Memory* (BiLSTM). Proses terakhir dalam penelitian ini adalah mengevaluasi model yang telah dibangun.



Gambar 3.1 Tahapan Penelitian

3.2 Pengumpulan Data

Data yang dikumpulkan pada penelitian ini berasal dari Komunitas Marah-Marah di Twitter. Data diambil melalui teknik *scraping*. Proses *scraping* adalah suatu cara atau teknik untuk mengekstrak informasi yang berada dalam sebuah *website*. Pengumpulan data dilakukan dengan menggunakan bahasa Python. Dalam proses pengambilan data melibatkan *library* dalam Python yang digunakan untuk automasi pada *browser* yaitu *library Selenium*. Penggunaan *Selenium* memungkinkan interaksi langsung dengan antarmuka web, termasuk mengklik tombol, menggulir halaman, dan mengekstrak teks. Proses pengumpulan dimulai dengan mengonfigurasi *Selenium* untuk membuka peramban web, melakukan *login* ke akun Twitter, dan mengambil postingan yang ada pada Komunitas Marah-Marah. Data yang ada dalam Komunitas Marah-Marah di Twitter jenisnya beragam, mulai dari data teks, data gambar, dan data video. Pada penelitian ini, data yang dikumpulkan untuk melakukan pemodelan adalah data yang berupa teks. Hasil dari pengumpulan data ini menghasilkan beragam twit yang menggunakan beragam diposting oleh anggota Komunitas Marah-Marah. Kumpulan data-data teks tersebut disimpan dalam format *Comma Separated Value* (CSV).

3.3 Pelabelan Data

Sebelum dilakukan pelabelan, data yang sudah tersimpan dalam format *Comma Separated Value* (CSV) dianalisis untuk mengetahui apakah terdapat duplikasi data atau tidak. Proses menghapus duplikasi data yang ada pada *dataset* twit Komunitas Marah-Marah merupakan suatu langkah yang penting. Duplikasi data terkadang ditemukan ketika melakukan pengambilan data menggunakan proses *scraping*. Duplikasi data dapat mengakibatkan bias dalam hal analisis data dan mengganggu kinerja model yang akan dibangun. Proses ini dimulai dengan mengidentifikasi dan membandingkan setiap entri data dalam *dataset* untuk menemukan kemungkinan adanya duplikasi. Proses identifikasi dilakukan berdasarkan isi teks dari setiap entri. Jika dua atau lebih entri memiliki teks yang sama persis, maka dianggap sebagai duplikat. Setelah duplikasi diidentifikasi, langkah selanjutnya adalah menghapus salah satu dari entri yang duplikat tersebut. Proses tersebut dilakukan sampai tidak terdapat duplikasi data pada *dataset* tersebut. Setelah data bersih dari duplikasi dilanjutkan untuk proses selanjutnya yaitu pelabelan data.

Pelabelan data adalah tahap yang sangat penting dalam jalannya penelitian ini, terutama dalam konteks penelitian klasifikasi masalah. Pada dasarnya, pelabelan data adalah proses memberikan label atau kategori tertentu kepada setiap contoh data dalam kumpulan data. Dalam konteks penelitian ini, pelabelan data menjadi fundamental karena bertujuan untuk mengidentifikasi setiap postingan dari Komunitas Marah-Marah di Twitter. Label yang digunakan pada penelitian ini berdasarkan penilaian dari Dosen Psikologi Universitas Islam Indonesia yaitu, Ibu Dr. Rina Mulyati, S.Psi., M.Si., Psikolog. Label tersebut terbagi menjadi enam kategori permasalahan yaitu, Studi, Percintaan, Keluarga, Karier/Pekerjaan, Person/Personal, dan Tidak Diketahui Masalahnya seperti yang penjelasannya ditunjukkan pada Tabel 3.1. Pelabelan ini dilakukan secara manual dengan cara membaca dan menganalisis konten dari setiap postingan untuk menentukan setiap konten cuitan *tweet* masuk pada kategori permasalahan apa. Hal ini dilakukan karena pelabelan data yang akurat dan konsisten merupakan langkah penting untuk memastikan bahwa model pembelajaran mesin dapat belajar dengan benar dan menghasilkan hasil yang valid dalam analisis selanjutnya. Pentingnya pelabelan data yang baik adalah untuk memastikan bahwa *dataset* yang digunakan dalam analisis memiliki informasi yang jelas dan valid, serta dapat mendukung pengembangan model klasifikasi yang efektif.

Tabel 3.1 Kategori Label

No	Kategori	Penjelasan
1	Studi	Permasalahan studi merujuk pada berbagai isu atau tantangan yang berkaitan dengan lingkup akademik atau pendidikan. Permasalahan studi mengacu pada berbagai masalah yang dihadapi oleh individu atau kelompok dalam lingkungan pendidikan, seperti mahasiswa, pelajar, atau peserta didik.
		Data twit yang masuk ke dalam label Studi adalah twit yang memiliki kata kunci seperti 'tugas', 'dosen', 'skripsi', 'uts', 'uas', 'kuliah', 'SKS', 'IPK', 'guru', 'jam kosong' dan lain sebagainya. Contoh pelabelan Studi seperti pada twit 'gamau kuliah besok plis males banget yaallah'. Pada twit tersebut terdapat kata kunci berupa 'kulliah', di mana konteks kata tersebut berkaitan dengan kata kunci pada label Studi dan berkaitan dengan pembelajaran akademik.
2	Percintaan	Permasalahan yang berkaitan dengan perihal perasaan senang kepada orang lain jenis (kelamin) dan (rasa) cinta. Dalam konteks penelitian ini, permasalahan percintaan mencakup berbagai isu yang dapat mempengaruhi hubungan antara pasangan, seperti konflik interpersonal, komunikasi yang buruk, kepercayaan yang terganggu, kesalahpahaman, perbedaan nilai atau tujuan, dan masalah emosional dalam hubungan.
		Pada label percintaan terdapat beberapa kata kunci, seperti 'cowo', 'cewe', 'pacar', 'selingkuh', 'nikah', 'restu orang tua', 'ldr', 'kawin', 'cerai', 'putus' dan lain sebagainya. Contoh pelabelan Percintaan, seperti pada twit "tolong dong buat laki kalo emang mau deketin untuk ke ranah pacaran atau nikah ya ngomong aja di awal kan jadi ga buang buang waktu energi perasan dan materi". Konteks dari kalimat tersebut yaitu mengenai permasalahan percintaan. Terlihat terdapat kata kunci seperti 'pacaran', 'nikah', dan 'perasaan'.
3	Keluarga	Permasalahan yang berkaitan dengan lingkup keluarga, termasuk keluarga inti dan keluarga besar. Permasalahan keluarga juga dapat melibatkan isu seperti pengasuhan anak, perawatan orang tua, dan hubungan dengan anggota keluarga yang lebih luas.
		Label Keluarga memiliki beberapa kata kunci sebagai acuan untuk pelabelan, seperti kata 'orang tua', 'kakak', 'adek', 'nenek', 'bapak', 'ibu', 'pilih kasih', 'generasi sandwich', 'tulang punggung keluarga', 'beban keluarga', 'saudara', 'warisan', 'nafkah', dan lain sebagainya. Contoh pelabelan Keluarga seperti pada twit 'nasib lahir dri keluarga miskinnn' tergambar secara jelas bahwa konteks dari kalimat tersebut adalah permasalahan keluarga dengan kata kunci 'keluarga' yang terdapat dalam kalimat tersebut.
4	Karier/Pekerjaan	Permasalahan karier atau pekerjaan merujuk pada berbagai masalah atau tantangan yang terkait dengan aspek profesional dan pekerjaan seseorang. Permasalahan ini dapat meliputi hal-hal seperti ketidaksesuaian antara kemampuan dan tuntutan pekerjaan, stres kerja, tekanan waktu yang tinggi, gangguan keseimbangan kerja-hidup, kurangnya peluang pengembangan, konflik dengan rekan kerja atau atasan, dan ketidakjelasan tujuan karier.
		Kategori permasalahan Karier/Pekerjaan memiliki beberapa kata kunci, seperti 'resign', 'jobdesk', 'masalah', 'bos', 'atasan', 'lembur', 'gaji', 'lingkungan toxic', 'lempar kerjaan', 'Burnout', 'Produktivitas', 'Pelatihan kerja', dan lain sebagainya. Contoh pelabelan Karier/Keluarga seperti pada twit 'Gua mau resign kalo masalah udh

		selesaii Seminggu gua tahan emosi nangis grgr ini buka jobdesk gua dan gua juga tertekan'. Konteks dari permasalahan tersebut berkaitan mengenai pekerjaan. Pada kalimat tersebut terdapat beberapa kata kunci seperti 'resign', 'jobdesc', dan 'masalah'.
5	Person/Personal	Permasalahan yang berhubungan dengan diri sendiri atau personal, dan permasalahan yang berkaitan antar manusia atau orang lain. Permasalahan ini dapat meliputi hal-hal seperti masalah kesehatan mental, ketidakmampuan mengelola emosi, konflik internal dan eksternal , rendahnya rasa percaya diri, hambatan dalam meraih tujuan, dan ketidakpuasan terhadap diri sendiri maupun orang lain.
		Label Person/Personal memiliki beberapa kata kunci, seperti 'temen', 'dia', 'capek', 'dasar', 'butuh', 'hidup', 'manusia', 'kalian', dan lain sebagainya. Contoh permasalahan dari kategori Person (orang lain), 'Minimal lu bayar utang dulu lah kontol'. Konteks dari kalimat tersebut adalah permasalahan yang berkaitan dengan orang lain yaitu permasalahan hutang. Selain itu, contoh twit dengan permasalahan Person (diri sendiri) seperti 'GW KNP Y KLO MARAH BAWAANYA NANGIS MULU ANJENG'. Kalimat tersebut memberikan pesan secara ekplisit bahwa individu pemilik twit sedang mengalami permasalahan dengan dirinya sendiri. Permasalahan tersebut berkenaan dengan penyebab pemilik twit menangis ketika sedang merasakan amarah.
6	Tidak Diketahui Masalahnya	Label Tidak Diketahui Masalahnya hanya berisi kata-kata umpatan atau berkaitan dengan penggunaan kata-kata kasar. Disebut Tidak Diketahui Masalahnya karena pada twit label ini hanya berisi umpatan yang tidak menjelaskan secara eksplisit masalah apa yang sedang dialami oleh individu pemilik twit. Pada twit dengan label Tidak Diketahui Masalahnya tidak terdapat permasalahan spesifik yang jelas. Hanya berisi umpatan yang mencakup penggunaan kata-kata kasar.
		Label permasalahan Tidak Diketahui Masalahnya diberikan pada twit yang tidak diketahui permasalahannya. Twit yang dilabeli dengan Tidak Diketahui Masalahnya memiliki beberapa kata kunci, seperti seperti 'kontol', 'anjing', 'babi', 'ngentot', 'kimak', 'bangsat', 'goblok', dan lain sebagainya. Contoh twit label Tidak Diketahui Masalahnya seperti 'KONTOL AH ANJING NGENTOD', 'ANJINGGGGGGGG', 'TOLOL ANJING BANGSAT NGENTOT', 'hadeh jancok', dan 'wanjengg wanjeng'. Beberapa contoh twit tersebut hanya berisi kata-kata kasar dan umpatan, tanpa ada suatu permasalahan yang jelas.

3.4 *Text Preprocessing*

Gambar 3.2 Tahapan *Preprocessing*

Text preprocessing merupakan langkah untuk mengolah data awal menjadi data yang lebih teratur dan terstruktur (Sari & Prasetya, 2022). *Preprocessing* memiliki tujuan untuk mengubah data awal menjadi informasi yang lebih tepat dan berkualitas (Nugroho et al., 2021). Ketika

data dikumpulkan, sering kali data tersebut masih dalam kondisi tidak teratur dan tidak konsisten. Oleh karena itu, pada tahap ini, dilakukan langkah pembersihan data sebelum data tersebut dapat digunakan untuk melatih model. Pada *preprocessing*, komponen-komponen yang dianggap tidak memiliki makna yang penting akan dieliminasi dari data, yang mana bertujuan untuk meningkatkan konsistensi dan kebermaknaan data. Terdapat beragam jenis *preprocessing* yang dapat diadaptasi, namun hal tersebut perlu disesuaikan dengan kebutuhan *dataset* yang akan digunakan. *Preprocessing* pada penelitian ini menggunakan bantuan dari *library* Sastrawi. Sehingga twit yang digunakan untuk *preprocessing* adalah twit yang menggunakan bahasa Indonesia. Twit yang menggunakan bahasa selain bahasa Indonesia, seperti bahasa Sunda, bahasa Jawa, bahasa Inggris dan bahasa lainnya akan dihapus. Jika terdapat satu sampai tiga kata selain bahasa Indonesia dalam sebuah kalimat, maka twit tersebut akan tetap digunakan untuk *preprocessing.* Sampel kalimat campuran tersebut dapat dilihat pada Tabel 3.2. Contohnya pada kolom dengan kalimat 'Lu **online** tapi gak pernah bales, tai', terdapat kata yang mengandung bahasa Inggris, yaitu '*online*'. Untuk itu kalimat tersebut tetap dimasukkan kedalam dataset yang akan digunakan untuk *preprocessing*. Pada penelitian ini, akan menggunakan beberapa proses seperti *case folding*, *symbol,* and *whitespace removal*, tokenisasi, normalisasi data, dan *stemming*.

Tabel 3.2 Contoh Kalimat Bahasa Campuran

Contoh Kalimat	Bahasa
Lu **online** tapi gak pernah bales, tai	Bahasa Inggris
childish amat sih lu monyet	Bahasa Inggris
Fakyu first media jahanam	Bahasa Inggris
santai aja **atuh** kontol. lebay amat	Bahasa Sunda
Buset dah **aya aya wae** hidup	Bahasa Sunda
DJIANCOKKK AKU WES BOLAK BALIK **NDELOK** YUTUB MALAH GAK MENANG, MEMANG ASU	Bahasa Jawa
ngenteni sp sih jancok	Bahasa Jawa

Preprocessing pada penelitian ini tidak menggunakan penghapusan *stopword* karena dapat mengurangi tingkat akurasi pada data latih dan data uji. Penghapusan *stopwords* merupakan proses dalam pemrosesan teks yang dimaksudkan untuk mengenali dan menghapus kata-kata umum atau *stopwords* dari dokumen atau teks (Ulfah & Anam, 2020). *Stopwords* adalah kata-kata yang sering muncul dalam bahasa tertentu dan tidak memberikan banyak nilai informatif dalam analisis teks karena mereka muncul secara umum dalam berbagai konteks. Contoh *stopwords* dalam bahasa Indonesia termasuk kata-kata seperti "dan", "atau", "dari", "ke", dan

“yang”. Penghapusan *stopwords* dilakukan untuk menghapus teks dari kata-kata yang tidak relevan atau yang tidak memiliki makna penting dalam analisis teks, sehingga memungkinkan fokus pada kata-kata kunci yang lebih informatif dan relevan dalam pemrosesan dan analisis berikutnya.

Penghapusan *stopwords* terjadi selama tahap *preprocessing*, jika stopwords dihilangkan ketika pemodelan, hal tersebut dapat berpengaruh pada hasil klasifikasi (Fachreza, 2023) dan akurasi (Faisal et al., 2018). Meskipun demikian, ada situasi tertentu di mana menghilangkan *stopwords* dapat mengakibatkan penurunan tingkat akurasi (Christianto et al., 2020; Dwiyansaputra et al., 2021). Namun demikian, dalam penelitian ini telah dilakukan eksperimen sebelumnya, yang mana dilakukan penghapusan *stopwords* dengan menggunakan kamus yang diperoleh dari akun GitHub masdevid (url: https://bit.ly/kamus-stopwords). Proses penjelasan mengenai *preprocessing* dalam penelitian ini adalah sebagai berikut:

a. *Case folding*

Case folding sering disebut sebagai langkah pembakuan atau normalisasi (Fitriyana et al., 2023). Pada tahapan *case folding* dilakukan pengubahan semua komponen teks menjadi huruf kecil, semua huruf kapital seperti ‘A’-‘Z’ akan diganti menjadi huruf kecil seperti ‘a’-‘z’. Tujuan dari tahap ini adalah untuk memastikan keseragaman dalam penulisan karakter dan menghilangkan variasi besar-kecil huruf yang dapat mengganggu proses analisis teks yang lebih mendalam.

b. *Symbol & Whitespace Removal*

Proses penghilangan simbol mencakup langkah-langkah untuk menghapus karakter-karakter yang tidak relevan, seperti tanda baca, angka, dan karakter lainnya yang tidak diperlukan seperti (!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~) (Putra et al., 2021). Contoh tanda baca meliputi koma, titik, tanda tanya, dan sejenisnya umumnya tidak memiliki relevansi dalam memberikan makna teks secara semantik. Begitu pula, karakter khusus seperti simbol matematika atau emotikon cenderung tidak relevan untuk analisis yang sedang dilakukan. Oleh karena itu, dalam langkah ini, dilakukan penghilangan semua karakter tanda baca, angka, dan karakter khusus dari teks yang sedang diolah. Sementara itu, *whitespace removal* merupakan proses eliminasi spasi yang terletak di awal dan akhir kalimat (Arief et al., 2023). Adanya penambahan spasi berlebihan pada bagian awal dan akhir kalimat atau kata dapat mempengaruhi proses analisis teks. Contohnya, kelebihan spasi di awal kalimat dapat

menyebabkan pemisahan kata-kata yang seharusnya satu kesatuan, mengganggu analisis kata-kata yang akurat. Dalam langkah ini, spasi yang tidak perlu dihilangkan untuk memastikan bahwa teks memiliki struktur yang lebih teratur dan siap untuk langkah berikutnya dalam proses analisis.

c. Tokenisasi

Tokenisasi merupakan tahap pemisahan kata-kata menjadi token-token yang terbentuk dari suatu kalimat (Arief et al., 2023). Langkah tokenisasi melibatkan pemisahan kata-kata yang memiliki makna dalam suatu kalimat dengan menggunakan sejumlah simbol, termasuk tanda strip (-), tanda petik dua ("), dan garis bawah (_), sebagai pemisah antara kata-kata (Putra et al., 2021). Proses tokenisasi sangat penting dalam pemrosesan bahasa alami karena membantu mengurai teks menjadi bagian yang lebih terstruktur dan dapat dimengerti oleh komputer.

d. Normalisasi Kata

Normalisasi kata dilaksanakan dengan tujuan mengubah beberapa kata yang pada awalnya tertulis dalam bentuk singkatan atau bahasa nonformal agar beralih menjadi bentuk kata asal yang sesuai (Al-Shufi & Erfina, 2021). Proses normalisasi data dalam penelitian ini mengandalkan kamus yang diperoleh dari akun GitHub milik nasalsabila (url: https://bit.ly/kamus-alay) sebagai data awal. Kamus normalisasi tersebut tersimpan dalam format CSV dengan nama berkas colloquial-indonesian-lexicon.csv. Dalam penelitian ini, kamus alay yang terdapat di akun nasalsabila telah diambil sebagai dasar dan disesuaikan agar sesuai dengan kebutuhan *dataset* penelitian dan mengubah nama berkas tersebut menjadi normalisasi.csv. Contoh penambahan kata pada kamus normalisasi dapat dilihat pada Tabel 3.3. Penambahan ini dilakukan karena sebagai bentuk penyesuaian dari kamus pedoman yang berasal dari GitHub nasalsabila dan *dataset* yang berisi postingan twit yang digunakan pada penelitian ini. Selain itu ada beberapa kata seperti ‘bloggggg’, ‘bgsd’, ‘babik’, ‘asw’ dan lain sebagainya yang tidak terdapat pada kamus, sehingga dilakukan penambahan untuk menyesuaikan *dataset* yang akan digunakan pada penelitian ini. Berkas kamus alay yang digunakan pada penelitian ini bisa diakses menggunakan link (url: bit.ly/normalisasi-kata).

Tabel 3.3 Tambahan Normalisasi

Sebelum	Sesudah
bloggggg	Goblok
coggggg	Jancok

ln	Luar negeri
resp	Respon
bgsd	Bangsat
taikk	Tai
esmosiii	Emosi
gj	Tidak Jelas
asw	Asu
anjeeeeeeeeeeeeeeeeeennnggggggg	Anjing
babik	Babi

Pengubahan kata singkatan melibatkan identifikasi dan penggantian kata-kata yang merupakan singkatan dengan bentuk lengkap atau baku yang sesuai. Singkatan adalah pemendekan kata menjadi beberapa huruf yang sering digunakan dalam bahasa sehari-hari. Contohnya, "dr" untuk "dokter", "pk" untuk "pukul", dan "dll" untuk "dan lain-lain". Dalam tahap ini, setiap singkatan dalam teks akan diterjemahkan ke bentuk lengkapnya, sehingga menghasilkan data yang lebih konsisten dan mudah dimengerti. Kemudian penggunaan slang dan bahasa gaul sering digunakan dalam komunikasi digital atau informal. Dalam tahap ini, kata-kata slang dan bahasa gaul diubah menjadi bentuk baku atau lebih umum. Misalnya, kata "asyik" dapat diubah menjadi "menyenangkan", "mantap" menjadi "bagus", atau "ngeh" menjadi "tahu". Ini membantu memastikan bahwa data teks memiliki makna yang lebih jelas dan konsisten, sehingga lebih mudah untuk dianalisis secara lebih lanjut.

e. *Stemming*

Stemming merupakan pemrosesan bahasa alami (NLP) yang digunakan untuk mereduksi kata dalam teks menjadi bentuk dasar atau akar kata (*stem*) (Syadid, 2019). Tujuannya adalah untuk menghilangkan infleksi kata, akhiran, dan awalan sehingga kata-kata yang berbeda tetapi berasal dari akar kata yang sama dianggap identik. Ini membantu dalam analisis teks, temuan informasi, dan pemahaman konten dalam NLP. Sebagai contoh, jika dalam sebuah kalimat memiliki kata-kata seperti "bermain", "bermainan", dan "bermainnya" dengan proses *stemming*, semuanya akan disederhanakan menjadi akar kata yang sama, yaitu "main.". Dengan cara ini, dapat mengelompokkan kata-kata ini bersama-sama dan mengurangi kerumitan dalam pemrosesan teks. Proses *stemmning* dilakukan dengan menggunakan fungsi *stemmer* dari *library* Sastrawi untuk mengembalikan kata ke bentuk dasarnya, dengan kamus *stemming* dari akun GitHub har07 (url: https://bit.ly/kamus-stemming). Fungsi `stemmer.stem()` pada *library* Sastrawi cenderung berjalan lambat. Maka dari itu, perlu dibantu menggunakan *library* Swifter dari Python untuk mempercepat proses *stemming* pada *dataframe* dengan menjalankan *task* secara paralel.

3.5 Klasifikasi

Salah satu kategori dalam *data mining* yang digunakan untuk mengelompokkan objek adalah proses klasifikasi (Norhikmah & Rumini, 2020). Data twit yang diklasifikasikan adalah data postingan twit dan tidak mengikutsertakan data twit komentar. Proses klasifikasi dilakukan dengan dua skenario berbeda. Skenario pertama yaitu klasifikasi dengan *preprocessing* menggunakan penghapusan *stopwords* dan skenario kedua yaitu klasifikasi dengan *preprocessin*g tanpa menggunakan penghapusan *stopwords*. S*topwords* merupakan kata-kata umum dalam bahasa alami yang seringkali tidak memberikan makna khusus dalam konteks analisis teks dan biasanya diabaikan atau dihapus untuk tujuan pemrosesan teks. Dua skenario tersebut masing-masing akan melewati empat kali percobaan,

1. Dua percobaan menggunakan pembagian data 80:20 (80% data latih dan 20% data uji) dengan menggunakan percobaan tiga nilai *batch size* yaitu 32, 64, 128 dan dua nilai *spatial dropout* yaitu 0,4 dan 0,5.
2. Dua percobaan sisanya akan menggunakan pembagian data 90:10 (90% data latih dan 10% data uji) dengan menggunakan tiga nilai *batch size* sebagai percobaan yaitu 32, 64, 128 dan dua nilai *spatial dropout* yaitu 0,4 dan 0,5.

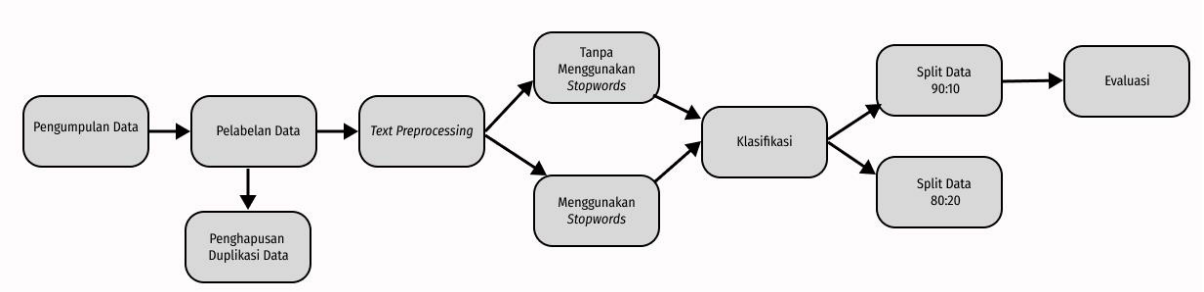
Klasifikasi pada penelitian ini menggunakan algoritma *Bidirectional Long Short-Term Memory* (BiLSTM) dengan menggunakan beberapa lapisan seperti *embedding layer*, BiLSTM *layer, dense layer,* dan *dropout layer*.

3.6 Evaluasi

Evaluasi ini menggunakan data uji sebesar 10% yang telah ditentukan sebelumnya. Data uji tersebut dievaluasi menggunakan *confussion matrix* untuk mengetahui seberapa bagus kinerja model yang sudah dibangun. Berdasarkan hasil evaluasi akan diperoleh nilai seperti nilai akurasi, presisi, *recall*, dan *F1-score*. Hal tersebut membantu mengenali jenis kesalahan yang dibuat oleh model dan memberikan pandangan yang lebih mendalam tentang kinerja model dalam mengklasifikasikan berbagai kategori.

BAB IV
HASIL & PEMBAHASAN

Detail tahapan penelitian yang dilakukan pada penelitian ini ditunjukkan pada Gambar 4.1. Tahapan pertama dalam penelitian ini dimulai dengan proses pengumpulan data menggunakan bahasa pemrograman Pyhton. Setelah berhasil mengumpulkan dan menyimpan data dalam format CSV, tahap berikutnya adalah memberi label pada data. Sebelum dilakukan pelabelan data, *dataset* akan melalui proses penghapusan duplikasi data. *Dataset* yang sudah melewati tahapan pelabelan data akan dilanjutkan ke tahapan *preprocessing text*, di mana proses ini dilakukan sebanyak dua kali. *Preprocessing* pertama dilakukan tanpa menggunakan penghapusan *stopwords*, *preprocessing* kedua dengan menggunakan *stopwords*. Data yang sudah bersih akan dilanjutkan ke tahapan klasifikasi. Pada tahapan ini *dataset* dibagi menjadi dua skenario pembagian data. Skenario pertama dataset dibagi menjadi 90% untuk data latih dan 10% untuk data uji. Skenario kedua dengan membagi dataset menjadi 80% untuk data latih dan 20% untuk data uji. Pada kedua skenario ini, hanya skenario pertama dengan pembagian data 90:10 dengan *preprocessing* tanpa penghapusan *stopwords* yang dilanjutkan ke tahap evaluasi karena memiliki hasil nilai akurasi yang lebih tinggi. Tahapan evaluasi merupakan tahapan penilaian di mana model akan dinilai seberapa bagus performanya dalam melakukan klasifikasi.

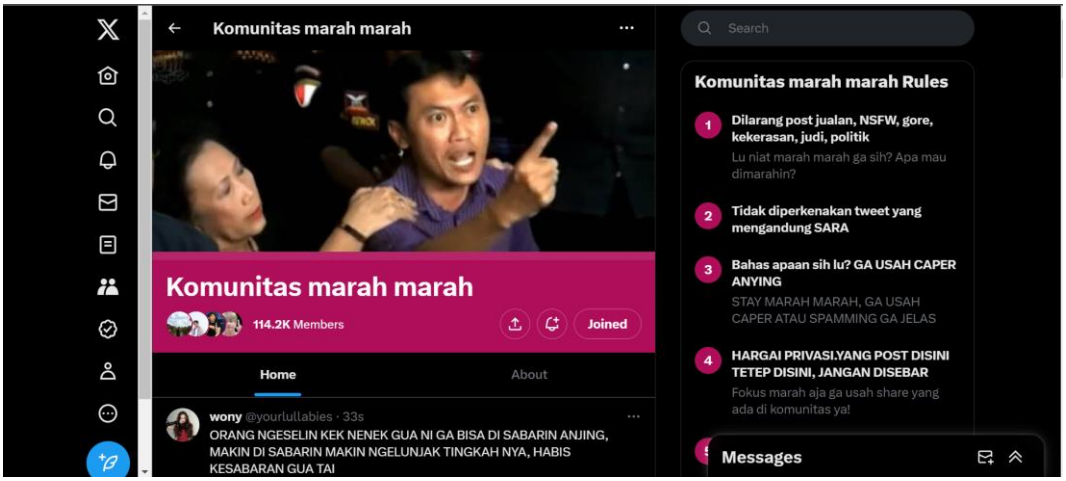


Gambar 4.1 Tahapan Penelitian

4.1 Pengumpulan Data

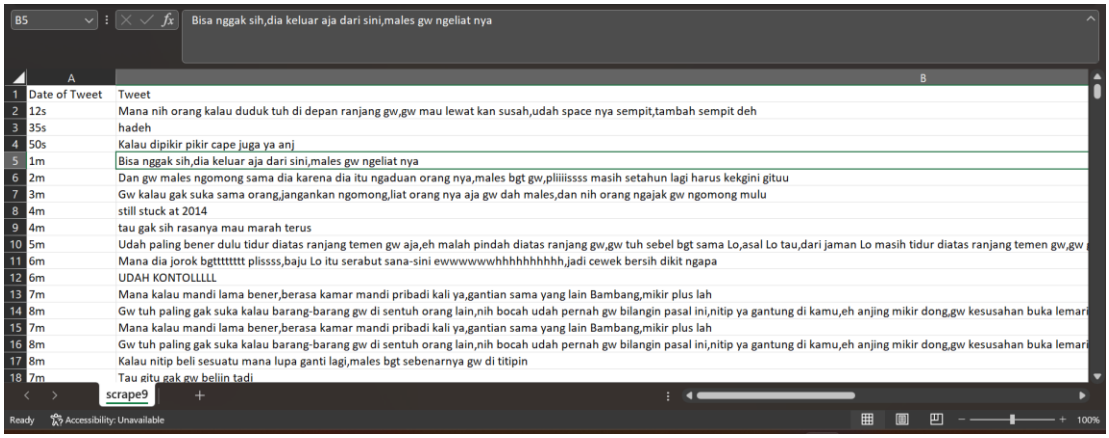
Pengumpulan data dalam penelitian ini dilakukan dengan menggunakan teknik *scraping*. Data yang akan diambil berasal dari Komunitas Marah-Marah di Twitter sebagaimana ditunjukkan pada Gambar 4.2. Untuk menunjang proses *scraping,* pada penelitian ini menggunakan teks editor *visual code studio*, dengan bahasa pemrograman yang digunakan

berupa Python. Proses automasi web dilakukan dengan menggunakan *library* dari bahasa pemrograman Python yaitu *Selenium*.



Gambar 4.2 Komunitas Marah-Marah

Dengan pemanfaatan *Selenium*, interaksi langsung dengan antarmuka web menjadi lebih mudah dilakukan. Ini melibatkan sejumlah tindakan seperti mengklik elemen, melakukan guliran halaman, dan mengekstrak teks dari konten web secara efisien. Penelitian ini hanya mengambil dua informasi dari Komunitas Marah-Marah di Twitter, yaitu informasi waktu pengunggahan *tweet* dan isi *tweet* atau teks, seperti pada Gambar 4.3.



Gambar 4.3 Hasil *Scraping*

Proses pengambilan data dilakukan dalam empat periode waktu. Periode pertama dilakukan pengambilan pada tanggal 1 Maret 2023 sampai 4 Maret 2023, diperoleh data sebanyak 2.576 data. Selanjutnya pada periode kedua pengumpulan data dimulai pada tanggal

11 Maret 2023 sampai 14 Maret 2023, yang mana diperoleh data sebanyak 12.050 data. Periode ketiga dilakukan pengambilan data pada tanggal 14 Maret 2023 sampai 15 Maret 2023, data yang berhasil dikumpulkan hanya 790 data. Kemudian pada periode terakhir dilakukan pengambilan data pada tanggal 14 Juli 2023 sampai 17 Juli 2023, berhasil dikumpulkan data sejumlah 14.080. Rekap periode pengambilan data di Komunitas Marah-Marah Twitter dapat dilihat pada Tabel 4.1.

Tabel 4.1 Periode Pengambilan Data

Periode	Tanggal	Jumlah
1	1 Maret 2023 - 4 Maret 2023	2576
2	11 Maret 2023 - 14 Maret 2023	12.050
3	14 Maret 2023 - 15 Maret 2023	790
4	14 Juli 2023 - 17 Juli 2023	14,080
	Jumlah	29.496

Proses pengambilan data dari Komunitas Marah-Marah di Twitter menggunakan Python dan *library Selenium* melibatkan serangkaian langkah yang terstruktur dan otomatis. Pertama, kode program perlu mengimpor *library Selenium* dan mengatur *driver* browser yang akan digunakan, yaitu *Chrome*. Kemudian, program membuka jendela browser dengan *Chrome driver* mengarahkannya ke halaman *login* Twitter untuk masuk menggunakan *username* dan *password*. Setelah berhasil masuk ke Twitter, program akan mengarahkan untuk masuk ke *link* yang dituju yaitu Komunitas Marah-Marah di Twitter. Kode program ditunjukkan pada Gambar 4.4.

```
# Set driver and initial array
Driver=webdriver.Chrome(executable_path=r"D:\chromeDriver\chromedriver.exe)
# navigate to login screen
driver.get('https://www.twitter.com/login')
driver.maximize_window()
# Input username
username = driver.find_element_by_xpath('//input[@name="text"]')
username.send_keys('username')
username.send_keys(Keys.RETURN)
# Input Password
password = driver.find_element_by_xpath('//input[@name="password"]')
password.send_keys('password')
password.send_keys(Keys.RETURN)
driver.get("https://twitter.com/i/communities/1562271278744354816")
```

Gambar 4.4 Kode program navigasi menuju Komunitas Marah-Marah

Selanjutnya, kode akan mengidentifikasi elemen-elemen penting dalam halaman, seperti tombol "*Load More*" untuk memuat lebih banyak postingan atau daftar postingan itu sendiri. Dengan menggunakan metode yang disediakan oleh *Selenium*, program dapat melakukan tindakan-tindakan seperti mengklik tombol, menggulir halaman, atau menginteraksi dengan elemen-elemen lainnya. Setelah konten tambahan dimuat atau postingan ditampilkan, kode akan mengekstrak informasi yang diinginkan dari setiap postingan. Ini dapat mencakup teks postingan, tanggal, pengguna yang membuat postingan, dan informasi lainnya. Proses ini dilakukan dengan mencari elemen-elemen HTML yang mengandung data tersebut, dan kemudian mengekstrak teks atau atribut yang sesuai. Kode program ditunjukkan pada Gambar 4.6

```
# Get scroll height after first time page load
last_height = driver.execute_script("return document.body.scrollHeight")
last_elem=''
current_elem=''
while True:
    # Scroll down to bottom
    driver.execute_script("window.scrollTo(0,document.body.scrollHeight);")
    # Wait to load page
    time.sleep(6)
    # Calculate new scroll height and compare with last scroll height
    new_height = driver.execute_script("return document.body.scrollHeight")
    if new_height == last_height:
        break
    last_height = new_height
```

Gambar 4.5 Kode Program *Scrolling*

Setelah postingan ditampilkan, kode akan mengekstrak informasi yang diinginkan dari setiap postingan. Pada penelitian ini mencakup teks postingan dan tanggal pengunggahan. Proses ini dilakukan dengan mencari elemen-elemen HTML yang mengandung data tersebut, dan kemudian mengekstrak teks atau atribut yang sesuai. Kode program ditunjukkan pada Gambar 4.6

```
#update all_tweets to keep loop
all_tweets =
driver.find_elements(By.XPATH,'//div[@datatestid]//article[@data-
testid="tweet"]')
    for item in all_tweets[1:]: # skip tweet already scrapped
        print('--- date ---')
        try:
            date = item.find_element(By.XPATH, './/time').text
        except:
            date = '[empty]'
        print(date)
```

```
        print('--- text ---')
        try:
            text = item.find_element(By.XPATH,
'.//div[@datatestid="tweetText"]').text
        except:
            text = '[empty]'
        print(text)
```

Gambar 4.6 Kode Program Pengambilan Data

Setelah berhasil mengekstrak informasi yang diinginkan dari setiap postingan di Komunitas Marah-Marah di Twitter menggunakan Selenium, langkah selanjutnya adalah melakukan penyimpanan data dalam format CSV. Data yang disimpan dalam format CSV dapat dengan mudah diakses dan dimanfaatkan untuk analisis lebih lanjut, seperti membangun model klasifikasi menggunakan metode *deep learning*. Kode program seperti pada Gambar 4.7.

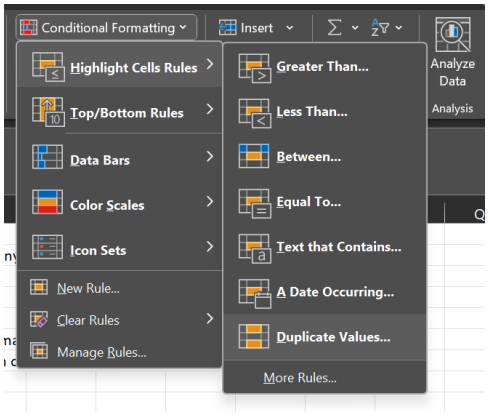
```
df = pd.DataFrame(tweets, columns=['Date of Tweet','Tweet'])
df.to_csv('scrape3.csv', index=False, encoding='utf-8')
```

Gambar 4.7 Kode Program Penyimpanan CSV

4.2 Pelabelan Data

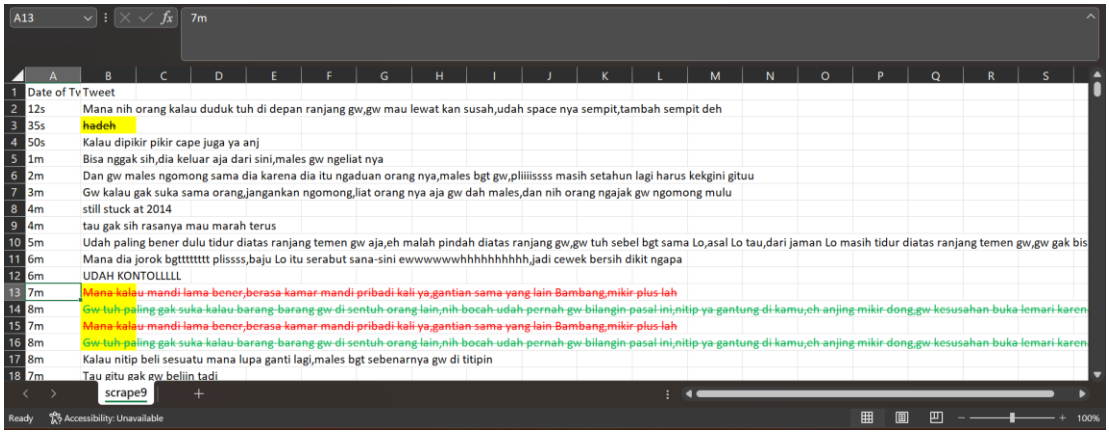
Tahap pelabelan data umumnya dilaksanakan setelah *preprocessing*. Namun, dalam penelitian ini, langkah pelabelan data dilakukan setelah data terkumpul. Pendekatan ini diambil untuk menghindari potensi ketidakvalidan dalam pemberian label kepada data. Dengan melakukan pelabelan setelah pengumpulan data, lebih memastikan bahwa setiap label yang diberikan sesuai dengan karakteristik dan konteks masing-masing data, sehingga hasil analisis dan evaluasi pada model klasifikasi menjadi lebih akurat dan dapat diandalkan. Pelabelan pada penelitian ini dilakukan secara manual, sehingga kemungkinan terjadinya kesalahan pelabelan atau *human error* sangat besar.

Sebelum dilakukan tahap pelabelan, data yang tersimpan dalam format CSV dianalisis menggunakan *Excel* untuk mengetahui apakah terjadi duplikasi data atau tidak. *Conditional formatting* digunakan untuk mengetahui apakah terdapat duplikasi dalam *dataset*. Sebelum melakukan proses penghilangan duplikasi data, dilakukan pemblokan kolom *Tweet*. Fitur *duplicates values* yang digunakan untuk mengetahui duplikasi dalam data terdapat dalam fitur *Highlight Cells Rules* yang ditunjukkan pada Gambar 4.8.



Gambar 4.8 *Conditional Formatting*

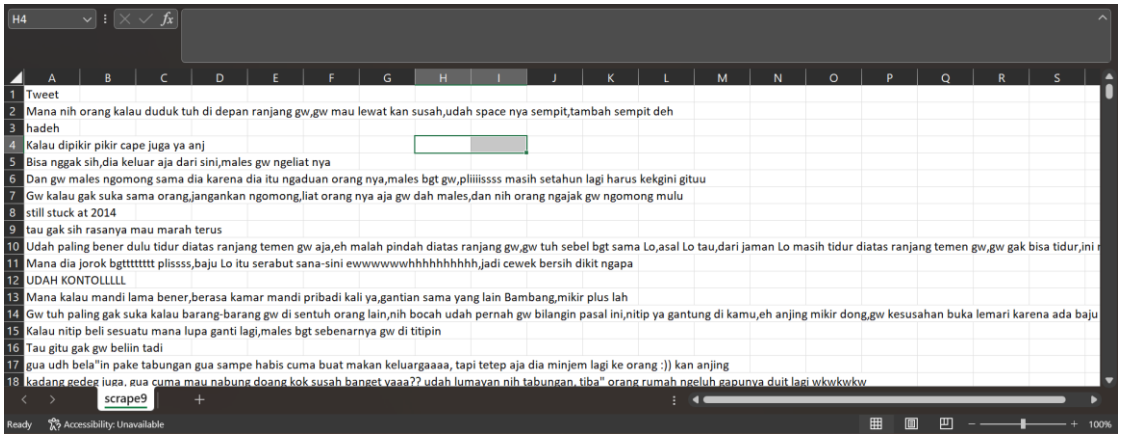
Untuk menandai data yang duplikat, diberikan warna merah dan hijau seperti pada Gambar 4.9. Sebagai contoh, pada Gambar 4.9 baris 13 duplikat dengan data pada baris 15. Kemudian baris 14 duplikat dengan data pada baris 16. Untuk menghilangkan duplikasi tersebut, perlu adanya penghapusan pada salah satu data. Dengan menghapus duplikasi data, dapat dipastikan bahwa setiap entri dalam *dataset* hanya diwakili satu kali, sehingga analisis dan model yang dihasilkan lebih mewakili keragaman dan distribusi yang sebenarnya dari data.



Gambar 4.9 Duplikasi Data dalam *Dataset*

Gambar 4.10 menunjukkan tidak adanya tulisan berwarna merah dan hijau. Hal tersebut mengindikasikan bahwa duplikasi data sudah dihilangkan. Berdasarkan hasil penghapusan duplikasi data dan penyesuaian kebutuhan untuk penelitian, didapatkan data sebanyak 17.276 data. Selain dilakukan penghapusan duplikasi, kolom pada dataset berupa '*Date of Tweet'* juga dihapus. Hal itu dilakukan karena menghindari adanya hal-hal yang tidak penting dalam proses

pelatihan data nantinya. Informasi mengenai tanggal pada *tweet* tidak akan digunakan pada penelitian ini.



Gambar 4.10 Hasil Penghapusan Duplikasi Data

Setelah dilakukan penghapusan duplikasi data, proses berikutnya adalah tahap pelabelan data. Pelabelan data dilakukan secara manual dengan menganalisis, mengidentifikasi setiap *tweet* yang ada dalam *dataset*. Penelitian ini memiliki enam kategori kelas yang sudah didefinisikan sebelumnya, yang mana penentuan jenis kategori kelas tersebut dibantu oleh pakar, yaitu Ibu Dr. Rina Mulyati, S.Psi., M.Si., Psikolog. Kategori kelas dalam penelitian ini diantaranya adalah kelas Studi, Percintaan, Keluarga, Karier/Pekerjaan, Person/Personal, dan Tidak Diketahui Masalahnya. Hasil dari pelabelan data dapat dilihat pada Tabel 4.2. Pelabelan data dalam penelitian ini telah melewati proses validasi acak oleh Ibu Dr. Rina Mulyati, S.Psi., M.Si., Psikolog.

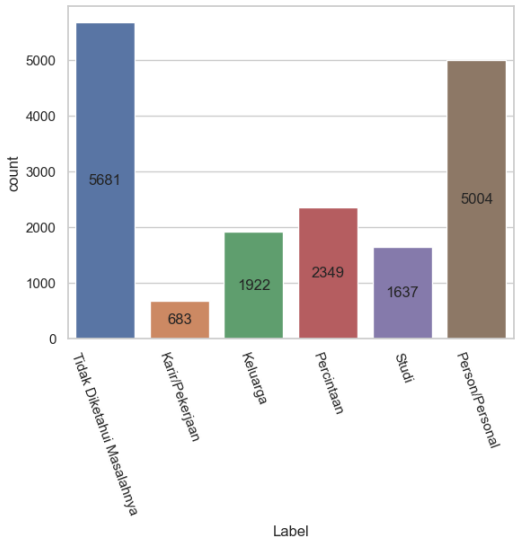
Tabel 4.2 Hasil Pelabelan Data

Tweet	Label
PUNYA ORANGTUA GOBLOKK TOLOLLL UDAH PADA TUA MASIH AJA RIBUT HAL SEPELE GOBLOKKKK	Keluarga
AH KONTOL	Tidak Diketahui Masalahnya
Sebenernya gue kesel karna mesti ke tempat kerja cuma buat ngambil dokumen doang Arrrghhhh gue takut disuruh ngerjain yang lain	Karier/Pekerjaan
sudah saatnya kita menangisi bab 2 alias anjing iki kapan rampunge satttttt	Studi
Udah bener bener jomblo malah cinta cintaan lagi anjng bangsat	Percintaan
Ngechat temen berasa ngechat artis, lama bgt jawabnyaa	Person/Personal

Jumlah keseluruhan data pada penelitian ini adalah 17.276 data. Perbandingan jumlah data setelah dilakukan penghapusan duplikasi dan sebelum dilakukan penghapusan duplikasi dapat dilihat pada Tabel 4.3. Di antara enam kategori, label Tidak Diketahui Masalahnya memiliki jumlah yang lebih banyak daripada label lainnya yaitu sejumlah 5.681. Posisi kedua untuk label terbanyak ada pada kategori Person/Personal dengan jumlah data sebanyak 5.004. Kemudian label Percintaan berada di posisi terbanyak ketiga dengan jumlah data sebanyak 2.349. Label Keluarga dengan jumlah data sebanyak 1.922 berada di posisi keempat diantara label-label lainnya. Setelah label Keluarga, ada label Studi yang berada di posisi kelima dengan jumlah data 1.637. Posisi terakhir dengan jumlah data yang paling sedikit ditempati oleh label Karier/Pekerjaan dengan jumlah data di bawah seribu, yaitu 683. Jumlah data per label dapat dilihat pada Gambar 4.11.

Tabel 4.3 Data penghapusan duplikasi

Sebelum Penghapusan Duplikasi	**Setelah Penghapusan Duplikasi**
29.496	17.276
	Selisih Jumlah: 12.220



Gambar 4.11 Total Jumlah Label dalam Setiap Kategori

Untuk mengetahui gambaran kata-kata yang sering muncul dalam data, maka dilakukan visualisasi *wordcloud*. *Wordcloud* merupakan representasi visual dari data teks yang mengilustrasikan seberapa sering kata-kata muncul dalam teks dengan cara yang menarik dan mudah dipahami. Dalam sebuah *wordcloud*, penempatan dan ukuran kata-kata mencerminkan

frekuensi kemunculan kata-kata tersebut dalam teks. Kata-kata yang muncul lebih sering akan ditampilkan lebih besar dan mungkin diberi warna yang mencolok, sementara kata-kata yang muncul lebih jarang akan ditampilkan lebih kecil. *Wordcloud* sering digunakan sebagai alat visualisasi untuk memberikan gambaran singkat tentang tema atau isu yang dominan dalam dataset teks. Kode program ditunjukkan pada Gambar 4.12.

```
normal_words =' '.join([text for text in df['Text']])
wordcloud = WordCloud(width=800, height=500, random_state=21,
max_font_size=110).generate(normal_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show
```

Gambar 4.12 Kode program *Wordcloud Dataset*

Berikut adalah hasil visualisasi untuk semua label yang dapat dilihat pada Gambar 4.13. Visualisasi ini memberikan gambaran komprehensif tentang distribusi kata-kata dalam *dataset* berdasarkan label-label yang ada. Pada visualisasi gabungan ini representasi kata-kata yang ditampilkan sangat beragam. Namun frekuensi kata-kata kasar sangat mendominasi pada visualisasi gabungan ini seperti kata ‘anjing’, ‘kontol’, dan ‘ngentot’.

Gambar 4.13 *Wordcloud Dataset*

Setelah mendapatkan hasil visualisasi *wordcloud* untuk seluruh label, tahap berikutnya adalah membuat visualisasi *wordcloud* secara terpisah untuk setiap label. Tujuan dari langkah ini adalah untuk mendalami dan memahami lebih jauh kata-kata yang muncul dalam setiap label klasifikasi secara terpisah, sehingga dapat memberikan wawasan yang lebih detail tentang isi teks dalam masing-masing kategori. Gambar 4.15 menunjukkan kode program yang digunakan untuk membuat visualisasi *wordcloud* label Studi. Sedangkan Gambar 4.14

merupakan hasil visualisasi dari kode program. Representasi kata-kata yang muncul pada label Studi kebanyakan mengenai kata-kata yang merepresentasikan kegiatan pembelajaran seperti kata ‘tugas’, ‘dosen’, ‘kuliah’, kelompok, dan ‘nilai’.

```
normal_words =' '.join([text for text in df['Text'][df['Label'] == 'Studi']])
wordcloud=WordCloud(width=800,height=500,random_state=21,max_font_size=110)
.generate(normal_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()
```

Gambar 4.14 Kode Program Label Studi

Gambar 4.15 *Wordcloud* Label Studi

Gambar 4.16 menunjukkan kode program yang digunakan untuk membuat visualisasi *wordcloud* label Percintaan. Sedangkan Gambar 4.17 merupakan hasil visualisasi dari kode program. Representasi kata-kata yang muncul pada label Percintaan kebanyakan mengenai kata-kata yang merepresentasikan kegiatan asmara seperti kata ‘pacar’, ‘cowo’, ‘mantan’, dan ‘cewe’.

```
normal_words =' '.join([text for text in df['Text'][df['Label'] ==
'Percintaan']])
wordcloud = WordCloud(width=800, height=500, random_state=21,
max_font_size=110).generate(normal_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()
```

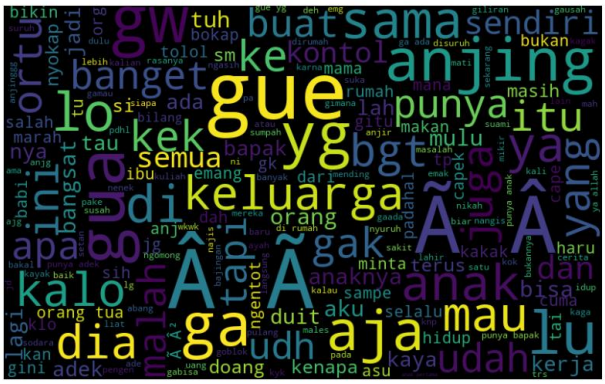
Gambar 4.16 Kode Program Label Percintaan

Gambar 4.17 Wordcloud Label Percintaan

Gambar 4.18 menunjukkan kode program yang digunakan untuk membuat visualisasi *wordcloud* label Keluarga. Sedangkan Gambar 4.19 merupakan hasil visualisasi dari kode program. Representasi kata-kata yang muncul pada label Keluarga kebanyakan mengenai kata-kata yang merepresentasikan permasalahan dan hubungan keluarga seperti kata ‘ibu’, ‘sodara’, ‘adek’, ‘bapak’, dan ‘nenek’.

```
normal_words =' '.join([text for text in df['Text'][df['Label'] ==
'Keluarga']])
wordcloud = WordCloud(width=800, height=500, random_state=21,
max_font_size=110).generate(normal_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()
```

Gambar 4.18 Kode Program Label Keluarga



Gambar 4.19 *Wordcloud* Label Keluarga

Gambar 4.20 menunjukkan kode program yang digunakan untuk membuat visualisasi *wordcloud* label Karier/Pekerjaan. Sedangkan Gambar 4.21 merupakan hasil visualisasi dari

kode program. Representasi kata-kata yang muncul pada label Karier/Pekerjaan kebanyakan mengenai kata-kata yang merepresentasikan tentang kegiatan dalam pekerjaan dan urusan mengenai jenjang karir seperti kata ‘kantor’, ‘kerja’, ‘kerjaan’, ‘karyawan’, ‘bos’, dan ‘libur’.

```
normal_words =' '.join([text for text in df['Text'][df['Label'] ==
'Karir/Pekerjaan']])
wordcloud = WordCloud(width=800, height=500, random_state=21,
max_font_size=110).generate(normal_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()
```

Gambar 4.20 Kode Program Label Karier/Pekerjaan

Gambar 4.21 Wordcloud Label Karier/Pekerjaan

Gambar 4.22 menunjukkan kode program yang digunakan untuk membuat visualisasi *wordcloud* label Person/Personal. Sedangkan Gambar 4.23 merupakan hasil visualisasi dari kode program. Hasil visualisasi menunjukkan bahwa dalam label Person/Personal memiliki kata-kata yang lebih beragam. Representasi kata-kata yang muncul pada label Person/Personal kebanyakan mengenai kata-kata yang merepresentasikan permasalahan internal dan eksternal seperti kata ‘aku’, ‘temen’, ‘masalah’, ‘cape, dan ‘gila’.

```
normal_words =' '.join([text for text in df['Text'][df['Label'] ==
'Person/Personal']])
wordcloud = WordCloud(width=800, height=500, random_state=21,
max_font_size=110).generate(normal_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()
```

Gambar 4.22 Kode Program Label Person/Personal

Gambar 4.23 *Wordcloud* Label Person/Personal

Gambar 4.24 menunjukkan kode program yang digunakan untuk membuat visualisasi *wordcloud* label Tidak Diketahui Masalahnya. Sedangkan Gambar 4.25 merupakan hasil visualisasi dari kode program. Representasi kata-kata yang muncul pada label Tidak Diketahui Masalahnya kebanyakan mengenai kata-kata yang merepresentasikan ungkapan yang tidak senonoh seperti kata ‘kontol’, ‘anjing’, ‘tolol’, ‘babi’, dan ‘jancok’.

```
normal_words =' '.join([text for text in df['Text'][df['Label'] == 'Tidak
Diketahui Masalahnya']])
wordcloud = WordCloud(width=800, height=500, random_state=21,
max_font_size=110).generate(normal_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()
```

Gambar 4.24 Kode Program Label Tidak Diketahui Masalahnya

Gambar 4.25 *Wordcloud* Label Tidak Diketahui Masalahnya

4.3 Text Preprocessing

Text preprocessing adalah proses untuk membersihkan dan mengorganisir data teks mentah menjadi format yang lebih terstruktur dan siap untuk analisis lebih lanjut. Tahapan awal adalah *case folding*, semua huruf kapital diubah menjadi huruf kecil untuk memastikan konsistensi dalam penulisan. Kemudian, dilakukan *symbol removal* untuk menghilangkan tanda baca, angka, dan karakter khusus yang tidak memberikan informasi semantik yang relevan. Langkah selanjutnya adalah *whitespace removal*, di mana spasi yang berlebihan di awal dan akhir kalimat atau kata dihapus untuk menjaga konsistensi dan kebersihan data. Setelah itu, dilakukan proses tokenisasi untuk memotong kalimat menjadi token-token yang lebih kecil, memfasilitasi pemrosesan lebih lanjut. Data yang berbentuk token-token kecil selanjutnya akan melewati proses penghapusan *stopwords,* proses ini menghapus kata-kata dalam bahasa tertentu yang muncul dalam teks tetapi tidak membawa makna penting dalam analisis teks. Kemudian setelah data melewati proses penghapusan *stopwords* dilanjutkan ke proses normalisasi kata untuk mengubah kata-kata berimbuhan menjadi kata dasar yang baku dan mengganti kata-kata dalam bentuk singkatan atau *slang* menjadi kata dasar. Proses terakhir dalam rangkaian *text preprocessing* yaitu proses *stemming.* Dalam proses *stemming* dilakukan pemrosesan teks yang bertujuan untuk menghapus awalan atau akhiran kata agar hanya sisa akar kata atau bentuk dasarnya. Semua langkah ini dilakukan untuk membersihkan data dan membuatnya lebih siap digunakan dalam tahap analisis dan pemodelan.

Penelitian ini melakukan dua kali percobaan. Percobaan pertama dilakukan *preprocessing* dengan menggunakan penghapusan *stopwords.* Sedangkan percobaan kedua dilakukan tanpa menggunakan tanpa menggunakan penghapusan *stopwords*. Alur percobaan kedua *preprocessing* tanpa menggunakan *stopwords*, yaitu dimulai dari *case folding, symbol, and whitespace removal,* tokenisasi, kemudian langsung melanjutkan ke proses normalisasi data dan yang terakhir yaitu proses *stemming.* Penjelasan mengenai alur *text preprocessing* sebagai berikut:

4.3.1 *Case Folding*

C*ase folding* adalah langkah awal dalam tahapan *text preprocessing* yang memiliki peran penting dalam menormalkan data teks. *Case folding* merupakan proses mengubah semua karakter huruf kapital menjadi huruf kecil tanpa mengubah struktur dan makna kalimat. Tujuan utamanya adalah untuk mencapai konsistensi dalam penulisan, sehingga kata-kata dengan

varian huruf besar dan kecil dianggap setara. Misalnya, kata "Teks" dan "teks" akan diubah menjadi "teks". Kode program untuk *case folding* ditampilkan pada Gambar 4.26.

```
df['Text'] = df['Text'].str.lower()
print('Case Folding Result : \n')
print(df['Text'].head(5))
```

Gambar 4.26 Kode Program *Case Folding*

Hasil dari pemrograman dapat dilihat pada Tabel 4.4. Penulisan menggunakan huruf kapital seperti ‘TAU DIRI’, dan ‘ANJING’ setelah melewati proses *case folding* mengalami perubahan menjadi huruf kecil seperti yang lainya.

Tabel 4.4 Hasil *Case Folding*

Sebelum	**Sesudah *Case Folding***
minimal TAU DIRI!!! lah ngentot dah tau senin nya mau ujian dari dulu kalo mau ujian sabtunya selalu diliburin ini apaan coba ANJINGGG!	minimal tau diri lah!!! ngentot dah tau senin nya mau ujian dari dulu kalo mau ujian sabtunya selalu diliburin ini apaan coba anjingggg!

4.3.2 *Symbol & Whitespace Removal*

Tahapan ini bertujuan untuk menghilangkan komponen teks yang tidak memiliki relevansi semantik, seperti tanda baca, simbol khusus, dan spasi berlebihan. Tanda baca seperti koma, titik, tanda tanya, dan sejenisnya sering kali tidak memberikan kontribusi yang berarti dalam analisis teks dan bisa mengganggu pemahaman lebih lanjut. Oleh karena itu, tanda baca ini dihapus untuk menjaga kejernihan dan konsistensi teks. Selain itu, karakter khusus seperti simbol matematika atau emotikon juga dihilangkan karena tidak memberikan informasi yang berguna dalam analisis. Spasi berlebihan di awal dan akhir kalimat atau kata juga dapat mempengaruhi analisis teks, seperti ketika spasi tambahan di awal kalimat mengakibatkan kata terpisah. Tahap *Symbol & White space Removal* berperan penting dalam menyederhanakan teks dan menghapus komponen yang tidak relevan, sehingga mempersiapkan data dengan lebih baik untuk proses selanjutnya dalam eksplorasi dan model pembelajaran mesin. Kode program *whitespace removal* terdapat pada Gambar 4.27 dan Kode program *symbol removal* terdapat pada Gambar 4.28.

```
def remove_tweet_special(text):
    # remove tab, new line, ans back slice
    text = text.replace('\\t'," ").replace('\\n'," ").replace('\\u'," 
").replace('\\',"")
    # remove non ASCII (emoticon, chinese word, .etc)
```

```
    text = text.encode('ascii', 'replace').decode('ascii')
    # remove mention, link, hashtag
    text = ' '.join(re.sub("([@#][A-Za-z0-9]+)|(\w+:\/\/\S+)"," ",
text).split())
    # remove incomplete URL
    return text.replace("http://", " ").replace("https://", " ")
df['Tweet'] = df['Text'].apply(remove_tweet_special)
```

Gambar 4.27 Kode Program *Symbol Removal*

```
#remove whitespace leading & trailing
def remove_whitespace_LT(text):
    return text.strip()
df['Text'] = df['Text'].apply(remove_whitespace_LT)
#remove multiple whitespace into single whitespace
def remove_whitespace_multiple(text):
    return re.sub('\s+',' ',text)
df['Text'] = df['Text'].apply(remove_whitespace_multiple)
```

Gambar 4.28 Kode Program *Whitespace Removal*

Hasil dari pemrograman dapat dilihat pada Tabel 4.5. Setelah melewati proses *Symbol & White space Removal* terdapat beberapa perbedaan seperti hilangnya tanda seru pada kata setelah ‘tau diri’, spasi ganda yang terdapat pada kata ‘dari dulu’ hilang menjadi normal dengan satu spasi, kemudian tanda seru pada kata setelah ‘anjingggg’ menjadi hilang sama seperti pada kata ‘tau diri’.

Tabel 4.5 Hasil *Symbol & Whitespace Removal*

Sebelum *Symbol & Whitespace Removal*	**Sesudah *Symbol & Whitespace Removal***
minimal tau diri lah!!! ngentot dah tau senin nya mau ujian dari dulu kalo mau ujian sabtunya selalu diliburin ini apaan coba anjingggg!	minimal tau diri lah ngentot dah tau senin nya mau ujian dari dulu kalo mau ujian sabtunya selalu diliburin ini apaan coba anjingggg
ini skripsi gue gimana???? disuruh ngeswitch perasaan ga semudah itu	ini skripsi gue gimana disuruh ngeswitch perasaan ga semudah itu
asu asu asuu!! fuck you guys!!	asu asu asuu fuck you guys
template ortu marah-marah "kamu udah ngerasa lebih pinter ya dari ortu?" "salah tuh minta maaf ga usah bela diri" (posisinya padahal dia yang salah wkwkwk kocak)	template ortu marah-marah kamu udah ngerasa lebih pinter ya dari ortu salah tuh minta maaf ga usah bela diri posisinya padahal dia yang salah wkwkwk kocak
duit abang gue sama dia semua tapi serasa dia yg biayain semua gajelas !! gue kesel tapi gak bisa ngomong ksar !!@-$-$(#!!#?@	duit abang gue sama dia semua tapi serasa dia yg biayain semua gajelas gue kesel tapi gak bisa ngomong ksar

4.3.3 Tokenisasi

Tokenisasi merupakan langkah penting dalam *text preprocessing* yang melibatkan pemotongan teks menjadi unit-unit yang lebih kecil yang disebut sebagai "token". Token ini biasanya merupakan kata-kata individual atau simbol tertentu yang memiliki makna dalam

konteks teks. Tokenisasi bertujuan untuk memecah teks menjadi bagian-bagian yang lebih terstruktur sehingga dapat lebih mudah diolah dan dianalisis oleh model pembelajaran mesin. Proses tokenisasi biasanya dilakukan dengan membagi teks berdasarkan spasi antara kata-kata. Kode program untuk proses tokenisasi terdapat pada Gambar 4.29.

```
# NLTK word tokenize
def word_tokenize_wrapper(text):
    return word_tokenize(text)
df['tweet_tokens'] = df['Text'].apply(word_tokenize_wrapper)
```

Gambar 4.29 Kode Program Tokenisasi

Hasil dari tokenisasi adalah sekumpulan token yang siap untuk diolah lebih lanjut dalam tahapan-tahapan berikutnya. Dengan proses tokenisasi, teks yang sebelumnya rumit dan panjang dipecah menjadi segmen-segmen yang lebih mudah untuk dikelola, sehingga mendukung proses analisis teks yang lebih efisien dan efektif. Hasil dari tokenisasi bisa dilihat pada Tabel 4.6.

Tabel 4.6 Hasil Tokenisasi

Sebelum	Sesudah Tokenisasi
minimal tau diri lah ngentot dah tau senin nya mau ujian dari dulu kalo mau ujian sabtunya selalu diliburin ini apaan coba anjingggg	['minimal', 'tau', 'diri', 'lah', 'ngentot', 'dah', 'tau', 'senin', 'nya', 'mau', 'ujian', 'dari', 'dulu', 'kalo', 'mau', 'ujian', 'sabtunya', 'selalu', 'diliburin', 'ini', 'apaan', 'coba', 'anjingggg']
kagak lagi dah gw demen sama cwo meodelan lu	['kagak', 'lagi', 'dah', 'gw', 'demen', 'sama', 'cwo', 'meodelan', 'lu']
balas chat ku dong karim	['balas', 'chat', 'ku', 'dong', 'karim']
ni jaman pembunuhan apa gmana dr kemaren denger banyak yang di bunuh lah keroyok lah di bacok lah bener singgahsana setan mau di ganti	['ini', 'jaman', 'pembunuhan', 'apa', 'bagaimana', 'dari', 'kemarin', 'denger', 'banyak', 'yang', 'di', 'bunuh', 'lah', 'keroyok', 'lah', 'di', 'bacok', 'lah', 'bener', 'singgahsana', 'setan', 'mau', 'di', 'ganti']
susah susah naikin value jd cwe malah ternyata dijadiin selingkuhan anjingggggggg	['susah', 'susah', 'naikin', 'value', 'jd', 'cwe', 'malah', 'ternyata', 'dijadiin', 'selingkuhan', 'anjingggggggg']

4.3.4 Penghapusan *Stopwords*

Stopwords merujuk pada kata-kata dalam bahasa tertentu yang secara rutin muncul dalam teks tetapi biasanya tidak membawa makna penting dalam analisis teks. Jenis kata-kata ini meliputi konjungsi, artikel, preposisi, dan kata-kata umum lainnya seperti "dan", "atau", "di", "dari", dan lain sebagainya. Kode program untuk proses penghapusan *stopwords* dapat dilihat pada Gambar 4.30.

```
list_stopwords = set(list_stopwords) # convert list to dictionary
def stopwords_removal(words): #remove stopword pada list token
    return [word for word in words if word not in list_stopwords]
df['tweet_tokens_WSW'] = df['tweet_tokens'].apply(stopwords_removal)
```

Gambar 4.30 Kode Program Penghapusan *Stopwords*

Hasil dari penghapusan *stopwords* dapat dilihat pada Tabel 4.7. Proses penghapusan *stopwords* membantu mengeliminasi kata-kata yang memiliki dampak makna yang minim. Seperti kata ‘ih’ pada baris pertama hilang setelah melewati proses penghapusan *stopwords*. Selain itu, kata ‘argh’ pada baris kedua, kata ‘di’ pada baris ketiga hilang setelah melalui proses penghilangan *stopwords*.

Tabel 4.7 Penghapusan *Stopwords*

Sebelum Penghapusan Stopwords	Sesudah Penghapusan Stopwords
['ih', 'asu', 'goblok', 'bangettttttttttt']	['asu', 'goblok']
['argh', 'jancoookkkkkk']	['jancoookkkkkk']
['hadeh', 'kontol', 'berisik', 'bgt']"	['kontol', 'berisik']
['ga', 'lucu', 'anjinggg', 'kalo', 'gua', 'ujungnya', 'di', 'ghostinggg']	['ga', 'lucu', 'anjinggg', 'gua', 'ujungnya', 'ghostinggg']
['dikira', 'ngerjain', 'tugas', 'ngicep', 'beres', 'bgst']	['ngerjain', 'tugas', 'ngicep', 'beres', 'bgst']

4.3.5 Normalisasi Kata

Normalisasi data bertujuan untuk mengubah kata-kata berimbuhan menjadi bentuk kata dasar yang baku. Selain itu, kata-kata yang ditulis dalam bentuk singkatan atau bahasa gaul akan diubah menjadi bentuk kata yang lebih standar dan umum. Proses ini membantu menyatukan variasi yang mungkin terjadi pada kata-kata yang memiliki arti yang sama, sehingga mempermudah analisis dan pemahaman konten teks. Kode program terdapat pada Gambar 4.31.

```
normalizad_word = pd.read_csv("normalisasi.csv", sep=';')
normalizad_word_dict = {}
for index, row in normalizad_word.iterrows():
    if row[0] not in normalizad_word_dict:
        normalizad_word_dict[row[0]] = row[1]
def normalized_term(document):
    return [normalizad_word_dict[term] if term in normalizad_word_dict else
term for term in document]
df['tweet_normalized'] = df['tweet_tokens'].apply(normalized_term)
df['tweet_normalized'].tail(10)
```

Gambar 4.31 Kode Program Normalisasi Kata

Normalisasi data dilakukan berdasarkan kamus alay yang diambil dari akun GitHub nasalsabila (url: https://bit.ly/kamus-alay) yang kemudian dimodifikasi untuk disesuaikan dengan kebutuhan penelitian. Sampel dari kamus alay bisa dilihat pada Gambar 4.8.

Tabel 4.8 Normalisasi Kata

Sebelum	Sesudah
woww	wow
aminn	amin
skrg	sekarang
bingit	banget
nyesel	menyesal
bnr2	benar-benar
esmosiii	emosi

Hasil dari pemrograman dapat dilihat pada Gambar 4.9. Setelah melewati proses normalisasi data, terdapat beberapa perbedaan seperti kata ‘dah’ menjadi kata ‘udah’. Kemudian jika dilihat pada kata terakhir dalam kalimat tersebut, kata ‘anjingggg’ memiliki huruf ‘g’ sebanyak empat kata. Setelah melewati fase ini huruf ‘g’ berkurang menjadi ‘anjing’.

Tabel 4.9 Hasil Normalisasi Data

Sebelum	Sesudah Normalisasi Data
['minimal', 'tau', 'diri', 'lah', 'ngentot', 'dah', 'tau', 'senin', 'nya', 'mau', 'ujian', 'dari’, ’dulu', 'kalo', 'mau', 'ujian', 'sabtunya', 'selalu', 'diliburin', 'ini', 'apaan', 'coba', 'anjingggg']	['minimal', 'tau', 'diri', 'lah', 'ngentot', 'udah', 'tau', 'senin', 'nya', 'mau', 'ujian', 'dari’, ’dulu', 'kalo', 'mau', 'ujian', 'sabtunya', 'selalu', 'diliburin', 'ini', 'apaan', 'coba', 'anjing']
['kagak', 'lagi', 'dah', 'gw', 'demen', 'sama', 'cwo', 'meodelan', 'lu']	['kagak', 'lagi', 'dah', 'gue', 'demen', 'sama', 'cowo', 'modelan', 'elu']
['balas', 'chat', 'ku', 'dong', 'karim']	['balas', 'chat', 'aku', 'dong', 'karim']
['ini', 'jaman', 'pembunuhan', 'apa', 'bagaimana', 'dari', 'kemarin', 'denger', 'banyak', 'yang', 'di', 'bunuh', 'lah', 'keroyok', 'lah', 'di', 'bacok', 'lah', 'bener', 'singgahsana', 'setan', 'mau', 'di', 'ganti']	['ini', 'zaman', 'pembunuhan', 'apa', 'bagaimana', 'dari', 'kemarin', 'dengar', 'banyak', 'yang', 'di', 'bunuh', 'lah', 'keroyok', 'lah', 'di', 'bacok', 'lah', 'bener', 'singgahsana', 'setan', 'mau', 'di', 'ganti']
['susah', 'susah', 'naikin', 'value', 'jd', 'cwe', 'malah', 'ternyata', 'dijadiin', 'selingkuhan', 'anjinggggggg']	['susah', 'susah', 'naikin', 'value', 'jd', 'cewe', 'malah', 'ternyata', 'dijadiin', 'selingkuhan', 'anjing']

4.3.6 *Stemming*

Stemming merupakan tahapan dalam pemrosesan teks yang memiliki maksud untuk mengidentifikasi dan menghilangkan infleksi kata dalam kata-kata. *Stemming* adalah alat yang berguna dalam analisis teks untuk mengurangi kerumitan dalam pemrosesan dan memungkinkan pengelompokan kata-kata serupa dalam teks. Selain menggunakan *libarary*

Sastrawi, pada proses *stemming* juga mengaplikasikan *library* Swifter untuk mempercepat proses *stemming*. Kode program stemming dapat dilihat pada Gambar 4.32.

```
# apply stemmed term to dataframe
def get_stemmed_term(document):
    return [term_dict[term] for term in document]
df['tweet_tokens_stemmed']                                                    =
df['tweet_normalized'].swifter.apply(get_stemmed_term)
print(df['tweet_tokens_stemmed'])
```

Gambar 4.32 Kode Program *Stemming*

Proses *stemming* menghasilkan token-token kata yang berdiri sendiri tanpa adanya imbuhan maupun pengubahan kata. Hal tersebut memudahkan dalam analisis teks yang lebih efisien dengan mengurangi variasi kata yang sama dalam teks sehingga memudahkan pengelompokan, pencarian, dan pemahaman kontennya. Gambar 4.10 menunjukkan hasil dari proses *stemming*. Beberapa kata yang memiliki imbuhan seperti pada kata ‘mainin’ berubah menjadi ‘main’. Kemudian kata ‘mengesalkan’ setelah melewati proses *stemming* berubah menjadi ‘kesal’.

Tabel 4.10 *Stemming*

Sebelum Stemming	Sesudah Stemming
['buset', 'kakak', 'gue', 'tiada', 'hari', 'tanpa', 'marahin', 'sama', 'mukulin', 'anaknya', 'anjir']	['buset', 'kakak', 'gue', 'tiada', 'hari', 'tanpa', 'marah', 'sama', 'pukul', 'anak', 'anjir']
['gabut', 'itu', 'dengerin', 'lagu', 'bukan', 'mainin', 'perasaan', 'orang', 'bangsat']	['gabut', 'itu', 'dengar', 'lagu', 'bukan', 'main', 'asa', 'orang', 'bangsat']
['bangsat', 'gue', 'kenapa', 'sih', 'nangisin', 'papa', 'ahgh']	['bangsat', 'gue', 'kenapa', 'sih', 'nangis', 'papa', 'ahgh']
['mengesalkan', 'banget', 'jancok']	['kesal', 'banget', 'jancok']
['bun', 'hidup', 'berjalan', 'seperti']	['bun', 'hidup', 'jalan', 'seperti']

4.4 Klasifikasi

Klasifikasi merupakan tahapan yang dilakukan untuk memperoleh hasil pengelompokan masalah yang ada di Komunitas Marah-Marah. Klasifikasi ini melakukan beberapa tahapan untuk membangun model *Bidirectional Long Short-Term Memory* (BiLSTM) seperti:

4.4.1 Data *Undersampling*

Dalam konteks penelitian mengenai Komunitas Marah-Marah di Twitter, metode *data undersampling* digunakan untuk menangani ketidakseimbangan kelas pada *dataset*. Ketidakseimbangan kelas adalah situasi di mana ada ketimpangan signifikan dalam jumlah data antara satu kelas dengan kelas lainnya, yang dapat menyebabkan model cenderung

memihak pada kelas mayoritas dan memiliki kinerja yang buruk dalam mengidentifikasi kelas minoritas. Pada *dataset* terdapat satu kategori yang berjumlah di bawah seribu yaitu kategori Karier/Pekerjaan. Kemudian ada dua kategori yang berjumlah diatas 5000 yaitu kelas Person/Personal dan Tidak Diketahui Masalahnya. Pada kategori lainya berkisar di angka 1000-2000 data.

Dalam upaya untuk mengatasi masalah ini, *undersampling* melibatkan pengurangan jumlah sampel dari kelas mayoritas sehingga seimbang dengan jumlah sampel dari kelas minoritas seperti pada Tabel 4.11. Pendekatan ini dapat membantu model lebih baik dalam mengenali pola-pola pada kedua kelas dan mengurangi bias terhadap kelas mayoritas. Jumlah pengurangan data per kategori disesuaikan dengan jumlah kategori data terendah, yang mana terdapat dua kategori yang memiliki data terendah yaitu kategori Karier/Pekerjaan dan Studi. Jumlah data dari kategori Studi dipilih agar kategori lain tidak membuang banyak data dalam kategorinya dibanding ketika menggunakan patokan jumlah dari kategori Karier/Pekerjaan. Kode program untuk melakukan *undersampling* data terdapat pada Gambar 4.33.

Tabel 4.11 Hasil *Undersampling*

Label Permasalahan	**Label Integer**	**Jumlah Sebelum *Undersampling***	**Jumlah Setelah *Undersampling***
Studi	1	1.637	1.637
Percintaan	2	2.349	1.637
Keluarga	3	1.922	1.637
Karir/Pekerjaan	4	683	683
Person/Personal	5	5.004	1.637
Tidak Diketahui Masalahnya	6	5.681	1.637
Jumlah		17.276	8.872
		Selisih Jumlah Sebelum vs Sesudah *Undersampling*: 8.404	

```
number_label = 1637
shuffled = df.reindex(np.random.permutation(df.index))
u = shuffled[shuffled['label']==6][:number_label]#Tidak DiketahuiMasalahnya
pp = shuffled[shuffled['label'] == 5][:number_label] #Person/Personal
pc = shuffled[shuffled['label'] == 2][:number_label] #Percintaan
ke = shuffled[shuffled['label'] == 3][:number_label] #Keluarga
s = shuffled[shuffled['label'] == 1]  #Studi
ka = shuffled[shuffled['label'] == 4] #Karier/Pekerjaan
concate = pd.concat([u, pp, pc, ke, s, ka], ignore_index=True)
concate = concate.reindex(np.random.permutation(concate.index))
```

Gambar 4.33 Kode Program *Undersampling*

4.4.2 Tokenisasi

Tokenisasi merupakan langkah untuk memproses teks yang melibatkan pemisahan teks menjadi unit-unit kecil yang disebut token. Proses tokenisasi pertama dilakukan pada saat *preprocessing* yang hasil akhirnya berupa token yang disimpan dalam format CSV, namun *file* CSV tersebut mengalami *error* dengan keterangan ada beberapa baris yang rusak dan tidak dapat dibuka menggunakan *library* Pandas. Hal tersebut yang menyebabkan dilakukannya proses tokenisasi kedua pada saat dilakukan pemodelan.

File preprocessing yang semula bertipe data *list* diubah dan digabungkan menjadi tipe data *string* yang disimpan dalam format CSV. *File dataset* tersebut yang akan digunakan untuk proses klasifikasi. Pada proses klasifikasi dilakukan tokenisasi kedua menggunakan fungsi "`fit_on_texts`" yang bertujuan untuk membuat indeks untuk setiap kata atau karakter dalam teks berdasarkan frekuensi kemunculannya. Ini menghasilkan kamus kata unik yang dikenal oleh model. Fungsi "`text_to_sequences`" kemudian mengubah setiap kata dalam teks menjadi urutan indeks sesuai dengan kamus yang telah dibangun sebelumnya. Langkah selanjutnya adalah menggunakan fungsi "`pad_sequences`" untuk menyamakan panjang dimensi dari semua urutan indeks sesuai dengan nilai "`maxlen`" yang telah ditetapkan sebelumnya. Nilai "`maxlen`" menentukan panjang maksimum dari urutan token yang digunakan. Hal ini penting karena model *neural network* membutuhkan *input* dengan dimensi yang konsisten. Dengan cara ini, tokenisasi mempersiapkan data teks untuk diolah oleh model dengan memastikan konsistensi dimensi dan representasi token. Kode program untuk proses ini terdapat pada Gambar 4.34.

```
X = tokenize_tweet.texts_to_sequences(concate['Tweet'].values)
X = pad_sequences(X, maxlen=MAX_SEQ_LENGTH)
print('Shape of data tensor:', X.shape)
```

Gambar 4.34 Kode Program Tokenisasi

4.4.3 *One Hot Encoding*

One-Hot Encoding merupakan proses mengubah setiap kategori dalam label menjadi kolom biner terpisah menggunakan fungsi *`pd.get_dummies`*. Proses tersebut dilakukan agar setiap label atau kategori dalam sebuah variabel diterjemahkan menjadi bentuk numerik yang lebih deskriptif dan bermakna. Hasil dari pemrosesan ini menghasilkan kolom-kolom biner terpisah untuk setiap kategori unik dalam variabel kategorikal. Proses ini menghasilkan

dataset baru yang lebih mudah diolah dan lebih siap untuk digunakan dalam model analisis. Kode pemrograman seperti pada Gambar 4.35.

```
Y = pd.get_dummies(concate['label']).values
print('Shape of label tensor:', Y.shape)
```

Gambar 4.35 Kode Program *One Hot Encoding*

4.4.4 Pembagian Data

Pada bagian pembagian data dilakukan dua skenario pembagian yang berbeda. Pada bagian pertama dilakukan pembagian data latih sebesar 90% (7.981 data) dan data uji sebesar 10% (887 data) menggunakan fungsi `train_test_split`. Fungsi *`train_test_split`* memisahkan dataset menjadi dua *subset* yang independen secara acak, memastikan bahwa model tidak "melihat" data uji selama proses pelatihan, sehingga evaluasi menjadi lebih objektif. Kode program dapat dilihat pada Gambar 4.36.

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.10,
random_state = 42)
print(X_train.shape,Y_train.shape)
print(X_test.shape,Y_test.shape)
```

Gambar 4.36 Kode Program Pembagian Data 90:10

Pembagian data bagian dua dilakukan dengan komposisi data latih sebesar 80% (7094 data) dan data uji yang digunakan untuk menguji model pada bagian kedua ini sebesar 20% (1774 data). Kode program ditunjukkan pada Gambar 4.37.

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.20,
random_state = 42)
print(X_train.shape,Y_train.shape)
print(X_test.shape,Y_test.shape)
```

Gambar 4.37 Kode Program Pembagian Data 80:20

4.4.5 Model

Model BiLSTM dilatih menggunakan beberapa hyperparamter sebagai berikut:

1. *Batch size* mengacu pada jumlah sampel pelatihan yang digunakan dalam satu iterasi pelatihan sebelum parameter model diperbarui. Penelitian ini melakukan percobaan dengan menggunakan tiga nilai *batch size*, yaitu 32, 64, 128. *Batch size* yang menghasilkan akurasi terbaik akan digunakan sebagai parameter pelatihan model.

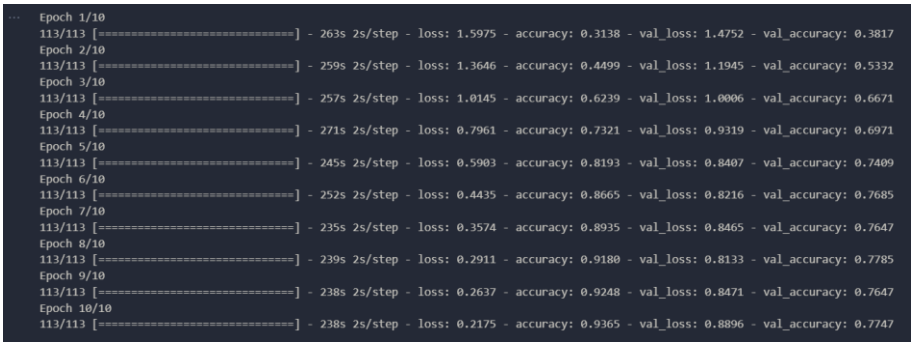
2. *Epochs* mengacu pada satu putaran lengkap dari seluruh *dataset* pelatihan. Proses pelatihan model akan menggunakan 10 *epochs* dikarenakan jumlah ukuran *dataset* yang kecil.
3. *Optimizer* merupakan algoritma atau metode yang digunakan untuk mengoptimalkan bobot (*weights*) model *neural network* selama proses pelatihan. Penelitian ini akan menggunakan Adam sebagai *optimizer*. Adam dipilih karena memiliki tingkat konvergensi yang baik dan bekerja secara efisien dalam banyak kasus.
4. *Loss* merupakan suatu ukuran yang mengukur sejauh mana prediksi model mendekati nilai sebenarnya dari data pelatihan. *Loss* yang digunakan pada penelitian ini adalah *Categorical Crossentropy*. Hal tersebut disesuaikan dengan penelitian ini yaitu klasifikasi dengan banyak kelas.
5. Fungsi aktivasi merupakan fungsi matematis yang digunakan dalam setiap *neuron* atau unit dalam jaringan saraf (*neural network*) untuk memproses *input* dan menghasilkan *output. Softmax* diaplikasikan sebagai fungsi aktivasi dalam penelitian ini karena memiliki kemampuan yang bagus dalam tugas klasifikasi.
6. *Validation split* merupakan parameter yang dugunakan dalam fungsi `model.fit`. Penelitian ini menggunakan *dataset* sebesar 17.276 data yang dibagi menggunakan fungsi `train_test_split` menjadi 90% (7981 data) untuk data latih dan 10% (887 data) untuk data uji. Kemudian, pada saat pelatihan model menggunakan fungsi `validation_split` = 0,1 untuk membagi data latih menjadi data latih (7.183 data) dan data validasi (798 data) selama pelatihan model. Penggunaan `validation_split` berguna untuk memantau kinerja model selama pelatihan dan menghindari *overfitting* tanpa perlu data validasi terpisah. Setelah pelatihan selesai, dilanjutkan menguji kinerja model pada data uji terpisah untuk evaluasi akhir. Penelitian ini melakukan percobaan dengan menggunakan nilai *validation split* sebesar 0,1 dan 0,2. Nilai yang menghasilkan akurasi terbaik akan digunakan sebagai hyperparameter untuk pelatihan model.
7. *Spatial dropout* merupakan teknik regulasi untuk mengurangi *overfitting*. Penelitian ini memiliki ukuran *dataset* yang kecil sehingga diperlukan nilai *spatial dropout* yang tinggi yaitu 0,4 dan 0,5. Penelitian ini melakukan percobaan menggunakan dua nilai *spatial dropout* yaitu 0,4 dan 0,5. Nilai *dropout* yang menghasilkan akurasi terbaik akan dipilih untuk pelatihan model.

8. *Dropout* merupakan teknik regulasi yang digunakan dalam *deep learning* untuk mengurangi *overfitting* dalam jaringan saraf. Model yang dibangun pada penelitian ini tergolong sederhana, sehingga diterapkan nilai *dropout* yang rendah. Untuk mengetahui nilai *dropout* yang cocok untuk model, dilakukan percobaan untuk nilai *dropout.* Percobaan dilakukan dengan menggunakan satu lapisan *spatial dropout* dengan nilai sebesar 0,4. Kemudian ditambahkan satu lapisan *dropout* dengan nilai sebesar 0,2 yang dijalankan menggunakan 10 *epochs*. Pemodelan ini dilakukan menggunakan *dataset* tanpa proses penghilangan *stopwords*. Kode program ditunjukkan pada Gambar 4.38.

```
epochs = 10
batch_size = 64
model = Sequential()
model.add(Embedding(MAX_NUMBER_WORDS,
                    EMBEDDING_DIMENSION,
                    input_length=X.shape[1]))
model.add(SpatialDropout1D(0.4))
model.add(Bidirectional(LSTM(EMBEDDING_DIMENSION,
                             dropout=0.2,
                             recurrent_dropout=0.2)))
model.add(Dense(EMBEDDING_DIMENSION, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(6))
model.add(Activation('softmax'))
model.compile(loss='categorical_crossentropy',
              optimizer='adam', metrics=['accuracy'])
history = model.fit(X_train,
                    Y_train,
                    epochs=epochs,
                    batch_size=batch_size,
                    validation_split=0.1,
                    callbacks=[EarlyStopping(monitor='val_loss',
patience=3, min_delta=0.0001)])
```

Gambar 4.38 Kode Program Percobaan *Dropout*

Berdasarkan hasil pemodelan diatas didapatkan hasil seperti pada Gambar 4.39. Akurasi antara data latih dan validasi set memiliki rentang yang cukup jauh, yaitu sebesar 0,1618. Dalam hal ini, model mengindikasikan adanya *overfiitng* atau model terlalu menghafal data pelatihan. Hal tersebut juga diperkuat dengan perubahan validasi akurasi yang naik turun. Perubahan terjadi pada *epochs* ke-7 hingga *epochs* ke-10.



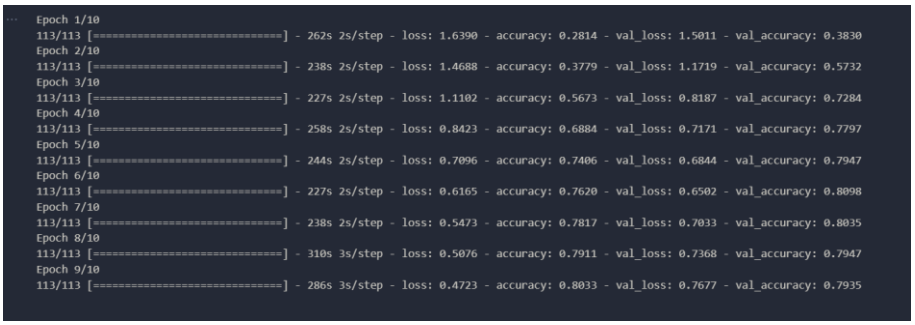
Gambar 4.39 Hasil Percobaan *Dropout* ke-1

Setelah melihat hasil dari pengujian pertama, dilakukan percobaan kedua dengan menambahkan satu lapisan *dropout* di bawah lapisan *dense* sebesar 0,3. Kemudian nilai *dropout* pada lapisan pertama berubah menjadi 0,2. Hal ini dilakukan karena perubahan nilai akurasi validasi berada pada *epochs* ke-7 sehingga dilakukan penambahan lapisan *dropout* sebesar 0,3. Kode program ditunjukkan pada Gambar 4.40.

```
epochs = 10
batch_size = 64
model = Sequential()
model.add(Embedding(MAX_NUMBER_WORDS,
                    EMBEDDING_DIMENSION,
                    input_length=X.shape[1]))
model.add(SpatialDropout1D(0.4))
model.add(Bidirectional(LSTM(EMBEDDING_DIMENSION,
                             dropout=0.1,
                             recurrent_dropout=0.1)))
model.add(Dense(EMBEDDING_DIMENSION, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(6))
model.add(Dropout(0.3))
model.add(Activation('softmax'))
model.compile(loss='categorical_crossentropy',
              optimizer='adam', metrics=['accuracy'])
history = model.fit(X_train, Y_train,
                    epochs=epochs,
                    batch_size=batch_size,
                    validation_split=0.1,
                    callbacks=[EarlyStopping(monitor='val_loss',
patience=3, min_delta=0.0001)])
```

Gambar 4.40 Kode Program Percobaan *Dropout* 2

Hasil dari percobaan kedua dapat dilihat pada Gambar 4.41. Nilai *epochs* menurun menjadi sembilan karena model sudah melakukan beberapa kali percobaan. Selain itu, pemodelan ini menerapkan teknik *early stopping*, maka pelatihan dapat berhenti lebih awal jika kinerja model sudah mencapai puncaknya, yang mungkin menghasilkan nilai *epochs* yang lebih kecil dari yang tetapkan.



Gambar 4.41 Hasil Percobaan *Dropout* ke-2

Penurunan akurasi tetap terjadi pada *epochs* ke-6 menuju ke-7. Namun selisih pengurangan akurasi tersebut tidak terlalu drastis yaitu sebesar 0,0063. Selanjutnya antara *epochs* ke-7 menuju *epochs* ke-8 memiliki perbedaan nilai sebesar 0,0088. Kemudian untuk *epochs* ke 8 menuju *epochs* ke-9 memiliki selisih nilai sebesar 0,0012. Selain itu, perbedaan selisih akurasi antara data latih dan validasi set hanya berkisar 0,0098. Total penurunan akurasi validasi yang terjadi dari *epochs* ke-7 hingga *epochs* ke-10 sebesar 0,01, penurunan itu tetap terjadi karena ukuran *dataset* keseluruhan yang terbatas. Untuk itu, kombinasi nilai *dropout* dan *spatial dropout* ini yang akan digunakan selama proses pengujian model. Kombinasi nilai *dropout* dan *spatial dropout* dapat dilihat pada Tabel 4.12.

Tabel 4.12 Nilai *Dropout*

Dropout	0,1	0,1	0,2	0,3

Penjelasan mengenai percobaan proses klasifikasi model dengan dua skenario, yaitu klasifikasi menggunakan *stopwords* dan klasifikasi tanpa menggunakan *stopwords* sebagai berikut:

a. Klasifikasi Menggunakan *Stopwords*

Percobaan ini menggunakan tiga `batch_size`, yaitu 32, 64, 128. Kemudian *dataset* dibagi menjadi data latih dan data uji. Pembagian *dataset* dilakukan dengan menggunakan dua skenario pembagian. Skenario pertama *split data* sebesar 90% digunakan untuk data latih yang termasuk *validation set* sebesar 10% dan 10% sisanya digunakan sebagai data uji. Kemudian data juga akan melewati dua kali proses percobaan pengguanaan nilai *spatial dropout* dengan nilai 0,4 dan 0,5.

Skenario kedua dengan melakukan pembagian data 80% untuk data latih yang termasuk *validation set* sebesar 20%. Sisa dari data latih yaitu sebesar 20% digunakan sebagai data uji. Pada skenario kedua ini juga akan melalui dua kali proses percobaan penggunaan nilai *spatial dropout* dengan nilai 0,4 dan 0,5. Penggunaan nilai `batch_size`, nilai *Dropout* dan *Spatial Dropout* yang ada pada Tabel 4.13.

Tabel 4.13 Perbandingan *Stopwords* dengan Dropout 0,4

Spatial Dropout	0,4			
Dropout	0,1	0,1	0,2	0,3

Pada Tabel 4.14, klasifikasi dilakukan dengan menggunakan *stopwords* dengan pembagian data latih sebesar 80% termasuk data validasi sebesar 20%, dan data uji sebesar 20%. Selain itu, percobaan 1 dilakukan dengan menggunakan nilai *batch size* yang berbeda. Akurasi pelatihan tertinggi didapatkan oleh kolom A1 dengan perolehan hasil *Accuracy Train* sebesar 0,8009. Namun, hasil akurasi terbaik untuk validasi set didapatkan oleh kolom A3 dengan nilai *Accuracy validation* sebesar 0,7674. Kemudian untuk nilai akurasi tes terbaik dengan hasil 0,7390 didapatkan oleh kolom A1.

Tabel 4.14 Percobaan 1 *Stopwords* 80:20

	A1	**A2**	**A3**
Batch Size	**32**	**64**	**128**
Train	80	80	80
Test	20	20	20
Val split	0,2	0,2	0,2
Accuracy Train	0.8009	0.7975	0.7921
Accuracy Validation	0.7435	0.7562	0.7674
Accuracy Test	0.7390	0.7339	0.7604

Percobaan ke-2 dilakukan dengan menggunakan pembagian data latih sebesar 90% yang termasuk *validation set* sebesar 10%. Kemudian sebanyak 10% data sisanya digunakan untuk data uji. Hasil dari Percobaan 2 dengan menggunakan *stopwords* dapat dilihat pada Tabel 4.15. Percobaan kedua ini menggunakan tiga nilai *batch size* yang berbeda. Berdasarkan hasil eksperimen didapatkan hasil *accuracy train* tertinggi sebesar 0,8016. *Accuracy Train* tersebut berasal dari kolom B1. Kemudian nilai *Accuracy validation* tertinggi didapatkan pada kolom B1 dengan perolehan 0,7922. *Accuracy Test* tertinggi ada pada kolom B2 dan B3 dengan perolehan sebesar 0,7655.

Tabel 4.15 Percobaan 2 *Stopwords* 90:10

	B1	**B2**	**B3**
Batch Size	**32**	**64**	**128**
Train	90	90	90
Test	10	10	10
Val_split	0,1	0,1	0,1
Accuracy Train	0.7874	0.8016	0.7843
Accuracy Validation	0.7922	0.7810	0.7572
Accuracy Test	0.7373	0.7655	0.7655

Pada percobaan ketiga dan keempat menggunakan nilai *spatial dropout* yang berbeda seperti pada Tabel 4.16. Hal ini dilakukan sebagai pembanding hasil antara hasil klasifikasi menggunakan nilai *spatial dropout* 0,4 dan 0,5.

Tabel 4.16 Perbandingan *Stopwords* dengan Dropout 0,5

Spatial Dropout	0,5			
Dropout	0,1	0,1	0,2	0,3

Percobaan ke-3 seperti pada Tabel 4.17 menggunakan nilai *spatial dropout* sebesar 0,5 dengan pembagian data latih sebesar 80% yang termasuk data validasi sebesar 20%, dan data uji sebesar 20%. Berdasarkan hasil percobaan ke tiga didapatkan nilai *Accuracy Train* tertinggi pada kolom C2 sebesar 0,7827. Kemudian untuk nilai akurasi validasi tertinggi berada pada kolom C3 dengan perolehan akurasi sebesar 0,7576, selisih 0,0050 dari kolom C2. Untuk yang terakhir yaitu nilai akurasi tes tertinggi didapatkan oleh kolom C2 sebesar 0,7616.

Tabel 4.17 Percobaan 3 *Stopwords* 80:20

	C1	**C2**	**C3**
Batch Size	**32**	**64**	**128**
Train	80	80	80
Test	20	20	20
Val_split	0,2	0,2	0,2
Accuracy Train	0.7237	0.7827	0.7697
Accuracy Validation	0.7477	0.7526	0.7576
Accuracy Test	0.7565	0.7616	0.7559

Percobaan ke empat dilakukan dengan menggunakan pembagian data latih sebesar 90% yang termasuk validation set sebesar 10%. Kemudian sebanyak 10% data sisanya digunakan untuk data uji. Hasil dari percobaan keempat dapat dilihat pada Tabel 4.18. Nilai akurasi tertinggi untuk hasil akurasi data latih atau *Accuracy Train* didapatkan oleh kolom D2 dengan

akurasi sebesar 0,7955. Kolom D1 dengan hasil akurasi validasi 0,7572 mendapatkan hasil akurasi tertinggi diantara kolom D2 yang mendapatkan akurasi sebesar 0,7509 dan kolom D3 dengan akurasi sebesar 0,7534. Kemudian yang terakhir yaitu akurasi tes tertinggi berada pada kolom D3 dengan perolehan akurasi sebesar 0,7554.

Tabel 4.18 Percobaan 4 *Stopwords* 90:10

	D1	**D2**	**D3**
Batch Size	**32**	**64**	**128**
Train	90	90	90
Test	10	10	10
Val_split	0,1	0,1	0,1
Accuracy Train	0.7888	0.7955	0.7751
Accuracy Validation	0.7572	0.7509	0.7534
Accuracy Test	0.7452	0.7631	0.7554

Berdasarkan hasil percobaan skenario menggunakan *stopwords*, pembagian data dengan komposisi 90:10, komposisi 80:20, percobaan tiga nilai *batch_size* dan percobaan *spatial dropout*. Didapatkan hasil terbaik seperti pada Tabel 4.19. Hasil terbaik didapatkan dengan menggunakan *spatial dropout* bernilai 0,4. Kemudian mayoritas akurasi terbaik didapatkan dengan menggunakan pembagian data 90:10. Hasil dari penelitian menggunakan *stopwords* didapatkan hasil terbaik dengan menggunakan *spatial dropout* bernilai 0,4 dengan *batch size* bernilai 64. Akurasi tinggi didapatkan dengan pembagian data dengan komposisi 90:10. Akurasi data latih atau *Accuracy Train* tertinggi berjumlah 0,8016 dengan *Accuracy Test* sebesar 0,7810.

Tabel 4.19 Hasil Terbaik Menggunakan *Stopwords*

Spatial Dropout	***Batch_size***	***Train***	***Test***	***Val_split***	***Acc Train***	***Acc Val***	***Acc Test***
0,4	64	0,90	0,10	0,10	0.8016	0.7810	0.7655
0,5	64	0,90	0,10	0,10	0.7955	0.7509	0.7631

b. Klasifikasi Tanpa Menggunakan *Stopwords*

Percobaan klasifikasi tanpa menggunakan *stopwords* ini akan melewati tiga pengujian *batch_size,* yaitu 32, 64, 128. Pembagian *dataset* dilakukan dengan menggunakan dua skenario pembagian. Skenario pertama dilakukan pembagian data sebanyak 90% digunakan untuk data latih yang 10% di dalamnya juga terdapat *validation set,* dan 10% data digunakan sebagai data uji. Kemudian data akan melewati dua kali proses percobaan pengguanaan nilai *spatial dropout* dengan nilai 0,4 dan 0,5.

Skenario kedua melakukan pembagian data yang berbeda seperti skenario pertama. Pembagian data 80% untuk data latih yang di dalamnya juga termasuk 20% data untuk melakukan *validation set*. Sisa dari data latih yaitu sebesar 20% digunakan sebagai data uji. Pada skenario kedua ini juga akan melalui dua kali proses percobaan penggunaan nilai *spatial dropout* dengan nilai 0,4 dan 0,5. Penggunaan nilai `batch_size`, nilai *dropout* dan *spatial dropout* yang ada pada Tabel 4.13.

Tabel 4.13 Perbandingan *Stopwords* dengan Dropout 0,4

Spatial Dropout	0,4			
Dropout	0,1	0,1	0,2	0,3

Percobaan pertama klasifikasi tanpa menggunakan *stopwords* mengaplikasikan nilai 0,4 pada *spatial dropout*. Kemudian data yang digunakan dibagi menjadi data latih sebesar 80% yang di dalamnya juga termasuk 20% data validasi. Sisa dari data latih yaitu sebesar 20% digunakan sebagai data uji. *Accuracy Train* tertinggi berada di kolom C2 dengan perolehan akurasi sebesar 0,8078. Pada *Accuracy validation* diperoleh hasil akurasi tertinggi sebesar 0,7801 pada kolom C2. Kemudian yang terakhir untuk *Accuracy test* tertinggi berada pada kolom C2 dengan hasil akurasi sebesar 0,7790. Percobaan 1 dapat dilihat pada Tabel 4.20.

Tabel 4.20 Percobaan 1 Tanpa Menggunakan *Stopwords* 80:20

	C1	**C2**	**C3**
Batch Size	**32**	**64**	**128**
Train	80	80	80
Test	20	20	20
Val split	0,2	0,2	0,2
Accuracy Train	0.7917	0.8078	0.7706
Accuracy Validation	0.7717	0.7801	0.7470
Accuracy Test	0.7610	0.7790	0.7463

Percobaan kedua klasifikasi tanpa menggunakan stopwords dengan pembagian data 90:10. Sebanyak 90% digunakan untuk data latih yang juga termasuk *validation set* sebanyak 10%. Pembagian 10% selanjutnya digunakan untuk data uji. *Accuracy Train* tertinggi pada percobaan keempat diperoleh oleh kolom D2 dengan akurasi sebesar 0,8033. Kolom D3 mendapatkan nilai *Accuracy Validation* tertinggi sebesar 0,7947 dengan selisih 0,0012 dari kolom D2. Perolehan *Accuract Test* tertinggi didapatkan oleh kolom D2 yaitu sebesar 0,8005. Percobaan 2 dapat dilihat pada Tabel 4.21.

Tabel 4.21 Percobaan 2 Tanpa Menggunakan *Stopwords* 90:10

	D1	**D2**	**D3**
Batch Size	**32**	**64**	**128**
Train	90	90	90
Test	10	10	10
Val_split	0,1	0,1	0,1
Accuracy Train	0.7953	0.8033	0.7889
Accuracy Validation	0.7772	0.7935	0.7947
Accuracy Test	0.7914	0.8005	0.7644

Pada percobaan ketiga dan keempat pada klasifikasi tanpa menggunakan *stopwords* mengaplikasikan nilai *spatial dropout* yang berbeda seperti pada Tabel 4.16. Hal ini dilakukan sebagai pembanding hasil antara hasil klasifikasi menggunakan nilai *spatial dropout* 0,4 dan 0,5.

Tabel 4.16 Perbandingan *Stopwords* dengan Dropout 0,5

Spatial Dropout	0,5			
Dropout	0,1	0,1	0,2	0,3

Percobaan ketiga adalah klasifikasi tanpa menggunakan *stopwords* dengan menggunakan *spatial dropout* 0,5. Hasil klasifikasi dapat dilihat pada Tabel 4.22. Nilai akurasi data latih tertinggi didapatkan pada kolom E1 sebesar 0,7870. Selanjutnya untuk nilai akurasi tertinggi untuk akurasi validasi di antara tiga percobaan didapatkan oleh kolom E1 dengan perolehan akurasi sebesar 0,7752. Kemudian yang terakhir untuk nilai akurasi tes tertinggi didapatkan oleh kolom E2 dengan akurasi sebesar 0,7835.

Tabel 4.22 Percobaan 3 Tanpa Menggunakan *Stopwords* 80:20

	E1	**E2**	**E3**
Batch Size	**32**	**64**	**128**
Train	80	80	80
Test	20	20	20
Val_split	0,2	0,2	0,2
Accuracy Train	0.7870	0.7831	0.7598
Accuracy Validation	0.7752	0.7724	0.7385
Accuracy Test	0.7616	0.7835	0.7587

Percobaan terkahir dilakukan dengan membagi *dataset* menjadi komposisi 90:10. Percobaan terakhir dilakukan tanpa menggunakan *stopwords*. Hasil dari pengklasifikasian dapat dilihat padaTabel 4.23. Akurasi tertinggi pada data latih didapatkan oleh kolom F2

dengan perolehan akurasi sebesar 0,7971. Kemudian untuk akurasi validasi dengan akurasi sebesar 0,7910 diperoleh oleh kolom F1. Tetapi, untuk akurasi data uji perolehan akurasi tertinggi sebesar 0,7847 diperoleh oleh kolom F2.

Tabel 4.23 Percobaan 3 Tanpa Menggunakan *Stopwords* 90:10

	F1	**F2**	**F3**
Batch Size	**32**	**64**	**128**
Train	90	90	90
Test	10	10	10
Val_split	0,1	0,1	0,1
Accuracy Train	0.7796	0.7971	0.7753
Accuracy Validation	0.7910	0.7860	0.7797
Accuracy Test	0.7756	0.7847	0.7666

Berdasarkan hasil percobaan skenario kalsifikasi tanpa menggunakan *stopwords*, pembagian data dengan komposisi 90:10, komposisi 80:20, percobaan tiga nilai *batch_size* dan percobaan *spatial dropout*. Didapatkan hasil terbaik seperti padaTabel 4.24. Hasil terbaik didapatkan dengan menggunakan *spatial dropout* dengan nilai 0,4 dengan *batch size* bernilai 64. Akurasi tinggi didapatkan dengan pembagian data dengan komposisi 90:10. Akurasi data latih atau *Accuracy Train* tertinggi berjumlah 0.8033 dengan akurasi validasi sebesar 0,7935.

Tabel 4.24 Hasil Terbaik Tanpa Menggunakan *Stopwords*

Spatial Dropout	***Batch_size***	***Train***	***Test***	***Val_split***	***Acc Train***	***Acc Val***	***Acc Test***
0,4	64	0,90	0,10	0,1	0.8033	0.7935	0.8005
0,5	64	0,90	0,10	0,1	0.7971	0.7860	0.7847

Berdasarkan hasil percobaan klasifikasi menggunakan *stopwords* dan tanpa menggunakan *stopwords* didapatkan hasil terbaik dari masing-masing percobaan. Hasil perbandingan klasifikasi dapat dilihat pada Tabel 4.25. Akurasi data latih tertinggi didapatkan oleh klasifikasi tanpa menggunakan *stopwords* dengan perolehan akurasi sebesar 0,8033. Dibandingkan dengan klasifikasi menggunakan *stopwords*, keduanya memiliki selisih sebesar 0,0017. Kemudian untuk hasil akurasi validasi tertinggi sebesar 0,7935 didapatkan oleh klasifikasi tanpa menggunakan *stopwords*. Kemudian untuk akurasi data tes tertinggi diperoleh oleh klasifikasi tanpa menggunakan *stopwords* dengan akurasi sebesar 0,8005. Selisih *accuracy test* dengan menggunakan *stopwords* dan tanpa menggunakan *stopwords* sebesar 0,0158.

Tabel 4.25 Perbandingan klasifikasi

Keterangan	***Spatial Dropout***	***Batch_size***	***Train***	***Test***	***Val_split***	***Acc Train***	***Acc Val***	***Acc Test***
Menggunakan *Stopwords*	0,4	64	0,90	0,10	0,1	0.8016	0.7810	0.7655
Tanpa Menggunakan *Stopwords*	0,4	64	0,90	0,10	0,1	0.8033	0.7935	0.8005

Setelah melakukan banyak percobaan, penelitian ini berkesimpulan untuk melakukan klasifikasi tanpa menggunakan penghapusan s*topwords* seperti Tabel 4.26. Nilai *spatial dropout* yang diaplikasikan bernilai 0,4 dengan *batch size* berukuran 64. Selain itu untuk pembagian data menggunakan komposisi 90:10. Sebanyak 90% data digunakan untuk data uji yang juga di dalamnya menjadi validasi set sebanyak 10%. Kemudian sisa 10% digunakan sebagai data uji. Komposisi tersebut menghasilkan akurasi data latih sebesar 0,8033 dengan akurasi validasi sebesar 0,7932. Pada model ini menghasilkan nilai akurasi tes sebesar 0,8005. Kode program ditunjukkan pada Gambar 4.42.

Tabel 4.26 Komposisi Klasifikasi

Keterangan	***Spatial Dropout***	***Batch_size***	***Train***	***Test***	***Val_split***	***Acc Train***	***Acc Val***	***Acc Test***
Tanpa Menggunakan *Stopwords*	0,4	64	0,90	0,10	0,1	0.8033	0.7935	0.8005

```
epochs = 10
batch_size = 64
model = Sequential()
model.add(Embedding(MAX_NUMBER_WORDS,
                    EMBEDDING_DIMENSION,
                    input_length=X.shape[1]))
model.add(SpatialDropout1D(0.4))
model.add(Bidirectional(LSTM(EMBEDDING_DIMENSION,
                             dropout=0.1,
                             recurrent_dropout=0.1)))
model.add(Dense(EMBEDDING_DIMENSION, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(6))
model.add(Dropout(0.3))
model.add(Activation('softmax'))
model.compile(loss='categorical_crossentropy',
              optimizer='adam', metrics=['accuracy'])
history = model.fit(X_train,
                    Y_train,
                    epochs=epochs,
                    batch_size=batch_size,
                    validation_split=0.1,
```

```
                    callbacks=[EarlyStopping(monitor='val_loss',
patience=3, min_delta=0.0001)])
```

Gambar 4.42 Kode Program Model

Setelah model dibangun, dilakukan percobaan klasifikasi menggunakan model tersebut dengan data masukan yang diambil secara *random* di Komunitas Marah-Marah. Postingan ini diambil pada tanggal 13 Agustus 2023. Berdasarkan hasil klasifikasi pada Gambar 4.44, model berhasil melakukan klasifikasi pada kalimat baru. Jika disesuaikan dengan kriteria dari setiap kategori label, kalimat 'CAPE BGT PUNYA KELUARGA ISINYA ORANG KAYAK KONTOL SEMUA' termasuk dalam kategori label Keluarga. Oleh karena itu, hasil klasifikasi dari model telah cocok dengan label yang sebenarnya. Kode program dapat dilihat pada Gambar 4.43.

```
tweet_percobaan = ['CAPE BGT PUNYA KELUARGA ISINYA ORANG KAYAK KONTOL
SEMUA']
sequence = tokenize_tweet.texts_to_sequences(tweet_percobaan)
padded = pad_sequences(sequence, maxlen=MAX_SEQ_LENGTH)
prediction_tweet = model.predict(padded)
labels =
['Studi','Percintaan','Keluarga','Karir/Pekerjaan','Person/Personal','
Tidak Diketahui Masalahnya ']
print(prediction_tweet, labels[np.argmax(prediction_tweet)])
```

Gambar 4.43 Kode Program Klasifikasi

```
tweet_percobaan = ['CAPE BGT PUNYA KELUARGA ISINYA ORANG KAYAK KONTOL SEMUA']
sequence = tokenize_tweet.texts_to_sequences(tweet_percobaan)
padded = pad_sequences(sequence, maxlen=MAX_SEQ_LENGTH)
prediction_tweet = model.predict(padded)
labels = ['Studi','Percintaan','Keluarga','Karir/Pekerjaan','Person/Personal','Tidak Diketahui Masalahnya']
print(prediction_tweet, labels[np.argmax(prediction_tweet)])
[22]

1/1 [==============================] - 1s 1s/step
[[1.4445443e-03 1.7462726e-03 9.9335831e-01 9.3685644e-04 1.9445254e-03
  5.6942314e-04]] Keluarga
```

Gambar 4.44 Hasil Klasifikasi

4.5 Evaluasi

Evaluasi model menggunakan *confusion matrix* adalah pendekatan yang penting dalam mengukur kinerja model klasifikasi. *Confusion matrix* adalah tabel yang menggambarkan hasil prediksi model dengan membandingkannya dengan nilai sebenarnya dari *dataset*. Evaluasi ini menggunakan data uji yang telah ditentukan di awal yaitu sebesar 10%. Dalam *confusion matrix*, empat kemungkinan hasil prediksi dikelompokkan menjadi empat kategori: *True Positive* (TP) menunjukkan prediksi benar positif, *True Negative* (TN) menunjukkan prediksi

benar negatif, *False Positive* (FP) menunjukkan prediksi salah positif, dan *False Negative* (FN) menunjukkan prediksi salah negatif. Dengan memperhatikan nilai-nilai ini, dapat dilakukan perhitungan matrik evaluasi penting seperti akurasi, presisi, *recall*, dan *F1-score*. *Confusion matrix* membantu mengidentifikasi jenis kesalahan yang dilakukan oleh model dan memberikan gambaran yang lebih komprehensif tentang kinerja model dalam mengklasifikasikan berbagai kategori. Kode program pada penelitian ini dapat dilihat pada Gambar 4.41.

```
def c_report(y_true, y_pred):
   print("Classification Report")
   print(classification_report(y_true, y_pred))
   acc_sc = accuracy_score(y_true, y_pred)
   print("Accuracy : "+ str(acc_sc))
   return acc_sc

def plot_confusion_matrix(y_true, y_pred):
   mtx = confusion_matrix(y_true, y_pred)
   sns.heatmap(mtx, annot=True, fmt='d', linewidths=.5,
               cmap="Blues", cbar=False)
   plt.ylabel('True label')
   plt.xlabel('Predicted label')
```

Gambar 4.45 Kode Program Matrik Konfusi

Tabel 4.27 merupakan hasil evaluasi data uji dari model *BiLSTM* (*Bidirectional Long Short-Term Memory*). Model yang sudah dibangun menggunakan *dataset* Komunitas Marah-Marah mendapatkan akurasi data latih sebesar 0,8033 dengan validasi akurasi sebesar 0,7935 dan akurasi data uji sebesar 0,8005. Berdasarkan dari Tabel 4.27 nilai *Precision* terbesar di antara enam kategori label diperoleh oleh kategori label Studi. Pada tabel *Recall* label Studi juga memperoleh nilai tertinggi dengan jumlah 0,85. Sehingga nilai *F1-Score* terbesar otomatis didapatkan oleh label Studi dengan perolehan 0,88. Hasil keseluruhan dari evaluasi dapat dikatakan baik berdasarkan perolehan nilai yang didapatkan. Gambar 4.46 menunjukan kode program untuk visualisasi *confusion matrix*.

Tabel 4.27 Hasil Evaluasi Data Uji

***Accuracy* 0.8005**				
Label	***Precision***	***Recall***	***F1- score***	***Support***
Studi	0.92	0.85	0.88	153
Percintaan	0.85	0.80	0.83	153
Keluarga	0.87	0.84	0.86	178
Karir/Pekerjaan	0.82	0.72	0.77	71
Person/Personal	0.64	0.65	0.65	168

Tidak Diketahui Masalahnya	0.87	0.79	0.83	164

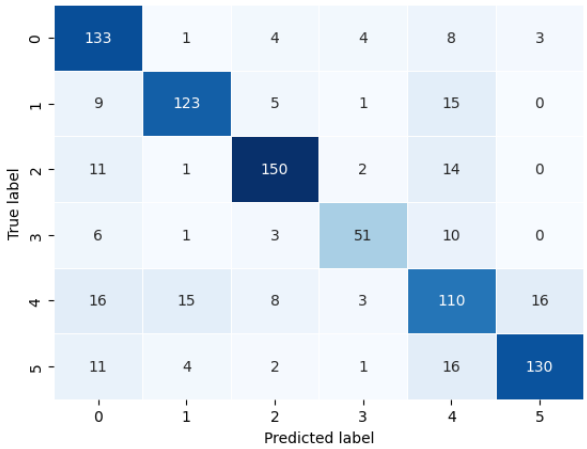
Pelabelan data pada penelitian ini dilakukan secara manual sehingga terjadinya kesalahan pelabelan atau *human error* sangat tinggi. Label Person/Personal mendapatkan nilai *precision, recall,* dan *f1-score* terendah daripada kategori label lainnya. Hal tersebut dikarenakan adanya kemungkinan kesalahan pelabelan atau *human error* untuk label Person/Personal. Artinya terdapat beberapa twit yang seharusnya berada pada label selain Person/Personal tetapi masuk ke dalam kategori label Person/Personal sehingga terjadi pelabelan data yang tidak akurat. Beberapa data twit yang ada pada label Person/Personal mengandung kata-kata kasar atau umpatan, sehingga model cenderung salah dalam mengklasifikasi. Ketika data pelatihan tidak akurat, model cenderung belajar dari informasi yang salah dan ini dapat mengurangi kemampuan model untuk membuat prediksi yang tepat.

Kebalikan dari label Person/Personal adalah label Studi yang memiliki nilai *precision, recall,* dan *f1-score* paling tinggi di antara label lainnya. Data dari label Studi berisi mengenai permasalahan yang berkaitan dengan lingkup akademik atau pendidikan. Pada kategori label Studi, data twit dilabeli dengan benar sehingga menghasilkan data yang akurat untuk pemodelan. Pada data twit label Studi, datanya murni mengandung permasalahan Studi tanpa ada kata-kata kasar atau kotor. Sehingga model tidak mengalami kesulitan dalam mengklasifikasi dan dapat memprediksi dengan benar.

```
plot_confusion_matrix(Y_test.argmax(axis=1), preds.argmax(axis=1))
```

Gambar 4.46 Kode Program Visualisasi *Confusion Matrix*

Pada Gambar 4.47 ditunjukkan hasil dari kode program evaluasi data uji. Pada gambar tersebut terdapat dua sisi, bagian sumbu x merupakan nilai prediksi, sedangkan sumbu y merupakan nilai asli dari klasifikasi. Pengindeksan label dimulai dari label 0, dan oleh karena itu, untuk setiap kategori labelnya meningkat sebesar 1 angka.



Gambar 4.47 Visualisasi dari *Confusion Matrix*

Pasa indeks 0 tertulis sebanyak 133 data dengan label 0 (kategori Studi) berhasil diprediksi dengan benar. Kemudian pada indeks 1 sebanyak 123 data dengan label 1 (label Percintaan) diprediksi dengan benar oleh model. Indeks ke 2 (label Keluarga) sebanyak 150 data berhasil diklasifikasikan dengan keluaran label 2 (label Keluarga) yang berarti berhasil diprediksi dengan benar. Selanjutnya indeks ke tiga (label Karier/Pekerjaan) mengklasifikasikan 57 data dengan benar menjadi label 3 (label Karier/Pekerjaan). Setelah itu, pada indeks ke-4 (label Person/Personal) mengklasifikasikan sebanyak 110 data dan berhasil diprediksi dengan benar. Untuk indeks yang terakhir yaitu indeks 5 (label Tidak Diketahui Masalahnya) berhasil melakukan prediksi dengan benar pada sebanyak 130 data. Berdasarkan Gambar 4.47 banyak terjadi kesalahan prediksi pada indeks ke-4 (label Person/Personal). Pada indeks pertama sebanyak 15 data berlabel 1 (label Percintaan) diprediksi salah oleh model menjadi label 4 (label Person/Personal). Selain itu, pada indeks ke-5, sebanyak 16 data berlabel 5 (label Tidak Diketahui Masalahnya) diprediksi salah oleh model menjadi label 4 (label Person/Personal). Kesalahan tersebut juga terjadi pada indeks 0 (label Studi) dengan jumlah data 8, indeks 2 (label Percintaan) dengan jumlah data 14, indeks 3 (label Karier/Pekerjaan) dengan jumlah data 10 yang seharusnya diprediksi menjadi label 0 (label Studi), label 2 (label Percintaan), dan label 3 (label Karier/Pekerjaan) oleh model diprediksi salah menjadi label 4 (label Person/Personal). Hal tersebut disebabkan oleh adanya ketidakseimbangan antara kelas, model cenderung memiliki kesulitan dalam mengklasifikasikan kelas minoritas dengan benar.

Mengacu pada hasil *scraping,* label Person/Personal dan label Tidak Diketahui Masalahnya mendapatkan jumlah data yang lebih banyak dari label-label lainnya. Selama empat periode *scraping* dua kategori tersebut selalu menduduki peringkat atas, sehingga

terdapat ketimpangan jumlah antar kategori label. Label Karier/Pekerjaan selalu mendapatkan data paling sedikit walaupun sudah dilakukan *scraping* selama empat periode waktu. Sehingga untuk mensiasati ketimpangan jumlah data, dilakukan proses *undersampling* yang jumlahnya disesuaikan dengan jumlah label terendah kedua yaitu label Studi sebanyak 1637. Hal tersebut dilakukan karena jika mengikuti jumlah dari label terendah pertama yaitu label Karier/Pekerjaan, penelitian ini akan mengalami kekurangan data karena jumlah data dari label Karier/Pekerjaan berada diangka 683. Data yang digunakan untuk pemodelan klasifikasi adalah data hasil *undersampling*. Untuk itu, dalam proses pengkalsifikasian dilakukan beberapa kali skenario pembagian data dan percobaan nilai *dropout* yang berbeda-beda. Hal tersebut dilakukan untuk meningkatkan nilai akurasi dan menghindari terjadinya *overfitting* pada model.

BAB V
KESIMPULAN & SARAN

Bab ini akan membahas ringkasan temuan yang dihasilkan dari analisis proses, algoritma, dan implementasi sistem dalam penelitian ini. Berdasarkan hasil penelitian, bab 5 ini akan memaparkan kesimpulan yang relevan, dan saran-saran akan diajukan untuk menyoroti perbaikan yang dapat dilakukan pada penelitian ini untuk masa depan.

5.1 Kesimpulan

Berdasarkan model klasifikasi yang sudah dibangun terhadap Komunitas Marah-Marah, dapat diperoleh kesimpulan sebagai berikut:

a. Proses klasifikasi melibatkan sejumlah tahapan, dimulai dari pengumpulan data, kemudian yang kedua yaitu, pelabelan data yang didalamnya juga termasuk proses penghapusan duplikasi data. Setelah itu dilakukan *text preprocessing*, pelaksanaan klasifikasi, dan yang terakhir adalah tahapan evaluasi model sebagai tahapan penutup dalam rangkaian proses klasifikasi.
b. Penelitian ini melakukan klasifikasi masalah menggunakan algoritma Bi-LSTM *(Bidirectional Long Short-Term Memory)* dengan sumber data yang digunakan berasal dari Komunitas Marah-Marah di Twitter. Hasil tingkat akurasi data latih sebesar 0,8033 dan akurasi data uji sebesar 0,8005.

5.2 Saran

Dari hasil analisis dan simpulan yang diperoleh, beberapa rekomendasi untuk pengembangan lebih lanjut dari sistem ini adalah sebagai berikut:

a. Penambahan jumlah *dataset* agar setiap label memiliki jumlah yang sama banyak, sehingga ketika proses pemodelan tidak perlu dilakukan *undersampling* yang mempengaruhi hasil dari proses pemodelan yaitu akurasi.
b. Penambahan periode waktu *scraping* dalam rentang waktu yang panjang, sehingga dapat didapatkan keseimbangan jumlah antar *dataset*.
c. Melakukan penambahan model klasifikasi sehingga dapat dilakukan perbandingan metode satu dengan lainnya.

DAFTAR PUSTAKA

Adiarso, H. N. P., Osmond, A. B., & Prasasti, A. L. (2019). PENENTUAN DIALEK JAWA MENGGUNAKAN METODE RECURRENT NEURAL NETWORK. *E-Proceeding of Engineering*, *6*(2), 5625.

Af'idah, D. I., Dairoh, D., Handayani, S. F., Pratiwi, R. W., & Sari, S. I. (2022). Sentimen Ulasan Destinasi Wisata Pulau Bali Menggunakan Bidirectional Long Short Term Memory. *MATRIK : Jurnal Manajemen, Teknik Informatika Dan Rekayasa Komputer*, *21*(3), 607–618. https://doi.org/10.30812/matrik.v21i3.1402

Ahluna, F., Tutuarima, C. J., & Santoso, I. (2023). METODE K-NEAREST NEIGHBOR UNTUK ANALISIS SENTIMEN TENTANG PENGHAPUSAN UJIAN NASIONAL. *Jurnal IKRAITH-INFORMATIKA*, *7*(2). https://www.instagram.com/p/B599fcnFxXR

Al-Shufi, M. F., & Erfina, A. (2021). *SENTIMEN ANALISIS MENGENAI APLIKASI STREAMING FILM MENGGUNAKAN ALGORITMA SUPPORT VECTOR MACHINE DI PLAY STORE*.

Amalia, J., Pakpahan, J., Pakpahan, M., & Panjaitan, Y. (2022). Model Klasifikasi Berita Palsu Menggunakan Bidirectional LSTM Dan Word2Vec Sebagai Vektorisasi. *Jurnal Teknik Informatika Dan Sistem Informasi*, *9*(4). http://jurnal.mdp.ac.id

Anggraeni, E., & Khasan, S. (2018). *PENGARUH MEDIA SOSIAL DAN STATUS SOSIAL EKONOMI ORANG TUA TERHADAP PERILAKU KONSUMTIF MAHASISWA*. http://journal.unnes.ac.id/sju/index.php/eeaj

Antinasari, P., Setya Perdana, R., & Fauzi, M. A. (2017). Analisis Sentimen Tentang Opini Film Pada Dokumen Twitter Berbahasa Indonesia Menggunakan Naive Bayes Dengan Perbaikan Kata Tidak Baku. *Jurnal Pengenmabangan Teknologi Informasi Dan Ilmu Komputer*, *1*(12), 1733–1741. http://j-ptiik.ub.ac.id

Ardiani, L., Sujaini, H., & Tursina, T. (2020). Implementasi Sentiment Analysis Tanggapan Masyarakat Terhadap Pembangunan di Kota Pontianak. *Jurnal Sistem Dan Teknologi Informasi (Justin)*, *8*(2), 183. https://doi.org/10.26418/justin.v8i2.36776

Arief, B., Kholifatullah, H., & Prihanto, A. (2023). Penerapan Metode Long Short Term Memory Untuk Klasifikasi Pada Hate Speech. *Journal of Informatics and Computer Science*, *04*.

Artika, I. W. (2021). KOMUNITAS SASTRA DI MEDIA SOSIAL DAN KAITANNYA DENGAN KEGIATAN LITERASI DI SEKOLAH. *Seminar Bahasa, Sastra Dan Pengajarannya(PEDALITRA I)* .

Astajaya, I. K. manik. (2020). ETIKA KOMUNIKASI DI MEDIA SOSIAL. *Widya Duta*, *15*.

Bagus, A. T. (2022). *KLASIFIKASI EMOSI PADA TEKS MENGGUNAKAN METODE DEEP LEARNING*.

Bastomi, H. (2020). Pemetaan Masalah Belajar Siswa… Pemetaan Masalah Belajar Siswa SMK Negeri 3 Yogyakarta Dan Penyelesaiannya (Tinjauan Srata Kelas). In *Konseling Edukasi: Journal of Guidance and Counseling*.

Budiman, A. A. (2018). *Pendeteksi Bahasa Daerah pada Twitter dengan Machine Learning*.

Budiman, A., Suryadibrata, A., & Young, J. C. (2021). *Implementasi Algoritma Naïve Bayes untuk Klasifikasi Konten Twitter dengan Indikasi Depresi*. *6*(1).

Cahyono, A. S. (2018). *DAMPAK MEDIA SOSIAL TERHADAP PERMASALAHAN SOSIAL ANAK*.

Cai, J., Li, J., Li, W., & Wang, J. (2018). *DEEPLEARNING MODEL USED IN TEXT CLASSIFICATION*. IEEE.

Christianto, M., Andjarwirawan, J., & Tjondrowiguno, A. (2020). Aplikasi Analisa Sentimen Pada Komentar Berbahasa Indonesia Dalam Objek Video di Website YouTube Menggunakan Metode Naïve Bayes Classifier. *JURNAL INFRA*, *8*(1).

Devita, R. N., Herwanto, H. W., & Wibawa, A. P. (2018). Perbandingan Kinerja Metode Naive Bayes dan K-Nearest Neighbor untuk Klasifikasi Artikel Berbahasa indonesia. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *5*(4), 427. https://doi.org/10.25126/jtiik.201854773

Dirgantara, A. R. S. (2022a). *DETEKSI KALIMAT MENGGUNAKAN METODE BIDIRECTIONAL LSTM--STUDI KASUS:INDONESIA DAN MALAYSIA*.

Dirgantara, A. R. S. (2022b). *DETEKSI KALIMAT MENGGUNAKAN METODE BIDIRECTIONAL LSTM--STUDI KASUS:INDONESIA DAN MALAYSIA*.

Dwiyansaputra, R., Nugraha, G. S., Bimantoro. Fitri, & Aranta, A. (2021). DETEKSI SMS SPAM BERBAHASA INDONESIA MENGGUNAKAN TF-IDF DAN STOCHASTIC GRADIENT DESCENT CLASSIFIER. *JTIKA*, *3*(2). http://jtika.if.unram.ac.id/index.php/JTIKA/

Fachreza, Moch. R. D. (2023). *KLASIFIKASI SENTIMEN MASYARAKAT TERHADAP PROSES PEMINDAHAN IBU KOTA NEGARA (IKN) INDONESIA PADA MEDIA SOSIAL TWITTER MENGGUNAKAN METODE NAÏVE BAYES*. http://etheses.uin-malang.ac.id/53766/

Fadli, H. F., & Hidayatullah, A. F. (2021). Identifikasi Cyberbullying pada Media Sosial Twitter Menggunakan Metode LSTM dan BiLSTM. *AUTOMATA*, *2*(1).

Faisal, A., Alkhalifi, Y., Rifai, A., & Gata, W. (2018). Analisis Sentimen Dewan Perwakilan Rakyat Dengan Algoritma Klasifikasi Berbasis Particle Swarm Optimization. *JOINTECS(Journal of Information Technology and Computer Science*, *7*(1), 61–70.

Fajar, R. (2018). Implementasi Algoritma Naive Bayes Terhadap Analisis Sentimen Opini Film Pada Twitter. *INOVTEK POLBENG*, *3*(1).

Fajriyanti, N. A. (20222). *BIMBINGAN ORANG TUA DALAM MEMBENTUKPENGENDALIAN EMOSI MARAH ANAK PADAMASA PANDEMI COVID-19 DI KELURAHAN BOBOSAN*.

Febrianti, N. A., Hidayat, S., & Ikhsan, A. (2021). PEMBUATAN WEB BACKEND UNTUK WEBSITE COMPANY PROFILE RA BAHRUL ULUM. *Jurnal Inovatif*, *4*(1). https://doi.org/https://doi.org/10.32832/inova-tif.v4i1.5474

Fikri, M. I., Sabrila, T. S., & Azhar, Y. (2020). Perbandingan Metode Naïve Bayes dan Support Vector Machine pada Analisis Sentimen Twitter. *SMATIKA*, *10*, 71.

Fitrinanda, G., & Djunaidy, A. (2022). Peramalan Harga Saham PT Adaro Energy Indonesia Tbk yang Mempertimbangkan Faktor Kurs Dolar Amerika Menggunakan Bidirectional Long-Short Term Memory. *JURNAL TEKNIKITS*, *11*(3).

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

Ginanjar, A. Y. (2019). *Pentingnya Penguasaan Konsep Matematika Dalam Pemecahan Masalah Matematika di SD*. www.jurnal.uniga.ac.id

Gunawan, R., Rahmatulloh, A., Darmawan, I., & Firdaus, F. (2019). *Comparison of Web Scraping Techniques: Regular Expression, HTML DOM and Xpath*. http://testing-ground.scraping.pro.

Gupta, S., Goel, L., Singh, A., Prasad, A., & Ullah, M. A. (2022). Psychological Analysis for Depression Detection from Social Networking Sites. *Computational Intelligence and Neuroscience*, *2022*. https://doi.org/10.1155/2022/4395358

Handayani, R. N. (2021). OPTIMASI ALGORITMA SUPPORT VECTOR MACHINE UNTUK ANALISIS SENTIMEN PADA ULASAN PRODUK TOKOPEDIA MENGGUNAKAN PSO. *Media Informatika*, *20*(2).

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

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

Hemanto, Mustopa, A., & Kuntoro, A. Y. (2020a). ALGORITMA KLASIFIKASI NAIVE BAYES DAN SUPPORT VECTOR MACHINE DALAM LAYANAN KOMPLAIN MAHASISWA. *JURNAL ILMU PENGETAHUAN DAN TEKNOLOGI KOMPUTER*, *5*(2). www.bsi.ac.id

Hemanto, Mustopa, A., & Kuntoro, A. Y. (2020b). ALGORITMA KLASIFIKASI NAIVE BAYES DAN SUPPORT VECTOR MACHINE DALAM LAYANAN KOMPLAIN MAHASISWA. *JURNAL ILMU PENGETAHUAN DAN TEKNOLOGI KOMPUTER*, *5*(2). www.bsi.ac.id

Hidayatullah, M. R., & Maharani, W. (2022). Depression Detection on Twitter Social Media Using Decision Tree. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *6*(4), 677–683. https://doi.org/10.29207/resti.v6i4.4275

Ihsan, M., Negara, B. S., & Agustian, S. (2022). LSTM (Long Short Term Memory) for Sentiment COVID-19 Vaccine Classification on Twitter. *Digital Zone: Jurnal Teknologi Informasi Dan Komunikasi*, *13*(1), 79–89. https://doi.org/10.31849/digitalzone.v13i1.9950

Indrawati, B. (2020). Tantangan Dan Peluang Pendidikan Tinggi Dalam Masa Dan Pasca Pandemi Covid-19. *Jurnal Kajian Ilmiah (JKI)*, *1*, 39–48. http://ejurnal.ubharajaya.ac.id/index.php/JKI

Indrayuni, E. (2019). *Klasifikasi Text Mining Review Produk Kosmetik Untuk Teks Bahasa Indonesia Menggunakan Algoritma Naive Bayes*. *VII*(1).

Karyadi, Y., & Santoso, H. (2022). Prediksi Kualitas Udara Dengan Metoda LSTM, Bidirectional LSTM, dan GRU. *JATISI*, *9*, 671–684.

Khder, M. A. (2021). Web scraping or web crawling: State of art, techniques, approaches and application. *International Journal of Advances in Soft Computing and Its Applications*, *13*(3), 144–168. https://doi.org/10.15849/ijasca.211128.11

Kristian, Y., Purnama, K. E., Sutanto, E. H., Zaman, L., Setiawan, E. I., & Purnomo, M. H. (2018). Klasifikasi Nyeri pada Video Ekspresi Wajah Bayi Menggunakan DCNN Autoencoder dan LSTM. In *JNTETI* (Vol. 7, Issue 3).

Maliki, M. A., Cholissodin, I., & Yudistira, N. (2022). Prediksi Pergerakan Harga Cryptocurrency Bitcoin terhadap Mata Uang Rupiah menggunakan Algoritme LSTM. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *6*(7), 3259–3268. http://j-ptiik.ub.ac.id

Mardiana, L., & Zi'ni, A. F. (2020). *PENGUNGKAPAN DIRI PENGGUNA AKUN AUTOBASE TWITTER @SUBTANYARL*.

Marinda, D. E., & Al Amin, I. H. (2023). Implementasi Metode Convolutional Neural Network untuk Deteksi Penggunaan Masker secara Real-Time. *Jurnal TeknikInformatika Unika ST. Thomas (JTIUST)*, *8*(1), 2657–1501.

Marta, D., Leonarde Ginting, G., & Hatuaon Sihite, A. (2022). Deteksi Berita Palsu Tentang Vaksinasi Covid-19 Dengan Menggunakan Text Mining Dan Algoritma Cosine Similarity. *Nasional Teknologi Informasi Dan Komputer)*, *6*(1). https://doi.org/10.30865/komik.v6i1.5738

Maulani, N. M., & Priyambodo, A. B. (2021). *Pengungkapan Diri pada Pengguna Akun Alter Twitter Dewasa Awal di Kota Malang*.

Mauro, A. De, Greco, M., Grimaldi, M., & Ritala, P. (2018). Human resources for Big Data professions: A systematic classification of job roles and required skill sets. *Information Processing and Management*, *54*(5), 807–817. https://doi.org/10.1016/j.ipm.2017.05.004

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., & Gao, J. (2020). *Deep Learning Based Text Classification: A Comprehensive Review*. *1*(1).

Mukhlis, M., Kustiyo, A., & Suharso, A. (2021). Peramalan Produksi Pertanian Menggunakan Model Long Short-Term Memory. *BINA INSANI ICT JOURNAL*, *8*(1), 22–32.

Munawir. (2023). Fungsi Komunikasi Dalam Memberikan Informasi, Mendidik, Menghibur dan Mempengaruhi (Studi Kajian Hadits Tematik). *UNIVERSAL GRACE JOURNAL: SCIENTIFIC MULTIDISCIPLINARY*, *1*. https://id.wikipedia.org

Mustofa, H., & Mahfudh, A. A. (2019). Klasifikasi Berita Hoax Dengan Menggunakan Metode Naive Bayes. *Walisongo Journal of Information Technology*, *1*(1), 1. https://doi.org/10.21580/wjit.2019.1.1.3915

Nandwani, P., & Verma, R. (2021). A review on sentiment analysis and emotion detection from text. *Social Network Analysis and Mining*, *11*(1). https://doi.org/10.1007/s13278-021-00776-6

Naquitasia, R. (2021). *ANALISIS SENTIMEN BERBASIS ASPEK PADA WISATA HALAL DENGAN DEEP LEARNING*.

Norhikmah, & Rumini. (2020). KLASIFIKASI PEMINJAMAN BUKU MENGGUNAKAN NEURAL NETWORK BACKPROPAGATION. *Jurnal Sistem Informasi*, *9*(1), 1–15.

Nugraha, I. D., & Azhar, Y. (2022). DETEKSI DEPRESI PENGGUNA TWITTER INDONESIA MENGGUNAKAN LSTM-RNN. *JANAPATI*, *11*(3). https://doi.org/10.23887/janapati.v11i3.50674

Nugraha, M. (2018). MANAJEMEN KELAS DALAM MENINGKATKAN PROSES PEMBELAJARAN. *Tarbawi*, *4*(01), 27–44. http://jurnal.uinbanten.ac.id/index.php/tarbawi

Nugroho, K. S., Akbar, I., & Suksmawati, A. N. (2021a). *DETEKSI DEPRESI DAN KECEMASAN PENGGUNA TWITTER MENGGUNAKAN BIDIRECTIONAL LSTM*.

Nugroho, K. S., Akbar, I., & Suksmawati, A. N. (2021b). *DETEKSI DEPRESI DAN KECEMASAN PENGGUNA TWITTER MENGGUNAKAN BIDIRECTIONAL LSTM*.

Nurhaliza, W. O. S., & Fauziah, N. (2020). Komunikasi Kelompok dalam Virtual Community. *Komunida: Media Komunikasi Dan Dakwah*, *10*(1), 18–38. https://doi.org/10.35905/komunida.v7i2

Ompusunggu, A. (2022). *ANALISIS KEMAMPUAN PEMECAHAN MASALAH MATEMATIS PESERTA DIDIK KELAS VII SMP ADHYAKSA MEDAN PADA MATERI ARITMATIKA SOSIAL*. http://repository.uhn.ac.id/handle/123456789/7505

Pachouly, S. J., Raut, G., Bute, K., Tambe, R., Bhavsar, S., & Students, U. (2021). Depression Detection on Social Media Network (Twitter) using Sentiment Analysis. *International Research Journal of Engineering and Technology*. www.irjet.net

Palma, B. K., Murdiansyah, D. T., & Astuti, W. (2021a). Klasifikasi Teks Artikel Berita Hoaks Covid-19 dengan Menggunakan Algotrima K-Nearest Neighbor. *E-Proceeding of Engineering*, *8*(5).

Palma, B. K., Murdiansyah, D. T., & Astuti, W. (2021b). Klasifikasi Teks Artikel Berita Hoaks Covid-19 dengan Menggunakan Algotrima K-Nearest Neighbor. *E-Proceeding of Engineering*, *8*(5).

Pamungkas, F. S., & Kharisudin, I. (2021a). Analisis Sentimen dengan SVM, NAIVE BAYES dan KNN untuk Studi Tanggapan Masyarakat Indonesia Terhadap Pandemi Covid-19 pada Media Sosial Twitter. *PRISMA, Prosiding Seminar Nasional Matematika*, *4*, 628–634. https://journal.unnes.ac.id/sju/index.php/prisma/

Pamungkas, F. S., & Kharisudin, I. (2021b). Analisis Sentimen dengan SVM, NAIVE BAYES dan KNN untuk Studi Tanggapan Masyarakat Indonesia Terhadap Pandemi Covid-19 pada Media Sosial Twitter. *PRISMA, Prosiding Seminar Nasional Matematika*, *4*, 628–634. https://journal.unnes.ac.id/sju/index.php/prisma/

Prabowo, Y. D., Lesmana Marselino, T., & Suryawiguna, M. (2019). Pembentukan Vector Space Model Bahasa Indonesia Menggunakan Metode Word to Vector. *Jurnal Buana Informatika*, *10*(1). https://doi.org/https://doi.org/10.24002/jbi.v10i1.2053

Pramunendar, R. A., Prabowo, D. P., & Megantara, R. A. (2022). METODE RECURRENT NEURAL NETWORK (RNN) DENGAN ARSITEKTUR LSTM UNTUK ANALISIS SENTIMEN OPINI PUBLIK TERKAIT VAKSIN COVID-19. *JURNAL INFORMATIKA UPGRIS*, *8*(1). https://doi.org/https://doi.org/10.26877/jiu.v8i1.11599

Prasetyanwar, H., & Jondri. (2018). Peramalan Nilai Tukar IDR-USD Menggunakan Long Short Term Memory. *E-Proceeding of Engineering*, *5*(2), 3820.

Puspitarini, D. S., & Nuraeni, R. (2019). PEMANFAATAN MEDIA SOSIAL SEBAGAI MEDIA PROMOSI (Studi Deskriptif pada Happy Go Lucky House). In *Jurnal Common* | (Vol. 3).

Pusporani, E., Qomariyah, S., & Irhamah. (2019a). Klasifikasi Pasien Penderita Penyakit Liver dengan Pendekatan Machine Learning. *INFERENSI*, *2*(1).

Pusporani, E., Qomariyah, S., & Irhamah. (2019b). Klasifikasi Pasien Penderita Penyakit Liver dengan Pendekatan Machine Learning. *INFERENSI*, *2*(1).

Putra, O. V., Musthafa, A., & Wibowo, K. P. (2021a). Klasifikasi Ekspresi Teks Berbahasa Jawa Menggunakan Algoritma Long Short Term Memory. *Komputika : Jurnal Sistem Komputer*, *10*(2), 137–143. https://doi.org/10.34010/komputika.v11i1.4616

Putra, O. V., Musthafa, A., & Wibowo, K. P. (2021b). Klasifikasi Ekspresi Teks Berbahasa Jawa Menggunakan Algoritma Long Short Term Memory. *Komputika : Jurnal Sistem Komputer*, *10*(2), 137–143. https://doi.org/10.34010/komputika.v11i1.4616

Putra, P., Pardede, A. M. H., & Syahputra, S. (2022). ANALISIS METODE K-NEAREST NEIGHBOUR (KNN) DALAM KLASIFIKASI DATA IRIS BUNGA. *Jurnal Teknik Informatika Kaputama (JTIK)*, *6*(1).

Putri, A. N. (2019). *PROBLEM SOLVING PADA REMAJA PEREMPUAN YANG TERJADI DI PANTI ASUHAN*.

Putri, O. N. (2020). *IMPLEMENTASI METODE CNN DALAM KLASIFIKASI GAMBAR JAMUR PADA ANALISIS IMAGE PROCESSING*.

Rahutomo, F., Saputra, P. Y., & Fidyawan, M. A. (2018). IMPLEMENTASI TWITTER SENTIMENT ANALYSIS UNTUK REVIEW FILM MENGGUNAKAN ALGORITMA SUPPORT VECTOR MACHINE. *Jurnal Informatika Polinema*, *4*(2). https://doi.org/https://doi.org/10.33795/jip.v4i2.152

Ridwan, A. (2020). Penerapan Algoritma Naïve Bayes Untuk Klasifikasi Penyakit Diabetes Mellitus. *Jurnal Sistem Komputer Dan Kecerdasan Buatan*, *4*(1).

Sani, D. A., & Sarwani, M. Z. (2022). Koreksi Jawaban Esai Berdasarkan Persamaan Makna Menggunakan Fasttext dan Algoritma Backpropagation. *Jurnal Nasional Pendidikan Teknik Informatika (JANAPATI)*, *11*(2), 92–111. https://doi.org/10.23887/janapati.v11i2.49192

Sari, Y., & Prasetya, H. (2022). LITERASI MEDIA DIGITAL PADA REMAJA, DITENGAH PESATNYA PERKEMBANGAN MEDIA SOSIAL. *Jurnal Dinamika Ilmu Komunikasi*, *8*(1), 12–25.

Sastria, Y. A. G., Hidayanto, E., & Irawati, S. (2022). *IMPLEMENTASI MODEL PROBLEM BASED LEARNING (PBL) UNTUK MENINGKATKAN KEMAMPUAN PEMECAHAN MASALAH SISWA KELAS XI*. *4*(2). http://journal2.um.ac.id/index.php/jkpm

Setyawati, H., & Sumekto, D. R. (2022). *ASPEK PSIKOLOGIS DAN EKSPRESI UMPATAN OLAHRAGAWAN*. https://doi.org/https://doi.org/10.1529/kp.v1i3.54

Sihombing, J. (2021). Klasifikasi Data Antroprometri Individu Menggunakan Algoritma Naïve Bayes Classifier. *BIOS : Jurnal Teknologi Informasi Dan Rekayasa Komputer*, *2*(1), 1–10. https://doi.org/10.37148/bios.v2i1.15

Suarez, H. A., Perez, S. G., Medina, T. K., Hernandez, M. V., Sanchez, V., & Meana, P. H. (2018). *A Web Scraping Methodology for Bypassing Twitter API Restrictions*. http://arxiv.org/abs/1803.09875

Suharno, C. F., Fauzi, M. A., & Perdana, R. S. (2017). Klasifikasi Teks Bahasa Indonesia pada Dokumen Pengaduan Sambat Online menggunakan Metode K-Nearest Neighbors (K-NN) dan Chi-Square. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *1*(10), 1000–1007. http://j-ptiik.ub.ac.id

Suharso, P. (2019). Pemanfaatan Drone Emprit dalam Melihat Trend Perkembangan Bacaan Digital melalui Akun Twitter. *ANUVA*, *3*(4), 333–346.

Syadid, F. (2019). *Analisis sentimen komentar netizen terhadap calon presiden Indonesia 2019 dari twitter menggunakan algoritma term frequency-invers document frequency (tf-idf) dan metode multi layer perceptron (mlp) neural network*. http://repository.uinjkt.ac.id/dspace/handle/123456789/48184

Syahra, A. (2021). *PENERAPAN KONSELING KELOMPOK DENGAN TEKNIK REFLEKSI PERASAAN UNTUK MENGENDALIKAN EMOSI MARAH SISWA MAN 1 BENER MERIAH*.

Syauqi, R. F., Purnaningsih, N., & Korespondensi, P. (2020). Penggunaan Internet di Kalangan Petani Talas dalam Memperoleh Informasi Pertanian pada Kelompok Tani Saluyu, Situgede, Bogor. *Jurnal Pusat Inovasi Masyarakat Juli*, *2020*(5), 782–787.

Talita, A. S., & Wiguna, A. (2019). Implementasi Algoritma Long Short-Term Memory (LSTM) Untuk Mendeteksi Ujaran Kebencian (Hate Speech) Pada Kasus Pilpres 2019. *MATRIK : Jurnal Manajemen, Teknik Informatika Dan Rekayasa Komputer*, *19*(1), 37–44. https://doi.org/10.30812/matrik.v19i1.495

Tasik, B. B., Karlina, Sapu', N., & Wulandari, D. (2022). PERAN PENALARAN LOGIKA DALAM PEMECAHAN MASALAH PAMALI DI LEMBANG RATTE KECAMATAN MASANDA. *Enggang: Jurnal Pendidikan, Bahasa, Sastra, Seni, Dan Budaya*.

Tineges, R., Triayudi, A., & Sholihati, I. D. (2020). Analisis Sentimen Terhadap Layanan Indihome Berdasarkan Twitter Dengan Metode Klasifikasi Support Vector Machine (SVM). *JURNAL MEDIA INFORMATIKA BUDIDARMA*, *4*(3), 650. https://doi.org/10.30865/mib.v4i3.2181

Ulfah, A. N., & Anam, M. K. (2020). Analisis Sentimen Hate Speech Pada Portal Berita Online Menggunakan Support Vector Machine (SVM). *Jurnal Teknik Informatika Dan Sistem Informasi* , *7*(1), 1–10. http://jurnal.mdp.ac.id

Waasiu, A., Ilham B, A., & Lawi, A. (2021). *Klasifikasi Audio Cats and Dogs Menggunakan Model Artifical Neural Network Multi-perceptron*.

Widhiyasana, Y., Semiawan, T., Gibran, I., Mudzakir, A., & Noor, M. R. (2021). Penerapan Convolutional Long Short-Term Memory untuk Klasifikasi Teks Berita Bahasa Indonesia (Convolutional Long Short-Term Memory Implementation for Indonesian News Classification). In *Jurnal Nasional Teknik Elektro dan Teknologi Informasi* | (Vol. 10, Issue 4).

Wiharto, D. H., Chandra, F. H., Setiawan, E. I., & Santoso, J. (2021). Akuisisi Relasi Semantic Antar Entity Untuk Pembuatan Test Generator Padanan Kata Pada Ujian TPA dengan menggunakan Algoritma Espresso. *The Journal on Machine Learning and Computational Intelligence (JMLCI)*, *1*(1). https://doi.org/https://doi.org/10.26740/vol1iss1y2021id5

Yu, J., Marujo, L., Jiang, J., Karuturi, P., & Brendel, W. (n.d.). *Improving multi-label emotion classification via sentiment classification with dual attention transfer network*. Association for Computational Linguistics. https://ink.library.smu.edu.sg/sis_research/4279

Yuliana, D., & Supriyanto, C. (2019). *Klasifikasi Teks Pengaduan Masyarakat Dengan Menggunakan Algoritma Neural Network*. *5*(3), 92–116. https://doi.org/10.29165/komtekinfo.v5i2

Yunanto, R., Purfini, A. P., & Prabuwisesa, A. (2021). Survei Literatur: Deteksi Berita Palsu Menggunakan Pendekatan Deep Learning. *Jurnal Manajemen Informatika (JAMIKA)* . https://doi.org/10.34010/jamika.v11i2.493

Yuniasti, K. R. (2021). Pemanfaatan Akun Twitter @ARMYTEAMIID Sebagai Media Komunikasi Di Kalangan Fans BTS (ARMY). *Jurnal Penelitian Pers Dan Komunikasi Pembangunan*, *25*(2), 198–216. https://doi.org/10.46426/jp2kp.v25i2.168

Zahara, S., Sugianto, & Ilmiddafiq, M. B. (2019). Prediksi Indeks Harga Konsumen Menggunakan Metode Long Short Term Memory (LSTM) Berbasis Cloud Computing. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)* , *3*(3), 357–363.

Zhao, B. (2017). Web Scraping. In *Encyclopedia of Big Data* (pp. 1–3). Springer International Publishing. https://doi.org/10.1007/978-3-319-32001-4_483-1

Zukhrufillah, I. (2018). *Gejala Media Sosial Twitter Sebagai Media Sosial Alternatif* (Vol. 1, Issue 2). https://id.wikipedia.org/wiki/Twitter,