PENERAPAN *GRID SEARCH* UNTUK OPTIMASI MODEL *MACHINE LEARNING* DALAM KLASIFIKASI SENTIMEN KOMENTAR *YOUTUBE*

Mahendra Bayu Prayoga, Nuri Cahyono, Subektiningsih, Kamarudin
Informatika, Universitas AMIKOM Yogyakarta
Jl. Ring Road Utara, Condong Catur, Sleman, Yogyakarta, Indonesia
mbayup@students.amikom.ac.id

ABSTRAK

YouTube bukan hanya *platform* berbagi video, tetapi juga ruang interaksi sosial melalui komentar yang mencerminkan opini dan persepsi pengguna. Penelitian ini bertujuan untuk menganalisis kinerja algoritma klasifikasi sentimen, yaitu *Support Vector Machine (SVM)*, *Logistic Regression*, dan *Random Forest*, yang dioptimasi menggunakan *Grid Search* untuk klasifikasi sentimen komentar *YouTube*. Dataset yang digunakan terdiri dari 12.000 komentar dari 12 topik berbeda, meliputi politik, pendidikan, hiburan, teknologi, dan lainnya. Data diproses melalui *preprocessing* yang meliputi *unique handling*, *case folding*, *text cleaning*, *slang words normalization*, *stopwords removal*, *tokenizing*, dan *stemming* menggunakan *Sastrawi*. Pelabelan sentimen dilakukan dengan menggunakan kamus sentimen dalam tiga kelas: positif, negatif, dan netral. *Oversampling* dengan *SMOTE* diterapkan untuk mengatasi ketidakseimbangan data. Hasil penelitian menunjukkan bahwa optimasi menggunakan *Grid Search* meningkatkan akurasi model: *SVM* meningkat dari 90,97% menjadi 92,55%, *Logistic Regression* dari 86,72% menjadi 92,34%, dan *Random Forest* dari 87,33% menjadi 88,84%. Model *SVM* dengan kombinasi parameter *'C'*: 10, *'kernel'*: *'linear'*, *'max_iter'*: 1000, *'tol'*: 0.0001 menunjukkan performa terbaik dibandingkan model lain yang sudah dioptimasi.

Kata kunci : *Grid Search, Klasifikasi Sentimen, Machine Learning, Optimasi Model, YouTube*

1. PENDAHULUAN

YouTube bukan hanya platform untuk berbagi dan menonton video, tetapi juga ruang interaksi sosial yang memungkinkan pengguna untuk meninggalkan komentar dan berkomunikasi dengan sesama pengguna. Komentar-komentar ini menjadi wadah ekspresi opini, mencerminkan respons dan persepsi pengguna terhadap konten yang ditonton [1].

Klasifikasi terhadap komentar-komentar tersebut sangat penting untuk memahami pola sentimen, membantu kreator konten, pelaku bisnis, dan peneliti dalam menganalisis tren opini publik.

Mengelola dan mengklasifikasi komentar *YouTube* menjadi tantangan karena volumenya besar dan mengandung sentimen berharga [2].

Tanpa teknik analisis yang tepat, wawasan ini bisa terabaikan, menyulitkan pemahaman opini publik. Kompleksitas bahasa, seperti slang, emotikon, dan konteks budaya, makin memperumit klasifikasi sentimen. Oleh karena itu, penelitian ini krusial untuk mengembangkan metode klasifikasi yang lebih efisien dan dapat diterapkan dalam pemantauan opini publik dan analisis tren media sosial.

Untuk mengatasi tantangan tersebut, penelitian ini menerapkan *Grid Search* sebagai metode optimasi untuk meningkatkan akurasi klasifikasi. *Grid Search* memungkinkan pencarian kombinasi parameter terbaik untuk model machine learning, sehingga dapat meningkatkan hasil prediksi [3].

Dengan menggunakan *Grid Search*, berbagai kombinasi *hyperparameter* diuji secara sistematis untuk menemukan konfigurasi terbaik. Proses ini membantu menghindari pemilihan parameter secara acak dan meningkatkan generalisasi model melalui validasi silang (*cross-validation*). Dengan demikian, optimasi yang dilakukan dapat meningkatkan akurasi dan stabilitas model dalam klasifikasi [4].

Algoritma seperti *SVM (Support Vector Machine)*, *Logistic Regression*, dan *Random Forest* terbukti efektif digunakan dalam klasifikasi sentimen [5], [6], [7].

Penelitian ini mengoptimalkan performa ketiga algoritma tersebut menggunakan *Grid Search* pada komentar *YouTube*. Selain itu, penelitian ini membandingkan performa model sebelum dan sesudah optimasi untuk menilai efektivitasnya. Diharapkan hasilnya dapat meningkatkan akurasi analisis sentimen otomatis serta mendukung pemanfaatan data komentar *YouTube* secara optimal.

2. TINJAUAN PUSTAKA

2.1. Penelitian Terdahulu

Penelitian oleh Azkiyatun Nadroh melakukan klasifikasi status gizi balita menggunakan algoritma *SVM* yang dioptimasi dengan *Grid Search Cross-Validation*. Studi ini bertujuan untuk meningkatkan akurasi prediksi status gizi balita berdasarkan data antropometri dari Posyandu Desa Jagalempeni. Setelah optimasi *Grid Search* dengan kernel *RBF*, akurasi meningkat dari 79,29% menjadi 85,71%. Penelitian ini sejalan dengan penelitian yang dilakukan, karena menerapkan metode serupa untuk meningkatkan performa model [3].

Selanjutnya, penelitian oleh Tri Sugihartono melakukan analisis sentimen terhadap ulasan aplikasi *JKN Mobile* menggunakan algoritma *SVM* yang dioptimasi dengan *Grid Search*. Penelitian ini menunjukkan bahwa optimasi dengan *Grid Search*

dapat secara signifikan meningkatkan akurasi model. Penelitian ini sejalan dengan fokus penelitian yang dilakukan, yang juga menggunakan *Grid Search* untuk meningkatkan performa model dalam klasifikasi sentimen [4].

Dari penelitian yang telah dilakukan, terlihat bahwa teknik optimasi *Grid Search* berperan penting dalam meningkatkan performa klasifikasi sentimen. Penelitian ini akan menerapkan algoritma *SVM*, *Logistic Regression*, dan *Random Forest* yang dioptimasi dengan *Grid Search* untuk mencari kombinasi parameter terbaik. Dengan pendekatan ini, diharapkan model dapat menghasilkan prediksi yang lebih akurat dalam klasifikasi sentimen komentar *YouTube*.

2.2. *YouTube*

YouTube adalah salah satu platform berbagi video terbesar di dunia dengan jutaan pengguna aktif setiap harinya. Selain sebagai sarana untuk mengunggah dan menonton *video*, *YouTube* juga menyediakan fitur interaksi seperti komentar, yang menjadi ruang bagi pengguna untuk mengungkapkan opini mereka. Komentar-komentar ini berisi sentimen yang beragam dan memiliki potensi untuk dianalisis menggunakan *sentiment analysis* guna memahami reaksi audiens terhadap suatu konten. Proses ini biasanya memanfaatkan teknik seperti *natural language processing* (*NLP*) dan *machine* learning untuk mengklasifikasikan sentimen, baik itu positif, negatif, maupun netral [2].

2.3. Klasifikasi Sentimen

Klasifikasi sentimen adalah proses untuk menentukan polaritas opini seperti positif, negatif, atau netral dalam sebuah teks. Pada analisis komentar *YouTube*, teknik ini berguna untuk memahami reaksi audiens terhadap konten video. Pendekatan awal untuk klasifikasi sentimen sering kali dilakukan pada tingkat dokumen, di mana seluruh ulasan dianalisis untuk menentukan polaritasnya secara keseluruhan. Pendekatan ini kemudian berkembang ke tingkat kalimat untuk menangkap opini yang lebih spesifik. Pada tingkat fitur atau aspek, analisis ini dapat memisahkan sentimen yang diarahkan pada elemen-elemen tertentu dari sebuah entitas [8].

Sebagai contoh, komentar "Kualitas video sangat baik, tetapi audionya kurang jelas" mencerminkan opini yang berbeda terhadap dua aspek, yaitu kualitas video (positif) dan kualitas audio (negatif).

2.4. *Support Vector Machine* (*SVM*)

Support Vector Machine (*SVM*) adalah algoritma machine learning berbasis *Structural Risk Minimization* (*SRM*) yang bertujuan untuk menemukan *hyperplane* terbaik guna memisahkan kelas data. Algoritma ini bekerja dengan mencari *hyperplane* yang memiliki *margin* maksimal antara dua kelas, sehingga meningkatkan generalisasi model. Jika data tidak dapat dipisahkan secara *linier*, *SVM* menggunakan *kernel function* seperti *Linear*, *RBF*, *Sigmoid*, dan *Polynomial* untuk mentransformasikan data ke ruang berdimensi lebih tinggi. Hasil penelitian menunjukkan bahwa *SVM* dengan *kernel Linear* memberikan akurasi tertinggi, yaitu 87,02% dalam klasifikasi sentimen komentar [5].

2.5. *Logistic Regression*

Logistic Regression adalah algoritma *supervised machine learning* yang digunakan untuk memprediksi probabilitas suatu kelas dalam variabel kategorikal. Model ini bekerja dengan menggunakan *sigmoid function*, yang mengubah hasil prediksi menjadi nilai probabilitas dalam rentang 0 hingga 1. Estimasi parameter dilakukan menggunakan *Maximum Likelihood Estimation* (*MLE*) untuk menemukan bobot terbaik yang meminimalkan kesalahan prediksi. Dalam analisis sentimen, *Logistic Regression* sering digunakan untuk mengklasifikasikan teks ke dalam kategori positif, negatif, atau netral. Penelitian sebelumnya menunjukkan bahwa *Logistic Regression* mencapai akurasi sebesar 85,17% dalam klasifikasi sentimen komentar media sosial [6].

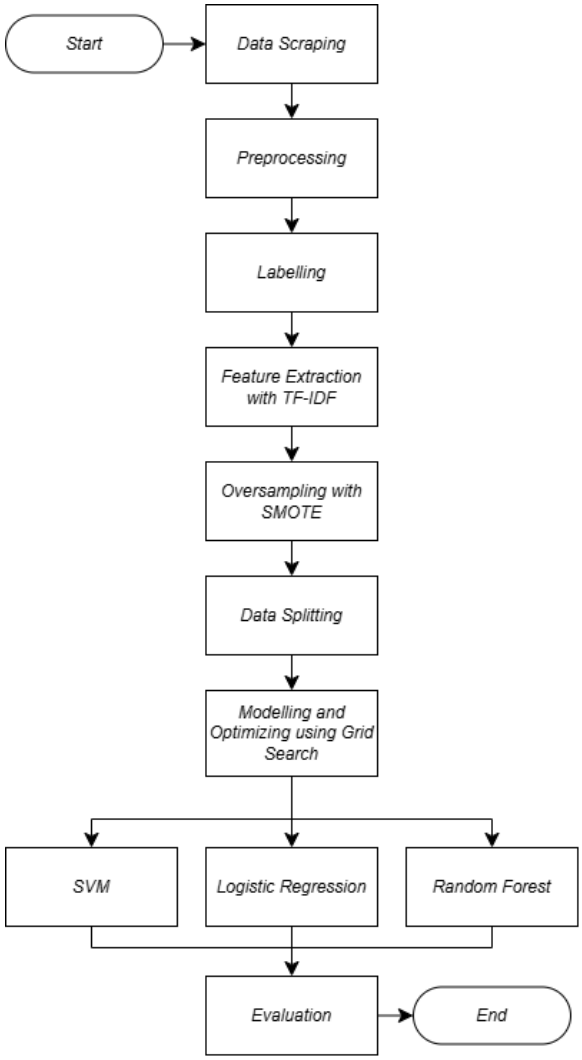
2.6. *Random Forest*

Random Forest adalah algoritma *ensemble learning* yang menggabungkan banyak *decision tree* untuk meningkatkan akurasi prediksi. Setiap pohon dalam *Random Forest* dilatih menggunakan *bootstrap sampling*, dan pemilihan fitur dilakukan secara acak pada setiap percabangan untuk mengurangi *overfitting*. Prediksi akhir ditentukan melalui *majority voting* untuk klasifikasi dan rata-rata untuk regresi. Algoritma ini dikenal memiliki performa yang baik dalam analisis sentimen, termasuk dalam klasifikasi komentar *YouTube*. Penelitian menunjukkan bahwa *Random Forest* menghasilkan akurasi hingga 79% dalam klasifikasi komentar terkait isu sosial [7].

2.7. *Grid Search*

Grid Search adalah metode otomatis yang digunakan untuk menemukan kombinasi terbaik dari *hyperparameter* model. Proses ini melibatkan evaluasi semua kombinasi nilai *hyperparameter* dalam ruang pencarian. Dengan pendekatan ini, setiap konfigurasi diuji secara menyeluruh untuk memastikan hasil terbaik pada data validasi. *Grid Search* bekerja dengan membangun *"grid"* dari semua kemungkinan nilai *hyperparameter* yang telah ditentukan sebelumnya. Kombinasi tersebut diuji melalui evaluasi seperti *cross-validation* untuk mengukur performa model berdasarkan metrik tertentu. Metode ini sering digunakan dalam pengaturan *hyperparameter* untuk berbagai algoritma *machine learning* [9].

3. METODE PENELITIAN



Gambar 1. Alur penelitian

3.1. Data *Scraping*

Pengumpulan data dilakukan dengan teknik *scraping* menggunakan *library youtube_comment_downloader*. Data komentar yang diambil berasal dari beberapa video dengan dua belas topik pembahasan yang berbeda, yaitu politik, pendidikan, kriminal, hiburan, teknologi, olahraga, kesehatan, ekonomi, lingkungan, sosial, *game*, dan religi. Kolom yang digunakan pada proses ini hanya kolom teks yang berisi komentar [10].

3.2. *Preprocessing*

a. *Unique Handling*
Sebelum melanjutkan ke tahap berikutnya, data hasil *scraping* dibersihkan terlebih dahulu dengan menghilangkan nilai *null* dan menghapus data duplikat untuk memastikan kualitas data.

b. *Case Folding*
Seluruh huruf dalam teks komentar diubah menjadi huruf kecil untuk menyamakan format data. Proses ini bertujuan agar teks memiliki konsistensi dalam pengolahan selanjutnya.

c. *Text Cleaning*
Pada tahap ini, berbagai karakter khusus seperti tanda baca, angka, dan simbol lainnya akan dihapus, serta *whitespace* yang tidak perlu akan dibersihkan. *Text cleaning* juga dapat mencakup penghapusan *URL*, mention pengguna, atau karakter *non-alfabet* lainnya yang tidak memberikan kontribusi berarti untuk klasifikasi sentimen. Hasil akhirnya adalah teks yang bersih dengan spasi yang konsisten.

d. *Slang Words Normalization*
Tahap ini mengubah kata tidak baku atau *slang* dalam komentar menjadi kata baku menggunakan kamus *slang words* dari empat sumber *GitHub* [11], [12], [13], [14], yang berisi total 5.433 pasangan kata. Keempat kamus digabung dalam satu file *JSON*, lalu dikonversi menjadi *fix_slangwords.txt*. Jika terdapat kata tidak baku yang sama dengan makna berbeda, dipilih kata baku dengan frekuensi kemunculan tertinggi. File ini digunakan untuk menormalkan teks komentar agar lebih konsisten sebelum diproses lebih lanjut.

e. *Stopwords Removal*
Tahap ini bertujuan untuk menghapus kata-kata yang tidak relevan dalam klasifikasi sentimen, seperti kata hubung, kata depan, dan kata ganti, guna mengurangi *noise* dalam data. Proses ini dilakukan menggunakan *library nltk*, yang menyediakan daftar *stopwords* bawaan untuk berbagai bahasa, termasuk bahasa Indonesia.

f. *Tokenizing*
Proses ini memecah kalimat menjadi kata atau frasa untuk mempermudah tahap klasifikasi berikutnya. Hasil tokenisasi adalah teks yang terstruktur dalam bentuk satuan kata.

g. *Stemming*
Pada tahap ini, kata-kata berimbuhan seperti me-, ber-, di-, ter-, dan lainnya diubah menjadi kata dasar menggunakan *library Sastrawi*. Proses *stemming* dilakukan karena penelitian ini menggunakan konteks berbahasa Indonesia.

3.3. *Labelling*

Pelabelan dilakukan setelah *preprocessing* menggunakan metode *lexicon-based* [15], dengan dua sumber kamus sentimen dari *GitHub*. Sumber pertama berisi 3.609 kata positif dan **6**.609 kata negatif [16], sementara sumber kedua mencakup 1.729 kata dengan skor sentimen campuran [17]. Kamus kedua didistribusikan ke kamus pertama untuk memperkaya data dan menghindari perbedaan skor. Proses ini menggabungkan kamus pertama, lalu menambahkan kata baru dari kamus kedua tanpa duplikasi (842 kata terhapus). Hasil akhirnya adalah kamus *positive.txt* (3.873 kata) dan kamus *negative.txt* (7.230 kata), yang digunakan untuk pelabelan sentimen komentar.

3.4. *TF-IDF*

Pada tahap ini, teks komentar yang telah diproses diubah menjadi representasi numerik menggunakan metode *TF-IDF (Term Frequency-Inverse Document Frequency)* [18].

TF-IDF menghitung bobot setiap kata berdasarkan frekuensi kemunculannya di dalam dokumen dan seluruh dataset, sehingga membantu mengidentifikasi kata-kata yang relevan untuk klasifikasi. Proses ini dilakukan untuk mengubah data teks menjadi bentuk matriks yang dapat digunakan oleh algoritma pembelajaran mesin. *Library Python* seperti *TfidfVectorizer* dari *Scikit-learn* digunakan untuk melakukan proses ini.

3.5. *SMOTE*

Tahap ini bertujuan untuk mengatasi masalah ketidakseimbangan data *(class imbalance)* yang sering terjadi pada dataset klasifikasi sentimen [19].

SMOTE (Synthetic Minority Oversampling Technique) digunakan untuk membuat data sintetis pada kelas minoritas sehingga jumlah data dari masing-masing kelas menjadi seimbang. Hal ini penting untuk memastikan model *machine learning* tidak bias terhadap kelas mayoritas. Proses *oversampling* dilakukan setelah data difiturkan dengan *TF-IDF*. *Library Python* seperti *imbalanced-learn* digunakan untuk implementasi *SMOTE*.

3.6. *Data Splitting*

Dataset yang telah diresample kemudian dibagi menjadi dua bagian: data *training* dan data *testing*. Pembagian ini dilakukan untuk melatih model pada data *training* dan mengevaluasi performanya pada data *testing*[20]. Proses pembagian dilakukan menggunakan metode stratifikasi untuk memastikan distribusi kelas tetap seimbang di kedua bagian data. *Library train_test_split* dari *Scikit-learn* digunakan dalam tahap ini, dengan rasio pembagian umum seperti 80% untuk *training* dan 20% untuk *testing* [21].

3.7. *Modelling and Optimizing using Grid Search*

Penelitian ini menggunakan tiga algoritma klasifikasi sentimen *YouTube*: *Support Vector Machine* (*SVM*), *Logistic Regression* (*LR*), dan *Random Forest* (*RF*). *SVM* dipilih karena kemampuannya memisahkan kelas dengan hyperplane optimal [22], *LR* sebagai model *baseline* yang mudah diinterpretasikan, dan *RF* sebagai ensemble model yang meningkatkan akurasi.

Untuk meningkatkan performa, dilakukan optimasi hyperparameter dengan *Grid Search Cross-Validation* guna mencari kombinasi parameter terbaik, seperti kernel pada *SVM*, regulasi pada *LR*, dan jumlah pohon pada *RF*. Teknik ini membantu menghindari *overfitting* dan meningkatkan stabilitas prediksi.

3.8. *Evaluation*

Evaluasi model dilakukan menggunakan akurasi, *precision*, *recall*, dan *F1-score*. Setelah optimasi *hyperparameter*, model diuji pada data uji untuk menentukan performa terbaik. Berikut persamaan rumus untuk setiap metrik evaluasi.

$$Accuracy = \frac{TP_A + TP_B + TP_C}{N} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

$$F1\text{-}Score = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

Confusion matrix digunakan untuk menganalisis kesalahan klasifikasi dan memahami kelemahan model dalam membedakan tiap kelas sentimen.

4. HASIL DAN PEMBAHASAN

4.1. *Data Scraping*

Tahap pengumpulan data dilakukan dengan teknik *scraping* menggunakan *library youtube_comment_downloader* pada tanggal 3 - 4 November 2024. Data diambil dari komentar *YouTube* pada 12 topik berbeda, yaitu: politik, pendidikan, kriminal, hiburan, teknologi, olahraga, kesehatan, ekonomi, lingkungan, sosial, *game*, dan religi. Untuk setiap topik, data diperoleh dari 4 hingga 5 video, dengan total 1.000 komentar per topik, sehingga diperoleh total data pada dataset adalah 12.000 data komentar, yang kemudian digabungkan menjadi satu dataset utama.

Tabel 1. Sesudah pembersihan data

	text
0	tangkap amin rais said didu asegaf refly harun…
…	…
11999	Orang gila juga waras dllu kali…\nYg percaya…

Dari Tabel 1, dapat dilihat bahwa dataset utama yang terdiri dari 12.000 data komentar hanya mengambil satu kolom atau fitur, yaitu fitur '*text'*, yang berisi komentar itu sendiri. Kolom ini dipilih karena merupakan data utama yang relevan untuk klasifikasi sentimen.

4.2. *Preprocessing*

4.2.1. *Unique Handling*

Sebelum digunakan untuk tahapan selanjutnya, dataset menjalani proses pembersihan awal, meliputi: pengecekan data kosong dan data duplikat. Tidak ditemukan nilai kosong dalam dataset, namun ditemukan sebanyak 46 data duplikat yang kemudian dihapus.

Tabel 2. Sesudah pembersihan data

	text
0	tangkap amin rais said didu asegaf refly harun…
…	…
11953	Orang gila juga waras dllu kali…\nYg percaya…

Dari Tabel 2, dapat dilihat hasil akhir berupa dataset bersih terdiri atas 11.954 baris komentar yang siap digunakan untuk proses *preprocessing* berikutnya.

4.2.2. *Case Folding*

Tabel 3. Hasil *case folding*

Sebelum *case folding*	**Sesudah *case folding***
Bangunan di jakarta banyak yang pake kaca	bangunan di jakarta banyak yang pake kaca
Mantab 👍 jiwa korsa guru	mantab 👍 jiwa korsa guru

Dari Tabel 3, dapat dilihat bahwa tahap *case folding* telah mengubah seluruh huruf dalam teks menjadi huruf kecil untuk memastikan konsistensi, karena bahasa Indonesia tidak membedakan arti berdasarkan kapitalisasi. Proses ini mencegah perbedaan pengolahan antara kata yang sama dalam bentuk kapital atau kecil dan diterapkan pada kolom *'text'* hasil scraping komentar *YouTube*.

4.2.3. *Slang Words Normalization*

Tabel 5. Hasil *slang words normalization*

Sebelum *slang words normalization*	**Sesudah *slang word normalization***
bangunan di jakarta banyak yang pake kaca	bangunan di jakarta banyak yang pakai kaca
mantab jiwa korsa guru	mantab jiwa korsa guru

Dari Tabel 5, dapat dilihat bahwa tahap ini mengganti kata *slang* dalam komentar dengan kata baku menggunakan kamus normalisasi dalam *fix_slangwords.txt*. Normalisasi penting untuk mencegah kesalahan dalam pemrosesan data, mengingat banyaknya penggunaan bahasa gaul di media sosial. Contohnya, kata "pake" diubah menjadi "pakai".

4.2.4. *Stopwords Removal*

Tabel 6. Hasil *stopwords removal*

Sebelum *stopwords removal*	**Sesudah *stopwords removal***
bangunan di jakarta banyak yang pakai kaca	bangunan jakarta pakai kaca
mantab jiwa korsa guru	mantab jiwa korsa guru

Dari Tabel 6, dapat dilihat bahwa tahap ini menghapus *stopwords* seperti kata hubung dan kata ganti yang tidak berkontribusi pada klasifikasi sentimen. Menggunakan pustaka *nltk*, kata-kata seperti "di" dan "yang" dihilangkan untuk meningkatkan akurasi analisis.

4.2.5. *Tokenizing*

Tabel 7. Hasil *tokenizing*

Sebelum *toknizing*	**Sesudah *tokenizing***
bangunan jakarta pakai kaca	['bangunan', 'jakarta', 'pakai', 'kaca']
mantab jiwa korsa guru	['mantab', 'jiwa', 'korsa', 'guru']

Dari Tabel 7, dapat dilihat bahwa tahap *tokenizing* memecah teks hasil *stopwords removal* menjadi token berupa kata atau frasa agar lebih mudah dianalisis.

4.2.6. *Stemming*

Tabel 8. Hasil *stemming*

Sebelum stemming	***Sesudah stemming***
['bangunan', 'jakarta', 'pakai', 'kaca']	*bangun jakarta pakai kaca*
['mantab', 'jiwa', 'korsa', 'guru']	*mantab jiwa korsa guru*

Dari Tabel 8, dapat dilihat bahwa tahap *stemming* mengubah kata berimbuhan menjadi kata dasar menggunakan *library Sastrawi*, sehingga mengurangi variasi kata dan meningkatkan efisiensi model.

4.3. *Labelling*

Pelabelan sentimen dilakukan setelah *preprocessing* menggunakan metode berbasis leksikon dengan dua kamus final: *positive.txt* (3.873 kata) dan *negative.txt* (7.230 kata). Skor kata dalam komentar dijumlahkan untuk menentukan polaritas: skor > 0 (*Positive*), skor < 0 (*Negative*), dan skor = 0 (*Neutral*).

Tabel 10. Hasil *labelling*

preprocessed_text	***sentiment***
syukur hendak lahir presiden wakil presiden republik indonesia bp prabowo subianto gibran allah akbar allah akbar	*Positive*
orang rusak hukum mati depan efek jera abang orang jahil tampa pinye asa gila pondok hanye senang ye	*Negative*
percaya jelas conny menteri uang	*Neutral*

Tabel 10 menunjukkan contoh hasil *labelling* pada dataset. Dataset hasil pelabelan memiliki 3.291 data *Positive*, 6.642 data *Negative*, dan 2.201 data *Neutral*. Setelah pengecekan, tidak ditemukan nilai *null*, tetapi terdapat 407 data duplikat yang dihapus untuk menjaga konsistensi. Meskipun metode ini cukup efektif, akurasinya masih bergantung pada kelengkapan kamus yang digunakan.

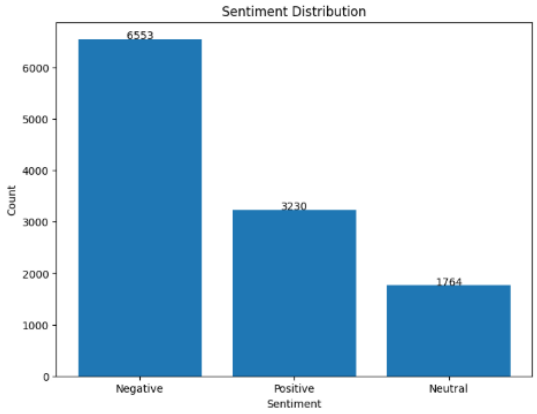
4.4. *TF-IDF*

Setelah *preprocessing*, fitur diekstraksi menggunakan *TF-IDF Vectorizer* dari *Scikit-learn* untuk mengubah teks menjadi vektor numerik yang dapat digunakan oleh algoritma *machine learning*. Proses ini dilakukan dengan *TfidfVectorizer()*, di mana setiap kata diberi bobot berdasarkan frekuensi dan kepentingannya dalam komentar. Ekstraksi fitur ini

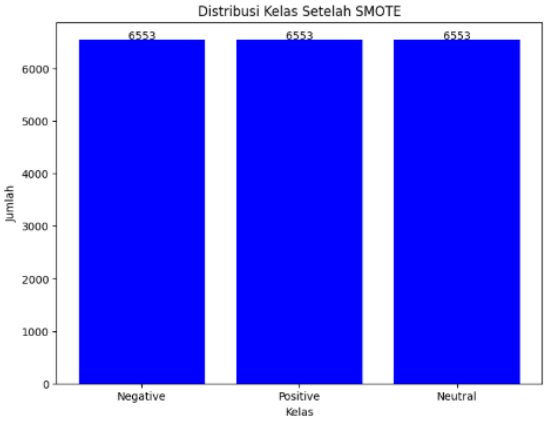
memudahkan identifikasi kata penting dalam klasifikasi sentimen, memungkinkan model memproses teks secara lebih efektif.

4.5. *SMOTE*

Eksperimen ini menerapkan *Synthetic Minority Oversampling Technique* (*SMOTE*) untuk menangani *class imbalance*, yang dapat menyebabkan model lebih cenderung memprediksi kelas mayoritas. *SMOTE* bekerja dengan membuat *synthetic* samples untuk kelas minoritas menggunakan teknik interpolasi antar data poin yang ada, bukan sekadar menduplikasi sampel yang sudah ada. Proses ini melibatkan pemilihan sampel dalam kelas minoritas, identifikasi *k-nearest neighbors*, dan pembuatan data baru di antara titik-titik tersebut untuk menyeimbangkan distribusi dataset.



Gambar 2. Distribusi dataset sebelum *SMOTE*



Gambar 3. Distribusi dataset setelah *SMOTE*

Pada Gambar 2, distribusi dataset sebelum penerapan *SMOTE* menunjukkan ketidakseimbangan signifikan antara kelas *negative*, *positive*, dan *neutral*, dengan kelas *negative* memiliki jumlah sampel yang jauh lebih tinggi. Setelah menerapkan *SMOTE*, seperti yang terlihat pada Gambar 3, distribusi kelas menjadi lebih seimbang. Dengan distribusi yang lebih proporsional ini, model dapat belajar lebih baik dalam mengenali pola dari kelas minoritas, sehingga meningkatkan kemampuan klasifikasinya.

Namun, meskipun *SMOTE* dapat meningkatkan akurasi pada kelas minoritas, teknik ini memiliki beberapa keterbatasan. *Synthetic* samples yang dihasilkan tidak selalu mencerminkan distribusi asli data, yang dapat menyebabkan *overfitting*, terutama jika digunakan secara berlebihan. Untuk mengatasi hal ini, eksperimen ini juga membandingkan performa model dengan dan tanpa *SMOTE* guna mengevaluasi dampaknya secara empiris. Selain itu, *cross-validation* diterapkan untuk memastikan model tetap memiliki *generalizability* yang baik terhadap *unseen* data.

4.6. *Data Splitting*

Setelah *oversampling* dengan *SMOTE*, dataset dibagi menjadi 80% data *training* dan 20% data *testing* untuk memastikan model dapat belajar dan diuji secara optimal. Pembagian dilakukan secara acak dengan tetap menjaga keseimbangan kelas guna menghindari bias terhadap kelas mayoritas.

Dataset hasil pembagian terdiri dari 15.727 sampel untuk *training* dan 3.932 sampel untuk *testing*. Dengan rasio ini, model memiliki cukup data untuk dilatih serta diuji secara representatif, sehingga meningkatkan akurasi dalam klasifikasi sentimen.

4.7. *Modelling and Optimizing using Grid Search*

Pada tahap ini, dilakukan penerapan dan optimasi tiga algoritma klasifikasi, yaitu *Support Vector Machine* (*SVM*), *Logistic Regression* (*LR*), dan *Random Forest* (*RF*). Setiap algoritma diuji dengan parameter default terlebih dahulu, kemudian dilakukan optimasi menggunakan *Grid Search Cross-Validation* untuk meningkatkan performa model.

4.7.1. *Support Vector Machine (SVM)*

Model *SVM* awalnya diuji dengan pengaturan default dan menghasilkan akurasi sebesar 90,97%. Untuk meningkatkan performa, dilakukan optimasi *hyperparameter* menggunakan *Grid Search* dengan parameternya antara lain: *'C'*: [0.1, 1, 10], *'kernel'*: [*'linear'*], *'max_iter'*: [1000, 2000], dan *'tol'*: [1e-4, 1e-5]. Hasil optimasi menunjukkan bahwa kombinasi parameter terbaik adalah *'C'*: 10, *'kernel'*: *'linear'*, *'max_iter'*: 1000, *'tol'*: 0.0001, yang meningkatkan akurasi menjadi 92,55%.

4.7.2. *Logistic Regression (LR)*

Logistic Regression diuji dengan pengaturan default dan memperoleh akurasi sebesar 86,72%. Selanjutnya, dilakukan optimasi menggunakan *Grid Search* dengan parameter: *'penalty'*: ['l1', 'l2'], *'C'*: [0.1, 1, 10, 100], *'solver'*: [*'liblinear'*, *'saga'*], *'max_iter'*: [100, 500]. Kombinasi terbaik yang diperoleh adalah *'C'*: 100, *'max_iter'*: 100, *'penalty'*: 'l1', *'solver'*: *'liblinear'*, yang meningkatkan akurasi menjadi 92,34%.

4.7.3. *Random Forest (RF)*

Random Forest diuji dengan pengaturan default dan memperoleh akurasi sebesar 87,33%. Untuk

meningkatkan performa, dilakukan optimasi *hyperparameter* dengan *Grid Search* menggunakan parameter: *'n_estimators'*: [100, 300], *'max_depth'*: [10, *None*], *'min_samples_split'*: [2, 5], *'min_samples_leaf'*: [1, 2], dan *'bootstrap'*: [*True, False*]. Kombinasi terbaik yang diperoleh adalah *'bootstrap*': *False*, *'max_depth'*: *None*, *'min_samples_leaf'*: 1, *'min_samples_split'*: 5, *'n_estimators'*: 300, yang meningkatkan akurasi menjadi 88,84%.

4.8. *Evaluation*

Evaluasi model dilakukan dengan menggunakan metrik akurasi, *precision*, *recall*, dan *F1-score*. Selain itu, digunakan *confusion matrix* untuk menganalisis pola kesalahan klasifikasi yang terjadi pada setiap model.

Tabel 12. Hasil evaluasi metrik klasifikasi dan *confusion matrix* (sebelum dan setelah optimasi)

Algorima		**Akurasi**	**Kelas**	***Precision***	***Recall***	***F1-Score***	***True Label***			***Predicted Label***		
										Neg	***Neu***	***Pos***
Metrik Evaluasi Sebelum Optimasi	*SVM*	90,97%	*Neg*	96%	84%	89%	***Confusion Matrix* Sebelum Optimasi**	*SVM*	*Neg*	1.104	147	69
			Neu	86%	96%	91%			*Neu*	26	1.257	31
			Pos	92%	94%	93%			*Pos*	23	59	1.216
	LR	86,72%	*Neg*	92%	79%	85%		*LR*	*Neg*	1.046	189	85
			Neu	81%	92%	86%			*Neu*	49	1.215	50
			Pos	89%	89%	89%			*Pos*	47	102	1.149
	RF	87,33%	*Neg*	91%	77%	83%		*RF*	*Neg*	1.013	147	160
			Neu	85%	96%	90%			*Neu*	28	1.265	21
			Pos	86%	89%	88%			*Pos*	72	70	1.156
Metrik Evaluasi Setelah Optimasi	*SVM*	**92,55%**	*Neg*	96%	85%	91%	***Confusion Matrix* Setelah Optimasi**	*SVM*	*Neg*	1.126	128	66
			Neu	88%	98%	93%			*Neu*	19	1.283	12
			Pos	94%	95%	94%			*Pos*	23	45	1.230
	LR	92,34%	*Neg*	95%	86%	90%		*LR*	*Neg*	1.131	128	61
			Neu	88%	97%	92%			*Neu*	25	1.280	9
			Pos	95%	94%	94%			*Pos*	32	46	1.220
	RF	88,84	*Neg*	90%	81%	85%		*RF*	*Neg*	1.064	104	152
			Neu	89%	95%	92%			*Neu*	36	1.254	24
			Pos	87%	91%	89%			*Pos*	78	45	1.175

Dari Tabel 12, dapat dilihat bahwa optimasi menggunakan *Grid Search* memberikan peningkatan pada akurasi, *precision*, *recall*, dan *F1-score* di semua model. Model *SVM* memiliki akurasi tertinggi baik sebelum dan setelah dioptimasi, dibandingkan dengan *Logistic Regression* dan *Random Forest*.

Analisis *confusion matrix* pada Tabel 12 menunjukkan bahwa model setelah optimasi memiliki tingkat kesalahan klasifikasi yang lebih rendah. Kesalahan klasifikasi pada kelas *"Neutral"* berkurang secara signifikan setelah optimasi, yang menandakan peningkatan kemampuan model dalam membedakan kategori yang lebih sulit.

5. KESIMPULAN DAN SARAN

Penelitian ini berhasil menganalisis performa *SVM*, *Logistic Regression*, dan *Random Forest* dalam klasifikasi sentimen komentar *YouTube* berbahasa *Indonesia*, dengan optimasi *Grid Search* yang meningkatkan akurasi masing-masing model. *SVM* mencapai akurasi tertinggi (92,55%), diikuti *Logistic Regression* (92,34%) dan *Random Forest* (88,84%). Teknik *TF-IDF*, *SMOTE*, dan *preprocessing* terbukti meningkatkan kinerja model. Untuk penelitian lanjutan, disarankan untuk menggunakan pelabelan manual, karena hasil *labelling* berbasis kamus masih menghasilkan banyak prediksi yang kurang akurat. Selain itu, dibandingkan *oversampling* dengan *SMOTE*, perlu dipertimbangkan penggunaan *undersampling* untuk mengatasi ketidakseimbangan data, karena data sintetis dari *oversampling* berpotensi menyebabkan penurunan akurasi model.

DAFTAR PUSTAKA

[1] A. A. S and H. Rajeev, "Indian Journal of Data Mining (IJDM) Youtube Comment Sentimental Analysis," 2024, doi: 10.54105/ijdm.A1633.04010524.

[2] R. F. Alhujaili and W. M. S. Yafooz, "Sentiment Analysis for Youtube Videos with User Comments: Review," in *Proceedings - International Conference on Artificial Intelligence and Smart Systems, ICAIS 2021*, Institute of Electrical and Electronics Engineers Inc., Mar. 2021, pp. 814–820. doi: 10.1109/ICAIS50930.2021.9396049.

[3] A. Nadroh, D. N. Triwibowo, and R. B. B. Sumantri, "Klasifikasi Status Gizi Balita

Menggunakan Algoritma Support Vector Machine dengan Optimasi Grid Search Cross-Validation," *METHOMIKA Jurnal Manajemen Informatika dan Komputerisasi Akuntansi*, vol. 8, no. 2, pp. 250–257, Oct. 2024, doi: 10.46880/jmika.Vol8No2.pp250-257.

[4] T. Sugihartono, R. Rian, and C. Putra, "Penerapan Metode Support Vector Machine dalam Classifikasi Ulasan Pengguna Aplikasi Mobile JKN," vol. 7, no. 2, pp. 144–153, 2024.

[5] A. Novantika, "Analisis Sentimen Ulasan Pengguna Aplikasi Video Conference Google Meet menggunakan Metode SVM dan Logistic Regression," *PRISMA, Prosiding Seminar Nasional Matematika*, vol. 5, pp. 808–813, 2022, [Online]. Available: https://journal.unnes.ac.id/sju/index.php/prisma/

[6] N. L. P. C. Savitri, R. A. Rahman, R. Venyutzky, and N. A. Rakhmawati, "Analisis Klasifikasi Sentimen Terhadap Sekolah Daring pada Twitter Menggunakan Supervised Machine Learning," *Jurnal Teknik Informatika dan Sistem Informasi*, vol. 7, no. 1, Apr. 2021, doi: 10.28932/jutisi.v7i1.3216.

[7] I. Afdhal *et al.*, "Penerapan Algoritma Random Forest Untuk Analisis Sentimen Komentar Di YouTube Tentang Islamofobia," *Jurnal Nasional Komputasi dan Teknologi Informasi*, vol. 5, no. 1, 2022.

[8] K. W. Trisna and H. J. Jie, "Deep Learning Approach for Aspect-Based Sentiment Classification: A Comparative Review," 2022, *Taylor and Francis Ltd.* doi: 10.1080/08839514.2021.2014186.

[9] Louis. Owen, *Hyperparameter tuning with Python : boost your machine learning model's performance via hyperparameter tuning*. Packt Publishing Ltd., 2022.

[10] A. R. D. Astuti and N. Cahyono, "Analisis Topic Modelling Persepsi Pengguna Internet Menggunakan Metode Latent Dirichlet Allocation," Indones. J. Comput. Sci., vol. 12, no. 1, pp. 326–334, 2023, doi: 10.33022/ijcs.v12i1.3155.

[11] D. F. Zhafira, "Kampus Merdeka Cross Validation - Kamus Perbaikan Kata." Accessed: Feb. 11, 2025. [Online]. Available: https://github.com/farahdhaifa/kampus-merdeka-crossvalidation/blob/main/kamus%20perbaikan%20kata.xlsx

[12] R. Prakoso, "Analisis Sentimen - Kamus KBBA." Accessed: Feb. 11, 2025. [Online]. Available: https://github.com/ramaprakoso/analisis-sentimen/blob/master/kamus/kbba.txt

[13] A. Makmun, "SentiStrengthID - Kamus Slangword." Accessed: Feb. 11, 2025. [Online]. Available: https://github.com/agusmakmun/SentiStrengthID/blob/master/id_dict/slangword.txt

[14] W. F. Abdillah, "Sentimen Bahasa - Colloquial Indonesian Lexicon." Accessed: Feb. 11, 2025. [Online]. Available: https://github.com/onpilot/sentimen-bahasa/blob/master/kamus/nasalsabila_kamus-alay/_json_colloquial-indonesian-lexicon.txt

[15] G. Andika Joni, N. Cahyono, A. Baita, and N. Aini, "ANALISIS SENTIMEN KOMENTAR MASYARAKAT TERHADAP RANGKA ESAF HONDA MENGGUNAKAN ALGORITMA NAIVE BAYES DAN SUPER VECTOR MACHINE," 2024.

[16] W. F. Abdillah, "Sentimen Bahasa - INSET Lexicon." Accessed: Feb. 11, 2025. [Online]. Available: https://github.com/onpilot/sentimen-bahasa/tree/master/leksikon/inset

[17] W. F. Abdillah, "Sentimen Bahasa - SentiStrength ID Lexicon." Accessed: Feb. 11, 2025. [Online]. Available: https://github.com/onpilot/sentimen-bahasa/blob/master/leksikon/sentistrength_id/_json_sentiwords_id.txt

[18] A. Oktavia Praneswara and N. Cahyono, "Analisis Sentimen Ulasan Aplikasi TikTok Shop Seller Center di Google Playstore Menggunakan Algoritma Naive Bayes," *Indonesian Journal of Computer Science Attribution*, vol. 12, no. 6, p. 3925, 2023.

[19] B. Wicaksono and N. Cahyono, "ANALISIS SENTIMEN KOMENTAR INSTAGRAM PADA PROGRAM KAMPUS MERDEKA DENGAN ALGORITMA NAIVE BAYES DAN DECISION TREE," 2024.

[20] R. G. A. Pratama and N. Cahyono, "Comparison of Deep Learning Methods on Sentiment Analysis Using Word Embedding," JITK, vol. 10, no. 1, pp. 1–8, Jul. 2024. DOI: https://doi.org/10.33480/jitk.v10i1.5280

[21] A. S. Pamungkas and N. Cahyono, "Analisis Sentimen Review ChatGPT di Play Store menggunakan Support Vector Machine dan K-Nearest Neighbor," Edumatic J. Pendidik. Inform., vol. 8, no. 1, pp. 1–10, 2024, doi: 10.29408/edumatic.v8i1.24114.

[22] R. Rakarahayu Putri and N. Cahyono, "Analisis Sentimen Komentar Masyarakat Terhadap Pelayanan Publik Pemerintah Dki Jakarta Dengan Algoritma Super Vector Machine Dan Naive Bayes," JATI (Jurnal Mhs. Tek. Inform., vol. 8, no. 2, pp. 2363–2371, 2024, doi: 10.36040/jati.v8i2.9472.