BAB 2
LANDASAN TEORI

Bab ini menjelaskan landasan teori yang berperan sebagai dasar untuk menganalisis masalah dan merancang solusi dalam penelitian ini. Landasan teori ini mencakup ujaran kebencian, platform X, deep learning, *Convolutional Neural Network* (CNN), dan Word2Vec. Selain itu juga ada penjelasan mengenai metrik evaluasi dan dataset yang digunakan pada penelitian ini.

2.1 Ujaran Kebencian

Ujaran kebencian atau *Hate speech* adalah sebuah fenomena sosial yang rumit, di mana komunikasi digunakan untuk merendahkan, mengejek, atau melukai individu atau kelompok [1]. Hal ini terjadi karena faktor identitas, seperti agama, etnis, atau jenis kelamin. Ujaran kebencian dapat ditemukan di mana-mana, baik di internet maupun di kehidupan nyata. Ujaran kebencian bisa muncul dalam berbagai bentuk seperti sindiran, sarkasme, informasi yang menyesatkan, kata-kata kasar, distorsi, ejekan, dan kritik yang tidak membangun, yang sering kali menargetkan identitas politik atau agama, terutama di saat-saat konflik politik [3]. Sepanjang sejarah, ujaran kebencian telah ada, termasuk dalam konteks keagamaan, yang dapat muncul seperti fitnah, gosip, dan kecemburuan, memengaruhi individu serta komunitas [28].

Ujaran kebencian dimengerti sebagai penurunan status kelompok yang tertekan secara sistematis, yang diidentifikasi berdasarkan jenis kelamin, preferensi seksual, atau kepercayaan agama. Pernyataan yang penuh kebencian dapat menghancurkan rasa aman serta kepercayaan masyarakat, terutama di komunitas yang memiliki beragam etnis. Ujaran semacam itu bisa menimbulkan perpecahan bangsa dengan memperdalam konflik etnis dan sosial, yang mengancam kesatuan serta stabilitas [29]. Di dunia politik, pernyataan kebencian dimanfaatkan untuk menggerakkan para pendukung serta menyerang lawan, sering kali dengan memanfaatkan identitas agama dan etnis [28]. Hal ini memperkuat politik identitas dan dapat merusak nilai-nilai demokrasi.

2.2 Platform X

Platform X atau yang biasa dikenal sebagai Twitter merupakan salah satu platform yang digunakan untuk berbagi informasi dan berekspresi secara *online*. Platform X merupakan sebuah platform media sosial yang populer, di mana pengguna dapat mengirim pesan singkat yang disebut *tweet* kepada audiens di seluruh dunia. Orang-orang dapat mengirim *tweet* yang panjangnya sampai 280 karakter. *Tweet* yang dikirim bisa berupa tulisan, foto, video, atau tautan. Secara standar, *tweet* dapat dilihat oleh semua orang dan tidak hanya oleh mereka yang mengikuti akun tersebut [30]. Platform X terkenal dengan komunikasi yang terjadi secara langsung, keterbukaan publik, dan kecepatannya dalam menyebarkan informasi [6]. Hal ini menjadikannya sebagai alat penting untuk berita, menjalin hubungan, dan diskusi di masyarakat [7].

Platform X juga biasa digunakan untuk menyebarluaskan berita, mendiskusikan peristiwa terbaru, serta berbagi pendapat. Baik individu, organisasi, politisi, maupun tokoh masyarakat memanfaatkan platform X untuk berinteraksi secara langsung dengan publik [31]. Platform X dapat digunakan sebagai platform untuk memahami ujaran kebencian karena jumlah datanya yang besar, sifatnya yang langsung, dan terbuka untuk umum. Peneliti memanfaatkan data dari platform X untuk merancang, menguji, dan mengukur model pembelajaran mesin dalam mendeteksi ujaran kebencian. Mereka juga menganalisis cara penyebaran dan perubahan konten yang berkaitan dengan kebencian, serta menilai langkah-langkah seperti kontra-ujaran. Sejumlah studi telah meneliti dan mempublikasikan basis data besar yang telah diberi anotasi dari platform X. Data yang telah dipublikasikan termasuk COVID-HATE, yang terdiri dari 206 juta tweet dan menyoroti kebencian terhadap orang Asia selama pandemi [32]. Selain itu, ada juga basis data untuk mendeteksi ujaran kebencian dalam berbagai bahasa, seperti Inggris, Hindi, dan Marathi [33].

2.3 Deep Learning

Deep learning merupakan bagian dari kecerdasan buatan serta *machine learning* yang memanfaatkan *artificial neural networks* dengan banyak lapisan yang terhubung untuk secara otomatis memproses dan memahami pola dari banyak data [34]. *Deep learning* sangat efektif dalam mengelola data yang kompleks dan berdimensi tinggi, dan telah memberikan kemajuan signifikan di bidang seperti

penglihatan komputer, pengenalan suara, dan pengolahan bahasa alami.

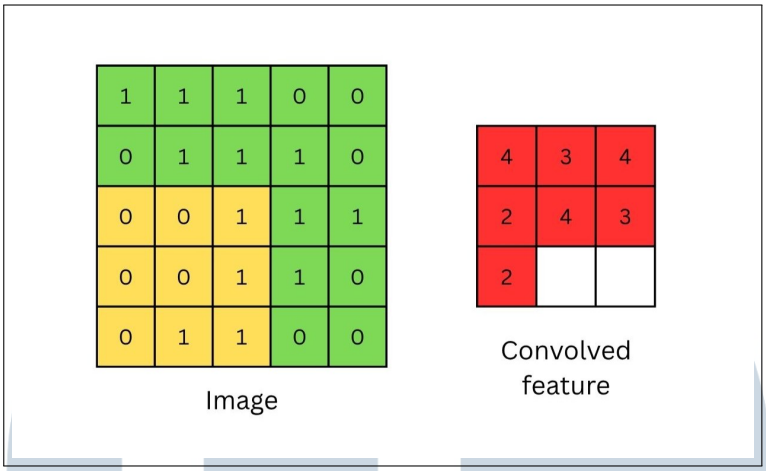
Deep learning dapat diperbesar secara efektif seiring dengan pertumbuhan ukuran dan kerumitan data, yang menjadikannya ideal untuk situasi big data. Model ini mampu disesuaikan untuk berbagai bidang, seperti kesehatan, keamanan, industri, dan sektor pemerintahan. Model yang menjadi arsitektur utama untuk tugas seperti *image recognition*, klasifikasi, dan deteksi adalah *Convolutional Neural Networks* (CNN) [26].

2.4 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) merupakan salah satu model *deep learning* yang dirancang untuk menangani data yang memiliki struktur spasial, seperti urutan kata atau gambar [26]. Secara umum, CNN terdiri dari beberapa lapisan (layer), yaitu *convolutional layer*, *activation layer*, *pooling layer*, dan *fully connected layer*. Proses kerja CNN diawali dengan operasi konvolusi. Pada tahap ini, sebuah filter (juga disebut kernel) akan meluncur di atas input dan melakukan perkalian elemen demi elemen antara nilai-nilai dalam filter dan bagian *input* yang diliputi oleh filter tersebut. Hasil perkalian kemudian dijumlahkan untuk menghasilkan satu nilai *output* yang menjadi bagian dari *feature map* [35].

$$S(i,j) = (I * K)(i,j) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(i+m, j+n) \cdot K(m,n) \tag{2.1}$$

di mana S(i,j) merupakan hasil konvolusi pada posisi (i,j), sedangkan M dan N menyatakan ukuran kernel dalam arah vertikal dan horizontal. Visualisasi proses operasi konvolusi dapat dilihat pada Gambar 2.1



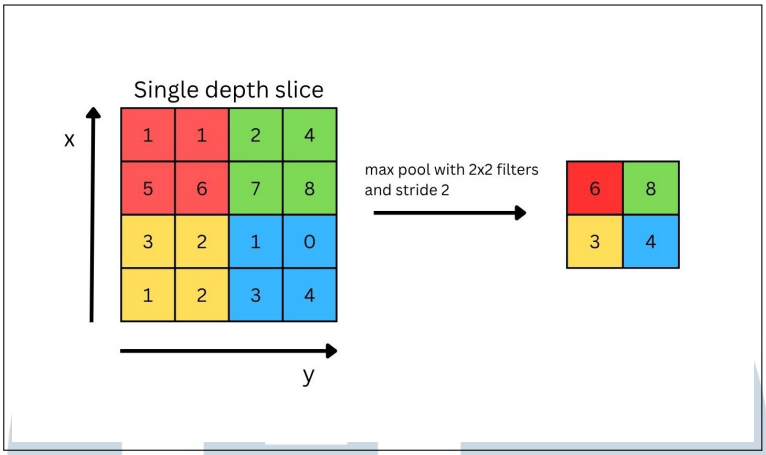
Gambar 2.1. *Convolutional operation*

Sumber: [35]

Gambar 2.1 menunjukkan operasi konvolusi dua dimensi pada gambar atau matriks input 5x5 yang diproses menggunakan filter 3x3. Filter ini bergerak melintasi *input* dan *multiply* elemen satu per satu di setiap posisi. Nilai hasil perkalian dijumlahkan agar dapat menampilkan satu nilai pada *feature map* yang ditampilkan di sebelah kanan. Setelah konvolusi selesai, hasil biasanya dikirim ke fungsi aktivasi *non-linear* seperti *Rectified Linear Unit* (ReLU). Fungsi ini dirancang untuk menghilangkan nilai negatif dan mempercepat proses pelatihan jaringan. Fungsi ReLU disusun sebagai berikut:

$$f(x) = \max(0, x) \tag{2.2}$$

Setelah proses konvolusi, *pooling* (umumnya *max pooling*) dilakukan untuk menyederhanakan representasi fitur dan mengurangi jumlah parameter. *Max pooling* bekerja dengan mengambil nilai maksimum dalam jendela tertentu, misalnya jendela 2x2. Ini membantu dalam meringkas informasi penting dari fitur yang diekstrak. Visualisasi proses *max pooling* dapat dilihat pada Gambar 2.2



Gambar 2.2. Max pooling operation

Sumber: [35]

Gambar 2.2 menunjukkan proses untuk membuat representasi baru yang lebih kecil (2x2), nilai tertinggi dari setiap blok 2x2 pada gambar kiri diambil. Hasil ini tetap mempertahankan informasi paling signifikan dari masing-masing area. CNN akan melanjutkan ke *fully connected layer* setelah fitur pentingnya diekstraksi melalui berbagai lapisan konvolusi dan *pooling*. Lapisan ini akan menggabungkan semua atribut menjadi satu vektor yang akan digunakan untuk klasifikasi akhir. Fungsi aktivasi pada lapisan *output* untuk klasifikasi biner biasanya *sigmoid*.

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{2.3}$$

Fungsi *sigmoid* mengonversi skor prediksi menjadi nilai antara 0 dan 1. Nilai ini kemudian dapat dipahami sebagai probabilitas keanggotaan terhadap kelas tertentu. Ukuran *input*, ukuran filter, dan langkah pergeseran (*stride*) menentukan ukuran *output* lapisan konvolusi. Ketika *padding* tidak diterapkan, rumus berikut digunakan untuk menghitung dimensi hasil konvolusi:

$$\text{Output Size} = \left(\frac{n-f}{s}+1\right) \tag{2.4}$$

Pada tahap pelatihan, CNN meminimalkan *loss function* untuk mengoptimalkan bobot. *Loss function* yang paling umum digunakan untuk klasifikasi biner adalah *binary cross-entropy*, yang memiliki rumus sebagai berikut:

$$\mathscr{L} = -\left[y \cdot \log(\hat{y}) + (1-y) \cdot \log(1-\hat{y})\right] \tag{2.5}$$

2.5 Word2Vec

Word2vec adalah model berbasis jaringan saraf yang mengubah kata menjadi vektor numerik dan menangkap maknanya serta hubungannya dengan cara yang komputer dapat gunakan untuk berbagai tugas bahasa [36]. Fungsi utama model adalah menunjukkan kata sebagai vektor sehingga kata yang serupa memiliki representasi vektor yang sama, yang memungkinkan komputer memahami dan memproses bahasa dengan lebih baik. Word2Vec juga merupakan alat representasi kata berbasis vektor yang digunakan untuk banyak tugas pemrosesan bahasa alami.

Model ini memiliki kemampuan untuk mengukur kedekatan makna kata-kata dalam analisis semantik dan menemukan hubungan seperti sinonimi dan analogi [37]. Word2Vec memberikan fitur numerik yang meningkatkan kinerja algoritma pembelajaran mesin dalam hal klasifikasi teks dan analisis sentimen. Word2Vec bekerja berdasarkan prinsip distributed representation, yaitu konsep bahwa makna suatu kata dapat dipelajari dari konteks lokalnya. Terdapat dua pendekatan utama dalam algoritma Word2Vec, yaitu *Continuous Bag of Words* (CBOW) dan *Skip-Gram*.

Pada pendekatan CBOW, model mempelajari cara memprediksi kata target berdasarkan kata-kata yang muncul di sekitarnya. Sebaliknya, pendekatan *Skip-Gram* digunakan untuk memprediksi kata-kata konteks dari satu kata target. Keduanya dilatih menggunakan jaringan saraf sederhana dengan satu lapisan tersembunyi, di mana bobot jaringan yang dipelajari akan menjadi vektor representasi kata [36]. Proses pelatihan ini dilakukan dengan memanfaatkan teknik optimasi seperti *negative sampling* atau *hierarchical softmax* untuk efisiensi komputasi.

Word2Vec memiliki sejumlah keunggulan dibandingkan teknik representasi teks sebelumnya, seperti *Bag of Words* (BoW) atau TF-IDF, antara lain:

- Menangkap makna semantik dan sintaksis: Word2Vec tidak hanya mencatat frekuensi kemunculan kata, tetapi juga memperhatikan posisi dan konteks kata dalam kalimat, sehingga mampu memahami makna kata berdasarkan hubungan antar kata.

- Menghasilkan representasi vektor yang kompak dan efisien: Representasi vektor Word2Vec bersifat dens, dengan dimensi rendah (misalnya 100–300 dimensi), namun tetap mampu merepresentasikan makna dengan akurat.
- Kemampuan generalisasi tinggi: Word2Vec mampu menangkap analogi dan hubungan semantik yang kompleks antara kata-kata. Sebagai contoh, hubungan antara Raja dan Ratu mirip dengan hubungan antara Pria dan Wanita dalam ruang vektor.
- Mengurangi masalah sparseness (kelangkaan data): Tidak seperti BoW atau TF-IDF yang menghasilkan representasi yang jarang (*sparse*), Word2Vec menghasilkan representasi yang padat sehingga lebih stabil saat digunakan dalam model pembelajaran lanjutan.

2.6 Evaluation Metrics

Evaluating classification model performance is a fundamental component of machine learning systems [38]. The goal is to measure how well a model can predict outcomes on test data it has never seen during training. In practice, evaluation metrics serve as benchmarks to assess both overall accuracy and the model's ability to identify specific classes correctly [38]. The four most commonly used classification metrics are accuracy, precision, recall, and F1-score, each offering a distinct and complementary perspective on prediction quality.

2.6.1 Accuracy

Accuracy atau akurasi merupakan metrik evaluasi paling dasar yang digunakan untuk mengukur proporsi jumlah prediksi yang benar dibandingkan dengan seluruh jumlah data [38]. Metrik ini memberikan gambaran umum mengenai seberapa sering model menghasilkan prediksi yang sesuai dengan label sebenarnya. Berikut adalah rumus yang digunakan untuk menghitung *Accuracy*:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.6}$$

TP (*True Positive*) menunjukkan jumlah data positif yang diprediksi dengan benar, TN (*True Negative*) adalah jumlah data negatif yang diprediksi benar, FP (*False Positive*) merupakan jumlah data negatif yang diprediksi sebagai positif,

dan FN (*False Negative*) adalah jumlah data positif yang diprediksi sebagai negatif. Misalkan sebuah model klasifikasi biner menghasilkan *confusion matrix* dengan komposisi sebagai berikut: 40 data diklasifikasikan sebagai positif secara benar (*True Positive*), 50 data diklasifikasikan sebagai negatif secara benar (*True Negative*), 10 data negatif diprediksi sebagai positif (*False Positive*), dan 5 data positif tidak terdeteksi oleh model dan diklasifikasikan sebagai negatif (*False Negative*). Berdasarkan data tersebut, perhitungan dimulai dari metrik *accuracy*. *Accuracy* mencerminkan proporsi total prediksi yang benar dibandingkan dengan seluruh data yang diuji. Dengan memasukkan nilai ke dalam rumus, diperoleh hasil sebagai berikut:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} = \frac{40+50}{40+50+10+5} = \frac{90}{105} \approx 0{,}857 \text{ atau } 85{,}7\%$$

Nilai ini menunjukkan bahwa sebanyak 85,7% prediksi yang dihasilkan oleh model berada dalam kategori benar, baik itu prediksi positif maupun negatif. Meskipun metrik ini sangat mudah dihitung dan diinterpretasikan, *accuracy* memiliki keterbatasan dalam kasus-kasus di mana data tidak seimbang, yaitu ketika salah satu kelas memiliki jumlah data yang jauh lebih banyak dibandingkan kelas lainnya. Dalam situasi seperti ini, nilai akurasi yang tinggi belum tentu menunjukkan bahwa model bekerja dengan baik secara menyeluruh.

2.6.2 Precision

Precision merupakan metrik yang digunakan untuk mengukur seberapa banyak dari seluruh prediksi positif yang benar-benar merupakan data positif. Metrik ini sangat relevan dalam situasi di mana kesalahan dalam mengklasifikasikan data negatif sebagai positif perlu dihindari [38]. Berikut adalah rumus yang digunakan untuk menghitung *Precision*:

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2.7}$$

Dengan menggunakan data yang sama seperti sebelumnya, diketahui bahwa terdapat 40 data positif yang diprediksi dengan benar (*true positive*), dan 10 data negatif yang salah diklasifikasikan sebagai positif (*false positive*). Precision dihitung dengan membagi jumlah *true positive* dengan total prediksi positif, yaitu

true positive ditambah *false positive*, hasil perhitungannya adalah:

$$\text{Precision} = \frac{TP}{TP+FP} = \frac{40}{40+10} = \frac{40}{50} = 0{,}8 \text{ atau } 80\%$$

Dengan demikian, 80% dari seluruh data yang diprediksi positif oleh model benar-benar merupakan data positif, menunjukkan tingkat kepercayaan terhadap prediksi positif yang cukup baik. Metrik ini menghitung persentase prediksi positif yang benar-benar sesuai dengan label sebenarnya.

2.6.3 Recall

Recall juga dikenal sebagai sensitivitas, mengukur kemampuan model dalam mendeteksi seluruh data yang benar-benar positif. Metrik ini penting dalam situasi di mana kesalahan dalam melewatkan data positif (*false negative*) dianggap kritis [38]. Berikut adalah rumus yang digunakan untuk menghitung *Recall*:

$$\text{Recall} = \frac{TP}{TP+FN} \tag{2.8}$$

Berdasarkan contoh yang sama, terdapat 40 data positif yang diklasifikasikan dengan benar (*true positive*), dan 5 data positif yang tidak terdeteksi oleh model (*false negative*), hasil perhitungannya adalah:

$$\text{Recall} = \frac{TP}{TP+FN} = \frac{40}{40+5} = \frac{40}{45} \approx 0{,}889 \text{ atau } 88{,}9\%$$

Maka *recall* adalah 40 dibagi 45, yang menghasilkan nilai sekitar 88,9 persen. Nilai tersebut menunjukkan bahwa model mampu mengidentifikasi hampir seluruh data positif yang ada, dengan tingkat deteksi mencapai 88,9%. Nilai *recall* yang tinggi menunjukkan bahwa sebagian besar data positif berhasil dikenali oleh model, meskipun hal ini terkadang dilakukan dengan mengorbankan nilai *precision*.

2.6.4 F1-Score

F1-score merupakan rata-rata harmonis antara *precision* dan *recall*. Metrik ini digunakan untuk mendapatkan keseimbangan antara keduanya, terutama dalam kondisi ketika data tidak seimbang dan diperlukan kompromi antara meminimalkan kesalahan positif palsu dan negatif palsu [38]. Berikut adalah rumus yang digunakan untuk menghitung *F1-score*:

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.9}$$

Dari perhitungan sebelumnya, precision sebesar 80 persen dan *recall* sebesar 88,9 persen. Nilai *F1-score* diperoleh dengan mengalikan dua kali *precision* dan *recall*, lalu dibagi dengan jumlah keduanya, hasil perhitungannya adalah:

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = 2 \times \frac{0{,}8 \times 0{,}889}{0{,}8 + 0{,}889} = 2 \times \frac{0{,}7112}{1{,}689} \approx 0{,}841 \text{ atau } 84{,}1\%$$

F1-score sebesar 84,1% menunjukkan bahwa model tidak hanya tepat dalam memberikan prediksi positif, tetapi juga cukup lengkap dalam menjangkau seluruh data positif yang relevan. Nilai *F1-score* yang tinggi menunjukkan bahwa model tidak hanya mampu mendeteksi sebagian besar data positif dengan baik, tetapi juga cukup selektif dalam menentukan mana yang benar-benar termasuk dalam kelas positif. Sebagai rata-rata dari *precision* dan *recall*, *F1-score* memberikan gambaran performa yang lebih stabil dan menyeluruh. *F1-score* sering digunakan dalam penelitian maupun implementasi praktis sebagai metrik utama untuk membandingkan performa berbagai model klasifikasi.

2.7 Dataset

Dataset yang digunakan dalam penelitian ini didapat dari Github. *Dataset* ini dibangun oleh Muhammad Okky Ibrohim dan Indra Budi karena permasalahan maraknya penyebaran ujaran kebencian dan bahasa kasar di media sosial, khususnya Twitter berbahasa Indonesia [39]. Ujaran kebencian memiliki potensi besar untuk memicu konflik sosial, diskriminasi, bahkan kekerasan fisik. *Dataset* ini dibangun dengan anotasi *multi-label* yang mencakup informasi terkait ujaran kebencian dan bahasa kasar di Twitter. Proses pembangunan *dataset* dilakukan dalam dua tahap anotasi. Tahap pertama dilakukan untuk menentukan apakah suatu tweet termasuk dalam ujaran kebencian, bahasa kasar, atau tidak keduanya. Tahap kedua difokuskan untuk mengidentifikasi target, kategori, dan tingkat kebencian pada tweet yang diklasifikasikan sebagai ujaran kebencian. Proses anotasi pada *dataset* ini melibatkan 30 anotator yang dipilih secara selektif berdasarkan kriteria demografis dan kompetensi linguistik untuk menjamin objektivitas dan kualitas data. Anotator berasal dari berbagai latar belakang usia, pendidikan, etnis, agama,

dan profesi yang mencerminkan keragaman masyarakat Indonesia.

Setiap anotator merupakan penutur asli bahasa Indonesia dan aktif menggunakan Twitter. Variasi latar belakang ini bertujuan untuk meminimalkan bias dalam penilaian teks yang sensitif secara sosial dan budaya. Anotasi dilakukan dalam dua tahap, di mana tahap pertama mengklasifikasikan tweet sebagai ujaran kebencian, bahasa kasar, atau bukan keduanya, sedangkan tahap kedua fokus pada identifikasi target, kategori, dan tingkat ujaran kebencian. Data yang dimasukkan ke dalam *dataset* akhir hanya yang memenuhi tingkat kesepakatan tertentu, menggunakan metode kesepakatan penuh pada tahap pertama dan *majority voting* pada tahap kedua.

Dataset yang dibangun berasal dari gabungan beberapa penelitian sebelumnya serta hasil *crawling* data Twitter selama tujuh bulan menggunakan kata kunci yang telah disusun berdasarkan studi linguistik dan konsultasi dengan ahli sosiolinguistik. Pendekatan klasifikasi pada penelitian *dataset* ini menggunakan metode *multi-label text classification* dengan algoritma *Naive Bayes*, *Support Vector Machine*, dan *Random Forest Decision Tree* yang dikombinasikan dengan transformasi data berupa *Binary Relevance*, *Label Power-set*, dan *Classifier Chains*. Fitur yang digunakan meliputi *word n-gram*, *character n-gram*, fitur ortografis, serta leksikon sentimen dan kata kasar. Hasil eksperimen menunjukkan bahwa kombinasi *word unigram*, algoritma *Random Forest*, dan metode *Label Power-set* memberikan performa terbaik, dengan akurasi 77,36% pada skenario deteksi ujaran kebencian dan bahasa kasar, namun menurun menjadi 66,12% saat mencakup target, kategori, dan tingkat kebencian.

Penurunan ini disebabkan oleh kompleksitas label dan ketidakseimbangan data, yang juga memicu kesalahan dominan berupa *false negative*. Untuk mengatasi hal tersebut, disarankan penggunaan pendekatan klasifikasi *multi-label* hierarkis dan penambahan fitur semantik seperti *word embedding* untuk meningkatkan akurasi dan pemahaman konteks ujaran. Sebagai penutup, *dataset* yang dihasilkan beserta panduan anotasinya dipublikasikan secara terbuka agar dapat digunakan oleh peneliti lain dalam studi lanjutan mengenai deteksi ujaran kebencian dan bahasa kasar di media sosial berbahasa Indonesia. Sebagai penutup, *dataset* yang dihasilkan beserta panduan anotasinya dipublikasikan secara terbuka agar dapat digunakan oleh peneliti lain dalam studi lanjutan mengenai deteksi ujaran kebencian dan bahasa kasar di media sosial berbahasa Indonesia.

Validitas *dataset* yang digunakan dalam penelitian ini didukung oleh hasil eksperimen dari berbagai studi terdahulu yang memanfaatkan *dataset* yang sama

dalam konteks deteksi ujaran kebencian dan bahasa kasar pada media sosial. Salah satu studi melaporkan bahwa penggunaan model *Support Vector Machine* (SVM) dengan parameter yang telah dioptimalkan menghasilkan akurasi rata-rata sebesar 82,65%, dengan rincian akurasi per kelas yaitu 84,92% untuk kelas ujaran kebencian, 86,60% untuk kelas bahasa kasar, dan 76,43% untuk tingkat kebencian [40]. Angka tersebut menunjukkan bahwa *dataset* mampu mendukung proses klasifikasi secara efektif, meskipun performa terbaik masih diperoleh melalui pendekatan berbasis *deep learning*.

Selain itu, eksperimen menggunakan algoritma *Logistic Regression* yang memanfaatkan fitur *word embedding* juga menunjukkan akurasi yang kompetitif. Dari beberapa kali pengujian untuk memperoleh model terbaik, diperoleh akurasi rata-rata sebesar 75,59% dengan komposisi data pelatihan dan pengujian 90:10. Akurasi per kelas menunjukkan nilai yang cukup stabil, yakni 75,86% untuk ujaran kebencian, 80,05% untuk bahasa kasar, dan 70,86% untuk tingkat kebencian [41]. Penelitian lainnya menunjukkan bahwa penggunaan algoritma *Recurrent Neural Network* (RNN) pada dataset ini memberikan hasil yang paling optimal dengan akurasi mencapai 91%, mengungguli model-model konvensional berbasis *machine learning* [42]. Tingginya tingkat akurasi ini menegaskan bahwa data yang digunakan memiliki kualitas anotasi yang baik, distribusi label yang representatif, serta struktur data yang mendukung penerapan berbagai pendekatan algoritma secara fleksibel.