Journal of Applied Informatics and Computing (JAIC)
Vol.9, No.6, December 2025, pp. 3177~3190
e-ISSN: 2548-6861

Sentiment Analysis of Coretax on Social Media X Using *Naive Bayes*, SVM, and LSTM for Service Improvement

Steven Dermawan[1]*, Afifah Trista Ayunda[2]*
* Sistem Informasi, Fakultas Sains dan Teknologi, Universitas Pradita
steven.dermawan@student.pradita.ac.id[1], afifah.trista@pradita.ac.id[2]

Article Info

Article history:

Received 2025-09-05
Revised 2025-10-01
Accepted 2025-11-05

Keyword:

Sentiment Analysis,
Coretax,
Naive Bayes,
Support Vector Machine,
LSTM.

ABSTRACT

In January 2025, Indonesia's Ministry of Finance launched Coretax to replace DJP Online. However, the launch triggered widespread dissatisfaction among users, reflecting negative public sentiment. This study aims to analyze public perception of Coretax and evaluate the performance of machine learning models in sentiment classification. A total of 6.036 Indonesian language tweets related to Coretax, posted between January and April 2025, were collected using Tweet Harvest. The dataset consists of 0,83% positive, 51,05% negative, and 48,11% neutral sentiments. The research methodology involved several stages: data crawling, manual labeling, preprocessing (cleaning, case folding, stopword removal, tokenization, normalization, stemming, and specifically for LSTM: conversion of tokens into numerical indices, padding, and embedding), feature representation using TF-IDF for classical models and word embedding for deep learning, data balancing with SMOTE, model implementation (Naive Bayes, Support Vector Machine with various kernels, and LSTM), model evaluation and comparison, and visualization through word clouds. The application of SMOTE succeeded in improving the performance of all algorithms. After applying SMOTE, the SVM with the RBF kernel achieved the best performance with 90,70% accuracy, 91% precision, 90,66% recall, and 90,66% F1-score. Keyword analysis revealed that terms such as "data" and "mudah" dominated positive sentiment, "silakan" and "kakak" were prevalent in neutral sentiment, while "sistem" and "error" frequently appeared in negative sentiment. The findings highlight the urgent need for system infrastructure improvements, user-centered features, responsive technical support, taxpayer training, and continuous updates to enhance Coretax and restore public trust.

I. PENDAHULUAN

Pajak merupakan kewajiban finansial yang harus dipenuhi oleh individu maupun badan usaha kepada negara sesuai dengan ketentuan hukum tanpa imbalan langsung, yang dimanfaatkan untuk mendanai kepentingan publik serta pembangunan nasional [1]. Sebagai sumber utama pendapatan negara, pajak memiliki peran penting dalam menjaga stabilitas ekonomi dan mendorong pembangunan berkelanjutan [2]. Oleh karena itu, sistem administrasi perpajakan yang efisien diperlukan guna meningkatkan kepatuhan wajib pajak serta mengoptimalkan penerimaan negara. Dengan kemajuan teknologi, pemerintah Indonesia melalui Direktorat Jenderal Pajak (DJP) mengembangkan Coretax sebagai solusi digital dalam pengelolaan pajak [3].

Implementasi Coretax bertujuan untuk meningkatkan efisiensi administrasi, mempercepat proses pelaporan pajak, serta mengurangi risiko kesalahan dalam pencatatan [4]. Meskipun sistem ini diharapkan memberikan kemudahan bagi wajib pajak, respons masyarakat terhadap penggunaannya masih beragam. Beberapa pengguna mungkin merasakan manfaat dari sistem ini, sementara yang lain menghadapi kendala teknis atau kesulitan dalam adaptasi. Oleh karena itu, analisis opini masyarakat terhadap

Coretax menjadi langkah penting dalam mengevaluasi efektivitasnya.

Analisis opini publik terhadap sistem perpajakan digital seperti Coretax menjadi krusial karena keberhasilan implementasi teknologi tidak hanya ditentukan oleh kecanggihan sistem, tetapi juga oleh tingkat penerimaan dan kepercayaan masyarakat. Kepercayaan publik yang tinggi akan mendorong adopsi yang lebih luas serta meningkatkan kepatuhan pajak, sedangkan persepsi negatif dapat menghambat efektivitas kebijakan digitalisasi perpajakan [5]. Dengan demikian, pemahaman atas opini publik memberikan gambaran apakah Coretax benar-benar mampu menjawab kebutuhan wajib pajak atau justru menimbulkan hambatan baru. Agar adopsi dapat berjalan optimal, kepercayaan publik perlu diperkuat melalui penyelarasan antara inovasi teknologi dan kebijakan yang akuntabel serta partisipatif, disertai integrasi digital yang ramah pengguna serta komunikasi dan edukasi fiskal yang efektif [6].

Untuk memperoleh pemahaman mendalam mengenai persepsi masyarakat terhadap Coretax, digunakan analisis sentimen sebagai metode untuk menggali dan mengevaluasi pandangan publik [7]. Analisis ini berfokus pada pengelompokan teks dari dokumen, kalimat, atau fitur tertentu [8][9]. Setiap kalimat dapat memiliki kecenderungan sentimen positif, negatif, atau netral [10]. Media sosial, khususnya X, kerap menjadi sarana masyarakat dalam menyampaikan opini terkait kebijakan pemerintah, termasuk perpajakan. Melalui teknik *web crawling*, data mengenai Coretax dapat dikumpulkan dan dianalisis guna mengidentifikasi pola sentimen. Oleh karena itu, analisis sentimen berbasis *machine learning* menjadi metode yang efektif untuk memahami persepsi publik [11].

Penelitian ini selanjutnya memanfaatkan tiga algoritma *machine learning*, yaitu *Naive Bayes Classifier* (NBC), *Support Vector Machine* (SVM), dan *Long Short Term Memory* (LSTM), untuk mengklasifikasikan sentimen masyarakat terhadap Coretax. Algoritma NBC dipilih karena sederhana, cepat, dan tingkat akurasinya yang tinggi [12]. Algoritma SVM dipilih karena mempunyai performa yang lebih baik apabila dibandingkan dengan metode *machine learning* lain [13]. Sementara itu, algoritma LSTM dipilih karena keunggulannya dalam memahami konteks jangka panjang dan urutan kata dalam sebuah kalimat, sehingga lebih efektif untuk menangani permasalahan data teks yang kompleks [14]. Dengan demikian, perbandingan antara NBC dan SVM sebagai algoritma klasik dengan LSTM sebagai algoritma *deep learning* menjadi menarik untuk mengevaluasi efektivitas masing-masing dalam menganalisis sentimen masyarakat terhadap Coretax.

Dalam mengkaji terkait penelitian ini, terdapat beberapa penelitian terdahulu. Pertama, penelitian Wijaya dan Rifqo [15] menganalisis sentimen publik terkait Coretax menggunakan algoritma NBC. Dari analisis terhadap 2.858 data *tweet*, didapati bahwa NBC menunjukkan akurasi yang baik, dengan akurasi mencapai 81%. Kedua, penelitian Fathoni et al. [16] mengenalisis sentimen publik terkait sistem perpajakan digital Coretax menggunakan algoritma NBC. Dari analisis terhadap 899 data *tweet*, didapati bahwa algoritma NBC efektif dengan akurasi mencapai 77%.

Ketiga, penelitian Ramadhani et al. [17] menganalisis sentimen terkait makan siang gratis menggunakan metode NBC dan SVM di platform Twitter. Dari 1.028 *tweet* yang dianalisis, ditemukan bahwa SVM menunjukkan performa lebih baik dengan akurasi 75,39% dibandingkan NBC yang hanya mencapai 68,97%. Keempat, penelitian Novianti et al. [18] yang menganalisis terkait pelayanan daring menggunakan komparasi algoritma NBC dan LSTM. Berdasarkan penelitian, ditemukan bahwa akurasi algoritma NBC lebih baik, yaitu mencapai 83,69%, sedangkan LSTM hanya mencapai 53,12%.

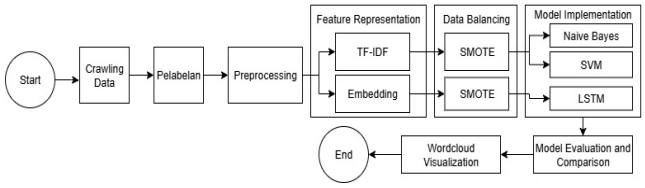
Kelima, penelitian Rabbani et al. [19], menganalisis sentimen terhadap kenaikan harga BBM dengan menggunakan tiga kernel SVM, yaitu linear, RBF, dan polinomial, serta menerapkan teknik SMOTE. Hasil penelitian menunjukkan bahwa kernel RBF dengan pembobotan kata TF-IDF memberikan kinerja terbaik, mencapai akurasi 87%. Selain itu, penerapan SMOTE *oversampling* pada kernel polinomial dengan pembagian data 70:30 dan 80:20 terbukti meningkatkan kinerja algoritma hingga 2%. Terakhir, penelitian Gea et al. [20] menganalisis sentimen terkait direktorat jenderal pajak pada tahun 2023. Hasil penelitian menunjukkan bahwa sentimen terdiri dari 40,5% netral, 39,4% positif, dan 20,1% negatif

Berdasarkan berbagai penelitian, analisis sentimen terkait kebijakan publik maupun sistem perpajakan digital masih didominasi oleh penggunaan algoritma klasik seperti NBC dan SVM. Kajian mengenai Coretax umumnya terbatas pada satu metode dan jumlah data yang relatif kecil, sementara penelitian terkait DJP secara umum masih berfokus pada pemetaan persepsi publik tanpa eksplorasi perbandingan algoritma. Hingga saat ini, belum banyak studi yang secara komprehensif membandingkan efektivitas algoritma klasik dengan model *deep learning* seperti LSTM, yang memiliki keunggulan dalam memproses data sekuensial dan konteks panjang. Selain itu, penelitian yang secara khusus menganalisis sentimen publik terhadap DJP Online saat awal peluncuran juga belum ditemukan, sehingga analisis terhadap Coretax menjadi relevan sekaligus mengisi celah penelitian. Penelitian ini menawarkan kontribusi baru dengan membandingkan performa NBC, SVM (dengan kernel linear, RBF, polinomial, dan sigmoid), serta LSTM, baik dengan maupun tanpa penerapan SMOTE.

Tujuan utama penelitian ini adalah mengevaluasi performa dan efektivitas masing-masing algoritma dalam menganalisis sentimen publik terhadap Coretax. Selain itu, penelitian ini juga menganalisis kata-kata dominan melalui visualisasi *word cloud* untuk memahami persepsi publik terhadap Coretax. Hasilnya diharapkan dapat menjadi dasar pemilihan metode analisis sentimen yang lebih akurat dan efisien, sekaligus memberikan wawasan penting bagi pemerintah dalam meningkatkan kualitas layanan perpajakan digital di Indonesia.

II. METODE

Alur tahap-tahap dalam penelitian ini dapat dilihat pada Gambar 1.



Gambar 1. Metodologi Penelitian

A. Pengumpulan Data

Tahapan awal dalam penelitian ini adalah pengumpulan data yang dilakukan dengan menggunakan teknik *crawling* data melalui platform X atau Twitter menggunakan *Tweet* Harvest yang dijalankan menggunakan google collab. Data dikumpulkan menggunakan kata kunci "coretax" pada bulan Januari - April 2025. Pengumpulan data dilakukan secara bertahap karena adanya limitasi dari platform X atau Twitter. Pengumpulan data di sini hanya dikhususkan untuk *tweet* berbahasa Indonesia. Setelah pengumpulan data, terkumpul data sebanyak 6.728 data. Kemudian, data akan disatukan untuk membentuk sebuah dataset mentah yang komprehensif dalam bentuk excel.

B. Pelabelan Data

Dataset yang telah dikumpulkan melalui media sosial X merupakan dataset yang belum berlabel, sehingga perlu dilakukan pelabelan pada dataset agar dapat dipelajari. Pelabelan data dilakukan secara manual yang dibagi berdasarkan 3 kelas, yaitu positif, negatif, dan netral.

C. Preprocessing

Pada tahap *preprocessing*, data diolah dengan cara dipilih dan diubah dari bentuk yang tidak terstruktur menjadi lebih terstruktur. Tujuan utama dari proses ini adalah untuk mendapatkan data yang lebih relevan, sehingga isinya lebih mudah diproses dalam suatu sistem [21]. Lebih lanjut, tahapan *preprocessing* dijelaskan sebagai berikut [22]:

a. *Cleaning*: Tahap ini melibatkan penghapusan karakter serta tanda baca yang tidak diperlukan dalam teks guna mengurangi *noise* pada data. Proses pembersihan mencakup eliminasi *mention*, *hashtag*, *URL*, serta karakter selain huruf dan angka.
b. *Case folding*: Proses ini bertujuan untuk mengonversi seluruh huruf pada komentar menjadi format huruf kecil. Tahapan ini dilakukan setelah proses pembersihan data selesai. Ketidakkonsistenan dalam penggunaan huruf besar dan kecil kerap dijumpai pada unggahan di platform X.
c. *Stopword-removal*: Penghapusan *stopword* berorientasi untuk mengeliminasi kata-kata yang kerap terlihat dalam jumlah besar tetapi kurang signifikan dalam analisis teks. Contoh kata yang termasuk dalam kategori ini antara lain "yang," "tapi," "masih," dan sejenisnya.
d. *Tokenization*: Pemisahan kalimat menjadi unit-unit kecil berupa kata atau karakter sesuai kebutuhan, yang disebut token, sehingga membentuk kata dengan makna tertentu.
e. *Normalization*: Proses ini dilakukan untuk menyeragamkan kata-kata yang tidak baku atau memiliki variasi penulisan agar konsisten.
f. *Stemming*: Proses ini mengekstraksi kata ke bentuk dasarnya dengan cara menghapus imbuhan, awalan, atau akhiran tanpa mengubah makna utama kata tersebut.
g. *Preprocessing* khusus LSTM: Selain tahapan a–e, untuk model berbasis LSTM teks diproses lebih lanjut dengan mengubah setiap token menjadi indeks numerik sesuai kamus (*vocabulary*). Karena panjang kalimat berbeda-beda, setiap sekuens kemudian dipadukan (*padding*) agar memiliki panjang yang sama, dengan menambahkan nilai nol di akhir sekuens yang lebih pendek. Selanjutnya, indeks kata dipetakan ke dalam vektor berdimensi tetap melalui *Embedding Layer*, sehingga kata-kata dengan makna serupa memiliki representasi vektor yang berdekatan. Proses ini memungkinkan LSTM menangkap hubungan semantik antar kata dan konteks kalimat secara efektif.

Penerapan teknik *preprocessing* yang optimal berperan krusial dalam meningkatkan kinerja model analisis, terutama ketika digunakan pada tugas klasifikasi teks [23].

D. Feature Representation

Feature representation yang digunakan dalam penelitian ini terdiri dari TF-IDF dan *word embedding*, yang pemilihannya tergantung dari jenis algoritma yang digunakan, dijabarkan sebagai berikut:

a. TF-IDF: Untuk model berbasis *machine learning* klasik seperti *Naive Bayes* dan SVM, representasi teks menggunakan TF-IDF (*Term Frequency – Inverse Document Frequency*). Tahap ini melibatkan pemberian bobot pada tiap kata menggunakan metode TF-IDF. Metode TF-IDF adalah teknik dalam pemrosesan teks untuk menilai pentingnya suatu kata dalam sebuah dokumen dibandingkan dengan koleksi dokumen lainnya. TF (*Term Frequency*) mengukur frekuensi kemunculan kata dalam satu dokumen, sementara IDF (*Inverse Document Frequency*) menilai seberapa jarang kata tersebut muncul di seluruh dokumen. Nilai TF dikombinasikan dengan IDF melalui perkalian untuk menghasilkan bobot kata. Bobot ini menunjukkan tingkat kepentingan kata dalam konteks dokumen dan koleksi dokumen secara keseluruhan [24].

b. Embedding: Sementara itu, untuk model berbasis *deep learning* (LSTM), digunakan representasi teks berupa *word embedding*. *Embedding* memungkinkan kata-kata direpresentasikan dalam bentuk vektor dengan dimensi tertentu yang mampu menangkap makna semantik antar kata.

E. Penerapan Algoritma

Tahapan ini berisi proses penerapan SMOTE, menerapkan algoritma SVM, NBC, dan LSTM, serta membandingkan ketiga algoritma berdasarkan performa. Tahap penerapan algoritma dijabarkan sebagai berikut:

a. Penyeimbangan data menggunakan SMOTE: Dataset dalam penelitian ini memiliki data yang tidak seimbang. Dataset yang tidak seimbang memiliki distribusi kelas target yang tidak proporsional, sehingga dapat menyebabkan model cenderung lebih akurat dalam mengenali kelas mayoritas dibandingkan kelas minoritas [25]. Untuk mengatasi permasalahan ini, diterapkan teknik *resampling* data, yaitu SMOTE *oversampling*. SMOTE merupakan salah satu metode *oversampling* yang paling banyak digunakan karena dirancang untuk memaksimalkan efektivitas *oversampling* acak.
b. Klasifikasi menggunakan *Support Vector Machine* (SVM) dan *Naive Bayes Classifier* (NBC): NBC adalah metode klasifikasi sederhana yang menggabungkan kombinasi nilai frekuensi basis data berdasarkan teorema *Bayes* untuk menghitung semua probabilitas [26]. Sementara SVM digunakan sebagai metode mencari *hyperplane* terbaik dengan memaksimalkan jarak antar kelas [27]. Lebih lanjut, algoritma SVM yang diterapkan dalam penelitian ini menggunakan 4 kernel, yaitu kernel Linear, Polinomial, RBF, dan Sigmoid. Persamaan dari fungsi kernel-kernel tersebut bisa dilihat pada nomor 1, 2, 3, dan 4 [28].

$$K = (Xi, Xj) = (Xi.Xj) \quad (1)$$

Kernel linear mengkalkulasikan hasil perkalian dalam *inner product* antara dua vektor Xi dan Xj. Kernel ini sesuai digunakan apabila data dapat dipisahkan secara linear.

$$K = (Xi, Xj) = (Xi, Xj + 1)^P \quad (2)$$

Kernel polinomial menyertakan konstanta pada *inner product* $Xi \cdot Xj$ kemudian menaikkannya ke pangkat p. Nilai p menunjukkan derajat polinomial yang berperan dalam menentukan tingkat kompleksitas pemisahan data.

$$K = (Xi, Xj) = e^{-\gamma(Xi, Xj)^2} \quad (3)$$

Kernel Gaussian atau RBF didasarkan pada perhitungan jarak kuadrat antara dua buah titik data. Parameter γ mengatur sejauh mana pengaruh tiap titik data: semakin besar nilai γ, semakin sempit cakupan pengaruhnya.

$$K = (Xi, Xj) = tanh(\eta Xi .Xj + v) \quad (4)$$

Kernel sigmoid memiliki kemiripan dengan fungsi aktivasi pada jaringan saraf tiruan. Parameter η berfungsi sebagai faktor skala, sedangkan v berperan sebagai bias yang memengaruhi bentuk kurva sigmoid.

Penentuan nilai optimal untuk parameter esensial pada setiap kernel, seperti C, gamma, dan *degree*, tidak dilakukan secara manual, melainkan dicari secara otomatis menggunakan metode *GridSearchCV*. Pendekatan ini secara sistematis menguji berbagai kombinasi parameter untuk menemukan set parameter terbaik yang mampu menghasilkan performa model paling optimal [29].

c. Klasifikasi menggunakan LSTM: LSTM (*Long Short-Term Memory*) adalah model *deep learning* berbasis *Recurrent Neural Network* (RNN) yang dirancang untuk mengolah data sekuensial dengan menangkap dependensi jangka panjang [30]. Penelitian ini menggunakan arsitektur LSTM *unidirectional* dengan lapisan *embedding*, satu *hidden layer* LSTM berisi 128 unit, serta *dense layer* untuk klasifikasi multi-kelas. Representasi teks diperoleh melalui *word embedding* berdimensi 128 yang dilatih bersamaan dengan model.
d. Perbandingan Performa Algoritma: Setelah melakukan pengujian dan mendapati nilai performa dari masing-masing algoritma, data akan dibandingkan. Perbandingan meliputi akurasi, presisi, recall, dan f1 score. Nilai-nilai tersebut didapat dari persamaan berikut.

$$Akurasi = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \quad (5)$$

$$Presisi = \frac{TP}{TP+FP} \times 100\% \quad (6)$$

$$Recall = \frac{TP}{TP+FN} \times 100\% \quad (7)$$

$$F1\ Score = 2 \times \frac{Presisi \times Recall}{Presisi \times Recall} \times 100\% \quad (8)$$

Keterangan dari rumus-rumus di atas [31]

- TP artinya *True* Positif,
- TN artinya *True* Negatif,
- FP artinya *False* Positif, dan
- FN artinya *False* Negatif.

Perbandingan ini bertujuan untuk menelaah algoritma mana yang menunjukkan performa yang lebih baik. Cakupan perbandingan juga akan mencakup perbandingan penggunaan SMOTE dan tanpa penggunaan SMOTE dalam masing-masing penerapan algoritma.

F. Visualisasi Data (Word Cloud)

Pada tahap ini, peneliti akan menampilkan *word cloud* dari keseluruhan kata-kata yang sering ditulis dari sentimen positif, netral dan negatif.

III. HASIL DAN PEMBAHASAN

A. Dataset

Data dalam penelitian ini dikumpulkan dari media sosial X melalui proses *crawling* yang dilakukan secara *gradual* dari awal peluncuran sistem Coretax (Januari 2025) hingga empat bulan setelahnya, yaitu bulan April 2025. Pengumpulan data diimplementasikan dengan memanfaatkan Google Collab dan bahasa pemrograman Python. Sebanyak 6.728 *tweet* terhimpun dan tersimpan dalam format excel. Hasil *crawling* data dapat dilihat pada Tabel I.

TABEL I
SAMPEL DATA MENTAH

No	*Tweet*
1	Coretax error kah?? Gag muncul tarif ppn nya mana invoice Januari blom baru sampe pertengahan yg udh baru diinput. @kring_pajak
…..	……
6727	Min @kring_pajak utk kode setor PPH Final Setor Sendiri biasanya saya pake akun 411128-403. Tp ini di Coretax kok gk ada ya kode bayar tsb? Mohon bantuannya
6728	#KawanPajak kini perubahan data alamat bisa dilakukan lebih mudah melalui Coretax DJP! Dengan fitur ini Anda tidak perlu datang ke Kantor Pajak untuk memperbarui data. https://t.co/0YEMRuEesg

B. Pelabelan Data

Dalam penelitian ini, proses pelabelan data dilakukan secara manual dengan cermat. Setiap *tweet* yang membahas tentang Coretax dikategorikan ke dalam tiga jenis sentimen, yaitu positif, negatif, dan netral. Positif berarti *tweet* memiliki arti yang baik dan menyatakan keunggulan dari Coretax. Negatif berarti *tweet* memiliki arti yang buruk, yaitu menyatakan kekurangan, makian, dan keluhan terkait Coretax. Sementara netral berarti *tweet* tidak bertendensi ke positif dan negatif. Metode manual dipilih untuk memastikan akurasi dan konsistensi dalam penentuan sentimen, mengingat beragamnya cara pengguna X mengekspresikan opini mereka. Dataset yang telah dilabeli kemudian digunakan sebagai dasar untuk melatih model *Naive Bayes*, SVM, dan LSTM. Hasil pelabelan tersebut dapat dilihat pada Tabel II.

TABEL II
SAMPEL PELABELAN DATA

No	*Tweet*	Label
1	Coretax error kah?? Gag muncul tarif ppn nya mana invoice Januari blom baru sampe pertengahan yg udh baru diinput. @kring_pajak	Negatif
…..	……	…….
6727	Min @kring_pajak utk kode setor PPH Final Setor Sendiri biasanya saya pake akun 411128-403. Tp ini di Coretax kok gk ada ya kode bayar tsb? Mohon bantuannya	Netral
6728	#KawanPajak kini perubahan data alamat bisa dilakukan lebih mudah melalui Coretax DJP! Dengan fitur ini Anda tidak perlu datang ke Kantor Pajak untuk memperbarui data. https://t.co/0YEMRuEesg	Positif

C. Preprocessing Data

Prapemrosesan data merupakan tahapan yang sangat penting dalam analisis sentimen ini. Data *tweet* yang berisi opini mengenai Coretax di Indonesia telah melalui serangkaian proses prapemrosesan untuk memastikan kebersihan data dan kesiapan penggunaannya. Proses ini mencakup pembersihan teks, *case folding*, *stopword removal*, tokenisasi, normalisasi, serta *stemming*. Dengan mengurangi elemen yang tidak relevan dan mengekstraksi fitur linguistik yang signifikan, diharapkan model mampu mengenali pola sentimen dalam data *tweet* dengan lebih akurat. Secara lebih jelas, process *preprocessing* yang dilakukan terlampir dalam Tabel III berikut.

TABEL III
SAMPEL HASIL *PREPROCESSING*

Proses	Hasil Pemrosesan
Text awal	Coretax error kah?? Gag muncul tarif ppn nya mana invoice Januari blom baru sampe pertengahan yg udh baru diinput. @kring_pajak
Cleaning	Coretax error kah Gag muncul tarif ppn nya mana invoice Januari blom baru sampe pertengahan yg udh baru diinput
Case folding	coretax error kah gag muncul tarif ppn nya mana invoice januari blom baru sampe pertengahan yg udh baru diinput
Stopword removal	coretax error gag muncul tarif ppn invoice januari blom baru sampe pertengahan udh baru diinput
Tokenisasi	['coretax', 'error', 'gag', 'muncul', 'tarif', 'ppn', 'invoice', 'januari', 'blom', 'baru', 'sampe', 'pertengahan', 'udh', 'baru', 'diinput']
Normalisasi	['coretax', 'error', 'tidak', 'muncul', 'tarif', 'ppn', 'invoice', 'januari', 'belum', 'baru', 'sampai', 'pertengahan', 'sudah', 'baru', 'input']
Stemming	['coretax', 'error', 'tidak', 'muncul', 'tarif', 'ppn', 'invoice', 'januari', 'belum', 'baru', 'sampai', 'tengah', 'sudah', 'baru', 'input']
Hasil akhir	coretax error tidak muncul tarif ppn invoice januari belum baru sampai tengah sudah baru input

Setelah tahap *preprocessing* yang dilakukan untuk model NBC dan SVM, data untuk model LSTM mengalami tahap tambahan berupa konversi token menjadi indeks numerik dan *padding* agar setiap input memiliki panjang sekuens yang seragam. Setiap kata hasil tokenisasi dipetakan menjadi angka unik sesuai kamus (*vocabulary*) yang dibangun selama proses tokenisasi. Kemudian, sekuens angka ini dipadukan (*padding*) hingga mencapai panjang maksimum 100 token (MAX_LEN=100), sehingga semua input memiliki dimensi yang sama dan siap diproses oleh jaringan LSTM. Selanjutnya, setiap indeks kata diubah menjadi vektor *embedding* berdimensi 128, yang merepresentasikan

kata dalam ruang vektor numerik sehingga LSTM dapat menangkap hubungan semantik antar kata dan konteks sekuensial dalam *tweet*. Hasil *preprocessing* tambahan tersebut terlampir dalam Tabel IV.

TABEL IV
SAMPEL HASIL PREPROCESSING TAMBAHAN LSTM

Proses	Hasil Pemrosesan
Tokenisasi	['coretax', 'error', 'tidak', 'muncul', 'tarif', 'ppn', 'invoice', 'januari', 'belum', 'baru', 'sampai', 'tengah', 'sudah', 'baru', 'input']
Konversi ke indeks	[12, 57, 103, 205, 78, 34, 56, 201, 89, 23, 145, 67, 98, 23, 11]
Padding (MAX_LEN=100)	[12, 57, 103, 205, 78, 34, 56, 201, 89, 23, 145, 67, 98, 23, 11, 0, 0,0,0,0 … sampai total 100 elemen]
Embedding (dimensi 128)	[[0.12, -0.05, 0.33, … , -0.10], …]

Dengan demikian, *preprocessing* untuk LSTM tidak hanya membersihkan dan menormalkan teks, tetapi juga mengubah data menjadi format numerik dan vektor yang kompatibel dengan arsitektur *deep learning*.

D. Feature Representation

Setelah tahap *preprocessing*, bobot kata dikalkulasi menggunakan metode *Term Frequency-Inverse Document Frequency* (TF-IDF), yang mengevaluasi frekuensi kemunculan kata dalam suatu dokumen (*term frequency*) serta membandingkannya dengan distribusi kata tersebut di seluruh dokumen dalam kumpulan teks (*inverse document frequency*). Kata yang kerap muncul pada sebuah *tweet* namun jarang muncul di *tweet* lain akan memperoleh bobot TF-IDF yang lebih tinggi, yang menunjukkan tingkat relevansi kata tersebut terhadap *tweet* yang dianalisis. Hasil perhitungan ini dimanfaatkan dalam berbagai analisis lanjutan, seperti klasifikasi teks atau analisis sentimen. Tabel V menyajikan contoh nilai bobot TF-IDF untuk setiap kata.

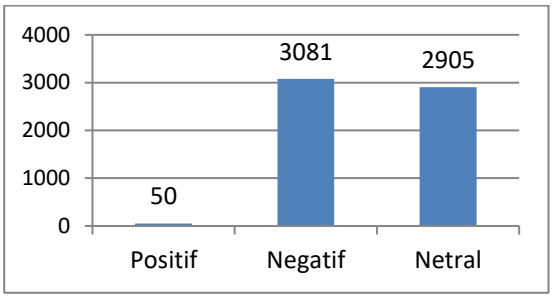
TABEL V
SAMPEL SKOR TF-IDF

Kata	Skor TF-IDF
Jelek	0.398
Program	0.613
Triliun	0.292
…..	…..
Turun	0.247

Selain TF-IDF yang digunakan pada model klasik (NBC dan SVM), untuk model berbasis *deep learning* seperti LSTM, representasi teks dilakukan menggunakan *word embedding*. Data *tweet* yang telah melalui tahapan *preprocessing* (Tabel III) dikonversi menjadi indeks numerik berdasarkan kamus *tokenizer*, kemudian dipadukan (*padding*) hingga panjang sekuens maksimum 100 token (MAX_LEN=100). Setiap indeks kata diubah menjadi vektor *embedding* berdimensi 128, yang menangkap makna semantik kata dan hubungan antar kata dalam konteks sekuensial. Vektor *embedding* inilah yang menjadi input LSTM untuk melakukan klasifikasi sentimen. Dengan cara ini, LSTM dapat mempelajari pola sekuensial dan konteks antar kata secara lebih efektif dibandingkan TF-IDF, tanpa perlu menampilkan tabel baru karena semua *preprocessing* sudah tercakup di Tabel IV.

E. Klasifikasi Data

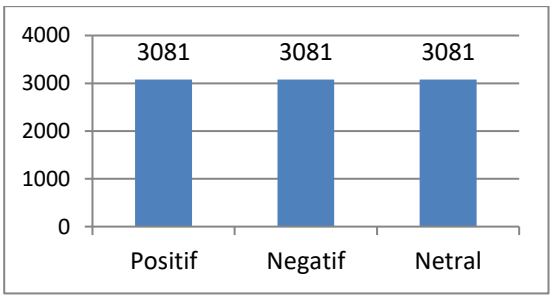
Sebelum memasuki implementasi algoritma, penting untuk melihat statistik data untuk mencegah ketidakseimbangan data dan untuk melihat keseluruhan sentimen. Statistik data yang dilatih berupa jumlah data yang positif, netral, dan negatif dapat dilihat pada gambar 2.



Gambar 2. Total Keseluruhan Klasifikasi Data

Setelah hasil *preprocessing* data dan pelabelan, didapati keseluruhan data *tweet* berjumlah 6.036, hasil ini mengalami pengurangan dari total *tweet* yang di*crawling* karena adanya redudansi data. Total data *tweet* terdiri dari *tweet* yang positif berjumlah 50 *tweet* atau 0,83%, negatif 3081 *tweet* atau 51,05%, dan netral 2905 *tweet* atau 48,11%. Dari distribusi awal ini terlihat adanya ketidakseimbangan kelas yang signifikan, di mana kelas positif memiliki proporsi yang sangat kecil dibandingkan dengan kelas netral maupun negatif. Ketimpangan distribusi tersebut berpotensi menurunkan performa model klasifikasi karena model cenderung bias terhadap kelas mayoritas.

Untuk mengatasi permasalahan ketidakseimbangan tersebut, digunakan teknik *Synthetic Minority Oversampling Technique* (SMOTE). Hasil distribusi data setelah penerapan SMOTE ditunjukkan pada Gambar 3, di mana jumlah data pada masing-masing kelas menjadi seimbang dengan total data positif sebanyak 3081, negatif sebanyak 3081, dan netral sebanyak 3081.



Gambar 3. Distribusi Data Setelah SMOTE

Hal ini diharapkan dapat membantu model dalam mengenali pola dari ketiga kelas secara lebih proporsional sehingga meningkatkan akurasi dan keandalan hasil analisis sentimen.

F. Penerapan Algoritma NBC

Setelah keseluruhan data dilakukan teknik SMOTE, maka selanjutnya diterapkan algoritma *Naive Bayes* Classifier (NBC). Pengujian dilakukan dua kali, yaitu pada data yang tidak diterapkan SMOTE dan data yang diterapkan SMOTE. Dataset dalam masing-masing pengujian dibagi menjadi dua menggunakan teknik *Holdout Validation*, yaitu 70% data sebagai data latih dan 30% sebagai data uji.

Pada skenario tanpa SMOTE, proses penerapan NBC dimulai dengan melakukan optimasi parameter alpha menggunakan metode *Grid Search* dengan validasi silang 10-fold dan metrik evaluasi F1-Macro. Berdasarkan hasil optimasi, diperoleh nilai alpha sebesar 1. Model dengan parameter tersebut kemudian dilatih menggunakan data latih hasil representasi TF-IDF dan digunakan untuk memprediksi data uji. Evaluasi performa dilakukan melalui perhitungan metrik evaluasi dan divisualisasikan menggunakan *classification report* untuk melihat distribusi prediksi pada masing-masing kelas. Hasil *classification report* dapat dilihat pada Gambar 4.

	precision	recall	f1-score	support
Negatif	0.86	0.85	0.86	924
Netral	0.84	0.87	0.85	872
Positif	0.00	0.00	0.00	15
accuracy			0.85	1811
macro avg	0.57	0.57	0.57	1811
weighted avg	0.84	0.85	0.85	1811

Akurasi: 0.8515

Gambar 4. Evaluasi Algoritma NBC Tanpa SMOTE

Berdasarkan hasil evaluasi, diperoleh akurasi sebesar 85,15%, presisi 56,66%, recall 57,33%, dan F1-score 57%. Nilai presisi dan recall yang relatif rendah menunjukkan bahwa model belum optimal dalam mengenali kelas minoritas. Hal ini terjadi karena distribusi data yang tidak seimbang membuat NBC lebih sering memprediksi kelas mayoritas dengan benar, tetapi gagal mengenali kelas lain secara konsisten.

Sementara itu, pada skenario dengan SMOTE, proses penerapan NBC dimulai dengan melakukan penyeimbangan data melalui penambahan sampel sintetis pada kelas minoritas agar distribusi kelas menjadi lebih proporsional. Setelah itu, dilakukan optimasi parameter menggunakan *Grid Search* dengan validasi silang 10-fold dan metrik evaluasi F1-Macro, yang menghasilkan nilai alpha sebesar 0,1. Model dengan parameter tersebut kemudian dilatih menggunakan data latih hasil representasi TF-IDF dan digunakan untuk memprediksi data uji. Evaluasi performa dilakukan dengan metrik yang sama dan divisualisasikan melalui *classification report*. Hasil *classification report* dapat dilihat pada Gambar 5.

	precision	recall	f1-score	support
Negatif	0.87	0.84	0.85	924
Netral	0.86	0.86	0.86	925
Positif	0.96	0.99	0.98	925
accuracy			0.90	2774
macro avg	0.90	0.90	0.90	2774
weighted avg	0.90	0.90	0.90	2774

Akurasi: 0.8969

Gambar 5. Evaluasi NBC dengan SMOTE

Berdasarkan hasil evaluasi, diperoleh akurasi sebesar 89,69%, presisi 89,66%, recall 89,66%, dan F1-score 89,66%. Peningkatan signifikan pada semua metrik menunjukkan bahwa SMOTE berhasil membuat model lebih sensitif terhadap kelas minoritas. Keseimbangan antara presisi dan recall mencerminkan distribusi prediksi yang jauh lebih adil di seluruh kelas.

G. Penerapan Algoritma SVM Kernel Linear

Pada skenario tanpa SMOTE, dataset dibagi menjadi 70% data latih dan 30% data uji menggunakan metode *Holdout Validation*. Proses optimasi parameter C dilakukan melalui *Grid Search* dengan validasi silang 10-fold menggunakan metrik F1-Macro. Dari hasil optimasi, diperoleh nilai C terbaik sebesar 1. Model dengan parameter tersebut kemudian dilatih menggunakan representasi TF-IDF dan diuji pada data uji. Evaluasi dilakukan menggunakan metrik akurasi, presisi, recall, dan F1-score, serta divisualisasikan dalam bentuk *classification report* sebagaimana ditunjukkan pada Gambar 6.

	precision	recall	f1-score	support
Negatif	0.82	0.89	0.86	924
Netral	0.87	0.81	0.84	872
Positif	0.00	0.00	0.00	15
accuracy			0.85	1811
macro avg	0.57	0.57	0.57	1811
weighted avg	0.84	0.85	0.84	1811

Akurasi: 0.8454

Gambar 6. Evaluasi SVM Kernel Linear Tanpa SMOTE

Hasil evaluasi menunjukkan bahwa kernel linear menghasilkan akurasi sebesar 84,54%, presisi 56,33%, recall 56,66%, dan F1-score 56,66%, dengan performa yang relatif stabil meskipun recall untuk kelas minoritas masih lebih rendah. Nilai metrik yang moderat menunjukkan bahwa kernel linear dapat memisahkan kelas dengan cukup baik, meskipun masih kesulitan dalam mengenali kelas minoritas. Hal ini karena distribusi data yang tidak seimbang mempengaruhi proses pembelajaran *hyperplane*.

Pada skenario dengan SMOTE, data terlebih dahulu diseimbangkan melalui penambahan sampel sintetis pada kelas minoritas agar distribusi kelas lebih proporsional. Proses pelatihan dan evaluasi kemudian dilakukan kembali dengan langkah yang sama seperti sebelumnya, dengan

parameter C = 1 pada model yang digunakan. Hasil *classification report* terlampir dalam Gambar 7.

	precision	recall	f1-score	support
Negatif	0.84	0.89	0.86	924
Netral	0.89	0.83	0.85	925
Positif	0.99	1.00	0.99	925
accuracy			0.90	2774
macro avg	0.90	0.90	0.90	2774
weighted avg	0.90	0.90	0.90	2774

Akurasi: 0.9037

Gambar 7. Evaluasi SVM Kernel Linear dengan SMOTE

Hasil evaluasi menunjukkan peningkatan dengan akurasi 90,37%, presisi 90,66%, recall 90,66%, dan F1-score 90%. Nilai metrik yang tinggi dan seimbang menandakan bahwa kernel linear sangat responsif terhadap distribusi data yang proporsional. Hal ini membuktikan bahwa SVM bekerja sangat efektif saat data telah diseimbangkan.

H. Penerapan Algoritma SVM Kernel RBF

Tanpa penerapan SMOTE, dataset dibagi menggunakan rasio 70% data latih dan 30% data uji, kemudian dilakukan optimasi parameter C dan γ melalui *Grid Search* dengan validasi silang 10-fold menggunakan metrik F1-Macro. Model terbaik yang diperoleh memiliki parameter C = 1 dan γ = 1, kemudian dilatih dengan representasi TF-IDF dan diuji pada data uji. Evaluasi dilakukan melalui perhitungan metrik performa dan divisualisasikan menggunakan *classification report* untuk melihat distribusi prediksi antar kelas. Hasil *classification report* terlampir dalam Gambar 8.

	precision	recall	f1-score	support
Negatif	0.82	0.91	0.86	924
Netral	0.89	0.80	0.84	872
Positif	0.00	0.00	0.00	15
accuracy			0.85	1811
macro avg	0.57	0.57	0.57	1811
weighted avg	0.85	0.85	0.85	1811

Akurasi: 0.8515

Gambar 8. Evaluasi SVM Kernel RBF Tanpa SMOTE

Evaluasi dengan *classification report* menunjukkan bahwa kernel RBF memperoleh akurasi 85,15%, presisi 57%, recall 57%, dan F1-score 56,66%. Metrik ini menunjukkan bahwa kernel RBF mampu menangkap pola non-linear dengan lebih baik dibanding kernel linear, tetapi performanya tetap terbatas oleh distribusi data yang tidak seimbang. Nilai F1 yang belum optimal menandakan masih adanya bias terhadap kelas mayoritas.

Dengan penerapan SMOTE, dataset diseimbangkan terlebih dahulu agar jumlah sampel antar kelas lebih proporsional. Setelah itu, proses pelatihan dan pengujian dilakukan dengan metode yang sama menggunakan kernel RBF. Model terbaik yang diperoleh memiliki parameter C = 1 dan γ = 1. Hasil *classification report* terlampir dalam Gambar 9.

	precision	recall	f1-score	support
Negatif	0.83	0.91	0.87	924
Netral	0.90	0.81	0.85	925
Positif	1.00	1.00	1.00	925
accuracy			0.91	2774
macro avg	0.91	0.91	0.91	2774
weighted avg	0.91	0.91	0.91	2774

Akurasi: 0.9070

Gambar 9. Evaluasi SVM Kernel RBF dengan SMOTE

Evaluasi menunjukkan peningkatan performa dengan akurasi 90,7%, presisi 91%, recall 90,66%, dan F1-score 90,66%, sehingga distribusi performa antar kelas menjadi lebih seimbang. Lonjakan performa ini mengindikasikan bahwa kernel RBF sangat unggul dalam mengidentifikasi pola kompleks ketika data seimbang. Keselarasan nilai presisi dan recall menunjukkan bahwa model telah mencapai generalisasi yang kuat.

I. Penerapan Algoritma SVM Kernel Polinomial

Pada skenario tanpa SMOTE, dataset dibagi menggunakan perbandingan 70% data latih dan 30% data uji. Parameter yang dioptimasi mencakup C, γ, dan *degree* pada kernel polinomial. Proses optimasi dilakukan menggunakan *Grid Search* dengan validasi silang 10-fold berbasis F1-Macro. Dari hasil optimasi tersebut, diperoleh kombinasi parameter terbaik yaitu C = 1, γ = 1, dan *degree* = 2. Model terbaik kemudian dilatih menggunakan representasi TF-IDF dan diuji pada data uji. Evaluasi performa dilakukan melalui perhitungan metrik dan divisualisasikan menggunakan *classification report* untuk menggambarkan distribusi prediksi pada setiap kelas. Hasil *classification report* ditampilkan pada Gambar 10.

	precision	recall	f1-score	support
Negatif	0.81	0.92	0.86	924
Netral	0.90	0.79	0.84	872
Positif	0.00	0.00	0.00	15
accuracy			0.85	1811
macro avg	0.57	0.57	0.57	1811
weighted avg	0.85	0.85	0.85	1811

Akurasi: 0.8493

Gambar 10. Evaluasi SVM Kernel Polinomial Tanpa SMOTE

Evaluasi melalui *classification report* menunjukkan akurasi sebesar 84,93%, presisi 57%, recall 57%, dan F1-score 56,66%. Kernel polinomial mampu menangkap hubungan antar fitur yang lebih kompleks, meski hasil presisi dan recall antar kelas masih kurang konsisten.

Pada skenario dengan SMOTE, distribusi data diseimbangkan sehingga kelas minoritas memiliki jumlah sampel yang lebih memadai. Setelah proses penyeimbangan dilakukan, pelatihan dan pengujian kembali dijalankan menggunakan kernel polinomial dengan prosedur yang sama. Parameter yang dioptimasi tetap mencakup C, γ, dan *degree*, dan hasil *Grid Search* menunjukkan kombinasi terbaik yaitu C = 10, γ = 1, dan *degree* = 2. Evaluasi performa kemudian dilakukan melalui perhitungan metrik

dan divisualisasikan menggunakan *classification report* untuk melihat distribusi prediksi pada setiap kelas. Hasil *classification report* ditampilkan pada Gambar 11.

	precision	recall	f1-score	support
Negatif	0.82	0.89	0.85	924
Netral	0.89	0.80	0.84	925
Positif	0.99	1.00	1.00	925
accuracy			0.90	2774
macro avg	0.90	0.90	0.90	2774
weighted avg	0.90	0.90	0.90	2774

Akurasi: 0.8965

Gambar 11. Evaluasi SVM Kernel Polinomial dengan SMOTE

Setelah dilakukan pelatihan ulang dengan prosedur yang sama, hasil evaluasi memperlihatkan adanya peningkatan dengan akurasi 89,65%, presisi 90%, recall 89,66%, dan F1-score 89,66%, sehingga kernel polinomial menghasilkan klasifikasi yang lebih merata dan adil antar kelas.

J. Penerapan Algoritma SVM Kernel Sigmoid

Tanpa penerapan SMOTE, dataset dibagi menjadi 70% data latih dan 30% data uji. Parameter model, yaitu C dan γ, kemudian dioptimasi menggunakan *Grid Search* dengan validasi silang 10-fold berdasarkan metrik F1-Macro. Dari proses optimasi tersebut diperoleh parameter terbaik yaitu C = 10 dan γ = 0,1. Model hasil optimasi kemudian dilatih menggunakan representasi TF-IDF dan diuji pada data uji. Evaluasi dilakukan menggunakan metrik performa (akurasi, presisi, recall, dan F1-score) dan divisualisasikan melalui *classification report* untuk melihat distribusi prediksi antar kelas. Hasil evaluasi ditunjukkan pada Gambar 12.

	precision	recall	f1-score	support
Negatif	0.82	0.89	0.86	924
Netral	0.87	0.81	0.84	872
Positif	0.00	0.00	0.00	15
accuracy			0.85	1811
macro avg	0.57	0.57	0.57	1811
weighted avg	0.84	0.85	0.84	1811

Akurasi: 0.8454

Gambar 12. Evaluasi SVM Kernel Sigmoid Tanpa SMOTE

Hasil evaluasi menunjukkan bahwa model menghasilkan akurasi sebesar 84,54%, presisi 56,33%, recall 56,66%, dan F1-score 56,66%. Performa ini relatif rendah dan tidak stabil yang mengindikasikan bahwa kernel sigmoid kurang optimal dalam menangani data dengan distribusi yang tidak merata.

Ketika SMOTE diterapkan, distribusi data diseimbangkan dengan menambahkan sampel sintetis pada kelas minoritas. Proses optimasi parameter dilakukan kembali dan menghasilkan nilai terbaik yang tetap konsisten, yaitu C = 10 dan γ = 0,1. Model kemudian dilatih menggunakan representasi TF-IDF dan diuji pada data uji yang telah seimbang. Evaluasi menggunakan metrik performa dilakukan untuk mengukur peningkatan kualitas prediksi, dan hasilnya divisualisasikan melalui *classification report* pada Gambar 13.

	precision	recall	f1-score	support
Negatif	0.84	0.89	0.86	924
Netral	0.89	0.83	0.86	925
Positif	0.99	1.00	0.99	925
accuracy			0.90	2774
macro avg	0.90	0.90	0.90	2774
weighted avg	0.90	0.90	0.90	2774

Akurasi: 0.9045

Gambar 13. Evaluasi SVM Kernel Sigmoid dengan SMOTE

Hasil evaluasi menunjukkan peningkatan performa model secara signifikan, dengan akurasi sebesar 90,45%, presisi 90,66%, recall 90,66%, dan F1-score 90,33%. Temuan ini mengindikasikan bahwa penerapan SMOTE mampu meningkatkan efektivitas kernel sigmoid dalam mengenali kelas minoritas sekaligus memperbaiki keseimbangan prediksi antar kelas.

K. Penerapan Algoritma LSTM

Pada skenario tanpa penerapan SMOTE, model LSTM dibangun dengan arsitektur yang terdiri atas *embedding layer* berukuran 128, diikuti oleh LSTM *layer* dengan 128 unit, *dropout layer* sebesar 0,3 untuk mencegah *overfitting*, *dense layer* dengan 64 unit, serta *output layer* dengan fungsi aktivasi softmax untuk klasifikasi tiga kelas sentimen. Proses pelatihan dilakukan menggunakan data latih asli dengan pemantauan nilai *validation loss* melalui mekanisme *early stopping* untuk menghentikan pelatihan secara otomatis ketika model tidak lagi mengalami peningkatan kinerja. Evaluasi terhadap performa model dilakukan pada data uji, dan hasilnya divisualisasikan melalui *classification report* sebagaimana ditunjukkan pada Gambar 14.

	precision	recall	f1-score	support
Negatif	0.51	1.00	0.68	924
Netral	0.00	0.00	0.00	872
Positif	0.00	0.00	0.00	15
accuracy			0.51	1811
macro avg	0.17	0.33	0.23	1811
weighted avg	0.26	0.51	0.34	1811

Akurasi: 0.5102

Gambar 14. Evaluasi LSTM Tanpa SMOTE

Berdasarkan hasil evaluasi, diperoleh akurasi sebesar 51,02%, presisi 17%, recall 33,33%, dan F1-score 22,66%. Hasil ini mengindikasikan bahwa meskipun LSTM telah mampu mempelajari pola sekuensial dalam teks, performanya masih terbatas akibat distribusi kelas yang tidak seimbang. Ketimpangan ini tercermin dari rendahnya nilai presisi dan recall pada kelas minoritas, serta kecenderungan model untuk lebih sering memprediksi kelas mayoritas. Dengan demikian, hasil tersebut menunjukkan bahwa keberhasilan model dalam mengklasifikasikan sentimen secara tepat masih terhambat oleh dominasi kelas tertentu dalam data latih.

Sementara itu, pada skenario dengan penerapan SMOTE berbasis *embedding*, representasi *embedding* rata-rata dari

data dimanfaatkan untuk melakukan proses *oversampling*, sehingga jumlah sampel pada setiap kelas menjadi lebih seimbang. Karena hasil *oversampling* berbasis *embedding* menghasilkan representasi vektor berdimensi tetap (128) dan tidak lagi berbentuk sekuensial, maka digunakan model klasifikasi berbasis *fully-connected layer* (*dense network*) alih-alih arsitektur LSTM penuh. Model tersebut kemudian dilatih menggunakan data hasil *oversampling*, dan evaluasi performanya pada data uji divisualisasikan melalui *classification report* sebagaimana pada Gambar 15.

	precision	recall	f1-score	support
Negatif	0.57	0.94	0.71	924
Netral	0.81	0.26	0.40	872
Positif	0.00	0.00	0.00	15
accuracy			0.61	1811
macro avg	0.46	0.40	0.37	1811
weighted avg	0.68	0.61	0.55	1811

Akurasi: 0.6074

Gambar 15. Evaluasi LSTM dengan SMOTE

Secara umum, penerapan SMOTE berbasis *embedding* menunjukkan peningkatan signifikan dalam kemampuan model mengenali kelas minoritas. Model menghasilkan prediksi yang lebih seimbang antar kelas dengan nilai akurasi sebesar 60,74%, presisi 46%, recall 40%, dan F1-score 37%. Perbandingan kedua skenario menunjukkan bahwa algoritma LSTM memiliki fleksibilitas tinggi dalam menangani permasalahan klasifikasi teks multikelas. Penerapan SMOTE berbasis *embedding* lebih efektif dalam menyeimbangkan prediksi antara kelas mayoritas dan minoritas. Hal ini disebabkan oleh kemampuan SMOTE dalam menghasilkan sampel sintetis yang merepresentasikan distribusi kelas minoritas secara lebih baik di ruang *embedding*, sehingga model dapat mempelajari pola representatif dari seluruh kelas secara lebih proporsional. Sementara itu, meskipun performa model tanpa SMOTE relatif lebih rendah, hasil tersebut tetap menunjukkan potensi yang baik ketika distribusi kelas tidak terlalu timpang.

L. Perbandingan Algoritma NBC, SVM, dan LSTM

Berdasarkan hasil pengujian yang telah dilakukan, Tabel VI menyajikan perbandingan performa seluruh algoritma klasifikasi, yaitu NBC, SVM (dengan berbagai kernel), dan LSTM, baik sebelum maupun sesudah penerapan teknik SMOTE. Perbandingan ini memberikan gambaran menyeluruh mengenai dampak signifikan dari proses penyeimbangan data terhadap kualitas prediksi, khususnya dalam konteks klasifikasi sentimen teks multikelas. Secara umum, hasil dalam tabel menunjukkan adanya peningkatan yang konsisten pada metrik akurasi, presisi, recall, dan F1-score setelah dilakukan SMOTE, yang menandakan bahwa teknik *oversampling* ini merupakan faktor krusial dalam mengatasi bias terhadap kelas mayoritas.

TABEL VI
PERBANDINGAN PERFORMA ALGORITMA

Kondisi SMOTE	Algoritma	Akurasi	Presisi	Recall	F1-score
Tanpa SMOTE	NBC	85,15%	56,66%	57,33%	57%
	SVM - Linear	84,54%	56,33%	56,66%	56,66%
	SVM - RBF	85,15%	57%	57%	56,66%
	SVM - Polinomial	84,93%	57%	57%	56,66%
	SVM - Sigmoid	84,54%	56,33%	56,66%	56,66%
	LSTM	51,02%	17%	33,33%	22,66%
Dengan SMOTE	NBC	89,69%	89,66%	89,66%	89,66%
	SVM - Linear	90,37%	90,66%	90,66%	90%
	SVM - RBF	90,7%	91%	90,66%	90,66%
	SVM - Polinomial	89,65%	90%	89,66%	89,66%
	SVM - Sigmoid	90,45%	90,66%	90,66%	90,33%
	LSTM	60,74%	46%	40%	37%

Hasil pada tabel di atas menunjukkan pola yang konsisten sebelum penerapan SMOTE. Seluruh model konvensional seperti NBC dan SVM menunjukkan performa yang relatif seragam, dengan akurasi berada pada rentang 84–85% dan F1-score sekitar 56–57%. Konsistensi ini menunjukkan bahwa seluruh model menghadapi permasalahan yang sama, yaitu ketidakseimbangan kelas yang signifikan. Akibatnya, model cenderung bias terhadap kelas mayoritas, sehingga akurasi tampak tinggi tetapi sebenarnya bersifat semu, karena presisi dan recall tetap rendah. LSTM menjadi anomali dengan performa yang jauh lebih rendah, yakni akurasi 51,02% dan F1-score hanya 22,66%. Hal ini mencerminkan ketergantungan model berbasis sekuensial terhadap jumlah data yang besar dan distribusi yang seimbang. Kompleksitas arsitektur LSTM yang seharusnya mampu menangkap dependensi temporal dalam teks tidak dapat bekerja optimal karena data yang terbatas dan ketimpangan label membuat model gagal belajar representasi yang efektif.

Setelah dilakukan SMOTE, perubahan signifikan terjadi di seluruh algoritma. Teknik *oversampling* ini memperbaiki distribusi kelas sehingga setiap kategori memiliki peluang pembelajaran yang setara, menghasilkan peningkatan yang konsisten di semua metrik evaluasi. SVM dengan kernel RBF menempati posisi teratas dengan akurasi 90,70% dan F1-score 90,66%, mencerminkan keunggulannya dalam memetakan data ke ruang berdimensi tinggi dan membentuk *decision boundary* yang lebih adaptif terhadap distribusi fitur yang kompleks. Kernel Sigmoid, yang sebelumnya menunjukkan performa paling rendah di antara keluarga SVM, mengalami peningkatan besar hingga mencapai akurasi 90,45%, bahkan melampaui kernel Linear. Hal ini menunjukkan bahwa setelah ketimpangan data diatasi,

fungsi aktivasi sigmoid dapat membentuk margin pemisah yang lebih efektif dan stabil. Sebaliknya, kernel Polinomial yang semula kompetitif justru mengalami sedikit penurunan relatif dibandingkan kernel lain setelah SMOTE, dengan akurasi 89,65%. Penurunan ini dapat dijelaskan oleh kecenderungan kernel polinomial yang lebih sensitif terhadap data sintetis dan berpotensi mengalami *overfitting*, sehingga kemampuan generalisasinya sedikit berkurang.

NBC mengalami peningkatan performa yang stabil hingga akurasi 89,69%, meskipun masih berada di bawah SVM karena keterbatasannya dalam menangkap korelasi antar fitur yang kompleks dalam teks. Perubahan paling menarik terlihat pada pendekatan berbasis LSTM. Setelah SMOTE, pendekatan yang digunakan tidak lagi berupa arsitektur LSTM murni, melainkan klasifikasi berbasis representasi rata-rata *embedding* yang telah diseimbangkan menggunakan SMOTE dan kemudian diproses oleh jaringan saraf *feed-forward*. Pendekatan ini meningkatkan akurasi hingga 60,74% dan F1-score menjadi 37%, membuktikan bahwa penyeimbangan distribusi data berperan penting dalam membantu model mengenali kelas minoritas. Namun, performanya tetap jauh di bawah algoritma klasik karena keterbatasan representasi *embedding* yang dilatih dari awal, ukuran dataset yang kecil, serta arsitektur jaringan yang relatif sederhana. Tanpa *pre-trained embeddings* yang kaya atau volume data yang lebih besar, model berbasis *deep learning* sulit menandingi model berbasis kernel klasik dalam skenario seperti ini.

Secara keseluruhan, hasil ini menegaskan bahwa pemilihan algoritma yang tepat sangat bergantung pada karakteristik data dan strategi pra-pemrosesan. SMOTE terbukti tidak hanya meningkatkan performa klasifikasi tetapi juga mengubah dinamika kekuatan relatif antar algoritma, terlihat dari perubahan posisi kernel Sigmoid dan Polinomial setelah *oversampling*. Untuk dataset teks berskala kecil hingga menengah, SVM (terutama dengan kernel RBF) tetap menjadi pilihan paling efisien dan konsisten. Sebaliknya, pendekatan berbasis *deep learning* seperti LSTM baru dapat menunjukkan potensi penuhnya jika didukung oleh jumlah data yang besar, representasi fitur yang kuat, dan arsitektur yang lebih kompleks. Temuan ini sekaligus menunjukkan bahwa kompleksitas model tidak menjamin performa yang lebih baik apabila aspek fundamental seperti distribusi data dan representasi fitur belum dioptimalkan.

M. Visualisasi Data (Word Cloud)

Dalam setiap sentimen, dilakukan pemetaan kata-kata kunci menggunakan *word cloud* untuk mengidentifikasi istilah yang paling sering muncul dalam setiap kategori sentimen. Analisis ini bertujuan untuk menyimpulkan kata kunci yang dominan dalam pembahasan terkait topik implementasi Coretax. Hasil visualisasi ini terbagi menjadi tiga bagian, yaitu sentimen positif, negatif, dan netral. Setiap *word cloud* menampilkan 35 kata yang paling sering muncul dalam masing-masing kategori sentimen tersebut.

Merujuk visualisasi sentimen positif pada Gambar 16, terdapat sejumlah kata yang paling sering muncul dalam kategori sentimen positif, termasuk di dalamnya “pajak” sejumlah 20 kali, “udah” sejumlah 9 kali, dan “lapor” sejumlah 8 kali. Penggunaan kata “pajak”, “udah”, dan “lapor” merujuk pada kelebihan dari Coretax yang dirasakan oleh masyarakat. Hal ini diperkuat dengan temuan sejumlah kata seperti “mudah”, “fitur”, dan “berhasil” yang pada umumnya muncul dalam konteks informasi atau pernyataan bahwa pelatihan serta sosialisasi penggunaan Coretax telah dilaksanakan oleh Kementerian Keuangan.

Gambar 16. *Wordcloud* Sentimen Positif

Meskipun demikian, terdapat pula beberapa *tweet* yang menyebutkan bahwa Coretax mudah digunakan, meskipun jumlahnya sangat terbatas. Selain itu, jumlah *tweet* dengan sentimen positif secara keseluruhan masih relatif sedikit dibandingkan dengan sentimen netral maupun negatif, yang mengindikasikan masih adanya sejumlah aspek yang perlu diperbaiki. Kondisi ini menunjukkan bahwa meskipun Coretax telah memberikan pengalaman positif bagi sebagian pengguna, optimalisasi fitur dan peningkatan kualitas layanan tetap diperlukan untuk memperluas penerimaan publik.

Pada visualisasi sentimen negatif yang ditampilkan dalam Gambar 17, beberapa kata yang paling sering muncul antara lain “pajak” sejumlah 612 kali, “bikin” sejumlah 375 kali, “error” sejumlah 243 kali, Kemunculan kata-kata tersebut mencerminkan berbagai kekurangan Coretax yang dirasakan oleh masyarakat. Hal ini turut memicu ungkapan frustrasi hingga makian terhadap sistem maupun pemerintah. Selain itu, kata-kata seperti “udah”, “error”, “gabisa”, “bener” (dalam konteks sistem yang dianggap tidak benar), serta “gimana” (menandakan kebingungan pengguna terhadap error yang terjadi), semakin menegaskan pengalaman negatif yang dialami. Keluhan mengenai “error” sendiri kerap berkaitan dengan server, yaitu kegagalan membuat faktur billing, kendala mengunduh PDF, hingga data yang diunggah tidak terbaca. Berbagai kata tersebut hanya sebagian kecil dari beragam keluhan yang dilontarkan, yang secara keseluruhan memperlihatkan ketidakpuasan publik terhadap kinerja Coretax yang dinilai belum optimal.

Gambar 17. *Wordcloud* Sentimen Negatif

Lebih jauh, persepsi negatif ini berpotensi menimbulkan dampak nyata terhadap keberhasilan implementasi Coretax. Tingginya tingkat frustrasi pengguna dapat memengaruhi kepatuhan pajak, misalnya dengan mendorong wajib pajak menunda pelaporan hingga sistem dirasa lebih stabil. Selain itu, citra Coretax sebagai sistem yang rumit dan penuh kendala dapat menghambat tingkat adopsi, terutama di kalangan wajib pajak yang sebelumnya sudah terbiasa dengan DJP Online. Apabila hal ini tidak segera diatasi, tujuan utama digitalisasi perpajakan untuk meningkatkan efisiensi, transparansi, dan kepatuhan justru bisa terhambat oleh rendahnya kepercayaan publik terhadap sistem. Oleh karena itu, diperlukan langkah cepat dan terukur dari pemerintah, baik dalam aspek teknis maupun dukungan layanan, agar Coretax benar-benar dapat menjadi instrumen yang efektif dalam modernisasi perpajakan.

Pada visualisasi sentimen netral yang ditampilkan dalam Gambar 18, beberapa kata yang paling sering muncul antara lain “pajak” sejumlah 1329 kali, “silakan” sejumlah 1084 kali, dan “kakak” sejumlah 829 kali. Kemunculan kata-kata tersebut mencerminkan adanya responsivitas dari Kementerian Keuangan dalam menanggapi kebingungan masyarakat terkait sistem Coretax. Respons yang diberikan umumnya berisi arahan mengenai perubahan prosedur layanan, penjelasan langkah-langkah yang perlu dilakukan, serta rujukan menuju laman atau panduan resmi.

Gambar 18. *Wordcloud* Sentimen Netral

Namun, pola respons tersebut juga menunjukkan batasan tertentu. Pertanyaan yang bersifat prosedural seperti tata cara pengajuan, lokasi fitur yang berpindah, atau alur penggunaan sistem cenderung dapat dijawab secara langsung melalui balasan informatif. Sebaliknya, untuk pertanyaan yang menyangkut aspek teknis seperti gangguan sistem, kesalahan akses, atau error pada laman, akun resmi umumnya tidak memberikan jawaban rinci karena penanganannya berada di bawah kewenangan tim teknologi informasi khusus. Karena sifat respons yang lebih berfokus pada pemberian informasi, arahan, dan klarifikasi prosedur tanpa ekspresi emosional, konten semacam ini tidak dapat dikategorikan sebagai sentimen positif maupun negatif.

Berdasarkan hasil analisis *word cloud* pada sentimen positif, negatif, dan netral yang telah dipaparkan sebelumnya, sejumlah rekomendasi dapat diberikan kepada Direktorat Jenderal Pajak (DJP) Kementerian Keuangan untuk mengoptimalkan kinerja Coretax. Langkah-langkah strategis berikut penting dilakukan guna meningkatkan kepercayaan publik dan efektivitas sistem:

1) Meningkatkan kapasitas infrastruktur agar sistem lebih stabil dan responsif, terutama pada periode pelaporan dengan trafik tinggi.
2) Mengembangkan fitur Coretax berdasarkan pendekatan *user-centered design* agar lebih sesuai dengan kebutuhan dan alur kerja pengguna.
3) Memperkuat layanan bantuan teknis melalui kanal komunikasi interaktif seperti live chat atau pusat bantuan terpadu untuk merespons keluhan secara cepat.
4) Melaksanakan pelatihan dan sosialisasi yang lebih masif dan berkelanjutan guna mempercepat adaptasi wajib pajak terhadap sistem baru.
5) Melakukan evaluasi dan pembaruan sistem secara berkala berdasarkan masukan pengguna untuk memastikan perbaikan berkelanjutan.

IV. KESIMPULAN

Penelitian ini menganalisis sentimen masyarakat terhadap Coretax, sistem perpajakan digital yang diluncurkan oleh Kementerian Keuangan pada Januari 2025, dengan menggunakan 6.036 *tweet* sebagai sumber data. Hasil analisis awal menunjukkan bahwa mayoritas sentimen bersifat negatif sebesar 51,05%, diikuti oleh sentimen netral sebesar 48,11%, dan sentimen positif hanya 0,83%. Kondisi ini mencerminkan bahwa implementasi Coretax masih menimbulkan banyak ketidakpuasan di kalangan pengguna.

Dalam hal performa algoritma, penerapan SMOTE terbukti mampu meningkatkan akurasi model secara signifikan. *Naive Bayes Classifier* (NBC) mengalami peningkatan akurasi dari 85,15% menjadi 89,69%. Sementara itu, algoritma *Support Vector Machine* (SVM) menunjukkan performa yang konsisten lebih tinggi dengan berbagai kernel. SVM linear meningkat dari 84,54% menjadi 90,37%, SVM RBF dari 85,15% menjadi 90,70%, SVM polinomial dari 84,93% menjadi 89,65%, dan SVM sigmoid dari 84,54% menjadi 89,44%. LSTM juga mengalami peningkatan, meskipun masih lebih rendah dibandingkan model klasik, dengan akurasi naik dari 51,02% menjadi

60,74%. Dari seluruh pengujian, SVM dengan kernel RBF mencapai hasil terbaik dengan akurasi 90,70%, presisi 91%, recall 90,66%, dan F1-score 90,66%, sehingga dapat disimpulkan sebagai algoritma paling efektif dalam klasifikasi sentimen Coretax.

Lebih lanjut, analisis kata kunci melalui *word cloud* memperlihatkan bahwa kata "pajak", "udah", dan "lapor" mendominasi sentimen positif, sementara "error", "bikin", dan "gabisa" banyak muncul pada sentimen negatif, serta "silakan" dan "kakak" menonjol dalam sentimen netral. Temuan ini menegaskan bahwa masalah teknis, khususnya error yang berkaitan dengan server seperti kegagalan membuat billing, mengunduh PDF, atau membaca data unggahan, menjadi sumber utama ketidakpuasan publik. Di sisi lain, responsivitas Kementerian Keuangan yang ditunjukkan melalui jawaban prosedural di platform X menunjukkan adanya upaya memberikan bantuan, meskipun tidak menyentuh aspek teknis yang paling banyak dikeluhkan pengguna.

Secara keseluruhan, penelitian ini menekankan pentingnya peningkatan kapasitas infrastruktur, pengembangan fitur berbasis kebutuhan pengguna, perbaikan layanan bantuan teknis, serta evaluasi berkelanjutan agar Coretax dapat berfungsi optimal. Dengan langkah-langkah strategis tersebut, Coretax berpotensi tidak hanya meningkatkan efisiensi administrasi perpajakan, tetapi juga memulihkan kepercayaan publik terhadap sistem digital perpajakan di Indonesia.

Meski begitu, penelitian ini memiliki keterbatasan, terutama karena data yang digunakan hanya bersumber dari X sehingga belum sepenuhnya merepresentasikan persepsi masyarakat luas, serta keterbatasan pada pemodelan algoritma yang masih berfokus pada NBC, SVM, dan LSTM. Oleh karena itu, penelitian selanjutnya disarankan untuk memperluas sumber data melalui forum diskusi, portal berita, atau survei langsung, serta mengeksplorasi model berbasis transformer seperti BERT atau RoBERTa untuk meningkatkan performa klasifikasi. Analisis temporal juga penting dilakukan, tidak hanya pada periode pelaporan SPT tahunan (Januari–April) yang telah menjadi fokus penelitian ini, tetapi juga pada bulan-bulan di luar periode tersebut guna melihat dinamika persepsi publik secara lebih menyeluruh. Selain itu, penelitian lanjutan dapat mengkaji faktor demografis pengguna untuk memahami kelompok masyarakat yang paling terdampak, serta menerapkan pendekatan *aspect-based sentiment analysis* (ABSA) guna mengidentifikasi aspek spesifik, seperti login, pembuatan billing, dan unggah dokumen, yang paling sering menimbulkan keluhan.

DAFTAR PUSTAKA

[1] S. A. Ilanoputri, "Pelayanan Yang Diterima Oleh Masyarakat Sebagai Pembayar Pajak Berdasarkan Penerapan Beban Pajak Daerah Yang Diatur Dalam Undang-Undang Pajak Dan Retribusi Daerah," *Cepalo*, vol. 4, no. 2, pp. 143–156, 2020, doi: 10.25041/cepalo.v4no2.2067.

[2] F. Aqmarina and I. K. Furqon, "Peran Pajak sebagai Instrumen Kebijakan Fiskal dalam Mengantisipasi Krisis Ekonomi pada Masa Pandemi Covid-19," *Finans. J. Akunt. dan Perbank. Syariah*, vol. 3, no. 2, pp. 255–274, 2020, doi: 10.32332/finansia.v3i2.2507.

[3] "Coretax: Sistem Canggih Tingkatkan Kepatuhan Sukarela | Direktorat Jenderal Pajak." Accessed: Feb. 22, 2025. [Online]. Available: https://pajak.go.id/id/artikel/coretax-sistem-canggih-tingkatkan-kepatuhan-sukarela

[4] "Panduan Core Tax System dan Cara Kerjanya." Accessed: Feb. 26, 2025. [Online]. Available: https://klikpajak.id/blog/core-tax-administration-system/

[5] Juniarti, L. Noersanti, A. Akhmadi, P. A. Ardheta, and S. N. Auzaini, "Digitalisasi Perpajakan: Tantangan, Peluang, dan Dampaknya terhadap Kepercayaan Publik serta Kewajiban Pajak di Tokopedia," *J. Akunt. STEI*, vol. 11, no. 1, pp. 1–12, 2025, [Online]. Available: https://doi.org/10.36406/jasstei.v11i1.37

[6] A. K. jama, H. N. Priyatna, and A. S. Tampunolon, "Dampak Perkembangan Aplikasi Dan Kebijakan Perpajakan Terhadap Kepercayaan Publik," *J. Ris. Ilmu Ekon.*, vol. 1, no. 1, pp. 39–49, 2025.

[7] Lady Agustine Fitrana, S. Linawati, N. Herlinawati, R. Sa'adah, and S. Seimahuria, "Analisis Sentimen Pengguna Twitter Terhadap Brand Indosat Menggunakan Metode Naïve Bayes Classifier," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 8, no. 3, pp. 4291–4297, 2024, doi: 10.36040/jati.v8i3.9866.

[8] Y. Mao, Q. Liu, and Y. Zhang, "Sentiment analysis methods, applications, and challenges: A systematic literature review," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 36, no. 4, p. 102048, 2024, doi: 10.1016/j.jksuci.2024.102048.

[9] Z. Drus and H. Khalid, "Sentiment analysis in social media and its application: Systematic literature review," *Procedia Comput. Sci.*, vol. 161, pp. 707–714, 2019, doi: 10.1016/j.procs.2019.11.174.

[10] M. R. Fahlevvi, "Analisis Sentimen Terhadap Ulasan Aplikasi Pejabat Pengelola Informasi Dan Dokumentasi Kementerian Dalam Negeri Republik Indonesia Di Google Playstore Menggunakan Metode Support Vector Machine," *J. Teknol. dan Komun. Pemerintah.*, vol. 4, no. 1, pp. 1–13, 2022, doi: 10.33701/jtkp.v4i1.2701.

[11] J. Homepage, D. Pramudita, Y. Akbar, and T. Wahyudi, "Sentiment Analysis of the Indonesian Smart College Card Program on Social Media X Using the Naive Bayes Algorithm Analisis Sentimen Terhadap Program Kartu Indonesia Pintar Kuliah Pada Media Sosial X Menggunakan Algoritma Naive Bayes," *Malcom*, vol. 4, no. October, pp. 1420–1430, 2024.

[12] A. Pebdika, R. Herdiana, and D. Solihudin, "Klasifikasi Menggunakan Metode Naive Bayes Untuk Menentukan Calon Penerima Pip," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 7, no. 1, pp. 452–458, 2023, doi: 10.36040/jati.v7i1.6303.

[13] A. C. Najib, A. Irsyad, G. A. Qandi, and N. A. Rakhmawati, "Perbandingan Metode Lexicon-based dan SVM untuk Analisis Sentimen Berbasis Ontologi pada Kampanye Pilpres Indonesia Tahun 2019 di Twitter," *Fountain Informatics J.*, vol. 4, no. 2, pp. 41–48, 2019, doi: 10.21111/fij.v4i2.3573.

[14] R. Liu, Y. Jiang, and J. Lin, "Forecasting the Volatility of Specific Risk for Stocks with LSTM," *Procedia Comput. Sci.*, vol. 202, pp. 111–114, 2022, doi: 10.1016/j.procs.2022.04.015.

[15] E. P. Wijaya and M. H. Rifqo, "Application of Naive Bayes Algorithm in Analyzing Public Sentiment towards Coretax on Platform X," *J. Artif. Intell. Softw. Eng.*, vol. 5, no. 2, pp. 564–572, 2025, doi: 10.30811/jaise.v5i2.6949.

[16] F. Fathoni, A. Faradhisa Ansori, I. Nailah Ramadhani, C. Rahmi Anissa, and S. Amelia Putri, "Analisis Sentimen Masyarakat Indonesia Di Twitter Terhadap Sistem Perpajakan 'Coretax' Menggunakan Metode Naïve Bayes," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 9, no. 4, pp. 6749–6753, 2025, doi: 10.36040/jati.v9i4.14214.

[17] A. Ramadhani, I. Permana, M. Afdal, and M. Fronita, "Analisis Sentimen Tanggapan Publik di Twitter Terkait Program Kerja

Makan Siang Gratis Prabowo–Gibran Menggunakan Algoritma Naïve Bayes Classifier dan Support Vector Machine," *Build. Informatics, Technol. Sci.*, vol. 6, no. 3, p. 1509−1516, 2024, [Online]. Available: https://ejurnal.seminar-id.com/index.php/bits/article/view/6188/3222

[18] D. N. Novianti, D. F. Shiddieq, F. F. Roji, and W. Susilawati, "Comparison of Support Vector Machine and Naïve Bayes Algorithms for Sentiment Analysis of the Metaverse," *MALCOM Indones. J. Mach. Learn. Comput. Sci.*, vol. 4, no. April, pp. 231–239, 2024.

[19] S. Rabbani, D. Safitri, N. Rahmadhani, A. A. F. Sani, and M. K. Anam, "Perbandingan Evaluasi Kernel SVM untuk Klasifikasi Sentimen dalam Analisis Kenaikan Harga BBM," *MALCOM Indones. J. Mach. Learn. Comput. Sci.*, vol. 3, no. 2, pp. 153–160, 2023, doi: 10.57152/malcom.v3i2.897.

[20] J. S. Gea, H. Budiati, and S. S. B. Kristian Juri Damai Lase, "Analisis Sentimen Masyarakat Terhadap Direktorat Jenderal Pajak," *J. InFact Sains Dan Komput.*, vol. 8, no. 01, pp. 30–36, 2024, doi: 10.61179/jurnalinfact.v8i01.466.

[21] P. Aditiya, U. Enri, and I. Maulana, "Analisis Sentimen Ulasan Pengguna Aplikasi Myim3 Pada Situs Google Play Menggunakan Support Vector Machine," *JURIKOM (Jurnal Ris. Komputer)*, vol. 9, no. 4, pp. 1020–1028, 2022, doi: 10.30865/jurikom.v9i4.4673.

[22] L. Hickman, S. Thapa, L. Tay, M. Cao, and P. Srinivasan, "Text Preprocessing for Text Mining in Organizational Research: Review and Recommendations," *Organ. Res. Methods*, vol. 25, no. 1, pp. 114–146, 2022, doi: 10.1177/1094428120971683.

[23] E. Alshdaifat, D. Alshdaifat, A. Alsarhan, F. Hussein, and S. Moh, "The Effect of Preprocessing Techniques, Applied to Numeric Features, on Classification Algorithms' Performance," *Data*, vol. 6, no. 11, 2021.

[24] D. Septiani and I. Isabela, "Analisis Term Frequency Inverse Document Frequency (TF-IDF) Dalam Temu Kembali Informasi Pada Dokumen Teks," *SINTESIA J. Sist. dan Teknol. Inf. Indones.*, vol. 1, no. 2, pp. 81–88, 2023.

[25] E. B. Fatima, O. Boutkhoum, E. M. Abdelmajid, F. Rustam, A. Mehmood, and G. S. Choi, "Minimizing the Overlapping *Degree* to Improve Class-Imbalanced Learning under Sparse Feature Selection: Application to Fraud Detection," *IEEE Access*, vol. 9, pp. 28101–28110, 2021, doi: 10.1109/ACCESS.2021.3056285.

[26] R. W. Pratiwi, S. F. H, D. Dairoh, D. I. Af'idah, Q. R. A, and A. G. F, "Analisis Sentimen Pada Review Skincare Female Daily Menggunakan Metode Support Vector Machine (SVM)," *J. Informatics, Inf. Syst. Softw. Eng. Appl.*, vol. 4, no. 1, pp. 40–46, 2021, doi: 10.20895/inista.v4i1.387.

[27] R. Rachman and R. N. Handayani, "Klasifikasi Algoritma Naive Bayes Dalam Memprediksi Tingkat Kelancaran Pembayaran Sewa Teras UMKM," *J. Inform.*, vol. 8, no. 2, pp. 111–122, 2021, doi: 10.31294/ji.v8i2.10494.

[28] T. M. Permata Aulia, N. Arifin, and R. Mayasari, "Perbandingan Kernel Support Vector Machine (Svm) Dalam Penerapan Analisis Sentimen Vaksinisasi Covid-19," *SINTECH (Science Inf. Technol. J.*, vol. 4, no. 2, pp. 139–145, 2021, doi: 10.31598/sintechjournal.v4i2.762.

[29] L. Rohmatun and A. Baita, "Machine Learning-Based Sentiment Analysis on Twitter (X): A Case Study of the ' Kabur Aja Dulu ' Issue Using SVM," *J. Appl. Informatics Comput.*, vol. 9, no. 4, pp. 1972–1983, 2025.

[30] G. Tamami, W. A. Triyanto, and S. Muzid, "Sentiment Analysis Mobile JKN Reviews Using SMOTE Based LSTM," *IJCCS (Indonesian J. Comput. Cybern. Syst.*, vol. 19, no. 1, pp. 13–24, 2025, doi: 10.22146/ijccs.101910.

[31] C. M. Putri, M. Afdal, R. Novita, and M. Mustakim, "Perbandingan Evaluasi Kernel Support Vector Machine dalam Analisis Sentimen Chatbot AI pada Ulasan Google Play Store," *J. Teknol. Sist. Inf. dan Apl.*, vol. 7, no. 3, pp. 1236–1245, 2024, doi: 10.32493/jtsi.v7i3.41354.