

Analisis Efektivitas Algoritma Machine Learning dalam Deteksi Hoaks: Pada Berita Digital Berbahasa Indonesia

M Dicky Desriansyah^a, Intan Utma Sari^b, Zulfahmi^c

^aSistem Informasi, Fakultas Farmasi, Sains dan Teknologi, Universitas Dharma Andalas, mdickyd16@gmail.com

^bSistem Informasi, Fakultas Farmasi, Sains dan Teknologi, Universitas Dharma Andalas, intanutna88@gmail.com

^cSistem Informasi, Fakultas Farmasi, Sains dan Teknologi, Universitas Dharma Andalas, zulfahmi1@gmail.com

Abstract

The rapid development of information technology has transformed how society accesses and disseminates information. Unfortunately, this phenomenon also creates opportunities for the massive spread of fake news or hoaxes through digital platforms. This research aims to analyze the effectiveness of several machine learning algorithms in detecting text-based hoaxes in Indonesian. The algorithms tested include Multilayer Perceptron (MLP), Naïve Bayes (NB), Support Vector Machine (SVM), and Random Forest (RF). The data used consists of online news articles that have undergone text preprocessing stages such as tokenizing, case folding, filtering, stopword removal, stemming, and weighting using the TF-IDF method with a combination of unigram and bigram features. Performance evaluation was conducted using precision, recall, F1-score, and accuracy metrics. The results show that the SVM and MLP algorithms yielded the highest performance with evaluation values above 99.8%, while RF demonstrated strong and stable performance, and NB showed decent performance with high efficiency. These findings provide insights into the effectiveness of text classification methods in hoax detection and serve as a reference for developing more efficient and accurate fake news detection systems.

Keywords: Hoax detection, machine learning, text classification, TF-IDF, NLP, fake news.

Abstrak

Perkembangan teknologi informasi yang pesat telah mengubah cara masyarakat mengakses dan menyebarkan informasi. Sayangnya, fenomena ini juga membuka peluang bagi penyebaran berita palsu atau hoaks secara masif melalui platform digital. Penelitian ini bertujuan untuk menganalisis efektivitas beberapa algoritma machine learning dalam mendeteksi berita hoaks berbahasa Indonesia berbasis teks. Algoritma yang diuji meliputi Multilayer Perceptron (MLP), Naïve Bayes (NB), Support Vector Machine (SVM), dan Random Forest (RF). Data yang digunakan berupa artikel berita online yang telah melalui tahapan preprocessing teks seperti tokenizing, case folding, filtering, stopword removal, stemming, serta pembobotan menggunakan metode TF-IDF dengan kombinasi fitur unigram dan bigram. Evaluasi kinerja dilakukan dengan menggunakan metrik presisi, recall, skor F1, dan akurasi. Hasil menunjukkan bahwa algoritma SVM dan MLP memberikan performa tertinggi dengan nilai evaluasi di atas 99,8%, sementara RF menunjukkan performa kuat dan stabil, dan NB menunjukkan performa yang layak dengan efisiensi tinggi. Temuan ini memberikan wawasan mengenai efektivitas metode klasifikasi teks dalam mendeteksi hoaks dan menjadi acuan bagi pengembangan sistem deteksi berita hoaks yang lebih efisien dan akurat.

Kata Kunci: Deteksi hoaks, machine learning, klasifikasi teks, TF-IDF, NLP, berita palsu.

This work is licensed under Creative Commons Attribution License 4.0 CC-BY International license



1. PENDAHULUAN

Penyebaran berita palsu (hoaks) telah menjadi salah satu tantangan serius di era digitalisasi media yang semakin masif. Kemudahan dalam memproduksi dan menyebarkan informasi melalui berbagai platform digital telah disalahgunakan oleh pihak tidak bertanggung jawab untuk menyebarkan konten yang tidak dapat diverifikasi kebenarannya. Fenomena ini tidak hanya menimbulkan keresahan sosial dan erosi kepercayaan terhadap institusi resmi, tetapi juga berpotensi memicu konflik dan perpecahan di masyarakat. Oleh karena itu, pengembangan sistem deteksi berita hoaks yang efektif, efisien, dan akurat menjadi krusial untuk menjaga integritas informasi dan stabilitas sosial.

Dalam ranah ilmu komputer, deteksi hoaks berbasis teks merupakan disiplin ilmu yang erat kaitannya dengan klasifikasi teks (text classification) dan Pemrosesan Bahasa Alami (Natural Language Processing/NLP). Berbagai pendekatan machine learning telah diterapkan untuk mengidentifikasi hoaks, meliputi metode probabilistik seperti Naïve Bayes, algoritma margin-based seperti Support Vector Machine, hingga pendekatan ensemble learning dan deep learning. Banyak penelitian sebelumnya telah mengeksplorasi berbagai algoritma untuk tujuan ini (Kurniawan & Mustikasari, 2020; Hanum et al., 2024).

Penelitian ini bertujuan untuk melakukan analisis komparatif terhadap empat algoritma machine learning yang populer dan efektif: *Multilayer Perceptron (MLP)*, *Naïve Bayes (NB)*, *Support Vector Machine (SVM)*, dan *Random Forest (RF)*. Kami juga menekankan pentingnya tahapan preprocessing teks yang komprehensif dan representasi fitur yang optimal dengan metode Term Frequency-Inverse Document Frequency (TF-IDF), yang

telah terbukti meningkatkan akurasi klasifikasi dalam berbagai studi (Weng et al., 2020; Al-Maqaleh et al., 2023).

2. METODOLOGI

2.1 Dataset dan Preprocessing

Dataset yang digunakan dalam penelitian ini terdiri dari artikel berita online berbahasa Indonesia yang telah dikategorikan secara manual sebagai hoaks dan non-hoaks. Keseimbangan distribusi antara kedua kategori kelas (hoaks dan non-hoaks) dalam dataset sangat diperhatikan untuk mencegah bias model. Proses preprocessing teks dilakukan secara sistematis untuk mengubah data teks mentah menjadi representasi numerik yang siap diproses oleh algoritma machine learning. Tahapan preprocessing ini meliputi:

1. *Tokenizing* Proses memecah rangkaian karakter teks menjadi unit-unit linguistik yang lebih kecil, seperti kata atau frasa, yang disebut token. Ini merupakan langkah awal krusial untuk analisis lebih lanjut.

2. *Case Folding* Mengubah semua karakter huruf kapital menjadi huruf kecil. Langkah ini memastikan bahwa kata yang sama dengan kapitalisasi berbeda (misalnya, "Berita" dan "berita") diperlakukan sebagai entitas yang sama.

3. *Filtering* Menghapus karakter non-alfabetik, angka, tanda baca, simbol, dan karakter khusus lainnya yang tidak relevan untuk analisis teks. Ini membantu mengurangi "noise" dalam data.

4. *Stopword Removal* Menghapus kata-kata umum yang sering muncul namun memiliki nilai informasi rendah atau tidak signifikan dalam menentukan kategori teks (misalnya, "yang", "dan", "adalah"). Daftar stopwords bahasa Indonesia yang telah terdefinisi digunakan dalam tahap ini.

5. *Stemming* Mengubah kata berimbuhan menjadi bentuk dasar atau akar katanya. Misalnya, "menyebarkan" menjadi "sebar". Ini membantu mengurangi variasi kata dan mengelompokkan kata-kata yang memiliki makna serupa.

6. *TF-IDF (Term Frequency-Inverse Document Frequency)* Setelah preprocessing, teks diubah menjadi representasi numerik menggunakan metode pembobotan TF-IDF. TF (Term Frequency) mengukur seberapa sering sebuah kata muncul dalam sebuah dokumen, sementara IDF (Inverse Document Frequency) mengukur seberapa penting sebuah kata di seluruh korpus dokumen. Kombinasi keduanya memberikan bobot yang tinggi pada kata-kata yang sering muncul dalam dokumen tertentu tetapi jarang di seluruh korpus, menunjukkan relevansinya.

Dalam penelitian ini, TF-IDF digunakan dengan pengaturan kombinasi unigram dan bigram, serta membatasi maksimal 5000 fitur. Penggunaan unigram (kata tunggal) dan bigram (pasangan dua kata berurutan) mampu menangkap konteks linguistik yang lebih kaya, sementara pembatasan fitur membantu mencegah overfitting dan menjaga efisiensi komputasi (Guo et al., 2021).

Rumus TF-IDF:

Rumus TF-IDF untuk suatu term t dalam dokumen d dari korpus D adalah:

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

dengan:

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

$$\text{IDF}(t, D) = \log[(\text{jumlah total dokumen dalam } D) / (\text{jumlah dokumen yang mengandung } t)]$$

Logaritma digunakan untuk meredam skala nilai IDF, dan basis logaritma (biasanya e atau 10) tidak terlalu signifikan asalkan konsisten.

2.2 Algoritma Klasifikasi

Empat algoritma klasifikasi teks yang dipilih untuk penelitian ini memiliki karakteristik dan prinsip kerja yang berbeda, sehingga memungkinkan analisis komparatif yang mendalam:

2.2.1 Multilayer Perceptron (MLP)

Merupakan jenis jaringan saraf tiruan (Artificial Neural Network/ANN) feedforward. Model yang diimplementasikan dalam penelitian ini menggunakan dua lapisan tersembunyi (hidden layers) dengan masing-masing 100 dan 50 neuron, serta fungsi aktivasi ReLU. MLP sangat cocok untuk menangkap hubungan non-linear yang kompleks antar fitur dalam data teks, menjadikannya powerful dalam tugas klasifikasi (Kusuma et al., 2021).

Cara Kerja MLP:

MLP terdiri dari setidaknya tiga lapisan: lapisan input, satu atau lebih lapisan tersembunyi, dan lapisan output. Setiap neuron di satu lapisan terhubung ke setiap neuron di lapisan berikutnya.

1. Propagasi Maju (Forward Propagation):

- Input (x_1, x_2, \dots, x_n) diterima oleh lapisan input
- Untuk setiap neuron di lapisan tersembunyi (dan selanjutnya di lapisan output), input yang diterima dihitung sebagai jumlah berbobot dari output neuron di lapisan sebelumnya, ditambahkan bias:

$$z_j = \sum_i w_{ij} x_i + b_j$$

- Output neuron dihitung dengan menerapkan fungsi aktivasi (misalnya ReLU, Sigmoid, atau Tanh) ke z_j :

$$a_j = f(z_j)$$

- Proses ini berlanjut hingga lapisan output menghasilkan prediksi

2. Propagasi Mundur (Backpropagation):

- Perbedaan antara prediksi output dan nilai aktual (error) dihitung
- Error ini kemudian "dipropagasi mundur" melalui jaringan, dan bobot (w_{ij}) serta bias (b_j) disesuaikan untuk meminimalkan error menggunakan algoritma optimasi seperti Gradient Descent
- Penyesuaian bobot dihitung berdasarkan gradien fungsi kerugian terhadap bobot
- Melalui iterasi proses ini (epoch), MLP belajar untuk mengenali pola dalam data dan mengklasifikasikannya

2.2.2 Naïve Bayes (NB)

Merupakan pendekatan probabilistik yang didasarkan pada Teorema Bayes dengan asumsi kuat independensi antar fitur. Dalam penelitian ini, Multinomial *Naïve Bayes* digunakan, yang seringkali efektif untuk klasifikasi teks karena beroperasi dengan frekuensi kata. Meskipun asumsi independensinya seringkali tidak terpenuhi di dunia nyata, NB sering memberikan hasil yang surprisingly baik dan sangat efisien secara komputasi.

Cara Kerja Naïve Bayes:

Naïve Bayes mengklasifikasikan dokumen berdasarkan probabilitas kemunculan fitur (kata) dalam setiap kelas. Asumsi "naïve" adalah bahwa kemunculan suatu fitur dalam sebuah kelas tidak bergantung pada kemunculan fitur lainnya (independensi fitur).

1. Fase Pelatihan:

- Menghitung probabilitas prior untuk setiap kelas ($P(C_k)$)
- Menghitung probabilitas likelihood untuk setiap fitur (kata) diberikan setiap kelas ($P(x_i|C_k)$). Ini adalah probabilitas kemunculan kata x_i dalam dokumen yang termasuk dalam kelas C_k
-

2. Fase Klasifikasi (Prediksi):

- Untuk dokumen baru D_{test} yang terdiri dari kata-kata (x_1, x_2, \dots, x_n), kelas yang paling mungkin ditentukan menggunakan Teorema Bayes:

$$P(C_k|D_test) \propto P(C_k) \prod_{i=1}^n P(x_i|C_k)$$

- Algoritma memilih kelas C_k yang memaksimalkan $P(C_k|D_test)$

Multinomial Naïve Bayes secara khusus cocok untuk klasifikasi teks karena memperhitungkan frekuensi kata dalam dokumen. Probabilitas $P(x_i|C_k)$ dihitung dengan:

$$P(x_i|C_k) = [\text{count}(x_i, C_k) + \alpha] / [\text{count}(C_k) + \alpha \times V]$$

di mana:

- $\text{count}(x_i, C_k)$ adalah jumlah kemunculan kata x_i dalam semua dokumen di kelas C_k
- $\text{count}(C_k)$ adalah jumlah total kata dalam semua dokumen di kelas C_k
- α adalah parameter smoothing (seringkali 1 untuk Laplace smoothing) untuk menghindari probabilitas nol
- V adalah ukuran kosakata (jumlah kata unik)

2.2.3 Support Vector Machine (SVM)

Algoritma ini bekerja dengan menemukan hyperplane optimal yang memisahkan kelas-kelas data dengan margin terbesar. SVM sangat efektif dalam ruang berdimensi tinggi, seperti yang sering ditemukan pada representasi teks menggunakan TF-IDF. Penelitian ini menggunakan kernel linier, yang terbukti efisien dan efektif untuk data tekstual (Wang et al., 2020).

Cara Kerja SVM:

SVM bertujuan untuk menemukan hyperplane (batas keputusan) di ruang fitur yang secara optimal memisahkan titik-titik data dari kelas yang berbeda dengan margin terbesar.

1. Konsep Hyperplane: Dalam ruang 2D, hyperplane adalah sebuah garis. Dalam ruang 3D, itu adalah sebuah bidang. Dalam ruang berdimensi tinggi (seperti data teks setelah TF-IDF), itu adalah subruang berdimensi $N-1$.
2. Margin: Jarak antara hyperplane dan titik-titik data terdekat dari setiap kelas (disebut support vectors). SVM mencari hyperplane yang memaksimalkan margin ini.
3. Fungsi Keputusan (untuk kernel linier):

$$f(x) = \text{sign}(w \cdot x + b)$$

dimana:

- w adalah vektor bobot normal terhadap hyperplane
 - x adalah vektor fitur dari titik data
 - b adalah bias (intercept)
 - $\text{sign}()$ adalah fungsi tanda yang mengembalikan +1 atau -1, sesuai dengan kelas
4. Optimasi: SVM menyelesaikan masalah optimasi untuk menemukan w dan b yang memaksimalkan margin, seringkali diformulasikan sebagai masalah optimasi kuadratik. Titik-titik data yang paling dekat dengan hyperplane adalah "support vectors" dan mereka memainkan peran kunci dalam mendefinisikan hyperplane. Untuk kasus non-linier, SVM menggunakan fungsi kernel untuk memetakan data ke ruang berdimensi lebih tinggi di mana pemisahan linier mungkin terjadi.

2.2.4 Random Forest (RF)

Merupakan metode ensemble learning yang membangun banyak pohon keputusan (decision trees) dan menggabungkan hasilnya untuk membuat prediksi akhir. Dalam penelitian ini, 100 decision trees digunakan. RF efektif dalam mengurangi masalah overfitting yang sering terjadi pada pohon keputusan tunggal dan memberikan klasifikasi yang stabil dan robust (Prakoso et al., 2022).

Cara Kerja Random Forest:

Random Forest membangun sekumpulan (forest) pohon keputusan secara independen selama fase pelatihan, dan kemudian menggabungkan prediksi mereka untuk menghasilkan prediksi yang lebih akurat dan stabil.

1. Bootstrap Aggregating (Bagging): RF menggunakan metode bagging untuk membuat setiap pohon. Ini berarti setiap pohon dilatih pada subset data pelatihan yang diambil secara acak dengan penggantian (bootstrap sample). Ini memastikan variasi antar pohon.

2. Fitur Acak (Random Feature Subsetting): Selain sampling data, pada setiap node saat membangun pohon, *RF* hanya mempertimbangkan subset acak dari total fitur yang tersedia untuk menentukan split terbaik. Ini further mengurangi korelasi antar pohon dan meningkatkan keragaman.

3. Proses Klasifikasi:

- Setiap pohon dalam hutan membuat prediksinya sendiri untuk instance data baru
- Untuk tugas klasifikasi, Random Forest melakukan "voting mayoritas" di antara semua pohon. Kelas yang paling banyak diprediksi oleh pohon-pohon individual menjadi prediksi akhir

Rumus matematis RF lebih kompleks karena melibatkan agregasi banyak pohon keputusan, tetapi secara konseptual, output klasifikasi $C_{RF}(x)$ untuk input x adalah:

$$C_{RF}(x) = \text{mode}\{C_1(x), C_2(x), \dots, C_K(x)\}$$

di mana $C_k(x)$ adalah klasifikasi dari pohon keputusan ke- k , dan K adalah jumlah pohon dalam hutan.

2.3 Evaluasi Model

Dataset dibagi secara proporsional menjadi 80% data pelatihan (*training data*) dan 20% data pengujian (*testing data*). Pembagian ini dilakukan dengan stratifikasi label untuk memastikan bahwa distribusi kelas (hoaks dan non-hoaks) dalam data pelatihan dan pengujian tetap seimbang, merepresentasikan populasi dataset secara akurat. Kinerja model dievaluasi menggunakan empat metrik standar dalam klasifikasi:

2.3.1 Akurasi (Accuracy)

Mengukur rasio jumlah prediksi yang benar (baik hoaks maupun non-hoaks) terhadap total jumlah prediksi. Ini memberikan gambaran umum tentang kinerja model.

$$\text{Akurasi} = (TP + TN) / (TP + TN + FP + FN)$$

2.3.2 Presisi (Precision)

Mengukur proporsi prediksi positif yang benar (true positives) dari seluruh prediksi yang diklasifikasikan sebagai positif. Ini penting untuk meminimalkan false positives (berita non-hoaks yang keliru diklasifikasikan sebagai hoaks).

$$\text{Presisi} = TP / (TP + FP)$$

2.3.3 Recall (Sensitivitas)

Mengukur proporsi prediksi positif yang benar (true positives) dari seluruh data aktual yang positif. Ini penting untuk meminimalkan false negatives (berita hoaks yang keliru diklasifikasikan sebagai non-hoaks).

$$\text{Recall} = TP / (TP + FN)$$

2.3.4 F1-score

Merupakan rata-rata harmonis dari Presisi dan Recall. F1-score sangat berguna ketika ada ketidakseimbangan kelas dalam dataset atau ketika kita ingin menyeimbangkan antara meminimalkan false positives dan false negatives. F1-score memberikan ukuran tunggal yang mencerminkan keseimbangan antara Presisi dan Recall.

$$\text{F1-score} = 2 \times (\text{Presisi} \times \text{Recall}) / (\text{Presisi} + \text{Recall})$$

Keterangan:

- *TP (True Positive)*: Jumlah instance positif yang diklasifikasikan dengan benar sebagai positif
- *TN (True Negative)*: Jumlah instance negatif yang diklasifikasikan dengan benar sebagai negatif
- *FP (False Positive)*: Jumlah instance negatif yang salah diklasifikasikan sebagai positif (Kesalahan Tipe I)
- *FN (False Negative)*: Jumlah instance positif yang salah diklasifikasikan sebagai negatif (Kesalahan Tipe II)

3. HASIL DAN PEMBAHASAN

Hasil pengujian komparatif menunjukkan bahwa semua algoritma yang diuji mampu memberikan performa yang tinggi dalam mendeteksi hoaks berbahasa Indonesia, namun terdapat perbedaan signifikan dalam tingkat akurasi dan stabilitasnya:

Algoritma	Presisi	Recall	Skor F1	Akurasi
Multilayer Perceptron	99,88%	99,88%	99,88%	99,88%
Naïve Bayes	97,68%	97,68%	97,68%	97,68%
Support Vector Machine	99,91 %	99,91 %	99,91 %	99,91 %
Random Forest	99,71%	99,70%	99,70%	99,70%

Tabel 1.1 Perbandingan Kinerja Algoritma dalam Pengklasifikasian Berita Hoax <http://bit.ly/4lx7wR5>

3.1 Support Vector Machine (SVM)

Support Vector Machine (SVM) menunjukkan kinerja superior, mencapai nilai presisi, recall, F1-score, dan akurasi sebesar **99,91%**. Keberhasilan luar biasa *SVM* ini dapat diatribusikan pada kemampuannya yang sangat baik dalam menangani data berdimensi tinggi, seperti representasi fitur teks TF-IDF, serta kemampuannya menemukan hyperplane pemisah yang optimal meskipun kompleksitas fitur teks tinggi. Temuan ini konsisten dengan studi lain yang menyoroti efektivitas *SVM* dalam klasifikasi teks (Wang et al., 2020).

3.2 Multilayer Perceptron (MLP)

Multilayer Perceptron (MLP) juga memperoleh hasil yang sangat baik di semua metrik, dengan nilai rata-rata **99,88%**. Keunggulan *MLP* terletak pada arsitektur jaringan sarafnya yang memungkinkan untuk menangkap pola non-linear dan interaksi kompleks antar fitur dalam data teks. Kemampuannya untuk belajar representasi data yang lebih abstrak seringkali memberikan performa tinggi pada dataset yang besar dan kompleks (Kusuma et al., 2021).

3.3 Random Forest (RF)

Random Forest (RF) menunjukkan performa yang kuat dan stabil dengan nilai rata-rata **99,70%**. Sebagai metode ensemble, *RF* efektif dalam mengurangi masalah overfitting dan varians, menghasilkan klasifikasi yang robust. Kemampuannya menggabungkan kekuatan dari banyak pohon keputusan menjadikannya pilihan yang handal untuk tugas klasifikasi teks.

3.4 Naïve Bayes (NB)

Naïve Bayes (NB) mencatatkan skor rata-rata **97,68%** pada semua metrik. Meskipun kinerjanya sedikit lebih rendah dibandingkan algoritma lain dalam studi ini, *NB* tetap unggul dalam efisiensi komputasi dan kesederhanaan implementasinya. Algoritma ini sering digunakan sebagai *baseline* karena performanya yang layak dengan biaya komputasi yang rendah, menjadikannya pilihan yang baik untuk sistem real-time dengan sumber daya terbatas.

3.5 Analisis Faktor Keberhasilan

Keberhasilan klasifikasi yang tinggi ini sangat dipengaruhi oleh tahapan preprocessing teks yang cermat dan pemilihan representasi fitur yang optimal. Penggunaan TF-IDF dengan kombinasi unigram dan bigram secara signifikan meningkatkan kemampuan model dalam mengenali pola linguistik dan semantik dalam data teks. Kombinasi ini memungkinkan model untuk menangkap tidak hanya keberadaan kata kunci tetapi juga konteks dan hubungan antar kata. Hasil ini sejalan dengan studi sebelumnya yang menunjukkan efektivitas metode TF-IDF dalam representasi kata untuk berbagai tugas NLP (Weng et al., 2020; Guo et al., 2021).

4. KESIMPULAN

Penelitian ini secara komprehensif mengevaluasi kinerja beberapa algoritma machine learning dalam klasifikasi berita hoaks berbasis teks berbahasa Indonesia. Hasil penelitian secara jelas menunjukkan bahwa algoritma *Support Vector Machine (SVM)* dan *Multilayer Perceptron (MLP)* adalah metode yang paling efektif, mampu menghasilkan nilai evaluasi di atas 99,8% pada metrik presisi, recall, F1-score, dan akurasi. Ini menggarisbawahi kapabilitas mereka dalam menangani kompleksitas data tekstual.

Random Forest (RF) juga memberikan hasil yang sangat baik dan stabil, menunjukkan ketahanan terhadap overfitting. Meskipun sedikit di bawah yang lain, *Naïve Bayes (NB)* tetap relevan sebagai *baseline* yang efisien dan akurat untuk kasus-kasus tertentu.

Temuan kunci lainnya adalah kontribusi signifikan dari teknik preprocessing teks yang teliti, terutama penggunaan TF-IDF dengan kombinasi unigram dan bigram, yang terbukti secara substansial meningkatkan kemampuan model dalam mengekstraksi fitur relevan dan mengenali pola dalam teks.

DAFTAR PUSTAKA

- [1] Asiyah, S. N. (n.d.). Klasifikasi Berita Online Menggunakan SVM dan KNN.
- [2] Kurniawan, A. A., & Mustikasari, M. (2020). Implementasi Deep Learning untuk Menentukan Berita Palsu dalam Bahasa Indonesia.
- [3] Hanum, A. R., et al. (2024). Analisis Kinerja Algoritma Klasifikasi Teks dalam Mendeteksi Berita Hoaks. JTHK
- [4] Weng, C., Wu, H., Wang, J., & Zhang, Y. (2020). Research on Text Classification Based on TF-IDF and SVM. *Journal of Physics: Conference Series*, 1635(1), 012002.
- [5] Wang, J., Yang, Y., Li, Q., & Zhang, G. (2020). Research on text classification based on improved TF-IDF and SVM. *Journal of Physics: Conference Series*, 1676(1), 012019.
- [6] Guo, M., Lin, J., Zheng, C., & Xie, Y. (2021). Chinese Text Classification Based on TF-IDF and Word2Vec. *Journal of Physics: Conference Series*, 1948(1), 012028.
- [7] Kusuma, M. A., Ardiansyah, B., & Setiawan, I. (2021). Deep Learning for Hoax Detection on Indonesian News Article: A Comparative Study of Convolutional Neural Networks and Long Short-Term Memory. *IOP Conference Series: Materials Science and Engineering*, 1087(1), 012053.
- [8] Al-Maqaleh, H., Al-Maqaleh, T., Al-Maqaleh, M. A., & Al-Maqaleh, W. A. (2023). Fake News Detection based on TF-IDF and Machine Learning Classifiers. *International Journal of Computer Science and Network Security (IJCSNS)*, 23(1), 26-31.
- [9] Prakoso, J. D., Yuniar, I. F., & Hidayat, R. (2022). Ensemble Learning for Indonesian Hoax News Detection Using Random Forest and Gradient Boosting. *Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI)*, 8(3), 267-276.