Journal of Computer and Information Systems Ampera
Vol. 7, No. 3, September 2026 e-ISSN: 2775-2496
https://journal-computing.org/index.php/journal-cisa/index

Perbandingan Metode Vektorisasi Teks dan Algoritma Pembelajaran Mesin Klasik untuk Klasifikasi Teks Bahasa Arab

Haidhi Angkawijana Tedja[1]
[1]Telkom Corporate University, Bandung, Indonesia
Email: [1]haidhiangkawijana@gmail.com

Abstract

Klasifikasi teks bahasa Arab merupakan tugas *Natural Language Processing* (NLP) yang menantang karena morfologi yang kaya, penggunaan harakat, serta keberagaman antara *Modern Standard Arabic* (MSA) dan ragam dialek. Penelitian ini membandingkan tiga metode vektorisasi teks — TF-IDF, *Count Vectorizer* (CV), dan *Word2Vec* — yang dikombinasikan dengan empat algoritma pembelajaran mesin klasik: *Logistic Regression*, *Linear SVM*, *Decision Tree*, dan *Multinomial Naïve Bayes*. Eksperimen dilakukan pada tiga dataset publik, yaitu SANAD (berita, 7 kelas), HARD (ulasan hotel, 4 kelas), dan OCLAR (ulasan dialek Lebanon, 5 kelas), dengan evaluasi menggunakan *accuracy* serta *precision*, *recall*, dan *F1-score* makro. Hasil menunjukkan metode berbasis frekuensi (TF-IDF dan CV) secara konsisten mengungguli *Word2Vec* pada model klasik, dengan akurasi terbaik 96,7% pada SANAD (SVM + TF-IDF) serta 71,3% pada HARD dan 62,0% pada OCLAR (keduanya LogReg + TF-IDF). Namun, *F1-score* makro mengungkap penurunan kinerja yang signifikan pada dataset tak seimbang (OCLAR hanya 0,29), yang tidak tampak dari akurasi. *Decision Tree* berkinerja terendah akibat *curse of dimensionality*, sedangkan *Word2Vec* dengan agregasi *average* hanya kompetitif pada dataset besar. Representasi berbasis frekuensi tetap menjadi *baseline* yang kuat dan efisien untuk klasifikasi teks bahasa Arab.

Keywords: klasifikasi teks, bahasa Arab, TF-IDF, Word2Vec, pembelajaran mesin

1. INTRODUCTION

Pertumbuhan pesat konten digital berbahasa Arab di internet — mulai dari portal berita, media sosial, hingga platform ulasan layanan — menghasilkan volume data teks tidak terstruktur yang sangat besar. Bahasa Arab merupakan salah satu bahasa dengan jumlah penutur terbanyak di dunia, namun sumber daya komputasi dan riset NLP untuk bahasa ini masih relatif terbatas dibandingkan bahasa Inggris [1], [2]. Kondisi ini mendorong kebutuhan akan metode otomatis yang mampu mengorganisasi, mengategorikan, dan menganalisis teks bahasa Arab secara efektif.

This work is licensed under a Creative Commons Attribution 4.0 International License.

Klasifikasi teks (*text classification*) adalah tugas menetapkan satu atau lebih label kategori pada sebuah dokumen teks. Tugas ini menjadi fondasi bagi berbagai aplikasi praktis seperti kategorisasi berita, penyaringan *spam*, dan analisis sentimen terhadap ulasan pelanggan. Berbeda dengan bahasa Inggris, klasifikasi teks bahasa Arab menghadapi sejumlah tantangan khusus, antara lain: (1) morfologi yang kompleks dan kaya, di mana satu akar kata dapat menurunkan ratusan bentuk turunan; (2) keberadaan harakat yang dapat mengubah makna namun sering tidak dituliskan; (3) tidak adanya huruf kapital sebagai penanda; serta (4) variasi yang signifikan antara *Modern Standard Arabic* (MSA) dan beragam dialek regional [3], [4].

Salah satu langkah krusial dalam klasifikasi teks adalah representasi teks ke dalam bentuk numerik (vektorisasi) agar dapat diproses oleh algoritma pembelajaran mesin. Dua pendekatan utama yang umum digunakan adalah representasi berbasis frekuensi, seperti *Term Frequency–Inverse Document Frequency* (TF-IDF) [5] dan *Count Vectorizer*, serta representasi berbasis *word embedding* seperti Word2Vec [6]. Metode berbasis frekuensi menghasilkan vektor berdimensi tinggi dan jarang (*sparse*) yang mengabaikan urutan dan konteks kata, sedangkan *word embedding* menghasilkan vektor padat (*dense*) yang mampu menangkap kedekatan semantik antarkata.

Meskipun pendekatan berbasis *deep learning* dan *transformer* (misalnya AraBERT) telah menunjukkan performa mutakhir [2], [7], [8], model pembelajaran mesin klasik tetap relevan karena efisiensinya yang tinggi, kebutuhan komputasi yang rendah, serta kemudahan interpretasi — faktor penting bagi penerapan di lingkungan dengan sumber daya terbatas. Model klasik juga masih banyak digunakan sebagai pembanding (*baseline*) pada studi-studi terkini analisis sentimen bahasa Arab [9]. Oleh karena itu, pemahaman mengenai kombinasi metode vektorisasi dan algoritma klasik mana yang paling efektif untuk bahasa Arab tetap memiliki nilai praktis.

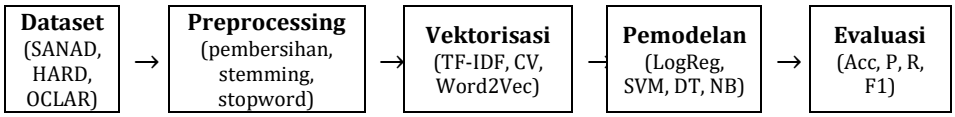
Penelitian ini bertujuan untuk:

1. Membandingkan efektivitas tiga metode vektorisasi (TF-IDF, *Count Vectorizer*, dan Word2Vec) pada tugas klasifikasi teks bahasa Arab;
2. Mengevaluasi kinerja empat algoritma pembelajaran mesin klasik (*Logistic Regression*, SVM, *Decision Tree*, dan *Naïve Bayes*) di atas representasi tersebut menggunakan accuracy serta precision, recall, dan F1-score;
3. Menganalisis pengaruh karakteristik dataset (ukuran, jumlah kelas, dan ragam bahasa) terhadap performa klasifikasi.

Kontribusi utama penelitian ini adalah penyajian perbandingan empiris yang komprehensif pada tiga dataset bahasa Arab dengan karakteristik berbeda (kategorisasi berita MSA, ulasan hotel campuran, dan ulasan layanan dialek Lebanon), yang dapat menjadi acuan *baseline* bagi penelitian klasifikasi teks bahasa Arab selanjutnya.

2. METHODS

Penelitian ini dilakukan melalui lima tahap utama: pengumpulan data, pra-pemrosesan teks, vektorisasi, pemodelan klasifikasi, dan evaluasi. Diagram alur metode secara umum disajikan pada Gambar 1.



Gambar 1. Alur penelitian klasifikasi teks bahasa Arab.

2.1 Dataset

Tiga dataset publik berbahasa Arab dengan karakteristik berbeda digunakan untuk menjamin keberagaman pengujian:

1. **SANAD (Single-label Arabic News Articles Dataset)** [1] — kumpulan artikel berita berbahasa Arab Standar Modern (MSA) yang dikumpulkan dari tiga portal berita (AlKhaleej, AlArabiya, dan Akhbarona). Dataset ini diberi label tujuh kategori: *culture*, *finance*, *medical*, *politics*, *religion*, *sports*, dan *tech*. Pada penelitian ini digunakan subset seimbang sebanyak 45.500 artikel (6.500 artikel per kategori).
2. **HARD (Hotel Arabic-Reviews Dataset)** [10] — kumpulan ulasan hotel dari situs Booking.com, ditulis dalam MSA maupun dialek. Setiap ulasan memiliki rating bintang. Pada penelitian ini digunakan 105.698 ulasan dengan empat kelas rating (1, 2, 4, dan 5).
3. **OCLAR (Opinion Corpus for Lebanese Arabic Reviews)** [11] — kumpulan ulasan layanan (restoran, hotel, toko, dan lainnya) dari situs Zomato, ditulis terutama dalam dialek Lebanon. Dataset terdiri atas 3.916 ulasan dengan skala rating 1–5, dan didominasi oleh ulasan berating tinggi (kelas tidak seimbang).

Ringkasan karakteristik ketiga dataset disajikan pada Tabel 1.

Tabel 1. Karakteristik dataset yang digunakan.

Dataset	Domain	Ragam Bahasa	Jumlah Sampel	Jumlah Kelas
SANAD	Berita	MSA	45.500	7
HARD	Ulasan hotel	MSA & Dialek	105.698	4
OCLAR	Ulasan layanan	Dialek Lebanon	3.916	5

2.2 Text Preprocessing

Tahap pra-pemrosesan bertujuan membersihkan dan menormalkan teks. Secara umum, prosesnya serupa dengan bahasa lain, namun disesuaikan dengan karakteristik bahasa Arab. Tahapan yang dilakukan adalah:

1. **Penghapusan tanda baca, angka, dan karakter non-Arab.** Seluruh karakter berkategori tanda baca (*punctuation*), huruf Latin, dan angka dihilangkan, kemudian spasi berlebih dirapikan.
2. **Penghapusan harakat (dediakritisasi).** Diakritik (harakat) dihapus menggunakan utilitas dari pustaka *CAMeL Tools* [12] agar bentuk kata menjadi konsisten.
3. **Tokenisasi.** Teks dipecah menjadi token kata menggunakan *simple word tokenizer* dari *CAMeL Tools*.
4. **Stemming.** Setiap token direduksi ke bentuk akarnya menggunakan **ISRI Stemmer** [13], yaitu algoritma *stemming* bahasa Arab yang tidak memerlukan kamus akar kata. Implementasi yang digunakan berasal dari pustaka NLTK.
5. **Penghapusan stop words.** Kata-kata umum yang tidak membawa informasi diskriminatif dihilangkan berdasarkan daftar *stop words* bahasa Arab.

Sebagai ilustrasi, Tabel 2 menyajikan penerapan berurutan tahapan tersebut pada satu contoh ulasan berbahasa Arab (tokenisasi dilakukan sebelum *stemming* dan tidak mengubah tampilan teks).

Tabel 2. Contoh penerapan tahapan pra-pemrosesan pada ulasan berbahasa Arab.

Tahap	Hasil
Teks asli	الفندق في وسط المدينة كان ممتازاً جداً، والخدمة رائعة 100%! أنصح به.
Penghapusan tanda baca, angka & non-Arab	الفندق في وسط المدينة كان ممتازاً جداً والخدمة رائعة أنصح به
Penghapusan harakat (dediakritisasi)	الفندق في وسط المدينة كان ممتازا جدا والخدمة رائعة أنصح به
Stemming (ISRI Stemmer)	ندق في وسط دين كان متز جدا خدم رئع نصح به
Penghapusan stop words (hasil akhir)	ندق وسط دين متز خدم رئع نصح

Arti ulasan: "Hotel di pusat kota sangat bagus, dan pelayanannya luar biasa 100%! Saya merekomendasikannya."

Perlu dicatat bahwa pra-pemrosesan tidak selalu menjamin peningkatan performa model; efektivitasnya bergantung pada karakteristik tugas dan dataset.

2.3 Vectorization

Teks hasil pra-pemrosesan dikonversi ke representasi numerik melalui tiga metode:

1. **Count Vectorizer (CV)** — merepresentasikan dokumen sebagai vektor frekuensi kemunculan kata (*bag-of-words*).
2. **TF-IDF** — memberi bobot pada setiap kata berdasarkan frekuensinya dalam dokumen (TF) dan kelangkaannya di seluruh korpus (IDF), sehingga kata-kata yang lebih diskriminatif memperoleh bobot lebih tinggi [5].
3. **Word2Vec** — melatih *word embedding* berdimensi 100 dari masing-masing korpus menggunakan arsitektur Word2Vec [6] yang diimplementasikan melalui pustaka *Gensim* [14], dengan parameter: vector_size=100, min_count=10, window=15, sample=6e-5, alpha=0,03, min_alpha=0,0007, negative=25, dan epochs=30. Karena Word2Vec menghasilkan vektor per kata, representasi tingkat dokumen diperoleh dengan mengagregasi vektor kata-kata penyusunnya menggunakan empat fungsi agregasi: *minimum*, *maximum*, *sum*, dan *average*.

2.4 Modeling and Evaluation

Empat algoritma pembelajaran mesin klasik dievaluasi, yaitu (1) *Logistic Regression* (LogReg); (2) *Linear Support Vector Machine* (SVM); (3) *Decision Tree* (DT); dan (4) *Multinomial Naïve Bayes* (NB), yang hanya diterapkan pada representasi berbasis frekuensi (TF-IDF dan CV) karena membutuhkan masukan bernilai non-negatif. Seluruh model diimplementasikan menggunakan pustaka *scikit-learn* [15] dengan hyperparameter bawaan (*default*), dan random_state=42 ditetapkan pada setiap komponen acak untuk menjamin reproduksibilitas.

Setiap korpus dibagi menjadi data latih dan data uji dengan rasio 80:20 menggunakan seed acak tetap (random_state=42), menghasilkan 3.116/779 (OCLAR), 84.494/21.124 (HARD), dan 36.400/9.100 (SANAD) sampel latih/uji. *TfidfVectorizer* dan *CountVectorizer* digunakan dengan parameter bawaan (unigram, tanpa pembatasan jumlah fitur). Untuk Word2Vec, vektor dokumen hasil agregasi digunakan sebagai masukan model.

Kinerja model diukur menggunakan empat metrik, yaitu *accuracy* (proporsi prediksi benar terhadap seluruh data uji) serta *precision*, *recall*, dan *F1-score* dengan rata-rata makro (*macro-average*). Rata-rata makro dipilih karena memberi bobot setara pada setiap kelas tanpa memandang jumlah sampelnya, sehingga mampu mengungkap kinerja model pada kelas minoritas — hal yang penting mengingat ketidakseimbangan distribusi kelas pada dataset OCLAR dan HARD.

3. RESULTS AND DISCUSSION

3.1 Frequency-Based Vectorization Results

Tabel 3 menyajikan akurasi keempat algoritma pada representasi TF-IDF dan *Count Vectorizer* untuk ketiga dataset, sedangkan Tabel 4 melengkapinya dengan metrik makro (*precision*, *recall*, dan *F1-score*) pada representasi TF-IDF sebagai vektorisasi terbaik.

Tabel 3. Akurasi model dengan vektorisasi TF-IDF dan Count Vectorizer (CV).

Model	OCLAR TF-IDF	OCLAR CV	HARD TF-IDF	HARD CV	SANAD TF-IDF	SANAD CV
Logistic Regression	**0,6200**	0,6110	**0,7129**	0,6960	0,9632	0,9574
Linear SVM	0,6072	0,5828	0,6933	0,6826	**0,9673**	0,9525
Decision Tree	0,5302	0,5404	0,6185	0,6181	0,8404	0,8416

Model	OCLAR TF-IDF	OCLAR CV	HARD TF-IDF	HARD CV	SANAD TF-IDF	SANAD CV
Naïve Bayes	0,5956	0,6175	0,6272	0,6936	0,9430	0,9426

Cetak tebal menunjukkan akurasi tertinggi pada setiap dataset.

Tabel 4. Metrik evaluasi rinci (rata-rata makro) model dengan vektorisasi TF-IDF.

Dataset	Model	Accuracy	Precision	Recall	F1-score
OCLAR	Logistic Regression	0,6200	0,4871	0,2787	0,2890
	Linear SVM	0,6072	0,4362	0,3429	0,3665
	Decision Tree	0,5302	0,3400	0,3032	0,3121
	Naïve Bayes	0,5956	0,4305	0,2145	0,1762
HARD	Logistic Regression	0,7129	0,6927	0,6595	0,6675
	Linear SVM	0,6933	0,6660	0,6443	0,6500
	Decision Tree	0,6185	0,5867	0,5849	0,5857
	Naïve Bayes	0,6272	0,7295	0,5189	0,4917
SANAD	Logistic Regression	0,9632	0,9633	0,9633	0,9633
	Linear SVM	0,9673	0,9673	0,9674	0,9673
	Decision Tree	0,8404	0,8412	0,8409	0,8410
	Naïve Bayes	0,9430	0,9445	0,9429	0,9433

Hasil pada Tabel 3 dan Tabel 4 memperlihatkan beberapa pola penting:
Dataset SANAD mencapai akurasi tertinggi secara keseluruhan, yakni **96,73%** menggunakan SVM + TF-IDF, diikuti *Logistic Regression* + TF-IDF (96,32%). Karena SANAD seimbang (6.500 sampel per kelas), nilai *F1-score* makro hampir identik dengan akurasi (0,9673), menandakan kinerja yang merata di seluruh kategori. Tingginya akurasi ini disebabkan oleh sifat dataset berupa artikel berita MSA yang panjang, formal, dan memiliki batas kategori yang jelas.
Dataset HARD memperoleh akurasi terbaik **71,29%** (*Logistic Regression* + TF-IDF), namun *F1-score* makronya lebih rendah (0,6675). Selisih ini menandakan kinerja yang belum merata antarkelas rating, karena tugas klasifikasi rating ulasan lebih subjektif dan batas antarkelas (misalnya rating berdekatan) lebih kabur.
Dataset OCLAR menghasilkan akurasi terbaik **62,00%** (*Logistic Regression* + TF-IDF), tetapi *F1-score* makronya hanya 0,2890 — bahkan

0,1762 untuk *Naïve Bayes*. Kesenjangan besar antara akurasi dan F1 ini terjadi karena dataset sangat tidak seimbang (didominasi rating 5), sehingga model cenderung memprediksi kelas mayoritas dan gagal mengenali kelas minoritas. Dataset ini paling menantang karena ukurannya kecil, berdialek Lebanon yang tidak baku, dan memiliki lima kelas.

3.2 Word2Vec Vectorization Results

Tabel 5 menyajikan akurasi model di atas representasi Word2Vec dengan empat fungsi agregasi (hanya LogReg, SVM, dan DT, karena vektor Word2Vec dapat bernilai negatif sehingga tidak kompatibel dengan Multinomial NB).

Tabel 5. Akurasi model dengan vektorisasi Word2Vec pada berbagai fungsi agregasi.

Dataset	Model	min	max	sum	average
OCLAR	LogReg	0,5879	0,5879	0,5956	0,5879
	SVM	0,5879	0,5879	**0,5995**	0,5879
	DT	0,5404	0,5032	0,5006	0,5058
HARD	LogReg	0,5927	0,5814	0,6513	**0,6606**
	SVM	0,5866	0,5766	0,6415	0,6552
	DT	0,5466	0,5298	0,5058	0,5098
SANAD	LogReg	0,8842	0,8769	0,9516	0,9546
	SVM	0,8871	0,8767	0,9527	**0,9563**
	DT	0,7025	0,6787	0,8730	0,8725

Dari Tabel 5 dapat diamati bahwa:
Pada dataset besar (SANAD dan HARD), fungsi agregasi **sum** dan **average** memberikan hasil jauh lebih baik dibanding *min* dan *max*. Hal ini karena penjumlahan dan rata-rata vektor kata menghasilkan representasi dokumen yang lebih stabil dan representatif terhadap makna keseluruhan. Word2Vec dengan agregasi *average* mencapai hasil terbaiknya pada **SANAD** (95,63% dengan SVM, *F1* makro 0,9564), namun tetap **sedikit di bawah** TF-IDF (96,73%). Pada **HARD**, agregasi *average* juga menjadi yang terbaik (66,06% dengan LogReg) tetapi masih tertinggal dari TF-IDF (71,29%).

Pada dataset kecil **OCLAR**, Word2Vec tidak memberikan keunggulan berarti (maksimum 59,95%) dengan *F1* makro yang sangat rendah (≤0,32), karena korpus yang kecil tidak cukup untuk melatih *embedding* yang berkualitas.

3.3 Discussion

TF-IDF/CV vs Word2Vec. Secara keseluruhan, representasi berbasis frekuensi (TF-IDF dan *Count Vectorizer*) mengungguli Word2Vec ketika dipadukan dengan model pembelajaran mesin klasik. Word2Vec dirancang untuk menangkap hubungan semantik antarkata dan umumnya menunjukkan keunggulannya saat digunakan bersama arsitektur *neural network* seperti LSTM atau RNN yang mampu memanfaatkan urutan vektor kata. Ketika vektor-vektor kata dipadatkan menjadi satu vektor dokumen melalui agregasi sederhana, sebagian besar informasi konteks dan urutan hilang, sehingga keunggulan Word2Vec tidak dapat dieksploitasi oleh model klasik. Temuan ini sejalan dengan studi komparatif yang menunjukkan bahwa keunggulan *word embedding* baru tampak nyata ketika dipasangkan dengan arsitektur *deep learning* seperti BiLSTM dan CNN [2]. Akibatnya, untuk *pipeline* berbasis pembelajaran mesin klasik, TF-IDF tetap menjadi pilihan yang lebih unggul dan efisien.

Akurasi vs F1 pada data tak seimbang. Penambahan metrik makro mengungkap bahwa akurasi saja dapat menyesatkan pada dataset dengan distribusi kelas yang timpang. Pada OCLAR yang didominasi rating 5, akurasi terbaik mencapai 62,00% tetapi *F1-score* makro hanya 0,2890, menandakan model cenderung memprediksi kelas mayoritas dan gagal pada kelas minoritas. Pola serupa, meski lebih ringan, tampak pada HARD (akurasi 71,29% vs *F1* makro 0,6675). Sebaliknya, pada SANAD yang seimbang, akurasi dan *F1* makro hampir identik (0,9673). Temuan ini menegaskan pentingnya pelaporan metrik makro di samping akurasi, khususnya untuk tugas analisis sentimen berbasis rating yang kerap tidak seimbang.

Kinerja Decision Tree. *Decision Tree* secara konsisten memberikan akurasi terendah pada hampir seluruh skenario. Hal ini disebabkan oleh permasalahan *curse of dimensionality*: representasi teks (terutama TF-IDF dan CV) menghasilkan ruang fitur berdimensi sangat tinggi dan jarang, sedangkan *Decision Tree* cenderung melakukan pemisahan berdasarkan satu fitur pada satu waktu sehingga mudah mengalami *overfitting* dan kesulitan menggeneralisasi pada data berdimensi tinggi. Model linier seperti *Logistic Regression* dan SVM jauh lebih sesuai untuk ruang fitur semacam ini.

Pengaruh karakteristik dataset. Hasil menunjukkan korelasi yang jelas antara karakteristik dataset dan tingkat kesulitan klasifikasi. Dataset SANAD yang besar, formal (MSA), dan memiliki batas kategori topik yang tegas mencapai akurasi tertinggi. Sebaliknya, OCLAR yang kecil dan berbahasa dialek menghasilkan akurasi dan F1 terendah. Hal ini menegaskan bahwa ukuran korpus, keformalan bahasa, dan ketegasan batas antarkelas merupakan faktor penentu utama keberhasilan klasifikasi teks bahasa Arab. Tingkat akurasi pada tugas analisis sentimen yang diperoleh penelitian ini juga sebanding dengan studi-studi terkini yang menggunakan model pembelajaran mesin klasik pada teks bahasa Arab [9].

Catatan mengenai pra-pemrosesan. Sesuai pengamatan selama eksperimen, pra-pemrosesan tidak menjamin peningkatan performa secara otomatis. Proses *stemming* yang agresif pada bahasa Arab dapat menyatukan kata-kata yang sebenarnya berbeda makna ke akar yang sama, sehingga dalam kasus tertentu justru menghilangkan informasi diskriminatif. Oleh karena itu, pemilihan strategi pra-pemrosesan sebaiknya disesuaikan dengan karakteristik tugas.

4. CONCLUSION

Penelitian ini membandingkan tiga metode vektorisasi teks dan empat algoritma pembelajaran mesin klasik untuk klasifikasi teks bahasa Arab pada tiga dataset dengan karakteristik berbeda, menggunakan metrik accuracy serta precision, recall, dan F1-score makro. Hasil utama yang diperoleh adalah:

1. Representasi berbasis frekuensi (TF-IDF dan *Count Vectorizer*) secara konsisten mengungguli Word2Vec ketika dipadukan dengan model klasik. Akurasi terbaik dicapai pada SANAD (96,73% dengan SVM + TF-IDF), HARD (71,29% dengan LogReg + TF-IDF), dan OCLAR (62,00% dengan LogReg + TF-IDF).
2. Metrik *F1-score* makro mengungkap bahwa kinerja pada dataset tak seimbang jauh lebih rendah daripada yang disiratkan akurasi (mis. OCLAR hanya 0,289), sehingga pelaporan metrik makro penting untuk evaluasi yang jujur.
3. Word2Vec dengan agregasi *average* hanya kompetitif pada dataset besar, namun tetap sedikit di bawah TF-IDF; representasi ini lebih cocok dipasangkan dengan arsitektur *neural network*.
4. *Decision Tree* berkinerja terburuk akibat *curse of dimensionality*, sedangkan model linier (*Logistic Regression* dan SVM) paling sesuai untuk representasi teks berdimensi tinggi. Karakteristik dataset

ukuran, keformalan bahasa, dan ketegasan batas antarkelas — sangat memengaruhi tingkat akurasi.

Untuk penelitian selanjutnya, disarankan eksplorasi terhadap *word embedding* kontekstual berbasis *transformer* (seperti AraBERT) serta arsitektur *deep learning* yang mampu memanfaatkan informasi urutan kata — seperti LSTM dan Bi-LSTM yang telah terbukti memberikan akurasi tinggi pada klasifikasi teks bahasa Arab [7], [8] — guna meningkatkan performa pada dataset berbahasa dialek seperti OCLAR. Penanganan ketidakseimbangan kelas (misalnya *oversampling* atau pembobotan kelas) juga layak dieksplorasi untuk meningkatkan F1 pada dataset seperti OCLAR dan HARD.

ACKNOWLEDGMENT

Penulis menyampaikan terima kasih kepada para penyedia dataset publik — SANAD, HARD, dan OCLAR — yang telah membuat data mereka tersedia secara terbuka bagi komunitas riset, sehingga penelitian ini dapat terlaksana. Penulis juga berterima kasih kepada para pengembang pustaka sumber terbuka *scikit-learn*, *NLTK*, *Gensim*, dan *CAMeL Tools* yang menjadi tulang punggung implementasi penelitian ini, serta kepada semua pihak yang telah memberikan dukungan dan masukan selama proses penelitian.

5. REFERENCES

[1] O. Einea, A. Elnagar, and R. Al Debsi, "SANAD: Single-label Arabic news articles dataset for automatic text categorization," *Data in Brief*, vol. 25, art. 104076, 2019, doi: 10.1016/j.dib.2019.104076.

[2] S. F. Sabbeh and H. A. Fasihuddin, "A comparative analysis of word embedding and deep learning for Arabic sentiment classification," *Electronics*, vol. 12, no. 6, art. 1425, 2023, doi: 10.3390/electronics12061425.

[3] F. Assiri and N. Alghamdi, "A comparative study of word embedding models for Arabic text," *IJCSNS International Journal of Computer Science and Network Security*, vol. 22, no. 8, pp. 399–406, 2022.

[4] A. Elnagar, R. Al-Debsi, and O. Einea, "Arabic text classification using deep learning models," *Information Processing & Management*, vol. 57, no. 1, art. 102121, 2020, doi: 10.1016/j.ipm.2019.102121.

[5] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988, doi: 10.1016/0306-4573(88)90021-0.

[6] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint* arXiv:1301.3781, 2013, doi: 10.48550/arXiv.1301.3781.

[7] A. Al-Qerem, M. Raja, S. Taqatqa, and M. R. A. Sara, "Utilizing deep learning models (RNN, LSTM, CNN-LSTM, and Bi-LSTM) for Arabic text classification," in *Artificial Intelligence-Augmented Digital Twins*, Studies in Systems, Decision and Control, vol. 503. Cham, Switzerland: Springer, 2024, pp. 267–278, doi: 10.1007/978-3-031-43490-7_22.

[8] W. Alosaimi et al., "ArabBert-LSTM: Improving Arabic sentiment analysis based on transformer model and Long Short-Term Memory," *Frontiers in Artificial Intelligence*, vol. 7, art. 1408845, 2024, doi: 10.3389/frai.2024.1408845.

[9] H. Alamoudi et al., "Arabic sentiment analysis for student evaluation using machine learning and the AraBERT transformer," *Engineering, Technology & Applied Science Research*, vol. 13, no. 5, pp. 11945–11952, 2023, doi: 10.48084/etasr.6347.

[10] A. Elnagar, Y. S. Khalifa, and A. Einea, "Hotel Arabic-reviews dataset construction for sentiment analysis applications," in *Intelligent Natural Language Processing: Trends and Applications*, Studies in Computational Intelligence, vol. 740. Cham, Switzerland: Springer, 2018, pp. 35–52, doi: 10.1007/978-3-319-67056-0_3.

[11] M. Al Omari, "Opinion Corpus for Lebanese Arabic Reviews (OCLAR)," UCI Machine Learning Repository, 2019, doi: 10.24432/C52894.

[12] O. Obeid et al., "CAMeL Tools: An open source Python toolkit for Arabic natural language processing," in *Proc. 12th Language Resources and Evaluation Conf. (LREC)*, Marseille, France, 2020, pp. 7022–7032.

[13] K. Taghva, R. Elkhoury, and J. Coombs, "Arabic stemming without a root dictionary," in *Proc. Int. Conf. Information Technology: Coding and Computing (ITCC)*, vol. 1, Las Vegas, NV, USA, 2005, pp. 152–157, doi: 10.1109/ITCC.2005.90.

Vol. 7, No. 3, September 2026 e-ISSN: **2775-2496**

https://journal-computing.org/index.php/journal-cisa/index

[14] R. Řehůřek and P. Sojka, "Software framework for topic modelling with large corpora," in *Proc. LREC 2010 Workshop on New Challenges for NLP Frameworks*, Valletta, Malta, 2010, pp. 45–50.

[15] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.