ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v23i1.14640

Sentiment Analysis of Traveloka App User Reviews Using Word2vec and LSTM

Analisis Sentimen Ulasan Pengguna Aplikasi Traveloka Menggunakan Word2vec dan LSTM

Danica Kirana[1], Frans Richard Kodong [2]

[1,2] Informatika, Universitas Pembangunan Nasional Veteran Yogyakarta, Indonesia

[1*]danicakirana13@gmail.com, [2]frans.ricard@upnyk.ac.id
**: Penulis korenspondensi (corresponding author)*

Informasi Artikel

Received: December 2025
Revised: January 2026
Accepted: January 2026
Published: February 2026

Abstract

Objective: As the travel trend in Indonesia increases, online travel agent (OTA) services such as Traveloka are becoming increasingly popular. However, with high competition in this industry, companies need to understand customer sentiment to improve service quality. This study aims to develop an automated sentiment analysis model on Traveloka app user reviews using a deep learning approach with Long 0Short-Term Memory (LSTM) and Word2Vec word representation. Design/method/approach: This study uses a quantitative method with stages including data collection, preprocessing, labeling, and modeling. The data used comes from Kaggle, which contains 12,689 Traveloka user reviews on the Google Play Store between January and October 2023. Preprocessing is carried out using case folding, tokenization, stopword removal, and stemming. Next, the data is represented in vector form using Word2Vec before being trained with an LSTM model. Experiments are conducted with various vector sizes (100, 200, 300, and 400) to evaluate their effect on model accuracy. Results: The test results show that the LSTM model with a vector size of 400 and a training dataset of 4073 achieved the highest accuracy of 89%, while vector sizes of 100, 200, and 300 produced accuracies of 85%, 86%, and 82%, respectively. This indicates that the dimensionality of word representation and the amount of data can affect the model's performance in understanding user sentiment. Originality/state of the art: This study contributes to the development of deep learning-based sentiment analysis models in Indonesian, especially in the travel services sector. Different from previous studies that mostly use shallow learning methods such as Naïve Bayes and SVM, this study adopts the Word2Vec and LSTM approaches that are

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v23i1.14640

more effective in capturing semantic relationships between words. This study also provides insights into the effect of Word2Vec vector size and training dataset size on LSTM in improving the accuracy of sentiment analysis models.

Abstrak

Keywords:LSTM; Sentiment Analysis; Word2Vec; traveloka

Kata kunci: LSTM; Analisis Sentimen; Word2Vec; traveloka

Tujuan: Seiring dengan meningkatnya tren traveling di Indonesia, layanan online travel agent (OTA) seperti Traveloka menjadi semakin populer. Namun, dengan tingginya persaingan di industri ini, perusahaan perlu memahami sentimen pelanggan untuk meningkatkan kualitas layanan. Penelitian ini bertujuan untuk mengembangkan model analisis sentimen otomatis pada ulasan pengguna aplikasi Traveloka menggunakan pendekatan deep learning dengan Long 0Short-Term Memory (LSTM) dan representasi kata Word2Vec. Perancangan/metode/pendekatan: Penelitian ini menggunakan metode kuantitatif dengan tahapan yang mencakup pengambilan data, preprocessing, labeling, dan pemodelan. Data yang digunakan berasal dari Kaggle, yang berisi 12.689 ulasan pengguna Traveloka di Google Play Store dalam rentang waktu Januari hingga Oktober 2023. Preprocessing dilakukan dengan case folding, tokenisasi, stopword removal, dan stemming. Selanjutnya, data direpresentasikan dalam bentuk vektor menggunakan Word2Vec sebelum dilatih dengan model LSTM. Eksperimen dilakukan dengan berbagai ukuran vektor (100, 200, 300, dan 400) untuk mengevaluasi pengaruhnya terhadap akurasi model. Hasil: Hasil pengujian menunjukkan bahwa model LSTM dengan vector size 400 dan jumlah data pelatihan sebanyak 4073 mencapai akurasi tertinggi sebesar 89%, sementara ukuran vektor 100, 200, dan 300 menghasilkan akurasi masing-masing 85%, 86%, dan 82%. Hal ini menunjukkan bahwa dimensi representasi kata dan juga jumlah data dapat mempengaruhi performa model dalam memahami sentimen pengguna. Keaslian/ state of the art: Penelitian ini berkontribusi dalam pengembangan model analisis sentimen berbasis deep learning dalam bahasa Indonesia, khususnya di sektor layanan perjalanan. Berbeda dari penelitian sebelumnya yang banyak menggunakan metode pembelajaran dangkal seperti Naïve Bayes dan SVM, penelitian ini mengadopsi pendekatan Word2Vec dan LSTM yang lebih efektif dalam menangkap hubungan semantik antar kata. Studi ini juga memberikan wawasan mengenai pengaruh ukuran vektor Word2Vec dan

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v23i1.14640

ukuran data pelatihan pada LSTM dalam meningkatkan akurasi model analisis sentimen.

1. Pendahuluan

Traveling telah menjadi tren yang berkembang di Indonesia, di mana sektor pariwisata berperan penting dalam perekonomian nasional. Kemunculan Online Travel Agent (OTA) seperti Traveloka semakin memudahkan masyarakat dalam merencanakan perjalanan, termasuk pemesanan transportasi, akomodasi, dan layanan terkait lainnya [1]. Minat masyarakat terhadap traveling terus meningkat. Data dari Google menunjukkan bahwa tren traveling di Indonesia mengalami kenaikan hingga 80% pada tahun 2023. Kementerian Pariwisata dan Ekonomi Kreatif (Kemenparekraf) juga memperkirakan jumlah kunjungan wisatawan mancanegara mencapai 9 juta kunjungan hingga akhir tahun [2]. Namun, dengan meningkatnya persaingan di industri ini, memahami sentimen pelanggan menjadi faktor krusial bagi perusahaan untuk meningkatkan kualitas layanan dan mempertahankan kepuasan pengguna [3].

Analisis sentimen memungkinkan perusahaan untuk memahami opini pelanggan berdasarkan ulasan yang mereka berikan. Analisis sentimen dapat dilakukan dengan berbagai metode, termasuk pendekatan berbasis machine learning seperti Naïve Bayes, Support Vector Machine (SVM), dan Maximum Entropy [4]. Metode konvensional seperti pendekatan berbasis leksikon dan pembelajaran mesin, termasuk Naïve Bayes dan Support Vector Machine (SVM), sering kali memerlukan rekayasa fitur yang kompleks dan memiliki keterbatasan dalam menangkap hubungan semantik antar kata [5].

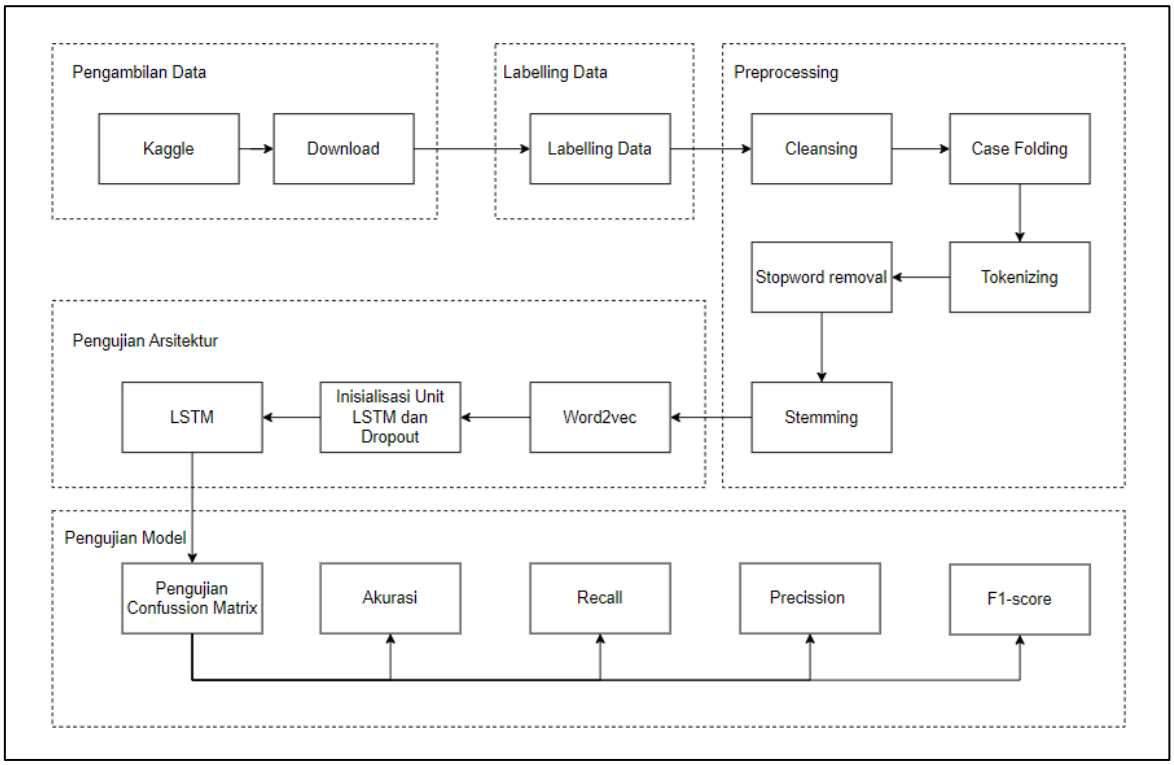
Teknik deep learning, khususnya Long Short-Term Memory (LSTM) yang dipadukan dengan Word2Vec sebagai representasi kata, menawarkan solusi yang lebih efektif dalam klasifikasi sentimen [4]. Word2Vec bekerja dengan memanfaatkan informasi lokal dari kata-kata sekitar untuk menentukan makna kata tertentu [6]. Untuk mengatasi tantangan tersebut, pendekatan berbasis deep learning, seperti Long Short-Term Memory (LSTM), menawarkan solusi yang lebih baik. LSTM, yang merupakan pengembangan dari Recurrent Neural Network (RNN), mampu menangani permasalahan vanishing gradient dan memiliki kemampuan untuk memproses data teks pendek dengan lebih baik [7]

Penelitian ini berfokus pada implementasi analisis sentimen terhadap ulasan pengguna Traveloka yang dikumpulkan dari Google Play Store. Dengan menggunakan Word2Vec untuk representasi kata dan LSTM sebagai model klasifikasi, penelitian ini bertujuan untuk mengevaluasi pengaruh berbagai ukuran vektor Word2Vec terhadap akurasi klasifikasi sentimen. Melalui pendekatan deep learning, penelitian ini berkontribusi dalam pengembangan model analisis sentimen yang lebih akurat untuk bahasa Indonesia, khususnya dalam sektor layanan perjalanan.

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v23i1.14640

2. Metode/Perancangan

Tahapan metodologi penelitian dimulai dari identifikasi masalah hingga memiliki hasil penelitian. Metodologi yang di gunakan pada penelitian ini adalah metode kuantitatif. Adapun tahapan metodologi penelitian dilakukan dapat dilihat pada **Gambar 1**.



Gambar 1. Metodologi Penelitian

2.1. Pengumpulan Data

Data yang digunakan diambil dari situs kaggle. Dataset berisikan ulasan pengguna aplikasi traveloka dari awal peluncuran aplikasi hingga bulan Oktober 2023. Data yang digunakan dalam penelitian ini yaitu data sepuluh bulan terakhir atau bulan Januari sampai dengan Oktober. Data yang digunakan dikumpulkan dalam format .csv dan berjumlah 12689 data. Sample dari data yang telah diunduh dari situs kaggle dapat dilihat pada gambar 2.

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v18i1.xxxx

	reviewId	userName	userImage	content	score	thumbsUpC	reviewCreatedVersion	at	replyContent	repliedAt	appVersion
0	9d5510:	Muh Sup	https://pla	Ok	5	0	3.88.0	2023-10-22 05:10:15	Hi, thank y	2023-10-22 05:12:09	3.88.0
1	a51b34:	Sry Haty	https://pla	Lebih Memudahkan Untu	5	0	3.88.0	2023-10-22 05:06:35	Hi Kak, sel	2023-10-22 05:12:11	3.88.0
2	fd7e4b(	Sigit Mar	https://pla	Selalu pakai Traveloka un	5	0	3.88.0	2023-10-22 04:59:50	Halo Kak,	2023-10-22 05:12:16	3.88.0
3	0dc695(	Awawy A	https://pla	memudahkan untuk bepe	5	0	3.88.0	2023-10-22 04:55:44	Hi Kak, sel	2023-10-22 05:12:19	3.88.0
4	10a64a1	Uchu Abr	https://pla	Up	5	0	3.82.0	2023-10-22 04:51:52	Thank you	2023-10-22 05:12:22	3.82.0
5	3b2b2d	Sumiati	https://pla	The best app nya	5	0	3.88.0	2023-10-22 04:50:03	Hi, thank y	2023-10-22 05:12:25	3.88.0
6	1d5d15(	Riyan An	https://pla	Aplikasinya bagus	5	0		2023-10-22 04:38:36	Halo Kak,	2023-10-22 04:42:06	
7	c390c68	Ipah Rah	https://pla	Untuk cari tiket akomoda	5	0		2023-10-22 04:33:32	Hi Kak, sel	2023-10-22 04:42:09	
8	1a8b72(	Asih Kary	https://pla	Mempermudah kita untu	5	0		2023-10-22 04:31:44	Hi Kak, ter	2023-10-22 04:42:11	
9	f8c90de	Ninda Kh	https://pla	Aplikasi andalan untuk ke	5	0	3.70.0	2023-10-22 04:31:30	Halo Kak,	2023-10-22 04:42:13	3.70.0
10	52ad09(	William T	https://pla	Mau telepon cs aja gk bis	1	0	3.88.0	2023-10-22 03:06:29	Hai Willia	2023-10-22 03:23:33	3.88.0
11	5874c21	Jonathan	https://pla	Traveloka mantap	5	0	3.88.0	2023-10-22 02:39:46	Halo Kak,	2023-10-22 02:42:08	3.88.0
12	a88c832	Lilis	https://pla	Penanganan masalah ,me	1	0	3.88.0	2023-10-22 02:37:12	Hai Lilis, n	2023-10-20 08:51:38	3.88.0
13	f192bcb	yanti oke	https://pla	ok	5	0	3.64.0	2023-10-22 00:55:20	Hi, thank y	2023-10-22 01:12:07	3.64.0
14	22c692(	Satriojag	https://pla	Bagus tingkatkan lg	5	0	3.88.0	2023-10-22 00:50:30	Hi Kak, sel	2023-10-22 01:12:10	3.88.0
15	c44546f	shufiatur	https://pla	bikin gampang buat pese	5	0	3.88.0	2023-10-22 00:10:24	Halo Kak,	2023-10-22 00:12:06	3.88.0

Gambar 1 Dataset sebelum preprocessing

Dataset pada gambar 2 memiliki 11 kolom yang berisi informasi lengkap dari ulasan yang diberikan dan juga informasi penggunanya. Data yang akan digunakan dalam penelitian ini hanyalah data score dan content. Data score nantinya akan digunkan untuk proses labelling dan data content akan masuk ke proses preprocessing untuk dibersihkan.

2.2. Labelling

Sebelum masuk ke preprocessing data diklasifikasikan menjadi dua kategori yaitu positif dan negatif. Pengklasifikasian dilakukan berdasarkan score yang diberikan oleh pengguna, ulasan dengan score 1 dan 2 akan masuk ke dalam kategori ulasan negatif dan diberikan label 0, sedangkan ulasan dengan score 3, 4, dan 5 akan masuk ke dalam kategori positif dan diberikan label 1.Pemilihan score sebagai acuan dalam proses labelling didasari oleh penelitian penelitian sebelumnya yang mendapatkan hasil cukup akurat dan leballing dengan metode ini juga lebih mudah. Proses labelling ini dilakukan agar data dapat dilatih menggunakan metode LSTM untuk bisa melakukan klasifikasi kalimat. Ilustrasi dari proses labelling dapat dilihat pada tabel 3.1

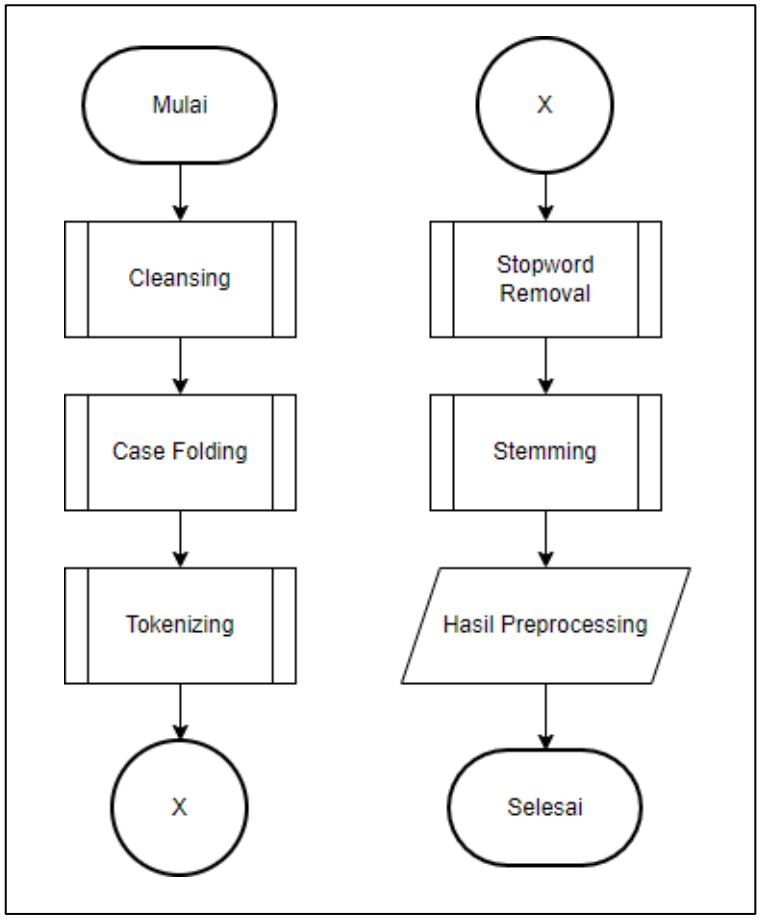
Tabel 3. 1 Ilustrasi Proses Labelling

content	score	label
The best app nya	5	1
Aplikasinya bagus	5	1
Untuk cari tiket akomodasi buat traveling benar-benar lebih mudah pake Traveloka. App yang sangat membantu	5	1
Mempermudah kita untuk beli tiket secara online pokoknya udah ga ribet ribet lagi deh	5	1
Aplikasi andalan untuk keperluan booking tiket pesawat dan hotel	5	1
Mau telepon cs aja gk bisa harus by email... Padahal mau refaund jadwal saya di ganti oleh pihak maskapai	1	0
Traveloka mantap	5	1
Penanganan masalah ,menyusahkan,ribet ,berbelit belit,uang tidak bisa kembali.parah!!!	1	0

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v18i1.xxxx

2.3. Data Preprocessing

Tahapan preprocessing dilakukan agar data yang digunakan dalam pemodelan lebih mudah untuk diolah, selain itu tahapan preprocessing juga dapat meningkatkan akurasi. Dalam penelitian ini terdapat beberapa langkah preprocessing seperti cleansing, case folding, tokenizing, stopword removal dan juga stemming. Flowchart dari tahapan preprocessing dapat dilihat pada gambar 3

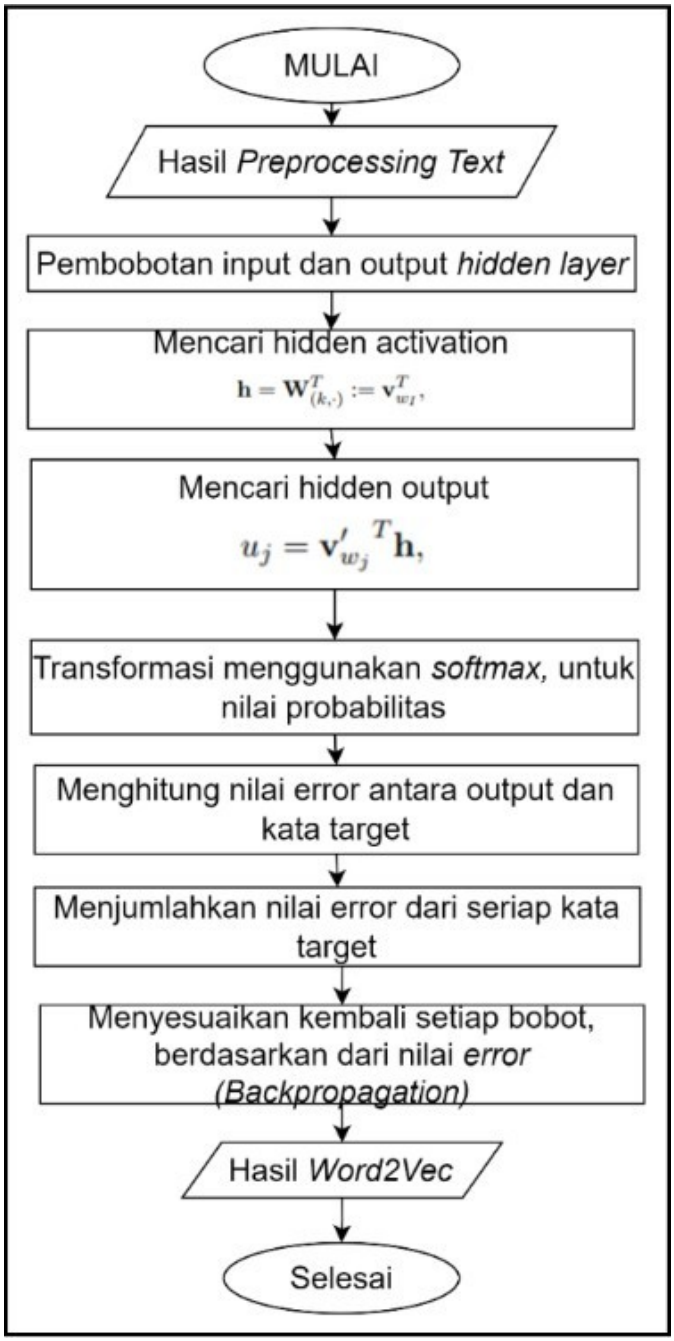


Gambar 3 Tahapan preprocessing

2.4. Pembobotan Word2vec

Pada tahapan ini data yang telah diproses dalam tahapan preprocessing akan masuk ke dalam proses word2vec. Data yang sudah bersih nantinya akan diubah ke dalam bentuk vektor. Pada tahap ini, penulis akan mendesain bagaimana cara proses data yang sudah dibersihkan bisa digunakan untuk analisis menggunakan LSTM. Berikut pada Gambar 4 untuk menjelaskan proses untuk word2vec.

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v18i1.xxxx



Gambar 4 Tahapan word2vec skip-gram

2.4.1. Data preparation

Merupakan tahapan untuk meyiapkan dua input pada proses word embedding. Suatu kata pada kalimat akan dilakukan one hot encoding atau mengubah kata menjadi bilangan biner, dengan kata target bernilai 1. Nilai window adalah 2 untuk menentukan kata kata yang bertetangga dengan target. Penentuan kata tetangga pada kalimat hasil preprocessing yaitu kalimat “pilih” “mudah” “harga” “murah” dapat dilihat pada tabel 1

Tabel 1 Penentuan kata tetangga Skip-Gram

NO	*Kata*	*Context Word*	*Target Word*
1	pilih	[1, 0, 0, 0]	[0, 1, 0, 0] [0, 0, 1, 0]
2	mudah	[0, 1, 0, 0]	[1, 0, 0, 0] [0, 0, 1, 0] [0, 0, 0, 1]
3	harga	[0, 0, 1, 0]	[1, 0, 0, 0] [0, 1, 0, 0] [0, 0, 0, 1]
4	murah	[0, 0, 0,]	[0, 1, 0, 0] [0, 0, 1, 0]

2.4.2. Hidden Layer

Tahap ini merupakan tahap pembobotan dari hasil *one hot encoding.* Dimana pada proses *word2vec,* bobot W dan W' akan diinisialisasikan sebagai *random number* sesuai dengan jumlah kosakata(N) dan dimensi *embedding* (V) sehingga dibentuk matriks dengan ukuran (4x5). Pembobotan diinisialisaikan dengan *range* -1 sampai dengan 1, yang diambil dari *numpy random* dengan ukuran 4x5

$$\begin{bmatrix} 0.123456 & 0.234567 & -0.145678 & 0.412345 & 0.123456 \\ 0.312345 & 0.512345 & 0.223456 & -0.334567 & 0.145678 \\ -0.456789 & 0.245678 & 0.178912 & 0.623456 & 0.334567 \\ 0.234567 & -0.356789 & 0.578912 & 0.167890 & -0.278912 \end{bmatrix}$$

Pada tahapannya untuk mendapatkan nilai hidden layer atau h dari hasil one hot encoding dilakukan dengan melakukan perkalian antara vector input dengan W. Pada tahapan ini kata yang digunakan adalah kata pertama yaitu kata "pilih", maka dari itu nilai x yang digunakan hasil one hot encoding dari kata "pilih" yang dapat dilihat pada tabel 1 di kolom context word

$$\textit{hidden layer} = \begin{bmatrix} 0.123456 & 0.312345 & -0.4567895 & 0.234567 \\ 0.234567 & 0.512345 & 0.245678 & -0.356789 \\ -0.145678 & 0.223456 & 0.178912 & 0.578912 \\ 0.412345 & -0.334567 & 0.623456 & 0.167890 \\ 0.123456 & 0.145678 & 0.334567 & -0.278912 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\textit{hidden layer} = \begin{bmatrix} 0.123456 \\ 0.234567 \\ -0.145678 \\ 0.412345 \\ 0.123456 \end{bmatrix}$$

Setelah didapatkan nilai hidden layer, selanjutnya menentukan *output layer(z)* dengan mengalikan dari *hidden layer* dan nilai bobot W' yang sudah diinisialisasi secara random. Untuk hasil perhitungannya dapat dilihat pada matriks dibawah

$$output\ layer = \begin{bmatrix} 0.742987 & 0.127747 & 0.32454 & 0.412345 & 0.824515 \\ 0.312345 & 0.974567 & 0.223456 & 0.334567 & 0.145678 \\ -0.456789 & 0.675632 & 0.178912 & 0.623456 & 0.877533 \\ 0.234567 & -0.356789 & 0.578912 & 0.875623 & -0.764429 \end{bmatrix} x \begin{bmatrix} 0.123456 \\ 0.234567 \\ -0.145678 \\ 0.412345 \\ 0.123456 \end{bmatrix}$$

$$output\ layer = \begin{bmatrix} 0.34609207 \\ 0.39055135 \\ 0.44143976 \\ 0.2950003 \end{bmatrix}$$

2.4.3. Softmax

Tahapan selanjutnya yaitu pencarian nilai probabilitas dengan menggunakan fungsi softmax sehingga didapatkan y_pred. Softmax berfungsi untuk mengembalikan distribusi probabilitas dengan jumlah entri sama dengan satu.

$$\text{softmax}(x_i) = \exp(x_i) / \sum_j \exp(x_j) \quad (1)$$

x_i adalah elemen ke-i dari vektor input, dan $\sum_j \exp(x_j)$ adalah jumlah dari semua nilai eksponensial dari elemen-elemen vektor input

Untuk menghitung softmax kita membutuhkan nilai z' dengan mengurangi nilai pada setiap output layer(z) dengan nilai z maksimum, yaitu 0.44143976, lalu didapatkan hasil

$$z' = \begin{bmatrix} 0.34609207 - 0.44143976 \\ 0.39055135 - 0.44143976 \\ 0.44143976 - 0.44143976 \\ 0.2950003 - 0.44143976 \end{bmatrix} = \begin{bmatrix} -0.09534769 \\ -0.05088841 \\ 0 \\ -0.14643946 \end{bmatrix} \quad (2)$$

Nilai z' yang telah didapatkan kemudian dihitung nilai eksponensialnya dengan rumus $e^{z'}$, yang nantinya akan mendapatkan hasil

$$\begin{bmatrix} 0.909068 \\ 0,950384 \\ 1 \\ 0.863778 \end{bmatrix}$$

Kemudian jumlahkan semua nilai $e^{z'}$

$$= 0.909068 + 0,950384 + 1 + 0.863778 = 3.72323 \quad (3)$$

Untuk mendapatkan nilai softmax, bagi masing masing nilai pada $e^{z'}$ dengan jumlah semua nilai $e^{z'}$ yang telah didapatkan tadi, yaitu 3.72323. Setelah perhitungan dilakukan maka didapatkan hasil softmax yaitu :

$$\begin{bmatrix} 0.244161 \\ 0.255795 \\ 0.268583 \\ 0.231999 \end{bmatrix}$$

2.4.4. Menghitung nilai error

Dari hasil *softmax* akan dilakukan pencarian *error* dengan cara melakukan pengurangan pada

nilai probabilitas dengan nilai kata target yang telah di *one hot encoded*. Seperti pada kata 'pilihan' memiliki dua kata target yaitu 'mudah, 'murah. Maka pada kata konteks akan dicari nilai *error,* sebagai contoh kita ambil kata 'pilihan' dengan vector [1,0,0,0] sehingga perhitungan nilai error sebagai berikut. dengan melakukan perhitungan sesuai dengan persamaan 4. Hasil perhitungan sum of error dapat dillihat pada tabel 2.

$$e = \sum^{c}(y_{jj} - x_c) \quad (4)$$

$$e = (0.244161 - 1), (0.255795 - 0), (0.268583 - 0), (0.231999 - 0) \quad (5)$$

$$e = -0.755839, 0.255795, 0.268583, 0.231999 \quad (6)$$

Tabel 2 Hasil sum of error

Y_Pred	Probabilitas	Vektor Input	Error	Sum (Error)
mudah	0.244161	[0	0.244161	[0.488322
	0.255795	1	−0.744205	-0.48841
	0.268583	0	0.268583	0.537166
	0.231999	0]	0.231999	-0.536002]
murah	0.244161	[0	0.244161	
	0.255795	0	0.255795	
	0.268583	0	0.268583	
	0.231999	1]	−0.768001	

Perhitungan tersebut dilakukan terhadap setiap kata target sebagai contoh pada kata 'pilihan' memiliki dua kata target yaitu 'mudah', dan 'murah'.

2.4.5. Backpropagation

Backpropagation memiliki tujuan untuk menyesuaikan kembali weight dan bias berdasarkan error yang didapatkan. Pada tahap ini dicari nilai delta dari masing masing W. Pertama mencari delta W' dilakukan dengan melakukan perkalian antara nilai hidden layer dengan sum(error) yang hasilnya dapat dilihat pada matriks dibawah

$$\Delta W' = \begin{bmatrix} 0.123456 \\ 0.234567 \\ -0.145678 \\ 0.412345 \\ 0.123456 \end{bmatrix} . \begin{bmatrix} 0.488322 \\ -0.48841 \\ 0.537166 \\ -0.536002 \end{bmatrix} \quad (7)$$

$$\Delta W' = 0.234567 \begin{bmatrix} 0.345678 & 0.543255 & -0.766532 & 0.234567 \\ -0.145678 & 0.354234 & 0.245678 & -0.356789 \\ 0.472345 & 0.223456 & 0.378912 & 0.478912 \\ 0.123456 & -0.334567 & 0.623456 & 0.167890 \end{bmatrix} \quad (8)$$

Karena matriks diatas tidak dapat dilakukan secara biasa, maka nilai error dihitung menggunakan transpose. Kemudian, pencarian *delta W* langkah pertama yang harus dilakukan adalah melakukan perkalian antara Sum (*error*) dengan bobot delta *W*' yang sudah dihasilkan pada proses sebellumnya. Hasil dari perkalian selanjutnya akan dikalikan dengan nilai vector konteks. Hasil perhitungan *delta W* adalah sebagai berikut :

$$\begin{bmatrix} 0.039987 \\ -0.043745 \\ 0.064754 \\ -0.046836 \\ 0.73876 \end{bmatrix}$$

2.4.6. Update Weight

Setelah didapatkan nilai delta dari setiap weight, selanjutnya nilai weight akan diperbarui dengan cara nilai weight – learning rate (0.05)×delta weight sebagi contoh pada kata ‘pilih’ dalam kata ‘pilih mudah harga murah’.Maka dari hasil *word embedding* dari perhitungan word2vec *skip-gram* di setiap kata dapat dilihat pada tabel 3

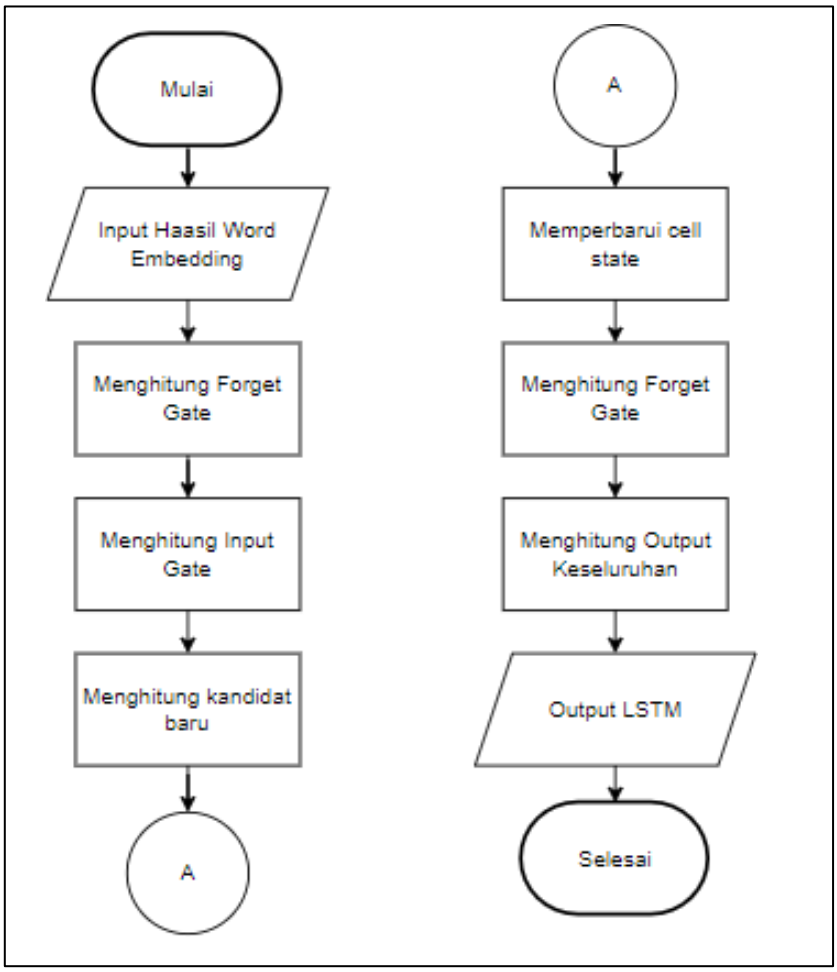
Tabel 3 Hasil pemrosesan word2vec

Kata	Dimensi				
	D1	D2	D3	D4	D5
pilih	0.029187	-0.003746	0.014654	-0.026836	0.019876
mudah	0.010729	0.043252	-0.028431	0.014032	-0.005961
harga	-0.036245	0.017132	0.001337	0.029356	-0.034908
murah	0.022169	-0.041168	-0.007224	0.033941	-0.008241

2.5. LSTM

Hasil output word embedding kemudian masuk pada proses LSTM. Di dalam LSTM terdapat memory cell dan gate units yang dapat dilihat flowchartnya pada gambar 5 Proses dimulai dengan penerimaan input dalam bentuk urutan data, yang kemudian diproses melalui lapisan embedding sebelum masuk ke unit LSTM. Setiap unit LSTM memperbarui status memori dengan mempertimbangkan informasi sebelumnya melalui mekanisme *gates*, sehingga memungkinkan model untuk mempertahankan informasi penting dan mengabaikan informasi yang tidak relevan. Setelah melewati lapisan LSTM, hasil keluaran diteruskan ke lapisan fully connected untuk menghasilkan prediksi akhir.

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v18i1.xxxx



Gambar 5 Flowchart tahapan LSTM

Langkah langkah lstm yang digunakan akan secara lengkap dijelaskan sebagai berikut :

2.5.1. Menghitung forget gate

Proses menghitung forget gate menggunakan rumus pada persamaan 4. Variabel weight forget gate dan bias forget gate di inisialisasi awal dengan nilai random, sedangkan untuk variabel output proses sebelumnya merupakan output dari proses sebelumnya. Sehingga masing masing variabel dijabarkan sebagai berikut :

$$h_{t-1} = [0 \quad 0 \quad 0 \quad 0 \quad 0] \tag{9}$$

$$x_t = [0.029187 \quad -0.003746 \quad 0.014654 \quad -0.026836 \quad 0.019876] \tag{10}$$

$$b_f = [0.112345 \quad 0.234567 \quad 0.167890 \quad -0.1789123] \tag{11}$$

$$W_f = \begin{bmatrix} 0.123456 & 0.234567 & -0.1456789 & 0.412345 & 0.123456 \\ 0.312345 & 0.512345 & 0.223456 & -0.334567 & 0.145678 \\ -0.456789 & 0.245678 & 0.178912 & 0.623456 & 0.334567 \\ 0.234567 & -0.356789 & 0.578912 & 0.167890 & -0.278912 \end{bmatrix} \tag{12}$$

$$\sigma = \frac{1}{1-e^x} \tag{13}$$

Nilai hidden state diambil dari timestep sebelumnya, namun karena ini perhitungan pertama maka nilai hidden state diinisialisasikan dengan nilai 0. Kemudian untuk nilai x_t diambil dari nilai representasi vektor yang dapat dilihat pada tabel 3.5. Nilai f_t dihitung menggunakan rumus berikut :

$$f_t = \sigma(W_f . [h_{t-1} + x_t] + b_f) \quad (14)$$

Setelah memasukkan semua angka yang telah dijabarkan ke dalam rumus, maka hasil perhitungan forget gate yaitu :

$$f_t = [0.52605717 \quad 0.56387751 \quad 0.53647936 \quad 0.45703254] \quad (15)$$

2.5.2. Menghitung input gate

Proses selanjutnya menghitung input gate menggunakan rumus pada persamaan 5, sama seperti forget gate, ada beberapa variabel yang dibutuhkan yaitu weight input gate, input, output dari proses sebelumnya, dan bias input gate. Nilai W_i dan b_i dijabarkan dengan angka random sebagai berikut :

$$W_i = \begin{bmatrix} 0.123456 & 0.234567 & -0.1456789 & 0.412345 & 0.123456 \\ 0.312345 & 0.512345 & 0.223456 & -0.334567 & 0.145678 \\ -0.456789 & 0.245678 & 0.178912 & 0.623456 & 0.334567 \\ 0.234567 & -0.356789 & 0.578912 & 0.167890 & -0.278912 \end{bmatrix} \quad (16)$$

$$b_i = [0.112345 \quad 0.234567 \quad 0.167890 \quad -0.1789123] \quad (17)$$

Langkah selanjutnya yaitu memasukkan nilai nilai tersebut ke dalam rumus input gate seperti yag dijabarkan pada langkah berikut :

$$i_t = \sigma(W_i . [h_{t-1} + x_t] + b_i) \quad (18)$$

Nilai input vektor kata dan juga hidden state yang digunakan pada langkah ini masih sama dengan langkah sebelumya. Setelah memasukkan semua angka yang telah dijabarkan ke dalam rumus, maka hasil perhitungan matriks dari input gate yaitu :

$$i_t = [0.55753076 \quad 0.55413628 \quad 0.43205974 \quad 0.59368759] \quad (19)$$

2.5.3. Menghitung nilai kandidat baru

Untuk menghitung kandidat cell state, diperlukan beberapa input utama, yaitu hidden state, input saat ini, bobot cell state kandidat, dan bias cell state kandidat. Variabel yang digunakan dijabarkan sebagai berikut :

$W_C =$ [[0.1890123, -0.2456789, 0.2890123, 0.3678901, 0.1456789],
[0.2456789, 0.3678901, -0.1789123, 0.1678901, 0.4789123], (20)
[0.1789123, 0.2456789, 0.5890123, -0.2678901, 0.1890123],
[-0.1890123, 0.3789123, 0.4890123, 0.2789123, 0.2789123]]

$b_C =$ [0.2789123, 0.1456789, 0.3678901, -0.1678901] (21)

$$\tanh = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \quad (22)$$

Pada langkah ini nilai W_i dan b_i dijabarkan dengan angka random, sedangkan vsriabel lainnya masih menggunakan nilai yang sama seperti pada langkah sebelumnya. Rumus untuk menghitung kandidat baru dapat dilihat sebagai berikut

$$\tilde{C}_t = tanh(W_C . [h_{t-1} + x_t] + b_C) \quad (23)$$

Setelah memasukkan semua angka yang telah dijabarkan ke dalam rumus, maka hasil perhitungan matriks dari nilai kandidat baru yaitu :

[0.27531669,0.15266021,0.37288476, −0.16799406]

2.5.4. Menghitung cell state

Setelah kandidat cell state dihitung, langkah berikutnya adalah menghitung cell state baru. Masing masing variabel yang digunakan dijabarkan sebagai berikut:

$$f_t = [0.52605717 \quad 0.56387751 \quad 0.53647936 \quad 0.45703254] \quad (24)$$

$$i_t = [0.55753076 \quad 0.55413628 \quad 0.43205974 \quad 0.59368759] \quad (25)$$

$$\tilde{C}_t = [0.27531669,0.15266021,0.37288476, -0.16799406] \quad (26)$$

Nilai forget gate, input gate dan nilai kandidat baru didapatkan dari perhitungan yang telah dilakukan sebelumnya, sehingga langkah selanjutnya adalah memasukkan nilai tersebut ke dalam rumus cell state dibawah :

$$C_t = f_t . C_{t-1} + i_t . \tilde{C}_t \quad (27)$$

Setelah memasukkan semua angka yang telah dijabarkan ke dalam rumus, maka hasil perhitungan matriks dari nilai cell state yaitu :

[0.15349753 0.08459456 0.16110849 −0.09973599]

2.5.5. Menghitung output gate

Setelah cell state diperbarui, langkah berikutnya adalah menghitung output gate. Variabel yang digunakan dijabarkan sebagai berikut :

$$W_o = [\ [0.4234567, 0.2890123, -0.1456789, 0.5123456, 0.3678901],\ [0.2678901, 0.3678901, 0.4678901, -0.2456789, 0.1567890],\ [-0.3789123, 0.5789123, 0.1456789, 0.4678901, 0.2456789],\ [0.5789123, -0.2789123, 0.3890123, 0.1678901, -0.3789123]] \quad (28)$$

$$b_o = [0.1456789 \quad 0.3678901 \quad -0.1456789 \quad 0.2789123] \quad (29)$$

Pada langkah ini nilai W_o dan b_o dijabarkan dengan angka random, sedangkan vsriabel lainnya masih menggunakan nilai yang sama seperti pada langkah sebelumnya. Rumus untuk menghitung kandidat output gate dilihat sebagai berikut

$$O_t = \sigma(W_o . [h_{t-1} + x_t] + b_o) \quad (30)$$

Setelah memasukkan semua angka yang telah dijabarkan ke dalam rumus, maka hasil perhitungan matriks dari nilai output gate yaitu :

[0.53702801 0.59649867 0.45898086 0.572123]

2.5.6. Menghitung output final

Setelah hidden state dihitung, langkah terakhir adalah menghasilkan output akhir yang menentukan apakah input memiliki sentimen positif atau negatif. Variabel yang digunakan dijabarkan sebagai berikut :

C_t = [0.15349753 0.08459456 0.16110849 −0.09973599] (31)

O_t = [0.53702801,0.59649867,0.45898086,0.572123] (32)

Nilai cell state dan nilai output gate diambil dari hasil perhitungan sebelumnya, kemudian nilai tersebut dimasukkan ke dalam rumus perhitungan output final sebagai berikut :

$h_t = O_t . \tanh(Ct)$ (33)

Setelah memasukkan semua angka yang telah dijabarkan ke dalam rumus, maka hasil perhitungan matriks dari nilai output final yaitu :

h_t= [0.0817911 0.05034052 0.07331251 −0.0568728]

Setelah hidden state (h_t) dihitung, langkah selanjutnya dalam kasus klasifikasi sentimen adalah menggunakan hidden state ini sebagai input ke lapisan dense (fully connected) untuk menghasilkan prediksi akhir, yang kemudian akan menentukan apakah kalimat tersebut termasuk dalam sentimen positif atau negatif. Pada lapisan ini, bobot tambahan (W_{dense}) akan dikalikan dengan hidden state, dan hasilnya dijumlahkan dengan bias (b_{dense}):

$$y_{logits} = W_{dense} . h_t + b_{dense} \quad (34)$$

Di sini, y_{logits} adalah output sebelum aktivasi, yang kemudian akan diteruskan ke fungsi softmax atau sigmoid untuk menghasilkan nilai probabilitas bagi setiap kelas (positif atau negatif). Misalkan bobot untuk fully connected layer W_{dense} dan bias b_{dense} diberikan seperti berikut (dengan contoh nilai):

W_{dense}= [0.2,−0.3,0.5,−0.4], b_{dense}=0.1, maka

y_{logits} = (0.2 · 0.0817911) + (−0.3 · 0.05034052) + (0.5 · 0.07331251) + (−0.4 · (−0.0568728)) + 0.1

Setelah mendapatkan y_{logits}, kita perlu menggunakan fungsi aktivasi untuk mengubah nilai tersebut menjadi probabilitas. Untuk klasifikasi biner (positif/negatif), biasanya digunakan fungsi sigmoid:

$$\gamma = sigmoid(y_{logits}) = \frac{1}{1+e^{-y_{logits}}} \quad (35)$$

Jika $\gamma > 0$, maka hasil klasifikasinya adalah positif, dan jika $\gamma \leq 0$, hasilnya adalah negatif.

Hasil perhitungan menunjukkan bahwa nilai logit output yang diperoleh adalah 0.1607. Setelah melalui fungsi aktivasi sigmoid, nilai probabilitas yang dihasilkan adalah 0.5401. Karena probabilitas ini lebih besar dari 0.5, maka model mengklasifikasikan kalimat ″pilihan mudah harga murah″ sebagai sentimen positif.

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v18i1.xxxx

3. Hasil dan Pembahasan

Hasil penelitian dapat dilihat pada tabel 4

Tabel 4 Hasil penelitian

Vector Size	Data	Accuracy	Class	Precision	Recall	F1-score
100	4073	0.85	Negative	0.54	0.90	0.64
			Positive	0.97	0.82	0.89
200	4073	0.86	Negative	0.58	0.90	0.71
			Positive	0.97	0.86	0.91
300	4073	0.82	Negative	0.50	0.95	0.66
			Positive	0.99	0.79	0.88
400	4073	0.89	Negative	0.74	0.89	0.79
			Positive	0.96	0.83	0.80
400	126989	0,83	Negative	0.50	0.95	0.66
			Positive	0.97	0.79	0.88

Penelitian ini menggunakan dua set data untuk pelatihan: pertama, seluruh 12.689 data dengan 10.562 sentimen positif dan 2.127 sentimen negatif; kedua, subset 4.073 data dari tiga bulan terakhir, terdiri dari 3.289 sentimen positif dan 784 sentimen negatif. Hasil pengujian menunjukkan bahwa model yang dilatih dengan 4.073 data mencapai akurasi tertinggi sebesar 89%, sedangkan penggunaan seluruh 12.689 data hanya menghasilkan akurasi 85%.

Selain jumlah data, ukuran vektor Word2Vec juga mempengaruhi hasil akurasi. Pada pengujian dengan 4.073 data, model dengan *vector size* 100 hanya mencapai akurasi 84%, sedangkan *vector size* 400 memberikan akurasi tertinggi 89%. Namun, peningkatan ukuran vektor tidak selalu menjamin kenaikan akurasi, terbukti dengan hasil *vector size* 300 yang justru menurun dibandingkan dengan ukuran 200. Ketidakseimbangan data juga berdampak signifikan terhadap akurasi model, sehingga jumlah data pelatihan perlu disesuaikan agar tidak memperparah ketidakseimbangan kelas. Untuk pelatihan Word2Vec, penggunaan data dari ulasan Traveloka terbukti lebih efektif dibandingkan dengan korpus besar dari sumber lain seperti Wikipedia.

Pengujian model dengan komentar terbaru menunjukkan adanya beberapa kesalahan analisis, kemungkinan disebabkan oleh kata-kata baru yang tidak ada dalam pelatihan (*Out of Vocabulary* - OOV). Faktor lain yang mungkin berkontribusi adalah jumlah data yang masih terbatas, kurang beragam, serta banyaknya ulasan yang hanya terdiri dari satu atau dua kata. Hal ini mengindikasikan bahwa meskipun Word2Vec dan LSTM memberikan hasil yang cukup baik, model masih dapat ditingkatkan dengan data yang lebih seimbang, lebih besar, serta teknik untuk menangani kata-kata yang tidak dikenal.

4. Kesimpulan dan Saran

Penelitian ini menganalisis sentimen ulasan pelanggan menggunakan model LSTM (Long Short-Term Memory) yang dikombinasikan dengan representasi kata Word2Vec. Dataset yang digunakan terdiri dari 12.689 ulasan pengguna Traveloka di Google Play Store, dengan 10.562 ulasan positif dan 2.127 ulasan negatif. Setelah dilakukan pengujian pada seluruh data dan subset sebanyak 4.073 data tanpa penyeimbangan kelas, hasil menunjukkan bahwa penggunaan
4.073 ulasan dengan 3.289 sentimen positif dan 784 sentimen negatif memberikan performa optimal. Model LSTM yang dilatih dengan Word2Vec mencapai akurasi tertinggi sebesar 89% saat menggunakan vector size 400, dibandingkan dengan vector size 100, 200, dan 300 yang masing-masing hanya mencapai akurasi 85%, 86%, dan 82%. Hasil ini menunjukkan bahwa

jumlah dimensi vektor berpengaruh terhadap akurasi model, terutama dalam analisis sentimen berbahasa Indonesia yang masih menghadapi keterbatasan dalam pemrosesan bahasa alami dibandingkan dengan bahasa Inggris. Untuk penelitian selanjutnya, disarankan menggunakan data yang lebih seimbang, menambahkan corpus guna mengatasi permasalahan *out of vocabulary*, serta mempertimbangkan metode yang dapat menangani analisis sentimen bertingkat.

Daftar Pustaka

[1] Abraham B. Nomleni, Maria M. Sakunab, Fransiskus Moda, Gaudensius Djuang, & Apryanus Fallo. (2023). Pengaruh Harga, Promosi Dan Gaya Hidup Terhadap Keputusan Pembelian Pada Tiket.Com. *ORGANIZE: Journal of Economics, Management and Finance*, *2*(2), 97–106. https://doi.org/10.58355/organize.v2i2.20

[2] Kusumadewi, R., Fitriadi, H., Sari, A. F. K., Zulfkar, A. L., Islami, P. Y. N., Amrina, D. H., ... & Adab, P. (2023). *Perkembangan Ekonomi Kreatif & Ekonomi Industri Berbasis Digital*. Penerbit Adab. S. P. Tahalea and A. SN, "Central Actor Identification of Crime Group using Semantic Social Network Analysis," *Indones. J. Inf. Syst.*, vol. 2, no. 1, p. 24, 2019.

[3] Cesariana, C., Juliansyah, F., & Fitriyani, R. (2022). Model Keputusan Pembelian Melalui Kepuasan Konsumen Pada Marketplace. *Jurnal Manajemen Pendidikan Dan Ilmu Sosial*, *3*(1), 211–224.

[4] Xu, W., Chen, J., Ding, Z., & Wang, J. (2024). Text sentiment analysis and classification based on bidirectional Gated Recurrent Units (GRUs) model. *arXiv preprint arXiv:2404.17123*.

[5] Jiang, H., Hu, C., & Jiang, F. (2022). Text Sentiment Analysis of Movie Reviews Based on Word2Vec-LSTM. *2022 14th International Conference on Advanced Computational Intelligence, ICACI 2022*, 129–134. https://doi.org/10.1109/ICACI55529.2022.9837505

[6] Nawangsari, R. P., Kusumaningrum, R., & Wibowo, A. (2019). Word2vec for Indonesian sentiment analysis towards hotel reviews: An evaluation study. *Procedia Computer Science*, *157*, 360–366. https://doi.org/10.1016/j.procs.2019.08.178

[7] Noh, S. H. (2021). Analysis of gradient vanishing of RNNs and performance comparison. *Information (Switzerland)*, *12*(11). https://doi.org/10.3390/info12110442