JTERA (Jurnal Teknologi Rekayasa), Vol. 7, No. 1, Juni 2022, Hal. 39-46
Terakreditasi "**Peringkat 3**" oleh Kemenristek/BRIN, Nomor SK: 85/M/KPT/2020
DOI: 10.31544/jtera.v7.i1.2022.39-46
p-ISSN 2548-737X
e-ISSN 2548-8678

Analisis Sentimen pada Data Ulasan Twitter dengan *Long-Short Term Memory*

Sharfina Febbi Handayani[#], Riszki Wijayatun Pratiwi, Dairoh, Dwi Intan Af'idah
Teknik Informatika, Politeknik Harapan Bersama Tegal
Jalan Mataram No 9, Kota Tegal, Jawa Tengah 52147, Indonesia
[#]sharfina.handayani@poltektegal.ac.id

Abstrak

Perkembangan media sosial telah memudahkan masyarakat dalam menyebarluaskan informasi. Salah satu bentuk informasi yang dimaksud berupa kebebasan dalam menyampaikan opini di media sosial. Perkembangan penelitian terkait analisis sentimen pada data ulasan teks bertujuan untuk mengetahui polaritas opini di media sosial yang telah mengalami peningkatan. Salah satu metode yang diterapkan dalam analisis sentimen teks ulasan yaitu penggunaan metode *Long Short-Term Memory* (LSTM). Tujuan penelitian ini adalah untuk mengetahui performa model LSTM terhadap berbagai sentimen ulasan teks Twitter berbahasa Indonesia. Proses pengujian dilakukan berdasarkan perhitungan dari nilai akurasi *tuning hyperparameter*. Pengujian akurasi penelitian ini menggunakan parameter *Word2Vec*, fungsi aktivasi, jumlah *epoch*, dan jumlah neuron. Hasil pengujian kinerja LSTM yang optimal dari penelitian ini diperoleh berdasarkan tuning arsitektur *Word2Vec Continuous Bag of Words* (CBOW) dengan akurasi 57,15%, *tuning* jumlah neuron sebanyak 150 menghasilkan nilai akurasi 57,35%, tuning jumlah *epoch* sebesar 30 menghasilkan nilai akurasi 57,40%, serta tuning fungsi aktivasi *softmax* menghasilkan nilai akurasi sebesar 57,35%.

Kata kunci: sentimen, ulasan, Twitter, LSTM, *Word2Vec*

Abstract

The development of social media has made it easier for people to disseminate information. One form of information in question is the freedom to express opinions on social media. The development of research related to sentiment analysis on text review data aims to determine the polarity of opinion on social media which has increased. One of the methods applied in the sentiment analysis of the review text is the use of the Long Short-Term Memory (LSTM) method. The purpose of this study was to determine the performance of the LSTM model on various sentiments of reviews of Indonesian-language Twitter texts. The testing process is carried out based on the calculation of the hyperparameter tuning accuracy value. Testing the accuracy of this study using Word2Vec parameters, activation function, number of epochs, and number of neurons. The optimal LSTM performance test results from this study were obtained based on tuning the Word2Vec Continuous Bag of Words (CBOW) architecture with an accuracy of 57.15%, tuning the number of neurons as much as 150 producing an accuracy value of 57.35%, tuning the number of epochs at 30 producing an accuracy value of 57. .40%, and tuning the softmax activation function produces an accuracy value of 57.35%.

Keywords: *sentiment, reviews, Twitter, LSTM, Word2Vec*

I. PENDAHULUAN

Salah satu layanan internet yang mengalami peningkatan dan popularitas adalah penggunaan media sosial. Hal ini dibuktikan dengan penggunaan media sosial telah berhasil mengubah cara seseorang dalam mengekspresikan suasana hati dan pemikiran. Melalui media sosial, masyarakat mudah dalam memberikan komentar, mengungkapkan sebuah opini, melakukan percakapan secara daring, dan dapat dengan leluasa menyebarkan informasi. Data yang dihasilkan media sosial semakin bertambah seiring bertambahnya jumlah pengguna. Masyarakat dapat memanfaatkan data yang dihasilkan media sosial untuk tujuan tertentu, salah satunya mendapatkan ulasan dari media sosial tersebut. Penanganan terkait pengolahan informasi dan opini membutuhkan algoritma dan program tertentu untuk mengolah dan menganalisis data terkait opini

Diterima: 28 November 2021; Direvisi: 19 Mei 2022; Disetujui: 27 Mei 2022
JTERA, Vol. 7, No. 1, Juni 2022

pengguna media sosial yang dikenal dengan istilah analisis sentimen [1].

Analisis sentimen dapat diwujudkan dengan melakukan analisis opini, sikap, evaluasi, sentimen, sikap serta emosi seseorang dari bahasa dan tulisan. Analisis sentimen dilakukan untuk menilai ulasan terhadap suatu objek apakah cenderung bernilai positif atau negatif [2]. Sedangkan menurut penelitian lain, analisis sentimen bisa dimengerti pada beberapa level, yaitu level dokumen, paragraf, kalimat, atau klausa. Adapun level yang terdapat pada sumber datanya membagi analisis sentimen tersebut menjadi dua kategori, yakni level *fine-grained sentiment analysis*, dan *coarse-grained analysis sentiment* [3].

Analisis sentimen merupakan salah satu cabang *text mining* yang membahas analisis dokumen teks [4]. Analisis sentimen dapat juga didefinisikan sebagai aktivitas untuk mengevaluasi pendapat atau komentar masyarakat dengan cara menuliskan subjek yang terkait dengan topik lainnya [5]. Sumber data yang banyak digunakan dalam kajian analisis sentimen adalah media sosial Twitter. Hal ini disebabkan Twitter memiliki struktur yang sesuai untuk memudahkan analisis, sehingga pada penelitian ini sumber *dataset* yang digunakan penulis berupa data ulasan teks berbahasa Indonesia yang berasal dari Twitter [6]. Twitter memberikan kemudahan pengguna untuk dapat mengekspresikan segala sesuatu yang terjadi dan memberikan kemudahan ketika mengakses data Twitter dengan API Twitter [7].

Deep learning merupakan salah satu satu teknik dalam *machine learning* yang berguna dalam melakukan proses ektraksi fitur, klasifikasi, dan pengenalan pola yang memanfaatkan banyak layer pada pengolahan informasi non-linier [8]. Penggunaan teknik *deep learning* pada analisis sentimen teks berbahasa Indonesia telah dilakukan beberapa peneliti dengan metode *Long Short-Term Memory* (LSTM). Metode LSTM sebelumnya telah digunakan dalam penelitian [9] dan menghasilkan akurasi hasil analisis sentimen lebih baik dibanding menggunakan metode konvensional [10].

Kajian tentang proses analisis sentimen dengan menerapkan metode LSTM telah dilakukan pada penelitian sebelumnya [11] dan penelitian yang membandingkan 3 dataset berbeda pada JD.com, CTRIP, dan ulasan film berbahasa Inggris dengan metode LSTM dan *Recurrent Neural Network* (RNN) [9]. Hasil pengujian pada penelitian menunjukkan bahwa penggunaan metode LSTM menghasilkan akurasi yang lebih baik dibanding dengan metode RNN. Penggunaan metode LSTM juga telah menghasilkan *error rate* terbaik dengan persentase 14,3% untuk *dataset* SSTb dan 11,32% untuk dataset IMDB dibandingkan dengan metode tradisional.

Pada literatur sebelumnya terdapat penelitian RNN dalam melakukan klasifikasi sentimen terhadap *dataset Chinese microblog* pada NLPCC2013. *Dataset* tersebut dilakukan representasi korpus data ke dalam bentuk vektor dengan *Continuous Bag of Word* (CBOW). Kemudian setelah dilakukan pengujian didapatkan bahwa hasil akurasi RNN lebih baik dengan *Naïve Bayes* dengan nilai akurasi RNN sebesar 68,29%, sedangkan dengan metode *Naïve Bayes* didapatkan akurasi 67,22%. Kemudian dari data tersebut dilakukan analisis sentimen dengan menggunakan *deep learning* LSTM dan *Dynamic Convolutional Neural Network* (DCNN) pada data Twitter. Hasilnya menunjukkan bahwa penggunaan metode *deep learning* lebih baik dibandingkan dengan menggunakan metode tradisional [12].

Penelitian lain yang mengkaji metode serupa dalam melakukan klasifikasi sentimen telah dilakukan pada data opini film berbahasa Indonesia yang berasal dari Twitter dengan metode DCNN dan diperoleh hasil bahwa akurasi dengan metode DCNN memiliki hasil analisis yang lebih baik dibanding dengan menggunakan metode *Naïve Bayes* [13]. Kesulitan dalam melakukan klasifikasi pada proses analisis sentimen dapat dilakukan menggunakan *deep learning* dengan melakukan kombinasi dari parameter *Word2Vec*. Hasil penelitian menunjukkan bahwa kombinasi 3 parameter dalam *Word2Vec* menghasilkan akurasi yang lebih baik pada arsitektur CBOW dengan menggunakan dimensi bernilai 100 dan *Hierarchical Softmax* [14].

Penelitian ini menerapkan pendekatan *deep learning* menggunakan metode LSTM dalam melakukan analisis sentimen pada data ulasan Tweet yang berbahasa Indonesia. Adapun untuk proses ekstraksi fitur pada penelitian ini menggunakan *Word2Vec* karena dapat mempelajari representasi vektor kata di ruang vektor dimensi tinggi dan menghitung jarak *cosine* antar kata sehingga model ini dapat menemukan hubungan semantik antar kata dalam dokumen [15].

Melalui penelitian ini penulis hendak mengkaji model *text mining* dengan menerapkan metode *deep learning* khususnya dengan metode LSTM pada sentimen data ulasan Twitter berbahasa Indonesia. Selain itu, melalui penelitian ini penulis juga hendak mengetahui tingkat akurasi penggunaan metode LSTM dalam melakukan klasifikasi teks.

II. METODE PENELITIAN

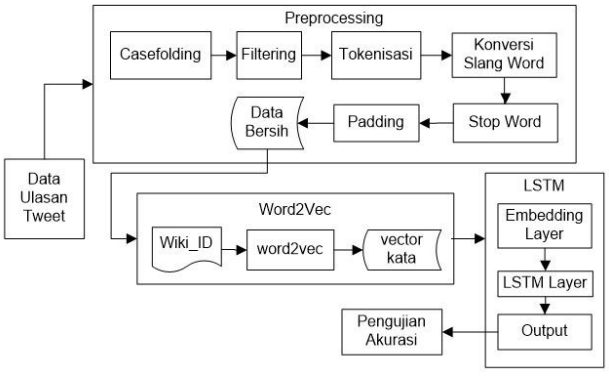
Penelitian ini menggunakan *dataset* ulasan Twitter dari penelitian sebelumnya. Tahapan penelitian diawali dengan menggunakan informasi yang sudah diperoleh dari penelitian tersebut [6]. Pada penelitian ini diusulkan rancangan model *attention based* LSTM pada data ulasan Twitter dengan menggunakan dua jenis data, yaitu data uji dan data latih. Data latih tersebut digunakan untuk mengklasifikasikan opini pada kelas sentimen dan data uji tersebut dibutuhkan untuk dapat mengukur sejauh mana *classifier* dapat berhasil mengklasifikasi dengan benar.

A. Preprocessing

Penelitian ini dimulai dari *preprocessing* data yang terbagi menjadi beberapa tahapan meliputi *casefolding*, *filtering*, tokenisasi, dan konversi *slang word*. Tahapan selanjutnya setelah proses *preprocessing* adalah *Word2Vec* yang bertujuan untuk mengkonversi kata menjadi vektor serta dilanjutkan dengan tahapan implementasi model dan pengujian model menggunakan *tuning hyperparameter*. Tahapan penelitian disajikan pada Gambar 1.

Tahapan *preprocesing* yaitu mengolah data ulasan agar lebih mudah dalam melakukan proses selanjutnya. Pada *preprocessing* terdapat beberapa tahapan yang harus dilakukan yaitu *casefolding*, *filtering*, tokenisasi, dan konversi *slangword*. Tahapan pertama *preprocessing* dimulai dengan *casefolding* yang dapat membuat teks konsisten dalam penggunaan huruf. Tahapan ini bertujuan untuk mengubah dari karakter sebuah huruf yang ada di dalam komentar menjadi sebuah karakter huruf yang kecil semua.

Tahapan kedua dalam *preprocessing* adalah *filtering*. Pada proses ini dilakukan penyesuaian untuk menghilangkan dari karakter yang khusus dan ulasan seperti tanda karakter lainnya ($, %, *, dan sebagainya). Proses ini juga menghilangkan kata yang tidak sesuai pada hasil *parsing* contohnya *username* yang diawali disimbol ”@”, hastag ”#”, *Uniform Resource Locator* (URL), dan *emoticon*. Tanda/simbol atau angka dihilangkan karena tidak banyak berpengaruh pada penentuan label.



Gambar 1. Tahapan penelitian

Tahapan ketiga adalah proses tokenisasi yang bertujuan untuk memecahkan ulasan menjadi satu kata. Tokenisasi dilakukan dengan melihat setiap spasi yang ada dalam ulasan yang selanjutnya dari spasi tersebut dihasilkan kata-kata yang dapat dipisahkan.

Setelah tokenisasi maka tahapan selanjutnya merupakan konversi *slang word* berupa proses mengubah kata tidak baku menjadi kata baku. Tahap ini dilakukan dengan menggunakan bantuan kamus *slang word* dan padanan di dalam kata baku. Pada tahap ini dilakukan pemeriksaan kata, baik kata yang ada dalam kamus *slang word* atau kata yang tidak ada dalam kamus *slang word*. Apabila terdapat kata tidak baku dalam kamus *slang word* maka kata tidak baku diubah menjadi kata baku yang terdapat dalam kamus *slang word*. Adapun hasil pengelompokkan kelas sentimen terhadap kelas emosi positif, netral dan negatif disajikan pada Tabel 1.

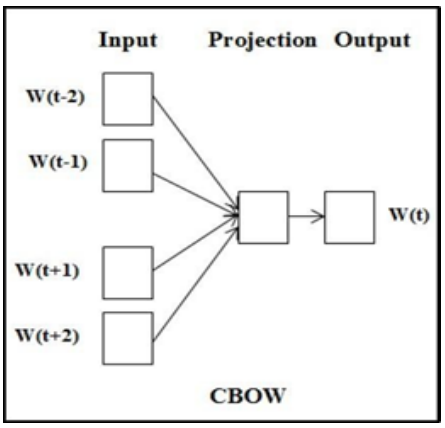
B. Arsitektur Word2Vec

Word2Vec adalah syaraf tiruan yang mempelajari representasi terdistribusi kata. Vektor merupakan kata terdistribusi yang kuat dan bisa digunakan dalam memprediksi kata serta terjemahan [16]. Representasi vektor dari pembelajaran kata dengan menggunakan model *Word2Vec* telah terbukti membawa makna semantik dan berguna dalam berbagai tugas *Natural Languange Processing* (NLP) [17]. Pada penelitian sebelumnya [10] telah disampaikan bahwa *Word2Vec* mempunyai dua lapisan jaringan syaraf yang memproses teks. *Input* adalah *corpus* dan *output*-nya berupa satu set vektor. Satu set vektor yang dimaksud yaitu fitur vektor untuk kata-kata dalam *corpus* tersebut.

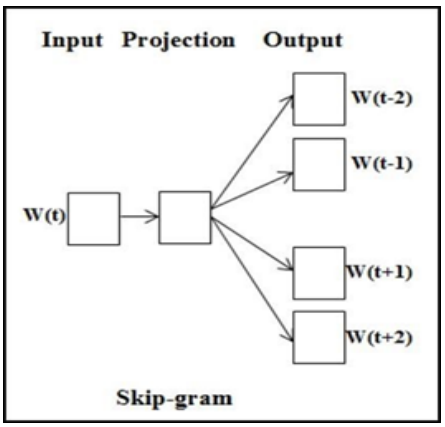
Adapun penelitian lain yang dilakukan [18] *Word2vec* mempunyai dua buah arsitektur yang berbeda yaitu CBOW dan *skip-gram.*

Tabel 1. Kelas sentimen

Teks	Kelas emosi
Jika saya berjuang terus hingga akhirnya saya pasti menemukan jalannya	Positif
Apakah sudah templatenya kalau udah jadi mamah mamah tuh kalau ada bagi2 hadiah atau apapun bilang aku satu lagi dong	Netral
Aku baru lihat trailer tapi otak kiriku tidak mudeng	Negatif



Gambar 2. Arsitektur CBOW



Gambar 3. Arsitektur *skip-gram*

Tujuan pada metode CBOW untuk memprediksi kata yang diberikan oleh kata-kata di sekitarnya. Arsitektur CBOW ketika memprediksi kata pada konteks diilustrasikan pada Gambar 2. Berbeda dengan CBOW, *skip-gram* justru menghasilkan jendela kata yang diberikan oleh kata tunggal. Ilustrasi arsitektur *skip-gram* dalam memprediksi kata yang diberikan oleh kata di sekitarnya disajikan pada Gambar 3.

Kedua arsitektur *Word2Vec* tersebut menggunakan jaringan syaraf tiruan sebagai algoritma. Awalnya setiap kata yang terdapat kosakata merupakan vektor acak N dimensi. Selama proses *training* data, lapisan *input* memiliki banyak neuron karena mewakili kata-kata yang terdapat dalam kosakata. Ukuran *hidden layer* disesuaikan jumlahnya sesuai dengan dimensi vektor kata yang nantinya dihasilkan. Ukuran lapisan *output*-nya mempunyai ukuran yang sama dengan lapisan *input*.

Diasumsikan bahwa V merupakan jumlah kosakata yang digunakan untuk pembelajaran vektor kata dan N merupakan dimensi vektor kata, koneksi dari lapisan *input* ke *hidden layer* dapat direpresentasikan oleh matriks W_1 yang mempunyai ukuran V x N dengan setiap baris mewakili sebuah kata dari kosakata. Melalui cara yang sama, koneksi dari *hidden layer* ke lapisan *output* dapat direpresentasikan oleh matriks W_0 dengan ukuran N x V.

Dalam hal ini, setiap kolom dari matriks W_0 merupakan kata dari kosakata yang diberikan. *Input* yang diberikan ke jaringan dikodekan menggunakan representasi "1-out of-V" yang berarti bahwa hanya satu baris *input* yang nilainya bernilai satu dan sisanya dari jalur input diatur ke nol [4]. *Output* pada neuron *hidden layer* dapat dihitung dengan (1).

$$H = x * W_1 \quad (1)$$

Output dari neuron tersembunyi yaitu vektor H meniru bobot dari baris kedua matriks dari W_1 hal ini disebabkan *input* yang diberikan ke jaringan dikodekan menggunakan representasi "1-out of-V" yang berarti bahwa hanya satu baris masukan yang nilainya bernilai satu dan sisanya dari jalur *input* diatur ke nol. Jadi fungsi dari input ke koneksi lapisan tersembunyi pada dasarnya dengan menyalin vektor kata input ke *hidden layer*. *Output* dari lapisan tersembunyi diperoleh dari (2).

$$O = H * W_0 \quad (2)$$

Keterangan:
H : *hidden layer*
X : *input neuron* sebelumnya
W_1 : matriks
W_0 : matriks
O : *output*

Selanjutnya untuk menghasilkan probabilitas kata-kata, *Word2Vec* menggunakan fungsi *softmax* pada lapisan *output* sehingga probabilitas untuk delapan kata-kata dalam *corpus* dapat dihitung menggunakan persamaan (3).

$$y_k = P_r(kata_k | kata_{contex}) = \frac{\exp(k)}{\sum m \exp(n)} \quad (3)$$

Keterangan:
y_k : probabilitas kata
$kata_k$: kata pada *corpus*
$kata_{contex}$: target kata pada *corpus*
k : nilai *output* layer target
n : nilai *output* layer

Input dari *Word2Vec* pada penelitian ini adalah kalimat opini berbahasa Indonesia yang telah melalui tahap tokenisasi dan kalimat tersebut diproses jadi input vektor menggunakan *Word2Vec*. *Word2Vec* bertujuan untuk mempelajari pemetaan

setiap kata pada *vocabulary* ke dimensi vektor. Pada tahap ini, *Word2Vec* digunakan dalam memproses teks. *Input* berupa *corpus* kata dari hasil *preprocessing* dan *output* yang dihasilkan berupa satu set vektor.

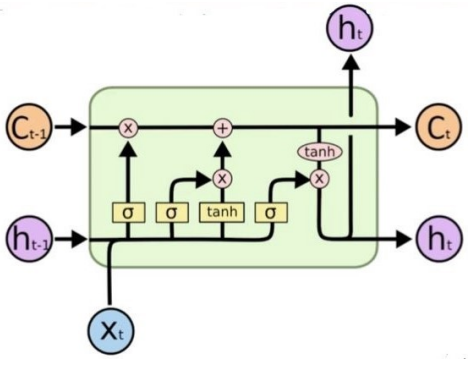
Pertama *corpus* diambil data dari Wikipedia Bahasa Indonesia yang diubah menjadi format teks. Lalu *corpus* tersebut akan di-*training* menggunakan *Word2Vec*. Setiap kata pada model ini diwakili oleh vektor dari m-dimensi. Lalu data ulasan yang sudah melalui proses *preprocessing* harus dikonversi ke dalam format numerik.

Data ulasan yang sudah dibagi dalam proses token direpresentasikan pada model *Word2Vec*. Jika token dapat ditemukan pada model *Word2Vec* (memiliki representasi vektor pada sebuah kata), sehingga digunakan vektor *Word2Vec* tersebut untuk mewakili token. Namun, jika token tidak ditemukan pada model maka akan diganti dengan vektor nol. Kemudian *input* akan menjadi sebuah vektor dimensi n x m dengan n jumlah kata maksimum pada ulasan dan m adalah dimensi vektor kata.

C. *Long Short-Term Memory* (LSTM)

LSTM merupakan varian RNN yang dapat mengingatkan sebuah informasi jangka panjang dan menggantikan simpul RNN pada *hidden layer* dengan sel yang dirancang untuk menyimpan informasi. LSTM mempunyai tiga gerbang (*gate*) yang terdiri dari *forget gate*, *input gate*, dan *output gate* untuk mengendalikan penggunaan dan *update* informasi teks terdahulu. Sel memori dan tiga gerbang dirancang untuk memungkinkan LSTM membaca, menyimpan, dan memperbaharui informasi [19]. Arsitektur LST) diilustrasikan pada Gambar 4.

Berbeda dengan arsitektur RNN konvensional yang tidak mampu mengingat hal-hal yang telah terjadi di jaringan dalam jangka waktu yang lama, *memory cell* LSTM memungkinkan penyimpanan data dan pengaksesan informasi dalam jangka waktu yang lebih lama sehingga LSTM dikembangkan sebagai solusi dari masalah *vanishing gradient* yang biasa ditemui pada RNN konvensional. Gradien yang semakin mengecil sampai layer terakhir sehingga nilai bobot tidak berubah menyebabkan proses *training* tidak pernah konvergen. Sebaliknya gradien yang semakin membesar menyebabkan nilai bobot pada beberapa *layer* juga membesar sehingga algoritma optimasi menjadi divergen (*exploding gradients*).



Gambar 4. Arsitektur LSTM

Pada literatur [20] disebutkan bahwa langkah utama pada LSTM adalah menentukan apakah informasi dari input X_{t-1} dan X_t bisa melewati *cell state* atau tidak. Kemudian keputusan tersebut dibuat dengan lapisan *sigmoid* yang disebut "*forget gate*". Jika keputusan menghasilkan angka 1 berarti "boleh lewat" dan jika menghasilkan angka 0 artinya "lupakan informasi". Nilai *forget gate* (f_t) dapat dihitung melalui persamaan, seperti pada (4).

$$f_t = \sigma\,(w_f[h_{t-1}, x_t\,] + b_t) \tag{4}$$

Langkah selanjutnya yaitu menentukan informasi baru yang disimpan di *cell state*. Lapisan pertama *sigmoid* yang disebut *input gate* menentukan bagian mana yang akan diperbarui. Kemudian lapisan *tanh* yang membuat vektor nilai kandidat baru, C_t dapat ditambahkan pada *cell state*. Setelah itu keduanya dapat digabungkan untuk *update state*. Nilai *input gate* dapat dihitung seperti pada (5) dan nilai kandidat baru dapat dihitung seperti pada (6).

$$i_t = \sigma\,(w_i[h_{t-1}, x_t\,] + b_i) \tag{5}$$

$$\check{C}_t = tanh\,(w_C[h_{t-1}, x_t\,] + b_C) \tag{6}$$

Setelah menghitung nilai *input gate* kemudian langkah selanjutnya memperbarui *cell state* lama, C_{t-1} ke *cell state* yang baru C_t lalu mengalihkan *cell state* lama dengan *forget gate* f_t kemudian ditambah $i_t * \check{C}_t$. Penjelasan ini dapat dilihat pada (7).

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{7}$$

Pada *output gate*, lapisan *sigmoid* menentukan sel mana yang akan menjadi *output* (h_t), lalu tempatkan *cell state* melalui *tanh* dan memperbanyak *output* dari *sigmoid gate* (O_t), sehingga hanya bagian yang ditentukan yang menjadi *output*. Perhitungan *output gate* dapat dilihat pada (8) dan (9).

$$O_t = \sigma\,(W_0[h_{t-1}, x_t] + b_0 \tag{8}$$

$$h_t = O_t * tanh(C_t) \tag{9}$$

Implementasi LSTM pada penelitian ini dibuat menggunakan bahasa pemrograman *python* yang selanjutnya dievaluasi untuk mengukur performa klasifikasi dokumen terhadap satu kelas atau lebih. Akurasi merupakan ukuran dari seberapa baik model dapat mengklasifikasikan dokumen dengan benar. Nilai akurasi yang tinggi dapat diperoleh saat banyak data yang berhasil diklasifikasikan dengan benar sesuai kelas sentimen.

III. HASIL DAN PEMBAHASAN

Tahap ini merupakan tahap pengujian untuk mengetahui performa sistem yang sudah dibangun menggunakan pengujian akurasi. Berdasarkan hasil pengujian akurasi yang dilakukan maka dapat diketahui parameter yang menghasilkan nilai akurasi terbaik. Pada penelitian ini data tweet yang digunakan berjumlah 10.806 Tweet yang diperoleh dari penelitian sebelumnya [6].

Data tersebut dikelompokkan dalam tiga polaritas sentimen, yaitu sentimen negatif, sentimen positif, dan sentimen netral. Kemudian data tweet dibagi ke dalam data uji dan data latih. Pada pengujian dilakukan *tuning parameter* untuk melihat performa terbaik yang dihasilkan oleh klasifikasi tweet menggunakan LSTM. Pada penelitian ini parameter yang diujikan adalah jumlah neuron, fungsi aktivasi, jumlah *epoch* dan *Word2Vec*. Masing-masing nilai parameter yang digunakan untuk pengujian ini disajikan pada Tabel 2.

Tahap pengujian dimulai dengan pengujian *Word2Vec*, yaitu arsitektur CBOW dan *Skip-gram*. Pengujian ini dilakukan untuk melihat arsitektur *word embedding* terbaik dari model *Word2Vec*. Proses ini dilakukan menggunakan metode *attention based LSTM*. Arsitektur yang menghasilkan akurasi optimal dari model *Word2Vec* akan digunakan pada pengujian selanjutnya.

Berdasarkan Tabel 3 dapat diperoleh hasil bahwa arsitektur CBOW memiliki nilai akurasi lebih baik dari pada arsitektur *skipgram*. Hal ini karena *Word2Vec* dengan arsitektur CBOW dapat menghasilkan *word embedding* yang lebih baik karena mampu memperhatikan makna semantik dari setiap kata yang ada. Selain itu, arsitektur CBOW juga cocok digunakan untuk data dengan jumlah besar. Sehingga CBOW dapat menghasilkan akurasi yang lebih baik.

Pengujian selanjutnya dilakukan terhadap jumlah neuron pada *hidden layer*. Pengujian ini perlu dilakukan untuk mengetahui jumlah neuron yang optimal. Melalui asumsi bahwa dengan penambahan jumlah neuron maka hasil dari akurasi akan semakin baik, dan sebaliknya dengan pengurangan jumlah neuron maka hasil akurasi akan menurun. Pada penelitian ini jumlah neuron yang akan diujikan adalah 100, 150, 200. Pengujian jumlah neuron dilakukan menggunakan metode LSTM untuk melihat pengaruh jumlah neuron terhadap akurasi. Seperti pada pengujian sebelumnya, pada pengujian jumlah neuron ini nilai parameter lain akan disamakan. Hasil pengujian jumlah neuron disajikan pada Tabel 4.

Berdasarkan Tabel 4 bahwa jumlah neuron 150 memiliki hasil akurasi optimal yaitu 57,35%. Menurut tuning jumlah neuron yang dilakukan, semakin banyak jumlah neuron maka semakin lama pula waktu yang dibutuhkan. Sedangkan, banyaknya jumlah neuron tidak menjamin dapat meningkatan nilai akurasi yang signifikan.

Selama pengujian belum ditemukan formula khusus yang dapat menentukan jumlah neuron yang optimal untuk peningkatan akurasi pada model yang dibuat. Penentuan jumlah *neuron* tergantung pada saat pengujian dilakukan, setelah melakukan percobaan maka akan didapatkan jumlah neuron dengan hasil akurasi terbaik.

Selanjutnya setelah menentukan jumlah neuron, dilakukan pengujian parameter jumlah *epoch* yaitu dengan menentukan jumlah *epoch* yang akan diujikan. Pada penelitian ini jumlah *epoch* yang diujikan adalah 10, 20, 30 sedangkan jumlah neuron

Tabel 2. Nilai parameter pengujian

Parameter	Nilai
Word2Vec	{CBOW, *Skip-gram*}
Jumlah neuron	{100, 150, 200}
Jumlah *epoch*	{10, 20, 30}
Fungsi aktivasi	{Softmax, Sigmoid}

Tabel 3. Pengujian *Word2Vec*

Arsitektur *Word2Vec*	Waktu (menit)	Akurasi (%)
CBOW	03:25	57,15
Skipgram	03:40	56,35

Tabel 4. Pengujian jumlah neuron

Jumlah neuron	Waktu (menit)	Akurasi (%)
100	02:41	54,10
150	02:50	57,35
200	03:07	57,16

Tabel 5. Pengujian jumlah *epoch*

Jumlah *epoch*	Waktu (menit)	Akurasi (%)
10	01:48	56,12
20	02:35	55,87
30	03:20	57,40

Tabel 6. Pengujian fungsi aktivasi

Fungsi aktivasi	Waktu (menit)	Akurasi (%)
Softmax	02:55	57,35
Sigmoid	02:42	32,28

adalah 200 dan arsitektur *Word2Vec* CBOW. Hasil pengujian parameter jumlah *epoch* pada LSTM disajikan pada Tabel 5.

Pada Tabel 5 dihasilkan bahwa pengujian *epoch* tertinggi yaitu pada jumlah *epoch* 30 dengan nilai akurasi sebesar 57,40%. Banyaknya jumlah *epoch* akan mempengaruhi hasil akurasi, akan tetapi peningkatan akurasi tidak signifikan. Sama halnya dengan pengujian neuron, semakin banyak jumlah *epoch* semakin lama waktu yang dibutuhkan untuk proses.

Sedangkan untuk pengujian parameter selanjutnya yang digunakan adalah pengujian fungsi aktivasi yang bertujuan untuk mencari nilai akurasi terbaik. Pada penelitian ini fungsi aktivasi yang akan diujikan adalah fungsi *softmax* dan *sigmoid.* Adapun untuk parameter lainnya diambil dari nilai parameter yang menghasilkan akurasi terbaik pada masing-masing pengujian sebelumnya yaitu arsitektur CBOW, jumlah neuron 150, jumlah *epoch* 30. Hasil pengujian aktivasi disajikan dalam Tabel 6.

Hasil pengujian yang telah dilakukan seperti Tabel 6 di atas menunjukan bahwa fungsi aktivasi *softmax* menghasilkan nilai akurasi terbaik yaitu 57,35% dibandingkan dengan fungsi aktivasi *sigmoid* dengan akurasi hanya 32,28%.

Berdasarkan hasil pengujian terhadap *dataset* yang digunakan dalam penelitian ini dihasilkan kenaikan akurasi dibandingkan dengan metode *machine learning* yang telah digunakan dalam penelitian [6]. Oleh karena itu, hasil akurasi LSTM dakam penelitian ini selaras dengan penelitian yang telah dilakukan dengan menggunakan [9] beberapa topik dataset yang berbeda.

IV. KESIMPULAN

Berdasarkan hasil pengujian, penggunaan metode LSTM dalam penelitian ini dengan klasifikasi teks ulasan twitter Berbahasa Indonesia menghasilkan empat parameter yang digunakan sebagai penentuan model performa yaitu tuning *Word2Vec*, tuning jumlah neuron, tuning jumlah epoch, dan tuning fungsi aktivasi dan hasil pengujian kinerja LSTM yang diperoleh berupa arsitektur *Word2Vec* yaitu CBOW, jumlah neuron 150, jumlah epoch 30, serta fungsi aktivasi *softmax* menghasilkan nilai akurasi sebesar 57,35%. Nilai akurasi yang dihasilkan di penelitian ini selaras dengan penelitian sebelumnya yang sudah menggunakan metode LSTM pada beberapa topik dataset yang berbeda.

UCAPAN TERIMA KASIH

Tim peneliti menyampaikan terima kasih pada Unit Pusat Penelitian dan Pengabdian Masyarakat (P3M) Politeknik Harapan Bersama Tegal melalui pembiayaan penelitian Hibah Institusi yang telah didapatkan kepada tim.

REFERENSI

[1] A. M. Ramadhani and H. S. Goo, "Twitter sentiment analysis using deep learning methods," *2017 7th Int. Annu. Eng. Semin.*, pp. 1–4, 2017.

[2] B. Liu, "Sentiment Analysis and Opinion Mining," *Synth. Lect. Hum. Lang. Technol.*, vol. 5, no. 1, pp. 1–167, May 2012.

[3] C. R. Fink, D. S. Chou, J. J. Kopecky, and A. J. Llorens, "Coarse- and fine-grained sentiment analysis of social media text," *Johns Hopkins APL Tech. Dig. (Applied Phys. Lab.*, vol. 30, no. 1, pp. 22–30, 2011.

[4] A. R. T. Lestari, R. S. Perdana, and M. A. Fauzi, "Analisis Sentimen Tentang Opini Pilkada DKI 2017 Pada Dokumen Twitter Berbahasa Indonesia Menggunakan Näive Bayes dan Pembobotan Emoji," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 1, no. 12, pp. 1718–1724, 2017.

[5] F. Nausheen and S. H. Begum, "Sentiment analysis to predict election results using Python," *2018 2nd Int. Conf. Inven. Syst. Control*, pp. 1259–1262, 2018.

[6] R. Ferdiana, F. Jatmiko, D. D. Purwanti, A. S. T. Ayu, and W. F. Dicka, "Dataset Indonesia untuk Analisis Sentimen," *J. Nas. Tek. Elektro dan Teknol. Inf.*, vol. 8, no. 4, p. 334, 2019.

[7] A. D. Rahmawan, Yohanes Suyanto, and S. Priyanta, "Analisis Emosi Pada Tweet Berbahasa Indonesia Tentang Ulasan Film," Universitas Gajah Mada, 2018.

[8] L. Deng and D. Yu, "Deep Learning: Methods and Applications," *Found. Trends Signal Process.*, vol. 7, no. 3–4, pp. 197–387, Jun. 2014.

[9] A. Hassan and A. Mahmood, "Deep learning for sentence classification," in *2017 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, 2017, pp. 1–5.

[10] M. A. Nurrohmat and A. SN, "Sentiment Analysis of Novel Review Using Long Short-Term Memory Method," *IJCCS (Indonesian J. Comput. Cybern. Syst.*, vol. 13, no. 3, p. 209, 2019.

[11] D. Li and J. Qian, "Text sentiment analysis based on long short-term memory," in *2016 First IEEE International Conference on Computer Communication and the Internet (ICCCI)*, 2016, pp. 471–475.

[12] Z. Yangsen, J. Yuru, and T. Yixuan, "Study of sentiment classification for Chinese Microblog

based on recurrent neural network," *Chinese J. Electron.*, vol. 25, no. 4, pp. 601–607, 2016.
[13] F. Ratnawati and E. Winarko, "Sentiment Analysis of Movie Opinion in Twitter Using Dynamic Convolutional Neural Network Algorithm," *IJCCS (Indonesian J. Comput. Cybern. Syst.*, vol. 12, no. 1, p. 1, 2018.
[14] D. Intan, S. F. Handayani, and R. W. Pratiwi, "Pengaruh Parameter Word2Vec terhadap Performa Deep Learning pada Klasifikasi Sentimen," vol. 6, no. 3, pp. 156–161, 2021.
[15] Z. Su, H. Xu, D. Zhang, and Y. xu, "Chinese sentiment classification using a neural network tool — Word2vec," 2014, pp. 1–6.
[16] A. S. Zharmagambetov and A. Pak, "Sentiment analysis of a document using deep learning approach and decision trees," *2015 Twelve Int. Conf. Electron. Comput. Comput.*, pp. 1–4, 2015.
[17] X. Rong, "word2vec Parameter Learning Explained," *CoRR*, vol. abs/1411.2, 2014.
[18] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *1st Int. Conf. Learn. Represent. ICLR 2013 - Work. Track Proc.*, pp. 1–12, 2013.
[19] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[20] C. Olah, "Understanding LSTM Networks," 2015. https://colah.github.io/posts/2015-08-Understanding-LSTMs/