JNATIA Volume 1, Nomor 1, November 2022

Jurnal Nasional Teknologi Informasi dan Aplikasinya

Penerapan *Long Short Term Memory* dalam Mengklasifikasi Jenis Ujaran Kebencian pada *Tweet* Bahasa Indonesia

Ni Putu Sintia Wati[a1], Cokorda Pramartha[b2]

[a]Program Studi Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Udayana
[b]Net-Centric Computing Laboratory, Universitas Udayana
Badung, Bali, Indonesia
[1]putu.sintia.wati@student.unud.ac.id
[2]cokorda@unud.ac.id

Abstract

Tweets are messages posted to Twitter and contain photos, videos, links, and text. Twitter is a social media service that allows everyone to communicate and stay connected through the rapid and frequent exchange of messages. However, as many communities have sprung up, user is getting less control while communicating on Twitter. One of them, more and more hate speech is being hurled either through retweets, or replies to each other in one of the threads belonging to a particular community. To minimize this impact, a classification is needed to find out whether the tweet contains hate speech or not before being uploaded to Twitter. Due to the rapid increase in the current data usage, it is necessary to review it with other methods to classify the data more deeply. Based on these problems, the method that can be used is Long Short Term Memory (LSTM). This study succeeded in providing predictions with different hyperparameter accuracy values for the LSTM application reaching 74%.

Keywords: *classification, LSTM, Tweet, hate speech*

1. Pendahuluan

Tweet atau cuitan merupakan pesan yang diposting ke Twitter dan berisi foto, video, tautan, serta teks. Sedangkan twitter merupakan layanan media sosial yang memungkinkan setiap orang untuk berkomunikasi dan tetap terhubung melalui pertukaran pesan yang cepat dan sering. Pengguna dapat memposting pesan ke profil mereka serta dapat dicari pada kolom pencarian. Dilansir oleh Varitey, selama tiga bulan terakhir tahun 2021, rata-rata pengguna aktif harian (*mean Daily Active User*) di seluruh dunia mencapai 217 juta atau naik 6 juta secara berurutan dari tahun ke tahun[1].

Namun, seiring banyaknya komunitas yang bermunculan membuat pengguna semakin tidak terkendali selama berkomunikasi di twitter. Salah satu diantaranya, semakin banyak ujaran kebencian yang dilontarkan baik itu melalui retweet, atau saling balas (*reply*) di salah satu utas milik komunitas tertentu. Ujaran Kebencian adalah tindakan komunikasi yang dilakukan oleh suatu individu atau kelompok dalam bentuk provokasi, hasutan, ataupun hinaan terhadap individu atau kelompok lain dalam hal berbagai aspek seperti ras, warna kulit, gender, cacat, orientasi seksual, kewarganegaraan, agama dan lain-lain[2]. Dalam penyampaiannya, ujaran kebencian juga sering disertai dengan menggunakan bahasa kasar.

Untuk meminimalisir dampak tersebut, diperlukan klasifikasi untuk mengetahui apakah tweet tersebut mengandung ujaran kebencian atau tidak, sebelum diunggah dalam twitter. Dikarenakan pesatnya peningkatan data saat ini, maka perlu ditinjau kembali dengan metode lain agar dapat mengklasifikasi data lebih dalam. Berdasarkan masalah tersebut, metode yang dapat digunakan

yaitu Long Short Term Memory (LSTM). LSTM adalah salah satu jenis *Recurrent Neural Network* (RNN) yang merupakan modifikasi dari RNN, dengan menambahkan *memory cell* agar dapat menyimpan informasi dalam jangka waktu yang lama. LSTM dapat meminimalisir terjadinya *vanishing gradient,* dimana kondisi nilai gradien pada input layer lebih kecil dari output layer.

2. Metode Penelitian

1. 2.1. Pengumpulan Data

Data yang digunakan dalam penelitian ini adalah data sekunder. Data sekunder merupakan data yang telah tersedia sebelum peneliti melakukan penelitian. Data yang akan digunakan bersumber dari Github yaitu multi-label hate speech and abusive language detection [3]. Data ini memiliki 13169 baris dengan 13 atribut dengan nilai 0 untuk nilai ‘tidak’ dan 1 untuk nilai ‘ya’. Selain itu, terdapat data yang memuat kata-kata penulisannya kurang tepat. Dokumen ini memiliki 15167 baris kata yang akan dinormalisasi serta data yang memuat kata-kata yang memuat *stopwords* dalam bahasa indonesia.

2. 2.2. *Preprocessing*

Setelah melakukan pengumpulan data, tahap selanjutnya yaitu preprocessing data. Tahapan ini dilakukan untuk mempersiapkan data teks agar dapat diproses di tahap selanjutnya. Preprocessing data meliputi beberapa tahapan berikut :

a. Remove Null Value
Memeriksa dan menghapus nilai null dengan dengan tujuan untuk meminimalisir error pada saat pelatihan dan pengujian model.

b. Remove Characters
Menghapus karakter khusus seperti karakter emoji serta kata kunci yang tidak digunakan seperti ‘RT’ untuk Retweet, ‘’USER” untuk username, “URL” untuk link.

c. Case Folding
Mengubah seluruh huruf-huruf pada satu data menjadi huruf kecil (lowercase)

d. Remove Punctuation
Menghapus simbol-simbol yang tidak dipergunakan pada data teks.

e. Text Normalization
Mengubah beberapa kosa kata gaul atau kata-kata yang penulisannya kurang tepat ke dalam kata baku.

f. Stopword removal
Menghapus kata-kata yang tidak dipergunakan.

g. Tokenization
Memecah kata-kata menjadi bagian-bagian yang lebih kecil

3. 2.3. *Word Embedding*

Word Embedding merupakan proses konversi kata yang telah di *preprocessing* sebelumnya ke dalam bentuk vektor. Setiap kata pada vektor yang merepresentasikan sebuah titik pada ruang dengan dimensi tertentu. Pada pustaka Keras terdapat lapisan *Embedding* yang digunakan dalam *Neural Network* pada data teks. lapisan *Embedding* didefinisikan sebagai layer yang tersembunyi dari jaringan. Terdapat 3 argumen yang digunakan, diantaranya :

a. input_dim : ukuran kata pada dokumen teks,
b. output_dim : ukuran vektor dimana kata-kata tersebut disimpan, dan
c. input_length : panjang ukuran input.

4. 2.4. Long Short-Term Memory (LSTM)

Long Short Term Memory merupakan pengembangan dari metode RNN melalui penambahan sel (cell) LSTM di dalam arsitektur RNN. LSTM telah berhasil menyelesaikan berbagai permasalahan, seperti *handwriting recognition, speech recognition, handwriting generation,* dan *image captioning[4].* LSTM memungkinkan arsitektur machine learning menyimpan bobot (*weight*) dari suatu perhitungan lebih lama dari RNN. Hal ini disebabkan karena LSTM memiliki

sebuah node yang *self-recurrent*. LSTM dapat bekerja lebih baik daripada RNN pada data dengan sekuens yang lebih panjang.

Pada bagian bawah arsitektur LSTM, terdapat *cell gates* yang berfungsi untuk meregulasi informasi yang akan dikeluarkan ke *cell state*. *Cell state* adalah jalur pada bagian atas untuk mengirimkan informasi ke unit selanjutnya[4].

Pada *forget gate*, informasi yang tidak dibutuhkan akan diolah dan dihilangkan menggunakan fungsi sigmoid. Persamaan pada forget gate adalah sebagai berikut [4].

$$ft = \sigma(Wf[h_{t-1}, x_t] + b_f) \quad (1)$$

Pada input gate, informasi akan dipilah dan ditentukan sebelum dibawa ke bagian cell state dengan mengunakan fungsi aktivasi sigmoid. Proses ini dapat dihitung dengan persamaan 2. Selain itu, akan ditentukan kandidat vektor baru menggunakan fungsi aktivasi tanh yang akan dibawa ke bagian cell state dengan menggunakan persamaan 3 [4].

$$i_t = \sigma(W_f\ [h_{t-1}, x_t]\ +\ b_i) \quad (2)$$

$$\sim c_t = tanh\ tanh\ (W_c\ [h_{t-1,} x_t]\ +\ b_c\ ) \quad (3)$$

Nilai pada *cell state* lama $\square_{\square-1}$ akan diperbarui ke nilai *cell state* baru $\square_{\square}$ melalui persamaan 4 [4].

$$c_t = f_t * c_{t-1} + i_t * \sim c_t \quad (4)$$

Pada proses ini, fungsi sigmoid dijalankan untuk menghasilkan nilai output pada *hidden state* dan menempatkan *cell state* pada tanh. Selanjutnya, nilai *output sigmoi*d dan nilai *output tanh* dilakukan perkalian sebelum diproses menuju tahap selanjutnya. Perhitugan *output* dilakukan pada persamaan 5 dan 6 [4].

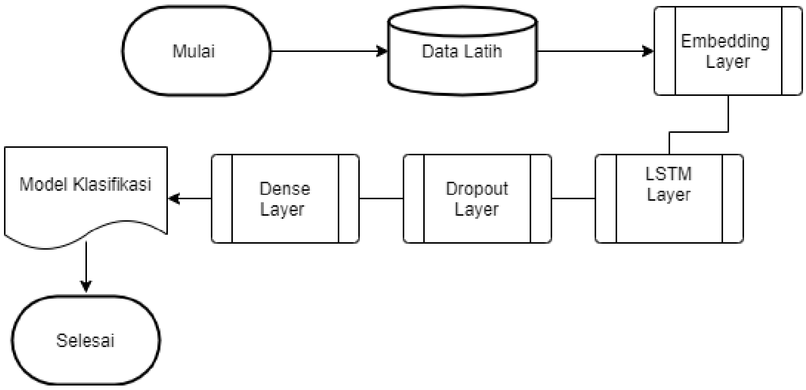
$$o_t = \sigma(W_o * [h_{t-1}, x_t] + b_o) \quad (5)$$

$$h_t = o_c * tanh(c_t) \quad (6)$$

Pada arsitektur LSTM terdapat fungsi aktivasi, yaitu *Sigmoid* dan *Tanh*. Fungsi *sigmoid* digunakan untuk mentransformasikan nilai antara -1 dan 1 menjadi 0 dan 1. Sedangkan fungsi *tanh* digunakan untuk mengatur nilai yang melalui jaringan selalu berada dekat antara -1 dan 1 [4].

Seperti pada Gambar 1, pembuatan model LSTM pada penelitian ini mengimplementasikan sequence model yang terdiri dari tujuh layer yang diantaranya menerapkan *Embedding layer, LSTM layer, Dropout layer,* serta *Dense layer.*

Gambar 1. Diagram Alir Proses Pelatihan dengan Model LSTM



Lapisan Embedding merupakan tahapan mengubah kata ke dalam bentuk vektor. Pendekatan yang digunakan untuk merepresentasikan vektor kata dan array pada bilangan rill. Hasil pada proses word embedding ini akan disimpan pada ruang embedding yang mana kata-kata yang terdapat pada satu ruang tersebut memiliki kesamaan semantik. Peneliti menggunakan library keras untuk proses ini. Cara kerja proses ini adalah sebagai berikut.

a. Dokumen teks yang telah dilakukan preprocessing akan diberikan indeks pada masing-masing data.
b. Parameter input_length digunakan untuk mengatur panjang ukuran vektor.

Pada LSTM layer terdapat parameter memory unit, input shape yang didapat dari hasil word embedding, dan dropout. Nilai dari memory unti dan dropout ditentukan secara eksplisit dengan rentang angka yang sesuai dengan data yang digunakan. Sedangkan parameter dropout digunakan untuk mencegah terjadinya overfitting/underfitting. Nilai yang digunakan berkisar diantara 0 hingga 1. Dense layer digunakan untuk menambahkan layer *fully-connected* berdasarkan jumlah class yang ditentukan.

3. Hasil dan Pembahasan

5. 3.1. Data Understanding

Data yang digunakan dalam penelitian diperoleh melalui github, yaitu *multi-label hate speech and abusive language detection* [3] dan kaggle yaitu *Indonesian Stoplist*. Dataset tersebut memiliki 13169 data dengan 13 label. Hasil pada tahap pengumpulan data ditampilkan pada Gambar 2.

Gambar 2. Hasil Pengumpulan Data

	Tweet	HS	Abusive	HS_Individual	HS_Group	HS_Religion	HS_Race	HS_Physical	HS_Gender	HS_Other	HS_Weak	HS_Moderate	HS_Strong
0	- disaat semua cowok berusaha melacak perhatia...	1	1	1	0	0	0	0	0	1	1	0	0
1	RT USER: USER siapa yang telat ngasih tau elu?...	0	1	0	0	0	0	0	0	0	0	0	0
2	41. Kadang aku berfikir, kenapa aku tetap perc...	0	0	0	0	0	0	0	0	0	0	0	0
3	USER USER AKU ITU AKU\n\nKU TAU MATAMU SIPIT T...	0	0	0	0	0	0	0	0	0	0	0	0
4	USER USER Kaum cebong kapir udah keliatan dong...	1	1	0	1	1	0	0	0	0	0	1	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
13164	USER jangan asal ngomong ndasmu. congor lu yg ...	1	1	1	0	0	0	1	0	0	1	0	0
13165	USER Kasur mana enak kunyuk'	0	1	0	0	0	0	0	0	0	0	0	0
13166	USER Hati hati bisu :(.g\n\nlagi bosan huft	0	0	0	0	0	0	0	0	0	0	0	0

6. 3.2. Data Preprocessing and Preparation

Data diatas selanjutnya akan dilakukan preprocessing data untuk meminimalisir terjadinya bias pada proses selanjutnya. rincian pembagian data ditunjukkan pada Tabel 1:

Table 1. Hasil Preprocessing

Process No.	Process Title	Result
1	**Data Awal**	USER USER USER USER BANCI KALENG MALU GA BISA JAWAB PERTANYAAN KAMI DARI 2 HARI LALU.... NYUNGSEP KOE USER URL
2	***remove character***	BANCI KALENG MALU GA BISA JAWAB PERTANYAAN KAMI DARI 2 HARI LALU.... NYUNGSEP KOE
3	***Case Folding***	banci kaleng malu ga bisa jawab pertanyaan kami dari dua hari lalu…. nyungsep koe
4	***remove punctuation***	banci kaleng malu ga bisa jawab pertanyaan kami dari dua hari lalu nyungsep koe
5	***text normalization***	banci kaleng malu tidak bisa jawab pertanyaan kami dari dua hari lalu nyungsep kau
6	***Tokenization***	['banci', 'kaleng', 'malu', 'tidak', 'bisa', 'pertanyaan', 'kami', 'dari', 'dua', ‘hari’, ‘lalu’,’nyungsep’,’kau’]

Pada Tahapan tokenizer, kamus data atau num_word ditentukan sebanyak 1000. Tahap Selanjutnya yaitu membagi data menjadi 80% data latih dan 20% data uji. Rincian pembagian data dapat dilihat pada tabel 2

	Kelompok Data	Jumlah
1	**Data Latih**	10.535

2	**Data Uji**	2.634
	jumlah	13169

Karena penulis akan melakukan klasifikasi berdasarkan jenis hate speech, maka fitur yang digunakan yaitu *tweet*, *hs_religion, hs_race, hs_physical, hs_gender, hs_other*. Selain dari fitur ini dapat dihapus.

3.3 Pemodelan menggunakan LSTM

Pemodelan diawali dengan menambahkan lapisan Embedding untuk mengubah kata dalam bentuk vektor dengan input_length sama dengan jumlah num_word. Setelah melalui tahapan embedding, kemudian dilanjutkan ke tahapan pemodelan dengan menerapkan parameter-parameter berikut pada masing-masing lapisan.

Layer	**jumlah neuron**	**addition**
embedding		input_dim=1000, output_dim=32
LSTM	**64**	
LSTM	***64***	return_sequences=True
Dense	***16***	
Dropout	***0.2***	
Dense	***6***	activation=softmax

3.4 Evaluasi Model

Berikut hasil evaluasi pengujian model pada evaluasi LSTM dengan menggunakan adam optimizer (Gambar 3).

Gambar 3. Hasil Evaluasi

	precision	recall	f1-score	support
0	0.60	0.54	0.57	167
1	0.67	0.60	0.64	91
2	0.00	0.00	0.00	60
3	0.60	0.06	0.11	50
4	0.73	0.68	0.70	727
5	0.80	0.89	0.84	1539
accuracy			0.77	2634
macro avg	0.57	0.46	0.48	2634
weighted avg	0.74	0.77	0.75	2634

7.

4. Kesimpulan

Berdasarkan hasil penelitian, nilai presisi yang didapat sebesar 74%, recall 77% dan f1-score sebesar 75%. Nilai tersebut dapat berubah jika dilakukan hyperparameter tuning untuk meningkatkan hasil akurasi.

References

[1] T. Spangler, "Variety," 2 February 2022. [Online]. Available: https://variety.com/2022/digital/news/twitter-q4-2021-earnings-users-growth-1235176882/.

[2] D. K. Teologi, “Studia Sosial Religia,” vol. 3, no. 1, pp. 70–82, 2020, [Online]. Available: http://jurnal.uinsu.ac.id/index.php/ssr

[3] M. O. Ibrohim and I. Budi, “Multi-label Hate Speech and Abusive Language Detection in Indonesian Twitter,” pp. 46–57, 2019, doi: 10.18653/v1/w19-3506.

[4] J. Brownlee, “How to Prepare Text Data for Machine Learning with Scikit-Learn,” 2019. [Online]. Available: https://machinelearningmastery.com/prepare-text-data-machine-learning-scikit-learn/.

Halaman ini sengaja dibiarkan kosong