LAPORAN DEEP LEARNING

Klasifikasi Isu Pada Berita Jakarta Menggunakan *Long-Short Term Memory* (LSTM)

Disusun oleh :
Kelompok 5 RA

Naomi Natasya	120450098
Bane Rael Sharin	120450101
Devri Zefanya	120450105
Rayhan Octianto	120450085

PROGRAM STUDI SAINS DATA
FAKULTAS SAINS
INSTITUT TEKNOLOGI SUMATERA
2023

DAFTAR ISI

DAFTAR ISI 2
BAB I PENDAHULUAN 3
1.1 Latar Belakang 3
1.2 Rumusan Masalah 3
1.3 Tujuan 3
BAB II TEORI DASAR 5
2.1 Klasifikasi Teks 5
2.2 Recurrent Neural Network 5
2.3 *Long Short-Term Memory* (LSTM) 6
BAB III METODE PENELITIAN 9
3.1 Perolehan Data 9
3.2 Diagram Alir 9
3.3 Pra-Pemprosesan Data 10
3.4 Pemodelan Data 10
3.5 Mesin Klasifikasi Berita 10
BAB IV HASIL DAN PEMBAHASAN 11
BAB V PENUTUP 17
5.1 Kesimpulan 17
5.2 Saran 17
DAFTAR PUSTAKA 18

BAB I PENDAHULUAN

1.1 Latar Belakang

Berita merupakan hal yang senantiasa menjadi salah satu sumber fakta bagi masyarakat. Berita juga merupakan media yang tak lepas dari kehidupan masyarakat dan menjadi salah satu tempat penyebaran informasi yang penting [1]. Seiring dengan perkembangan teknologi dan internet pada masyarakat, kini berita menjadi pilihan utama dalam mengakses informasi. Hal tersebut dikarenakan mudah diakses atau diperoleh dan juga tersebar di berbagai perangkat [2]. Berdasarkan hasil penelitian yang dilakukan oleh lembaga global GFK dan juga Indonesia Digital Association (IDA) dengan objek penelitian di lima kota besar Indonesia pada tahun 2015, diketahui bahwa persentase konsumsi berita online mencapai 96 persen. Hal tersebut merupakan angka tertinggi, apabila dibandingkan dengan pengkonsumsian berita pada media lainnya [1].

Begitu pentingnya berita sebagai sumber informasi yang akan mempengaruhi pemahaman masyarakat mengenai suatu isu tidak dapat diabaikan. Berita saat ini juga merupakan suatu informasi yang mudah diakses dikarenakan distribusi berita *online* pada era teknologi informasi memberikan pengaruh yang signifikan terhadap pola konsumsi informasi masyarakat. Walaupun berita *online* merupakan suatu kelebihan dalam aksesibilitas dan juga distribusi informasi, tantangan baru akan muncul dalam mengelola jumlah besar data teks yang dihasilkan tiap hari. Klasifikasi akan menjadi solusi yang penting untuk menyaring atau menjadi *filter* berita secara efisien. Oleh karena itu, dilakukan penelitian yang berfokus mengenai pengembangan model klasifikasi teks menggunakan *Long Short Term Memory* (LSTM) yang bertujuan mengkategorikan judul berita berdasarkan isu dan tonalitas. Penggunaan LSTM berguna untuk membantu mengenali pola-pola dalam teks yang muncul dalam berita, sehingga memungkinkan pemahaman yang lebih baik terhadap isu dan tonalitas yang terkandung dalam isu berita.

1.2 Rumusan Masalah

Adapun rumusan masalah dari penelitian ini, yaitu:

1. Bagaimana pengembangan model klasifikasi teks berdasarkan isu berita menggunakan metode *Long Short Term Memory* (LSTM) guna menghasilkan pengelompokkan isu berita berdasarkan kategori tonalitas?
2. Bagaimana performa sistem klasifikasi *Long Short Term Memory* (LSTM) pada klasifikasi teks dalam isu-isu yang ada pada berita Jakarta tahun 2019–2020?

1.3 Tujuan

Tujuan dari penelitian adalah sebagai berikut :

1. Mengembangkan model klasifikasi teks pada isu berita menggunakan metode *Long Short Term Memory* (LSTM), sehingga dapat menghasilkan pengelompokkan isu berita berdasarkan kelas atau kategori yang telah ditentukan sebelumnya.

2. Dapat mengetahui akurasi dan performa model klasifikasi teks menggunakan *Long Short Term Memory* (LSTM) pada berita Jakarta tahun 2019-2020.

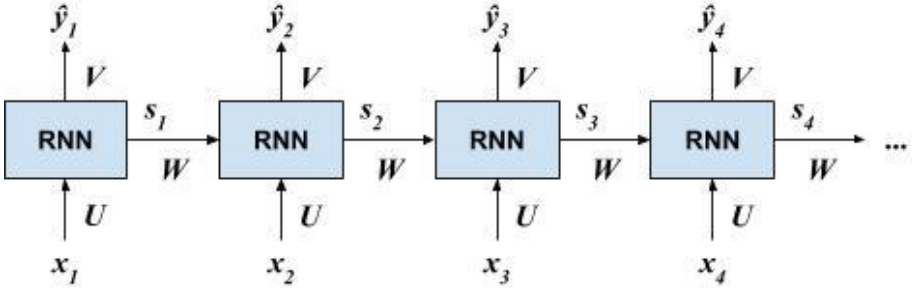
BAB II TEORI DASAR

2.1 Klasifikasi Teks

Klasifikasi teks merupakan suatu proses mengkategorikan teks ke dalam kelas atau kategori tertentu. Klasifikasi teks akan berfokus pada tiga topik utama, yaitu rekayasa fitur (*feature engineering*), pemilihan fitur (*feature selection*), dan menggunakan berbagai jenis algoritma pembelajaran mesin. Pada rekayasa fitur (*feature engineering*) sebagian besar menggunakan fitur bag-of-words. Selain itu, beberapa fitur lebih kompleks telah dirancang, seperti *tag part-of-speech*, frasa kata benda, dan kernel pohon. Selain rekayasa fitur, terdapat pemilihan fitur yang bermanfaat untuk mengatur fitur yang tidak diperlukan sehingga dapat meningkatkan kinerja klasifikasi [4].

2.2 Recurrent Neural Network

Recurrent Neural Network (RNN) merupakan salah satu metode pengklasifikasian *deep learning* yang bekerja dengan menganalisis semua data dan parameter yang digunakan pada tiap kali dibagikan [5]. Berikut merupakan gambar dari arsitektur RNN:



Gambar 2.1. Arsitektur RNN (Sumber : Graves, A., Schmidhuber, J. (2013))

Gambar 2.1. merepresentasikan skema dari *Recurrent Neural Network* (RNN) yang merupakan salah satu jenis jaringan saraf tiruan yang digunakan untuk memproses data secara berurutan. RNN mempunyai *loop* di dalamnya, hal tersebut memungkinkan dalam mengingat informasi dari waktu ke waktu dan memungkinkan RNN memahami pola dalam data yang berurutan tersebut, seperti pola dalam teks atau sinyal audio [6]. Jaringan RNN menggunakan fungsi aktivasi *sigmoid* untuk *hidden layer* yang memiliki *output* dengan rentang 0 sampai dengan 1. Terdapat dua persamaan pada fungsi *sigmoid* yang dapat dilihat pada Persamaan 1 dan Persamaan 2.

$$f(x) = \frac{1}{1 + e^{-x}} \quad (1)$$

$$f'(x) = f(x)(1 - f(x)) \quad (2)$$

Pada RNN, fungsi *sigmoid* digunakan untuk mentransformasikan nilai input menjadi nilai yang dapat digunakan oleh jaringan saraf tiruan. Fungsi *sigmoid* digunakan untuk

mengatasi masalah *vanishing gradient* yang kerap digunakan pada RNN yang kompleks dengan membantu RNN memahami pola dalam data yang lebih kompleks. Pada penggunaan RNN dalam komputasi terdapat hubungan perulangan (*loop*) dengan fungsi aktivasi, sehingga fungsi akan bergantung pada bobot. Hal tersebut dilanjutkan dengan proses yang selanjutnya menggunakan Persamaan 3.

$$h_t = f_2(h_{t-1}, x_t) \qquad (3)$$

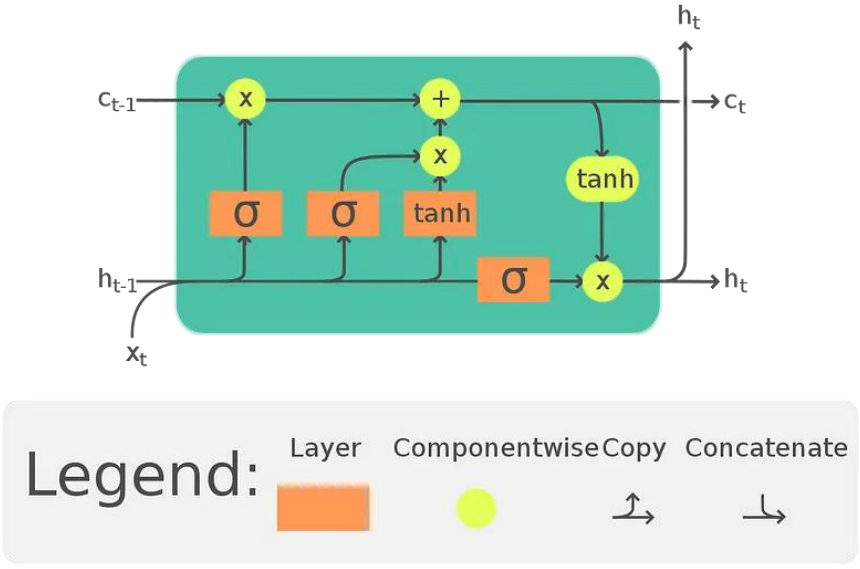
Nilai x akan diinput ke dalam fungsi aktivasi dan bobot yang sama tiap dilakukannya perhitungan. Apabila terdapat data nonlinier, maka akan dilakukan persamaan seperti Persamaan 4.

$$y_t = W_{hy} h_t \qquad (4)$$

Pada RNN, parameter yang dibagikan secara merata pada tiap *time step*, maka gradient guna *output* tergantung tidak hanya pada kalkulasi pada *time step* saat ini, akan tetapi pada *time step* sebelumnya.

2.3 *Long Short-Term Memory* (LSTM)

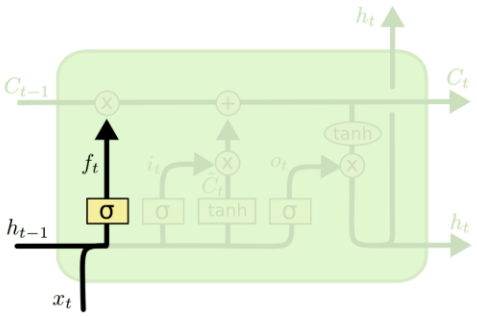
LSTM merupakan modifikasi dari RNN yang memiliki memori dan banyak jenis gerbang yaitu *input gate*, *forget gate*, dan *output gate* [7]. LSTM mampu mempelajari lebih dari 1000 langkah sebelumnya tergantung pada kompleksitas jaringan.



Gambar 2.2. Arsitektur LSTM (Sumber : Wikipedia)

Konsep inti dari LSTM adalah *cell state* dan 4 *gates* pada LSTM. *Cell state* merupakan alat yang berfungsi membawa informasi penting yang sudah melalui seluruh *gates* pada 1 cell LSTM menuju *cell* berikutnya. Komponen yang terdapat pada LSTM:

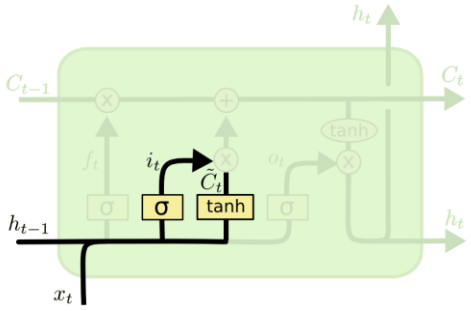
- *Forget gate* memutuskan informasi yang akan disimpan ataupun dibuang. *Forget gate* akan menerima informasi yang berupa *hidden state* yang berasal dari *cell* sebelumnya dan informasi baru yang berasal dari *input* saat ini, kemudian informasi tersebut akan digabung dan diproses dengan menggunakan fungsi *sigmoid* yang akan menghasilkan hasil berupa 0 hingga 1. Semakin dekat hasil yang diperoleh dengan 0 maka berarti informasi akan dibuang, sebaliknya semakin dekat dengan 1 maka informasi akan disimpan.



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Gambar 2.3. (Sumber : Colah's Blog (2015))

- *Input gate* akan menerima informasi yang berupa *hidden state* yang berasal dari *cell* sebelumnya dan informasi baru yang berasal dari input saat ini, kemudian informasi tersebut akan digabung dan diproses dengan menggunakan fungsi *sigmoid* dan fungsi *tanh*. Hasil dari fungsi *sigmoid* akan mengubah nilai menjadi 0 hingga 1 untuk menentukan informasi mana yang akan di-*update*. Semakin dekat dengan 0 berarti informasi tidak penting, maka semakin dekat dengan 1 yang artinya informasi penting. Hasil dari fungsi *tanh* berupa nilai -1 hingga 1 digunakan untuk mendukung agar *cell* dapat mempelajari informasi dengan lebih baik.

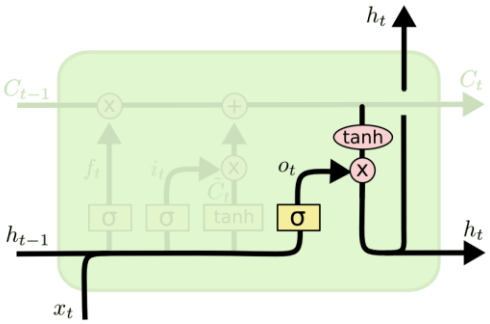


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Gambar 2.4. (Sumber : Colah's Blog (2015))

- *Output gate* akan menentukan *hidden state* yang akan dikirim ke *cell* selanjutnya. *Output gate* akan menerima informasi yang berupa *hidden state* yang berasal dari *cell*

sebelumnya dan informasi baru yang berasal dari *input* saat ini kemudian informasi akan digabung dan diproses dengan fungsi *sigmoid*. *Cell state* yang baru kemudian akan diproses melalui fungsi *tanh*. Hasil dari fungsi tanh akan dikalikan dengan hasil dari fungsi *sigmoid* untuk memperoleh informasi yang akan disimpan pada *hidden state* yang baru. *Hidden state* dan *cell state* yang baru kemudian akan diteruskan ke *cell* selanjutnya.



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

$$h_t = o_t * \tanh\left(C_t\right)$$

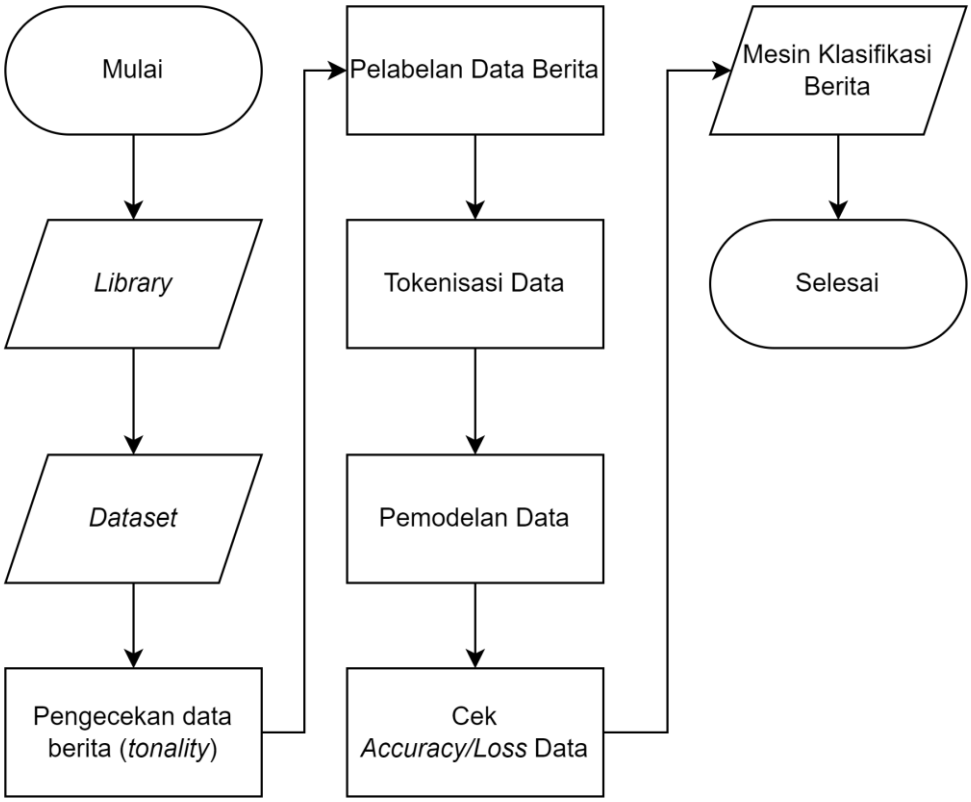
Gambar 2.5. (Sumber : Colah's Blog (2015))

BAB III METODE PENELITIAN

3.1 Perolehan Data

Perolehan data atau *dataset* berupa kumpulan berita yang didapatkan dari Jakarta Open Data dengan rentang waktu 2019–2020. Data berisikan Berita, Tanggal, Isu, Tonalitas (Positif, Netral, dan Negatif), serta Tautan. Dalam hal Tonalitas, berita sudah disematkan label sesuai isinya, yaitu positif jika berisikan positif, begitu pun dengan negatif dan netral.

3.2 Diagram Alir



Gambar 3.1. Diagram Alir

Pengerjaan dimulai dengan mengumpulkan *library* dan *dataset*. Setelahnya, pengecekan data(*tonality*), pelabelan data berita, dan tokenisasi data menjadi pra-pemprosesan data dalam pengerjaan. Data dimodelkan dan dicek *accuracy/loss*-nya. *Output* dari pemodelan data berupa mesin klasifikasi berita.

3.3 Pra-Pemprosesan Data

Pada tahap pra-pemrosesan data dilakukannya pelabelan data berita berdasarkan *tonality* pada data yang berisikan *negatif*, *positif*, dan *netral*. Lalu, dilakukan *case folding* yang bertujuan untuk mengubah huruf dalam teks menjadi huruf kecil atau huruf besar dan *stopword removal* yang bertujuan untuk menghapus kata yang tak memiliki makna atau tidak relevan pada teks. Setelah itu dilakukannya tokenisasi yang bertujuan memecahkan teks menjadi bagian yang lebih kecil sehingga dapat mudah diproses dalam algoritma pemodelan.

3.4 Pemodelan Data

Setelah dilakukan pengecekan data berita (tonalitas) dan memberikan label pada berita (Positif, Negatif, dan Netral) agar dapat terarah ketika dilakukan pemodelan. Setelah itu, dilakukan pembagian (*split*) data latih dan data tes. Kedua kelompok data tersebut–data latih dan tes–diberikan tokenisasi dengan metode *Long Short-Term Memory* (LSTM). Pada tokenisasi, dibubuhkan fungsi aktivasi dengan masing-masing 32 dan 3 lapisan tersembunyi. Data yang sudah memasuki tahap tokenisasi akan dimodelkan dan dicek akurasinya. Dalam pemodelan, data akan dilakukan *epoch* sebanyak 40 kali, dengan *verbose* sebanyak 2. Lalu, cek *accuracy* dan *loss*-nya.

3.5 Mesin Klasifikasi Berita

Setelah dilakukan pemodelan menggunakan LSTM, dilakukan pembuatan fungsi yang berguna untuk mesin klasifikasi berdasarkan isu berita sehingga mendapatkan tonalitas berita tersebut. Menggunakan mesin klasifikasi tersebut dapat membuat prediksi mengenai tonalitas (*Negatif, Netral,* dan *Positif)* dari teks berita yang dimasukkan oleh pengguna. Prediksi tersebut menggunakan model dan *tokenizer* yang telah diberikan, sehingga hasil prediksi akan dicetak ke konsol.

BAB IV HASIL DAN PEMBAHASAN

Penelitian ini menggunakan data pelatihan yang telah dilakukannya proses pelabelan kalimat berdasarkan tiga pengelompokan kelas. Setiap kalimat terdiri dari Negatif, Netral, Positif dari kategori *tonality*. Data dibagi menjadi 2 bagian, 80% untuk pelatihan 20% untuk pengujian. Algoritma klasifikasi yang digunakan dalam penelitian ini adalah LSTM (*Long Short-Term Memory*) yang diterapkan menggunakan teknik pembobotan menggunakan *dataset* teks isu berita Jakarta NLP.

Gambar 4.1. adalah sampel *dataset* pada penelitian ini. Selanjutnya dilakukan *One-hot Encoding* pada kolom "tonality", maka didapatkan hasil dari *dataset* tersebut memiliki 3 kategori isu berita yaitu "Negatif", "Netral", dan "Positif".
Pada Model jaringan saraf dibangun dengan berbagai lapisan, termasuk lapisan *embedding*, LSTM, dan lapisan *dropout* untuk mencegah *overfitting*. Model ini dikompilasi dengan fungsi kerugian dan pengoptimal yang sesuai untuk tugas klasifikasi teks.

	tanggal	isu	tonality	berita_jakarta	link/page
0	2019-08-01	Nasib DKI Jakarta Setelah Ibu Kota Pindah ke K...	NETRAL	NaN	NaN
1	2019-08-01	Integrasi Transportasi Oleh Jaklingko	POSITIF	Musyawarah Kerja Daerah Organda Jakarta Anies ...	https://www.beritajakarta.id/read/70758/musyaw...
2	2019-08-01	Integrasi Transportasi Oleh Jaklingko	POSITIF	Warga Apresiasi Pengoperasian Dua Jak Lingko d...	https://www.beritajakarta.id/read/70829/warga-...
3	2019-08-01	Pengelolaan Sampah Jakarta vs Surabaya	NETRAL	NaN	NaN
4	2019-08-02	Sidang Perdana Polusi Udara	NETRAL	Dishub Ingin Penggunaan Kendaraan Listrik Lebi...	https://www.beritajakarta.id/read/70807/dewan-...
5	2019-08-02	Sidang Perdana Polusi Udara	NETRAL	Pemprov DKI-ICLEI Sepakati Kerja Sama Penuruna...	https://www.beritajakarta.id/read/70819/pempro...
6	2019-08-02	Sidang Perdana Polusi Udara	NETRAL	Terbitkan Ingub Pemprov DKI Siapkan 7 Inisiati...	https://www.beritajakarta.id/read/70850/terbit...
7	2019-08-	Sidang Perdana Polusi Udara	NETRAL	DKI Matangkan Aplikasi Pemantauan Kualitas	https://www.beritajakarta.id/read/70888/dki-ma

Gambar 4.1. *Dataset* yang Dipakai

Untuk membangun dan melatih arsitektur model, sebelumnya dataset dibagi menjadi 80%training set dan 20% test set. Training set digunakan untuk melatih model sedangkan test set untuk memvalidasi performa dari model. Selanjutnya dilakukan tokenizing untuk memecahkan kalimat menjadi kumpulan kata penyusun kalimat tersebut dan kata tersebut dipresentasikan menjadi angka atau numerik.

Kemudian membangun arsitektur model sebelum melakukan training model dimana menggunakan model sequential dengan arsitektur RNN (Recurrent Neural Network). Gambar 4.2. menunjukkan bahwa lapisan embedding dengan dimensi input 2000 dan dimensi keluaran 8. lapisan LSTM menggunakan 3 unit. Lapisan dense 32 dengan aktivasi ReLu dan 3 unit dengan aktivasi softmax. Pada penelitian ini dilakukan beberapa pengujian dengan mengubah nilai parameter untuk mencari nilai terbaik. Parameter yang akan diubah antara lain jumlah neuron, fungsi aktivasi dan nilai *epoch.*

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, None, 8)           16000

 dropout (Dropout)           (None, None, 8)           0

 lstm (LSTM)                 (None, 32)                5248

 dropout_1 (Dropout)         (None, 32)                0

 dense (Dense)               (None, 32)                1056

 dropout_2 (Dropout)         (None, 32)                0

 dense_1 (Dense)             (None, 3)                 99

=================================================================
Total params: 22403 (87.51 KB)
Trainable params: 22403 (87.51 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

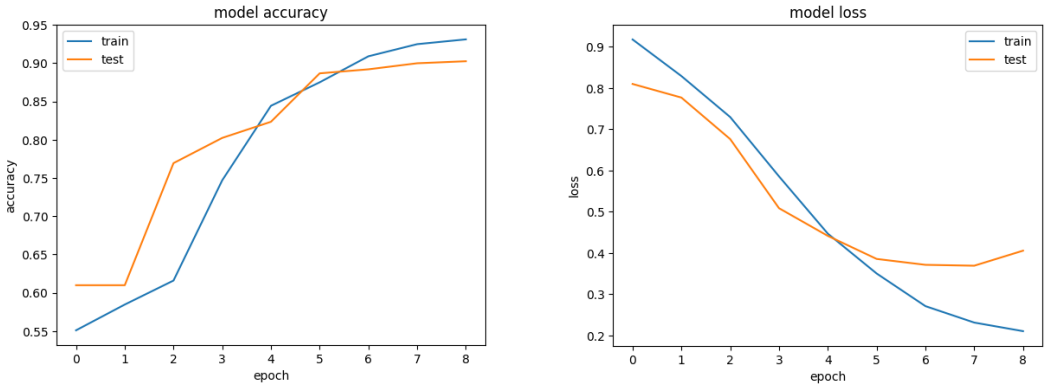
Gambar 4.2. LSTM Parameters

Selama pelatihan model, sebuah *callback* disertakan untuk memantau kinerja model, dan pelatihan akan berhenti jika akurasi pelatihan dan validasi mencapai lebih dari 90%. Hasil pelatihan, termasuk akurasi dan kerugian, divisualisasikan dalam grafik. Pelatihan model klasifikasi teks yang mencakup penggunaan data latih, data validasi, jumlah *epoch*, dan *callback* untuk mengendalikan proses pelatihan model. Kemudian proses pelatihan model menggunakan 40 *epoch* sesuai dengan nilai yang ditentukan, namun proses latih bisa selesai sebelum *epoch* 40 karena *callback* tersebut agar proses latih berhenti jika akurasi sudah mencapai 90%.

```
Epoch 1/40
95/95 - 5s - loss: 0.9176 - accuracy: 0.5511 - val_loss: 0.8098 - val_accuracy: 0.6100 - 5s/epoch - 51ms/step
Epoch 2/40
95/95 - 1s - loss: 0.8289 - accuracy: 0.5848 - val_loss: 0.7768 - val_accuracy: 0.6100 - 1s/epoch - 12ms/step
Epoch 3/40
95/95 - 1s - loss: 0.7295 - accuracy: 0.6161 - val_loss: 0.6761 - val_accuracy: 0.7694 - 1s/epoch - 14ms/step
Epoch 4/40
95/95 - 2s - loss: 0.5857 - accuracy: 0.7470 - val_loss: 0.5086 - val_accuracy: 0.8024 - 2s/epoch - 19ms/step
Epoch 5/40
95/95 - 1s - loss: 0.4470 - accuracy: 0.8443 - val_loss: 0.4415 - val_accuracy: 0.8235 - 1s/epoch - 13ms/step
Epoch 6/40
95/95 - 1s - loss: 0.3504 - accuracy: 0.8750 - val_loss: 0.3857 - val_accuracy: 0.8867 - 1s/epoch - 11ms/step
Epoch 7/40
95/95 - 1s - loss: 0.2712 - accuracy: 0.9090 - val_loss: 0.3714 - val_accuracy: 0.8920 - 1s/epoch - 11ms/step
Epoch 8/40
95/95 - 1s - loss: 0.2314 - accuracy: 0.9248 - val_loss: 0.3692 - val_accuracy: 0.8999 - 1s/epoch - 11ms/step
Epoch 9/40

Akurasi telah mencapai >90%!
95/95 - 1s - loss: 0.2106 - accuracy: 0.9311 - val_loss: 0.4057 - val_accuracy: 0.9025 - 1s/epoch - 12ms/step
```

Gambar 4.3. *Accuracy/Loss* Memakai *Epoch*



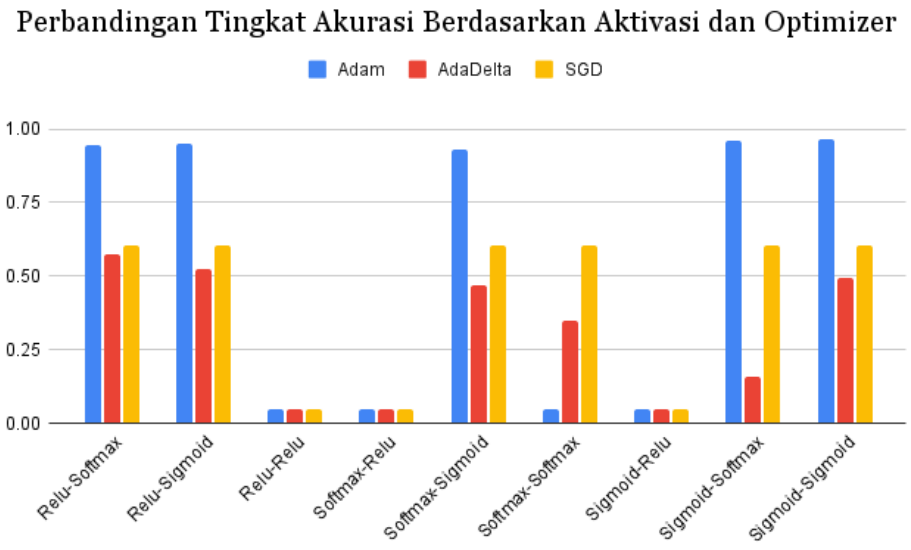
Gambar 4.4. Model *Accuracy/Loss* Memakai Grafik

Akurasi untuk data uji paling tinggi dan *loss* data uji paling rendah ditunjukkan pada gambar 4.4. Dalam proses peningkatan parameter pembelajaran klasifikasi, optimasi 'adam' melakukan lebih cepat, sehingga akurasi yang didapatkan dari penelitian ini sebesar 0.93 atau 93%. Kemudian untuk mendapatkan prediksi tonalitas (Negatif, Netral, Positif) dari model yang dilatih sebelumnya, dibentuklah fungsi uang memungkinkan pengguna untuk memasukkan teks berita tersebut. Dengan fungsi ini, pengguna dapat dengan mudah menguji model klasifikasi teks yang telah dilatih dengan memasukkan teks berita dan melihat bagaimana model mengkategorikan tonisitasnya. Model jaringan saraf dibangun dengan berbagai lapisan, termasuk lapisan embedding, LSTM, dan lapisan *dropout* untuk mencegah *overfitting*. Model ini dikompilasi dengan fungsi kerugian dan pengoptimal yang sesuai untuk tugas klasifikasi teks.

Untuk mendapatkan hasil akurasi dari performa pemodelan terbaik dilakukan perbandingan dengan membandingkan penggunaan aktivasi yang digunakan dan juga optimizer. Berikut merupakan perbandingan nilai akurasi tersebut.

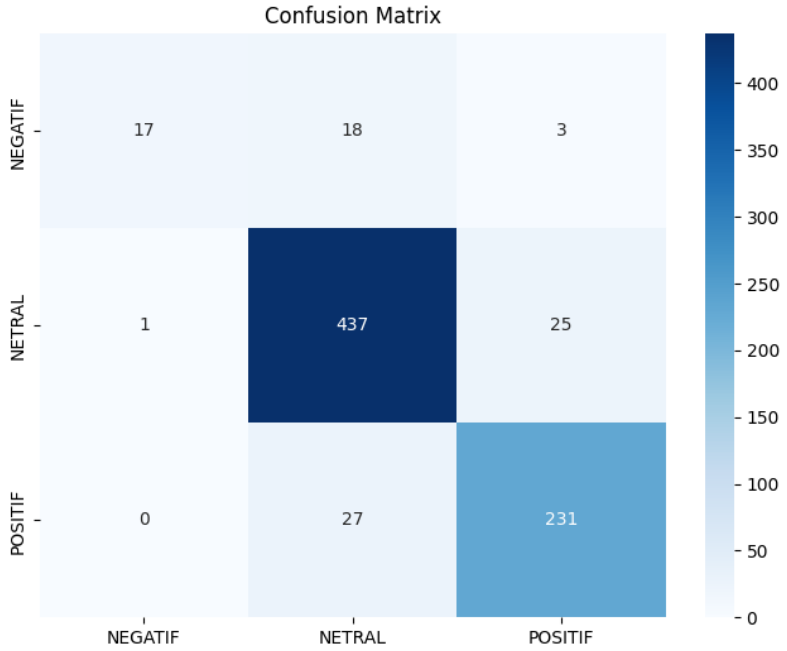
		OPTIMIZER		
		Adam	AdaDelta	SGD
ACTIVATION	Relu-Softmax	0.9456	0.5762	0.6026
	Relu-Sigmoid	0.9518	0.5247	0.6026
	Relu-Relu	0.0491	0.0491	0.0491
	Softmax-Relu	0.0491	0.0491	0.0491
	Softmax-Sigmoid	0.9307	0.4677	0.6026
	Softmax-Softmax	0.0491	0.3496	0.6026
	Sigmoid-Relu	0.0491	0.0491	0.0491
	Sigmoid-Softmax	0.9627	0.1567	0.6026
	Sigmoid-Sigmoid	0.9677	0.4944	0.6026

Tabel 4.1 Tabel Perbandingan Akurasi Performa Model



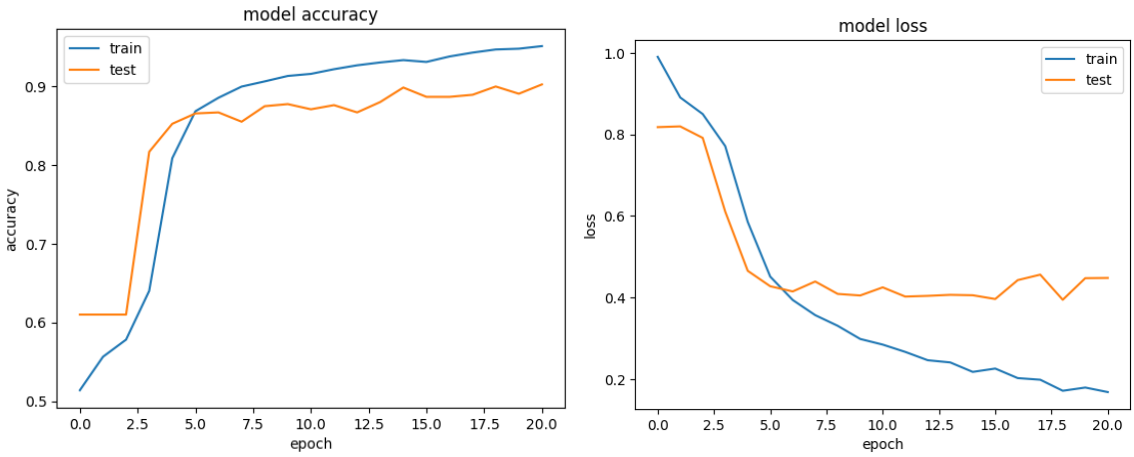
Gambar 4.5 Grafik Perbandingan Tingkat Akurasi Berdasarkan Aktivasi dan Optimizer

Berdasarkan perbandingan tersebut dapat diketahui bahwa pemodelan dengan nilai akurasi terbaik dengan menggunakan aktivasi Sigmoid dan Sigmoid dengan jenis optimizer yaitu Adam. Sehingga didapatkan *confusion matrix* dan grafik *accuracy* serta loss sebagai berikut.



Gambar 4.6 Grafik *Confusion Matrix*

Berdasarkan pada gambar 4.6, grafik tersebut menunjukkan hasil klasifikasi ke dalam tiga kategori yaitu *negatif*, *netral*, dan *positif*. Matriks tersebut menggunakan skala warna yang mana warna biru gelap hingga biru terang di sebelah kanan matriks menunjukkan jumlah dari rendah hingga tinggi. Angka pada kotak mewakili jumlah prediksi untuk tiap kombinasi label sebenarnya dan label yang diprediksi. Terdapat 17 kasus *negatif* yang diklasifikasikan dengan benar, 437 kasus *netral* yang diklasifikasikan dengan benar, dan 231 kasus *positif* yang diklasifikasikan dengan benar. Ada juga beberapa kesalahan seperti 18 kasus *negatif*yang salah diklasifikasikan sebagai *netral* dan 27 kasus *netral* yang salah diklasifikasikan sebagai *positif*.



Grafik 4.7 Grafik Model *Accuracy* dan *Loss*

Berdasarkan gambar 4.7, pada grafik model *accuracy* menunjukan akurasi model terhadap data latih dan uji selama beberapa *epoch*. Awalnya, meningkatnya akurasi yang cepat untuk kedua data, akan tetapi ketika *epoch* meningkat, akurasi terus meningkat namun dengan laju yang lebih lambat. Lalu pada grafik model *loss* menunjukkan kehilangan model selama pelatihan dan pengujian. Kedua garis tersebut menunjukkan penurunan dan kerugian seiring bertambahnya *epoch*.

Berdasarkan model tersebut, dapat menunjukkan bahwa model dapat dilakukan dari data latih sehingga menghasilkan prediksi yang akurat pada data uji.

BAB V PENUTUP

5.1 Kesimpulan

Kesimpulan dari laporan ini adalah metode klasifikasi teks pada isu berita Jakarta dapat menggunakan *Long Short Term Memory* (LSTM) dapat menjadi solusi yang cukup efektif untuk memilah dan mengkategorikan berita. Klasifikasi menggunakan *Long Short Term Memory* (LSTM) mendapatkan hasil akurasi terbaik senilai 0.9677 atau mencapai 96,77% dengan aktivasi Sigmoid dengan Sigmoid dan menggunakan Adam sebagai *optimizer*.

5.2 Saran

Berdasarkan hasil penelitian, saran untuk penelitian selanjutnya adalah sebagai berikut:

1. Meningkatkan jumlah data pelatihan untuk meningkatkan akurasi model;
2. Menggunakan jenis recurrent neural network yang lebih kompleks untuk meningkatkan akurasi model;
3. Menerapkan metode klasifikasi teks untuk kategori berita yang lebih spesifik.

DAFTAR PUSTAKA

[1] F. S. A. H. and S. Y. , "Klasifikasi Topik terhadap Judul Berita Kasus Covid-19 dengan Multilayer Perceptron," *Techno.COM,* vol. 21, 2022.

[2] W. Afandi, S. N. Saputro, H. Ardiansyah, M. H. Kafabi and S. Sudianto, "Klasifikasi Judul Berita Clickbait Menggunakan RNN-LSTM," *Jurnal Informatika : Jurnal pengembangan IT (JPIT),* vol. 7, 2022.

[3] A. A. Irga, A. and M. S. Mubarok, "Klasifikasi Topik Berita Berbahasa Indonesia Menggunakan K-Nearest Neighbor," *e-Proceeding of Engineering,* vol. 5, 2018.

[4] M. Kaur and A. Mohta, "A Review of Deep Learning with Recurrent Neural Network," *2019 International Conference on Smart Systems and Inventive Technology (ICSSIT),* 2019.

[5] S. F. S. Hadi, J. and K. M. Lhaksmana, "Analisis Sentimen Menggunakan Recurrent Neural Network Terkait Isu Anies Baswedan Sebagai Calon Presiden 2024," *e-Proceeding of Engineering,* vol. 10, 2023.

[6] S. Lai, L. Xu and J. Zhao, "Recurrent Convolutional Neural Networks for Text Classification," *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence,* 2015.

[7] V. Gallan, "LSTM (Long Short Term Memory)," 29 Maret 2023. [Online]. Available: https://medium.com/bina-nusantara-it-division/lstm-long-short-term-memory-d29779e2ebf8.