p-ISSN : 2301-8402, e-ISSN : 2685-368X , available at : https://ejournal.unsrat.ac.id/index.php/elekdankom

Sentiment Analysis of Social Media Users Using Long-Short Term Memory Method

Analisis Sentimen Pengguna Sosial Media Menggunakan Metode *Long Short Term Memory*

Cassey K. N. Paputungan , Agustinus Jacobus
Dept. of Electrical Engineering, Sam Ratulangi University Manado, Kampus Bahu St., 95115, Indonesia
E-mail : cassey.paputugan@gmail.com , a.jacobus@unsrat.ac.id
Received: 4 March 2021; revised: 14 June 2021; accepted: 16 June 2021

Abstract* — Social media is a medium that can be used to share information between users who serve it with various sentiments about something that is being discussed. Twitter is a social media that is filled with various sentiments from its users and can be in the form of negative, neutral, or positive sentiments. In this case, we need a tool that can aggregate the overall sentiment shown by its users for a product, issue, or service, but it takes a longer time if the sentiment is classified manually so that a system is needed that can perform this sentiment analysis using one of the artificial neural network method, namely Long Short Term Memory. Based on the test results of several parameters carried out on the Long Short Term Memory method to carry out the sentiment analysis process, the best performance was obtained with an accuracy value of 76% using the ADAM learning rate of 0.0001, the number of neurons 100 in the LSTM layer, dropout 0.9, epoch 55, and a batch size of 64.

***Keywords* —Long Short Term Memory; Sentiment Analysis; Social Media; Twitter**

Abstrak — Sosial media merupakan media yang dapat digunakan untuk saling berbagi informasi antar pengguna yang menyebabkannya dipenuhi dengan berbagai sentimen akan suatu hal yang sedang marak dibicarakan. Twitter merupakan salah satu media sosial yang dipenuhi dengan beragamnya sentimen dari penggunanya dan dapat berupa sentimen negatif, netral, ataupun positif. Dalam hal ini, diperlukan suatu *tools* yang mampu mengagregasi keseluruhan sentimen yang ditunjukkan oleh penggunanya akan suatu produk, isu, atau layanan namun dibutuhkan waktu yang lebih lama jika sentimen tersebut di klasifikasikan secara manual sehingga diperlukan suatu sistem yang dapat melakukan analisis sentimen tersebut dengan menggunakan salah satu metode jaringan saraf tiruan, yaitu *Long Short Term Memory*. Berdasarkan hasil pengujian dari beberapa parameter yang dilakukan pada metode *Long Short Term Memory* untuk melakukan proses analisis sentimen, didapatkan performa terbaik yaitu nilai akurasi sebesar 76% dengan menggunakan *learning rate* ADAM sebesar 0.0001, jumlah *neuron* 100 pada lapisan LSTM, *dropout* 0.9, *epoch* 55, dan *batch size* 64.

Kata kunci —Analisis Sentimen*; Long Short Term Memory; Media Sosial; Twitter*

I. PENDAHULUAN

Seiring dengan berkembangnya internet di Indonesia, segala informasi di media sosial. Media sosial sendiri merupakan media yang dapat digunakan untuk saling berbagi informasi yang dilakukan secara dalam jaringan (daring) sehingga para penggunanya bisa saling berkomunikasi tanpa perlu terjadinya pertemuan tatap muka secara langsung. Salah satu media sosial yang paling sering digunakan oleh para warganet adalah *Twitter.* Berbeda dengan media sosial lain, Twitter membatasi jumlah penulisan di dalamnya, yaitu 280 karakter, naik dari jumlah sebelumnya yaitu 140 karakter. Hal ini membuat *tweet* -tulisan yang di*posting* oleh pengguna Twitter, berisikan pesan yang lebih singkat, padat, dan jelas sehingga para pengguna lebih ringkas untuk memberikan suatu informasi atau pendapatnya. Dengan lebih ringkasnya suatu *tweet* untuk memberikan informasi tersebut membuat Twitter menjadi salah satu media sosial yang berisikan sentimen-sentimen akan suatu hal. Sentimen tersebut bisa berupa sentimen positif, netral, maupun negatif. Untuk dapat menganalisa berbagai macam sentimen pengguna Twitter maka diperlukan metode analisis sentimen atau *opinion mining*.

Terdapat banyak macam model klasifikasi untuk menentukan suatu sentimen, salah satunya adalah metode *Long Short Term Memory* (LSTM). LSTM merupakan bagian dari model *deep learning* yang dapat digunakan untuk melakukan klasifikasi sentimen. Metode ini dapat memproses data secara sekuensial seperti teks, suara, dan video.[1] Pada penelitian yang dilakukan oleh M. A. Nurrohmat dan Azhari SN. yang berjudul *Sentiment Analysis of Novel Review Using Long Short-Term Memory Method,* LSTM menghasilkan akurasi yang tinggi jika dibandingkan dengan penggunaan metode lainnya dalam proses analisis sentimen.[2]

Dalam hal ini, perlu dibuatnya suatu *tools* yang mampu mengagregasi keseluruhan sentimen-sentimen yang ditunjukkan pengguna Twitter akan suatu produk, isu atau layanan yang sedang marak dibicarakan dan mengklasifikasikannya secara otomatis dengan menggunakan *opinion mining* atau analisis sentimen sehingga menghasilkan presentase sentimen yang muncul dengan menggunakan salah satu metode jaringan saraf tiruan, yaitu *Long Short-Term Memory*.

Tujuan yang ingin dicapai dari penelitian ini adalah untuk membangun suatu sistem yang dapat mengklasifikasikan sentimen *tweet* pengguna Twitter dalam menanggapi suatu topik atau isu yang sedang marak secara otomatis menggunakan metode *Long Short-Term Memory.*

A. Penelitian Terkait

Penelitian yang dilakukan oleh W. Umboh melakukan analisis sentimen pengguna sosial media akan objek wisata. Penelitian ini membahas tentang bagaimana membuat sistem yang dapat mencari status pengguna dari sosial media akan suatu topik objek wisata dan mengkategorikannya ke dalam sentimen positif, netral, dan negatif sehingga dapat diukur tingkat sentimen pengguna sosial media terhadap objek wisata tersebut. Metode klasifikasi yang digunakan adalah algoritma *Support Vector Machine.* Berdasarkan hasil dari penelitian dengan menggunakan *Support Vector Machine*, kernel terbaik yang didapatkan dalam sistem ini adalah kernel linear yang mencapai akurasi sebesar 68%.[3]

Penelitian yang dilakukan oleh A. Faadilah melakukan penelitian mengenai analisis sentimen pada ulasan aplikasi Tokopedia di Google Play Store. Penelitian ini membahas bagaimana mencari model yang optimal yang dapat memprediksi sentimen dengan model *Long Short Term Memory* pada ulasan aplikasi Tokopedia di Google Play Store. Untuk mendapatkan model terbaik, digunakan pengujian parameter untuk mendapatkan model yang terbaik. Dari pengujian tersebut, menghasilkan akurasi yang cukup baik pada parameter 400 jumlah neuron dan dengan fungsi aktivasi sigmoid. Selanjutnya dilakukan pengujian pada 20% dataset, yang menghasilkan akurasi 93.32%, presisi 95.17%, dan nilai *recall* 97.15%.[4]

Penelitian yang dilakukan Elisabeth melakukan perbandingan pada metode *Naïve Bayes* dan *Long Short Term Memory* (LSTM) dalam melakukan analisis sentimen. Penelitian ini melakukan eksperimen pada dua jenis model analisis sentimen yang berbeda yaitu *Long Short Term Memory* (LSTM) dan *Naïve Bayes* yang dilatih dan dites dengan dataset yang serupa, yaitu studi kasus pada brand *Indomie* dan dicari tingkat akurasi tertinggi untuk bisa dikategorikan apakah kata-kata di dalam *tweets* bersifat dependen atau independen. Berdasarkan hasil eksperimen yang telah dilakukan, model *Long Short Term Memory*(LSTM) mengungguli kinerja *Naïve Bayes*. Model LSTM mendapatkan akurasi sebesar 77.92%, sementara *Naïve Bayes* hanya mencapai 66.31% yang menandakan bahwa kata-kata di dalam tweet bersifat dependen dan mempengaruhi makna suatu tweet.[5]

Berdasarkan penelitian-penelitian terkait yang telah dilakukan sebelumnya, terdapat beberapa perbedaan dengan penelitian yang saat ini dilakukan, yaitu pada penelitian ini dibuat suatu sistem analisis sentimen yang dapat melakukan klasifikasi sentimen-sentimen *tweet* dari pengguna Twitter dengan menggunakan metode *Long Short Term Memory.*

B. Text Mining

Text mining adalah proses untuk mengekstraksi pola yang menarik dan signifikan untuk mengeksplorasi pengetahuan dari sumber data tekstual.[6] Proses penemuan pola pada data miningdapat dilakukan dengan metode klasifikasi. *Text Mining* memiliki tujuan dan proses yang mirip dengan *data mining,* namun berbeda pada proses masukannya. *Text Mining* menggunakan teks sebagai masukannya, yang berarti merupakan data tidak terstruktur, seperti dokumen pada PDF, Word, maupun kutipan,

C. Klasifikasi Teks

Klasifikasi teks atau kategorisasi teks merupakan proses yang secara otomatis menempatkan dokumen teks ke dalam suatu kategori berdasarkan isi dari teks tersebut. Klasifikasi teks terdiri dari bermacam-macam metode yang terbagi menjadi 3 jenis, yaitu klasifikasi teks berbasis statistik contohnya *K-Nearest Neighbor*, *Naive Bayes*, dan *Support Vector Machine*. Yang kedua adalah klasifikasi teks berbasis koneksi, misalnya menggunakan Artificial Neural Network. Kelompok yang terakhir adalah klasifikasi teks berbasis aturan (*rule-based*), seperti *Decision Tree*.

D. Analisis Sentimen

Analisis sentimen atau yang biasa disebut sebagai *opinion mining* merupakan sebuah proses memahami, mengekstrak dan mengolah data berupa teks secara otomatis sehingga didapatkan informasi berupa sentimen yang berasal dari kalimat opini.[7] Analisis sentimen dilakukan untuk melihat pendapat atau opini yang muncul dari suatu entitas atau objek dan mengolahnya sehingga menghasilkan suatu kecenderungan akan makna dari pendapat tersebut apakah bernilai negatif, positif, atau netral.

E. Recurrent Neural Network

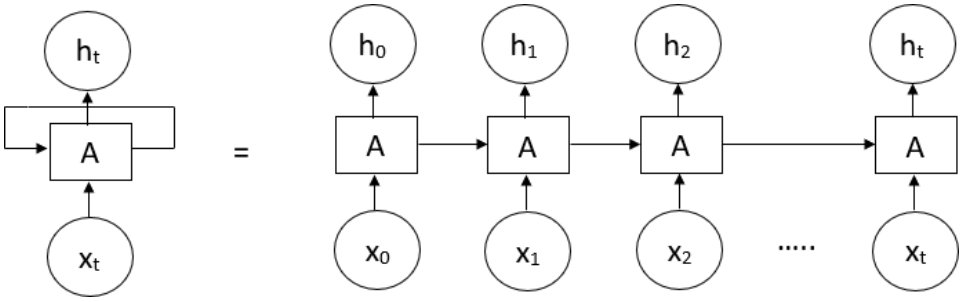
Recurrent Neural Network (RNN) adalah jenis *neural network* yang bekerja sebagai jaringan saraf berulang. Dikatakan jaringan saraf berulang karena nilai *neuron* yang terdapat pada *hidden layer* sebelumnya akan digunakan kembali sebagai data input. Penggunaan *neuron* pada *hidden layer* akan disimpan ke dalam sebuah *layer* yang dinamakan *context layer*. Nilai *neuron* yang terdapat pada *context layer* akan terus diperbarui sampai kondisi RNN terpenuhi.[8] Struktur RNN, mengacu pada gambar 1, memiliki relasi antar unit pada *hidden layer* yang sama yang berguna untuk menyampaikan informasi dari masukan waktu sebelumnya untuk memengaruhi keputusan berikutnya.[9]

Jumlah perulangan dalam RNN berubah menjadi jumlah *layer* pada bentuk yang telah terurai. Bila jumlah *layer* lebih dari dua, maka bentuk tersebut dikenal sebagai *deep network*. Namun, jika jumlah *layer* mencapai lima atau lebih, maka akan menghadapi masalah *vanishing gradient*. Dapat dilihat dalam gambar 2, saat layer sudah mencapai lebih dari 3 *layer,* inputan X_0 dan X_1 akan memiliki jarak yang sangat besar dengan , X_t dan X_{t+1} sehingga ketika h_{t+1} memerlukan informasi yang sesuai dengan X_0 dan X_1, RNN tidak dapat lagi belajar untuk menghubungkan informasi-informasi karena memori yang sudah lama tersimpan tidak akan berguna dengan seiring berjalannya waktu karena akan digantikan dengan memori yang baru.[10]

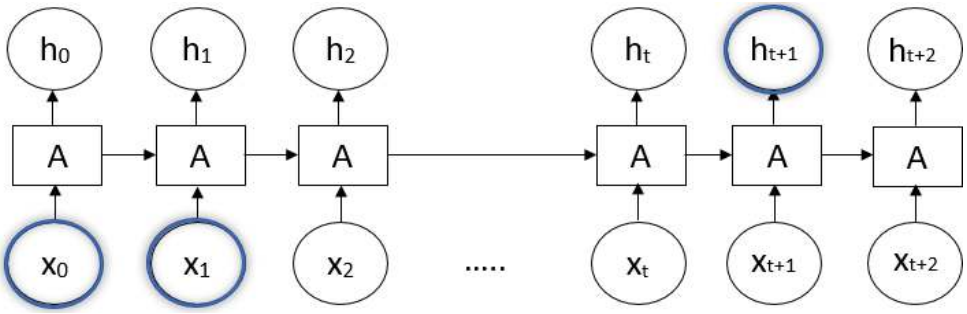
F. Long Short Term Memory

Long Short Term Memory (LSTM) merupakan salah satu jenis pengklasifikasian teks yang berevolusi dari arsitektur RNN, yang pertama kali diperkenalkan oleh Hochreiter dan Schmidhuber.[11] Pada LSTM, nilai eror saat proses *backpropagation* dilakukan melalui waktu dan *layer* akan disimpan, sehingga memungkinkan LSTM untuk menyimpan

p-ISSN : 2301-8402, e-ISSN : 2685-368X , available at : https://ejournal.unsrat.ac.id/index.php/elekdankom



Gambar 1. *Layer* RNN



Gambar 2. Memori RNN

informasi lebih dari 1000 langkah waktu tanpa mengalami permasalahan vanishing gradient.

Hal tersebut terjadi karena LSTM dapat mengatur memori pada setiap masukannya dengan menggunakan *memory cells* dan *gate units.* Informasi yang masuk melalui *memory cells* dan *gate units* hanya akan mengalami perubahan linier sehingga nilai didalamnya tidak gampang berubah.[9]

G. Learning Rate

Learning rate merupakan parameter dari *gradient descent* yang berfungsi untuk mencari nilai *local minima* pada fugsi parametrik. *Learning rate* secara perlahan akan mengoptimalkan nilai perubahan *weight* sehingga menghasilkan nilai *error* yang lebih kecil. Penentuan nilai *leaning rate* dimulai dari 0.0 – 1.0 yang menunjukan besaran langkah yang diambil oleh model untuk melakukan pembelajaran. Nilai *learning rate* terlalu tinggi dapat membuat model melakukan pembelajaran dengan cepat, namun memungkinkan model untuk melewati *local minimal,* sedangkan penggunaan *learning rate* yang terlalu kecil memungkinkan model untuk mendapatkan *weight* yang optimal, namun membutuhkan waktu yang lebih lama untuk dilatih.

H. Dropout

Dropout merupakan proses yang dilakukan untuk mencegah adanya kondisi *overfitting* pada model. *Dropout* bekerja dengan menonaktifkan beberapa *neuron* yang dipilih secara acak dalam *layer* model. Dengan menggunakan *dropout,* unit *neuron* yang dinonaktifkan selama proses *training* akan memaksa *neuron* untuk dapat bekerja secara independen dan tidak terlalu bergantung pada *neuron* lainnya. Rentang nilai *dropout* dimulai dari 0.0 – 1.0.

II. METODE

Penelitian ini diawali dengan mengindentifikasikan sebuah masalah yang sedang terjadi dan perumusan dari masalah tersebut yang akan diteliti. Adapun masalah yang berhasil diidentifikasi adalah bagaimana mengetahui sentimen pengguna Twitter akan suatu topik secara otomatis dan mengagregasi keseluruhan sentimen tersebut. Untuk menyelesaikan masalah tersebut, dirumuskan, perlunya dibangun sebuah sistem berbasis web yang dapat menganalisis sentimen-sentimen dari pengguna Twitter dengan menggunakan metode *Long Short Term Memory*. Setelah dirumuskan masalah, selanjutnya dilakukan tahap studi literatur yang berkaitan dengan analisis sentimen maupun metode *Long Short Term Memory.* Pada tahap pengumpulan data, dilakukan pengumpulan data berupa *tweet* yang melalui proses *scrapping* pada Twitter sesuai dengan *keyword* yang dibutuhkan. Pada tahap perancangan sistem dilakukan perancangan prinsip kerja sistem. Selanjutnya, data *tweet* yang telah didapatkan tersebut dilakukan tahap *preprocessing.* Tahapan ini dilakukan agar data *tweet* yang telah dilakukan siap untuk memasuki tahapan selanjutnya, yaitu pembentukan dan pemilihan model. Dalam tahap ini, dilakukan proses pembentukan model *Long Short Term Memory* dan pemilihan model terbaik untuk menjadi masukan dalam sistem yang dibuat. Setelah didapatkan model *Long Short Term Memory* yang sesuai, selanjutnya dilakukan pengimplementasian sistem berbasis *website* yang dapat melakukan analisis sentimen dari pengguna sosial media Twitter secara *real-time.* Selanjutnya dilakukan evaluasi untuk menguji keakuratan dari model yang telah dibentuk menggunakan *confusion matrix, recall,* dan *precision*.

A. Data Yang Digunakan

Data yang digunakan dalam penelitian ini adalah data *tweet* berbahasa Indonesia yang diambil dari media sosial Twitter. *Tweet* yang dikumpulkan dari Twitter menggunakan *library Tweepy* dengan proses autentikasi oleh Twitter API. *Tweet* yang diambil untuk penelitian bedasarkan 3 topik utama, yaitu tokoh, pariwisata, dan produk.

Dari keseluruhan topik tersebut, didapatkan *tweet* mengenai topik tokoh sebanyak 1.500 *tweet,* topik produk sebanyak 1.488 *tweet*, dan topik pariwisata sebanyak 1.417 *tweet.* Total keseluruhan *tweet* yang didapatkan adalah sebanyak 4.405 *tweet*. Persebaran jumlah sentimen dalam *tweet* yang telah dilabeli dengan sentimen masing-masing adalah *tweet* yang bersentimen negatif berjumlah 1.441 *tweet*, *tweet* bersentimen netral berjumlah 1.501 *tweet,* dan *tweet* bersentimen positif berjumlah 1.463 *tweet.*

B. Prinsip Kerja Sistem

Pada tahap pertama prinsip kerja sistem, pengguna akan memasukan *keyword* yang ingin diketahui klasifikasi sentimennya dalam sistem. *Keyword* tersebut selanjutnya akan menjadi masukan dalam proses *scrapping* atau pengambilan data *tweet* pada sosial media Twitter menggunakan Twitter API. Hasil *scrapping* yang berupa kumpulan data *tweet* yang sesuai dengan *keyword,* selanjutnya akan melalui tahap *preprocessing* seperti *cleansing, case folding, tokenizing,* dan *stopwords removal.* Setelah melewati *pre-processing,* selanjutnya data *tweet* akan melalui tahap *word embedding/word vector* yang merupakan proses pemetaan kata menjadi vektor agar dapat diketahui persamaan semantik dari tiap kata dan siap untuk menjadi masukan dalam proses klasifikasi/prediksi yang hanya bisa menerima *input* berupa angka. Pada tahap *word embedding/word vector,* digunakan *tokenizer word index* yang telah didapatkan dalam proses *training* sebelumnya dan disimpan dalam file berformat *Pickle*. Setelah proses pemetaan selesai, data *tweet* yang sudah berupa angka akan dilakukan proses klasifikasi/prediksi menggunakan model *Long Short Term Memory* yang telah didapatkan. Hasil dari proses klasifikasi/prediksi adalah kelas sentimen dari data *tweet* yang telah terlabeli secara otomatis, seperti kelas negatif, positif, dan netral. Gambar 3 menggambarkan prinsip kerja sistem yang dibuat.

C. Pre-processing

Pre-processing merupakan proses dimana *tweet* yang diambil dari Twitter akan dibersihkan dari hal-hal yang tidak diperlukan sehingga mendapat data yang berkualitas dan sesuai dengan keinginan. *Pre-processing* terbagi dalam beberapa tahap, yaitu yaitu pembersihan data*, case folding, tokenizing* dan *stopword removal.*

1) Pembersihan Data

Tahapan dimana *tweet* yang diambil akan dihilangkan URL, *retweet, link* gambar, dan simbol-simbol atau karakter yang ada. Pembersihan dilakukan dengan *tools* Python.

2) Case Folding

Proses *case folding* merupakan tahap dimana *tweet* yang telah selesai dibersihkan*,* akan diubah menjadi huruf kecil.

3) Tokenizing

Pada tahap ini *tweet* yang sudah melalui tahap *case folding* akan dibagi menjadi potongan-potongan kecil perkata yang disebut token.

4) Stopword Removal

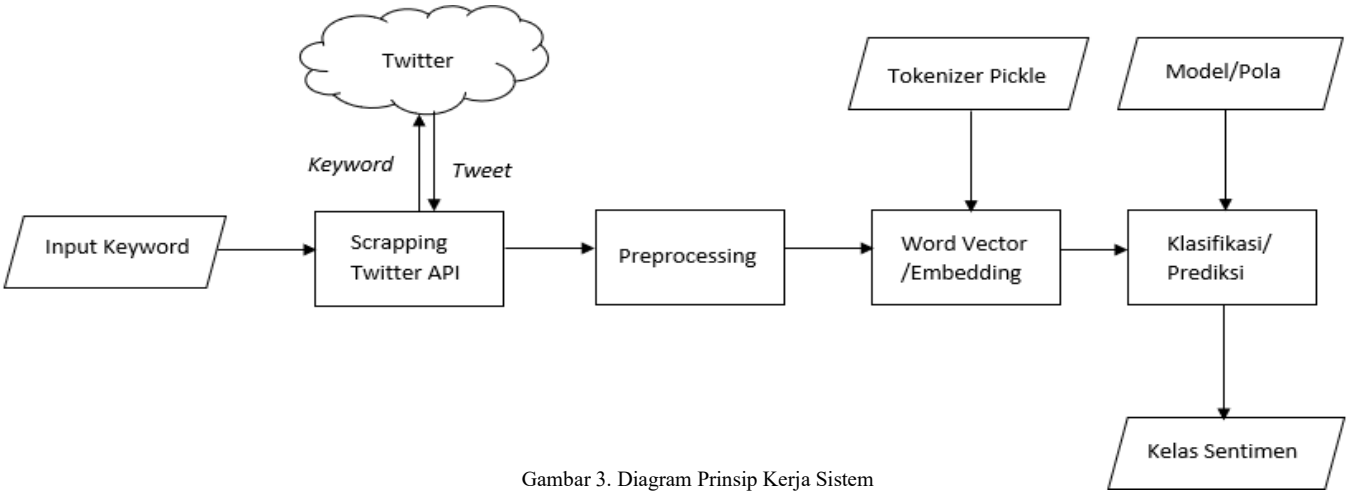
Pada tahap ini *tweet* yang telah berbntuk token kata akan dihilangkan kata-kata yang memiliki arti tidak terlalu penting. Contoh *stopword* adalah dia, mereka, saya, pada, di, ke, yang, dan lain sebagainya. Sebelum proses *stopword removal* dilakukan, harus dibuat daftar *stopword*(*stoplist*) terlebih dahulu. Jika suatu kata termasuk di dalam *stoplist* maka kata tersebut akan dihapus dari *tweet.*

D. Word Vector/Word Embedding

Setelah kata-kata yang ada pada data *tweet* melalui tahap *preprocessing*, kata-kata tersebut selanjutnya akan diubah menjadi angka dan dibuat pemetaannya atau *word embedding* yang nantinya akan menjadi *input* untuk metode klasifikasi *Long Short-Term Memory*.

Word embedding atau *word vector* merupakan suatu metode yang bekerja dengan memetakan kata-kata dalam bentuk vektor. Kata-kata yang saling berkaitan secara *semantic* akan dipetakan dalam nilai vektor yang berdekatan.. Hasilnya adalah, kata-kata yang memiliki kesamaan *semantic* akan berada di dalam satu area dengan kata yang memiliki kesamaan dengan kata tersebut.

Proses *word vector* dalam penelitian ini menggunakan salah satu fitur yang disediakan oleh *Keras* yang bernama *Embedding.*



Gambar 3. Diagram Prinsip Kerja Sistem

Berikut adalah beberapa tahap saat melakukan proses *word vector:*

1) Indeks Kata

Data yang telah melalui proses *preprocessing* dan telah terbentuk *list* akan dilakukan pengindeksan yang merupakan proses mengubah kata unik menjadi angka yang nantinya akan mereprentasikan kata tersebut sebelum masuk dalam tahap *word embedding.*

2) Padding

Setelah didapatkan indeks dari tiap kata, selanjutnya adalah melakukan *padding* untuk tiap kalimat yang ada. *Padding* dilakukan untuk menyamakan panjang *list* dari tiap kalimat agar bisa menjadi masukan saat proses klasifikasi. Dalam pembuatan *padding,* harus di deklarasikan terlebih dahulu panjang maksimal dari seluruh vektor kalimat yang ada, Dengan menggunakan *padding, list* kalimat yang panjangnya tidak sampai dengan nilai yang ditentukan akan ditambahkan nilai 0 didepannya.

3) Word Embedding

Setelah pembentukan indeks kata dan *padding* pada *list* kalimat*, list* kalimat tersebut dapat diubah menjadi vektor untuk diketahui persamaan *semantic* antara kata. Pembentukan vektor tersebut dapat dilakukan dengan proses word embedding *yang merupakan* layer *pertama dalam proses* training.

E. Pembentukan dan Pemilihan Model

Pembentukan model *Long Short Term Memory* atau LSTM dilakukan sebelum proses klasifikasi/prediksi, dimana akan didapatkan terlebih dahulu model dengan melalukan *training* pada dataset yang telah dilabeli secara manual.

Proses *training* berfungsi untuk menjadi bahan pembelajaran model sehingga model dapat melakukan proses prediksi dengan baik. Pembentukan model dan proses *training* dilakukan dalam Jupyter Notebook. Alur yang dilalui saat proses *training* dapat dilihat pada gambar 4.

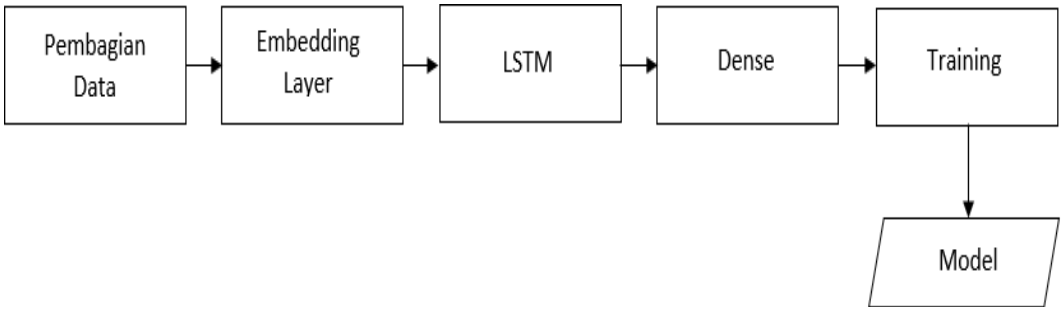
Data yang telah diubah menjadi angka dan didapatkan vektornya selanjutnya akan dibagi dalam 2 bagian, yaitu data *training* dan data *testing.* Jumlah data *training* yang digunakan adalah 70% dari total data, yaitu 3.083 data dan data *testing* sejumlah 30% dari total data, yaitu 1.322. Setelah didapatkan data *training* dari hasil pembagian, data tersebut selanjutnya akan digunakan sebagai masukan dari proses *training* pada lapisan pertama yaitu *Embedding*.

Dalam proses *training* LSTM untuk mencari model, digunakan *library Keras* dengan *Sequential API* sebagai model pembangunnya. Arsitektur LSTM terdapat 3 lapisan utama, yaitu lapisan *Embedding,* LSTM, dan *Dense.* Gambar 5 merupakan kode sumber untuk membuat arsitektur LSTM.

III. HASIL DAN PEMBAHASAN

A. Pemilihan Model

Pemilihan model merupakan tahapan yang dilakukan untuk memilih model dengan performa terbaik untuk menjadi masukan dalam sistem agar dapat dilakukan proses klasifikasi. Pada tahapan pemilihan model ini digunakan 4.405 data *tweet* yang telah dikumpulkan. Data *tweet* tersebut telah dilabel sentimennya secara manual dengan kategori sentimen negatif, netral, dan positif. Setelah itu, data *tweet* akan dibagi sebesar 70% untuk data *training* dan 30% untuk data *testing.*



Gambar 4. Diagram Pembentukan Model

```
MAX_NB_WORDS = 9799
MAX_SEQUENCE_LENGHT = 47
EMBEDDING_DIM = 50
model = Sequential()
model.add(Embedding(MAX_NB_WORDS+1, EMBEDDING_DIM, input_legth=
                    X_train.shape[1]))
model.add(LSTM(300, dropout=0.9)
model.add(Dense(3, activation='softmax'))
opt = keras.optimizers.Adam(learning_rate=0.0001)
model.compile(loss='categorical_crossentropy', optimizer=opt,  metr
cs=['accuracy'])
model.summary()
```

Gambar 5 Potongan Kode Sumber Arsitektur LSTM

Total dari data *training* yang telah dibagi adalah 3.083 data dan data *testing* 1.322. Setelah dilakukan pembagian data, selanjutnya data akan melalui tahap *preprocessing, word vector,* dan klasifikasi.

Pada metode *Long Short Term Memory,* akan dilakukan beberapa pengujian parameter seperti pengujian jumlah *neuron, learning rate* ADAM*,* dan *dropout* pada lapisan LSTM untuk diketahui parameter yang memiliki performa terbaik. Proses pengujian akan dilakukan dengan data validasi yang berjumlah 10% dari total data *training* yang telah dibagi sebelumnya.

Dalam pengujian yang pertama, dilakukan pengujian jumlah *neuron* 100, 200, 300, dan 400 pada lapisan LSTM yang dapat dilihat pada tabel I. Dari proses pengujian *neuron* pada lapisan LSTM, nilai *accuracy* dan *loss* pada data *validation* memiliki peningkatan fluktuatif yang tidak bergantung pada semakin tinggi-rendahnya nilai *neuron* LSTM yang digunakan. Hal ini terjadi karena tidak terdapatnya ketentuan khusus dalam pemilihan jumlah *neuron* LSTM karena beragamnya variasi data yang dimiliki model. Penggunaan jumlah *neuron* 300 memiliki nilai akurasi pada data *validation* yang paling tinggi, yaitu 74.1% dan nilai *loss* yang paling rendah yaitu 1.984, sehingga digunakan jumlah *neuron* tersebut untuk selanjutnya digunakan pada proses pemilihan model selanjutnya berdasarkan nilai *learning rate* ADAM yang dapat dilihat pada tabel II.

Pada proses pengujian ini, digunakan pengujian *learning rate* ADAM 0.01, 0.001, 0.0001, dan 0.00001 dengan penggunaan jumlah *neuron* LSTM sebesar 300. Hasil dari pengujian jumlah *learning rate* dapat dilihat pada tabel II. Seperti yang dilihat dalam tabel II, saat penggunaan *learning rate* yang paling kecil, nilai *loss* pada data *validation* akan berkurang, namun nilai *accuracy* juga menurun. Saat penggunaan *learning rate* yang tinggi, *accuracy* data *validation* menurun dan nilai *loss* semakin tinggi, dan akhirnya model tidak dapat melakukan pembelajaran dengan baik, sehingga dicari nilai dengan performa terbaik yang paling cocok dengan model yang dibangun, yaitu *learning rate* 0.0001 dan *learning rate* 0.001. Penggunaan *learning rate* 0.0001 menghasilkan nilai *accuracy* pada data *validation* yang tinggi berupa 73.7%, dan nilai *loss* cenderung rendah dibandingkan dengan yang lainnya yaitu berupa 1.065. Penggunaan *learning rate* 0.001 menghasilkan nilai *accuracy* yang paling tinggi pada data *validation,* namun memiliki nilai *loss* yang cukup tinggi juga, sehingga digunakan kedua nilai *learning rate* tersebut untuk selanjutnya dilakukan pengujian *dropout* pada lapisan LSTM untuk nantinya dibandingkan dan dipilih sebagai parameter terbaik.

Pengujian nilai *dropout* dimulai pada nilai 0.9, 0.8, 0.7, dan 0.6 pada lapisan LSTM. Pengujian *dropout* dilakukan pada jumlah *neuron* LSTM yaitu sebesar 300 dan *learning rate* 0.0001 yang dapat dilihat pada tabel III, dan *learning rate* 0.001 yang dapat dilihat pada tabel IV.

Dari proses pengujian *dropout* rentang 0.9 - 0.6 pada lapisan LSTM menggunakan *neuron* 300 dan *learning rate* 0.0001, didapatkan hasil terbaik pada penggunaan *dropout* yang berjumlah 0.9. Penggunaan jumlah *dropuout* 0.9 memiliki performa terbaik, dimana nilai *accuracy* pada data *validation* memiliki nilai tertinggi yaitu 77.6% dan nilai *loss* yang paling kecil, yaitu 0.567. Hal ini juga menunjukan bahwa model melakukan pembelajaran dengan baik karena nilai *accuracy* dan *loss* pada data *validation* dan *training* memiliki selisih yang paling kecil diantara penggunaan nilai *dropout* lainnya. Penggunaan *dropout* rentang 0.9 - 0.6 pada lapisan LSTM menggunakan *neuron* 300 dan *learning rate* 0.001 memiliki hasil terbaik juga pada penggunaan *dropout* 0.9, dimana nilai *accuracy* pada data *validation* mendapatkan nilai terbesar yaitu 74.4% dan *loss* terkecil senilai 0.816. Hasil dari kedua perbandingan tersebut dapat dilihat pada tabel V.

Berdasarkan kedua perbandingan tersebut, dapat diketahui bahwa nilai *dropout* 0.9 melakukan performa terbaik pada penggunaan *learning rate* 0.0001, dimana model memiliki nilai *accuracy* yang paling tinggi yaitu 77.6%, dan *loss* paling kecil yaitu 0.567. Penggunaan *dropout* 0.9 memiliki performa yang baik pada penggunaan kedua *learning rate* 0.0001 dan 0.001 dibandingkan dengan nilai *dropout* lainnya. Hal ini dikarenakan semakin besarnya nilai *dropout* akan membuat model semakin sederhana sehingga mencegah adanya kondisi *overfitting,* dimana model terlalu kompleks dalam melakukan pembelajaran pada data *training* sehingga gagal menggeneralisir data baru yang belum pernah dilihat oleh model sebelumnya.

Setelah dilakukan perbandingan jumlah *neuron, learning rate,* dan *dropout* didapatkan hasil performa maksimal dari model yang melakukan pembelajaran dengan baik pada *data training* dan berhasil melakukan klasifikasi pada data *validation* dengan *accuracy* tertinggi dan *loss* terendah saat penggunaan jumlah *neuron* pada lapisan LSTM sebesar 300, *learning rate* sebesar 0.0001, dan jumlah *dropout* pada lapisan LSTM sebesar 0.9. Grafik akurasi dari hasil *training* model yang menampilan perbandingan antara performa data *training* dan data *validation* dapat dilihat pada gambar 6 dan gambar 7. Grafik pada gambar 6 menunjukan dapat bahwa data *validation* mengalami peningkatan akurasi yang cukup stabil dengan tingkat akurasi sebesar 77.6% di *epoch* ke 45 dan data *training* mencapai nilai 84.2% di *epoch* ke 45. Sedangkan, grafik nilai *loss* yang menampilkan perbandingan performa antara data *training* dan data *validation* dalam proses *training* dapat dilihat pada gambar 7. Pada gambar 7, nilai *loss* pada data *validation* mengalami penurunan yang stabil dengan mencapai nilai 0.567 pada *epoch* ke 45 dan *loss* pada data *training* mencapai nilai 0.405 pada *epoch* ke 45.

Berdasarkan kedua grafik tersebut dapat disimpulkan bahwa model melakukan pembelajaran dengan cukup baik pada penggunaan *neuron* 300, *learning rate* 0.0001, dan *dropout* 0.9, sehingga dipilih parameter-parameter tersebut dalam pembangunan model *Long Short Term Memory*.

TABEL I
PENGUJIAN *NEURON* LSTM

Neuron	Data Training		Data Validation	
	Accuracy %	Loss	Accuracy %	Loss
100	99.6%	0.007	70.8%	2.228
200	99.7%	0.003	70.2%	2.005
300	99.8%	0.004	74.1%	1.984
400	99.9%	0.001	72.8%	2.160

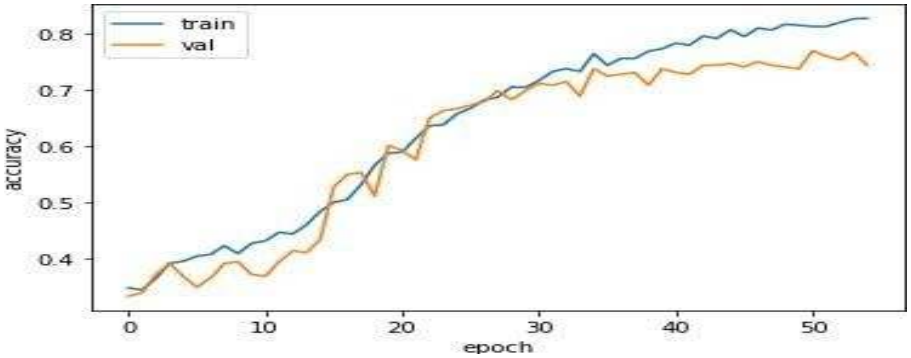
.

TABEL II
PENGUJIAN LEARNING RATE ADAM

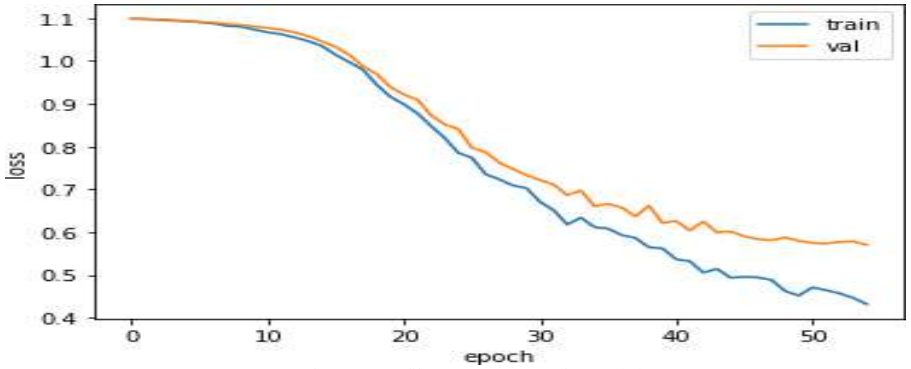
Learning Rate	Neuron	Data Training		Data Validation	
		Accuracy %	Loss	Accuracy %	Loss
0.00001	300	76.8%	0.686	55.3%	0.927
0.0001		99.5%	0.016	73.7%	1.065
0.001		99.6%	0.003	75.0%	1.657
0.01		99.8%	0.002	67.9%	2.619

TABEL III
PENGUJIAN *DROPOUT* PADA *LEARNING RATE* 0.0001

Drop out	Neuron	*Learning Rate*	Data Training		Data Validation	
			Accuracy %	Loss	Accuracy %	Loss
0.9	300	0.0001	84.2%	0.405	77.6%	0.567
0.8			92.4%	0.205	76.3%	0.620
0.7			95.4%	0.125	74.1%	0.752
0.6			97.8%	0.071	76.3%	0.816



Gambar 6. Grafik Akurasi *Training* dan *Validation*



Gambar 7. Grafik *Loss Training* dan *Validation*

TABEL IV
PENGUJIAN *DROPOUT* PADA *LEARNING RATE* 0.001

Drop out	Neuron	*Learning Rate*	Data Training		Data Validation	
			Accuracy %	Loss	Accuracy %	Loss
0.9	300	0.001	96.5%	0.098	74.4%	0.816
0.8			99.0%	0.024	74.4%	1.141
0.7			99.6%	0.015	73.4%	1.148
0.6			99.6%	0.008	73.4%	1.590

TABEL V
PERBANDINGAN *DROPOUT* PADA *LEARNING RATE* 0.0001 DAN 0.001

Drop out	Neuron	*Learning Rate*	Data Training		Data Validation	
			Accuracy %	Loss	Accuracy %	Loss
0.9	300	0.0001	84.2%	0.405	77.6%	0.567
0.9	300	0.001	96.5%	0.098	74.4%	0.816

B. Evaluasi Sistem

Setelah didapatkan parameter terbaik dari hasil pengujian menggunakan data validasi, selanjutnya dilakukan pemodelan arsitektur dan *testing* model LSTM menggunakan parameter-parameter terbaik yang telah diuji sebelumnya untuk dapat dilakukan evaluasi.

Model yang terbentuk selanjutnya akan dilakukan evaluasi dan prediksi menggunakan data *testing* untuk diketahui keakuratannya. Hasil prediksi model akan terbentuk dalam 3 jenis kelas sentimen, yaitu sentimet negatif, positif, dan netral. Kelas sentimen yang telah diprediksi tersebut akan dihitung jumlah kelas yang merupakan total hasil prediksi dan dibandingkan dengan total jumlah kelas yang aktual menggunakan *Confusian Matrix* seperti pada tabel VI.
Tabel *Confusian Matrix* dibagi dalam 2 bagian utama yaitu nilai Hasil Prediksi dan Aktual seperti pada tabel VII.

Setelah dilihat dibandingkan jumlah kelas dengan *Confusian Matrix*, selanjutnya akan diukur performa dari

model dengan melihat hasil dari *accuracy*, *precision,* dan *recall* yang dapat dilihat pada tabel VI.

Nilai *Accuracy* yang didapatkan dengan model yang menggunakan jumlah *neuron* sebesar 300, *learning rate* 0.0001, dan *dropout* 0.9 adalah 77%, sehingga dapat diketahui bahwa model melakukan prediksi dengan tingkat keakuratan cukup baik. Nilai *Precision* dari kelas Negatif dan Positif, masing-masing mendapatkan hasil sebesar 80% dan 86% yang berarti model dapat melakukan prediksi dengan benar dari keseluruhan kelas Negatif dan Positif yang berhasil diprediksi dengan baik. Namun, nilai *Precision* pada kelas Netral memiliki performansi yang cukup rendah yaitu sebesar 68%, hal ini dapat diartikan bahwa model tidak terlalu baik dalam memprediksi dengan benar pada kelas Netral. Performansi *Recall* pada kelas Negatif, Netral, dan Positif memiliki nilai yang cukup baik yaitu 73%, 80%, dan 76%, hal ini berarti model melakukan prediksi dengan benar dari keseluruhan data aktual pada ketiga kelas tersebut dengan cukup baik.

Berdasarkan penelitian yang dilakukan, saat model akan memprediksi data baru untuk diklasifikasikan, model menyesuaikannya dengan fitur-fitur kata yang telah direpresentasikan dalam angka atau *word index* sebelumnya dalam proses *training*, sehingga saat model mendapatkan fitur kata baru yang belum pernah muncul dalam *word index,* model tidak akan dapat merepresentasikannya. Hal ini berarti, perlu ditambahkan jumlah data *training* sehingga didapatkan lebih banyak variasi fitur kata dalam *word index.*

IV. KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan hasil pengujian yang telah dilakukan, dapat ditarik kesimpulan bahwa penerapan metode *Long Short Term Memory* dapat melakukan klasifikasi sentimen *tweet* pengguna Twitter secara otomatis dengan tingkat akurasi 77% saat menggunakan data *training* sebanyak 3.083 data dan data *testing* sebanyak 1.322 data. Model apat melakukan analisis sentimen dan mengelompokannya ke dalam 3 jenis kelas, yaitu kelas negatif, netral, dan positif. Model memiliki performa terbaik dengan menggunakan 300 *neuron* LSTM, *learning rate* ADAM senilai 0.0001, dan *dropout* 0.9 pada lapisan LSTM.

B. Saran

Dalam penelitian ini masih terdapat beberapa hal yang dapat ditingkatkan dan dikembangkan lagi, seperti menambahkan lebih banyak data untuk meningkatkan nilai akurasi dari model dan menggunakan metode *word embedding* lainnya seperti Word2Vec atau FastText.

TABEL VI
CONFUSIAN MATRIX PARAMETER TERBAIK

Aktual	Hasil Prediksi		
	Negatif	Netral	Positif
Negatif	302	96	14
Netral	54	373	40
Positif	23	83	337

TABEL VII
PERFORMANSI MODEL

Kelas	*Accuracy*	Performansi *Precision*	*Recall*
Negatif	77%	80%	73%
Netral		68%	80%
Positif		86%	76%

V. KUTIPAN

[1] O. Harnantyo, "Analisis Sentimen Tempat Wisata Di Yogyakarta Menggunakan Metode Recurrent Neural Network Dengan Long Short Term Memory," Sekolah Tinggi Manajemen Informatika Dan Komputer Akakom, 2019.

[2] 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, Doi: 10.22146/Ijccs.41236.

[3] W. Umboh, "Analisis Sentimen Pengguna Social Media Terhadap Objek Wisata," Universitas Sam Ratulangi, 2018.

[4] A. Faadilah, "Analisis Sentimen Pada Ulasan Aplikasi Tokopedia Di Google Play Store Menggunakan Metode Long Short Term Memory," 2020.

[5] Elisabeth, "Perbandingan Sentimen Analisis Terhadap Brand Indomie Menggunakan Naïve Bayes & Long Short Term Memory (Lstm)," Universitas Multimedia Nusantara, 2018.

[6] R. Talib, M. Kashif, S. Ayesha, And F. Fatima, "Text Mining: Techniques, Applications And Issues," *Int. J. Adv. Comput. Sci. Appl.*, Vol. 7, No. 11, Pp. 414–418, 2016, Doi: 10.14569/Ijacsa.2016.071153.

[7] I. F. Rozi, S. Hadi, And E. Achmad, "Implementasi Opinion Mining (Analisis Sentimen) Untuk Ekstraksi Data Opini Publik Pada Perguruan Tinggi," *J. Eeccis*, Vol. 6, No. 1, Pp. 37–43, 2012.

[8] A. A. Rizal And S. Soraya, "Multi Time Steps Prediction Dengan Recurrent Neural," Vol. 18, No. 1, Pp. 115–124, 2018.

[9] A. Tedja, "Implementasi Long Short-Term Memory Untuk Sistem Speaker Recognition Menggunakan Spektrogram," Universitas Multimedia Nusantara, 2018.

[10] M. Wildan, P. Aldi, And A. Aditsania, "Analisis Dan Implementasi Long Short Term Memory Neural Network Untuk Prediksi Harga Bitcoin," *E-Proceeding Eng.*, Vol. 5, No. 2, Pp. 3548–3555, 2018.

[11] S. Hochreiter And J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, Vol. 9, No. 8, Pp. 1735–1780, 1997, Doi: 10.1162/Neco.1997.9.8.1735.

TENTANG PENULIS

Penulis bernama lengkap Cassey K.N. Paputungan. Lahir di Manado 19 April 2001. Penulis tinggal di Malalayang, Manado, Sulawesi Utara. Penulis menempuh pendidikan di SMA Negeri 9 BINSUS Manado (2014-2016). Di tahun 2016, penulis melanjutkan pendidikan di Universitas Sam Ratulangi dan mengambil Program Studi S-1 Teknik Informatika di Jurusan Teknik Elektro Fakultas Teknik. Selama masa perkuliahan, penulis tergabung dalam organisasi kemahasiswaan yaitu, Badan Tadzkir Fakultas Teknik (BTFT), LPM INOVASI Unsrat, Unsrat IT Community (UNITY), dan Badan Tadzkir Unsrat (BTU).