PENGGUNAAN CONVOLUTIONAL NEURAL NETWORK UNTUK ANALISIS SENTIMEN OPINI LINGKUNGAN HIDUP KOTA DEPOK DI TWITTER

[1]Putri Lannidya Parameswari, [2]Prihandoko
[1,2]Fakultas Teknologi Industri, Universitas Gunadarma
Jl. Margonda Raya No. 100, Depok 16424, Jawa Barat
[1]putrilannidya@student.gunadarma.ac.id, [2]pri@staff.gunadarma.ac.id

Abstrak

Lingkungan hidup Kota Depok seperti sungai, hutan, udara, dan sebagainya perlu untuk dijaga dan dikelola dengan baik. Hal ini dapat dibantu dengan membuat program analisis sentimen terhadap lingkungan hidup Kota Depok. Program ini dibuat dengan tujuan untuk memberikan alternatif solusi kegiatan evaluasi kondisi dan pengelolaan lingkungan hidup Kota Depok. Metode penelitian yang digunakan adalah metode Cross-Industry Standard Process for Data Mining (CRISP-DM). Dengan metode ini, tahapan penelitian yang dilakukan adalah pengumpulan data, pemahaman bisnis, pemahaman data, data preprocessing, pemberian label data, pemodelan, evaluasi, dan deployment. Analisis sentimen dilakukan terhadap opini masyarakat yang ada pada Twitter atau disebut dengan tweet. Data tweet yang terkumpul kemudian diolah dan dianalisis sentimennya menggunakan metode Convolutional Neural Network (CNN). Metode CNN ini diterapkan pada program sebagai metode klasifikasi. Pelatihan model menggunakan 8.975 data latih. Hasil uji coba model terhadap 3.051 data uji dengan epoch sebanyak 100 menunjukkan akurasi sebesar 86%. Program analisis sentimen ini dibuat menggunakan bahasa pemrograman Python. Website dengan kerangka kerja Flask berhasil dibuat sebagai representasi program untuk memudahkan pengguna melakukan analisis sentimen.

Kata Kunci: *Analisis Sentimen, Convolutional Neural Network, Deep Learning, Twitter, Lingkungan Kota Depok*

Abstract

Environment of Depok City such as rivers, forests, air, and many others must be maintained and managed properly. This can be helped by creating a sentiment analysis program for the environment of Depok City. This program was created with the aim of providing alternative solution for evaluating condition and management of Depok City's environment. This research method is Cross-Industry Standard Process for Data Mining (CRISP-DM) method. By using this method, the steps of this research are data collecting, business understanding, data understanding, data preprocessing, data labeling, modeling, evaluation, and deployment. Sentiment analysis was carried out on public opinion in Twitter or usually known as tweet. The collected tweet data was further being processed and analyzed for its sentiment using the Convolutional Neural Network (CNN) method. This CNN method was applied to the program as a classification method. The model's training used 8,975 training data. The result of model testing with 100 epoch on 3,051 test data showed 86% accuracy. Sentiment analysis of this program was created using Python programming language. A website with Flask framework has been successfully created as the program representation to make it easier for users to perform sentiment analysis.

Keyword: *Sentiment Analysis, Convolutional Neural Network, Deep Learning, Twitter, Depok City Environment*

PENDAHULUAN

Menurut Undang-undang No. 23 Tahun 1997, lingkungan hidup adalah kesatuan ruang dengan semua benda, daya, keadaan, dan makhluk hidup, termasuk manusia dan perilakunya, yang mempengaruhi kelangsungan perikehidupan dan kesejahteraan manusia serta makhluk hidup lain [1]. Lingkungan hidup Kota Depok mencakup elemen geografis, seperti sungai, situ, kawasan terbuka hijau, udara, air, tanah, dan sebagainya. Permasalahan terkait lingkungan hidup di Kota Depok beragam, seperti banjir, penurunan kualitas udara, serta pengurangan kawasan terbuka hijau. Permasalahan tersebut perlu diatasi dengan berbagai strategi, salah satunya yaitu evaluasi kondisi elemen-elemen lingkungan hidup beserta pengelolaan yang telah ada. Evaluasi ini dapat dilakukan salah satunya dengan meninjau opini masyarakat.

Analisis sentimen dapat dilakukan untuk melihat pendapat atau kecenderungan pandangan masyarakat terkait pengelolaan dan kondisi lingkungan hidup Kota Depok. Analisis sentimen adalah studi komputasional dari opini-opini orang, sentimen, dan emosi melalui entitas dan atribut yang dimiliki yang diekspresikan dalam bentuk teks [2]. Analisis sentimen masyarakat terhadap lingkungan Kota Depok melalui opini pada media sosial Twitter dapat dilakukan dengan menciptakan sebuah program yang mampu melakukan analisis secara otomatis. Metode *Convolutional Neural Network* (CNN) dapat diterapkan sebagai algoritma pembelajaran mesin untuk program analisis sentimen. *Convolution Neural Network* (CNN) adalah jenis khusus dari *feed-forward neural network* yang dimana lapisan tersembunyi atau *hidden layer* yang memiliki serangkaian operasi konvolusi dan lapisan *pooling* (lapisan pengurangan parameter) [3].

Penelitian mengenai analisis sentimen menggunakan metode CNN sejauh ini telah dilakukan. Budi M. Mulyo dan Dwi H. Widyantoro dalam penelitian berjudul "*Aspect-Based Sentiment Analysis Approach with* CNN" mengembangkan model CNN dengan melakukan penyeleksian data pelatihan yang terbaik untuk meningkatkan ketepatan analisis sentiment. Hasil pengujian membuktikan CNN-T bekerja rata-rata lebih baik dibandingkan dengan CNN biasa. F-1 *score* yang didapatkan sebesar 71%. [4]. Moch. Ari Nasichuddin, dkk dalam penelitian berjudul "*Performance Improvement Using CNN for Sentiment Analysis*" memodifikasi ukuran filter menjadi kecil untuk meningkatkan ketepatan dan kecepatan analisis sentimen [5]. Faiz Adil Khatami, dkk dalam penelitian berjudul "*Analisis Sentimen Terhadap Review Aplikasi Layanan E-commerce Menggunakan Metode Convolutional Neural Network*" memodifikasi aspek seperti *epoch* pada model CNN untuk meningkatkan ketepatan analisis. Dengan mengubah partisi data, *learning rate*, *batch size*, dan *epoch*, rata-rata hasil akurasi yang didapatkan adalah sebesar 85% [6].

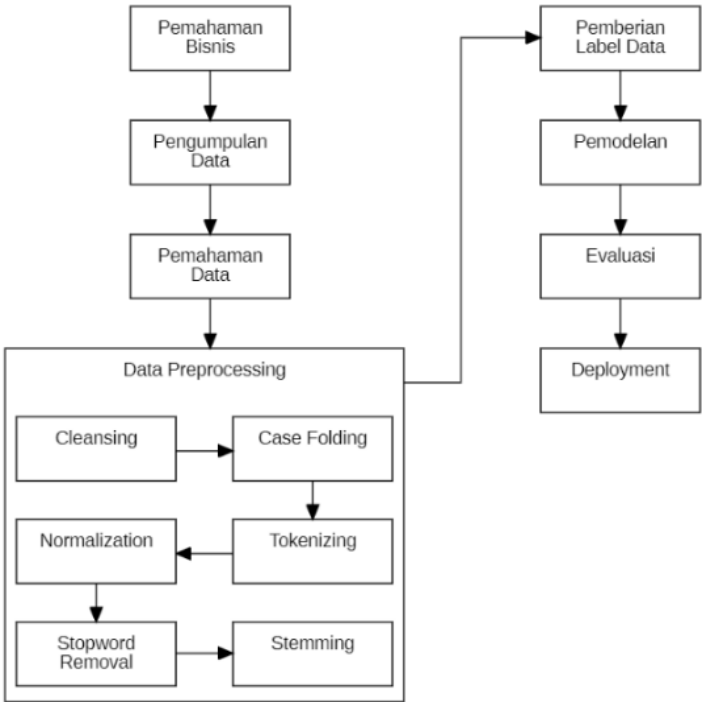
Peneliti dengan implementasi ilmu komputasi yang berkaitan bertujuan untuk mengembangkan aplikasi analisis sentimen pengguna Twitter terhadap lingkungan hidup Kota Depok dengan menggunakan metode klasifikasi CNN, sehingga dapat dilakukan secara otomatis dan mengetahui efektivitas penerapan metode ini.

METODE PENELITIAN

Penelitian menerapkan metode *Cross-Industry Standard Process for Data Mining* (CRISP-DM) sebagai metodologi *data mining.* Tahapan penelitian dapat dilihat pada Gambar 1.

Tahapan penelitian terdiri dari 8 tahapan yaitu pemahaman bisnis, pengumpulan data, pemahaman data, data *preprocessing*, pemberian label data, pemodelan, evaluasi dan *deployment*. Tahapan pemahaman bisnis adalah tahapan awal untuk memahami bisnis dari penelitian ini, seperti masalah dan tujuan penelitian, serta solusi yang dapat ditempuh untuk mencapai tujuan penelitian. Pemahaman ini memerlukan studi dan referensi dari berbagai sumber termasuk buku dan internet.

Tahapan berikutnya adalah Data tweet dikumpulkan dengan melakukan ekstraksi atau scrapping tweet dari media sosial twitter. Scrapping dilakukan dengan bantuan alat atau tools bernama Twint.



Gambar 1. Tahapan Penelitian

Scrapping data pada penelitian ini mengambil data berupa nomor identitas atau id, nama pemilik akun atau *username*, tanggal posting *tweet*, dan teks *tweet*. Kriteria data yang diambil adalah *tweet* berbahasa Indonesia yang memiliki paduan kata 'Depok' dengan satu atau lebih dari 30 kata yang berkaitan dengan lingkungan. Terdapat dua kali pengambilan data, yaitu set data pelatihan dan uji. Set data pelatihan merupakan *tweet* yang diposting dari tanggal 1 Juni 2019 sampai 1 Januari2021, sedangkan set data uji diposting dari tanggal 2 Januari 2021 sampai 1 Mei 2021. Data yang dikumpulkan dari proses ini sebanyak 16.451 baris data latih dan 3.405 baris data uji.

Tahapan selanjutnya adalah pemahaman data yang dilakukan dengan memeriksa per baris, per kolom, maupun keseluruhan dari data, memeriksa panjang dari setiap baris data untuk memantau ada tidaknya data yang terlampau panjang atau pendek, serta memeriksa fungsi dan jumlah keseluruhan data. Data ini kemudian diproses terlebih dahulu dan diberikan label untuk kemudian digunakan dalam melatih model.

Performa model diuji dan dianalisis pada tahap evaluasi. Kemudian, penelitian dilanjutkan dengan *deployment* untuk membuat *website* sebagai representasi proses dan informasi hasil analisis model analisis sentimen.

Data Preprocessing

Data yang telah dikumpulan kemudian diproses terlebih dahulu. Pemrosesan data mencakup enam tahapan, yaitu *cleansing, case folding, normalization, tokenizing, stopword removal,* dan *stemming* sebagai berikut:

1. *Cleansing* membersihkan data *tweet* dari nama akun atau *username,* angka, kata 'RT', *hashtag, Uniform Resource Locator* (URL), simbol, emoji, dan ruang kosong atau *white space*. Pembersihan ini banyak menggunakan fungsi dari pustaka *regular expression.*
2. *Case folding* merubah setiap karakter huruf pada seluruh data *tweet* menjadi huruf kecil atau non-kapital. Perubahan ini menggunakan fungsi menggunakan atribut *preserve_case* dari pustaka *Natural Language Toolkit* (NLTK).
3. *Normalization* merubah setiap kata pada data *tweet* menjadi kata yang baku. Normalisasi ini menggunakan fungsi menggunakan daftar kata-kata normalisasi dari penelitian yang telah dilakukan oleh Navi Atri Lestari dkk [7].
4. *Tokenizing* memotong atau memecah *string* data *tweet* masukan menjadi token-token per kata. Proses ini memanfaatkan fungsi dari pustaka *Natural Language Toolkit* (NLTK).
5. *Stopword removal* melakukan eliminasi terhadap kata-kata yang kurang penting atau tidak deskriptif. Proses ini memanfaatkan fungsi dari pustaka *Natural Language Toolkit* (NLTK).
6. *Stemming* merubah kata yang memiliki imbuhan menjadi kata dasar. Proses ini memanfaatkan pustaka Sastrawi.

Pemberian Label Data Latih

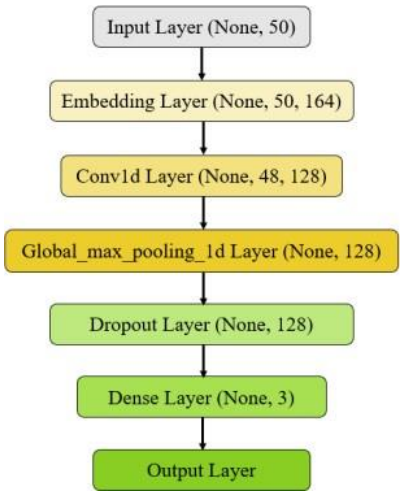
Pemberian label data dilakukan agar setiap baris data *tweet* memiliki label sentimen untuk kepentingan pelatihan model *Convolution Neural Network* (CNN). Proses ini menggunakan korpus Bahasa Indonesia yang diciptakan pada penelitian terdahulu oleh Devid dan Azhari [8]. Setiap kata pada tweet yang terdapat pada suatu berkas korpus, maka nilai atau score kata dijumlahkan dengan score baris tweet yang bersangkutan. Pada penelitian ini terdapat 3 label tweet yaitu positive, negative dan neutral. Jika score positif lebih besar daripada score negatif, maka baris tweet akan diberikan label "positive". Jika score positif lebih kecil dari score negatif, maka baris tweet akan diberikan label "negative". Jika score positif tidak lebih kecil dari negatif juga, maka baris data tweet diberikan label "neutral".

Pemodelan

Pemodelan merupakan tahapan untuk memilih dan menerapkan berbagai teknik dalam membangun model *Convolutional Neural Network* (CNN). Tahapan pemodelan ini melibatkan penetapan set data yang akan digunakan dalam melatih model, perancangan arsitektur model, pelatihan model, dan validasi model.

Set data penelitian terdiri dari data pelatihan sejumlah 11.219 data *tweet* dan data uji sejumlah 3.051 data *tweet*. Data pelatihan dibagi menjadi 8975 data latih dan 2244 data validasi. Data ini kemudian dikonversi menjadi angka bulat dan disusun dalam sebuah urutan agar dapat digunakan pada model *Convolutional Neural Network* (CNN) dengan menggunakan fungsi-fungsi dari pustaka Keras yaitu *fit_on_texts()* dan *texts_to_sequences()*.

Setelah data siap digunakan untuk melatih model, pembuatan model CNN analisis sentimen pun dilakukan. Arsitektur model *Convolutional Neural Network* (CNN) pada penelitian ini dibuat menggunakan fungsi-fungsi dari *Keras* yang dapat dilihat pada Gambar 2.



Gambar 2. Arsitektur Model CNN

HASIL DAN PEMBAHASAN

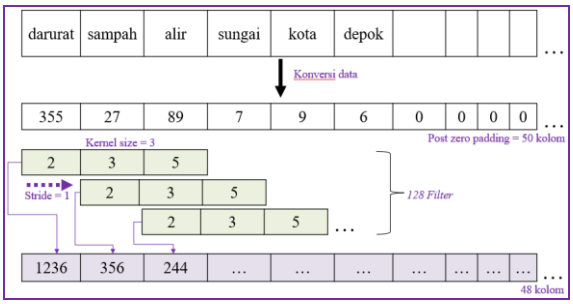
Berdasarkan Gambar 2, model CNN yang dibangun memiliki beberapa lapisan inti yang membentuk model. Informasi mengenai model tersebut didapatkan dari pembentukan model yang kemudian ditampilkan menggunakan *summary().* Arsitektur model CNN dengan lapisan-lapisan yang terbentuk dapat dilihat pada Tabel 1.

Lapisan masukan data pertama kali pada model ini secara implisit dapat dikatakan sebagai *input layer*. Lapisan *embedding* dengan *output shape (None, 50, 64)* menggunakan keluaran dari *input layer*. *None* berarti dimensi lapisan adalah variabel, yaitu ukuran *batch* tidak ditetapkan secara tetap, melainkan ditentukan secara otomatis dalam metode *fit* atau *predict.* Lapisan ini bertugas untuk membuat proyeksi vektor dengan nilai proyeksi atau dimensi sebesar 64 sesuai dengan nilai variabel *embedding dim.*

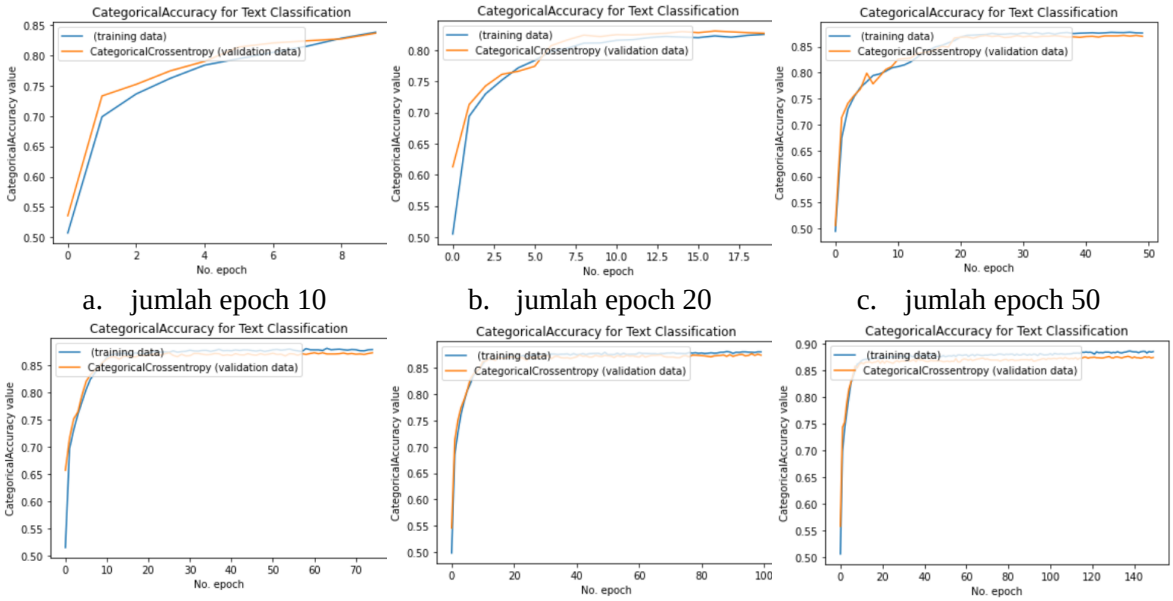
Lapisan *conv1d* merupakan lapisan yang menerima masukan berupa keluaran dari lapisan *embedding*. Lapisan dengan *output shape (None, 48, 128)* berfungsi untuk membuat *kernel* untuk melakukan *filter* (operasi *dot matrix*) dari keluaran lapisan *embedding*. *Kernel* bergeser sebanyak 1 kolom (horizontal) dengan tiga titik data di setiap waktu. Terdapat 128 filter untuk melakukan konvolusi. Dengan ukuran *kernel* sebesar 3 dan pergeseran atau *stride* sebanyak 1, maka dihasilkan *output* vector 1 x 48 . Ilustrasi konvolusi yang terjadi dapat dilihat pada Gambar 3 Pada lapisan ini, *regularizer* bertipe l2 bernilai 0.0005 digunakan untuk menghindari *overfitting* dan fungsi aktivasi *Rectified Linear Unit* (ReLU). Kemudian, hasil nilai maksimum dari perhitungan konvolusi oleh *kernel* diambil oleh lapisan *global_max_pooling_1d* dan ditetapkan sebagai nilai elemen kata terkuat.

Tabel 1. Lapisan Model CNN yang Terbentuk

Layer (type)	Output Shape	Param #
embedding (Embedding)	(none, 50,64)	1280064
conv1d(Conv1D)	(none, 48 128)	24704
global_Max_pooling1d (Global	(none, 128)	0
dropout (Dropout)	(none, 128)	0
dense (Dense)	(none, 3)	387
Total params : 1,305,155		
Trainable params : 1,305,155		
Non trainable params : 0		



Gambar 3. Ilustrasi Konvolusi Model CNN



a. jumlah epoch 10 b. jumlah epoch 20 c. jumlah epoch 50

d. jumlah epoch 75 e. jumlah epoch 100 f. jumlah epoch 150

Gambar 4. *Accuracy* Data Latih dan Validasi Beberapa Pengujian

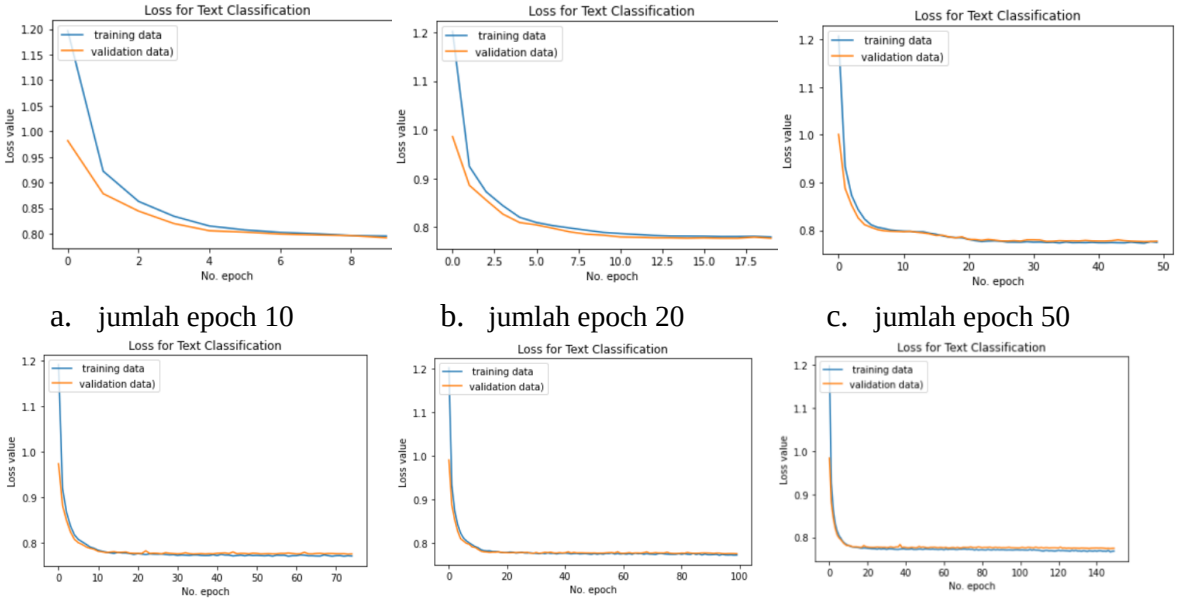
Lapisan *dropout* menerima masukan dari *global_max_pooling* dan membuang nilai *signal* yang kurang baik atau diabaikan, yaitu nilai 0.5 atau 5% pada nilai terbawah. Lapisan *dense* berlaku sebagai *fully connected layer* yang memiliki neuron-neuron terhubung ke lapisan *dropout* tersebut. Lapisan *dense* tersebut menciptakan lapisan tersembunyi atau *hidden layer* dengan *output* 3 menuju *output layer.*

Model yang telah terbentuk tersebut kemudian dilatih. Pelatihan menggunakan 80% dari data pelatihan. Data latih ini berjumlah 8975 data latih. Data tersebut kemudian dikonversi menjadi urutan integer unik yang digunakan dalam melatih model. Pelatihan dengan menerapkan beberapa variasi jumlah *epoch* menghasilkan hasil berbeda yang dapat dilihat pada Gambar 4 dan Gambar 5.

Pada Gambar 4a. jumlah epoch 10, perbedaan terjauh ada pada *epoch* 1 dengan perbedaan masih di bawah 0,05. Akurasi keduanya secara umum meningkat dari *epoch* 2 sampai 10. Pada Gambar 4b. jumlah epoch 20, perbedaan terjauh ada pada *epoch* 0-10

dengan perbedaan masih di bawah 0,05. Akurasi keduanya secara umum meningkat dari *epoch* 1 sampai 8 dan cukup stabil pada *epoch* 9 sampai 20. Pada Gambar 4c. jumlah epoch 50, perbedaan terjauh ada pada *epoch* 7 dan 11 dengan perbedaan masih di bawah 0,05. Akurasi keduanya secara umum meningkat dari *epoch* 1 sampai 20 dan cukup stabil pada *epoch* 21 sampai 50. Pada Gambar 4d. jumlah epoch 75, perbedaan terjauh masih di bawah 0,05. Akurasi keduanya secara umum meningkat dari *epoch* 1 sampai 10 dan cukup stabil pada *epoch* 11 sampai 75. Pada Gambar 4e. Pada jumlah epoch 100, perbedaan terjauh masih di bawah 0,05. Akurasi keduanya secara umum meningkat dari *epoch* 1 sampai 11 dan cukup stabil pada *epoch* 12 sampai 100. Pada Gambar 4f. jumlah epoch 150, perbedaan terjauh masih di bawah 0,05. Akurasi keduanya secara umum meningkat dari *epoch* 1 sampai 12 dan cukup stabil pada *epoch* 13 sampai 150.

Pada Gambar 5a. jumlah epoch 10, perbedaan terjauh ada pada *epoch* 1 dengan perbedaan masih di bawah 0,05. *Loss* keduanya secara umum menurun dari *epoch* 1 sampai 8 dan cukup stabil pada *epoch* 9 sampai 10. Pada Gambar 5b. jumlah epoch 20, perbedaan terjauh ada pada *epoch* 1 dengan perbedaan masih di bawah 0,05. *Loss* keduanya secara umum menurun dari *epoch* 1 sampai 11 dan cukup stabil pada *epoch* 12 sampai 20. Pada Gambar 5c. jumlah epoch 50, *loss* data latih dan data uji secara umum tidak berbeda jauh pada setiap *epoch*.



a. jumlah epoch 10 b. jumlah epoch 20 c. jumlah epoch 50

d. jumlah epoch 75 e. jumlah epoch 100 f. jumlah epoch 150

Gambar 5. *Loss* Data Latih dan Validasi Beberapa Pengujian

Loss keduanya secara umum menurun dari *epoch* 1 sampai 22 dan cukup stabil pada *epoch* 23 sampai 50. Pada Gambar 5d. jumlah epoch 75, *loss* data latih dan data uji secara umum tidak berbeda jauh pada setiap *epoch*. *Loss* keduanya secara umum menurun dari *epoch* 1 sampai 11 dan cukup stabil pada *epoch* 22 sampai 75. Pada Gambar 5e. jumlah epoch 100, *loss* data latih dan data uji secara umum tidak berbeda jauh pada setiap *epoch*. *Loss* keduanya secara umum menurun dari *epoch* 1 sampai 17 dan cukup stabil pada *epoch* 18 sampai 100. Pada Gambar 5f. jumlah epoch 150, *loss* data latih dan data uji secara umum tidak berbeda jauh pada setiap *epoch*. *Loss* keduanya secara umum menurun dari *epoch* 1 sampai 15 dan cukup stabil pada *epoch* 16 sampai 150.

Evaluasi

Evaluasi memberikan gambaran mengenai kinerja model dengan melakukan pengujian menggunakan 3.051 data uji. Pengujian performa model dilakukan dengan menghitung *accuracy, precision, recall,* dan *f1-score* model terhadap set data uji. Evaluasi model diawali dengan memanggil berkas model CNN yang telah disimpan sebelumnya, yaitu 'model_analisis.h5' menggunakan *load_model()* dari Keras dan meng-inisialisasikan ke dalam variabel *new_model*. Setelah itu, model tersebut diterapkan pada *x_test* baru, yaitu data pengujian. Penerapan model terhadap *x_test* uji menggunakan *new_model.predict()*. Hasil prediksi disimpan ke dalam variabel *predict_results*, lalu diterjemahkan ke dalam label sentimen yang tercakup dalam *test_data['pred_sentiment']*.

Pengujian performa model dilakukan dengan menghitung *accuracy, precision, recall,* dan *f1-score* model terhadap set data uji. Perhitungan dilakukan terhadap data dengan membentuk confussion matrix dengan nilai *True Positive* atau TP, *False Positive* atau FP, *True Negative* atau TN, dan *False Negative* atau FN. Matriks tersebut dibentuk menggunakan *confusion_matrix()* dari *scikit-learn*.

confusion_matrix() diberikan parameter berupa data pengujian (*test_data['polarity']*) dan data hasil prediksi model (*test_data ['pred_sentiment']*) untuk kemudian dibentuk sebagai *confusion matrix*. Setelah *confusion matrix* terbuat, perhitungan *accuracy, precision, recall*, dan *f1-score* dapat dilakukan menggunakan *classification_report()* dari *scikit-learn*. Perintah-perintah ini memiliki parameter yang sama dengan *confusion_ matrix(),* namun memiliki parameter tambahan berupa label yang bernilai 'labels'. Labels merupakan array bernilai kata-kata sentimen yang digunakan sebagai label pada penelitian ini, yaitu 'positive', 'negative', dan 'neutral'.

Hasil dari evaluasi model dapat dilihat pada gambar 5 sampai dengan 10. Pada gambar tersebut, terlihat *confussion matrix* dan hasil *classification report* untuk setiap model dengan jumlah *epoch* yang berbeda.

Tabel 2. Hasil Evaluasi Model dengan *Epoch* 10

	precision	recall	F1-score	support
positive	0.68	0.76	0.72	504
negative	0.85	0.95	0.90	1465
neutral	0.83	0.65	0.72	1082
accuracy			0.81	3051

Tabel 3. Hasil Evaluasi Model dengan *Epoch* 20

	precision	recall	F1-score	support
positive	0.61	0.30	0.40	504
negative	0.96	0.93	0.95	1465
neutral	0.69	0.88	0.77	1082
accuracy			0.81	3051

Tabel 4. Hasil Evaluasi Model dengan *Epoch* 50

	precision	recall	F1-score	support
positive	0.76	0.85	0.80	504
negative	0.87	0.95	0.91	1465
neutral	0.88	0.72	0.79	1082
accuracy			0.85	3051

Berdasarkan Tabel 2, dapat dilihat bahwa pelatihan model dengan *epoch* 10 menghasilkan akurasi yang cukup baik, yaitu sebesar 81%. Namun *precision* untuk *tweet* positif sebesar 68% menunjukkan tingkat ketepatan antara label actual bernilai positif dengan jawaban berupa label prediksi bernilai positif yang diberikan oleh sistem cukup rendah. *Recall* untuk *tweet* netral sebesar 65% menunjukkan tingkat keberhasilan sistem dalam menemukan kembali informasi berupa label bernilai netral pun rendah. Sedangkan untuk *f1-score tweet* negatif sebesar 90% menunjukkan perbandingan rata-rata *precision* dan *recall tweet* negatif cukup baik.

Berdasarkan Tabel 3, dapat dilihat bahwa pelatihan model dengan *epoch* 20 menghasilkan akurasi yang masih sama dengan *epoch* 10, yaitu 81%. *Precision, recall,* dan *f1-score* pada percobaan kali ini paling rendah dihasilkan pada prediksi dengan label *tweet* positif. Hal ini menunjukkan ketepatan antara label aktual bernilai positif dengan jawaban berupa label prediksi bernilai positif yang diberikan oleh sistem cukup rendah, tingkat keberhasilan sistem dalam menemukan kembali informasi berupa label bernilai positif rendah, serta perbandingan rata-rata *precision* dan *recall tweet* positif pun rendah. Sebaliknya, pada *tweet* berlabel negatif, *precision, recall,* dan *f1-score* pada percobaan kali ini paling tinggi. Hal ini menunjukkan model cenderung lebih banyak mempelajari pola prediksi *tweet* negatif dibandingkan dengan positif.

Berdasarkan Tabel 4, dapat dilihat bahwa pelatihan model dengan *epoch* 50 menghasilkan akurasi yang lebih baik

dibandingkan dengan percobaan pada *epoch* sebelumnya, yaitu sebesar 85%. *Precision, recall,* dan *f1-score* pada ketiga jenis label pun cukup baik secara merata. Nilai paling rendah didapatkan pada *recall tweet* netral, yaitu sebesar 72% menunjukkan tingkat keberhasilan sistem dalam menemukan kembali informasi berupa label bernilai netral cukup rendah. Hal ini berbanding terbalik dengan *recall tweet* negatif yang mencapai 95%.

Berdasarkan Tabel 5, dapat dilihat bahwa pelatihan model dengan *epoch* 75 menghasilkan akurasi yang sedikit lebih baik dibanding *epoch* 50, yaitu sebesar 86%. *Precision, recall,* dan *f1-score* pada ketiga jenis label pun cukup baik secara merata. Nilai paling rendah didapatkan pada *recall tweet* netral, yaitu sebesar 72% menunjukkan tingkat keberhasilan sistem dalam menemukan kembali informasi berupa label bernilai netral cukup rendah. Hal ini berbanding terbalik dengan *recall tweet* negatif yang mencapai 96%.

Berdasarkan Tabel 6, dapat dilihat bahwa pelatihan model dengan *epoch* 100 menghasilkan akurasi yang masih sama dengan *epoch* 75, yaitu sebesar 86%. *Precision, recall,* dan *f1-score* pada ketiga jenis label pun cukup baik secara merata. Nilai paling rendah masih didapatkan pada *recall tweet* netral, yaitu sebesar 72% menunjukkan tingkat keberhasilan sistem dalam menemukan kembali informasi berupa label bernilai netral cukup rendah. Hal ini berbanding terbalik dengan *recall tweet* negatif yang mencapai 96%.

Tabel 5. Hasil Evaluasi Model dengan *Epoch* 75

	precision	recall	F1-score	support
positive	0.79	0.86	0.82	504
negative	0.86	0.96	0.91	1465
neutral	0.89	0.72	0.80	1082
accuracy			0.86	3051

Tabel 6. Hasil Evaluasi Model dengan *Epoch* 100

	precision	recall	F1-score	support
positive	0.82	0.86	0.84	504
negative	0.85	0.96	0.90	1465
neutral	0.89	0.72	0.80	1082
accuracy			0.86	3051

Tabel 7. Hasil Evaluasi Model dengan *Epoch* 150

	precision	recall	F1-score	support
positive	0.80	0.88	0.84	504
negative	0.86	0.96	0.91	1465
neutral	0.91	0.72	0.80	1082
accuracy			0.86	3051

Berdasarkan Tabel 7, dapat dilihat bahwa pelatihan model dengan *epoch* 150 menghasilkan akurasi yang masih sama dengan *epoch* 100, yaitu sebesar 86%. *Precision, recall,* dan *f1-score* pada ketiga jenis label pun cukup baik secara merata. Nilai paling rendah masih didapatkan pada *recall tweet* netral, yaitu sebesar 72% menunjukkan tingkat keberhasilan sistem dalam menemukan kembali informasi berupa label bernilai netral cukup rendah. Hal ini berbanding terbalik dengan *recall tweet* negatif yang mencapai 96%.

Hasil tersebut dapat dibuktikan dengan menghitung secara manual *accuracy, precision, recall,* dan *f-1 score* berdasarkan *confusion matrix* yang didapatkan. Contoh pembuktian perhitungan menggunakan hasil *confussion matrix* model dengan *epoch* sebanyak 100.

Accuracy model didapatkan dengan rumus pada persamaan (1) berikut:

$$Accuracy = \frac{TP}{Jumlah\ data} \quad (1)$$

Perhitungan *accuracy* model dengan *epoch* 100 berdasarkan *confusion matrix* pada gambar 9 sebagai berikut:

$$Accuracy = \frac{1404 + 780 + 432}{3051} = 0.86$$

Akurasi pada *epoch* 100 tersebut bernilai 0.86 atau 86%. Nilai ini didapatkan dengan membagi jumlah *true positive* dengan seluruh jumlah data pada *confussion matrix*. Nilai 86% menunjukkan ketepatan prediksi terhadap seluruh label yang cukup baik, yaitu sebanyak 86%, sedangkan sisanya menunjukkan program masih melakukan kesalahan prediksi sebanyak 14%.

Precision didapatkan dengan rumus pada persamaan (2) berikut:

$$Precision = \frac{TP}{(TP + FP)} \quad (2)$$

Perhitungan *precision* model dengan *epoch* 100 berdasarkan *confusion matrix* pada gambar 9 untuk *tweet* positif sebagai berikut:

$$Precision\ tweet\ positif = \frac{432}{(14 + 84 + 432)} = 0.82$$

Precision tweet positif pada *epoch* 100 tersebut bernilai 0.82 atau 82%. Nilai ini didapatkan dengan membagi jumlah *true positive* dengan jumlah *true positive* dan *false positive* pada *confussion matrix*. Nilai 82% menunjukkan ketepatan antara label aktual bernilai positif dengan jawaban berupa label prediksi bernilai positif yang diberikan oleh sistem cukup baik, sedangkan sisanya menunjukkan program masih melakukan

kesalahan prediksi label positif terhadap *tweet* yang diprediksi positif sebanyak 18%.

Recall model didapatkan dengan rumus pada persamaan (3) berikut:

$$Recall = \frac{TP}{(TP + FN)} \quad (3)$$

Perhitungan *recall* model dengan *epoch* 100 berdasarkan *confusion matrix* pada gambar 9 untuk *tweet* positif sebagai berikut:

$$Recall\ tweet\ positif\ = \frac{432}{(23 + 49 + 432)} = 0.86$$

Recall tweet positif pada *epoch* 100 tersebut bernilai 0.86 atau 86%. Nilai ini didapatkan dengan membagi jumlah *true positive* dengan jumlah *true positive* dan *false negative* pada *confussion matrix*. Nilai 86% menunjukkan tingkat keberhasilan sistem dalam menemukan kembali informasi berupa label bernilai positif cukup baik, sedangkan sisanya menunjukkan program masih melakukan kesalahan prediksi label positif terhadap *tweet* yang memang benar positif sebanyak 14%.

F-1 score didapatkan dengan rumus pada persamaan (4):

$$F1 - Score = 2 \times \frac{Recall\ \times Precision}{Recall + Precision} \quad (4)$$

Perhitungan *f-1 score* model dengan *epoch* 100 berdasarkan *confusion matrix* pada gambar 9 untuk *tweet* positif sebagai berikut:

$$F - 1\ score\ tweet\ positif\ = 2 \times \frac{0.86 \times 0.82}{(0.86 + 0.82)} = 0.84$$

F1-score tweet positif pada *epoch* 100 tersebut bernilai 0.84 atau 84%. Nilai ini didapatkan dengan membagi dua kali hasil pembagian dari perkalian *recall* dan *precision* yang dibagi dengan hasil penjumlahan *recall* dengan *precision.* Nilai 84% menunjukkan perbandingan rata-rata *precision* dan *recall tweet* positif cukup baik.

KESIMPULAN DAN SARAN

Program analisis sentimen terhadap lingkungan hidup Kota Depok pada media sosial Twitter telah berhasil dibuat dengan implementasi metode *Convolutional Neural Network* (CNN). Program ini mampu melakukan klasifikasi *tweet* pengguna Twitter menjadi tiga kelas sentimen, yaitu positif, negatif, dan netral. Program dibuat dengan menggunakan bahasa Python dan direpresentasikan menjadi sebuah *website* menggunakan kerangka kerja Flask.

Penelitian ini menggunakan data pelatihan sejumlah 11.219 data *tweet.* Data pelatihan tersebut terlebih dahulu melalui *data preprocessing* dan pelabelan sentimen, kemudian dibagi menjadi 8.975 data latih dan

2.244 data validasi untuk membangun dan melakukan uji model CNN. Model yang sudah dilatih kemudian diuji. Hasil pengujian model terhadap set data uji berisi 3.051 data *tweet* menghasilkan akurasi paling tinggi dengan menggunakan *epoch* sebanyak 100 dan 150 sebesar 86% dan sentimen tertinggi yaitu sentimen negatif. Akurasi tersebut menunjukkan program sudah mampu melakukan analisis sentimen meskipun masih memiliki kesalahan klasifikasi sentimen sebesar 14%. Pengembang-an yang dapat dilakukan kedepannya di-antaranya, yaitu memperbanyak data dengan keragaman yang lebih unik untuk melatih model CNN, memperbaiki *data preprocessing* agar data lebih sesuai dengan kebutuhan pelatihan dan implementasi model, menambahkan jumlah *epoch* dalam melakukan pelatihan model, serta memperbanyak lapisan model dan memadukan dengan model lain, misalnya *Long Short-Term Memory* (LSTM).

DAFTAR PUSTAKA

[1] Dinas Perumahan Rakyat Kawasan Permukiman dan Lingkungan Hidup, "Definisi Lingkungan Hidup Indonesia", https://dprkplh.tanahlautkab.go.id. [Diakses: 28 April 2021].

[2] Liu Bing, *Sentiment Analysis and Opinion Mining*, California: Morgan and Claypool Publisher, 2012.

[3] Chengqing Zong, et al, *Text Data Mining*, Beijing: Tsinghua University Press, 2021.

[4] Budi M. M. & Dwi H.W, "Aspect-Based Sentiment Analysis Approach with CNN", 5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), vol. 1, hal. 142-147, 2018, https://doi.org/10.1109/EECSI.2018.8752857.

[5] Moch. Ari Nasichuddin, et al, "Performance Improvement Using CNN for Sentiment Analysis", International Journal of Information Technology and Electrical Engineering (IJITEE) vol. 2, no. 1, hal. 9-14, 2018, https://doi.org/10.22146/ijitee.36642.

[6] Faiz Adil K., et al, "Analisis Sentimen Terhadap Review Aplikasi Layanan E-Commerce Menggunakan Metode Convolutional Neural Network", E-Proceeding of Engineering, vol. 7, no. 2, hal. 4559-4566, 2020.

[7] Navi Atri L., et al, "Metode Naïve Bayes Classifier dengan Textblob untuk Analisis Sentimen Terhadap Pelayanan Indihome dan First Media", Seminar Nasional Teknologi Informasi dan Komunikasi STI&K , vol. 4, no. 1 , hal. 283-288, 2020.

[8] Devid Haryalesmana W. & Azhari S. N., "Peringkasan Sentimen Esktraktif di Twitter Menggunakan Hybrid TF-IDF dan Cosine Similarity", Indonesian Journal of Computing and Cybernetics Systems, vol.10, no.2, hal. 207-218, 2016, https://doi.org/10.22146/ijccs.16625.