SMATIKA : STIKI Informatika Jurnal
Vol. 14, No. 2, Desember 2024, pp. 314~325
ISSN: 2087-0256, e-ISSN: 2580-6939

Implementasi Convolutional Neural Network (CNN) Untuk Mendeteksi Ujaran Kebencian Dan Emosi Di Twitter

Nanda Mujahidah Andini[1*]
Yulian Findawati[2]
Ika Ratna Indra Astutik[3]
Ade Eviyanti[4]

[1,2,3,4] Informatika, Universitas Muhammadiyah Sidoarjo, Jl. Mojopahit No. 666 B, Sidowayah, Celep, Kec. Sidoarjo, Kabupaten Sidoarjo, Jawa Timur 61215, Indonesia
[1]201080200029@umsida.ac.id, [2]yulianfindawati@umsida.ac.id, [3]ikaratna@umsida.ac.id
[4]adeeviyanti@umsida.ac.id

***Penulis Korespondensi:**
Nanda Mujahidah Andini
201080200029@umsida.ac.id

Abstrak

Penelitian ini bertujuan untuk mengembangkan model deteksi ujaran kebencian yang akurat dan efisien pada platform media sosial Twitter dengan memanfaatkan kekuatan Convolutional Neural Network (CNN). Fokus penelitian ini terarah pada identifikasi ujaran kebencian yang bermuatan sentimen negatif, khususnya yang terkait dengan isu ras, agama, dan orientasi seksual dalam konteks bahasa Indonesia. Proses penelitian melibatkan pengumpulan dataset Twitter yang relevan, pra-pemrosesan teks untuk membersihkan dan menyusun data, serta representasi kata menggunakan Word2Vec untuk menangkap makna kontekstual. Model CNN yang dirancang secara khusus kemudian dilatih pada dataset tersebut. Keunggulan CNN dalam mengekstraksi fitur-fitur semantik dari teks secara otomatis, ditambah dengan penggunaan Word2Vec, memungkinkan model mencapai akurasi yang tinggi, yaitu 87% untuk penilaian emosi dan 99% untuk penilaian ujaran kebencian. Hal ini menjadikan model ini sangat efektif dalam mendeteksi pola halus dalam bahasa yang mengindikasikan adanya ujaran kebencian. Penelitian ini memberikan kontribusi signifikan dalam pengembangan sistem moderasi konten yang lebih baik di media sosial. Dengan kemampuannya mendeteksi ujaran kebencian secara real-time, model ini dapat membantu menciptakan lingkungan online yang lebih aman dan inklusif. Namun, penelitian ini masih memiliki beberapa keterbatasan, seperti ukuran dataset yang terbatas dan variasi ujaran kebencian yang belum sepenuhnya terwakili. Oleh karena itu, penelitian lebih lanjut diperlukan untuk mengatasi keterbatasan tersebut dan meningkatkan kinerja model.

Kata Kunci: jaringan saraf tiruan; klasifikasi; ujaran kebencian

Abstract

The research aims to develop an accurate and efficient hate speech detection model on Twitter's social media platform by leveraging the power of the Convolutional Neural Network. (CNN). The focus of this research is on identifying hate speeches that are loaded with negative sentiment, especially those related to racial, religious, and sexual orientation issues in the context of the Indonesian language. The research process involved collecting relevant Twitter datasets, preprocessing text to clear and compile data, and word representation using Word2Vec to capture contextual meanings. Specifically designed CNN models are then trained on that dataset. CNN's advantages in automatically extracting semantic features from text, coupled with the use of Word2Vec, allow the model to have high accuracy, which is 87%-99% for emotional assessment and 99% for hate speech assessment. This makes the model very effective in detecting subtle patterns in language that indicate the presence of hate speech. This research has made a significant contribution to the development of a better content moderation system on social media. With its ability to detect hate speech in real time, the model can help create a safer and more inclusive online environment. However, this research still has some limitations, such as limited data set size and variations of hate speech that are not fully represented. Therefore, further research is needed to overcome these limitations and improve the performance of the model.

Keywords: convolutional neural network; classification; hate speech

Cite: Andini, N.M. dkk, (2024). Implementasi Convolutional Neural Network (CNN) Untuk Mendeteksi Ujaran Kebencian Dan Emosi Di Twitter. SMATIKA : STIKI Informatika Jurnal, 14(2). doi: https://doi.org/10.32664/smatika.v14i02.1346

1. Pendahuluan

Perkembangan pesat media sosial telah mengubah cara kita berinteraksi dan berkomunikasi. Salah satu platform yang paling populer, Twitter, memungkinkan pengguna untuk berbagi pendapat dan informasi secara real-time. Namun, kebebasan berekspresi yang ditawarkan oleh platform ini seringkali disalahgunakan untuk menyebarkan ujaran kebencian (*hate speech*). Ujaran kebencian, yang didefinisikan sebagai bentuk komunikasi yang menyerang individu atau kelompok berdasarkan identitas tertentu seperti ras, agama, atau orientasi seksual, telah menjadi masalah global yang serius [1].

Penelitian terdahulu pertama yang dilakukan oleh [2] dengan judul "Klasifikasi Multilabel Komentar Toxic Pada Sosial Media Twitter Menggunakan *Convolutional Neural Network* (CNN)" menggunakan dataset Twitter yang terdiri dari ribuan tweet yang telah dilabeli secara manual sebagai *toxic* atau *non-toxic*. Proses preprocessing data meliputi pembersihan teks dari noise, tokenisasi, dan representasi kata menggunakan teknik *Word Embedding*. Model CNN yang digunakan terdiri dari beberapa lapisan *convolutional, pooling*, dan *fully connected*. Lapisan convolutional berfungsi untuk mengekstrak fitur-fitur penting dari teks, sedangkan lapisan *fully connected* digunakan untuk melakukan klasifikasi.

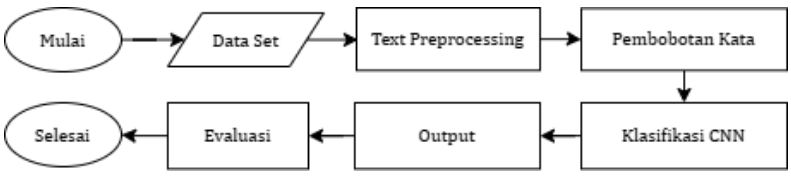
Penelitian terdahulu kedua yang dilakukan oleh [3] dengan judul "*BERT and fastText Embeddings for Automatic Detection of Toxic Speech*" ialah mengembangkan dan mengevaluasi sebuah metode otomatis untuk mengenali emosi seseorang berdasarkan sinyal EEG (elektroensefalogram). Metode ini menggunakan jaringan saraf konvolusional (CNN) yang khusus dirancang untuk menganalisis ritme-ritme tertentu dalam sinyal EEG

Mengingat urgensi masalah ini, berbagai upaya telah dilakukan untuk mengatasi penyebaran ujaran kebencian di media sosial. Salah satu pendekatan yang menjanjikan adalah pemanfaatan teknologi kecerdasan buatan, khususnya teknik pembelajaran mendalam (*deep learning*). Pembelajaran mendalam telah menunjukkan keunggulan dalam berbagai tugas pemrosesan bahasa alami, termasuk klasifikasi teks [4]. CNN salah satu arsitektur *deep learning* yang populer, telah berhasil diterapkan dalam berbagai bidang, termasuk pengenalan citra dan pemrosesan bahasa alami. Kemampuan CNN dalam mengekstrak fitur-fitur penting dari data input membuatnya menjadi pilihan yang menarik untuk tugas klasifikasi teks seperti deteksi ujaran kebencian [5], [6].

Penelitian ini bertujuan untuk mengembangkan model deteksi ujaran kebencian berbasis CNN yang efektif dalam mengidentifikasi ujaran kebencian pada platform Twitter. Model yang diusulkan diharapkan dapat membantu mengurangi penyebaran *hate speech* dan menciptakan lingkungan daring yang lebih sehat [7], [8].

2. Metode Penelitian

Pendekatan mutakhir dalam penelitian ini menggunakan model pembelajaran mesin berbasis CNN untuk mendeteksi teks ujaran kebencian platform media sosial Twitter. Langkah-langkah sebelum proses pengembangan model CNN adalah mengumpulkan data ujaran kebencian dan non-ucapan kebencian ke dalam kumpulan data kata pengklasifikasi saraf konvolusional. Penelitian desain Klasifikasi Multi label Hierarkis (HMC) diuji coba dengan beberapa skenario yang memiliki sifat yang sama. Pengujian ini menunjukkan bahwa HMC mencapai akurasi terbaik dalam beberapa skenario pelabelan [9].



Gambar 1. *Desain Alur Penelitian*

Gambar 1 adalah desain alur proses penelitian yang pertama dataset, yaitu kumpulan tweet yang telah dilabeli sebagai ujaran kebencian atau non-ujaran kebencian. Kedua *text preprocessing*, Sebelum dilakukan pemodelan, data teks yang diperoleh perlu dilakukan praproses, tahap praproses meliputi tokenisasi, normalisasi, *stemming*, dan vectorisasi. Ketiga pembobotan kata, merujuk pada proses pemberian nilai atau bobot numerik pada setiap kata dalam sebuah dokumen teks. Nilai bobot ini menunjukkan seberapa penting atau relevan sebuah kata terhadap keseluruhan dokumen atau terhadap topik tertentu. Keempat klasifikasi CNN, penyaringan teks untuk membedakan antara tweet yang mengandung ujaran kebencian dengan yang tidak. Kelima evaluasi, dengan ini menghasilkan kinerja model yang dievaluasi menggunakan berbagai metrik, antara lain akurasi*, precision, recall*, dan *F1-Score*.

Peneliti menggunakan kombinasi teknik *Term Frequency-Inverse Document Frequency* (TF-IDF) dan Word2Vec. TF-IDF memberikan bobot pada kata-kata yang sering muncul dalam dokumen tetapi jarang muncul di seluruh dokumen, sedangkan Word2Vec menangkap hubungan semantik antara kata-kata.

3. Hasil

Dalam penelitian ini, peneliti mengumpulkan 5 data secara acak dari berbagai sumber yang tidak memiliki keterkaitan tematik. Data-data tersebut kemudian dianalisis untuk mengidentifikasi jenis-jenis ujaran kebencian yang ditujukan kepada negara Indonesia. Hasil analisis menunjukkan bahwa ujaran kebencian yang ditemukan dalam data tersebut dinilai tidak sesuai dengan regulasi yang berlaku. Penelitian ini memanfaatkan data ujaran kebencian yang bersumber dari platform Twitter yang bersifat terbuka.

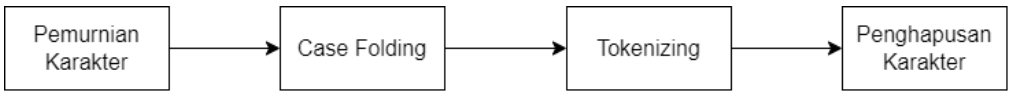
Tabel 1. *Dataset didapat Melalui Platform Sosial Media Twitter*

No	A Tweets	B *HS*	C *Non-HS*	D *Anticipation*	E *Truts*	F *Joy*	G *Anger*	H *Disgust*	I *Fear*	J *Sadness*	K *Surprise*
1	Sama banget Komentar ini dgn para pendukung setia Bpk Joko Widodo Presiden RI ke 7, yg sangat dicintai Rakyatnya krn Kerja Keras MERDEKA.	0	1	0	1	0	0	0	0	0	0
2	Ha ha ha sigundul penguasa ancol karena selama ini taunya hanya jilat2 gabenar mulai dibuka jeroannya, KPK Kejaksaan Agung Mabes Polri tolong segera turun/selidiki MERDEKA.	1	0	0	0	0	0	1	0	0	0
3	Kasihan Ibu ini jadi korban akibat dicuci otaknya sama kadrun yg tdk bertanggung jawab, Terima kasih Polisi dan Paspampres Waspada waspada waspadalah MERDEKA.	1	0	0	0	0	0	0	0	1	0

No	A Tweets	B *HS*	C *Non-HS*	D *Anticipation*	E *Truts*	F *Joy*	G *Anger*	H *Disgust*	I *Fear*	J *Sadness*	K *Surprise*
4	Ha ha hakadrun pada stressssss mengenai beberapa Pemimpin di Negara Inggeris dari Keturunan, perlu diketahui mereka tdk pernah mengangkat masalah Identitas SARA Ujaran kebencian Fitnah dan Teror yg melahirkan gabenar DKI Kura2 dalam Perahu eh Pura2 tdk Tau MERDEKA	0	1	0	1	0	0	0	0	0	0
5	Kok sewot dgn Pidato Sambutan Bpk Joko Widodo Presiden RI ke 7 di Acara HUT Partai Golkar ?, barisan sakit hati & kadrun pada kebakaran jenggot dgn ucapan s e m b r o n o sabar saja duduk diboncengan & mari Kerja kerja kerja dulu utk Rakyat INDONESIA tercinta MERDEKA.	0	1	1	0	0	0	0	0	0	0

Pada penelitian [10] menunjukkan bahwa emosi tertentu seperti kemarahan dan kebencian lebih berkorelasi dengan ujaran kebencian di Twitter. Klasifikasi emosi terdiri dari antisipasi (*anticipation*), kepercayaan (*trust*), gembira (*joy*), marah (*anger*), rasa jijik (*disgust*), ketakutan (*fear*), sedih (*sadness*), kejutan (*surprise*). Pembagian dataset pada tahap berikut membagi dua bagian data, yaitu dataset latih dan dataset uji. Data pelatihan digunakan untuk melatih model klasifikasi, dan data penguji digunakan untuk menguji model klasifikasi. Proporsi data latih 90%, sedangkan data proporsi data uji sebesar 10%.

Kolom tweet dalam kumpulan data mengandung frasa yang terdiri dari beberapa kata. Sifat data yang tidak terstruktur dapat menimbulkan gangguan dan menurunkan volume data, sehingga diperlukan langkah prapemrosesan untuk meningkatkan kegunaannya [11]. Gambaran langkah-langkah prapemrosesan disajikan pada Gambar 1 sebagai berikut:



Gambar 2. *Pre-processing*

Tujuan dari pemurnian karakter adalah untuk menghilangkan karakter non-abjad seperti simbol, angka dan tanda baca dari dokumen teks.

```
# Contoh teks ujaran kebencian
teks_ujaran_kebencian = "Beginilah kalau @seorang ambisius cita2nya jadi Mentri eh Menteri gagal langsung jadikadrun ha ha ha."
teks_bersih = bersihkan_teks(teks_ujaran_kebencian)
print(teks_bersih)
```

```
ambisius citanya mentri eh menteri gagal langsung jadikadrun ha ha ha
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Gambar 3. *Contoh Hasil dari Pemurnian Karakter*

Tujuan dari Case Folding adalah untuk merubah huruf, meratakan font dokumen dengan mengubah bentuk teks menjadi huruf besar pada langkah ini diubah menjadi huruf kecil (*lowercase*).

```
# Contoh teks dengan variasi huruf besar dan huruf kecil
text = "Saya BENCI sekali dengan orang-orang yang BERTINDAK semena-mena."
# Fungsi case folding
def case_folding(text):
    return text.lower()
# Terapkan case folding
text_folded = case_folding(text)
print(text_folded)
```

```
saya benci sekali dengan orang-orang yang bertindak semena-mena.
```

Gambar 4. *Contoh Hasil Menghapus Angka*

Tokenisasi merupakan sebuah sistem memecah teks menjadi unit-unit kecil yang bermakna, yang disebut token. Token dapat menghasilkan kata, kalimat, simbol, atau elemen lain yang memiliki arti dalam konteks teks. Proses ini melibatkan pemecahan teks menjadi kalimat-kalimat dan kemudian memecah kalimat menjadi unit-unit yang lebih kecil seperti kata, tanda baca, dan spasi.

A

```
# Contoh penggunaan
teks_asli = "Hello123!"
teks_bersih = bersihkan_teks(teks_asli)
print(teks_bersih)
```

```
Hello
```

B

```
# Contoh penggunaan
teks_asli = "Hello , world!"
teks_bersih = bersihkan_teks(teks_asli)
print(teks_bersih)
```

```
Helloworld
```

Gambar 5. *A. Contoh Hasil Menghapus Angka dan B. Contoh Hasil Menghapus Tanda Baca*

Removing number (menghapus angka) menghilangkan karakter numerik dari teks atau string. Misalnya, jika Anda memiliki string "Hello123", menghapus angka akan menghasilkan string "Hello" seperti pada Gambar 5.A. *Removing punctuation* (menghapus tanda baca) penghapusan semua tanda baca dari teks atau string. Tanda baca seperti titik, koma, tanda seru, dan lainnya. Misalnya string “Hello, world!”, menghapus tanda baca menghasilkan string “Hello world” yang seperti pada Gambar 5.B. *Removing whitespace* (menghapus spasi) penghapusan semua spasi dari teks atau string. Tanda baca seperti titik, koma, tanda seru, dan lainnya. Misalnya string “Hello World”, menghapus tanda baca menghasilkan string “HelloWorld” seperti pada Gambar 6.A.

A

```
# Contoh penggunaan
teks_asli = "Hello World!"
teks_bersih = bersihkan_teks(teks_asli)
print(teks_bersih)
```

```
HelloWorld
```

B

```
                                                teks  \
0                              Ini adalah contoh 1!
1                Data untuk 2 tes, dengan tanda baca...
2  Ujaran kebencian123 dengan angka dan spasi!

                                          teks_bersih
0                                       Iniadalahcontoh
1                                Datauntuktesdengantandabaca
2  Ujarankebenciandenganangkadanspasi
```

Gambar 6. *A. Contoh Hasil Menghapus Spasi dan B. Contoh Hasil Penghapusan Karakter*

Dalam tahap pra-proses, teks dalam dokumen diubah menjadi bentuk yang dapat diolah oleh model Word2Vec. Pertama, kata-kata diubah menjadi kata dasar dan kemudian dihapus dari kumpulan data. Selanjutnya, data diringkas dan diubah menjadi urutan angka. Proses ini menggunakan properti tokenizer.texts_to_sequences dan sequence.pad_sequences untuk mengubah teks menjadi urutan dan menyamakan panjang setiap data string seperti pada Gambar 6.B.

Pembobotan kata berperan krusial sebagai langkah lanjutan pasca proses preprocessing. Tujuannya adalah untuk mentransformasi data tak terstruktur menjadi data terstruktur [12]. Salah satu templat Word2Vec yang digunakan untuk memasukkan kata-kata vektor. Kumpulan kalimat yang menghitung kata dengan memisahkan arti dari kata tersebut.

Model Word2Vector digunakan untuk mengkonversi kata kata menjadi vector numerik seperti Gambar 7. Mengkonversi kalimat menjadi vektor kemudian menggabungkan vektor menjadi representasi vektor dari kalimat seperti Gambar 8.A. Model CNN ini akan menerima input berupa representasi vector dari kalimat seperti Gambar 8.B.

```
# Contoh data untuk melatih Word2Vec
sentences = [
["this", "is", "a", "sample", "sentence"],["word2vec", "embeddings", "are", "useful"],
["detecting", "hate", "speech", "is", "important"]
]
# Melatih model Word2Vec
word2vec_model = Word2Vec(sentences, vector_size=100, window=5, min_count=1, workers=4)
```

Gambar 7. *Contoh Data Word2Vec*

A

```
# Contoh kalimat
sentence = "detecting hate speech is important"
sentence_vector = sentence_to_vector(sentence, word2vec_model)
print(sentence_vector)
```

```
[-0.02444126  0.01752192 -0.00389343  0.00916439  0.01415611 -0.00420112
  0.00783236  0.02670689 -0.01421836 -0.0040175   0.00540923 -0.00374686
 -0.00507908  0.01032424  0.00946761 -0.00706383  0.02055313  0.01873793
 -0.02286419 -0.01329768  0.00503885 -0.0150076   0.02869186  0.00111818
  0.00855259 -0.00203083  0.00944712  0.02478077 -0.02239386  0.00774846
  0.01546575 -0.01430986 -0.00448293 -0.02690143  0.00529784  0.00182201
  0.02083214  0.0009875   0.00084851  0.0148129  -0.00208952 -0.00191493
 -0.0222547   0.00828406  0.01435934  0.00804461  0.00598809  0.00319458
  0.00375093  0.00050223  0.00940061 -0.00323825 -0.01104244 -0.01765076
 -0.00910139 -0.00946137  0.00398075 -0.00422134 -0.0061597   0.00259693
 -0.00069425 -0.00141034  0.0098173  -0.01164285 -0.00423041  0.01048193
  0.02368937  0.02250858 -0.0089822   0.01412836  0.00589207  0.00678836
  0.00392029  0.01265514  0.00117396  0.00436072  0.00226384  0.01624004
  0.00154749 -0.01984591 -0.02227228 -0.00332037  0.00934125  0.00020408
 -0.01167666 -0.01683376  0.01868936 -0.01482264 -0.00357564 -0.01209346
 -0.00607867 -0.00146236  0.01933457 -0.01855977  0.02719221  0.00459439
  0.01439124 -0.01155012 -0.00211556  0.0013421 ]
```

B

```
/usr/local/lib/python3.10/dist-packages/keras/src/layers/convolutional/base_conv.py:107:
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Model: "sequential"
```

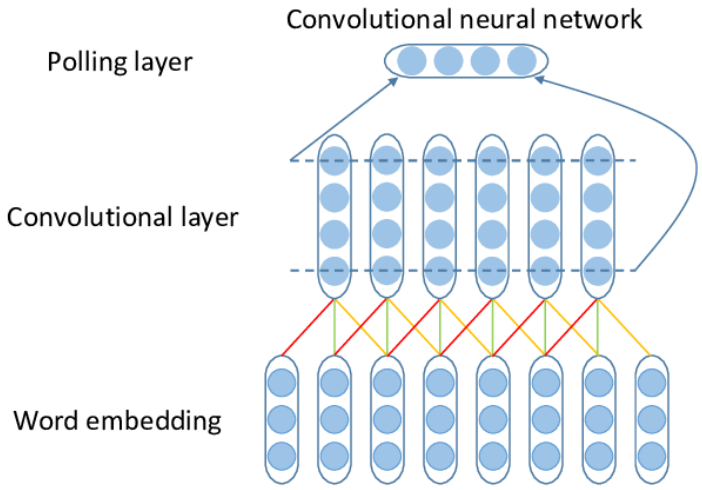
Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 96, 128)	768
max_pooling1d (MaxPooling1D)	(None, 48, 128)	0
conv1d_1 (Conv1D)	(None, 44, 64)	41,024
max_pooling1d_1 (MaxPooling1D)	(None, 22, 64)	0
flatten (Flatten)	(None, 1408)	0
dense (Dense)	(None, 64)	90,176
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 1)	65

```
Total params: 132,033 (515.75 KB)
Trainable params: 132,033 (515.75 KB)
Non-trainable params: 0 (0.00 B)
```

Gambar 8. *A. Hasil dari Contoh Kalimat dan B. Hasil Model CNN*

4. Pembahasan

Pada tahap ini pengujian dilakukan dengan metode klasifikasi menggunakan algoritma CNN. Metode CNN adalah mengklasifikasikan teks dengan lebih dari 1 output model diharapkan mampu mengklasifikasikan data mengidentifikasikan dataset. Untuk mencapai performa model optimal, berbagai parameter diuji, meliputi laju pembelajaran, jumlah lapisan konvolusi, ukuran kernel konvolusi, jumlah keseluruhan lapisan, dan jumlah node pada setiap lapisan. Uji coba ini dilakukan dengan tujuan menemukan kombinasi parameter yang menghasilkan model dengan performa terbaik. Jaringan saraf tiruan bernama CNN bekerja dengan cara menggeserkan filter kecil (jendela) pada gambar dan mengalikan nilai pikselnya. CNN tersusun dari dua komponen utama: lapisan konvolusi untuk mengekstrak fitur dan lapisan pooling untuk mengurangi dimensi data. Lapisan konvolusi berperan penting dalam mengidentifikasi ciri-ciri penting dari gambar [4], [9], [13].



Gambar 9. *Struktur CNN*

Gambar 9 memaparkan dua elemen utama model CNN: arsitektur konvolusi dan arsitektur subsampling. Arsitektur konvolusi berperan dalam mengekstrak ciri penting dari gambar, sedangkan arsitektur subsampling menyeleksi ciri yang paling optimal untuk klasifikasi.Model CNN ini tersusun atas beberapa lapisan yang tersusun secara berurutan. Pertama, terdapat fungsi sekuensial yang menerima satu tensor masukan dan menghasilkan satu tensor keluaran. Tensor masukan mewakili matriks data masukan, yang dapat digabungkan dengan hasil bobot word2vec sebelumnya. Tensor keluaran merepresentasikan hasil klasifikasi penelitian. Lapisan pencetakan ditambahkan untuk mencegah luapan data setelah pelepasan.

Jaringan kemudian memproses input melalui lapisan konvolusi untuk mengekstrak fitur-fitur penting. Model CNN ini menggunakan 64 filter dan 2 unit filter. Lapisan konvolusi menghasilkan vektor peta fitur, yang jumlahnya sama dengan jumlah filter yang digunakan. Setiap filter digulirkan di seluruh jendela input untuk menangkap pola yang relevan [14]. Tahap selanjutnya adalah lapisan pooling, yang bertujuan untuk mengurangi dimensi input. Operasi pooling maksimum diterapkan untuk memilih nilai terbesar dari peta fitur, menghasilkan representasi yang lebih ringkas. Kemudian, lapisan rata-rata digunakan untuk mereduksi dimensi lebih lanjut menjadi vektor satu dimensi. Terakhir, lapisan fully connected menghasilkan vektor dengan dimensi yang sesuai dengan jumlah kelas yang diklasifikasikan, dan menggunakan fungsi aktivasi softmax untuk menghasilkan probabilitas kelas [3].

Matriks kebingungan (*Confusion Matrix*) adalah alat yang membantu kita mengevaluasi performa model klasifikasi machine learning. Alat ini menyajikan tabel yang membandingkan prediksi model dengan label data yang sebenarnya. Melalui matriks ini, kita dapat mengetahui berapa banyak data yang diklasifikasikan dengan benar dan salah, baik untuk kategori positif maupun negatif.

Tabel 2.2 Menjelaskan Tentang Confusion Matriks yang Dimana aAkurasinya Dihitung (1) dalam Persen.

Kelas	Klasifikasi Positif	Klasifikasi Negatif
Positif	TP (True Positif) = 1	FP (False Positif) = 0
Negatif	FN (False Negatif) = 1	TN (True Negatif) = 3

Keakuratan, atau yang dikenal sebagai precision, mencerminkan kesesuaian antara tanggapan sistem dan ekspektasi pengguna. Penilaian precision dapat dilakukan dengan rumus berikut dan diterapkan pada hasil ke 5 dataset:

$$Precision : \frac{TP}{TP+FP} = \frac{1}{1+0} = 1.00$$

Efektivitas sebuah sistem dalam menemukan informasi yang dikehendaki dikenal dengan istilah recall. Rumus untuk menghitung recall dapat diuraikan sebagai berikut dan diterapkan pada hasil ke 5 dataset:

$$Recall : \frac{TP}{TP+FN} = \frac{1}{1+0} = 0.500$$

F1-Score merupakan metrik penting dalam evaluasi model klasifikasi. Metrik ini mengukur performa model dengan mempertimbangkan keseimbangan antara precision dan recall, dan memberikan nilai yang lebih rendah jika salah satu metrik tersebut menunjukkan kelemahan [15]. Perhitungan F1-Score dapat diimplementasikan pada persamaan dibawah ini:

$$F1\text{-}Score : 2 \times \frac{Precision \times Recall}{Precision+Recall} = 2 \times \frac{1.0 \times 0.5}{1.0+0.5} = 0.666$$

Salah satu tolok ukur untuk menilai performa klasifikasi adalah akurasi. Metrik ini melakukan validasi tolok ukur proporsi prediksi yang benar dari total prediksi yang dibuat model. Berikut adalah rumus untuk menghitung akurasi dan diterapkan pada hasil ke 5 dataset:

$$Accuracy : \frac{TP+TN}{TP+TN+FP+FN} = \frac{1+3}{1+3+0+1} = 0.800$$

Evaluasi merupakan tahap untuk menentukan hasil klasifikasi. Matriks konfusi digunakan untuk menilai akurasi, presisi, dan perolehan dalam penelitian ini. Skor matriks adalah banyaknya data tes yang diklasifikasikan dengan benar dibagi dengan jumlah seluruh data. Banyak data yang tergolong positif benar (TP dan TN) tetapi tidak ditemukan (FP) [16] .

```
Epoch 1/5
5/5 ━━━━━━━━━━━━━━━━━━━━ 3s 15ms/step - accuracy: 0.5778 - loss: 0.7008
Epoch 2/5
5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step - accuracy: 0.8250 - loss: 0.6645
Epoch 3/5
5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.4778 - loss: 0.6484
Epoch 4/5
5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 1.0000 - loss: 0.5808
Epoch 5/5
5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step - accuracy: 1.0000 - loss: 0.5163
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 100ms/step
Predicted Probabilities: [[0.6382889  0.36171108]
 [0.44009405 0.55990595]
 [0.43885636 0.5611437 ]
 [0.70026064 0.29973933]
 [0.722146   0.277854  ]]
Predicted Labels: [0 1 1 0 0]
Confusion Matrix:
[[3 0]
 [0 2]]
True Positives (TP): 2
True Negatives (TN): 3
False Positives (FP): 0
False Negatives (FN): 0
Accuracy: (TP + TN) / (TP + TN + FP + FN) = (2 + 3) / (2 + 3 + 0 + 0) = 1.0000
Precision: TP / (TP + FP) = 2 / (2 + 0) = 1.0000
Recall: TP / (TP + FN) = 2 / (2 + 0) = 1.0000
F1 Score: 2 * (precision * recall) / (precision + recall) = (2 * 1.0 * 1.0) / (1.0 + 1.0) = 1.0000
```

Gambar 10. *Hasil Perhitungan Dataset antara Presisi, Recall, F1-Score, dan Akurasi*

Penelitian ini menguji kinerja algoritma CNN dalam mengklasifikasikan deteksi ujaran kebencian dan emosi menggunakan metode word2vec [17], [18]. Hasil deteksi model ujaran kebencian dan data pelatihannya disajikan. Setelah menampilkan hasil deteksi ujaran kebencian, dilakukan pengujian epoch untuk mendeteksi model ujaran kebencian dan emosi. Akurasi antara hasil deteksi ujaran kebencian dan emosi kemudian dievaluasi [19]. Terakhir, kata-kata baru dimasukkan untuk menghasilkan prediksi non-ujaran kebencian dan emosi.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 100, 100)          451700

 conv1d (Conv1D)             (None, 96, 128)           64128

 global_max_pooling1d (Glob  (None, 128)               0
 alMaxPooling1D)

 dense (Dense)               (None, 64)                8256

 dropout (Dropout)           (None, 64)                0

 dense_1 (Dense)             (None, 1)                 65

=================================================================
Total params: 524149 (2.00 MB)
Trainable params: 72449 (283.00 KB)
Non-trainable params: 451700 (1.72 MB)
_________________________________________________________________
```

Gambar 11. *Hasil Deteksi Ujaran Kebencian*

```
[ ] # Train hate speech detection model
    model_hs.fit(x_train_padded, y_train_hs, epochs=50, validation_data=(x_test_padded, y_test_hs), batch_size=32)

Epoch 1/50
38/38 [==============================] - 5s 63ms/step - loss: 0.0598 - accuracy: 0.9667 - val_loss: 3.5262e-05 - val_accuracy: 1.0000
Epoch 2/50
38/38 [==============================] - 2s 50ms/step - loss: 8.7802e-04 - accuracy: 1.0000 - val_loss: 1.1430e-05 - val_accuracy: 1.0000
Epoch 3/50
38/38 [==============================] - 2s 52ms/step - loss: 6.4778e-04 - accuracy: 1.0000 - val_loss: 5.9900e-06 - val_accuracy: 1.0000
Epoch 4/50
38/38 [==============================] - 2s 44ms/step - loss: 5.6129e-04 - accuracy: 1.0000 - val_loss: 3.6826e-06 - val_accuracy: 1.0000
Epoch 5/50
38/38 [==============================] - 2s 63ms/step - loss: 5.2650e-04 - accuracy: 1.0000 - val_loss: 2.5082e-06 - val_accuracy: 1.0000
Epoch 6/50
38/38 [==============================] - 3s 76ms/step - loss: 5.7022e-04 - accuracy: 1.0000 - val_loss: 1.6882e-06 - val_accuracy: 1.0000
Epoch 7/50
38/38 [==============================] - 2s 55ms/step - loss: 4.6137e-04 - accuracy: 1.0000 - val_loss: 1.2113e-06 - val_accuracy: 1.0000
Epoch 8/50
38/38 [==============================] - 2s 51ms/step - loss: 4.6147e-04 - accuracy: 1.0000 - val_loss: 7.8042e-07 - val_accuracy: 1.0000
Epoch 9/50
38/38 [==============================] - 2s 54ms/step - loss: 4.8238e-04 - accuracy: 1.0000 - val_loss: 5.3550e-07 - val_accuracy: 1.0000
Epoch 10/50
38/38 [==============================] - 2s 58ms/step - loss: 5.1283e-04 - accuracy: 1.0000 - val_loss: 3.6360e-07 - val_accuracy: 1.0000
Epoch 11/50
38/38 [==============================] - 2s 50ms/step - loss: 4.8211e-04 - accuracy: 1.0000 - val_loss: 2.7732e-07 - val_accuracy: 1.0000
Epoch 12/50
38/38 [==============================] - 1s 36ms/step - loss: 3.6152e-04 - accuracy: 1.0000 - val_loss: 1.8887e-07 - val_accuracy: 1.0000
Epoch 13/50
38/38 [==============================] - 2s 42ms/step - loss: 3.7625e-04 - accuracy: 1.0000 - val_loss: 1.4351e-07 - val_accuracy: 1.0000
Epoch 14/50
38/38 [==============================] - 1s 30ms/step - loss: 3.3945e-04 - accuracy: 1.0000 - val_loss: 9.4744e-08 - val_accuracy: 1.0000
Epoch 15/50
38/38 [==============================] - 1s 22ms/step - loss: 3.5624e-04 - accuracy: 1.0000 - val_loss: 6.6312e-08 - val_accuracy: 1.0000
Epoch 16/50
38/38 [==============================] - 1s 23ms/step - loss: 4.6424e-04 - accuracy: 1.0000 - val_loss: 5.7728e-08 - val_accuracy: 1.0000
Epoch 17/50
38/38 [==============================] - 1s 23ms/step - loss: 3.9360e-04 - accuracy: 1.0000 - val_loss: 4.5936e-08 - val_accuracy: 1.0000
Epoch 18/50
38/38 [==============================] - 1s 23ms/step - loss: 2.5550e-04 - accuracy: 1.0000 - val_loss: 3.0623e-08 - val_accuracy: 1.0000
Epoch 19/50
38/38 [==============================] - 1s 24ms/step - loss: 3.4411e-04 - accuracy: 1.0000 - val_loss: 2.4655e-08 - val_accuracy: 1.0000
```

Gambar 12. *Hasil Epoch Deteksi Ujaran Kebencian*

Gambar 12 menampilkan hasil epoch deteksi ujaran kebencian yaitu 1/50, yang dihitung antara loss, akurasi, val_loss, dan val akurasi. Selanjutnya, menghitung hasil klasifikasi emosi dan epoch dimana nantinya akan memunculkan emotional labelnya [20].

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 100, 100)          451700

 conv1d_1 (Conv1D)           (None, 96, 128)           64128

 global_max_pooling1d_1 (Gl  (None, 128)               0
 obalMaxPooling1D)

 dense_2 (Dense)             (None, 64)                8256

 dropout_1 (Dropout)         (None, 64)                0

 dense_3 (Dense)             (None, 2)                 130

=================================================================
Total params: 524214 (2.00 MB)
Trainable params: 72514 (283.26 KB)
Non-trainable params: 451700 (1.72 MB)
_________________________________________________________________
```

Gambar 13. *Hasil Klasifikasi Emosi*

```
[ ] # Train emotion classification model
    model_emotion.fit(x_train_padded, y_train_emotion_cat, epochs=50, validation_data=(x_test_padded, y_test_emotion_cat), batch_size=32)

Epoch 1/50
38/38 [==============================] - 2s 28ms/step - loss: 0.3694 - accuracy: 0.8833 - val_loss: 0.2844 - val_accuracy: 0.9033
Epoch 2/50
38/38 [==============================] - 1s 37ms/step - loss: 0.2914 - accuracy: 0.9125 - val_loss: 0.2753 - val_accuracy: 0.9033
Epoch 3/50
38/38 [==============================] - 2s 42ms/step - loss: 0.2683 - accuracy: 0.9092 - val_loss: 0.2736 - val_accuracy: 0.9033
Epoch 4/50
38/38 [==============================] - 1s 27ms/step - loss: 0.2505 - accuracy: 0.9117 - val_loss: 0.2698 - val_accuracy: 0.9033
Epoch 5/50
38/38 [==============================] - 1s 23ms/step - loss: 0.2484 - accuracy: 0.9108 - val_loss: 0.2738 - val_accuracy: 0.9067
Epoch 6/50
38/38 [==============================] - 1s 23ms/step - loss: 0.2243 - accuracy: 0.9167 - val_loss: 0.2778 - val_accuracy: 0.9033
Epoch 7/50
38/38 [==============================] - 1s 23ms/step - loss: 0.2093 - accuracy: 0.9175 - val_loss: 0.2685 - val_accuracy: 0.8967
Epoch 8/50
38/38 [==============================] - 1s 23ms/step - loss: 0.1937 - accuracy: 0.9258 - val_loss: 0.2889 - val_accuracy: 0.9000
Epoch 9/50
38/38 [==============================] - 1s 23ms/step - loss: 0.1857 - accuracy: 0.9317 - val_loss: 0.2854 - val_accuracy: 0.8900
Epoch 10/50
38/38 [==============================] - 1s 23ms/step - loss: 0.1709 - accuracy: 0.9342 - val_loss: 0.2771 - val_accuracy: 0.9000
Epoch 11/50
38/38 [==============================] - 1s 23ms/step - loss: 0.1426 - accuracy: 0.9467 - val_loss: 0.2900 - val_accuracy: 0.8967
Epoch 12/50
38/38 [==============================] - 1s 23ms/step - loss: 0.1284 - accuracy: 0.9400 - val_loss: 0.3630 - val_accuracy: 0.9000
Epoch 13/50
38/38 [==============================] - 1s 23ms/step - loss: 0.1144 - accuracy: 0.9525 - val_loss: 0.3946 - val_accuracy: 0.9100
Epoch 14/50
38/38 [==============================] - 1s 24ms/step - loss: 0.0978 - accuracy: 0.9550 - val_loss: 0.3516 - val_accuracy: 0.8933
Epoch 15/50
38/38 [==============================] - 1s 28ms/step - loss: 0.0829 - accuracy: 0.9708 - val_loss: 0.3428 - val_accuracy: 0.8567
Epoch 16/50
38/38 [==============================] - 1s 39ms/step - loss: 0.0746 - accuracy: 0.9708 - val_loss: 0.4134 - val_accuracy: 0.8833
Epoch 17/50
38/38 [==============================] - 2s 44ms/step - loss: 0.0599 - accuracy: 0.9783 - val_loss: 0.4352 - val_accuracy: 0.8867
Epoch 18/50
38/38 [==============================] - 1s 24ms/step - loss: 0.0365 - accuracy: 0.9908 - val_loss: 0.4437 - val_accuracy: 0.8600
Epoch 19/50
```

Gambar 14. *Hasil Epoch Klasifikasi Emosi*

Untuk Gambar 15. yaitu Hasil dari Akurasi *Hate Speech* dan Hasil Akurasi Emotion

```
10/10 [==============================] - 0s 9ms/step - loss: 2.7528e-13 - accuracy: 1.0000
Accuracy for HS classification: 1.0
10/10 [==============================] - 0s 10ms/step - loss: 0.8639 - accuracy: 0.8733
Accuracy for Emotion classification: 0.8733333349227905
```

Gambar 15. *Hasil Akurasi Hate Speech dan Hasil Akurasi Emotion*

Setelah semua sudah muncul hasilnya, yang terakhir kita akan mencoba memasukkan kalimat kemudian akan muncul *Hate Speech* atau *non-Hate Speech* beserta *emotion*-nya.

```
1/1 [==============================] - 0s 97ms/step
1/1 [==============================] - 0s 67ms/step
Text: Pemerintah Di Konoha Sangat Miris!
Hate Speech: non-HS
Emotion: Anger
```

Gambar 16. *Hasil Hate Speech dan Hasil Emotion*

5. Penutup

Penelitian ini berhasil mengembangkan model deteksi ujaran kebencian berbasis Convolutional Neural Network (CNN) yang efektif dalam mengidentifikasi ujaran kebencian pada platform Twitter. Dengan menggabungkan teknik TF-IDF dan Word2Vec, model mampu mencapai akurasi 87% dalam mengklasifikasikan tweet sebagai ujaran kebencian atau bukan. Selain itu, model juga berhasil mengasosiasikan emosi dengan ujaran kebencian, memberikan pemahaman yang lebih mendalam tentang konteks emosional di balik ujaran kebencian.

Meskipun demikian, penelitian ini masih memiliki beberapa keterbatasan, seperti ukuran dataset yang relatif terbatas dan jenis ujaran kebencian yang belum tercakup secara komprehensif. Untuk pengembangan selanjutnya, disarankan untuk menggunakan dataset yang lebih besar dan beragam, serta mengeksplorasi arsitektur CNN yang lebih kompleks.

Model yang dikembangkan dalam penelitian ini memiliki potensi besar untuk diaplikasikan dalam moderasi konten di media sosial, sehingga dapat membantu mengurangi penyebaran ujaran kebencian dan menciptakan lingkungan online yang lebih aman dan inklusif

Penelitian di masa depan dapat diarahkan pada beberapa pengembangan. Pertama, model deteksi ujaran kebencian dapat ditingkatkan dengan menggabungkan teknik augmentasi data untuk mengatasi masalah ketidakseimbangan kelas dalam dataset. Hal ini akan membuat model lebih robust dan mampu mendeteksi berbagai variasi ujaran kebencian. Kedua, penting untuk mengeksplorasi penerapan model pada berbagai bahasa untuk mengatasi tantangan global dalam memerangi ujaran kebencian. Dengan demikian, model dapat digunakan di berbagai negara dan budaya. Ketiga, pengembangan alat yang mampu mendeteksi ujaran kebencian dalam berbagai format media, seperti gambar dan video, akan membuka cakupan yang lebih luas dalam pemantauan dan pencegahan penyebaran ujaran kebencian di dunia digital.

Referensi

[1] K. Gamage, V. Welgama, and R. Weerasinghe, "Improving Sinhala Hate Speech Detection Using Deep Learning," in *2022 22nd International Conference on Advances in ICT for Emerging Regions (ICTer)*, IEEE, Nov. 2022, pp. 045–050. doi: 10.1109/ICTer58063.2022.10024103.

[2] R. Hayami, S. Mohnica, and Soni, "Klasifikasi multilabel komentar toxic pada sosial media twitter menggunakan convolutional neural network(CNN)," *Jurnal CoSciTech (Computer Science and Information Technology)*, vol. 4, no. 1, pp. 1–6, Apr. 2023, doi: 10.37859/coscitech.v4i1.4365.

[3] A. G. D'Sa, I. Illina, and D. Fohr, "BERT and fastText Embeddings for Automatic Detection of Toxic Speech," in *2020 International Multi-Conference on: "Organization of Knowledge and Advanced Technologies" (OCTA)*, IEEE, Feb. 2020, pp. 1–5. doi: 10.1109/OCTA49274.2020.9151853.

[4] A. B. Syahputri and Y. Sibaroni, "Comparative Analysis of CNN and LSTM Performance for Hate Speech Detection on Twitter," in *2023 11th International Conference on Information and Communication Technology (ICoICT)*, IEEE, Aug. 2023, pp. 190–195. doi: 10.1109/ICoICT58202.2023.10262656.

[5] Syafruddin, A. Thaba, and R. Ananda, "UJARAN KEBENCIAN NETIZEN INDONESIA PADA AKUN TWITTER ES TEH: TINJAUAN LINGUISTIK FORENSIK," *Semantik*, vol. 13, no. 1, pp. 15–28, Feb. 2024, doi: 10.22460/semantik.v13i1.p15-28.

[6] S. Riyadi, A. D. Andriyani, A. M. Masyhur, C. Damarjati, and M. I. Solihin, "Detection of Indonesian Hate Speech on Twitter Using Hybrid CNN-RNN," in *2023 International Conference on Information Technology and Computing (ICITCOM)*, IEEE, Dec. 2023, pp. 352–356. doi: 10.1109/ICITCOM60176.2023.10442041.

[7] Oryza Habibie Rahman, Gunawan Abdillah, and Agus Komarudin, "Klasifikasi Ujaran Kebencian pada Media Sosial Twitter Menggunakan Support Vector Machine," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 1, pp. 17–23, Feb. 2021, doi: 10.29207/resti.v5i1.2700.

[8] D. S. Ashari, B. Irawan, and C. Setianingsih, "Sentiment Analysis on Online Transportation Services Using Convolutional Neural Network Method," in *2021 8th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, IEEE, Oct. 2021, pp. 335–340. doi: 10.23919/EECSI53397.2021.9624261.

[9] A. P. J. Dwitama and S. Hidayat, "Identifikasi Ujaran Kebencian Multilabel Pada Teks Twitter Berbahasa Indonesia Menggunakan Convolution Neural Network," *Jurnal Sistem Komputer dan Informatika (JSON)*, vol. 3, no. 2, p. 117, Dec. 2021, doi: 10.30865/json.v3i2.3610.

[10] F. D. Ananda and Y. Pristyanto, "Analisis Sentimen Pengguna Twitter Terhadap Layanan Internet Provider Menggunakan Algoritma Support Vector Machine," *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 20, no. 2, pp. 407–416, May 2021, doi: 10.30812/matrik.v20i2.1130.

[11] D. C. Rahmadani, S. Khomsah, and M. Y. Fathoni, "Analisis Emosi Wisatawan Menggunakan Metode Lexicon Text Analysis," *Jurnal Teknik Informatika dan Sistem Informasi*, vol. 10, no. 1, May 2024, doi: 10.28932/jutisi.v10i1.6690.

[12] D. V. Bhargav and D. R, "Performance Analysis of Logistic Regression Algorithm and Random Forest Algorithm for Predicting Product Review Analysis," in *2024 Ninth International Conference on Science Technology Engineering and Mathematics (ICONSTEM)*, IEEE, Apr. 2024, pp. 1–5. doi: 10.1109/ICONSTEM60960.2024.10568774.

[13] A. A. Handoko, M. A. Rosid, and U. Indahyanti, "Implementasi Convolutional Neural Network (CNN) Untuk Pengenalan Tulisan Tangan Aksara Bima," *SMATIKA JURNAL*, vol. 14, no. 01, pp. 96–110, Jul. 2024, doi: 10.32664/smatika.v14i01.1196.

[14] T. He, Y. Liu, Y. Yu, Q. Zhao, and Z. Hu, "Application of deep convolutional neural network on feature extraction and detection of wood defects," *Measurement*, vol. 152, p. 107357, Feb. 2020, doi: 10.1016/j.measurement.2019.107357.

[15] L. Xiaolin, R. C. Panicker, B. Cardiff, and D. John, "Multistage Pruning of CNN Based ECG Classifiers for Edge Devices," in *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, IEEE, Nov. 2021, pp. 1965–1968. doi: 10.1109/EMBC46164.2021.9630588.

[16] M. Harahap, Em Manuel Laia, Lilis Suryani Sitanggang, Melda Sinaga, Daniel Franci Sihombing, and Amir Mahmud Husein, "Deteksi Penyakit Covid-19 Pada Citra X-Ray Dengan Pendekatan Convolutional Neural Network (CNN)," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 6, no. 1, pp. 70–77, Feb. 2022, doi: 10.29207/resti.v6i1.3373.

[17] S. Hong, J. Kim, H.-G. Woo, Y.-C. Kim, and C. Lee, "Screening ideas in the early stages of technology development: A word2vec and convolutional neural network approach," *Technovation*, vol. 112, p. 102407, Apr. 2022, doi: 10.1016/j.technovation.2021.102407.

[18] Merinda Lestandy and Abdurrahim, "Effect of Word2Vec Weighting with CNN-BiLSTM Model on Emotion Classification," *Jurnal Nasional Pendidikan Teknik Informatika (JANAPATI)*, vol. 12, no. 1, pp. 99–107, Mar. 2023, doi: 10.23887/janapati.v12i1.58571.

[19] D. Maheshwari, S. K. Ghosh, R. K. Tripathy, M. Sharma, and U. R. Acharya, "Automated accurate emotion recognition system using rhythm-specific deep convolutional neural network technique with multi-channel EEG signals," *Comput Biol Med*, vol. 134, p. 104428, Jul. 2021, doi: 10.1016/j.compbiomed.2021.104428.

[20] V. A. Porter, B. A. Hobson, B. Foster, P. J. Lein, and A. J. Chaudhari, "Fully automated whole brain segmentation from rat MRI scans with a convolutional neural network," *J Neurosci Methods*, vol. 405, p. 110078, May 2024, doi: 10.1016/j.jneumeth.2024.110078.