Identifikasi Ujaran Kebencian pada Twitter Menggunakan Metode *Convolutional Neural Network* (CNN)

1st Zidan Adhari
Fakultas Informatika
Universitas Telkom
Bandung, Indonesia
zidanadhari@student.telkomuniversity.ac.id

2nd Yulian Sibaroni
Fakultas Informatika
Universitas Telkom
Bandung, Indonesia
yuliant@telkomuniversity.ac.id

Abstrak—Ujaran kebencian merupakan suatu tindak kejahatan yang dijadikan sebagai alat provokasi ke suatu kelompok, yang dimana suatu kelompok tersebut di bagi ke dalam beberapa kelompok seperti ras, warna kulit, gender, cacat, dan kewarnganegaraan. Ujaran kebencian yang terjadi biasanya dalam bentuk kalimat maupun gambar dan disebarluaskan melalui internet. Media sosial Twitter merupakan jejaring sosial yang banyak menampung opini masyarakat tentang apapun yang dapat di sebarluaskan dengan cepat diterima oleh pengguna Twitter yang lain. Pada penelitian sebelumnya akurasi yang didapat sebesar 79% dengan menggunakan metode *Convolutional Neural Network*. Berdasarkan penelitian tersebut, maka dalam penelitian ini dikembangkan sebuah sistem untuk mengidendentifikasi ujaran kebencian dan diklasifikasikan menggunakan metode Convolutional Neural Network dengan membandingkan beberapa *hyperparameter tuning* dan menghasilkan *hyperparameter tuning* terbaik untuk model CNN yaitu *dropout* 0.3 *dan learning rate* 0.001 yang menghasilkan nilai akurasi model CNN sebesar 81%.

Kata kunci—ujaran kebencian, *convolutional neural network* (CNN), *hyperparameter tuning*

Abstract—Hate speech is a crime that is used as a means of provocation to a group, where a group is divided into several groups such as race, skin color, gender, disability, and nationality. Hate speech that occurs is usually in the form of sentences and images and is disseminated via the internet. Twitter social media is a social network that accommodates a lot of public opinion about anything that can be disseminated quickly received by other Twitter users. In previous research, the accuracy obtained was 79% using the Convolutional Neural Network method. Based on this research, in this study a system was developed to identify hate speech and classified using the Convolutional Neural Network method by comparing several tuning hyperparameters and producing the best tuning hyperparameters for the CNN model, namely dropout 0.3 and learning rate 0.001 which resulted in a CNN model accuracy value of 81%.

Keywords—hate speech, convolutional neural network (CNN), hyperparameter tuning

I. PENDAHULUAN

Direktorat Tindak Pidana Siber Bareskrim Polri mendapati 89 konten media sosial terverifikasi mengandung ujaran kebencian selama periode 23 Februari hingga 11 Maret 2021. Konten terbanyak berasal dari Twitter. Berdasarkan data Virtual Police (Dit Tipisiber) Bereskrim Polri pada periode itu 125 konten diajukan untuk diberikan peringatan Virtual Police didominasi platform Twitter 79 konten, Facebook 32 konten, Instagram 8 konten, Youtube 5 konten, dan Whatsapp satu konten[1].

Tindak ujaran kebencian hadir dengan peningkatan yang signifikan dalam interaksi sosial di jejaring sosial online, peningkatan yang terjadi memanfaatkan infrastruktur media sosial. Di Twitter, ujaran kebencian adalah kicauan yang berisi kata-kata kasar yang ditujukan untuk menyerang individu (perundungan siber, politisi, selebriti, produk) atau kelompok tertentu (negara, LGBT, agama, gender, organisasi, dll.). Mendeteksi kebencian seperti pada *tweet* itu penting untuk menganalisis sentimen publik suatu kelompok pengguna terhadap kelompok lain, dan untuk mencegah aktivitas yang salah terkait tentang ujaran kebencian[2]. Hal ini menjadi acuan untuk mengidentifikasi ujaran kebencian di media sosial, khususnya di Twitter, karena pengguna dapat menggunakannya sebagai sarana untuk mengungkapkan ujaran kebencian dengan berbagai fitur yang ditawarkan Twitter.Dikarenakan pada media sosial Twitter dinilai dapat menggiring opini masyarakat dengan mudah, maka dengan adanya penelitian ini diharapkan dapat membantu masyarakat untuk mengidentifikasi ujaran kebencian pada Twitter, apakah hasilnya mengandung ujaran kebencian atau tidak pada hastag yang sedang ramai diperbicarakan di media sosial Twitter.

Pembelajaran *Deep Learning* dan penggunaan fitur seperti *bag of words*, *word,* dan *n-grams* sangat efektif untuk mendeteksi ujaran kebencian dan hasilnya menunjukkan bahwa kinerja yang lebih baik daripada metode *Machine Learning* dalam mendeteksi ujaran kebencian[3]. Penerapan metode Convolutional Neural Network (CNN) untuk klasifikasi teks juga menghasilkan performasi yang lebih baik dari metode *Decision Tree, Support Vector Machine,* dan *Naïve Bayes*[4]. Pada penelitian ini akan digunakan metode klasifikasi deep learning Convolutional Neural Network (CNN). CNN merupakan salah satu deep neural network yang telah terbukti secara efektif menyelesaikan beberapa pemrosesan Bahasa seperti penandaan kalimat, analisis sentimen dan pengenalan entitas nama. Dengan berbagai fitur CNN dapat digunakan untuk kategorisasi ujaran kebencian berdasarkan vektor kata informasi semantik yang dibuat untuk semua token menggunakan algoritma *unsupervised learning* dan *word2vec*[5].

Pada penelitian [3] membandingkan antara metode *Convolutional Neural Network* (CNN) dan *Recurrent Neural Network* (RNN) untuk mendeteksi ujaran kebencian dalam Tweet berbahasa Arab, yang menghasilkan model CNN memberikan kinerja terbaik dengan F1 skor 0,79. Lalu, pada penelitian [5] membahas tentang membangun sistem klasifikasi teks ujaran kebencian menggunakan CNN dengan dibagi menjadi empat kategori yaitu rasisme, seksisme, netral, dan non ujaran kebencian. Sistem dibangun dengan menggunakan *word2vec* yang diuji 10 kali yang menghasilkan bahwa *word2vec* embeddings berkinerja dengan baik dengan nilai presisi lebih tinggi dari recall dan F1-skor 78,3% daripada menggunakan *fast text embedding*. Kekurangan pada penelitian [5] adalah hanya fokus pada membandingkan *word embedding* antara *word2vec* dengan *fast text* tetapi tidak menerapkan perbandingan *hyperparameter tuning* agar mendapatkan akurasi tertinggi. Pada penelitian [6] digunakan model *deep learning* yaitu metode *Convolutional Neural Network* (CNN) dan *Reccurent Neural Network* (RNN) untuk klasifikasi kata menghasilkan bahwa metode RNN dapat menangani klasifikasi kata ini dengan baik. Meskipun demikian, metode RNN tidak dapat memparalelkan kata dengan baik dan lebih cocok untuk pemrosesan teks yang pendek. Waktu pelatihan juga lama saat memproses teks dengan lebih dari puluhan kata. CNN telah banyak digunakan dalam model klasifikasi teks panjang karena dapat memparalelkan kata dengan baik.

Dibandingkan RNN, CNN menggunakan beberapa *channel*, CNN memilih untuk menggunakan ukuran *filter* yang berbeda dan *max pooling* untuk memilih kata yang berpengaruh dan karakteristik klasifikasi *low-latitude* yang lebih rendah kemudian menggunakan *dropout* dari lapisan *full connection* dengan fitur ekstraksi sebagai hasil klasifikasi akhir.

Berdasarkan kelebihan metode CNN tersebut, dalam tugas akhir ini diusulkan menggunakan metode *Convolutional Neural Network* untuk mengidentifikasi ujaran kebencian pada Twitter dengan menggunakann *hyperparameter learning rate* [0.01,0.001,0,0001] dan *batch size* 256, nilai *hyperparameter* yang dipilih memberikan solusi yang lebih baik[25]. Diharapkan hasil dari penelitian ini dapat mengetahui performansi metode klasifikasi CNN untuk mengidentifikasi ujaran kebencian pada Twitter dan juga dapat mengetahui hasil *hyperparameter tuning* model terbaik pada CNN.

. Batasan masalah pada penelitian ini adalah menggunakan dataset yang bersumber dari hasil data *crawling* yang dilakukan berdasarkan teks yang mengandung ujaran kebencian pada *twitter*, lalu teks yang dapat di identifikasi hanya menggunakan Bahasa Indonesia.

II. KAJIAN TEORI

Twitter merupakan jejaring sosial yang dikelola secara real time menghubungkan penggunanya dengan cerita, ide, pendapat, dan berita baru tentang apa saja yang dianggap menarik oleh banyak orang[7]. Tapi terkadang komunikasi antar pengguna berubah menjadi kasar karena perbedaan pendapat, Bahasa seperti itu bisa bermacam-macam bentuk, seperti intimidasi yang merupakan salah satu alasan utama di balik ujaran kebencian bisa terjadi[8].

Ujaran kebencian adalah tindakan menyebarkan kebencian terhadap individu atau kelompok atas dasar suku, agama, ras, dan karakteristik lain yang dapat menyebabkan diskriminasi, kekerasan, dan konflik sosial, beberapa peneliti telah melaporkan rasisme sebagai jenis ujaran kebencian yang paling umum di platform media sosial, dengan 48,73% *tweet* ujaran kebencian twitter menjadi rasis[8][9].

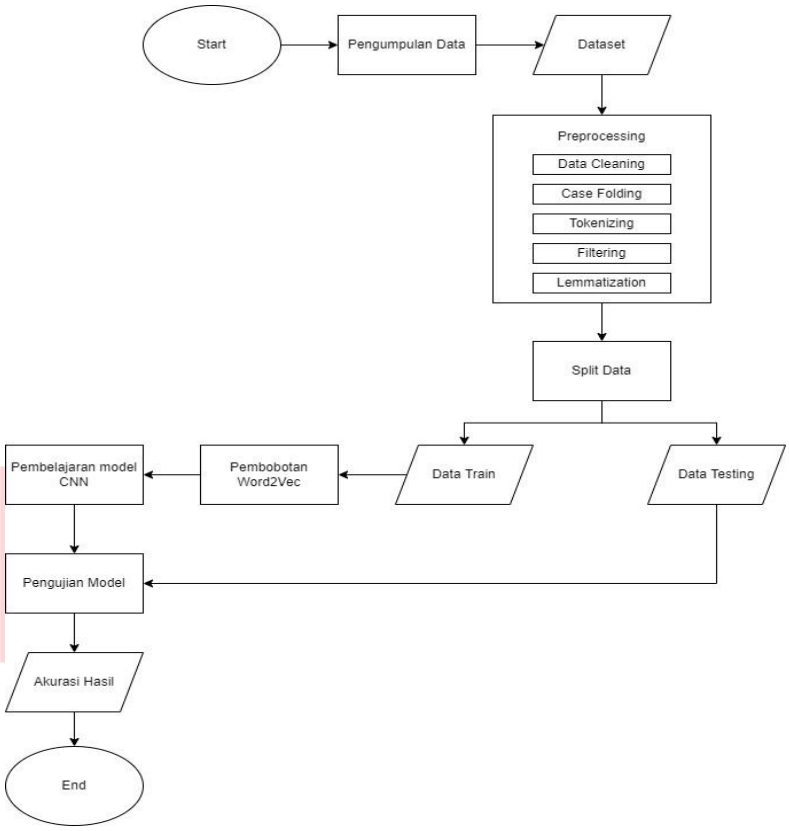
Penelitian ini menggunakan representasi vektor *Word2Vec*. *Word2vec* merupakan salah satu *word embedding* yang paling banyak digunakan, kemampuannya untuk melatih model dalam jumlah data yang sangat besar dengan kompleksitas komputasi yang lebih rendah dan digunakan untuk memproses teks sebelum teks diterima oleh algoritma *deep learning*[11][12].

Pada penelitian [5] *Convolutional Neural Network* digunakan untuk klasifikasi ujaran kebencian pada Twitter dengan menghasilkan tingkat akurasi yang tinggi dan memiliki performansi presisi yang baik dalam mengklasifikasikan tweet daripada algoritma lainnya. Penelitian [2] mengklasifikasikan tweet yang mengandung ujaran kebencian dengan menggunakan beberapa *deep learning* seperti *FastText*, *Convolutional Neural Network* (CNN), dan *Long Short-Term Memory Network* (LSTM). Pengujian tersebut menghasilkan bahwa performansi CNN lebih baik dari metode LSTM yang perfomansinya melebihi *FastText*.

Pada penelitian [13] mengidentifikasi adanya efek potensial yang dikarenakan *User Feature* dalam klasifikasi ujaran kebencian. Dengan adanya penelitian ini bertujuan untuk lebih memahami karakteristik pengguna dengan tidak adanya korelasi antara teks kebencian dan karakteristik pengguna yang telah memposting *tweet* dan pada penelitian [14] melakukan riset tentang pengunaan CNN untuk mendeteksi ujaran kebencian pada gambar dan menghasilkan sebuah aplikasi berbasis *website* yang dapat mendeteksi ujaran kebencian pada gambar melalui teks yang terdapat pada gambar.

III. METODE

A. Gambaran Sistem



GAMBAR 1
SISTEM KLASIFIKASI MENGGUNAKAN METODE CNN

Pada Gambar 1 menjelaskan tentang sistem yang akan dibangun mulai dari pengumpulan data *crawling* dari Twitter lalu selanjutnya pelabelan data sampai pengujian model *Convolutional Neural Network* (CNN). Setelah itu dilakukan akurasi hasil yang telah didapatkan dari pengujian model yang menghasilkan prediksi dari model dan dilakukan evaluasi untuk menguji performansi metode CNN.

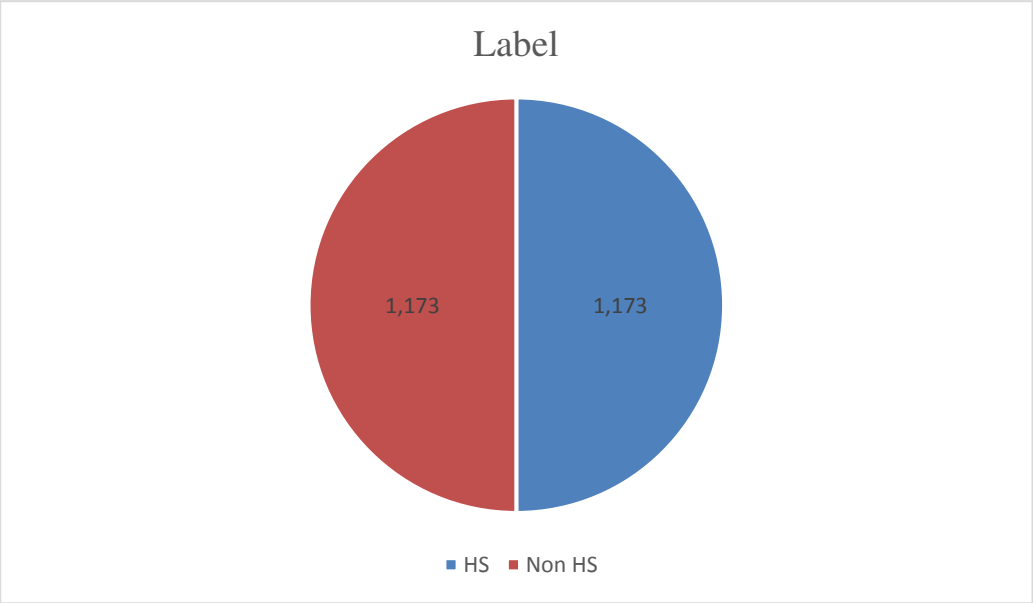
B. Pengumpulan Data

Pengumpulan data dalam tugas akhir ini menggunakan Twitter API dengan *library Tweepy* yang diambil dari tanggal 03 Agustus 2021 hingga 15 Maret 2022. Data yang digunakan berasal dari *hastag* seperti: #Tolak3Periode, #PembajakKonstitusi, #JokowiKejarSetoran, dan #BubarkanKhilafatulMuslimin. Hasil dari pengumpulan data ini sebanyak 3.365 data. Pelabelan data dilakukan oleh 2 dengan tujuan untuk mengurangi subjektivitas dalam melakukan pelabelan. Lalu proses pelabelan juga dilakukan secara manual dengan dua kategori yaitu HS dan Non HS. Berikut adalah hasil dari pengambilan data *tweet*.

TABEL 1
HASIL PENGAMBILAN DATA *TWEET*

No.	HS	Non HS
1	Kadrun jenis baru ini mah... #BubarkanKhilafatulMuslimin #BubarkanKhilafatulMuslimin #BubarkanKhilafatulMuslimin https://t.co/0Q3AdCnIBE	Spoiler Lanjutan One Piece 1052: Edan! Kedatangan Angkatan Laut Admiral Greenbull ke Wano, Mau Ngapain? !!!!!!!!!! !!!!!!!!! !!!! #ONEPIECE1052 #B9352MW #FormulaESuksesMenduni a Sabrina Chairunnisa FPI Palsu #JokowiKejarSetoran iOS 16 #BoycottIndia https://t.co/Oe00ABDitM"
2	Nikmatnya uang rakyat #JokowiKejarSetoran #JokowiKejarSetoran	@rinidyah6 Kita harus Demi Masa Depan Bangsa Negara lebih baik.
3	Mau ditentang didemo diboikot tetap aja jalan terus sma keputusannya. Rezim ini memang amat bebal. Ga ada obat. #Tolak3Periode	AHY : Penundaan pemilu adalah pemufakatan jahatLenteng Agung Jawa Menteri Klithih Wak Haji

Pesebaran data yang sudah dilabeli dapat dilihat pada gambar 2.



GAMBAR 2.
KELAS LABEL

Pada gambar 2 diatas merupakan *dataset* yang sudah masing-masing kelasnya seimbang yang berjumlah 1.173 dan data sebelumnya berjumlah 3.365 data. Proses penyeimbangan data ini sangat penting untuk menghitung hasil akurasi model[15].

C. Preprocessing Data

Pada tahap ini, data yang telah di *crawling* akan diseleksi dan kemudian diproses oleh sistem pada setiap dokumennya. Proses *preprocessing* antara lain:

1.. *Data Cleaning*
Pada proses data *cleaning* dilakukan penghapusan karakter HTML, ikon, username, url, surat elektronik. Data *cleaning* bertujuan untuk meringankan beban dalam proses komputasi dan juga menghemat waktu[15].

2. *Case Folding*
Pada tahap ini dilakukan perubahan semua huruf yang ingin diproses menjadi huruf kecil. Hanya huruf ‘a’ sampai ‘z’ yang diterima. Selain dari huruf tersebut akan dihilangkan[16].

3.. *Tokenizing*
Pada tahap ini dilakukan pemotongan terhadap string input berdasarkan tiap kata yang membangunnya. Selain itu, spasi digunakan untuk memisahkan antar kata[17].

4. *Filtering*
Pada tahap ini dilakukan pemilihan kata penting dari hasil sebelumnya yaitu tokenizing. Proses ini menggunakan algoritma stoplist (membuang kata yang tidak penting) dan wordlist (menyimpan kata penting) dengan menggunakan kamus list *Indonesian* [18].

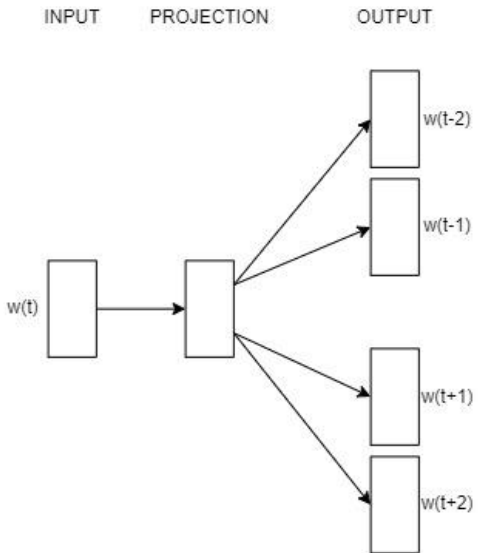
5. *Lemmatization*
Pada tahap ini dilakukan pertimbangan morfologi analisis kata-kata yang diinfleksikan dengan benar dengan kata dasar dari kata tersebut. *Lemmatization* yang digunakan berasal dari library nltk yaitu *WordNetLemmatizer*[19].

D. *Split Data*

Dataset yang digunakan akan dibagi menjadi dua yaitu data *training* dan data *testing*. Dengan rasio 80% data *training* dan 20% data *testing*. Data *training* akan digunakan untuk *training embedding* menggunakan model *Word2Vec* sedangkan data *testing* digunakan untuk melakukan evaluasi performansi.

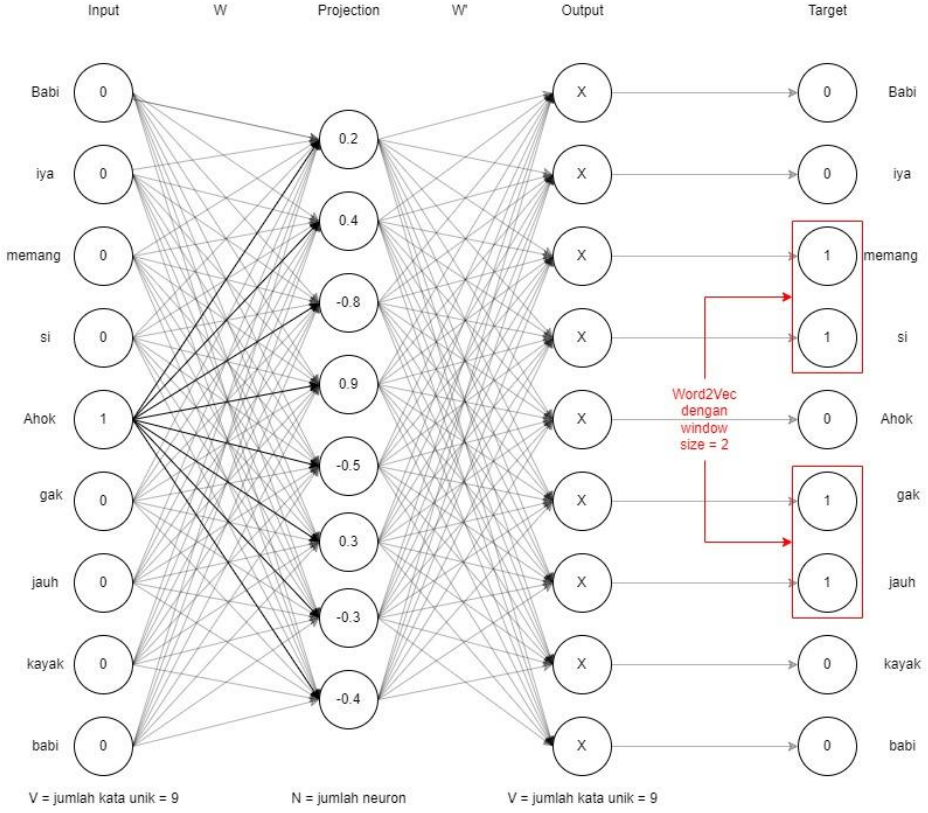
E. Word2Vec

Metode *Word2Vec* memiliki 2 buah arsitektur, yaitu *Continuous Bag of Word* (CBOW) dan *Skip-Gram*. *Word2vec* mengambil korpus teks sebagai input dan menghasilkan vektor kata sebagai keluaran, representasi vektor yang diperoleh setelah *Word2vec* membangun kosakata dari *train* korpus. Hasil vektor kata digunakan sebagai fitur untuk algoritma *Convolutional Neural Network*[12].



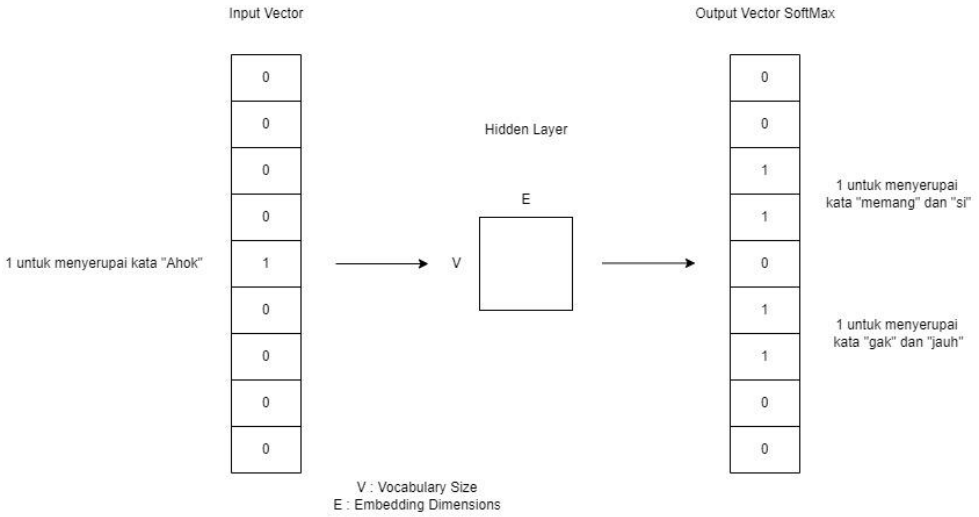
GAMBAR 2
ARISTEKTUR *SKIP-GRAM*

Word2Vec terdiri dari banyak fitur yang berguna untuk pemrosesan *Natural Language Processing* yang berbeda-beda. Makna semantik yang diberikan oleh *word2vec* untuk setiap kata dalam representasikan vektor yang telah disajikan yang sangat berguna dalam klasifikasi teks pada *deep learning*. *Word2vec* digunakan untuk menemukan analogi, sintaksis, dan analisis semantic dari setiap kata[20].



GAMBAR 3
ILUSTRASI WORD2VEC SKIP-GRAM

Dalam penelitian ini, penulis melakukan *pre-trained* vektor yang dilatih terlebih dahulu dengan menggunakan arsitektur *Word2vec Skip-Gram* karena sistem melakukan prediksi terhadap *pre-trained* vektor berdasarkan penggunaan kata dan teks akan diubah menjadi vektor *low-latitude* sebagai input dari model CNN yang ditingkatkan untuk mencapai klasifikasi kata yang maksimal[21].

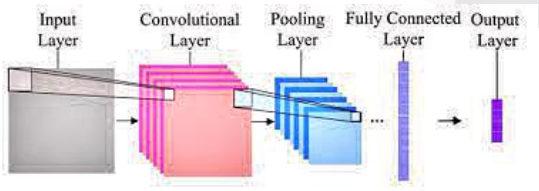


GAMBAR 4
CONTOH REPRESENTASI VEKTOR WORD2VEC

Pada gambar 3 Penulis mengilustrasikan penggunaan *Word2vec Skip-Gram* dengan mengambil contoh dari data yang digunakan yaitu sebuah kalimat "Babi iya memang si Ahok gak jauh kayak Babi" dengan *window size* = 2 dan *current word* = 'Ahok'. Dalam *Skip-Gram*, lapisan *input* terdiri dari vektor kosakata (R x V) dimana R adalah jumlah sampel dalam pelatihan dan V adalah ukuran kosakata. Setiap kata dalam kosakata direpresentasikan oleh *encoded vector*. Pada lapisan *output*, nilai vektor menahan angka probabilitas untuk setiap kata dalam kosakata untuk *input* kata yang diberikan.

F. Pembelajaran *Convolutional Neural Network*

Convolutional Neural Network (CNN) merupakan salah satu algoritma *deep learning* yang memiliki penerapan fungsi untuk mengklasifikasi teks. Pada dasarnya CNN memiliki arsitektur yang terdiri dari *convolutional layers, pooling layers,* dan *fully connected layers*. Oleh karena itu, CNN memiliki kecepatan dalam perhitungan dalam setiap pengujiannya. CNN juga memiliki efek *handling* yang baik dalam aspek *natural language processing*[22].



GAMBAR 5
ARSITEKTUR *CONVOLUTIONAL NEURAL NETWORK*

Fungsi *convolutional layers* adalah untuk mengekstrak informasi semantik dalam kalimat yang sebelumnya telah di *train* dengan model *word embedding*. Masing-masing *convolutional kernel* mengekstrak kedalam beberapa bagian fitur pada teks.

Setelah operasi *convolutional layers* telah selesai, teks akan diproses oleh *pooling layers*. *Pooling layers* berfungsi untuk mengurangi kompleksitas dimensi.

Lalu setelah *pooling layers* telah bekerja dengan baik, teks akan memasuki bagian *fully connected* untuk diklasifikasin yang berdasarkan kelas yang telah ditentukan yaitu HS dan Non HS.

TABEL 2.
ARSITEKTUR MODEL CNN YANG DIGUNAKAN

Input Layer	***Embedding Layer***	***Conv1D Layer***	***Global Max Pooling***	***Dropout Layer***	***Strides***
(*None*, 50)	(*None*, 50, 200)	(*None*, 49, 100)	(*None*, 100)	(*None*, 256)	(*None*, 1)

Convolutional Neural Network (CNN) digunakan untuk mengekstrak kemiripan informasi penting *n-gram* pada suatu kalimat. Pada CNN ini terdiri dari beberapa arsitektur yaitu *input layer*, *embedding layer, convolutional layer,* dan *max-pooling layer*. *Input layer* berfungsi sebagai *preprocessing* teks data sebelum diproses oleh model, *embedding layer* untuk mengekstraksi *input* teks dari *input layer*, *convolutional layer* berfungsi untuk mendapatkan hasil dari ukuran *convolutional layer* yang sudah ditetapkan, dan *max-pooling layer* berfungsi untuk mengurangi dimensi lapisan dari *convolutional layer*[6].

Berdasarkan Tabel 2, lapisan *input data* pertama kali dilakukan secara implisit atau dikatakan sebagai *input layer*. Lalu pada *embedding layer* dengan *output shape* (*None*, 50, 200) dengan 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 bertujuan untuk membangun proyeksi vektor dengan

nilai proyeksi sebesar 200 sebesar dengan nilai variable *embedding dim*.

Pada lapisan conv1d adalah lapisan yang menerima hasil keluaran dari lapisan *embedding*. Lapisan dengan (*None*, 49, 100) yang berfungsi membuat kernel untuk melakukan *filter* dari *output* lapisan *embedding*. Terdapat 100 filter untuk melakukan konvolusi dengan ukuran *kernel size 2* dan *strides* sebanyak 1, maka akan menghasilkan vektor 1 x 49. Kemudian, hasil nilai maksimum dari perhitungan konvolusi oleh kernel diambil oleh lapisan global_max_pooling_1d dan ditetapkan sebagai nilai elemen kata terkuat. Untuk 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 3.

TABEL 3
LAPISAN MODEL CNN YANG TERBENTUK

Layer(type)	***Output Shape***	***Param#***
embedding (Embedding)	(*None*, 50, 200)	2000000
conv1d(Conv1D)	(*None*, 49, 100)	40100
global_Max_pooling1d (Global)	(*None,* 100)	0
dropout (Dropout)	(*None,* 256)	0
strides (Dense)	(*None,* 1)	257
Total params: 2,263,313		
Trainable params: 2,263,313		
Non-trainable params: 0		

1. *Hyperparameter Tuning*

Hyperparameter tuning dilakukan untuk melatih model dengan melakukan beberapa kombinasi parameter yang telah ditentukan. Pada proses ini dilakukan terhadap 2.346 data yang sebelumnya telah di seimbangkan. Pada *hyperparameter* ini terdiri dari 4 struktur yaitu *Kernel Size, Embed-dim, Number filter,* dan *Output size*.

TABEL 4
NILAI STRUKTUR *HYPERPARAMETER* YANG DIGUNAKAN

Filter Size	***Number of Filters***	***Dropout***	***Dense Layer***	***Dense Layer 2***
1 (unigram)	100	0.2, 0.3, 0.4	256	1
2 (bigram)	100	0.2, 0.3, 0.4	256	1
3 (trigram)	100	0.2, 0.3, 0.4	256	1
4 (fourgram)	100	0.2, 0.3, 0.4	256	1

Pada penelitian ini, nilai parameter yang digunakan dapat dilihat pada tabel 4. Penelitian [23] yang sebelumnya menggunakan nilai *filter size* [1,2,3], *number of filters* [100,256], *dense layer* [128,256] dan *dense layer* 1 sebesar [32,64,128] dan pada penelitian [24] melakukan penggabungan *filter size* untuk mencapai hasil yang maksimal terbukti dengan hasil yang meningkat secara signifikan dari percobaan yang sebelumnya.

. Penelitian ini menggunakan *filter size* [1,2,3,4], *number of filters* 100 dan *dropout* [0.2,0.3,0.4] untuk mendapatkan letak nilai yang dapat memberikan hasil terbaik dari parameter yang sudah ditentukan dengan menggabungkan keempat *filter size* dan memperbanyak nilai kombinasi oleh sebab itu menggunakan *unigram, bigram, trigram,* dan *fourgram*

TABEL 5
NILAI *HYPERPARAMETER* YANG AKAN DIUJI

Hyperparameter	**Nilai**
Learning Rate	0.01,0.001,0.0001
Batch Size	256

Pada tabel 5 diatas merupakan nilai parameter *learning rate* dan *batch size* yang akan diuji. Berdasarkan penelitian [25] *Learning rate* 0.01, 0.001, dan 0.0001 digunakan untuk membandingkan penggunaan *learning rate* dengan hasil yang terbaik dan memperoleh hasil yang baik dengan pelatihan yang cepat. *Batch size* ukuran 256 digunakan untuk proses *training* dapat melatih banyak data. Oleh karena itu, penulis menggunakan nilai *hyperparameter* tersebut agar mendapatkan hasil uji yang baik dan pelatihan yang cepat.

G. Pengujian Model dengan *Confusion Matrix*

Confusiion Matrix hanya menunjukkan prediksi dan membandingkan nilai aktual dan nilai prediksi pada model.

TABEL 6
NILAI *CONFUSION MATRIX*

	Aktual Positif	**Aktual Negatif**
Prediksi Positif	*True Positive* (TP)	*False Positive* (FP)
Prediksi Negatif	*False Negative* (FN)	*True Negative* (TN)

Pada tabel 6, menjelaskan empat nilai yang dihasilkan di dalam tabel *confusion matrix. Confusion*

matrix akan menghitung nilai *accuracy* untuk mendapatkan nilai performa dari model CNN karena dilakukannya optimasi pada arsitektur CNN mempengaruhi nilai akurasi yang dihasilkan[14]. Rumus untuk menghitung nilai akurasi dapat dilihat pada persamaan berikut.

$$Akurasi = \frac{TP+TN}{TP+FP+FN+TN} \quad (1)$$

IV. HASIL DAN PEMBAHASAN

Pada penelitian ini menggunakan *dataset* yang berjumlah 2.346 data *tweet* hasil dari pengumpulan data menggunakan *Twitter API*. Evaluasi dilakukan dengan beberapa skenario pengujian yang bertujuan untuk mengetahui kombinasi nilai *hyperparameter tuning* dan membandingkan performansi antara dua *word embedding* yaitu *word2vec* dan *python library keras*.

A. Skenario dan Hasil Pengujian

Dalam penelitian ini memiliki 2 skenario, yaitu :

a. Menentukan representasi vektor terbaik untuk model pelatihan dan validasi.
b. Melakukan performansi metode CNN menggunakan *Hyperparameter tuning*.

Pada pengujian awal melakukan perbandingan performansi hasil antara representasi vektor kata hasil *embedding python library keras* dan penyempurnaan kata representasi vektor kata yang telah dilatih terlebih dahulu. Pengujian ini dilakukan bertujuan untuk menentukan representasi vektor terbaik untuk metode CNN dengan mengambil hasil dari akurasi *training* dan akurasi validasi terbaik antara kedua *word embedding*.

Pada skenario kedua melakukan pengujian performansi metode CNN dengan *hyperparameter tuning* dengan menentukan parameter terbaik yaitu kombinasi antara nilai *learning rate* dan *dropout* yang menghasilkan nilai akurasi *training* dan validasi terbaik.

1. Hasil Pengujian dan Analisis Skenario 1

Pada skenario ini dataset yang sudah melakukan tahap *preprocessing* dan data *pre-trained* pada *Word2vec* sebelum diuji oleh model CNN akan memasuki proses persiapan model CNN yaitu menentukan representasi vektor terbaik untuk model pelatihan dan validasi. Pada penelitian ini terdapat dua representasi vektor kata yang akan di uji yaitu representasi vektor kata hasil *embedding python library keras* dan penyempurnaan kata representasi vektor kata yang telah dilatih terlebih dahulu. Pengujian dilakukan dengan menggunakan K-fold dengan nilai K=5, untuk hasil pengujian dapat dilihat pada tabel 5.

TABEL 7
HASIL AKURASI UNTUK BERBAGAI NILAI *EPOCH*

No	Representasi Vektor	Akurasi	
		Training	Validasi
1	**Representasi Vektor Dengan *Python Library Keras***		
	Epoch-1	0.6919	0.7766
	Epoch-2	0.8358	0.8106
	Epoch-3	0.9179	0.8128
	Epoch-4	0.9622	0.8170
	Epoch-5	0.9717	0.7979
2	**Penyempurnaan Representasi Vektor**		
	Epoch-1	0.7639	0.7766
	Epoch-2	0.8555	0.8085
	Epoch-3	0.9067	0.7574
	Epoch-4	0.9408	0.7830
	Epoch-5	0.9600	0.7872

Pada tabel 7 menampilkan hasil akurasi *training* dan validasi yang dilakukan oleh representasi vektor dengan *python library keras* dan penyempurnaan representasi vektor kata yang dilatih terlebih dahulu. Pada tabel 7 menunujukkan bahwa hasil akurasi validasi terbaik yaitu pada *epoch-4* dengan nilai 0.8170 menggunakan menggunakan representasi vektor kata dengan *python library keras* dan hasil akurasi *training* terbaik terdapat pada *epoch-5* dengan nilai 0.9717 menggunakan representasi vektor kata dengan *python library keras*. Berdasarkan hasil pada tabel 7 ini, untuk pemodelan pelatihan dan validasi akan menggunakan representasi vektor dengan *embedding python library keras*.

2. Hasil Pengujian dan Analisis Skenario 2

Dataset yang sebelumnya sudah dilatih di representasi vektor dengan *python library keras,* dataset selanjutnya akan dilatih oleh model CNN

dengan beberapa nilai *learning rate* dan *dropout*. Pengujian dilakukan dengan nilai *learning rate* 0.01, 0.001, dan 0.0001. Lalu dengan nilai dropout 0.2, 0.3, dan 0.4. Berikut adalah hasil pengujian dengan menggunakan nilai-nilai tersebut.

TABEL 8
HASIL PENGUJIAN OLEH CNN DENGAN NILAI *DROPOUT* 0.2 UNTUK BERBAGAI NILAI *LEARNING RATE*

K-fold	**Kombinasi Parameter**					
	***Learning Rate* = 0,01**		***Learning Rate* = 0,001**		***Learning Rate* = 0,0001**	
	Training	Validasi	Training	Validasi	Training	Validasi
K-Fold = 1	0.7327	0.8431	0.8520	0.8351	0.8080	0.8378
K-Fold = 2	0.9147	0.9653	0.9340	0.9760	0.9087	0.8827
K-Fold = 3	0.9687	0.9680	0.9767	0.9867	0.9187	0.9413
K-Fold = 4	0.9654	0.9867	0.9820	0.9947	0.9600	0.9867
K-Fold = 5	0.9567	0.9387	0.9873	0.9813	0.9813	0.9707

TABEL 9
HASIL PENGUJIAN OLEH CNN DENGAN NILAI *DROPOUT* 0.3 UNTUK BERBAGAI NILAI *LEARNING RATE*

K-fold	**Kombinasi Parameter**					
	***Learning Rate* = 0,01**		***Learning Rate* = 0,001**		***Learning Rate* = 0,0001**	
	Training	Validasi	Training	Validasi	Training	Validasi
K-Fold = 1	0.7340	0.8112	0.8440	0.8298	0.8173	0.8298
K-Fold = 2	0.9127	0.9707	0.9327	0.9760	0.9187	0.8613
K-Fold = 3	0.9660	0.9520	0.9687	0.9867	0.9141	0.9387
K-Fold = 4	0.9587	0.9813	0.9793	0.9973	0.9547	0.9760
K-Fold = 5	0.9720	0.9573	0.9900	0.9600	0.9727	0.9813

TABEL 10
HASIL PENGUJIAN OLEH CNN DENGAN NILAI *DROPOUT* 0.4 UNTUK BERBAGAI NILAI *LEARNING RATE*

K-fold	**Kombinasi Parameter**					
	***Learning Rate* = 0,01**		***Learning Rate* = 0,001**		***Learning Rate* = 0,0001**	
	Training	Validasi	Training	Validasi	Training	Validasi
K-Fold = 1	0.7393	0.8032	0.7000	0.8378	0.8080	0.8298
K-Fold = 2	0.8941	0.9467	0.9334	0.9760	0.9181	0.8667
K-Fold = 3	0.9507	0.9493	0.9813	0.9813	0.9101	0.9333
K-Fold = 4	0.9567	0.9627	0.9820	0.9947	0.9560	0.9733
K-Fold = 5	0.9574	0.9280	0.9873	0.9787	0.9707	0.9813

Dari ketiga tabel yang sudah menampilkan hasil dari masing-masing *dropout* dan *learning rate*, menghasilkan bahwa pada tabel 9 dengan *dropout* 0.3 dan *learning rate* 0.001 mendapatkan kombinasi nilai terbaik antara nilai akurasi *training* pada *K-fold* 4 dan *K-fold* 5 sebesar 0.9793 dan 0.9900. Sedangkan pada nilai akurasi validasi pada *K-fold* 4 sebesar 0.9973. Berikut ini adalah gambar *confusion matrix* dari nilai *dropout* 0,3 dan *learning rate* 0,001.

TABEL 11.
CONFUSION MATRIX DARI MODEL CNN

		Predicted Class	
		0	1
Actual Class	0	TN = 175	FP = 66
	1	FN = 21	TP = 208
	Total	196	274

Tabel 11 menampilkan *confusion matrix* dan dapat dibaca sebagai berikut :

a. Label negatif yang diprediksi negatif sebanyak 175, label negatif yang di prediksi positif sebanyak 21 yang ditunjukkan pada tabel 11.
b. Label positif yang diprediksi positif sebanyak 208, label positif yang di prediksi negatif sebanyak 66 yang ditunjukkan pada tabel 11.

Tabel 11 menampilkan hasil dari data *testing* yang menggunakan nilai *dropout* 0,3 dan *learning rate* 0,001 terdapat *tweet* yang diprediksi negatif sebanyak 175 dan *tweet* yang diprediksi positif sebanyak 208 dengan total masing-masing yaitu 196 dan 274 bila dijumlahkan menjadi 470 yang dimana sesuai dengan tahap *split data* dengan data *testing* sebesar 20% dari data keseluruhan dan menunjukkan bahwa dengan nilai *hyperparameter* tersebut menghasilkan akurasi yang baik yaitu sebesar 81% .

TABEL 12
CONTOH HASIL PREDIKSI YANG TELAH DIUJI OLEH MODEL

No	**Tweet**	***Actual***	***Predicted***
1	auzdubillahiminassyaitonirrojim bismillahirahmanirrahim insyaallahuallahu akbarallahu akbar	0	1
2	semoga pak ahok selalu dalam lindungan tuhan, saya sayang sama pak ahok, pak ahok orang baik	0	1

Pada tabel 12 menampilkan contoh hasil prediksi yang telah di uji oleh model tetapi nilai aktual dan nilai prediksi berbeda yang disebabkan oleh kalimat yang bias dan mengakibatkan model tidak dapat memprediksi dengan benar sesuai dengan nilai aktual dari kalimat tersebut.

B. Analisis Hasil Pengujian

Pada penelitian ini, telah dilakukan beberapa skenario pendukung untuk menentukan *hyperparameter tuning* terbaik dan mengeluarkan hasil perfomansi terbaik pada model CNN. Pada tabel 7 telah menentukan representaasi vektor kata terbaik antara penyempurnaan representasi vektor kata dan *embedding python library keras*, dengan hasil pada tabel tersebut membuktikan bahwa *embedding python library keras* adalah representasi vektor kata terbaik dengan nilai akurasi validasi terbaik yaitu pada *epoch-4* dengan nilai 0.8170 dan hasil akurasi *training* terbaik terdapat pada *epoch-5* dengan nilai 0.9717. Pengujian tersebut dilakukan secara berkali-kali untuk mendapatkan hasil yang terbaik dan maksimal.

Pada tabel 9 dilakukan pengujian dari masing-masing *dropout* dan *learning rate* untuk mendapatkan kombinasi nilai terbaik antara nilai akurasi training dan nilai akurasi validasi. Dari pengujian tersebut didapatkan nilai akurasi training sebesar 0.9793 dan 0.9900 pada *K-fold* 4 dan *K-fold* 5, lalu nilai akurasi validasi sebesar 0.9973 pada *K-fold* 4. Dengan hasil tersebut nilai *learning rate* 0,001 dan *dropout* 0,3 adalah kombinasi nilai terbaik yang selanjutnya akan di uji perfomansi pada model CNN yang menghasilkan nilai akurasi sebesar 81%.

V. KESIMPULAN

Berdasarkan hasil penelitian dari pengujian dan analisa yang telah dilakukan, menyimpulkan bahwa metode klasifikasi CNN dengan representasi vektor kata menggunakan *word2vec* untuk mengidentifikasi ujaran kebencian pada twitter mendapatkan nilai performansi yang memuaskan. Hasil kombinasi *hyperparameter tuning* CNN terbaik ada di *learning rate* 0,001 dan *dropout* 0,3 yang menghasilkan akurasi 81%.

Untuk penelitian selanjutnya, dapat dikembangkan dengan penambahan dataset agar proses pelatihan program dapat melatih lebih banyak variasi kata. penelitian ini pelabelan data dilakukan secara manual, diharapkan pada penelitian selanjutnya akan lebih baik melibatkan orang yang ahli dalam menentukan frasa untuk menentukan apakah mengandung ujaran kebencian atau tidak, agar menghasilkan nilai lebih akurat dan tidak bias.

REFERENSI

[1] N. Zuraya, "Virtual Police: Konten Ujaran Kebencian Terbanyak di Twitter," *Republika*, 2021. https://www.republika.co.id/berita/qputx3383/emvirtual-policeem-konten-ujaran-kebencian-terbanyak-di-twitter.

[2] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, "Deep Learning for Hate Speech Detection in Tweets," vol. 2, no. 2, pp. 427–431, 2017, doi: 10.18653/v1/e17-2068.

[3] R. Alshalan and H. Al-Khalifa, "A deep learning approach for automatic hate speech detection in the saudi twittersphere," *Appl. Sci.*, vol. 10, no. 23, pp. 1–16, 2020, doi: 10.3390/app10238614.

[4] A. Gune and M. Nene, "Convolutional Neural Networks for Text Categorization with Latent Semantic Analysis," *2017 Int. Conf. Energy, Commun. Data Anal. Soft Comput.*, pp. 499–503, 2017.

[5] B. Gambäck and U. K. Sikdar, "Using Convolutional Neural Networks to Classify Hate-Speech," no. 7491, pp. 85–90, 2017, doi: 10.18653/v1/w17-3013.

[6] J. Cai, J. Li, W. Li, and J. Wang, "Deeplearning Model Used in Text Classification," *2018 15th Int. Comput. Conf. Wavelet Act. Media Technol. Inf. Process. ICCWAMTIP 2018*, pp. 123–126, 2019, doi: 10.1109/ICCWAMTIP.2018.8632592.

[7] S. Ro, "Hate Speech Detection in the Indonesian Language: A Dataset and Preliminary Study," pp. 473–481, 1999, [Online]. Available: https://support.twitter.eom/articles/l.

[8] G. Koushik, K. Rajeswari, and S. K. Muthusamy, "Automated hate speech detection on Twitter," *Proc. - 2019 5th Int. Conf. Comput. Commun. Control Autom. ICCUBEA 2019*, 2019, doi: 10.1109/ICCUBEA47591.2019.9128428.

[9] A. Marpaung, R. Rismala, and H. Nurrahmi, "Hate Speech Detection in Indonesian Twitter Texts using Bidirectional Gated Recurrent Unit," *KST 2021 - 2021 13th Int. Conf. Knowl. Smart Technol.*, pp. 186–190, 2021, doi: 10.1109/KST51265.2021.9415760.

[10] O. Istaiteh, R. Al-Omoush, and S. Tedmori, "Racist and Sexist Hate Speech Detection: Literature Review," *2020 Int. Conf. Intell. Data Sci. Technol. Appl. IDSTA 2020*, pp. 95–99, 2020, doi: 10.1109/IDSTA50958.2020.9264052.

[11] H. Mohaouchane, A. Mourhir, and N. S. Nikolov, "Detecting Offensive Language on Arabic Social Media Using Deep Learning," *2019 6th Int. Conf. Soc. Networks Anal. Manag. Secur. SNAMS 2019*, pp. 466–471, 2019, doi: 10.1109/SNAMS.2019.8931839.

[12] A. H. Om, "Deep Learning Framework based on Word2Vec and CNN for Users Interests Classification," 2017.

[13] E. Fehn Unsvåg and B. Gambäck, "The Effects of User Features on Twitter Hate Speech Detection," no. 2012, pp. 75–85, 2019, doi: 10.18653/v1/w18-5110.

[14] B. P. Putra, B. Irawan, C. Setianingsih, F. T. Elektro, U. Telkom, and D. Learning, "Convolutional Neural Network Pada Gambar Hatespeech Detection Using Convolutional Neural Network Algorithm Based on Image," no. 3, 2019.

[15] N. Amruthmath, "Why balancing your data set is important?," 2021. https://www.iamnagdev.com/?p=863.

[16] J. Patihullah and E. Winarko, "Hate Speech

Detection for Indonesia Tweets Using Word Embedding And Gated Recurrent Unit," *IJCCS (Indonesian J. Comput. Cybern. Syst.*, vol. 13, no. 1, p. 43, 2019, doi: 10.22146/ijccs.40125.

[17] S. Symeonidis, D. Effrosynidis, and A. Arampatzis, "A comparative evaluation of pre-processing techniques and their interactions for twitter sentiment analysis," *Expert Syst. Appl.*, vol. 110, pp. 298–310, 2018, doi: 10.1016/j.eswa.2018.06.022.

[18] A. M. Ramadhani and H. S. Goo, "Twitter sentiment analysis using deep learning models," *2020 IEEE 17th India Counc. Int. Conf. INDICON 2020*, 2020, doi: 10.1109/INDICON49873.2020.9342279.

[19] A. Amalanathan and P. Nikil, "Data Preprocessing In Sentiment Analysis Using Twitter Data," *Int. Educ. Appl. Res. J.*, vol. 03, no. July, pp. 89–92, 2019.

[20] S. Sivakumar, L. S. Videla, T. Rajesh Kumar, J. Nagaraj, S. Itnal, and D. Haritha, "Review on Word2Vec Word Embedding Neural Net," *Proc. - Int. Conf. Smart Electron. Commun. ICOSEC 2020*, no. Icosec, pp. 282–290, 2020, doi: 10.1109/ICOSEC49089.2020.9215319.

[21] M. Wen, Y. Chen, H. Wu, S. Hou, and Y. Yang, "Automatic Classification of Government Texts Based on Improved CNN and Skip-gram Models," in *2021 2nd International Conference on Information Science and Education (ICISE-IE)*, Nov. 2021, pp. 578–583, doi: 10.1109/ICISE-IE53922.2021.00138.

[22] T. Shoryu, L. Wang, and R. Ma, "A Deep Neural Network Approach using Convolutional Network and Long Short Term Memory for Text Sentiment Classification," *Proc. 2021 IEEE 24th Int. Conf. Comput. Support. Coop. Work Des. CSCWD 2021*, pp. 763–768, 2021, doi: 10.1109/CSCWD49262.2021.9437871.

[23] N. M. Aszemi and P. D. D. Dominic, "Hyperparameter optimization in convolutional neural network using genetic algorithms," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 6, pp. 269–278, 2019, doi: 10.14569/ijacsa.2019.0100638.

[24] N. L. Beebe, L. A. Maddox, L. Liu, and M. Sun, "Sceadan: Using Concatenated N-Gram Vectors for Improved File and Data Type Classification," no. c, 2013.

[25] S. Y. Sen and N. Ozkurt, "Convolutional Neural Network Hyperparameter Tuning with Adam Optimizer for ECG Classification," *Proc. - 2020 Innov. Intell. Syst. Appl. Conf. ASYU 2020*, no. 978, 2020, doi: 10.1109/ASYU50717.2020.9259896.