Deteksi Bunuh Diri pada Media Sosial Twitter Menggunakan Metode CNN-LSTM dengan Ekspansi Fitur Word2vec

1st Fathin Thariq Wiyono
Informatika
Telkom University
Bandung, Indonesia
fathinthariq@student.telkomuniversity.ac.id

2nd Erwin Budi Setiawan
Informatika
Telkom University
Bandung, Indonesia
erwinbudisetiawan@telkomuniversity.ac.id

Abstrak* — Deteksi bunuh diri melalui media sosial telah menjadi tantangan besar dalam beberapa tahun terakhir, terutama pada platform seperti Twitter yang berisi unggahan singkat dan emosional. Penelitian ini bertujuan untuk mengembangkan model deep learning hybrid yang dapat mendeteksi potensi bunuh diri pada tweet di Twitter, menggunakan metode CNN-LSTM dan fitur semantik yang diperluas dengan Word2Vec. Dengan meningkatnya angka bunuh diri di kalangan generasi muda, yang membutuhkan sistem deteksi dini berbasis teknologi canggih. Deteksi dini ini dapat membantu memberikan intervensi lebih cepat bagi individu yang berisiko tinggi. Pendekatan yang diusulkan menggunakan kombinasi Convolutional Neural Network (CNN) untuk menangkap pola lokal dalam teks, Long Short-Term Memory (LSTM) untuk memahami urutan kata dalam teks, serta Word2Vec untuk memperkaya representasi semantik kata-kata dalam tweet. Sistem ini memanfaatkan ekstraksi fitur TF-IDF dan ekspansi fitur menggunakan Word2Vec untuk meningkatkan kemampuan model dalam mengenali pola emosional dan semantik yang ada dalam tweet. Hasil eksperimen menunjukkan bahwa model hybridCNN-LSTM dengan ekspansi fitur Word2Vec dan optimasi menghasilkan akurasi sebesar 91,31%. Hasil model hybrid CNN-LSTM belum mununjukkan hasil yang lebih baik dari model non-hybrid. Kontribusi utama dari penelitian ini adalah mengeksplorasi pengaruh ekspansi fitur Word2vec pada model hybrid deep learning untuk deteksi bunuh diri dan mengintegrasikan ekstraksi fitur TF-IDF sertaoptimasi untuk meningkatkan performa klasifikasi teks.

Kata kunci— deteksi bunuh diri, hybrid deep learning, word2vec, TF-IDF, optimasi

I. PENDAHULUAN

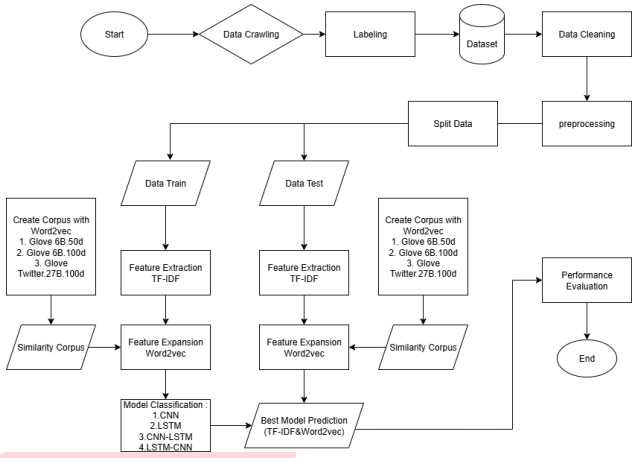
Bunuh diri menelan korban hampir 800.000 jiwa setiap tahunnya dan menjadi penyebab kematian tertinggi kedua di kalangan generasi muda. Tingkat bunuh diri global mencapai 10,5 per 100.000 penduduk, dan diprediksi bahwa pada tahun 2020, satu orang akan meninggal akibat bunuh diri setiap 20 detik. Sebagian besar (sekitar 79%) kasus bunuh diri terjadi di negara-negara berpenghasilan rendah dan menengah, di mana akses terhadap sumber daya untuk mendeteksi dan menangani masalah ini masih sangat terbatas dan kurang memadai [1]. Deteksi dini bunuh diri memegang peranan penting dalam upaya pencegahan, terutama di kalangan individu yang berisiko tinggi [2]. Proses deteksi ini melibatkan identifikasi tanda- tanda peringatan seperti perubahan perilaku drastis, perasaan putus asa, pernyataan ingin mengakhiri hidup, atau kecenderungan menyakiti diri sendiri [3]. Program deteksi dini di berbagai negara fokus pada pelatihan tenaga kesehatan, pendidik, dan komunitas untuk mengenali gejala awal dan memberikan intervensi yang tepat. Selain itu, teknologi seperti aplikasi kesehatan mental dan layanan bantuan daring turut membantu memantau kondisi psikologis seseorang. Dengan deteksi dini yang efektif, individu yang memiliki kecenderungan bunuh diri dapat segera menerima dukungan, konseling, atau perawatan yang dibutuhkan untuk mencegah terjadinya tindakan fatal [1].Penelitian ini menggunakan pendekatan hybrid deep learning CNN-LSTM untuk model yang digunakan adalah penerapan Convolutional Neural Network (CNN) dan Long Short-Term Memory (LSTM) dengan ekspansi fitur Word2Ve [4]. Metode ini efektif untuk mengolah data teks dari media sosial, seperti Twitter, yang sering kali berisi ungkapan singkat dan emosional. Word2Vec merepresentasikan kata sebagai vektor numerik berdasarkan konteksnya, memungkinkan model mengenali hubungan antar kata, termasuk sinonim dan ekspresi emosional. CNN membantu mengidentifikasi pola lokal seperti frasa dengan makna negatif atau emosional [5], sementara LSTM unggul dalam memahami urutan kata dan konteks yang lebih luas, termasuk emosi tersembunyi dalam struktur kalimat kompleks [6]. Dengan kombinasi CNN dan LSTM serta dukungan Word2Vec, sistem ini mampu mendeteksi tanda- tanda ide bunuh diri secara lebih akurat dalam mendeteksi ide bunuh diri[7].Meskipun teknologi seperti CNN, LSTM, dan Word2Vec menjanjikan deteksi dini ide bunuh diri, keterbatasan dataset dan bias anotasi manual masih menjadi kendala. Ukuran dataset yang kecil membatasikemampuan model mengenali berbagai variasi ekspresi emosional, sementara bias dari anotator dapat menyebabkan model salah menafsirkan konteks. Akibatnya, akurasi deteksi menjadi tidak konsisten, mengurangi efektivitas sistem ini dalam penerapan nyata [4].

II. KAJIAN TEORI

Dalam kajian teori ini, terdapat beberapa penelitian yang relevan dengan masalah deteksi dini niat bunuh diri di media sosial. Penelitian pertama oleh Tadesse et al. (2020) mengidentifikasi tantangan dalam mendeteksi ide bunuh diri akibat variasi bahasa dan konteks sosial media. Mereka

mengembangkan metode berbasis deep learning menggunakan CNN dan LSTM untuk meningkatkan akurasi deteksi. Kelebihan dari penelitian ini adalah penggabungan kekuatan CNN untuk ekstraksi fitur dan LSTM untuk menangkap ketergantungan jangka panjang, meskipun terdapat kekurangan terkait ukuran dataset yang terbatas [8].Selanjutnya, Tam et al. (2022) mengevaluasi efektivitas model ANN, termasuk CNN, LSTM, dan BERT, dalam menganalisis konten Twitter yang berisiko bunuh diri. Penelitian ini menunjukkan bahwa BERT memiliki kinerja terbaik dalam hal presisi dan stabilitas. Namun, penelitian ini juga memilikiketerbatasan, seperti dataset yang hanya mencakup tweet berbahasa Inggris [11].Penelitian oleh Lin et al. (2024) mengusulkan model RoBERTa-CNN yang menggabungkan kekuatan RoBERTa dalam memahami konteks dan CNN dalam mengekstrak fitur lokal. Hasil penelitian ini menunjukkan akurasi tinggi (98%) dan AUC tinggi (97,5%), tetapi memerlukan daya komputasi yang besar dan terbatas pada data teks [12].Penelitian mengenai deteksi ide bunuh diri telah banyak dilakukan dengan pendekatan teknologi kecerdasan buatan dan pembelajaran mesin. Ji et al. (2021) melakukan tinjauan menyeluruh terhadap metode deteksi ide bunuh diri menggunakan analisis fitur dan teknik pembelajaran mendalam seperti Convolutional Neural Networks (CNN) dan Long Short-Term Memory (LSTM) pada konten dari media sosial seperti Twitter. Studi tersebut menekankan efektivitas penggunaan word embedding seperti Word2Vec untuk menangkap hubungan antar kata dalam teks singkat dan emosional. Namun, keterbatasan dataset menjadi tantangan signifikan [13] .Aldhyani et al. (2022) melakukan studi mengenai deteksi ide bunuh diri dengan menggunakan metode deep learning dan machine learning, khususnya model hybrid CNN-BiLSTM dan algoritma XGBoost, pada data Reddit yang berisi postingan dari platform SuicideWatch. Penelitian ini menekankan pentingnya penggunaan Word2Vec untuk merepresentasikan kata dalam bentuk vektor, yang memungkinkan model mengenali hubungan antar kata dalam teks singkat dan emosional. Namun, keterbatasan dataset masih menjadi tantangan utama, terutama dalam hal akurasi model dan kemampuannya untuk mengenali variasi bahasa yang lebih beragam di media social [14].Fokus utama dari penelitian ini adalah untuk mengembangkan dan mengevaluasi metode deteksi dini ide bunuh diri pada media sosial Twitter dengan menggabungkan model CNN-LSTM dan ekspansi fitur menggunakan Word2Vec. Penelitian ini bertujuan untuk meningkatkan akurasi dan sensitivitas deteksi ide bunuh diri dengan memperkuat representasi semantik kata-kata dalam tweet, yang sering kali bersifat singkat dan emosional. Dengan menggunakan Word2Vec, penelitian ini berharap dapat memperkaya representasi teks, memungkinkan model untuk menangkap hubungan semantik antar kata secara lebih efektif, dan meningkatkan kemampuan model dalam memahami konteks yang kompleks.

III. METODE



Gambar 1 Sistem Deteksi Bunuh Diri

Langkah-langkah sistem deteksi bunuh diri meliputi data *crawling*, data labelling, pre-processing data, ekstraksi fitur TF-IDF, ekspansi fitur Word2vec, dan splitting data menjadi data uji dan data latih. Serta klasifikasi data dengan empat model: CNN, LSTM, CNN-LSTM, CNN-LSTM. Terakhir, kinerja sistem yang dibangun akan dievaluasi.

A. Pengumpulan Data

Mengumpulkan data melalui proses crawling merupakan salah satu cara untuk mengambil informasi dari berbagai sumber digital seperti media sosial. Dalam penelitian ini, data diambil dari platform Twitter yang menggunakan bahasa Indonesia dengan bantuan API resmi dari Twitter yang mempermudah proses pengambilan data [15]. Pada table 1, proses crawling difokuskan pada tweet yang berpotensi mengandung indikasi potensi bunuh diri, seperti ungkapan putus asa, keinginan mengakhiri hidup, atau perasaan tidak berharga. Semua perilaku ini menjadi fokus dalam proses identifikasi bunuh diri. Proses pengumplan data menghasilkan total 29,923 data, di mana 13.200 di antaranya dilabeli sebagai bunuh diri dan 16.723 sebagai non-bunuh diri [16].

Tabel 1 Kata Kunci

Kata Kunci	Total
die	12895
suicide	7160
hopeless	9868
Total Data	**29923**

B. Pelabelan Data

Pelabelan data merupakan tahap berikutnya setelah proses pengumpulan data selesai. Langkah ini bertujuan untuk membantu model dalam membedakan antara data yang mengandung indikasi keinginan bunuh diri atau tidak bunuh diri [17]. Setiap label melibatkan 3 annotator dengan prinsip *majority vote* dan sebelumnya dilakukan kesamaan persepsi terkait pengertian bunuh diri dan ciri cirnya. Pada tabel 2, proses pelabelan dilakukan dengan menggunakan format biner, di mana data yang memiliki indikasi keinginan bunuh diri diberi label "1," sedangkan data tanpa indikasi tersebut diberi label "0.".

Tabel 2 Pelabelan data

Tweet	Label
This takes me from down to hopeless. Art is an embarrassment. i dont wanna do this anymore, i want to die	1
Mouth open smile as your profile pic biggest beta flag possible disregard my previous comment you are hopeless	0
WTH is going to happen to us? I m a fighter and an optimist but I m feeling pretty helpless and hopeless right now.	0

Setelah dilakukan proses pelabelan, jumlah kelas indikasi keinginan bunuh diri dan tanpa indikasi keinginan bunuh diri setelah di labelling dapat dilihat pada Tabel 3.

Tabel 3 Labeling

Label	Total
1	13200
0	16723
Total	**29,923**
Label	**Total**

C. Pre-Processing Data

Data yang diperoleh dari media sosial seperti Twitter sering kali merupakan data mentah yang belum terstruktur dan dipenuhi dengan gangguan, seperti simbol, angka, atau karakter yang tidak relevan. Oleh karena itu, langkah pertama dalam pre-processing adalah cleaning, yang bertujuan untuk menghilangkan elemen-elemen mengganggu tersebut, termasuk tagar (#), nama pengguna (@), URL, dan emoticon yang sering muncul dalam tweet. Setelah data dibersihkan, tahap berikutnya adalah case folding, yang mengubah semua teks menjadi huruf kecil untuk memastikan konsistensi dalam analisis dan menghindari perbedaan arti akibat perbedaan kapitalisasi. Selanjutnya, tokenization dilakukan untuk memecah teks menjadi unit-unit terkecil, yaitu kata-kata atau token yang lebih mudah dianalisis [18].

Setelah data di-tokenized, tahap berikutnya adalah change word, yang mengganti kata-kata dengan bentuk baku atau dasar, sehingga variasi kata seperti plural atau konjugasi dapat diseragamkan. Melalui serangkaian tahapan pre-processing ini, teks dari media sosial yang sebelumnya tidak terstruktur dan penuh gangguan dapat diubah menjadi format yang lebih bersih dan terstruktur [19]. Setelah proses selesai, dataset akhir pada tanpa indikasi keinginan bunuh diri menjadi 15,613 dan pada indikasi keinginan bunuh diri menjadi 12,020. Dapat dilihat pada Tabel 4.

Tabel 4 Data setelah Preprocessing

Label	Total
1	12,020
0	15,613
Total	**27,633**
Label	**Total**

D. Feature Extraction TF-IDF

Ekstraksi fitur merupakan langkah untuk menghitung bobot setiap kata dalam teks dan mengubah kata- kata tersebut menjadi representasi vektor digital. Proses ini adalah tahap pertama dalam klasifikasi teks. Dalam penelitian ini, metode yang digunakan untuk ekstraksi fitur adalah TF-IDF. Term Frequency-Inverse Document Frequency (TF-IDF) adalah teknik yang digunakan untuk memberikan vektor pada kata-kata (term) dalam sebuah dokumen. Term Frequency (TF) mengukur frekuensi kemunculan suatu kata dalam dokumen, sedangkan Inverse Document Frequency (IDF) menghitung logaritma dari kebalikan proporsi dokumen yang mengandung kata tersebut dalam seluruh korpus. Dengan menggabungkan keduanya, TF-IDF memberikan cara untuk menilai sejauh mana pentingnya suatu kata dalam dokumen tertentu dibandingkan dengan dokumen-dokumen lainnya dalam korpus[20]. Berikut adalah langkah untuk menghitung vektor dalam metode TF-IDF.

$$tf_t = 1 + \log(tf_t) \quad (1)$$

$$idf_t = \log \frac{D}{df_t} \quad (2)$$

$$W_{t,d} = tf_t \times idf_t \quad (3)$$

Bobot dokumen ke-i terhadap kata ke-j ($W_{t,d}$) menunjukkan pentingnya kata ke-j dalam dokumen ke-i. Nilai *Term Frequency* (TF_t) mencerminkan frekuensi kemunculan kata ke-t dalam dokumen, semakin sering muncul semakin tinggi nilainya, dihitung dengan rumus $1 + \log(tf_t)$. Sebaliknya, *Inverse Document* (IDF_t) menunjukkan kelangkaan kata ke-t di seluruh dokumen dalam korpus, dihitung dengan rumus $\log(\frac{D}{df_t})$, di mana D adalah total jumlah dokumen dan (DF_t) adalah jumlah dokumen yang mengandung kata ke-t. Semakin jarang kata muncul di banyak dokumen, semakin tinggi nilai IDF-nya.

Penelitian ini menggunakan *TfidfVectorizer* untuk ekstraksi fitur teks. *TfidfVectorizer* memanfaatkan parameter N-gram_range untuk menentukan rentang N-gram yang akan digunakan sebagai fitur dalam proses ekstraksi. N-gram, yang merujuk pada urutan n-kata dalam teks, memungkinkan model untuk menangkap pola kata dengan lebih tepat. Dalam penelitian ini, diterapkan lima jenis n-gram, yaitu Unigram (satu kata), Bigram (dua kata), Trigram (tiga kata), Uni-Bigram (kombinasi satu dan dua kata), dan Uni-Trigram (kombinasi satu hingga tiga kata). Tabel 5 menunjukkan contoh penggunaan parameter N-gram pada TF-IDF untuk mengekstraksi kata “die, suicide, hopeless”.

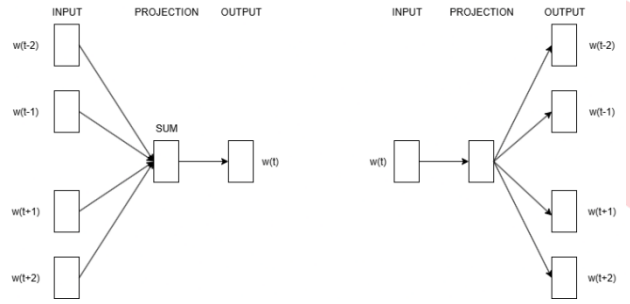
Tabel 5 Contoh TF-IDF Ngram

N-Gram	*Tweet*
Unigram	[die], [suicide], [hopeless]
Bigram	[die, suicide], [suicide, hopeless]
Trigram	[die, suicide, hopeless]
Unigram + Bigram	[die], [suicide], [hopeless], [die, suicide], [suicide, hopeless]
Unigram + Trigram	[die], [suicide], [hopeless], [die, suicide, hopeless]

E. Feature Expansion Word2vec

Penelitian ini menggunakan metode word embedding yaitu word2vec sebagai word embedding. Word2vec merupakan

model klasifikasi teks yang dikembangkan oleh Google. Word2vec digunakan dalam penelitian ini untuk mengukur tingkat kesamaan antar kata, memanfaatkan pustaka representasi kata yang memungkinkan pembuatan vektor kata. Metode ini mengubah teks menjadi vektor dan mempresentasikan setiap kata dalam ruang vektor, sehingga dapat menangani kata-kata yang tidak ada dalam korpus dengan cara memanfaatkan representasi kata dari kata-kata yang ada di sekitar kata tersebut. Tujuan utama menggunakan Word2vec adalah mengurangi ketidaksesuaian kosakata dengan menggunakan "word embeddings" [21]. Penelitian ini menggunakan dua jenis korpus yaitu Twitter dan Wikipedia dari korpus Glove. Korpus dibuat dengan memanfaatkan data dari berbagai sumber untuk memastikan representasi yang lebih luas dan beragam. Gambar 2 menunjukkan bagaimana penggunaan word2vec.

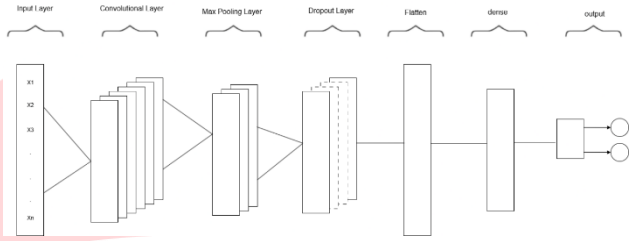


Gambar 2 Feature Expansion Word2vec

Ekspansi fitur dengan Word2Vec dalam penelitian ini bertujuan untuk memperkaya representasi semantik teks dengan memetakan kata-kata ke dalam vektor berdimensi tinggi berdasarkan konteks penggunaannya, yang memungkinkan model untuk menangkap hubungan semantik antar kata, seperti sinonim dan ekspresi emosional. Digabungkan dengan TF-IDF, yang mengukur frekuensi kata, Word2Vec membantu model dalam memahami makna implisit dan emosional dalam *tweet* singkat dan ambigu di media sosial, seperti Twitter. Dengan demikian, kombinasi ini meningkatkan kemampuan model *hybrid* CNN-LSTM dalam mendeteksi potensi bunuh diri dengan akurasi yang lebih tinggi, karena model dapat menangkap pola emosional dan konteks yang lebih dalam.

F. Model Klasifikasi

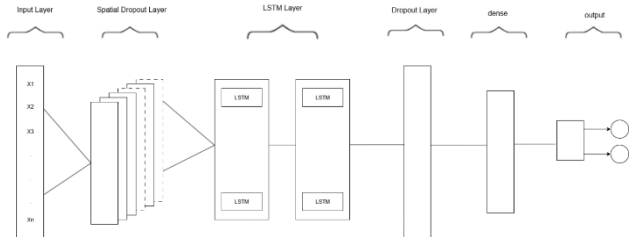
Setelah data melewati tahapan pre-processing dan dipresentasikan dalam bentuk vektor, data akan diklasifikasikan menggunakan beberapa algoritma klasifikasi, seperti Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), model *hybrid deep learning* CNN-LSTM, dan model *hybrid deep learning* LSTM-CNN. Convolutional Neural Network (CNN) termasuk dalam salah satu jaringan yang banyak digunakan dalam bidang pemrosesan bahasa alami (NLP), khususnya untuk tugas-tugas yang melibatkan data teks. CNN memanfaatkan lapisan konvolusi untuk mengekstrak vektor dari data input, yang melibatkan penggunaan filter atau kernel yang bergerak di atas data input untuk mendeteksi vektor fitur lokal. Arsitektur CNN ini terdiri dari beberapa layer, antara lain input, convolutional layer, pooling layer, dropout layer, flattening, fully connected layer, dan output [1].Arsitektur CNN dimulai dengan lapisan input yang menerima data dalam format vektor. Lapisan Conv1D dengan 32 filter, kernel 5, dan fungsi aktivasi ReLU digunakan untuk mengekstrak fitur teks yang penting. Lapisan MaxPooling1D berukuran 5 kemudian melakukan down-sampling. Untuk mencegah overfitting, lapisan Dropout berukuran 0.3 digunakan. Setelah output konvolusi dan pooling diratakan dengan flattening, dua lapisan densitas digunakan untuk memprosesnya. Lapisan pertama memiliki 32 neuron dan ReLU, dan lapisan kedua memiliki 2 neuron dan sigmoid untuk klasifikasi biner. Model ini menggunakan loss function binary crossentropy dan Adam *optimizer*, dengan EarlyStopping digunakan untuk mencegah overfitting. Arsitektur CNN dapat dilihat pada Gambar 3.



Gambar 3 CNN Architecture

Long Short-Term Memory (LSTM) adalah salah satu tipe Recurrent Neural Network (RNN) yang dirancang untuk mengatasi ketergantungan jangka panjang dalam data berurutan. LSTM menggantikan neuron standar pada RNN dengan sel LSTM. Arsitektur LSTM terdiri dari tiga gerbang utama, yaitu Forget Gate, Input Gate, dan Output Gate. Forget Gate berfungsi untuk menentukan seberapa banyak informasi lama yang harus dihapus dari memori sel, sementara Input Gate mengatur seberapa banyak informasi baru yang perlu disimpan. Output Gate bertugas untuk memilih informasi yang akan diambil dari memori sel untuk menghasilkan output pada waktu tertentu [2].

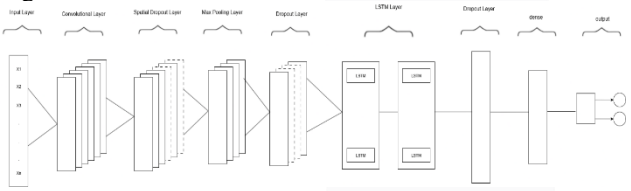
Model ini menggunakan lapisan LSTM dengan 128 neuron pada tahap pertama dan 64 neuron pada tahap kedua untuk menangkap hubungan jangka panjang dalam data sekuensial. Masing-masing lapisan LSTM diikuti oleh Dropout dengan rate 0.2 untuk mengurangi overfitting. Setelah itu, lapisan Dense dengan 32 neuron dan aktivasi ReLU digunakan untuk menangkap hubungan non-linear dalam data. Pada tahap akhir, lapisan dense dengan jumlah neuron yang sesuai dengan jumlah kelas (numclass) dan menggunakan aktivasi *softmax* diterapkan untuk klasifikasi multi-kelas. Model ini dikompilasi menggunakan Adam *optimizer* dengan learning rate 0.001 dan categorical_crossentropy sebagai *loss function*. Jika val_loss tidak menunjukkan perbaikan, EarlyStopping akan diterapkan untuk menghentikan pelatihan lebih awal. Arsitektur LSTM dapat dilihat pada Gambar 4.



Gambar 4 LSTM Architecture

Model *deep learning hybrid* diterapkan dengan menggabungkan dua arsitektur model, yaitu CNN (Convolutional Neural Network) dan LSTM (Long Short-Term Memory). Keunggulan utama dari CNN-LSTM adalah kemampuannya dalam menggabungkan CNN untuk mengekstraksi fitur lokal teks, seperti pola huruf, kata, dan frasa. Sementara itu, LSTM memiliki kemampuan untuk memahami konteks dan informasi urutan dalam data. Kombinasi kedua model ini menghasilkan representasi teks yang lebih kaya dan lebih informatif, yang memungkinkan model untuk lebih efektif menangkap ketergantungan antara kata-kata dan kalimat [3].

Model *hybrid* CNN-LSTM ini terdiri dari beberapa lapisan yang dimulai dengan Conv1D dengan 128 filter dan kernel size 3, serta menggunakan aktivasi ReLU dan padding same. Setelah itu, lapisan MaxPooling1D digunakan dengan pool size 2 dan strides 2 untuk mengurangi dimensi fitur. Model ini kemudian memiliki Conv1D tambahan dengan 128 filter dan kernel size 4, diikuti oleh lapisan MaxPooling1D dengan pool size 2 dan strides 2. Untuk mencegah overfitting, lapisan SpatialDropout1D dengan rate 0.2 dan Dropout dengan rate 0.2 ditambahkan. Kemudian, model dilanjutkan dengan dua lapisan LSTM, masing-masing dengan 128 unit pada lapisan pertama dan 64 unit pada lapisan kedua, dengan pengaturan return_sequences yang sesuai. Lapisan Dense dengan 32 unit dan aktivasi ReLU diikuti oleh lapisan output Dense dengan jumlah neuron sesuai jumlah kelas (numclass) dan aktivasi softmax. Gambar 5 menunjukkan arsitektur model *hybrid* yang digunakan untuk mendeteksi Bunuh Diri di Twitter.



Gambar 5 Hybrid deep learning CNN- LSTM Architecture

G. Evaluasi Performansi

Dalam penelitian ini menggunakan confusion matrix, yang merupakan alat penting untuk menilai efektivitas model klasifikasi. Confusion matrix dapat memberikan gambaran seberapa baik model membedakan antar kelas dengan membandingkan antara hasil klasifikasi yang sebenarnya dan prediksi yang dihasilkan oleh model. *Confusion matrix* terdiri dari empat komponen hasil klasifikasi, yaitu *True Positive* (TP) yang menunjukkan data positif yang terprediksi dengan benar, *False Positive* (FP) yang menunjukkan data negatif yang salah diprediksi sebagai positif, *True Negative* (TN) yang menunjukkan data negatif yang terprediksi dengan benar, dan *False Negative* (FN) yang menunjukkan data positif yang salah diprediksi sebagai negatif. Nilai-nilai yang dihasilkan dari confusion matrix digunakan untuk mengevaluasi kinerja model klasifikasi melalui perhitungan Accuracy dan F1-score, yang dapat diberi bobot [4]. Keempat metrik tersebut dapat dihitung menggunakan rumus berikut.

$$Accuracy = \frac{(TP + TN)}{(TP + FP + FN + TN)} \quad (4)$$

$$F1 - score = 2 \times \frac{Recall \times Presicion}{Recall \times Presicion} \quad (5)$$

IV. HASIL DAN PEMBAHASAN

Pada penelitian ini akan dilakukan beberapa skenario pengujian terhadap empat model klasifikasi yaitu CNN, LSTM, *hybrid deep learning* CNN-LSTM dan *hybrid deep learning* LSTM- CNN. Skenario pertama dilakukan untuk menentukan model CNN dan LSTM terbaik. Skenario kedua menerapkan Max-Feature pada *baseline* dari skenario pertama dengan *baseline* digunakan untuk ekstraksi fitur menggunakan TF-IDF. Eksperimen dilakukan dengan membandingkan hasil akurasi pada jumlah maksimum fitur TF-IDF sebesar 5000, 10000, 15000, 20000 dan x_train.shape. Skenario ketiga dilakukan dengan menggunakan TF-IDF untuk pembobotan kata dengan N-gram pada hasil terbaik skenario kedua. Skenario keempat menguji model *baseline* CNN dan LSTM yang diperoleh dari skenario ketiga, kemudian dikombinasikan dengan ekspansi fitur menggunakan Word2vec. Skenario kelima menguji efektivitas dari tiga *optimizer* populer yang sering digunakan dalam training model *deep learning*, yaitu Adam, Nadam, dan Adamax.

A. Skenario Pertama

Skenario pertama adalah menerapkan TF-IDF pada ekstraksi fitur untuk menentukan model *baseline* terbaik menggunakan unigram. Parameter CNN yang digunakan adalah filter 128, kernel size 3 dan 4, ukuran batch 32, dan epoch 10. LSTM menggunakan parameter unit 128, dropout 0.2, learning rate 0.0001, dan epoch 10. Penelitian ini menggunakan rasio pembagian data 70:30, 80:20, dan 90:10. Rasio pembagian data 70:30 berarti 70% untuk data penelitian dan 30 % untuk data pengujian. Tabel 6 menunjukkan hasil pengujian skenario pertama masing-masing model. Untuk model CNN mendapatkan akurasi 89,10%, untuk model LSTM mendapatkan akurasi 89,11%, untuk model *hybrid* CNN-LSTM mendapatkan akurasi 87,39% dan untuk model *hybrid* LSTM-CNN mendapatkan akurasi 89,39%.

Tabel 6 Hasil Uji Skenario Pertama

Splitting Ratio	CNN		LSTM		CNN-LSTM		LSTM-CNN	
	Acc%	**F1%**	**Acc%**	**F1%**	**Acc%**	**F1%**	**Acc%**	**F1%**
70:30	87,80	87,99	87,62	88,65	**87,39**	87,14	88,02	87,87
80:20	88,14	87,67	87,57	87,78	87,38	**87,30**	88,69	87,03
90:10	**89,10**	**88,63**	**89,11**	**89,06**	87,22	86,71	**89,39**	**89,49**

B. Skenario Kedua

Skenario kedua adalah menerapkan hasil terbaik dari *baseline* dengan vektor fitur 5000, 10000, 15000, 20000 dan x_train.shape. Tabel 7 menunjukkan hasil akurasi terbaik menggunakan vektor fitur x_train.shape dengan model CNN 91,35% dan LSTM 91,27%, untuk model *hybrid* CNN-LSTM dan *hybrid* LSTM-CNN menunjukkan hasil akurasi

terbaik menggunakan vektor fitur 20,000 masing masing model yaitu *hybrid* CNN-LSTM 90,26% dan *hybrid* LSTM-CNN 91,16%. Dengan hasil tersebut akan digunakan fitur terbaik untuk skenario pengujian selanjutnya.

Tabel 7 Hasil Uji Skenario Kedua

Max Feature	CNN		LSTM		CNN-LSTM		LSTM-CNN	
	Acc%	F1%	Acc%	F1%	Acc%	F1%	Acc%	F1%
5000	90,76	90,48	90,83	90,79	89,70	89,14	90,88	90,72
10000	91,08	90,78	91,01	90,81	89,86	89,58	91,15	90,68
15000	91,24	91,09	91,09	90,94	90,15	89,43	91,06	91,04
20000	91,19	91,24	91,09	91,10	**90,26**	**89,95**	**91,16**	**90,77**
x_train.shape	**91,35**	**91,25**	**91,27**	**91,40**	90,23	89,90	90,83	90,55

C. Skenario Ketiga

Skenario ketiga adalah menerapkan beberapa parameter N-Gram dalam ekstraksi fitur TF-IDF untuk pembobotan kata. Skenario ini menghasilkan akurasi yang akan dibandingkan untuk menentukan hasil terbaik dari penggunaan beberapa kombinasi parameter N-Gram. Kombinasi yang digunakan pada penelitian ini adalah Unigram, Bigram, Trigram, Unigram-Bigram, dan Unigram-Trigram. Tabel 8 menunjukkan hasil skenario ketiga yang memiliki akurasi terbaik pada kombinasi TF-IDF dan Unigram-Bigram pada model CNN dengan akurasi 79,65%,model LSTM dengan akurasi 79,46%, model *hybrid* CNN-LSTM dengan akurasi 90,45% dan model *hybrid* LSTM-CNN dengan akurasi 91,14%. Hasil tersebut akan digunakan model TF-IDF Unigram-Trigram pada skenario pengujian selanjutnya.

Tabel 8 Hasil Uji Skenario Ketiga

N-Gram	CNN		LSTM		CNN-LSTM		LSTM-CNN	
	Acc%	F1%	Acc%	F1%	Acc%	F1%	Acc%	F1%
Unigram	89,87	89,34	88,95	88,89	89,59	89,43	89,47	89,23
Bigram	88,08	88,01	87,41	87,32	87,81	87,68	87,54	87,42
Trigram	79,14	79,05	79,03	79,01	78,85	78,45	78,86	78,57
Uni-Bigram	**91,42**	**91,36**	**91,32**	**91,21**	**90,45**	**90,24**	**91,14**	**91,03**
Bi-Trigram	87,93	87,93	87,26	87,14	87,83	87,48	87,49	87,33
Uni-Bi-Trigram	90,81	90,81	90,63	90,43	91,00	89,89	90,80	90,38

D. Skenario Keempat

Pada Skenario Keempat ini, kami menguji berbagai kombinasi model *deep learning* dengan menggunakan embedding Word2Vec dan korpus GloVe (dengan dimensi vektor 100d dan 50d) pada arsitektur CNN, LSTM, *hybrid* CNN-LSTM dan *hybrid* LSTM-CNN. Tujuan utama dari eksperimen ini adalah untuk membandingkan kinerja berbagai model dalam klasifikasi teks, dengan mengukur akurasi pada metrik Top 1, Top 2, Top 3, Top 5, dan Top 10. Tabel 9 menunjukkan akurasi terbaik pada model CNN, LSTM, *hybrid* CNN-LSTM dan *hybrid* LSTM-CNN. Pada model CNN glove.6B.100d dengan 92,23% Top 2, glove.6B.50d dengan 91,81% Top 5, glove.twitter.27B.100d dengan 91,22% Top 10. Pada model LSTM glove.6B.100d dengan 91,49% Top 2, glove.6B.50d dengan 91,78. Pada model *hybrid* CNN-LSTM glove.6B.100d dengan 91,22% Top 3, glove.6B.50d dengan 91,00% Top 3, glove.twitter.27B.100d dengan 90,40% Top 10, Pada model *hybrid* LSTM-CNN glove.6B.100d dengan 91,64% Top 3, glove.6B.50d dengan 91,80% Top 5, glove.twitter.27B.100d dengan 90,79% Top 5.

Tabel 9 Hasil Uji Skenario Keempat

Model	Rank	glove.6B.100d		glove.6B.50d		glove.twitter.27B.100d	
		Acc%	F1%	Acc%	F1%	Acc%	F1%
CNN	Top 1	91,18	91,04	90,94	90,89	90,95	90,87
	Top 2	**92,23**	**92,15**	90,70	90,79	90,52	90,43
	Top 3	91,83	91,57	91,63	91,54	90,15	90,04
	Top 5	91,95	91,78	**91,81**	**91,76**	90,36	90,23
	Top 10	91,48	91,32	90,43	90,34	**91,22**	**91,14**
LSTM	Top 1	91,41	91,23	90,86	90,76	90,34	90,22
	Top 2	**91,49**	**91,21**	90,50	90,37	90,40	90,32
	Top 3	91,39	91,24	91,53	91,24	90,42	90,36
	Top 5	91,46	91,38	**91,78**	**91,47**	90,77	90,57
	Top 10	91,46	91,10	90,43	90,22	**91,22**	**91,08**
CNN-LSTM	Top 1	89,98	89,76	90,09	89,83	89,47	89,29
	Top 2	90,57	90,39	89,73	89,68	89,42	89,28
	Top 3	**91,22**	**91,13**	**91,00**	**90,93**	89,23	89,12
	Top 5	91,08	91,00	90,68	90,59	89,36	89,18
	Top 10	90,22	90,16	89,81	89,69	**90,40**	**90,28**
LSTM-CNN	Top 1	90,79	90,66	90,38	90,25	90,53	90,32
	Top 2	90,56	90,52	90,43	90,37	90,04	89,86
	Top 3	91,33	91,24	91,08	91,01	90,34	90,21
	Top 5	**91,64**	**91,58**	**91,80**	**91,69**	**90,79**	**90,58**
	Top 10	91,23	91,11	90,74	90,57	90,40	90,29

E. Skenario Kelima

Pada skenario kelima, tujuan utama adalah untuk membandingkan kinerja dari tiga *optimizer* populer dalam *deep learning*, yaitu Adam, Nadam, dan Adamax. Ketiga *optimizer* ini memiliki karakteristik yang berbeda dalam cara mereka memperbarui bobot selama proses pelatihan, yang dapat memengaruhi kinerja model. Model yang digunakan untuk eksperimen ini adalah CNN, LSTM, *hybrid* CNN-LSTM dan *hybrid* LSTM-CNN yang diterapkan pada embedding Word2Vec dan GloVe (dengan dimensi vektor 100d dan 50d). Tabel 10 menunjukkan akurasi terbaik pada tiga *optimizer* yaitu Adam, Nadam, dan Adamax. Pada model CNN glove.6B.100d 92,28% Nadam, glove.6B.50d 91,81% Adam, glove.twitter.27B.100d 91,22% Adam. Pada model LSTM glove.6B.100d 91,49% Adam, glove.6B.50d 92,27% Nadam, glove.twitter.27B.100d 91,22% Adam. Pada model *hybrid* CNN-LSTM glove.6B.100d 91,22% Adam, glove.6B.50d 91,31% Adamax, glove.twitter.27B.100d 90,68% Nadam. Pada model *hybrid* LSTM-CNN glove.6B.100d 91,64% Adam, glove.6B.50d 91,80% Adam, glove.twitter.27B.100d 90,79% Adam.

Tabel 10 Hasil Uji Skenario Kelima

Model	**Rank**	**glove.6B.100d**		**glove.6B.50d**		**glove.twitter.27B.100d**	
		Acc%	**F1%**	**Acc%**	**F1%**	**Acc%**	**F1%**
CNN	**Adam**	92,23	92,08	**91,81**	**91,67**	**91,22**	**91,07**
	Nadam	**92,28**	**92,11**	91,76	91,45	90,76	90,59
	Adamax	92,14	92,02	91,15	89,93	89,75	89,58
LSTM	**Adam**	**91,49**	**91,28**	91,78	91,42	**91,22**	**91,09**
	Nadam	91,31	91,23	**92,27**	**92,11**	90,41	90,24
	Adamax	89,61	89,49	89,99	89,79	88,97	88,75
CNN-LSTM	**Adam**	**91,22**	**91,13**	91,00	90,83	90,40	90,23
	Nadam	91,17	91,05	91,07	90,87	**90,68**	**90,34**
	Adamax	90,82	90,78	**91,31**	**91,16**	87,71	87,54
LSTM-CNN	**Adam**	**91,64**	**91,49**	**91,80**	**91,68**	**90,79**	**90,61**
	Nadam	91,36	91,17	91,55	91,33	90,49	90,24
	Adamax	90,31	90,24	90,87	90,68	89,85	89,46

F. Analisis Hasil Pengujian

Analisis Pengujian dilakukan untuk memilih model *baseline* terbaik melalui pembagian data (splitting ratio), vektor fitur, kombinasi nilai N-gram dengan TF-IDF, serta korpus dengan peringkat teratas menggunakan ekspansi fitur Word2vec dan *optimizer* yang sudah tertera.

Berdasarkan hasil pengujian skenario pertama, model CNN, LSTM dan *hybrid* LSTM-CNN menunjukkan performa terbaik dengan akurasi tertinggi pada rasio 90:10 yaitu 89,10%, 89,11% dan 89,39%. Sedangkan CNN-LSTM memiliki akurasi pada rasio 70:30 dengan akurasi 87,39%.

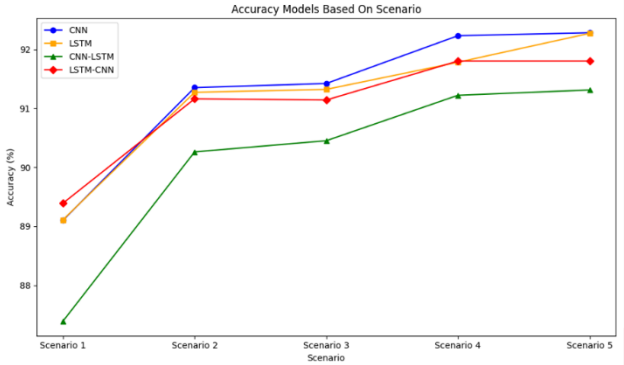
Berdasarkan hasil pengujian skenario kedua, model CNN dan LSTM menunjukkan kinerja terbaik dengan akurasi masing-masing 91,35% dan 91,27% menggunakan fitur x_train.shape. Untuk model *hybrid*, CNN-LSTM dan LSTM-CNN, akurasi tertinggi tercatat pada vektor fitur 20,000, yaitu 90,26% untuk CNN-LSTM dan 91,16% untuk LSTM-CNN. Peningkatan akurasi dibandingkan *baseline* pada scenario pertama adalah sebesar 2,25% untuk model CNN, 2,16% untuk model LSTM, 2,87% untuk model *hybrid* CNN-LSTM dan 1,77% untuk model *hybrid* LSTM-CNN.

Berdasarkan hasil pengujian skenario ketiga, menunjukkan peningkatan yang lebih signifikan dengan penggabungan TF-IDF dengan Unigram-Bigram. Penggunaan kombinasi n-gram yang lebih kaya membantu model menangkap konteks dan hubungan antar kata dalam teks. Penggunaan n-gram memperluas representasi fitur, membantu model menangani variasi bahasa dan kompleksitas data teks, sehingga meningkatkan akurasi model. kombinasi Unigram-Bigram menunjukkan performa terbaik pada model CNN dengan akurasi 91,42%, model LSTM dengan akurasi 91,32%, serta pada model *hybrid* CNN-LSTM dengan akurasi 90,45% dan *hybrid* LSTM-CNN dengan akurasi 91,14%. Peningkatan akurasi dibandingkan dengan *baseline* pada Skenario pertama adalah sebesar 2,32% untuk model CNN, 2,21% untuk model LSTM, 3,06% untuk model *hybrid* CNN-LSTM, dan 1,75% untuk model *hybrid* LSTM-CNN.

Berdasarkan hasil pengujian skenario keempat, menunjukkan peningkatan akurasi lebih lanjut dengan penggunaan embedding korpus glove.6B.100d dan glove.6B.50d pada model. Hal ini disebabkan oleh kualitas embedding yang lebih baik, yang mengandung informasi lebih kaya dan representatif, sehingga mendukung kinerja model yang lebih optimal. Model LSTM-CNN dan CNN dengan embedding glove.6B.100d secara konsisten menunjukkan hasil yang lebih tinggi, terutama pada metrik Top 2 dan Top 3, yang menandakan bahwa korpus ini memberikan informasi yang lebih relevan dan membantu model dalam meningkatkan kemampuannya untuk mengenali pola yang lebih kompleks dalam data teks. Peningkatan akurasi dibandingkan dengan *baseline* pada Skenario Pertama dan Skenario Keempat adalah sebesar 3,13% untuk model CNN, 2,38% untuk model LSTM, 3,83% untuk model *hybrid* CNN-LSTM, dan 2,25% untuk model *hybrid* LSTM-CNN.

Pada Skenario 5, model CNN-LSTM menunjukkan hasil yang solid meskipun tidak sebaik CNN dalam hal akurasi tertinggi. CNN-LSTM menggabungkan kekuatan dari CNN untuk mengekstraksi fitur lokal dan LSTM untuk memahami konteks jangka panjang dalam teks, yang sangat penting untuk tugas deteksi ide bunuh diri. Pada embedding Word2Vec dan GloVe, model CNN-LSTM dengan optimasi Adam memberikan akurasi yang kompetitif (91,22%) meskipun sedikit lebih rendah dibandingkan dengan CNN yang mencapai 92,28%. Hal ini menunjukkan bahwa meskipun CNN-LSTM tidak menghasilkan akurasi tertinggi, kemampuan untuk menangkap pola lokal melalui CNN dan hubungan urutan kata dengan LSTM memberikan kekuatan pada model ini dalam memahami nuansa emosional dan

konteks lebih dalam pada teks. Dalam skenario pengujian ini, CNN-LSTM masih merupakan pilihan yang baik terutama ketika model perlu menangani teks yang lebih kompleks dan kaya akan informasi kontekstual, meskipun untuk akurasi maksimal, model CNN tetap lebih unggul, terlihat pada gambar 6.



	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5
CNN	89.1	91.35	91.42	92.23	92.28
LSTM	89.11	91.27	91.32	91.78	92.27
CNN-LSTM	87.39	90.26	90.45	91.22	91.31
LSTM-CNN	89.39	91.16	91.14	91.8	91.8

Gambar 6 Peningkatan akurasi semua scenario

V. KESIMPULAN

Dalam penelitian ini, peneliti berhasil mengembangkan model *hybrid deep learning* untuk mendeteksi potensi bunuh diri pada *tweet* berbahasa Inggris di platform Twitter. Model ini menggabungkan metode CNN-LSTM dengan ekspansi fitur Word2Vec dan TF-IDF. Dataset yang digunakan terdiri dari 29,923 *tweet*, dengan 13.200 *tweet* berlabel bunuh diri dan 16.723 *tweet* berlabel non-bunuh diri. Dataset ini dianalisis menggunakan empat model klasifikasi, yaitu CNN, LSTM, model *hybrid* CNN-LSTM, dan model *hybrid* LSTM-CNN. Pada model CNN, LSTM, dan *hybrid* CNN-LSTM, ekstraksi fitur menggunakan TF-IDF untuk menghasilkan representasi vektor teks. Sementara itu, pada model *hybrid* CNN-LSTM, fitur Word2Vec digunakan untuk memperkaya representasi semantik kata-kata dalam *tweet*, yang memungkinkan model menangkap makna implisit dan emosional yang tidak dapat ditangkap oleh TF-IDF saja.

Model ini mengintegrasikan korpus GloVe dan Word2Vec, yang terdiri dari beberapa sumber dataset, termasuk glove.6B.100d dan glove.twitter.27B.100d, untuk menangkap pola semantik yang lebih dalam. Kombinasi ini meningkatkan kemampuan model dalam memahami konteks emosional yang lebih kompleks dalam *tweet*, yang sering kali bersifat singkat dan penuh nuansa.

Hasil pengujian pada model *hybrid* CNN-LSTM, pada skrenario 2 yang menunjukkan peningkatan akurasi paling tinggi dan ditambah ekspansi fitur Word2Vec pada skrenario 4 serta optimasi dari Adamax di skrenario 5, dengan hasil akhir akurasi 91,31%, yang menunjukkan peningkatan akurasi sebesar 2.21% dibandingkan *baseline* CNN dan 2,7% dibandingkan *baseline* LSTM. Hasil model *hybrid* CNN-LSTM belum mununjukkan hasil yang lebih baik dari model *non-hybrid*. Akurasi yang dihasilkan CNN-LSTM sudah cukup bagus sekitar 91,31%.

REFERENSI

[1] World Health Organization, “National Suicide Prevention Strategies: Progress, Examples and Indicators,” Geneva, Switzerland, 2018.

[2] M. Weishaar and A. Beck, “Hopelessness and suicide,” *International Review of Psychiatry*, vol. 4, Nov. 2009, doi: 10.3109/09540269209066315.

[3] M. A. Silver, “Relation of Depression of Attempted Suicide and Seriousness of Intent,” *Arch Gen Psychiatry*, vol. 25, no. 6, p. 573, Dec. 1971, doi: 10.1001/archpsyc.1971.01750180093015.

[4] J. Philip, “INTERNATIONAL JOURNAL FOR INNOVATIVE RESEARCH IN MULTIDISCIPLINARY FIELD CNN-LSTM Hybrid Deep Learning Model for Remaining Useful Life Estimation,” 2024, doi: 10.2015/IJIRMF/ICSETI-2024/P04.

[5] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient Estimation of Word Representations in Vector Space,” Sep. 2013, [Online]. Available: http://arxiv.org/abs/1301.3781

[6] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[7] B. R. Babu, S. Ramakrishna, and S. K. Duvvuri, “Advanced Sentiment and Trend Analysis of Twitter Data Using CNN-LSTM and Word2Vec,” in *2025 4th International Conference on Sentiment Analysis and Deep Learning (ICSADL)*, 2025, pp. 1536–1543. doi: 10.1109/ICSADL65848.2025.10933031.

[8] M. M. Tadesse, H. Lin, B. Xu, and L. Yang, “Detection of Suicide Ideation in Social Media Forums Using Deep Learning,” 2020, doi: https://doi.org/10.3390/a13010007.

[9] H. Kour and M. K. Gupta, “An hybrid deep learning approach for depression prediction from user tweets using feature-rich CNN and bi-directional LSTM,” *Multimed Tools Appl*, vol. 81, no. 17, pp. 23649–23685, Jul. 2022, doi: 10.1007/s11042-022-12648-y.

[10] X. Rong, “word2vec Parameter Learning Explained,” Nov. 2014, [Online]. Available: http://arxiv.org/abs/1411.2738

[11] Tam CC, Huan N, Cai R, Zhang J, Li z, and Li X, “Evaluation of Artificial Neural Networks in Natural Language Processing to Identify Suicide-Risk Messages on Twitter,” 2022, doi: 10.2196/preprints.42557.

[12] E. Lin, J. Sun, H. Chen, and M. H. Mahoor, “Data Quality Matters: Suicide Intention Detection on Social Media Posts Using a RoBERTa-CNN Model,” 2024. [Online]. Available: http://mohammadmahoor.com

[13] S. Ji, S. Pan, X. Li, E. Cambria, G. Long, and Z. Huang, “Suicidal Ideation Detection: A Review of Machine Learning Methods and Applications,” *IEEE Trans Comput Soc Syst*, vol. 8, no. 1, pp. 214 – 226, 2021, doi: 10.1109/TCSS.2020.3021467.

[14] T. H. H. Aldhyani, S. N. Alsubari, A. S. Alshebami, H. Alkahtani, and Z. A. T. Ahmed, “Detecting and Analyzing Suicidal Ideation on Social Media Using Deep Learning and Machine Learning Models,” *Int J Environ Res Public Health*, vol. 19, no. 19, Oct. 2022, doi: 10.3390/ijerph191912635.

[15] Z. Wang and J. Lv, "Data Crawling and Research Based on Topic Web Crawler," in *2022 International Conference on Computer Network, Electronic and Automation (ICCNEA)*, 2022, pp. 183–188. doi: 10.1109/ICCNEA57056.2022.00049.

[16] M. E. Weishaar and A. T. Beck, "Hopelessness and suicide," *International Review of Psychiatry*, vol. 4, no. 2, pp. 177–184, 1992, doi: 10.3109/09540269209066315.

[17] M. Chatterjee, P. Kumar, P. Samanta, and D. Sarkar, "Suicide ideation detection from online social media: A multi-modal feature based technique," *International Journal of Information Management Data Insights*, vol. 2, no. 2, p. 100103, 2022, doi: https://doi.org/10.1016/j.jjimei.2022.100103.

[18] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," *Inf Process Manag*, vol. 50, no. 1, pp. 104–112, 2014, doi: https://doi.org/10.1016/j.ipm.2013.08.006.

[19] S. Kotsiantis, D. Kanellopoulos, and P. E. Pintelas, "Data Preprocessing for Supervised Learning," 2014. [Online]. Available: https://www.researchgate.net/publication/228084519

[20] S. Robertson, "Understanding inverse document frequency: On theoretical arguments for IDF," *Journal of Documentation*, vol. 60, no. 5, pp. 503–520, 2004, doi: 10.1108/00220410410560582.

[21] E. B. Setiawan, D. H. Widyantoro, and K. Surendro, "Feature expansion using word embedding for tweet topic classification," in *2016 10th International Conference on Telecommunication Systems Services and Applications (TSSA)*, 2016, pp. 1–5. doi: 10.1109/TSSA.2016.7871085.

[22] L. Alzubaidi *et al.*, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *J Big Data*, vol. 8, no. 1, Dec. 2021, doi: 10.1186/s40537-021-00444-8.

[23] A. Pulver and S. Lyu, "LSTM with Working Memory," May 2016, [Online]. Available: http://arxiv.org/abs/1605.01988

[24] X. She and D. Zhang, "Text Classification Based on Hybrid CNN-LSTM Hybrid Model," in *2018 11th International Symposium on Computational Intelligence and Design (ISCID)*, 2018, pp. 185–189. doi: 10.1109/ISCID.2018.10144.

[25] I. Düntsch and G. Gediga, "Confusion Matrices and Rough Set Data Analysis," *J Phys Conf Ser*, vol. 1229, no. 1, p. 12055, May 2019, doi: 10.1088/1742-6596/1229/1/012055.