Jurnal Sistem Komputer dan Informatika (JSON)
Volume 5, Nomor 2, Desember 2023
e-ISSN 2685-998X
DOI 10.30865/json.v5i2.6950

Implementasi Speech Recognition Menggunakan Long Short-Term Memory untuk Software Presentasi

Satriya Adhitama*, Donny Avianto

Fakultas Sains & Teknologi, Program Studi Informatika, Universitas Teknologi Yogyakarta, Yogyakarta, Indonesia
Email: [1,*]satadhitama@gmail.com, [2]donny@uty.ac.id
Email Penulis Korespondensi: satadhitama@gmail.com
Submitted: **31/10/2023;** Accepted: **31/12/2023;** Published: **31/12/2023**

Abstrak–Presentasi merupakan salah satu metode untuk menyampaikan pikiran, ide, maupun gagasan kepada audiens secara lisan. Kegiatan presentasi dapat didukung dengan perangkat lunak presentasi yang dapat digunakan untuk mengatur urutan materi yang akan disampaikan dengan visual yang menarik. Pengoperasian perangkat lunak presentasi membutuhkan bantuan teknis seperti remote, mouse, keyboard, bahkan juga asisten personal yang dapat menjadi distraksi bagi presenter karena menjadi kurang leluasa dalam menyampaikan materi. Distraksi ini dapat diatasi dengan implementasi pengenalan ucapan sebagai kata perintah untuk mengoperasikan perangkat lunak presentasi untuk memudahkan presenter. Sistem pengenalan ucapan dibuat dengan Long Short-Term Memory (LSTM) yang mampu menangani masalah dependensi jangka panjang dan vanishing gradient dari Reccurent Neural Network (RNN). Terdapat 10 kata perintah yang digunakan untuk mengoperasikan perangkat lunak presentasi. LSTM mampu mengungguli metode lain seperti DNN, CNN, dan Simple RNN dengan hasil akurasi latih 96.5%, validasi 94.8%, dan uji sebesar 94%. Metode LSTM dapat dengan baik digunakan untuk data sequential seperti agar dapat mengenali ucapan yang muncul secara real-time.

Kata Kunci: Pengenalan Ucapan; Klasifikasi; LSTM; Perintah Ucapan; Perangkat Lunak Presentasi

Abstract–Presentation is one of the methods for delivering thoughts, ideas, and concepts to an audience verbally. Presentation activities can be supported by presentation software that can be used to organize the sequence of material to be presented with visually appealing visuals. Operating presentation software requires technical assistance such as a remote, mouse, keyboard, and even a personal assistant, which can be distracting to the presenter as it limits their freedom in delivering the material. This distraction can be addressed through the implementation of speech recognition as a command to operate presentation software, making it easier for the presenter. A speech recognition system is developed using Long Short-Term Memory (LSTM), which can handle the issues of long-term dependency and vanishing gradient associated with Recurrent Neural Networks (RNN). There are 10 command words used to operate the presentation software. LSTM demonstrates superior performance when compared to alternative techniques like DNN, CNN, and SimpleRNN, achieving a training accuracy of 96.5%, a validation accuracy of 94.8%, and a testing accuracy of 94%. The LSTM method can be effectively used for sequential data to recognize real-time speech.

Keywords: Speech Recognition; Classification; LSTM; Speech Command; Presentation Software

1. PENDAHULUAN

Presentasi merupakan kegiatan untuk mengungkapkan ide, gagasan, dan argumen dengan bahasa lisan[1]. Presentasi dibuat sedemikian rupa sehingga penyampaian dan penampilan materi menjadi lebih menarik sehingga muncul rasa keingintahuan audiens terhadap informasi yang akan disampaikan. Tak jarang juga presentasi dibuat dalam bentuk yang interaktif sehingga dapat tercapai komunikasi yang efektif antara pembicara dengan audiens. Software presentasi seperti Power Point, Google Slide, Canva menjadi alat pendukung untuk menyusun materi tampilan materi presentasi yang dapat dioperasikan melalui komputer dengan bantuan perangkat proyeksi tampilan layar.

Namun, upaya lebih harus dikeluarkan pembicara untuk dapat mengoperasikan perangkat pendukung presentasi. Pembicara harus selalu mengarahkan pointer ke bagian yang dinginkan atau membutuhkan asisten untuk dapat membantu melakukan interaksi dengan software presentasi. Pembicara memerlukan alat tambahan untuk dapat mengoperasikan software presentasi seperti keyboard, mouse, atau remote control. Hal ini menimbulkan distraksi bagi pembicara karena harus menangani hal teknis sekaligus menyampaikan materi secara bersamaan. Distraksi penggunaan software presentasi ini dapat diatasi dengan adanya teknologi automatic speech recognition (ASR).

ASR adalah metode untuk decoding dan transkripsi ucapan lisan [2]. Umumnya, AR digunakan untuk konversi ucapan menjadi teks [3]. Sistem dirancang dengan teknik tertentu yang bertujuan untuk mengenali dan memproses suara manusia [4]. Prinsip dasar dari ASR yaitu seseorang berbicara mengeluarkan variasi tekanan suara pada larynx (pangkal tenggorokan), kemudian suara yang dihasilkan akan digitalisasi menggunakan microphone dan dikirimkan melalui sebuah perantara atau jaringan [5].

Deep Neural Network (DNN) dapat digunakan untuk penerapan speech recognition. Arsitektur RNN mampu untuk menunjukkan pengaruh dari output lampau untuk dapat menghasilkan perhitungan output baru, sehingga membuat metode ini cocok digunakan untuk data time series [6]. Recurrent Neural Network (RNN) muncul untuk menangani dependensi sementara. Kemudian, LSTM hadir untuk memperbaiki permasalahan dependensi jangka panjang pada RNN yang menyebabkan vanishing atau exploding gradient [7].

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 5, Nomor 2, Desember 2023
e-ISSN 2685-998X
DOI 10.30865/json.v5i2.6950

Pada penelitian [8] mengujikan 3 metode sequential, yaitu RNN, LSTM, dan GRU menggunakan data TED-LIUM yang berasal dari audio percakapan serta transkripsi dari website TED, dengan menggunakan 11.7GB dari 34.3GB data yang tersedia . Penelitian tersebut menunjukkan bahwa hasil Word of Error (WER) untuk arsitektur 500 layer node adalah 87.02% untuk RNN, 77.55% untuk LSTM, dan 79.39% untuk GRU. Sedangkan, WER untuk arsitektur dengan 1000 layer node menghasilkan 78.66% untuk RNN, 65.04% untuk LSTM, dan 67.42% untuk GRU. LSTM menghasilkan scoring yang lebih baik daripada kedua metode sequential lainnya, namun memiliki running time yang lebih lama dibandingkan dengan GRU.

Pada penelitian lain, RNN juga digunakan untuk pengenalan ucapan Bahasa Indonesia. Arsitektur MFCC dan ekstraksi ciri mel-frequency spectral coefficient (MFCC) menghasilkan nilai rata-rata akurasi tertinggi sebesar 73.55% [9]. Namun, RNN dinilai belum dapat memberikan hasil yang optimal karena sulit untuk mengenali kata yang sama.

LSTM telah banyak diterapkan untuk kasus pengenalan ucapan pada bahasa tertentu. Klasifikasi ucapan nama hewan dalam Bahasa Sunda dengan metode LSTM menggunakan 1,600 data suara menghasilkan akurasi sebesar 97.50% [10]. Penelitian lain menerapkan pengenalan ucapan dalam Bahasa Inggris untuk keyword spotting pada perangkat microcontroller. Dengan dataset speech_command_v0.01, model LSTM menghasilkan akurasi sebesar 92.9% untuk NN ukuran kecil (batas memori 80KB & batas operasi 6MOps), 93.9% untuk NN ukuran sedang (batas memori 200KB & batas operasi 20MOps), dan 94.8% untuk NN ukuran besar (batas memori 500KB & batas operasi 80MOps) [11]. LSTM juga dapat digunakan untuk mengenali dialek China Sichuan. Rekaman suara dialek Sichuan yang digunakan memiliki rentang waktu 5-10 detik dengan terdapat 4.000 teks pada 3 kategori berbeda. LSTM mampu mengenali dialek Sichuan dengan training frame accuracy 72.98% dan validation training frame 66.95%. Training character error rate (CER) yang dihasilkan sebesar 18.09% mengungguli model lain, yaitu Gaussian Mixture Modelling (GMM) dan DNN [12].

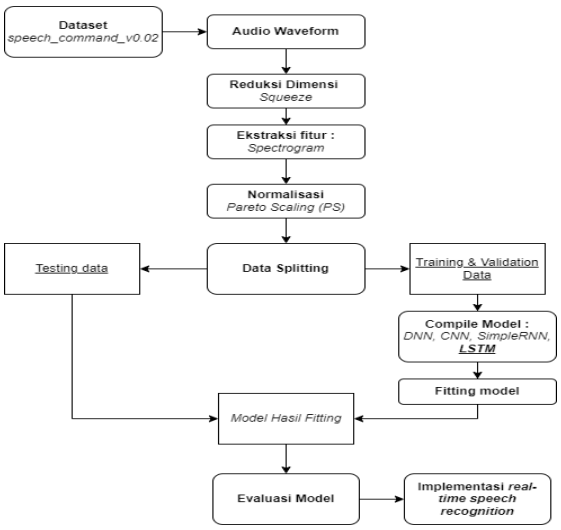
Selain pengenalan ucapan, LSTM juga dapat digunakan dengan baik untuk kasus lainnya, seperti klasifikasi teks. Penggunaan LSTM pada penelitian tersebut karena struktur dari LSTM yang merupakan rangkaian terintegrasi sehingga makna kalimat tidak akan berubah [13]. Dengan menggunakan ekstraksi fitur Global Vector (GloVe), LSTM menghasilkan akurasi sebesar 95.17%.

Implementasi pengenalan ucapan (speech recognition) dilakukan menggunakan model sequential learning. LSTM yang merupakan turunan dari RNN telah mencapai peningkatan performa untuk Automatic Speech Recognition [14]. LSTM dapat menangkap informasi time series yang cocok digunakan untuk pengenalan ucapan (speech recognition) [15]. LSTM akan dapat mengenali kata tertentu yang muncul pada sinyal suara (keyword spotting).

2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Penelitian ini memiliki tahapan yang terdiri dari pemrosesan dataset speech_command_v0.02, ekstraksi fitur dengan spectrogram, normalisasi, pembuatan dan fitting model, evaluasi, kemudian implementasi real-time speech recognition. Gambar 1 menunjukkan tahapan-tahapan yang dilakukan pada penelitian ini.



Gambar 1. Tahapan penelitian implementasi sistem real-time speech recognition

2.2 Dataset

Dataset untuk penelitian ini menggunakan data suara yang berisi ucapan kata perintah untuk mengoperasikan perangkat lunak presentasi. Dataset speech_command_v0.02 tersedia secara publik dengan termuat data ucapan

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 5, Nomor 2, Desember 2023
e-ISSN 2685-998X
DOI 10.30865/json.v5i2.6950

terdiri atas kata berbahasa inggris berdurasi 1 detik yang diambil melalui microphone telepon atau laptop pengguna [16]. Dataset terdiri atas 34 kosa kata yang diucapkan oleh subjek yang beragam dan jenis kelamin yang berbeda. Sistem yang dirancang hanya akan menggunakan 10 kata yang tersedia dengan jumlah file sebanyak 33.782 audio sesuai dengan kebutuhan ditunjukkan pada Tabel 1.

Tabel 1. Kata kunci suara dan perintah

No	Kata Kunci	Jumlah File	Perintah
1	Backward	1.664	Menuju ke slide sebelumnya
2	Down	3.917	Menuju ke slide berikutnya saat tidak mode slide show
3	Forward	1.557	Menuju ke slide berikutnya
4	Go	3.880	Memainkan media player
5	Left	3.801	Menuju ke slide sebelumnya saat mode slide show
6	Off	3.745	Mengubah state app menjadi OFF (tidak menerima perintah) hingga dihidupkan kembali
7	On	3.845	Mengubah state app menjadi ON (mampu menerima perintah)
8	Right	3.778	Menuju ke slide berikutnya saat mode slide show
9	Stop	3.872	Mematikan streaming audio atau terminate program
10	Up	3.723	Menuju ke slide sebelumnya ketika tidak sedang slide show

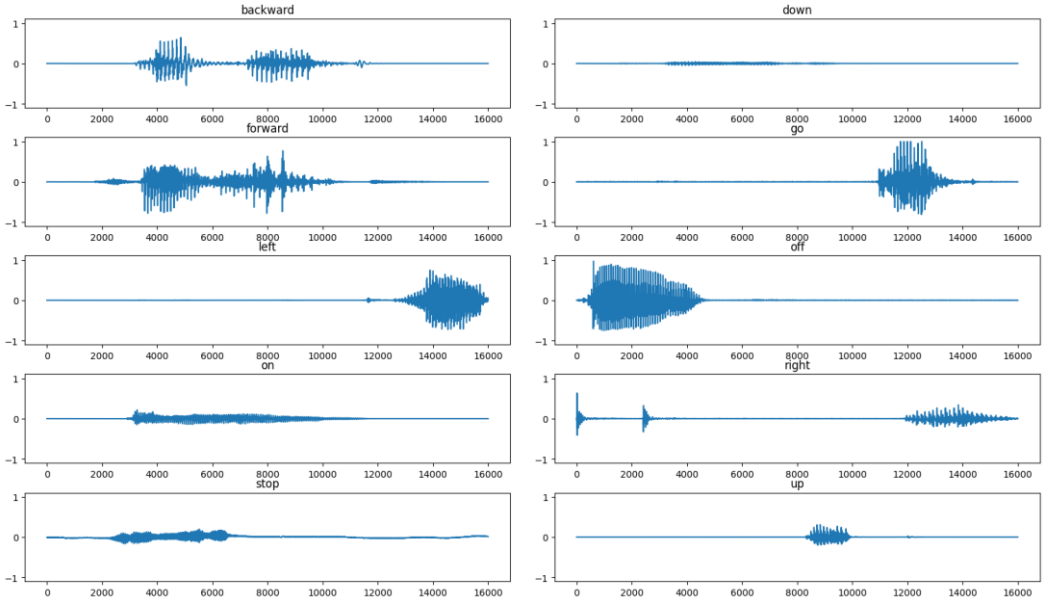
Dataset dibagi menjadi 3 sesuai dengan kegunaannya, yaitu data latih (training), validasi (validation), dan uji (testing). Persentase splitting terbagi menjadi 80% (27.026) untuk training, 10% (3.378) untuk validation, dan 10% (3.378) untuk testing.

Tabel 2. Pembagian dataset

Pembagian	Jumlah Data Audio
Training	27.026
Validation	3.378
Testing	3.378

2.3 Waveform

Waveform merupakan representasi dari sinyal suara berdasarkan domain waktu dengan menunjukkan perubahan amplitudo yang terjadi. Gambar 2 menunjukkan visualisasi sampel waveform dari sinyal suara untuk 10 kata perintah.

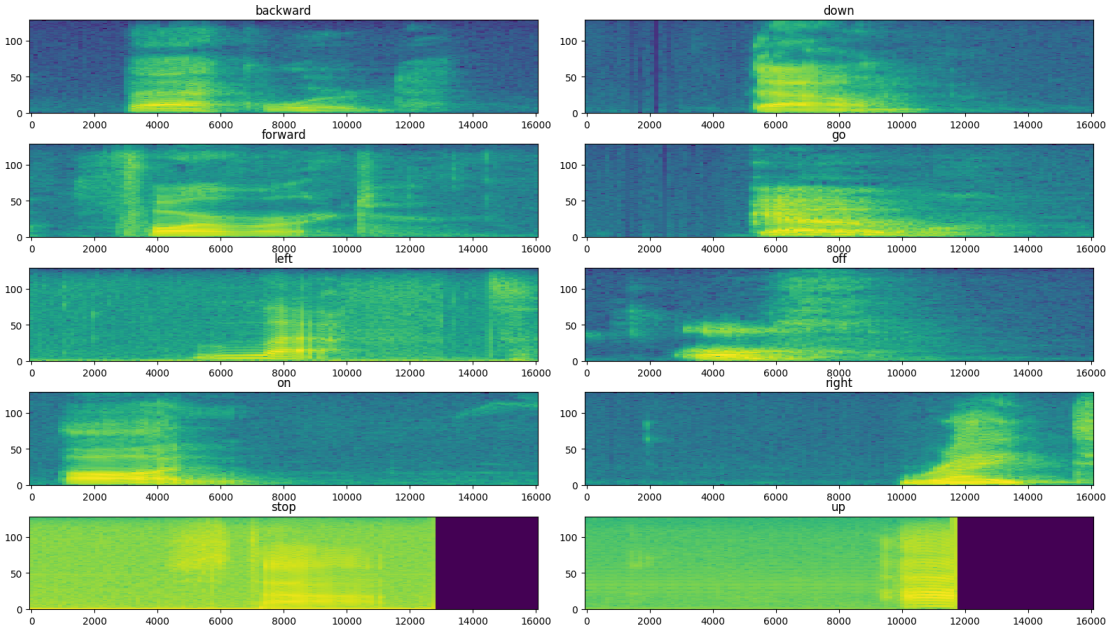


Gambar 2. Visualisasi waveform 10 kata perintah

2.4 Spectrogram

Spectrogram merepresentasikan perubahan frekuensi pada sinyal berdasarkan waktu. Representasi spectrogram dapat digunakan dengan baik untuk pemrosesan, deteksi, dan klasifikasi ucapan [17]. Gambar 4 menunjukkan visualisasi sampel spectrogram untuk 10 kata perintah yang terpilih.

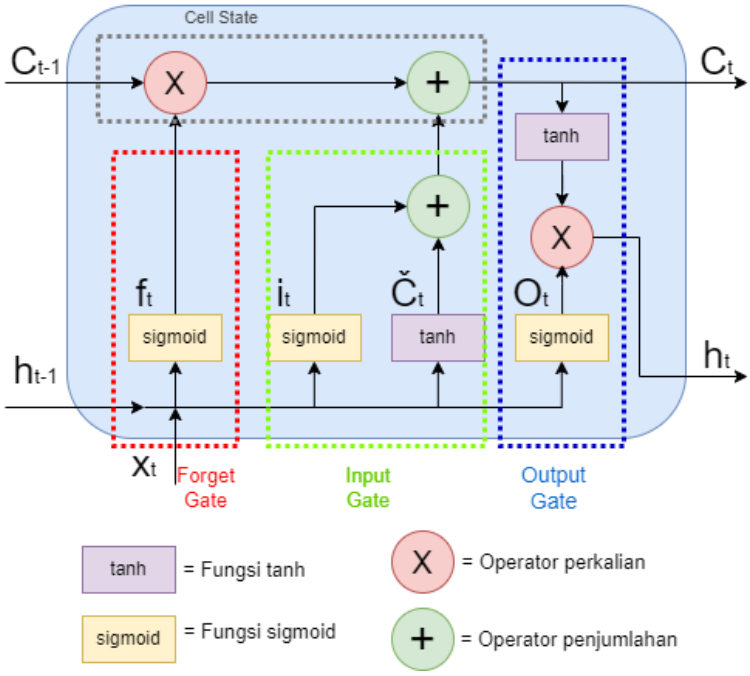
Jurnal Sistem Komputer dan Informatika (JSON)
Volume 5, Nomor 2, Desember 2023
e-ISSN 2685-998X
DOI 10.30865/json.v5i2.6950



Gambar 3. Visualisasi spectrogram 10 kata perintah

2.5 Long Short-Term Memory (LSTM)

Long short-term memory (LSTM) merupakan pengembangan metode dari recurrent neural network (RNN) yang dirancang dengan memory cell yang mampu merepresentasikan dependensi jangka panjang (long-term dependency) terhadap urutan waktu yang terjadi pada data [18]. LSTM dirancang untuk menangani dependensi jangka panjang (long-term dependency) [19], [20]. LSTM cocok digunakan untuk mengatasi data sequential seperti time series.



Gambar 4. Arsitektur LSTM

LSTM terdiri atas tiga gerbang yang berperan untuk mengelola cell state [21]. Cell state bertanggung jawab atas menjaga dan mengatur informasi pada jaringan atas sequcence atau time steps yang panjang. Input gate digunakan untuk mengontrol informasi baru yang masuk pada cell state. Nilai input i_t dan nilai sel memori $\tilde{C}_t$ pada waktu t, dapat dilakukan perhitungan sebagai berikut:

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \quad (1)$$

$$\tilde{C}_t = \tanh(W_c[h_{t-1}, x_t] + b_c) \quad (2)$$

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 5, Nomor 2, Desember 2023
e-ISSN 2685-998X
DOI 10.30865/json.v5i2.6950

$W_i.[h_{t-1}, x_t]$ dan b merupakan bobot matriks dan bias, secara berurutan. Forget gate bertanggung jawab untuk menentukan informasi yang harus dibuang atau disimpan dari cell state sebelumnya. Nilai forget gate dapat dihitung sebagai berikut:

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \quad (3)$$

Cell state membawa informasi penting berdasarkan waktu untuk menyimpan dan mengelola dependensi jangka panjang pada data sequential. State baru dari sel memori diperbarui dengan perhitungan berikut:

$$\tilde{C}_t = i_t.\tilde{C}_t + f_t.\tilde{C}_{t-1} \quad (4)$$

Output gate digunakan untuk mengontrol output hidden state dan keseluruhan sel. Setelah mendapatkan state sel memori yang baru, output gate menggunakan perhitungan berikut:

$$o_t = \sigma(W_x.[h_{t-1}, x_t] + b_o) \quad (5)$$

Dengan demikian, hasil akhir dari sel didapatkan:

$$h_t = o_t * \tanh(c_t) \quad (6)$$

2.6 Normalisasi

Normalisasi adalah metode transformasi data menjadi bentuk yang lebih standar dan terstruktur sehingga data dapat dianalisis dan diinterpretasi dengan baik. Penggunaan normalisasi pada penelitian ini juga bertujuan untuk mengurangi beban komputasi dan meningkatkan performa model. Jenis metode normalisasi yang diterapkan pada penelitian ini menggunakan Pareto Scaling (PS), dengan perhitungan berikut:

$$X'_{i,n} = \frac{x_{i,n} - \mu_i}{\sqrt{\sigma_i}} \quad (7)$$

Dengan hasil normalisasi $X'_{i,n}$ diperoleh dari pengurangan input data $x_{i,n}$ dengan mean input μ_i dibagi dengan akar varians σ_i. Metode ini meningkatkan representasi dari fitur dengan konsentrasi rendah dengan juga meminimalkan noise dari data [22].

2.7 Regularisasi Dropout

Regularisasi adalah metode sebuah proses untuk mengatasi permasalahan overfitting pada model yang dapat memengaruhi performa model. Konsep dari fully connected layer menimbulkan masalah eksponensial memori yang disebut overfitting sebagai akibat dari koneksi neuron yang terlalu banyak dan berlebih sehingga pembelajaran yang dilakukan terlalu berlebihan [23]. Teknik regularisasi dropout dilakukan dengan secara acak menjadikan bagian input unit menjadi 0 pada setiap langkah training. Beberapa neuron akan dinonaktifkan untuk mengurangi interdependensi neuron dan terlalu bergantung kepada fitur spesifik. Metode dropout memiliki peran yang penting untuk dapat menghasilkan akurasi model yang tinggi dengan loss yang sangat rendah [24].

2.8 Sparse Categorical Cross Entropy

Cost function menjadi kunci utama untuk menyesuaikan bobot neural network sehingga dapat menghasilkan model machine learning yang baik. Secara spesifik, pada saat propagasi maju, neural network dijalankan untuk melatih data dan dihasilkan output klasifikasi yang mengindikasikan probabilitas kebenaran label [25]. Sparse categorical crossentropy adalah fungsi loss yang umum digunakan untuk kasus klasifikasi multi kelas ketika label target merupakan bilangan bulat (integer). Matriks sparse merepresentasikan label sebagai sebuah nilai indeks tunggal daripada vektor one-hot encoding. Perhitungan loss dapat dilakukan perhitungan sebagai berikut [26]:

$$H(Y, \hat{Y}) = -\frac{1}{n}\sum_{i=1}^{n} \log(\hat{y}_{i,y_i}) \quad (8)$$

Dimana y_i adalah kelas true dari sampel ke-i dan $\hat{y}_{i,y_i}$ adalah kemungkinan terprediksi dari sampel ke-i untuk kelas yang benar y_i.

3. HASIL DAN PEMBAHASAN

3.1 Skenario Pengujian Model

Pengujian dilakukan pada 4 (empat) arsitektur model neural network yang berbeda, yaitu Deep Neural Network (DNN), Convolutional Neural Network (CNN), Simple RNN, dan Long Short-Term Memory (LSTM). Pengujian pada metode yang berbeda bertujuan untuk mengetahui perbandingan performa model LSTM terhadap model neural network lainnya.

Keempat model machine learning dilatih dengan tiga (3) hyperparameter yang sama, yaitu batch size sebanyak 64, algoritma optimisasi Adam, dan learning rate sebesar 0,001. Penggunaan jumlah layer dan dropout

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 5, Nomor 2, Desember 2023
e-ISSN 2685-998X
DOI 10.30865/json.v5i2.6950

rate ditetapkan secara berbeda sesuai dengan kompleksitas komputasi model serta jumlah layer dan unit yang tersedia untuk mencegah terjadinya overfitting.

Tabel 3. Arsitektur model yang diuji

Model	Layer
DNN	Input - Normaliasasi - Flatten - Dense(512) - Dropout(0.2)* - Dense(256) - Dropout(0.25)* - Dense(128) - Dropout(0.2) - Dense(64) - Dense(10)
CNN	Input - Resize(32,32) - Normalisasi - Conv2D(32) - Conv2D(64) - MaxPooling2D - Dropout(0.25)* - Flatten - Dense(512) - Dropout(0.5)* - Dense(10)
SimpleRNN	Input - Normalisasi - SimpleRNN(256) - Dropout(0.3)* - SimpleRNN(128) - Dropout(0.2)* - SimpleRNN(64) - Dense(10)
LSTM	Input - Normalisasi - Reshape - LSTM(256) - Dropout(0.4)* - LSTM(128) - Dropout(0.2)* - LSTM(64) - Dense(10)

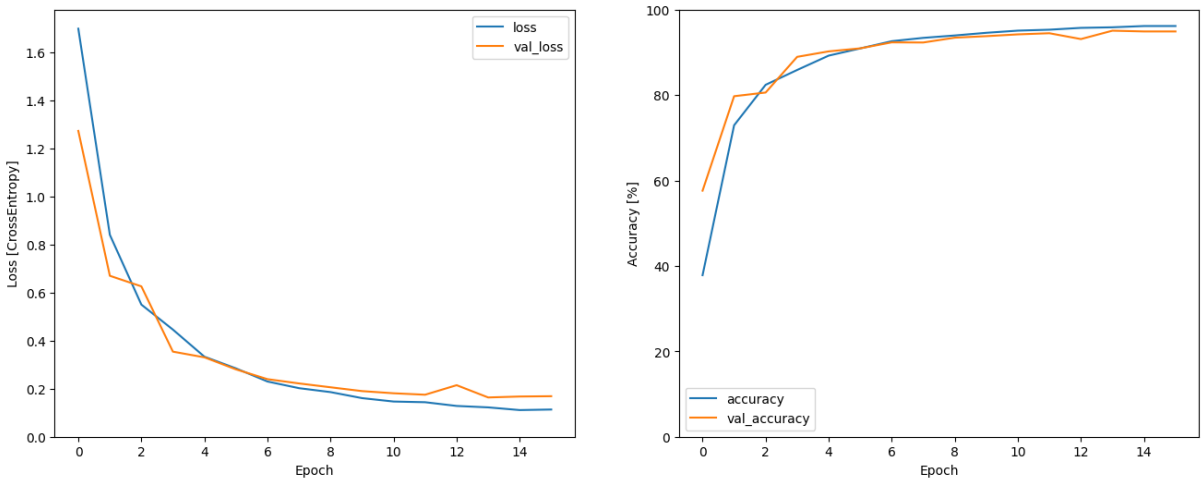
Keempat arsitektur model yang terdapat pada Tabel 3 dilakukan eksperimen dengan variabel pengujian dropout layer. Pengujian ini dilakukan untuk mengetahui pengaruh dari regularisasi dropout layer terhadap performa model yang dihasilkan. Jumlah unit dan layer pada arsitektur model yang diuji akan sama dengan yang terdapat pada Tabel 3. Namun, dropout layer dapat dihilangkan untuk menguji pengaruh pada arsitektur model yang menggunakan dan tidak menggunakan regularisasi.

3.2 Hasil Pengujian Model

Keempat arsitektur model dilakukan fitting dengan data yang sama untuk mengetahui perbandingan performa yang dapat dihasilkan. Kondisi early stopping diterapkan untuk setiap proses fitting dengan melihat nilai loss yang dihasilkan pada setiap epoch-nya. Apabila loss tidak mengalami penurunan pada epoch berikutnya, maka proses fitting akan dihentikan. Hal ini menunjukkan bahwa model telah mencapai konvergensi, sehingga dapat ditetapkan sebagai bobot final.

3.2.1 Hasil Pengujian Model Tanpa Regularisasi

Pengujian pertama dilakukan dengan menghilangkan dropout layer pada arsitektur model yang terdapat pada Tabel 3.



Gambar 5. Grafik loss dan accuracy proses fitting LSTM tanpa regularisasi dropout

Gambar 5 menunjukkan grafik kurva hasil fitting model LSTM tanpa regularisasi dropout. Pada kedua grafik loss dan akurasi terlihat bahwa terjadi selisih yang cukup signifikan pada hasil fitting epoch pertama. Pada epoch pertama, menghasilkan nilai loss sebesar 1.6971 untuk training dan 1.2719 untuk validation. Sedangkan, akurasi yang didapatkan sebesar 37.82% untuk training dan 57.61% untuk validation. Selisih hasil antara training dan validation dapat menjadi pertanda bahwa model kurang mampu untuk menggeneralisasi data yang ada. Gambar 6 menunjukkan menurunnya selisih antara hasil training dan validation karena dilakukan regularisasi pada model LSTM.

Tabel 4. Perbandingan hasil evaluasi model tanpa regularisasi dropout

Model	Training		Validation		Testing		Epochs	Fitting Time
	Loss	Acc	Loss	Acc	Loss	Acc		
DNN	0.3494	0.8810	0.7697	0.7967	0.8027	0.7900	7	2m 28.2s

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 5, Nomor 2, Desember 2023
e-ISSN 2685-998X
DOI 10.30865/json.v5i2.6950

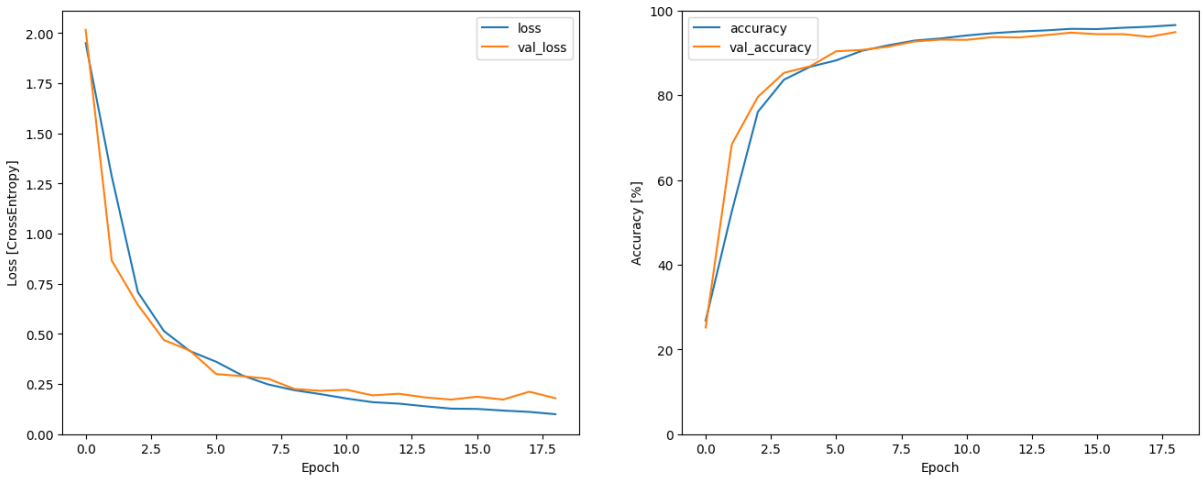
Model	Training		Validation		Testing		Epochs	Fitting Time
	Loss	Acc	Loss	Acc	Loss	Acc		
CNN	0.0931	0.9695	0.4419	0.8859	0.4760	0.8867	6	3m 21.s
SimpleRNN	2.2717	0.1139	2.2726	0.1049	2.2678	0.1208	3	2m 36.s
LSTM	0.1132	0.9616	0.1686	0.9489	0.1755	0.9436	16	81m 26s

Hasil fitting model DNN, CNN, SimpleRNN, dan LSTM ditunjukkan pada Tabel 4. DNN mengalami konvergensi pada epoch ke-7 dengan nilai loss 0.3494 untuk training, 0.7697 untuk validation, dan 0.8027 untuk testing. Akurasi yang dihasilkan DNN adalah 88.1% untuk training, 79.67% untuk validation, dan 79% untuk testing. CNN memiliki performa training yang paling baik di antara model lainnya tanpa regularisasi dengan hasil nilai loss 0.0931 dan akurasi 96.95%. Namun, terjadi gap yang signifikan antara hasil training dengan validation dan testing. Hasil training CNN menunjukkan nilai loss 0.4419 dan akurasi 88.59% serta validation loss 0.4760 dan akurasi 88.67%. SimpleRNN memiliki performa yang paling buruk di antara model lainnya dengan hasil loss 2.2717 untuk training, 2.2726 untuk validation, dan 2.2678 untuk testing. Akurasi SimpleRNN menunjukkan hasil 11.39% untuk training, 10.49% utnuk validation, dan 12.08% untuk testing. LSTM menjadi model yang terbaik dengan nilai loss 0.1132 untuk training, 0.1686 untuk validation, dan 0.1755 untuk testing.

Perbedaan signifikan yang terjadi pada hasil evaluasi CNN merupakan pertanda bahwa model mengalami overfitting. Overfit terjadi ketika model mengalami penurunan performa untuk menangani data yang berbeda. Dalam hal ini, hasil training yang cukup baik tidak berbanding lurus dengan hasil validation dan testing yang menurun cukup signifikan. Adapun yang dapat menjadi penyebab overfitting karena model tidak mampu melakukan generalisasi dengan baik. Pada Tabel 5 menunjukkan perbaikan hasil model CNN dengan menggunakan regularisasi dropout.

LSTM memiliki performa yang cukup mengungguli ketiga model lainnya. Namun, kompleksitas perhitungan yang dimiliki LSTM menyebabkan waktu fitting yang jauh lebih lama hingga 81 menit 26 detik. Tanpa ada dropout layer semua unit akan terkoneksi sepenuhnya dengan unit pada layer berikutnya. Hal tersebut menyebabkan beban komputasi yang meningkat sebagai akibat dari setiap sel LSTM melakukan perhitungan yang sama sesuai dengan jumlah unit pada layer arsitektur. Tabel 5 menunjukkan adanya penurunan waktu fitting yang cukup signifikan dengan adanya dropout layer.

3.2.2 Hasil Pengujian Model Tanpa Regularisasi



Gambar 6. Grafik loss dan accuracy proses fitting LSTM menggunakan regularisasi dropout

Gambar 6 menunjukkan kurva hasil fitting untuk model LSTM dengan menggunakan regularisasi dropout. Terlihat pada grafik hasil fitting bahwa model dapat mampu untuk menggeneralisasi data dibandingkan dengan Gambar 5. Pada epoch pertama, model menghasilkan nilai loss sebesar 1.9497 untuk training dan 2.0164 untuk testing. Sedangkan, akurasi yang didapatkan adalah 26.81% untuk training dan 25.18% untuk validation. Menurunnya selisih pada hasil awal fitting dan kemiringan kurva yang lebih halus pada setiap epoch-nya menandakan bahwa regularisasi dapat berjalan dengan baik untuk LSTM.

Tabel 5. Perbandingan hasil evaluasi model menggunakan regularisasi dropout

Model	Training		Validation		Testing		Epochs	Fitting Time
	Loss	Acc	Loss	Acc	Loss	Acc		
DNN	0.5989	0.8024	0.6255	0.8100	0.6622	0.8004	13	5m 50s
CNN	0.1562	0.9479	0.2975	0.9171	0.3355	0.9038	9	4m 58.9s
SimpleRNN	2.2646	0.1101	2.2689	0.1177	2.2658	0.1217	10	9m 14.9s
LSTM	0.0996	0.9656	0.1791	0.9489	0.2106	0.9404	19	59m 1.1s

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 5, Nomor 2, Desember 2023
e-ISSN 2685-998X
DOI 10.30865/json.v5i2.6950

Dari tabel 4 dapat diketahui bahwa DNN dengan epoch 13 menghasilkan akurasi dengan nilai 80.2% untuk training, 81% untuk validation, dan 80% untuk testing. Model DNN memiliki nilai loss 0.598 untuk pelatihan, 0.625 untuk validasi, dan 0.662 untuk pengujian. CNN dengan jumlah epoch yang paling sedikit, menghasilkan akurasi sebesar 94.7% untuk training, 91.7% untuk validation, dan 90.3% untuk testing. Evaluasi loss CNN sebesar 0.156 untuk training, 0.297 untuk validation, dan 0.335 untuk testing. Simple RNN menjadi yang terburuk di antara ketiga model lainnya dengan hasil akurasi 11% untuk training, 11.7% untuk validation, dan 12.1% untuk testing. Hasil loss SimpleRNN juga merupakan yang tertinggi dengan 2.264 untuk training, 2.268 untuk validation, dan 2.265 untuk testing. Sedangkan, LSTM menjadi model yang terbaik di antara ketiga model lainnya. LSTM menghasilkan akurasi training sebesar 96.5%, validation sebesar 94.8%, dan testing sebesar 94%. Nilai loss LSTM juga menjadi yang paling rendah di antara model lainnya, dengan 0.096 untuk training, 0.179 untuk validation, dan 0.21 untuk testing.

Model LSTM menunjukkan performa yang lebih mengungguli dari DNN, CNN, dan SimpleRNN. Namun, model yang diusulkan baru mengalami konvergensi pada epoch ke-19 dan fitting time hingga 59 menit 1,1 detik. Lamanya waktu fitting disebabkan oleh komputasi dari arsitektur LSTM yang cukup kompleks. Adanya gerbang-gerbang yang membentuk sel LSTM dengan setiap tahapan perhitungannya menjadikan beban komputasi lebih berat. Jumlah layer dan node yang menyusun arsitektur model juga memengaruhi waktu fitting.

3.3 Hasil Model Terbaik

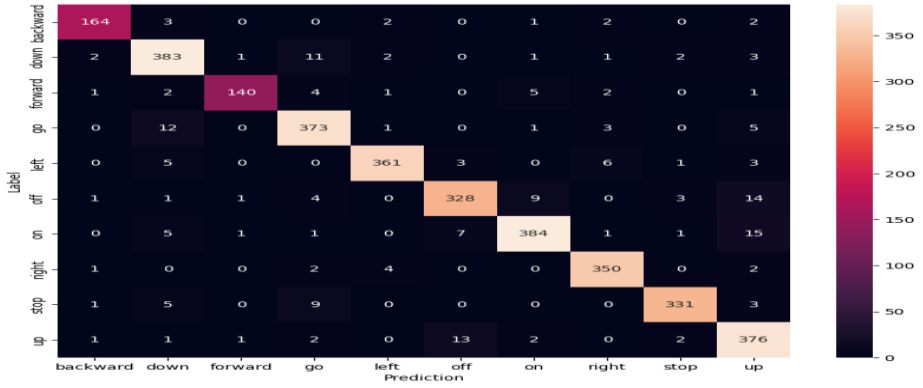
Dari eksperimen yang telah dilakukan, LSTM menjadi model yang terbaik dibandingkan dengan DNN, CNN, dan SimpleRNN. Pengujian dengan regularisasi dropout juga menunjukkan bahwa LSTM tetap menghasilkan performa yang paling baik di antara ketiga model lainnya. Tabel 4 berikut menunjukkan perbandingan evaluasi kedua arsitektur model LSTM yang diusulkan.

Tabel 6. Perbandingan evaluasi LSTM dengan dan tanpa regularisasi dropout

Model	Training		Validation		Testing		Epochs	Fitting Time
	Loss	Acc	Loss	Acc	Loss	Acc		
LSTM (tanpa dropout)	0.1132	0.9616	0.1686	0.9489	0.1755	0.9436	16	81m 26s
LSTM (dengan dropout)	0.0996	0.9656	0.1791	0.9489	0.2106	0.9404	19	59m 1.1s

Evaluasi model menunjukkan adanya perbedaan antara arsitektur LSTM yang menggunakan dan tidak menggunakan regularisasi dropout. Hasil training dari LSTM dengan menggunakan dropout lebih baik memiliki loss 0.0996 dan akurasi 96.56% dibandingkan tanpa dropout yaitu loss 0.1132 dan akurasi 96.16%. Pada validation akurasi yang dihasilkan sama, yaitu 94.89%. Namun, terjadi selisih yang tidak terlalu signifikan pada nilai loss LSTM tanpa dropout menghasilkan 0.1686 dan menggunakan dropout menghasilkan 0.1791. Hasil testing menunjukkan bahwa tanpa menggunakan regularisasi, LSTM menghasilkan hasil yang baik namun tidak terlalu signifikan dengan menggunakan dropout menghasilkan loss 0.1755 dan akurasi 94.36%. LSTM tanpa menggunakan regularisasi dropout menghasilkan loss sebesar 0.2106 dan akurasi 94.04%.

Berdasarkan evaluasi hasil keduanya pada loss dan akurasi, tidak terjadi selisih hasil yang signifikan terhadap penggunaan regularisasi dropout. Namun, perbedaan terjadi cukup signifikan pada epoch dengan early stopping dan waktu fitting. LSTM dengan regularisasi mengungguli performa waktu fitting dengan selisih 22 menit 4,9 detik. Penggunaan dropout layer pada arsitektur LSTM menurunkan beban komputasi model dengan memangkas beberapa node secara acak pada setiap epoch. Hal tersebut membuktikan bahwa LSTM dengan menggunakan regularisasi lebih efisien karena beban komputasi yang digunakan lebih ringan dengan hasil yang tetap baik. Dengan demikian, arsitektur LSTM dengan menggunakan regularisasi dropout dipilih sebagai model yang terbaik dan digunakan untuk implementasi sistem real-time speech recognition.



Gambar 7. Confusion matrix data testing dengan arsitektur LSTM menggunakan dropout

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 5, Nomor 2, Desember 2023
e-ISSN 2685-998X
DOI 10.30865/json.v5i2.6950

Gambar 7 menunjukkan confusion matrix dengan arsitektur LSTM menggunakan dropout terhadap data uji. Dengan hasil tersebut, dapat ditentukan nilai recall, precision, dan F1-score untuk 10 kelas kata perintah yang tersedia pada data testing yang ditunjukkan pada Tabel 5.

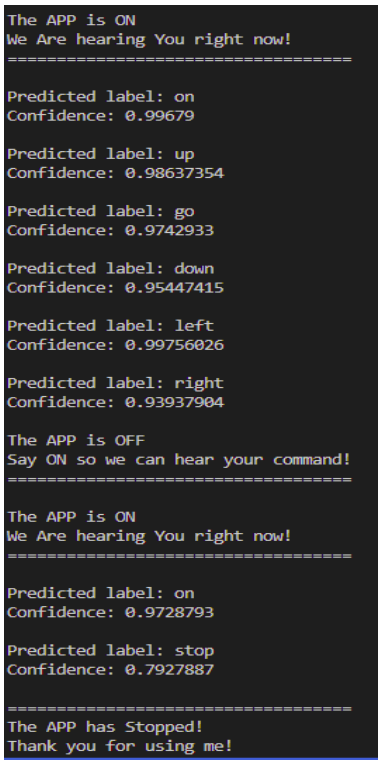
Tabel 7. Recall, Precision, dan F1-Score untuk setiap kata perintah

Class	Recall	Precision	F1-Score
Backward	0.943	0.959	0.951
Down	0.943	0.918	0.931
Forward	0.897	0.972	0.933
Go	0.944	0.919	0.931
Left	0.953	0.973	0.963
Off	0.909	0.934	0.921
On	0.925	0.953	0.939
Right	0.975	0.959	0.967
Stop	0.948	0.974	0.961
Up	0.945	0.887	0.915
Average	0.9382	0.9448	0.9412

Arsitektur model LSTM menggunakan regularisasi dropout menghasilkan performa yang cukup baik untuk dapat mengenali 10 kata perintah yang telah ditentukan. Tabel 5 menunjukkan evaluasi data testing dengan hasil nilai rata-rata recall 0.9382, precision 9448, dan F1-score 0.9412.

3.4 Implementasi Real-Time Speech Recognition

Implementasi sistem real-time speech recognition untuk presentasi telah berhasil dilakukan menggunakan model arsitektur LSTM untuk mengenali 10 kata perintah yang ada. Sistem dikembangkan dengan menggunakan Python. Interface sistem dibuat pada command line untuk menampilkan hasil log setiap kata yang terdeteksi. Sistem akan menunjukkan kata yang terdeteksi dan nilai confidence jika pengguna mengucapkan kata perintah yang tersedia. Adapun nilai threshold confidence yang ditentukan untuk dapat mengeksekusi perintah adalah 0.75.



Gambar 8. Hasil implementasi sistem real-time speech recognition

4. KESIMPULAN

LSTM dapat dengan baik digunakan untuk implementasi sistem real-time speech recognition. Model dengan arsitektur LSTM terbukti lebih baik daripada beberapa arsitektur model lainnya, yaitu DNN, CNN, dan SimpleRNN dalam menangani data sequential seperti sinyal suara. LSTM menghasilkan performa yang cukup baik dengan akurasi 96.56% untuk training, 94.89% untuk validation, dan 94.04% untuk testing. Penggunaan

regularisasi dropout cukup berpengaruh untuk menggeneralisasi dan meningkatkan performa model, serta mencegah terjadinya overfitting. Dengan menggunakan arsitektur LSTM yang telah dibuat, sistem mampu mengenali 10 kata perintah yang telah ditentukan dengan rata-rata recall 0.9382, precision 0.9448, dan F1-score 0.9412. Untuk pengembangan penelitian selanjutnya, perlu adanya eksperimen yang lebih bervariasi terhadap model yang diusulkan. Perlu adanya pengujian hyperparameter seperti jumlah layer dan node, algoritma optimasi, nilai learning rate, dan ukuran batch. Lebih lanjut, penelitian dapat dikembangkan untuk mengujikan dengan jenis ekstraksi fitur yang lain dan penanganan noise pada data suara.

REFERENCES

[1] I. Lisnawati dan Y. Ertinawati, "Literat Melalui Presentasi," Jurnal Metaedukasi, vol. 1, no. 1, hlm. 1–12, 2019, doi: https://doi.org/10.37058/metaedukasi.v1i1.976.
[2] J. Levis dan R. Suvorov, "Automatic Speech Recognition," dalam The Encyclopedia of Applied Linguistics, Wiley, 2012. doi: 10.1002/9781405198431.wbeal0066.
[3] W. Mustikarini, R. Hidayat, dan A. Bejo, "Real-Time Indonesian Language Speech Recognition with MFCC Algorithms and Python-Based SVM," IJITEE (International Journal of Information Technology and Electrical Engineering), vol. 3, no. 2, hlm. 55, Okt 2019, doi: 10.22146/ijitee.49426.
[4] S. Sen, A. Dutta, dan N. Dey, Audio Processing and Speech Recognition. Singapore: Springer Singapore, 2019. doi: 10.1007/978-981-13-6098-5.
[5] J. L. K. E. Fendji, D. C. M. Tala, B. O. Yenke, dan M. Atemkeng, "Automatic Speech Recognition Using Limited Vocabulary: A Survey," Applied Artificial Intelligence, vol. 36, no. 1, Des 2022, doi: 10.1080/08839514.2022.2095039.
[6] Gustav Bagus Samanta, "Classification Analysis using CNN and LSTM on Wheezing Sounds," International Journal On Information And Communication Technology (IJOICT), vol. 8, no. 1, hlm. 60–88, Agu 22M, doi: https://doi.org/10.21108/ijoict.v8i1.621.
[7] A. W. Saputra, A. P. Wibawa, U. Pujianto, A. B. Putra Utama, dan A. Nafalski, "LSTM-based Multivariate Time-Series Analysis: A Case of Journal Visitors Forecasting," ILKOM Jurnal Ilmiah, vol. 14, no. 1, hlm. 57–62, Apr 2022, doi: 10.33096/ilkom.v14i1.1106.57-62.
[8] A. Shewalkar, D. Nyavanandi, dan S. A. Ludwig, "Performance Evaluation of Deep Neural Networks Applied to Speech Recognition: RNN, LSTM and GRU," Journal of Artificial Intelligence and Soft Computing Research, vol. 9, no. 4, hlm. 235–245, Okt 2019, doi: 10.2478/jaiscr-2019-0006.
[9] P. Tridarma dan S. N. Endah, "Pengenalan Ucapan Bahasa Indonesia Menggunakan MFCC dan Recurrent Neural Network," JURNAL MASYARAKAT INFORMATIKA, vol. 11, no. 2, hlm. 36–44, Nov 2020, doi: 10.14710/jmasif.11.2.34874.
[10] N. Aini Lailla Asri, R. Ibnu Adam, dan B. Arif Dermawan, "SPEECH RECOGNITION UNTUK KLASIFIKASI PENGUCAPAN NAMA HEWAN DALAM BAHASA SUNDA MENGGUNAKAN METODE LONG-SHORT TERM MEMORY," JATI (Jurnal Mahasiswa Teknik Informatika), vol. 7, no. 2, hlm. 1242–1247, Sep 2023, doi: 10.36040/jati.v7i2.6744.
[11] Y. Zhang, N. Suda, L. Lai, dan V. Chandra, "Hello Edge: Keyword Spotting on Microcontrollers," Nov 2017, [Daring]. Tersedia pada: http://arxiv.org/abs/1711.07128
[12] W. Ying, L. Zhang, dan H. Deng, "Sichuan dialect speech recognition with deep LSTM network," Front Comput Sci, vol. 14, no. 2, hlm. 378–387, Apr 2020, doi: 10.1007/s11704-018-8030-z.
[13] W. K. Sari, D. P. Rini, dan R. F. Malik, "Text Classification Using Long Short-Term Memory With GloVe Features," Jurnal Ilmiah Teknik Elektro Komputer dan Informatika, vol. 5, no. 2, hlm. 85, Feb 2020, doi: 10.26555/jiteki.v5i2.15021.
[14] J. Oruh, S. Viriri, dan A. Adegun, "Long Short-Term Memory Recurrent Neural Network for Automatic Speech Recognition," IEEE Access, vol. 10, hlm. 30069–30079, 2022, doi: 10.1109/ACCESS.2022.3159339.
[15] L. Xiang, S. Lu, X. Wang, H. Liu, W. Pang, dan H. Yu, "Implementation of LSTM Accelerator for Speech Keywords Recognition," dalam 2019 IEEE 4th International Conference on Integrated Circuits and Microsystems (ICICM), IEEE, Okt 2019, hlm. 195–198. doi: 10.1109/ICICM48536.2019.8977176.
[16] P. Warden, "Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition," hlm. 1–11, Apr 2018, [Daring]. Tersedia pada: http://arxiv.org/abs/1804.03209
[17] M. Lech, M. Stolar, R. Bolia, dan M. Skinner, "Amplitude-Frequency Analysis of Emotional Speech Using Transfer Learning and Classification of Spectrogram Images," Advances in Science, Technology and Engineering Systems Journal, vol. 3, no. 4, hlm. 363–371, Agu 2018, doi: 10.25046/aj030437.
[18] A. Sagheer dan M. Kotb, "Time series forecasting of petroleum production using deep LSTM recurrent networks," Neurocomputing, vol. 323, hlm. 203–213, Jan 2019, doi: 10.1016/j.neucom.2018.09.082.
[19] R. Sari, K. Kusrini, T. Hidayat, dan T. Orphanoudakis, "Improved LSTM Method of Predicting Cryptocurrency Price Using Short-Term Data," IJCCS (Indonesian Journal of Computing and Cybernetics Systems), vol. 17, no. 1, hlm. 33, Feb 2023, doi: 10.22146/ijccs.80776.
[20] I. Lezhenin, N. Bogach, dan E. Pyshkin, "Urban Sound Classification using Long Short-Term Memory Neural Network," Sep 2019, hlm. 57–60. doi: 10.15439/2019F185.
[21] A. Yadav, C. K. Jha, dan A. Sharan, "Optimizing LSTM for time series prediction in Indian stock market," Procedia Comput Sci, vol. 167, hlm. 2091–2100, 2020, doi: 10.1016/j.procs.2020.03.257.
[22] D. Singh dan B. Singh, "Investigating the impact of data normalization on classification performance," Appl Soft Comput, vol. 97, hlm. 105524, Des 2020, doi: 10.1016/j.asoc.2019.105524.
[23] B. Jabir dan N. Falih, "Dropout, a basic and effective regularization method for a deep learning model: a case study," Indonesian Journal of Electrical Engineering and Computer Science, vol. 24, no. 2, hlm. 1009, Nov 2021, doi: 10.11591/ijeecs.v24.i2.pp1009-1016.

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 5, Nomor 2, Desember 2023
e-ISSN 2685-998X
DOI 10.30865/json.v5i2.6950

[24] N. Bacanin, R. Stoean, M. Zivkovic, A. Petrovic, T. A. Rashid, dan T. Bezdan, "Performance of a Novel Chaotic Firefly Algorithm with Enhanced Exploration for Tackling Global Optimization Problems: Application for Dropout Regularization," Mathematics, vol. 9, no. 21, hlm. 2705, Okt 2021, doi: 10.3390/math9212705.
[25] Y. Ho dan S. Wookey, "The Real-World-Weight Cross-Entropy Loss Function: Modeling the Costs of Mislabeling," IEEE Access, vol. 8, hlm. 4806–4813, 2020, doi: 10.1109/ACCESS.2019.2962617.
[26] J. Terven, D. M. Cordova-Esparza, A. Ramirez-Pedraza, dan E. A. Chavez-Urbiola, "Loss Functions and Metrics in Deep Learning," Jul 2023, doi: https://doi.org/10.48550/arXiv.2307.02694.