DECODE: JURNAL PENDIDIKAN TEKNOLOGI INFORMASI
P-ISSN: 2775-2984 | E-ISSN: 2775-1813

Vol. 4, No 3, November 2024, Hal: 809-820
Doi: http://dx.doi.org/10.51454/decode.v4i3.631
https://journal.umkendari.ac.id/index.php/decode

Identifikasi Fitur Suara Menggunakan Model *Convolutional Neural Network* (CNN) pada *Speech-to-Text* (STT)

Rodiah[1], Diana Tri Susetianingtias[2], Eka Patriya[3]*

[1]Program Studi Informatika, Universitas Gunadarma, Indonesia.
[2]Program Studi Sistem Komputer, Universitas Gunadarma, Indonesia.
[3]Program Studi Manajemen, Universitas Gunadarma, Depok, Indonesia

Artikel Info

Kata Kunci:

Convolutional Neural Network;
Kata;
Speech-to-Text;
Suara.

Keywords:

Convolutional Neural Network;
Word;
Speech-to-Text;
Voice.

Riwayat Artikel:

Submitted: 18 Juni 2024
Accepted: 5 Oktober 2024
Published: 5 Oktober 2024

Abstrak: Identifikasi pola ucapan dilakukan untuk dapat mengenali kata yang diucapkan. Salah satu metode yang dapat digunakan untuk mengidentifikasi *Speech-to-Text* (STT) adalah dengan menggunakan *Convolutional Neural Network* (CNN). Penelitian ini menggunakan metode CNN untuk mengidentifikasi STT pada *raw speech dari* sejumlah 23000 data dari *open dataset* suara Kaggle. Tahap awal dilakukan *resampling* durasi, untuk mengambil data rekaman yang memiliki durasi yang cukup untuk masuk dalam proses selanjutnya yaitu inisialisasi frekuensi. Tahap ini mengubah frekuensi asli dari suara rekaman. Inisialisasi dilakukan dengan mengubah frekuensi dari 16000Hz menjadi rentang 8000Hz. Tahap selanjutnya pelabelan data, yaitu data input dan output diberi label untuk klasifikasi sebagai dasar pembelajaran untuk pemrosesan data. Data yang sudah dilabeli kemudian dilakukan pembagian kedalam rasio 8:2. Tahap Akhir Perancangan arsitektur model CNN dilakukan untuk dapat mengenali pola suara yang sudah direkam pada dataset dan dapat mengidentifikasi ucapan. Hasil penelitian bertujuan untuk mengidentifikasi pola suara yang diucapkan dengan akurasi tinggi.

Abstract: *Identification of speech patterns is carried out to be able to recognize the words spoken. One method that can be used to identify Speech-to-Text (STT) is to use a Convolutional Neural Network (CNN). This research uses the CNN method to identify STTs in raw speech from 23,000 data from the Kaggle open voice dataset. The initial stage is Duration Resampling, to retrieve recorded data that has sufficient duration to enter the next process, namely frequency initialization. This stage changes the original frequency of the recorded sound. Initialization is done by changing the frequency from the 16000Hz range to the 8000Hz range. The next stage is data labeling, namely input and output data are labeled for classification as a basis for learning in data processing. The data that has been labeled is then divided in a ratio of 8:2. Final Stage The design of the CNN model architecture is carried out to be able to recognize sound patterns that have been recorded in the dataset and be able to identify speech. The research results aim to identify spoken sound patterns with high accuracy.*

Corresponding Author:
Eka Patriya
Email: ekapatriya@staff.gunadarma.ac.id

PENDAHULUAN

Salah satu cara manusia dalam berkomunikasi yang digunakan sebagai sarana penghubung antara sesama manusia ataupun manusia dengan komputer adalah ucapan. Salah satu cara berkomunikasi antara manusia dengan komputer adalah dengan menggunakan *Speech Recognition* (pengenalan ucapan) (Nagajyothi & Siddaiah, 2018). Sistem *Speech Recognition* dapat diartikan sebagai proses mengubah masukan (*input*) suara ke media lain. *Speech Recognition* disebut juga sebagai *Speech-to-Text* (STT) (Sung et al., 2023) merupakan sistem pengenalan pola suara dibangun oleh beberapa komponen utama diantaranya *microphone* untuk *input* suara, perangkat lunak pengenalan suara, komputer untuk mengambil data suara, dan *soundcard* untuk *input* atau *output* (Pratibha Rashmi & Manu Pratap Singh, 2023). Proses pengenalan ucapan dapat dilakukan dengan menggunakan pola dari kata yang telah diucapkan. Setiap kata yang diucapkan memiliki pola yang saling berbeda. Pada setiap kata yang diucapkan, ada pola yang terbentuk dan memiliki ciri ciri yang bisa dikenali. Berdasarkan pola tersebut, dapat diketahui kata apa yang diucapkan dan terbentuk dari ucapan tersebut. Penggunaan pengenalan ucapan biasa digunakan untuk mengenali suara yang diucapkan oleh manusia agar dapat dikenali oleh komputer.

Salah satu metode yang dapat digunakan untuk mengidentifikasi *Speech-to-Text* adalah dengan menggunakan *Convolutional Neural Network* (CNN). CNN menggunakan teknik *supervised learning* dengan mengenalkan pola yang dibuat dari setiap ucapan (Chandel et al., 2023). Beberapa penelitian terkait implementasi metode CNN pada pengenalan suara dilakukan peneliti sebelumnya. Penelitian (Nagajyothi & Siddaiah, 2018) menggunakan CNN dalam *Automatic Speech Recognition* (ASR) menggunakan data yang didapat dari sistem pada bandara. Peneliti menggunakan filter CNN dengan normalisasi data input untuk meningkatkan kecepatan latih data. Step akhir dari metode ini menggunakan max pooling pada akhir layer, tetapi hasil dari penelitian ini tidak menyebutkan angka dari tingkat akurasi penggunaan metode ini. Penelitian (Kubanek et al., 2019) menggunakan CNN untuk mengenali *continuous speech* dan *isolated words* menggunakan tiga oeprasi konvolusi dari tiga teknik (*Mel Frequency Cepstral Coefficients* (MFCC), waktu dan spektrum). Hasil terbaik didapatkan untuk pengambilan kata terpisah yang mencapai hasil 98%, sedangkan untuk *continuous speech* memperoleh *recognition rate* yang rendah.

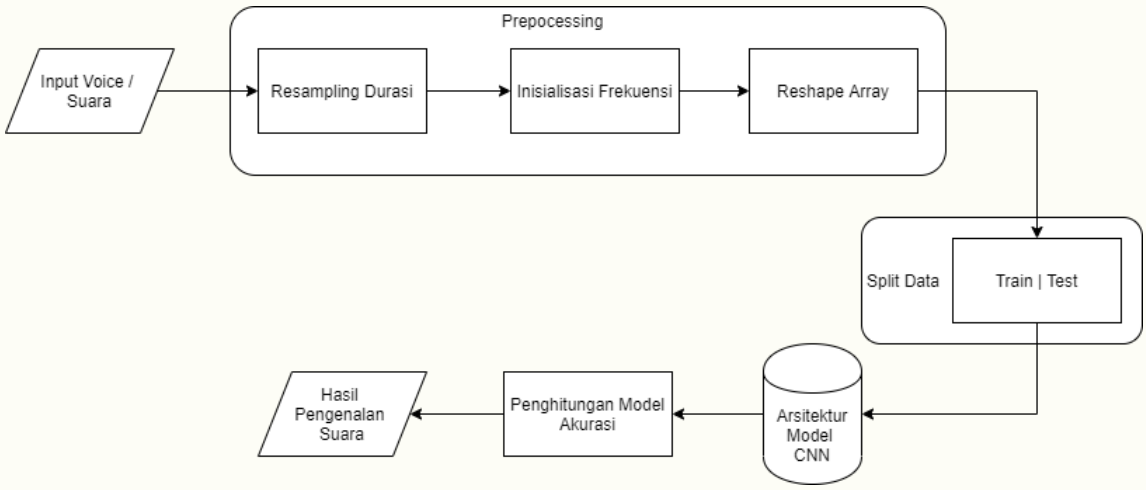
Penelitian (Arul Edwin Raj et al., 2023) menggunakan *Deep Belief Networks* (DBN) pada *speech* recognition meenggunakan *Mel-Frequency Cepstral Coefficient* (MFCC) sebagai *feature extraction* dan *dataset* TIMIT *acoustic-phonetic corpus* untuk mengevaluasi performa dari *deep learning* ini. Penelitian ini menggunakan DBN dalam *speech recognition* dengan *word error rate* yang didapatkan dengan 3 *hidden layers* dan 2048 *hidden units per layer* adalah 24%. Penelitian (Kaur et al., 2018) menggunakan 200 kata sebagai *dataset* yang diambil dari 2 perempuan dan 2 laki-laki dengan umur 27-34 tahun. Setelah pengambilan *speech signal*, lalu dilakukan *pre-processing* dari *speech signal* dengan cara penghapusan *speech signal, pre-emphasis, framing* dan *windowing*. Penelitian ini menggunakan *Mel-Frequency Cepstral Centrum* (MFCC) untuk *feature extraction* yang kemudian dilatih dengan *Deep Neural Network* (DNN) dengan akurasi yang didapatkan sejumlah 97%. Penelitian (Kheddar et al., 2024) membandingkan metode DNN untuk penggunaan *speech recognition* dan berbagai *datasets* yang digunakan dengan penerapan yang dilakukan untuk mendapatkan peningkatan performa dengan perbandingan *words error rate*. Secara umum tahap pengenalan suara dibagi menjadi dua bagian, yakni tahap pembelajaran pola dan tahap pengenalan suara melalui perbandingan pola. Tahap awal adalah dengan ekstraksi ciri fitur suara. Bagian ini merupakan proses mendapatkan sederetan besaran pada bagian sinyal masukan untuk menetapkan pola pembelajaran atau pola uji (Ibrahim et al., 2022). Tahap kedua merupakan tahap pembelajaran Pola Satu atau lebih. Pola uji yang berhubungan dengan bunyi suara dari kelas yang sama, digunakan untuk membuat pola representatif dari ciri-ciri kelas tersebut. Hasilnya yang biasa disebut dengan pola referensi, dapat menjadi sebuah model yang mempunyai karakteristik bentuk statistik dari ciri-ciri pola referensi (Jha et al., 2022).

Pada penelitian ini akan diimplementasikan metode CNN dalam mengidentifikasi *Speech Recogniton* pada raw speech dataset yang merupakan *open dataset* Kaggle (Huy, 2023) sejumlah 23000 file suara dengan format .wav. *Preprocessing* pada dataset suara dilakukan dengan *resampling* durasi

rekaman suara, inisialisasi frekuensi, labeling dataset dan reshape array. Pola suara yang sudah terbentuk pada tahapan preprocessing kemudian dibagi menjadi *training set* dan *validation set* dengan rasio 8 : 2 dimana 80% citra sebagai training set dan 20% sebagai validation set (Kubanek et al., 2019) menggunakan model CNN. Penelitian ini melakukan resampling dari frekuensi 16000 Hz menjadi 8000 Hz yang merupakan frekuensi suara yang dapat didengar manusia. Pada penelitian(Ibrahim et al., 2022) melewati tahapan melabeli data yang dilakukan pada penelitian ini. Pelabelan data merupakan bagian penting dari *preprocessing* data untuk *deep learning*, khususnya pada aristektur *supervised learning*. Tahapan ini memberikan dasar pembelajaran untuk pemrosesan data kedepannya, baik pada data input dan output dalam melakukan klasifikasi. Keluaran dari proses ini menghasilkan model CNN yang sudah dilatih lalu dapat digunakan untuk pengenalan ucapan. Model CNN keluaran tersebut pada *validation set* untuk mengevaluasi kinerja dari model. Akurasi identifikasi *Speech-to-Text* pada raw speech dihitung menggunakan *cross entropy*. Hasil penelitian ini diharapkan dapat mengindentifikasi *Speech-to-Text* pada *raw speech* yang dapat diimplementasikan ke beberapa aplikasi keamanan dengan suara sebagai autentikasinya.

METODE

Penelitian ini melakukan beberapa tahapan dalam pembuatan pengenalan ucapan seperti dapat dilihat pada Gambar 1. Kumpulan suara yang digunakan pada penelitian ini didapatkan dari *dataset* rekaman suara (Huy, 2023) yang merupakan kumpulan dari rekaman suara dari berbagai kata yang ada dengan format berupa *wav.*



Gambar 1. Metode Penelitian *Speech-to-Text* pada *Raw Speech*

Tahap pertama yang dilakukan adalah dengan melakukan *preprocessing* terhadap pola rekaman suara yang ada pada *dataset*. Kumpulan dari pola suara yang telah masuk pada tahap *preprocessing* kemudian dibagi menjadi dua jenis yaitu *training set* untuk digunakan pada pelatihan model CNN dan *validation set* untuk keperluan perbandingan agar tidak terjadi *overfitting*, uji coba identifikasi, dan evaluasi kinerja model. Pembagian menggunakan rasio 8 : 2 dimana 80% citra sebagai *training set* dan 20% sebagai *validation set*. Setelah itu dilakukan perancangan arsitektur dari model CNN yang akan digunakan untuk pengenalan ucapan yang dibuat. Model yang sudah dirancang kemudian dilatih dengan *training set*. Keluaran dari proses ini akan menghasilkan model CNN yang sudah dilatih lalu dapat digunakan untuk pengenalan ucapan. Model CNN keluaran tersebut pada *validation set* untuk mengevaluasi kinerja dari model.

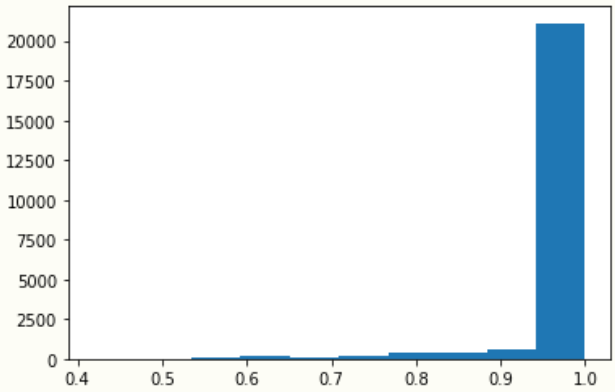
Preprocessing

Tahap awal pengolahan data yang dilakukan terhadap pola suara yang sudah tersedia pada *dataset*. Dari kumpulan rekaman yang ada pada *dataset*, suara yang ada diolah untuk dapat masuk ke

dalam model arsitektur CNN. Tahapan – tahapan yang dilakukan pada saat *preprocessing* adalah sebagai berikut:

1. Resampling durasi rekaman suara

Proses *resampling* durasi ini merupakan proses dimana mengambil data rekaman yang memiliki durasi yang cukup untuk masuk dalam proses selanjutnya. Tahap ini dilakukan agar semua rekaman memiliki pola dengan durasi maksimal 1 detik dan frekuensi dari 16000 Hz diubah menjadi 8000 Hz. Diagram jumlah rekaman dengan durasinya dapat dilihat pada Gambar 2.

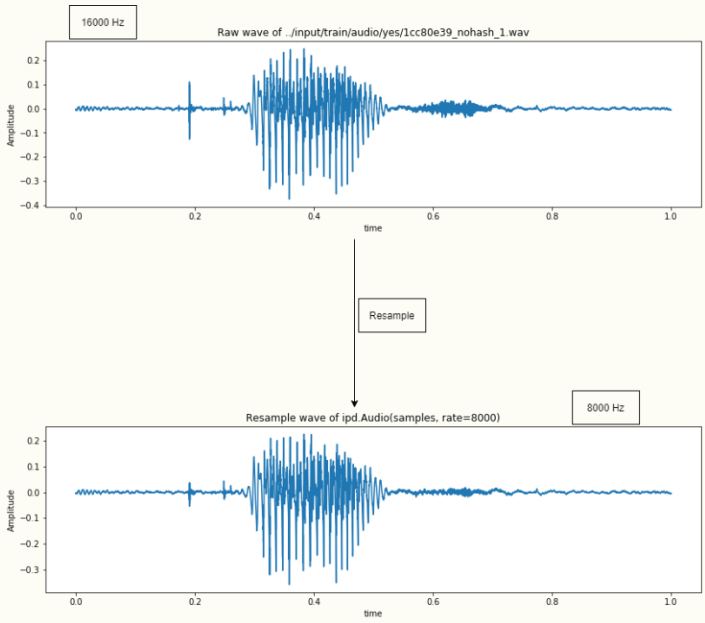


Gambar 2. Diagram Jumlah Rekaman Dengan Durasi

Seperti dapat dilihat pada diagram Gambar 2, pada sumbu *x* merupakan durasi dari rekaman dengan maksimal durasi per suara adalah 1 detik. Pada sumbu *y* merupakan jumlah dari rekaman pada *dataset* sejumlah 20000 suara. Rekaman suara yang memiliki durasi yang cukup, lalu masuk pada tahap *preprocessing* selanjutnya yaitu inisialisasi frekuensi.

2. Inisialisasi Frekuensi

Tahap inisialisasi frekuensi merupakan tahap untuk mengubah frekuensi asli dari suara rekaman. Rekaman suara yang ada pada memiliki frekuensi di 16000Hz. Frekuensi dari bicara manusia ada pada rentang 8000Hz (Monson et al., 2014) maka *sample* yang ada pada *dataset* ini diubah menjadi pada frekuensi tersebut. Pada *raw speech* dengan kata "*yes*" memiliki frekuensi 16000 Hz dilakukan perubahan menjadi 8000 Hz seperti dapat dilihat pada ilustrasi Gambar 3.



Gambar 3. Visualisasi Proses *Resampling* Kata "*Yes*"

Pseudocode berikut merupakan perulangan untuk me-resample frekuensi rekaman suara yang ada pada dataset yang sebelumnya menggunakan sample rate 16000Hz menjadi 8000Hz.

```
for label in labels:
    print(label)
    waves = [f for f in
os.listdir(train_audio_path + '/'+ label) if
f.endswith('.wav')]
    for wav in waves:
        samples, sample_rate =
librosa.load(train_audio_path + '/' + label + '/'
+ wav, sr = 16000)
        samples = librosa.resample(samples,
sample_rate, 8000)
        if(len(samples)== 8000) :
            all_wave.append(samples)
```

Pada baris pertama menggunakan perulangan untuk membaca seluruh data yang ada pada dataset dengan format wav. Perulangan kedua membaca dataset suara dengan sample awal dan diubah menjadi 8000Hz. Setiap data yang sudah di resample dicetak setiap labelnya.

3. *Labeling Dataset* dan *Reshape Array*

Pelabelan data adalah bagian penting dari *preprocessing* data untuk *deep learning*, khususnya untuk *supervised learning*, dimana data *input* dan *output* diberi label untuk klasifikasi untuk memberikan dasar pembelajaran untuk pemrosesan data kedepannya. *Dataset* yang sudah diberi label dan dimasukan ke dalam folder yang sama setiap rekaman suaranya, lalu dikonversi kedalam pengkodean bilangan *integer*. Proses *labeling dataset* dan *reshape array* pada penelitian ini menggunakan *Pseudocode* berikut:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y=le.fit_transform(all_label)
classes= list(le.classes_)
from keras.utils.np_utils import to_categorical
y=to_categorical(y, num_classes=len(labels))
```

Pseudocode tersebut meng*import* fungsi *label encoder* (Yao et al., 2022) yang disediakan untuk *preprocessing* dari *sklearn.* Pengkodean bilangan *integer* tersebut lalu dikonversikan ke dalam *one-hot vector* karena masalah yang diselesaikan *categorical variable*. Langkah terakhir dalam *preprocessing* adalah mengubah *array* yang ada pada data *preprocessing* yang masuk kedalam model CNN ke dalam *array* 3D.

```
all_wave = np.array(all_wave).reshape(-1,8000,1)
print (all_wave)
```

Dalam *pseudocode* tersebut, merupakan *pseudocode* untuk melakukan perubahan data *array* karena *array* yang dihasilkan sebelumnya adalah *array* 2D sedangkan *array* yang bisa masuk ke dalam model *Conv* 1D hanya berupa *array* 3D.

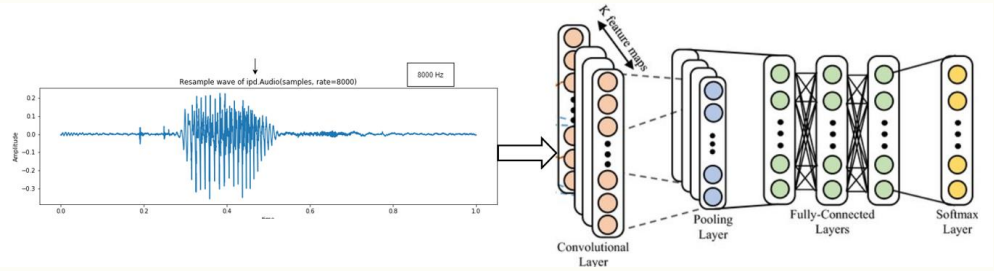
4. Pembagian dataset (*Splitting* Dataset)

Kumpulan dari *dataset* rekaman suara ini nantinya diacak dan dibagi dengan rasio 8:2 untuk digunakan sebagai *training set* dan *validation set*. Data yang digunakan untuk *training* berisi 80% dari total *dataset* yang ada. Dari total *dataset* yang tersedia ada sebanyak 23000 data, namun jumlah yang

akan masuk kedalam data training adalah sebanyak 16550 data, karena data sudah dilakukan *preprocessing*, ada sebagian data yang hilang karena tidak masuk dalam kriteria untuk masuk pada model arsitektur CNN. Data *testing* digunakan untuk mencoba dan mengurus model dari arsitektur yang telah dibuat. *Testing* dilakukan dengan maksud untuk menguji hasil *training* apakah sudah akurat dan layak digunakan. Data yang masuk pada *testing set*, merupakan data yang tidak berlabel. Jumlah data yang ada pada *testing set* sesuai rasio dari awal saat menentukan *split data*. Jumlah dari data untuk *testing set* adalah sebanyak 20% dari total *dataset* yaitu 23000 yang seharusnya adalah 4600 data. Namun saat pemilihan rekaman yang memenuhi durasi hanya ada 4138 data. Data tersebut digunakan untuk menghitung akurasi dari arsitektur model yang telah dibuat. Pada *training set* nanti data juga dilakukan perulangan pelatihan atau *epoch* sebanyak 20 kali untuk pelatihan dengan setiap *batch* berisi 32 data.

Arsitektur Model *Speech-to-Text* dengan CNN

Perancangan arsitektur model CNN dilakukan untuk dapat mengenali pola suara yang sudah direkam pada *dataset* dan dapat mengidentifikasi apa yang diucapkan. Model yang dibuat dirancang dengan menggunakan *framework tensorflow* yang berfungsi sebagai *backend engine* dan *library* keras yang berfungsi sebagai *high-level neural networks* API. Proses pembentukan model *neural network* pada pembuatan pengenalan ucapan ini dilakukan dengan cara mencoba nilai dan model hingga model yang dibuat sudah dirasa cukup baik dalam hal akurasi dan nilai *loss* yang tidak terlalu tinggi. Secara umum arsitektur model *Speech-to-Text* pada penelitian ini dapat dilihat pada Gambar 4.



Gambar 4. Arsitektur model *Speech-to-Text* dengan CNN

Pendefinisian Modul dan Library

Dalam tahap perancangan pembuatan model, terlebih dahulu mempersiapkan *library* dan modul yang digunakan dalam pembentukan model. Modul dari model yang bersifat sekuensial agar proses pembentukan modelnya terurut. Modul *layer*s berisi *Dense, Flatten, Conv1D, Input, maxPooling1D* serta model. Modul ini menggunakan *optimizer* adam untuk proses optimasi untuk menurunkan nilai *loss* dan memaksimalkan hasil akurasi model. Modul *callbacks* berupa *modelcheckpoint* dan *earlystopping* untuk menangani kasus pelatihan data yang tidak stabil (Wang et al., 2024). Semua modul tersebut merupakan *back-end* yang disediakan oleh keras untuk pembuatan arsitektur model pengenalan ucapan

```
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from keras.layers import Conv1D
from keras.layers import Input
from keras.layers import MaxPooling1D
from keras.models import Model
from keras.callbacks import EarlyStopping
from keras.callbacks import ModelCheckpoint
from keras import backend as K
```

Pseudocode ini digunakan untuk mengambil *library* yang disediakan oleh keras untuk digunakan pada pembuatan arsitektur model CNN. *Layer Dense* berfungsi untuk menjalankan *full connection neural network. Dense* juga berfungsi untuk menambahkan *layer* yang *fully connected. Layer Flatten* digunakan untuk membentuk ulang fitur (*reshape feature map*) menjadi sebuah *vector* agar bisa kita gunakan sebagai

input dari *fully-connected layer* (Purwono et al., 2022). Selanjutnya adalah *layer pooling* fungsi dari *pooling* ini adalah untuk mereduksi *input* secara spasial (mengurangi an neuron yang berupa *hidden* mjumlah parameter) dengan operasi *down-sampling. Stride* adalah parameter yang menentukan berapa jumlah pergeseran *filter*, jika nilai stride adalah 1, maka *conv. filter* bergeser sebanyak 1 *pixel* secara *horizontal* lalu *vertical*. Selanjutnya *padding*, digunakan untuk mengukur dimensi *output* agar tetap sama seperti dimensi *input* atau setidaknya tidak berkurang secara drastis (Diep et al., 2024). *Dropout* merupakan proses mencegah terjadinya *overfitting* dan juga mempercepat proses *learning*. *Dropout* mengacu kepada menghilangkapun *layer* yang *visible* di dalam jaringan. Dengan menghilangkan suatu neuron, berarti menghilangkannya sementara dari jaringan yang ada.

Parameter Arsitektur CNN

Model arsitektur CNN yang dirancang untuk pengenalan ucapan ini terdapat beberapa lapisan. Lapisan konvolusi yang terdapat dalam model ini berjumlah 4 lapisan. Diantara masing-masing lapisan terdapat lapisan *pooling* dengan jenis *max pooling*. Lapisan ini digunakan untuk mereduksi dimensi dari *feature map* sehingga dapat mempercepat proses komputasi karena parameter yang di *update* semakin berkurang jumlahnya (Dhar et al., 2023). Setelah itu ada juga lapisan *dropout* digunakan untuk meminimalisir hasil *overfitting* pada pembuatan model arsitektur CNN ini. Hasil parameter arsitektur CNN untuk pengenalan ucapan dapat dilihat pada Tabel 1.

Tabel 1. Parameter CNN untuk Model *Speech-to-Text*

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 8000, 1)	0
conv1d_1 (Conv1D)	(None, 7988, 8)	112
max_pooling1d_1 (MaxPooling1	(None, 2662, 8)	0
dropout_1 (Dropout)	(None, 2662, 8)	0
conv1d_2 (Conv1D)	(None, 2652, 16)	1424
max_pooling1d_2 (MaxPooling1	(None, 884, 16)	0
dropout_2 (Dropout)	(None, 884, 16)	0
conv1d_3 (Conv1D)	(None, 876, 32)	4640
max_pooling1d_3 (MaxPooling1	(None, 292, 32)	0
dropout_3 (Dropout)	(None, 292, 32)	0
conv1d_4 (Conv1D)	(None, 286, 64)	14400
max_pooling1d_4 (MaxPooling1	(None, 95, 64)	0
dropout_4 (Dropout)	(None, 95, 64)	0
flatten_1 (Flatten)	(None, 6080)	0
dense_1 (Dense)	(None, 256)	1556736
dropout_5 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 128)	32896
dropout_6 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 10)	1290
Total params: 1,611,498		
Trainable params: 1,611,498		
Non-trainable params: 0		

Jumlah parameter dari setiap lapisan memiliki perhitungan seperti pada Persamaan 1 (Sung et al., 2023) :

$$parameter = (F * c + 1) * f \quad (1)$$

Persamaan 1 menunjukan cara perhitungan parameter pada lapisan yang ditentukan oleh dengan F merupakan *kernel filter*, *c* adalah jumlah *channel* dan f kecil merupakan *filter size* yang digunakan untuk membangun lapisan arsitektur model. Lapisan awal pada lapisan konvolusi menggunakan *kernel filter* dengan jumlah 13 dengan *channel input* bernilai 1 dan *filter size* dengan jumlah 8. Pada *dense layer* kedua, parameter yang didapatkan yaitu 32.896 selanjutnya yaitu pada *layer dense* ketiga parameter yang didapatkan adalah 1.290 Total parameter yang didapatkan dari model yang dibuat adalah sebanyak 1.611.498. Perhitungan setiap bobot dari masing-masing lapisan dapat dihitung dari jumlah *input, kernel filter, stride* dan *padding* yang digunakan pada model di setiap lapisannya. Bobot atau *weight output* dapat dihitung dengan perhitungan yang dapat dilihat pada Persamaan 2.

$$weight = \frac{W-F+2P}{S} + 1 \qquad (2)$$

Persamaan 2 menunjukan perhitungan *Weight* sebagai hasil bobot, W merupakan jumlah *input, F* merupakan *kernel filter, P* merupakan *Padding,* dan *S* merupakan *Stride.* Sebagai contoh perhitungan bobot dari lapisan konvolusi pertama menggunakan *input* dengan jumlah 8000, *kernel filter* yang digunakan berjumlah 13, nilai *stride* 1, dan nilai *padding* 0 menghasilkan $weight = 7988$.

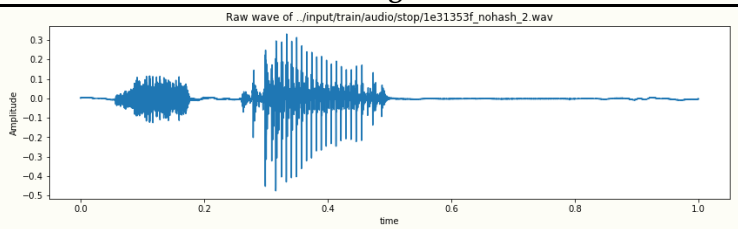
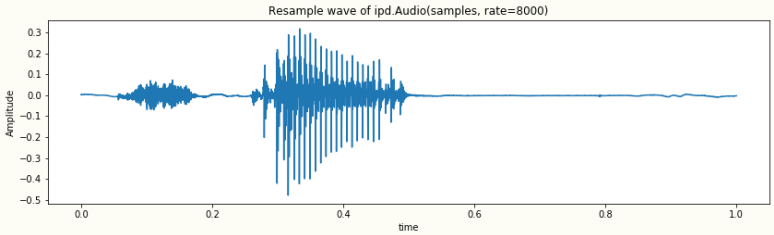
HASIL DAN PEMBAHASAN

Jumlah dari data untuk *testing set* adalah sebanyak 20% dari total *dataset* yaitu 23000 yang seharusnya adalah 4600 data suara, namun pada saat pemilihan rekaman yang memenuhi durasi hanya ada 4138 data suara. Uji coba dilakukan pada beberapa rekaman suara yang peneliti pilih secara acak, dan evaluasi model dilakukan pada *validation set* yang berjumlah 4138 jumlah rekaman suara yang terdiri dari 10 kata.

Hasil *Preprocessing*

Pada penelitian ini, tahap *preprocessing* terdiri dari proses *resample* pada rekaman suara yang ada pada *dataset*. Proses *resample* merupakan pemilihan rekaman dan *resample* frekuensi dari rekaman. Tahap *preprocessing* ini dilakukan agar semua rekaman memiliki pola dengan durasi 1 detik dan frekuensi dari 16000 Hz diubah menjadi 8000 Hz. Suara yang ada dari *dataset* dengan frekuensi yang diubah. Contoh Hasil *preprocessing* pada kata "Stop" dapat dilihat pada Tabel 2.

Tabel 2. Hasil *Preprocessing* Pola Suara

Kata	Gelombang Suara
Stop	 Sebelum *Preprocessing*
Stop	 Hasil *Preprocessing*

Hasil Pelatihan Model CNN

Pelatihan dilakukan sebanyak 20 *epoch* dengan peneliti mencatat performa pelatihan pada tiap *epoch*. Tabel 3 menunjukkan hasil pelatihan pada setiap *epoch* beserta dengan *epoch* terbaik yang diambil pada penelitian ini dengan total kurun waktu sekitar 50 menit.

Tabel 3. Hasil Pelatihan Model CNN

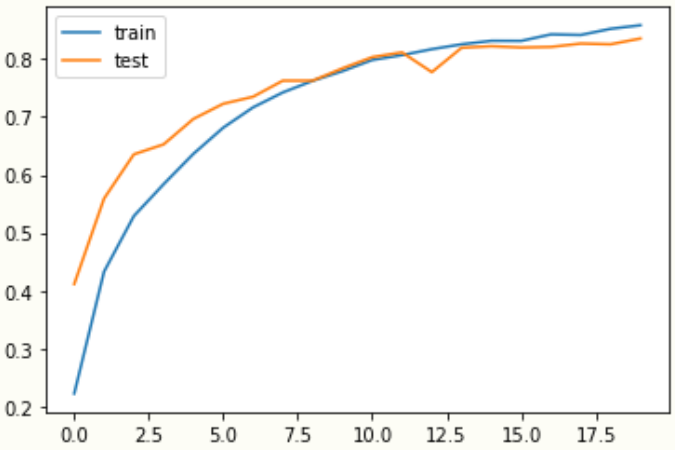
Epoch	*Train Accuracy*	*Validation Accuracy*	*Train Loss*	*Validation Loss*
1	0.2225	0.4116	2.0547	1.5775
2	0.4328	0.5590	1.5200	1.2606
3	0.5288	0.6353	1.2919	1.0960
4	0.5838	0.6525	1.1581	1.0288
5	0.6362	0.6965	1.0244	0.9096
6	0.6813	0.7226	0.9289	0.8147
7	0.7163	0.7347	0.8250	0.7754
8	0.7421	0.7624	0.7493	0.7062
9	0.7622	0.7624	0.6905	0.7125
10	0.7789	0.7837	0.6427	0.6454
11	0.7982	0.8030	0.5866	0.5865
12	0.8065	0.8113	0.5645	0.5655
13	0.8168	0.7772	0.5345	0.6533
14	0.8253	0.8192	0.5134	0.5571
15	0.8312	0.8219	0.4851	0.5246
16	0.8311	0.8197	0.4849	0.5358
17	0.8424	0.8207	0.4548	0.5380
18	0.8413	0.8265	0.4563	0.5437
19	0.8520	0.8253	0.4349	0.5413
20	0.8581	0.8352	0.4078	0.4947

Berdasarkan Tabel 3 didapat bahwa dari seluruh hasil pelatihan pada model CNN terdapat nilai akurasi yang merupakan hasil terbaik dari keseluruhan pelatihan yang sudah dilakukan. Hasil tersebut diambil dengan melihat akurasi terbaik dari *testing accuracy,* seperti dapat dilihat pada Tabel 4.

Tabel 4. Nilai Hasil Akurasi Pelatihan CNN

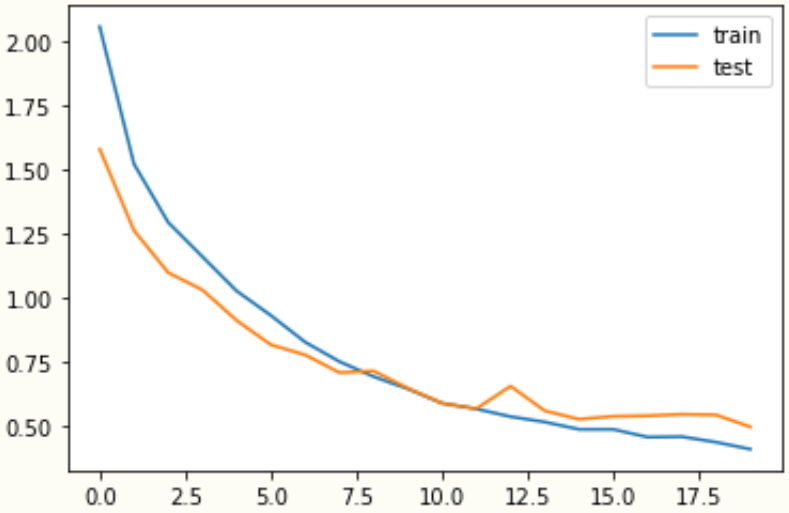
	Accuracy	*Loss*
Training Set	0.8581	0.4078
Testing Set	0.8352	0.4947

Tabel 4 menunjukan bahwa pelatihan model CNN yang dilakukan mendapatkan hasil yang maksimal pada *epoch* ke 20 dengan nilai akurasi *testing* yang mendapatkan 83%. Hasil yang didapatkan pada model pelatihan CNN ini termasuk pada kategori klasifikasi model yang baik karena mencapai rentang 80%– 90%. Dalam *testing* model tidak mengalami proses *overfitting* maupun *underfitting*. Diagram untuk akurasi model CNN dapat dilihat pada Gambar 5 sebagai visualisasi hasil akurasi yang didapatkan pada model ini.



Gambar 5. Grafik Nilai Akurasi Model

Pergerakan dari grafik akurasi dari model *testing set* terdapat penurunan nilai yang tidak terbilang jauh pada iterasi ke-12 namun selanjutnya tetap mengalami kenaikan nilai dari akurasi. Hasil grafik *loss* juga bisa dilihat untuk memvisualisasikan kerugian dari model yang telah dilatih dapat dilihat pada Gambar 6.



Gambar 6. Grafik Nilai *Loss* Model

Gambar 6 menunjukan bahwa penurunan nilai *loss* dari *training set* mengalami penurunan yang stabil dibandingkan dengan grafik nilai *loss* untuk *testing set*. Hasil dari nilai *loss* yang tidak terlalu kecil juga didapatkan karena nilai *loss* yang didapatkan masih di angka 0.4947 untuk *testing set*.

Hasil Uji Coba Pengenalan Ucapan

Uji coba model yang dihasilkan dan telah melewati *validation test* untuk melihat apakah model bisa mengidentifikasi kata yang diucapkan dan bukan merupakan bagian dari *dataset.* Penelitian ini menggunakan rekaman suara untuk melihat kata yang diucapkan dapat diidentifikasi sesuai dengan pola dari suara yang telah dilatih dalam model. Ujicoba dilakukan dengan mengucapkan 10 kata dari dataset yang nantinya diidentifikasi oleh model dan menghasilkan teks yang sesuai dengan ucapan yang diucapkan oleh penulis. Hasil lengkap dapat bisa dilihat pada Tabel 4.

Tabel 4. Hasil Uji Pengenalan Ucapan

Uji Coba ke-	Kata Ucapan	Hasil Identifikasi
1	*Yes*	*Yes*
2	*No*	*No*
3	*Up*	*Up*
4	*Down*	*Down*
5	*Left*	*Left*
6	*Right*	*Right*
7	*On*	*On*
8	*Off*	*Off*
9	*Stop*	*Stop*
10	*Go*	*Go*

Tabel 4 menunjukan hasil dari pengenalan ucapan, semua ucapan yang diidentifikasi sesuai dengan kata yang diucapkan. Model yang dibuat berhasil mengidentifikasi kata-kata sesuai dengan hasil yang baik. dari 10 kata yang diucapkan, tidak ada kesalahan dalam mengidentifikasi ucapan. Meski akurasi yang dihasilkan oleh model berkisar 83% namun hasil dari percobaan semua berhasil dilakukan dengan benar dan saat uji coba menggunakan data baru akurasi yang didapatkan 100%.

KESIMPULAN

Berdasarkan hasil uji coba identifikasi pengenalan ucapan dilakukan dengan model CNN dapat diambil beberapa kesimpulan. Model CNN berhasil dibentuk dengan jumlah 18 *hidden layer*, terdiri dari 4 lapisan konvolusi, 4 lapisan *max pooling*, 6 lapisan *dropout,* 3 lapisan *dense,* dan 1 lapisan *flatten*. Model menerima masukan untuk dilatih berupa data suara dengan frekuensi 8000 Hz dan menghasilkan keluaran berupa prediksi kata ucapan dari pola suara yang dimasukan. Total parameter bobot yang dimiliki oleh model berjumlah 1.611.498 variabel. Pelatihan model berhasil membentuk *training set* dan *validation set*, dengan rasio sebesar 8:2 dimana 80% (16550 rekaman) masuk ke dalam *training set* dan 20% (4138 rekaman) masuk ke dalam *validation set*. Pelatihan dilakukan sebanyak 20 iterasi (*epoch*) dan diambil *epoch* dengan akurasi pada *validation set* terbaik, yaitu pada *epoch* ke-20. Nilai akurasi yang dihasilkan dari model untuk mengidentifikasi ucapan dari rekaman yang tersedia pada *validation set* mencapai 83.52% atau dengan kata lain, dari 4138 jumlah rekaman suara yang ada pada *validation set model* dapat memprediksi sebanyak 3456 suara berhasil. Berdasarkan hasil penelitian yang dilakukan, dapat diberikan beberapa saran antara lain penambahan *dataset* yang lebih bervariatif serta implementasi model menggunakan *command line* membuat tampilan yang digunakan dalam percobaan menjadi lebih sulit dipahami, diharapkan bisa membuat *interface* untuk membuat tampilan dari program lebih menarik dan mudah digunakan.

DAFTAR PUSTAKA

Arul Edwin Raj, A., Karan Kumar, B., Shajivan, S., & Rohit, A. (2023). Speech Emotion Recognition using Deep Learning. *International Conference on Innovative Data Communication Technologies and Application, ICIDCA 2023 - Proceedings*, *6*(3), 505–509. https://doi.org/10.1109/ICIDCA56705.2023.10100056

Chandel, G., Matete, E., Nandy, T., Gaur, V., & Kumar Saini, S. (2023). Ambient Sound Recognition using Convolutional Neural Networks. *E3S Web of Conferences*, *405*. https://doi.org/10.1051/e3sconf/202340502017

Dhar, S., Sen, A., Bandyopadhyay, A., Jana, N. D., Ghosh, A., & Sarayloo, Z. (2023). Differential Evolution Algorithm Based Hyper-Parameters Selection of Convolutional Neural Network for Speech Command Recognition. *International Joint Conference on Computational Intelligence*, 315-

322. https://doi.org/10.5220/0012251500003595

Diep, Q. B., Phan, H. Y., & Truong, T. C. (2024). Crossmixed convolutional neural network for digital speech recognition. *PLoS ONE*, *19*(4), 1-22. https://doi.org/10.1371/journal.pone.0302394

Huy, N. B. (2023). *Raw Audio Dataset*. https://www.kaggle.com/datasets/nguyenbahuy/raw-audio-data

Ibrahim, W., Candra, H., & Isyanto, H. (2022). Voice Recognition Security Reliability Analysis Using Deep Learning Convolutional Neural Network Algorithm. *Journal of Electrical Technology UMY*, *6*(1), 1–11. https://doi.org/10.18196/jet.v6i1.14281

Jha, A., Gupta, S., Dubey, P., & Chhabria, A. (2022). Music Feature Extraction And Recommendation Using CNN Algorithm. *ITM Web of Conferences*, *44*, 03026. https://doi.org/10.1051/itmconf/20224403026

Kaur, G., Srivastava, M., & Kumar, A. (2018). Speaker and Speech Recognition using Deep Neural Network. *International Journal of Emerging Research in Management and Technology*, *6*(8), 118. https://doi.org/10.23956/ijermt.v6i8.126

Kheddar, H., Hemis, M., & Himeur, Y. (2024). Automatic speech recognition using advanced deep learning approaches: A survey. *Information Fusion*, *109*. https://doi.org/10.1016/j.inffus.2024.102422

Kubanek, M., Bobulski, J., & Kulawik, J. (2019). A method of speech coding for speech recognition using a convolutional neural network. *Symmetry*, *11*(9), 1-12. https://doi.org/10.3390/sym11091185

Monson, B. B., Hunter, E. J., Lotto, A. J., & Story, B. H. (2014). The perceptual significance of high-frequency energy in the human voice. *Frontiers in Psychology*, *5*(JUN), 1-11. https://doi.org/10.3389/fpsyg.2014.00587

Nagajyothi, D., & Siddaiah, P. (2018). Speech recognition using convolutional neural networks. *International Journal of Engineering and Technology(UAE)*, *7*(4.6 Special Issue 6), 133-137. https://doi.org/10.14419/ijet.v7i4.6.20449

Pratibha Rashmi, & Manu Pratap Singh. (2023). Convolution neural networks with hybrid feature extraction methods for classification of voice sound signals. *World Journal of Advanced Engineering Technology and Sciences*, *8*(2), 110–125. https://doi.org/10.30574/wjaets.2023.8.2.0083

Purwono, Ma'arif, A., Rahmaniar, W., Fathurrahman, H. I. K., Frisky, A. Z. K., & Haq, Q. M. U. (2022). Understanding of Convolutional Neural Network (CNN): A Review. *International Journal of Robotics and Control Systems*, *2*(4), 739-748. https://doi.org/10.31763/ijrcs.v2i4.888

Sung, W. T., Kang, H. W., & Hsiao, S. J. (2023). Speech Recognition via CTC-CNN Model. *Computers, Materials and Continua*, *76*(3), 3833-3858. https://doi.org/10.32604/cmc.2023.040024

Wang, F., Hao, M., Shi, Y., & Xu, B. (2024). Lead ASR Models to Generalize Better Using Approximated Bias-Variance Tradeoff. *Communications in Computer and Information Science*, *1961 CCIS*, 174-185. https://doi.org/10.1007/978-981-99-8126-7_14

Yao, Z., Ren, S., Chen, S., Ma, Z., Guo, P., & Xie, L. (2022). *TESSP: Text-Enhanced Self-Supervised Speech Pre-training*. http://arxiv.org/abs/2211.13443