Jurnal Sistim Informasi dan Teknologi

https://jsisfotek.org/index.php

2023 Vol. 5 No. 2 Hal: 120-130 e-ISSN: 2686-3154

Analisis Pengaruh Kombinasi Fitur Spektral terhadap Tingkat Akurasi Speech Emotion Recognition

Mifta Nur Farid[1]✉, Arya Fatur Rahman[2], Himawan Wicaksono[3]

[1,2,3]Institut Teknologi Kalimantan

miftanurfarid@lecturer.itk.ac.id

Abstrak

Telah banyak penelitian tentang pengenalan emosi suara dengan akurasi yang berbeda-beda. Hal tersebut disebabkan oleh dataset, fitur-fitur, dan model klasifikasi yang digunakan. Hal yang paling mempengaruhi tingkat akurasi pengenalan emosi suara adalah fitur-fitur yang digunakan. Sehingga pada penelitian ini dilakukan eksplorasi terhadap kombinasi kombinasi fitur-fitur spektral dan bagaimana pengaruhnya terhadap akurasi dari pengenalan emosi suara. Kombinasi fitur yang digunakan adalah fitur-fitur spektral low level descriptor yaitu mel-frequency cepstral coefficient (mfcc), chroma, mel-spectrogram, spectral contrast, spectral bandwidth, dan tonnetz, dan fitur-fitur *high-statistical function* (HSF), yaitu *mean*, standar deviasi, jangkauan interkuartil, skewness, dan kurtosis dari fitur-fitur LLD sebelumnya. Model yang digunakan adalah long short-term memory (LSTM). Hasil yang didapatkan adalah dari keseluruhan percobaan kombinasi fitur LLD dan HSF, fitur mfcc dan spectral contrast memberikan nilai akurasi dan UAR tertinggi. Jika fitur mfcc ini dihilangkan maka nilai akurasi dan UAR akan turun secara signifikan. Selain itu penelitian ini memberikan bukti bahwa semakin banyak fitur yang digunakan tidak selalu memberikan hasil akurasi dan UAR yang baik. Namun yang paling mempengaruhi adalah fitur apa yang digunakan, bukan seberapa banyak fitur yang digunakan.

Kata Kunci: Speech Emotion Recognition, LSTM, MFCC.

1. Pendahuluan

Pembelajaran mendalam dan pembelajaran mesin adalah metode paling umum untuk membuat sistem lebih pintar seiring kemajuan teknologi di bidang kecerdasan buatan [1]. Emosi memainkan peran penting dalam menentukan kepuasan pengguna dan opini pelanggan dan merupakan komponen penting dari interaksi manusia [2]. Dalam penelitian pemrosesan sinyal audio digital saat ini, sistem pengenalan ucapan dan emosi atau *speech emotion recognition* (SER) yang cerdas adalah prasyarat mendasar. Banyak aplikasi yang terkait dengan interaksi manusia-komputer atau *human-computer interaction* (HCI) sangat bergantung pada SER. Penting untuk meningkatkan kinerja prediktif dari sistem SER tercanggih saat ini agar dapat digunakan untuk aplikasi komersial waktu nyata [1]. Salah satu teknologi yang sering digunakan dalam implementasi sistem pengenalan suara adalah asisten virtual seperti Alexa, Siri, dan Google Assistant. Teknologi asisten virtual diketahui tidak dapat memahami emosi manusia, yang berarti akan mengurangi kepuasan pengguna saat berkomunikasi dengan orang lain [3].

Pada penelitian SER, umumnya terbagi menjadi dua bagian. Bagian pertama adalah ektraksi fitur dan bagian kedua adalah klasifikasi. Pada ekstraksi fitur, sinyal suara akan dikonversi menjadi beberapa nilai-nilai salah satunya adalah fitur-fitur spektral. Fitur-fitur spektar yang umum digunakan adalah mel-frequency cepstral coefficient (MFCC), linear prediction cepstral coefficients (LPCC), short-time energy, fundamental frequency (F0), formants, [4], [5]. Terdapat juga penelitian yang menggunakan fitu-fitur low-level descriptor (LLD) dan high-statistical function (HSF) [6]. LLD yang dimaksud adalah fitur-fitur spektral sedangkan HSF adalah nilai-nilai HSF dari LLD tersebut. Penelitian yang berkaitan dengan bagian kedua, yaitu klasifikasi, pun bermacam-macam. Mulai dari yang paling sederhana seperti penggunaan *Gaussian mixture model* (GMM) [4], [7], [8], *hidden markov model* (HMM) [9], dan *support vector machine* (SVM) [10]–[12]. Hingga penerapan *neural network* seperti *multi-layer perceptron* (MLP) [8], extreme learning machine (ELM) [13], *convolutional neural network* (CNN) [14], [15], *residual neural networks* (ResNet) [16], dan *recurrent neural networks* (RNN) [6], [17].

Terdapat penelitian yang menggunakan dataset TESS Toronto, fitur Mel-Frecuency Cepstral Coefficients (MFCC), dan model Artificial Neural Network (ANN). Akurasi yang dihasilkan dari penelitian tersebut adalah sebesar 99,71% [18]. Nilai akurasi yang tinggi tersebut didapatkan karena dataset yang digunakan hanya memiliki 2 aktor. Selain itu, penelitian tersebut tidak melakukan proses *speaker independent* dan *cross-validation*. Pada penelitian yang lain, digunakan tiga rangkaian fitur: GeMAPS, pyAudioAnalysis, dan LibROSA; dua jenis fitur: *low-level descriptors* (LLD) dan *high-statistical function* (HSF); dan empat model klasifikasi: *multilayer perceptron* (MLP), *long short-term memory* (LSTM), *gated recurrent unit* (GRU), dan *one-dimentional*

Diterima: 23-06-2023 | Revisi: 26-06-2023 | Diterbitkan: 29-06-2023 | doi: 10.37034/jsisfotek.v5i1.234

convolutional neural network (1-D CNN) [19]. Pada penelitian ini, dilakukan proses *speaker independent* dan *corss-validation*. Hasil yang didapatkan adalah penggunaan model LSTM memberikan akurasi yang paling tinggi sebesar 77,4%. Berdasarkan dua penelitian ini disimpulkan bahwa penerapan *speaker independent* dan *cross-validation* dapat menurunkan nilai akurasi.

Variasi di antara penutur, seperti jenis kelamin, usia, dan faktor emosional lain yang tidak relevan, dapat menghasilkan perbedaan yang signifikan dalam distribusi ciri-ciri emosional, membuat berbicara mandiri menjadi tugas yang sulit . Dengan pendekatan yang sama, SER dilakukan untuk menunjukkan bahwa pembicara independen berdampak pada penelitian sebelumnya [18]. Akurasi tertinggi yang didapatkan adalah 62,31% .

Pada penelitian yang telah disebutkan sebelumnya , akurasi yang didapatkan cukup baik yaitu 77,4% jika dibandingkan dengan yang lainnya. Hal ini dikarenakan fitur yang digunakan adalah fitur kombinasi *low-level descriptor* (LLD) dan *high-level statistic function* (HSF) [5]. Namun, akurasinya lebih rendah yaitu 57% pada penelitian lain yang menggunakan lebih banyak HSF dan LLD [6]. Penelitian lain yang menggunakan metode LTSM saja hanya mencapai nilai akurasi sebesar 54,2% ketika menggunakan kombinasi LLD dan HSF [20]. Oleh sebab itu, pada penelitian ini akan dilakukan analisis terhadap pengaruh dari kombinasi antara LLD dan HSF terhadap performa SER menggunakan model LSTM. Dataset yang digunakan dalam penelitian ini adalah The Ryerson Audio-Visual Database of Emotional Speech and Song Dataset (RAVDESS Dataset) [21].

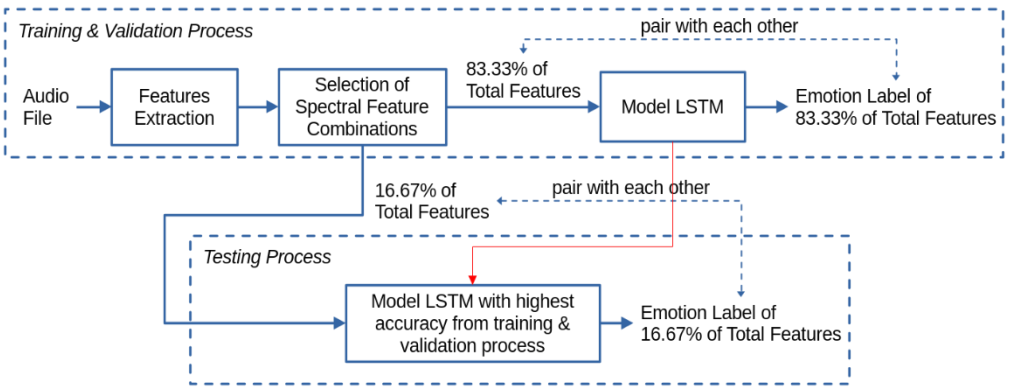
2. Metode Penelitian

Metode penelitian yang dilakukan ditunjukkan pada

Gambar 1. Gambaran umum dari proses yang dilakukan terbagi menjadi dua tahap. Tahapan pertama adalah proses pelatihan dan validasi dan tahapan kedua adalah proses pengujian. Proses pelatihan dan validasi dilakukan untuk mendapatkan model LSTM yang terbaik. Sedangkan proses pengujian dilakukan untuk menguji model yang telah didapatkan tersebut jika diberikan dataset yang berbeda dari proses pelatihan dan validasi.

Audio file yang digunakan pada penelitian ini berasal dari The Ryerson Audio-Visual Database of Emotional Speech and Song Dataset (RAVDESS Dataset) . RAVDESS Dataset yang digunakan pada penelitian ini berjumlah 1440 data rekaman suara dari 12 aktor laki-laki dan 12 aktor perempuan yang mengandung emosi netral, tenang, senang, sedih, marah, takut, jijik, dan terkejut. Label emosi yang terkandung data rekaman suara ditunjukkan dalam nama file yang digunakan sebagaimana yang ditunjukkan dalam dokumentasi RAVDESS Dataset [21].

Data rekaman suara dari RAVDESS akan diektrak fitur-fitur spektralnya menggunakan *library* yang disediakan oleh librosa. Fitur-fitur spektral LLD yang diekstrak dari data rekaman suara adalah mel-frequency cepstral coefficient (mfcc), chroma, mel-spectrogram, spectral contrast, spectral bandwidth, dan tonnetz. Sedangkan fitur-fitur spektral HSF adalah standar deviasi, jangkauan interkuartil (*interquartil range*), *skewness*, dan kurtosis. *Source code* yang digunakan dalam proses ektraksi fitur-fitur, baik LLD maupun HSF, ditunjukkan oleh Algoritma 1. Sebagaimana yang tertulis di dalam *source code* di Algoritma 1, 16.67% dari total fitur-fitur yang diekstrak akan digunakan di proses *training* dan *validation* sedangkan 83.33% dari total fitur-fitur yang diekstrak akan digunakan di proses *testing.*



Gambar 1. Metodologi penelitian

Algoritma 1. Features extraction

```
# Extract all spectral features (LLD & HSF)
import glob
import os
import librosa as lr
import numpy as np
from scipy import stats as sc
```

```
dataset = '../dataset/'
files = glob.glob(os.path.join(dataset + '*/*.wav'))
files.sort()
files[0]
result = '../result'
def extract_features(filename):
    X, sr = lr.load(filename, sr=None)
    stft = np.abs(lr.stft(X))
    # MFCC
    mfcc = np.mean(lr.feature.mfcc(y=X, sr=sr, n_mfcc=40).T, axis=0)
    mfcc_std = np.std(lr.feature.mfcc(y=X, sr=sr, n_mfcc=40).T, axis=0)
    mfcc_rng = sc.iqr(lr.feature.mfcc(y=X, sr=sr, n_mfcc=40).T, axis=0)
    mfcc_skew = sc.skew(lr.feature.mfcc(y=X, sr=sr, n_mfcc=40).T, axis=0)
    mfcc_krt = sc.kurtosis(lr.feature.mfcc(y=X, sr=sr, n_mfcc=40).T, axis=0)
    # Chroma
    chroma = np.mean(lr.feature.chroma_stft(S=stft, sr=sr).T, axis=0)
    chroma_std = np.std(lr.feature.chroma_stft(S=stft, sr=sr).T, axis=0)
    chroma_rng = sc.iqr(lr.feature.chroma_stft(S=stft, sr=sr).T, axis=0)
    chroma_skew = sc.skew(lr.feature.chroma_stft(S=stft, sr=sr).T, axis=0)
    chroma_krt = sc.kurtosis(lr.feature.chroma_stft(S=stft, sr=sr).T, axis=0)
    # Melspectogram
    mel = np.mean(lr.feature.melspectrogram(y=X, sr=sr).T, axis=0)
    mel_std = np.std(lr.feature.melspectrogram(y=X, sr=sr).T, axis=0)
    mel_rng = sc.iqr(lr.feature.melspectrogram(y=X, sr=sr).T, axis=0)
    mel_skew = sc.skew(lr.feature.melspectrogram(y=X, sr=sr).T, axis=0)
    mel_krt = sc.kurtosis(lr.feature.melspectrogram(y=X, sr=sr).T, axis=0)
    # Spectral Contrast
    spc_c = np.mean(lr.feature.spectral_contrast(S=stft, sr=sr).T, axis=0)
    spc_c_std = np.std(lr.feature.spectral_contrast(S=stft, sr=sr).T, axis=0)
    spc_c_rng = sc.iqr(lr.feature.spectral_contrast(S=stft, sr=sr).T, axis=0)
    spc_c_skew = sc.skew(lr.feature.spectral_contrast(S=stft, sr=sr).T, axis=0)
    spc_c_krt = sc.kurtosis(lr.feature.spectral_contrast(S=stft, sr=sr).T, axis=0)
    # Spectral Bandwidth
    spc_b = np.mean(lr.feature.spectral_bandwidth(S=stft, sr=sr).T, axis=0)
    spc_b_std = np.std(lr.feature.spectral_bandwidth(S=stft, sr=sr).T, axis=0)
    spc_b_rng = sc.iqr(lr.feature.spectral_bandwidth(S=stft, sr=sr).T, axis=0)
    spc_b_skew = sc.skew(lr.feature.spectral_bandwidth(S=stft, sr=sr).T, axis=0)
    spc_b_krt = sc.kurtosis(lr.feature.spectral_bandwidth(S=stft, sr=sr).T, axis=0)
    # Tonnetz
    tonnetz = np.mean(lr.feature.tonnetz(y=lr.effects.harmonic(X), sr=sr).T, axis=0)
    tonnetz_std = np.std(lr.feature.tonnetz(y=lr.effects.harmonic(X), sr=sr).T, axis=0)
     tonnetz_rng = sc.iqr(lr.feature.tonnetz(y=lr.effects.harmonic(X), sr=sr).T, axis=0)
     tonnetz_skew = sc.skew(lr.feature.tonnetz(y=lr.effects.harmonic(X), sr=sr).T, axis=0)
    tonnetz_krt = sc.kurtosis(lr.feature.tonnetz(y=lr.effects.harmonic(X), sr=sr).T, axis=0)
    return (mfcc, mfcc_std, mfcc_rng, mfcc_skew, mfcc_krt,
            chroma, chroma_std, chroma_rng, chroma_skew, chroma_krt,
            mel, mel_std, mel_rng, mel_skew, mel_krt,
            spc_c, spc_c_std, spc_c_rng, spc_c_skew, spc_c_krt,
            spc_b, spc_b_std, spc_b_rng, spc_b_skew, spc_b_krt,
            tonnetz, tonnetz_std, tonnetz_rng, tonnetz_skew, tonnetz_krt)

# create empty list to store features and labels
feat_train = []
feat_test = []
lab_train = []
lab_test = []
# iterate over all files
for file in files:
    print("Extracting features from ", file)
    feat_i = np.hstack(extract_features(file))
```

```
    lab_i = os.path.basename(file).split('-')[2]
    # create speaker independent split
    if int(file[-6:-4]) > 20:
        feat_test.append(feat_i)
        lab_test.append(int(lab_i)-1)
    else:
        feat_train.append(feat_i)
        lab_train.append(int(lab_i)-1) # make labels start 0
# save all features as npy files
np.save(result + 'train_x.npy', feat_train)
np.save(result + 'test_x.npy', feat_test)
np.save(result + 'train_y.npy', lab_train)
np.save(result + 'test_y', lab_test)
```

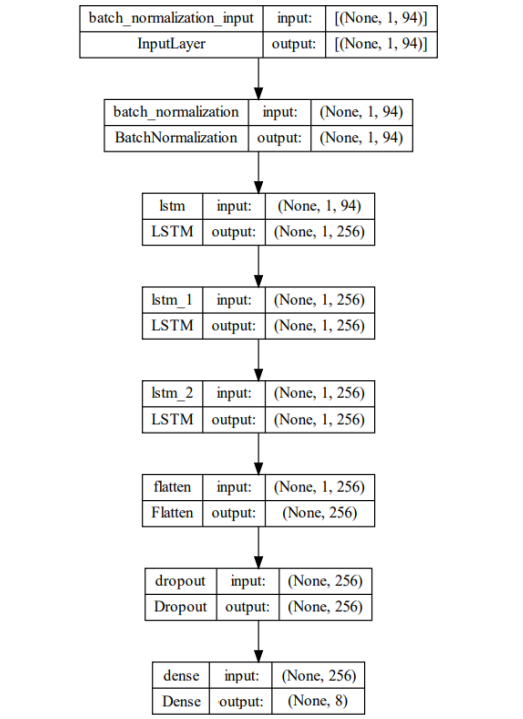
Kombinasi fitur-fitur dilakukan menggunakan sistem percobaan faktorial. Pendekatan ini dimulai dengan mencoba seluruh fitur secara bersamaan. Setelah itu, hasil akurasi dievaluasi, dan jika dinilai kurang memadai, dilanjutkan dengan melakukan kombinasi lain dari fitur yang tersedia. Kombinasi fitur yang dilakukan ditunjukkan oleh Tabel 1.

Tabel 1. Kombinasi dari fitur-fitur spektral yang digunakan

Kombinasi dari 6 fitur	Kombinasi dari 5 Fitur	Kombinasi dari 4 Fitur	Kombinasi dari 3 Fitur	Kombinasi dari 2 Fitur
1. mfcc,chroma,mel,spcc, spcb,tonnetz	1. mfcc,chroma,mel,spcc, spcb 2. mfcc,chroma,mel,spcc, tonnetz 3. mfcc,chroma,mel,spcb, tonnetz 4. mfcc,mel,spcc,spcb,tonnetz 5. mfcc,chroma,spcc,spcb ,tonnetz 6. chroma,mel,spcc,spcb,tonnetz	1. mfcc,chroma,mel,spcc 2. mfcc,chroma,mel,spcb 3. mfcc,chroma,mel,tonnetz 4. mfcc,mel,spcc,spcb 5. mfcc,mel,spcc,tonnetz 6. mfcc,spcc,spcb,tonnetz 7. chroma,mel,spcc,spcb 8. chroma,mel,spcc,tonnetz 9. chroma,mel,spcb,tonnetz 10. chroma,spcc,spcb,tonnetz 11. mel,spcc,spcb,tonnetz	1. mfcc-chroma-mel 2. mfcc-chroma-spcc 3. mfcc-chroma-spcb 4. mfcc-chroma-tonnetz 5. mfcc-mel-spcc 6. mfcc-mel-spcb 7. mfcc-mel-tonnetz 8. mfcc-spcc-spcb 9. mfcc-spcc-tonnetz 10. mfcc-spcb-tonnetz 11. chroma-mel-spcc 12. chroma-mel-spcb 13. chroma-mel-tonnetz 14. chroma-spcc-spcb 15. chroma-spcc-tonnetz 16. chroma-spcb-tonnetz 17. mel-spcc-spcb 18. mel-spcc-tonnetz 19. mel-spcb-tonnetz 20. spcc-spcb-tonnetz	1. mfcc,chroma 2. mfcc,mel 3. mfcc,spcc 4. mfcc,spcb 5. mfcc,tonnetz 6. chroma,mel 7. chroma,spcc 8. chroma,spcb 9. chroma,tonnetz 10. mel,spcc 11. mel,spcb 12. mel,tonnetz 13. spcc,spcb 14. spcc,tonnetz 15. spcb,tonnetz

Model *Long Short Term Memori* (LSTM) yang digunakan sebagai pengenalan emosi suara untuk mendapatkan hasil akurasi ditunjukkan oleh **Error! Reference source not found.**. Hal ini karena model tersebut memiliki hasil terbaik dalam mendapatkan akurasi dari pengenalan emosi suara . Sedangkan *source code* dari model LSTM yang digunakan ditunjukkan oleh Algoritma 2. Selain pembuatan model LSTM, proses training, validation, dan testing juga ditulis dalam *source code* Algoritma 2.

Data input yang masuk melalui input layer akan di-normalize. Hal ini dilakukan untuk membentuk suatu konvergensi data. Kemudian layer-layer LSTM akan dilatih sesuai dengan emosi yang ada sehingga menemukan ciri dari data lain yang sesuai dengan ciri awal yang diingat oleh LSTM. Data yang telah dilatih di layer LSTM akan diproses dengan cara mengurangi koneksi neuron dan membuat sel yang terdiri dari anyak dimensi menjadi satu dimensi. Dilanjutkan dengan pembuatan *fully connected layer* dengan aktivasi *softmax* sehingga data yang sebelumnya berbeda-beda kan menjadi 1 daengan identitas model utuh menggunakan *flatten*. Model tersebut merupakan data pelatihan dari sistem yang ada yang kemudian akan dilakukan pengujian dengan data uji yang telah di siapkan pada tahap input. Hasil dari pengujian dari data pelatihan dan data pengujian tersebut akan dievaluasi sesuai permintaan. Hasil dari evaluasi tersebut akan menghasilkan hasil dari pembelajaran mesin dan memasuki tahap output.



batch_normalization_input | input: | [(None, 1, 94)]
InputLayer | output: | [(None, 1, 94)]
batch_normalization | input: | (None, 1, 94)
BatchNormalization | output: | (None, 1, 94)
lstm | input: | (None, 1, 94)
LSTM | output: | (None, 1, 256)
lstm_1 | input: | (None, 1, 256)
LSTM | output: | (None, 1, 256)
lstm_2 | input: | (None, 1, 256)
LSTM | output: | (None, 1, 256)
flatten | input: | (None, 1, 256)
Flatten | output: | (None, 256)
dropout | input: | (None, 256)
Dropout | output: | (None, 256)
dense | input: | (None, 256)
Dense | output: | (None, 8)

Gambar 2. Model LSTM yang digunakan

Algoritma 2. LSTM model

```
# LSTM Model

import numpy as np
import tensorflow as tf
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import random as rn
import os
from sklearn.metrics import confusion_matrix
import seaborn as sns
import pandas as pd

np.random.seed(123)
rn.seed(123)
tf.random.set_seed(123)
datapath = '../result/'

x_train = np.load(datapath + 'train_x.npy')
x_test = np.load(datapath + 'test_x.npy')
y_train = np.load(datapath + 'train_y.npy')
y_test = np.load(datapath + 'test_y.npy')

# reshape x untuk lstm
x_train = x_train.reshape((x_train.shape[0], 1, x_train.shape[1]))
x_test = x_test.reshape((x_test.shape[0], 1, x_test.shape[1]))

# if labels are not in integer, convert it, otherwise comment it
y_train = y_train.astype(int)
y_test = y_test.astype(int)

earlystop = tf.keras.callbacks.EarlyStopping(monitor='val_loss',patience=10,
restore_best_weights=True)
checkpointer = tf.keras.callbacks.ModelCheckpoint(filepath='/tmp/weights.hdf5', verbose=1,
save_best_only=True)

# define lstm model
def lstm():
    model = tf.keras.models.Sequential()
    model.add(tf.keras.layers.BatchNormalization(axis=-1,
                                                 input_shape=(x_train.shape[1],
                                                              x_train.shape[2])))
    model.add(tf.keras.layers.LSTM(256, return_sequences=True))
    model.add(tf.keras.layers.LSTM(256, return_sequences=True))
    model.add(tf.keras.layers.LSTM(256, return_sequences=True))
    model.add(tf.keras.layers.Flatten())
    model.add(tf.keras.layers.Dropout(0.4))
    model.add(tf.keras.layers.Dense(8, activation='softmax'))

    # compile model: set loss, optimizer, metric
    model.compile(loss=tf.keras.losses.SparseCategoricalCrossentropy(),
                  optimizer=tf.keras.optimizers.Adam(), metrics = ['accuracy'])
    return model
# create model
model = lstm()
print(model.summary())
# plot model
tf.keras.utils.plot_model(model, datapath + 'model_lstm.pdf',show_shapes=True)
```

```
# train the model
hist = model.fit(x_train, y_train, epochs=100, shuffle=True, callbacks=earlystop,
                 validation_split=0.1, batch_size=16)
evaluate = model.evaluate(x_test, y_test, batch_size=16)
```

3. Hasil dan Pembahasan

Hasil dari kombinasi 6, 5, 4, 3 dan 2 fitur, yang mencakup nilai akurasi, UAR dan *loss*, terdokumentasikan dalam **Error! Reference source not found.**, **Error! Reference source not found.**, **Error! Reference source not found.**, **Error! Reference source not found.** dan
Tabel 6. Hasil yang ditunjukkan oleh **Error! Reference source not found.** mengindikasikan bahwa penggunaan HSF dapat menurunkan nilai akurasi dan UAR. Sedangkan untuk nilai *loss*, kombinasi 6 fitur ini dapat nilai tersebut. Hal serupa terjadi pada kombinasi 5, 4, 3 dan 2 fitur.

Error! Reference source not found. menunjukkan bahwa kombinasi 5 fitur LLD dari mfcc, chroma, mel-spectrogram, spectral contrast, dan tonnetz memberikan nilai akurasi dan UAR tertinggi dibandingkan dengan kombinasi 5 fitur lainnya. Nilai akurasi dan UAR tersebut adalah 77% dan 75%. Hal ini menunjukkan bahwa kelima fitur spektral tersebut memberikan kontribusi tertinggi daripada kombinasi lima fitur lainnya. Namun hasil yang berbeda yang didapatkan ketika kombinasi lima fitur HSF yang digunakan. Ketika kombinasi lima fitur HSF digunakan, kombinasi lima fitur mfcc, chroma, spectral contrast, spectral bandwidth dan tonnetz yang menghasilkan akurasi dan UAR tertinggi. Nilai akurasi dan UAR yang dimaksud adalah 67% dan 66%. Meskipun demikian, terdapat kesamaan dari kombinasi 5 fitur tersebut. Kesamaan yang dimaksud adalah ketika fitur mfcc digunakan, nilai akurasi dan UAR yang dihasilkan tetap berada di nilai yang tinggi, yaitu di atas 70% untuk LDD dan di atas 55% untuk HSF. Namun, ketika fitur mfcc tidak digunakan, maka nilai akurasi dan UAR akan turun secara signifikan menjadi 54% untuk LDD dan 45% untuk HSF. Dari fenomena ini, dapat disimpulkan bahwa fitur spektral mfcc memberikan kontribusi paling besar dalam kombinasi 5 fitur ini baik ketika menggunakan LLD maupun HSF.

Error! Reference source not found. menunjukan bahwa kombinasi 4 fitur LLD dari mfcc, chroma, mel spectrogram, dan spectral contrast memberikan nilai akurasi dan UAR tertinggi dibandingkan dengan kombinasi 4 fitur lainnya. Nilai akurasi dan UAR tersebut adalah 76% dan 77%. Sedangkan untuk kombinasi 4 fitur HSF, kombinasi antara fitur mfcc, chroma, mel spectrogram, dan spectral bandwidth yang memberikan nilai akurasi dan UAR tertinggi. Nilai akurasi dan UAR yang dimaksud adalah 66%. Namun terdapat hasil yang berbeda dengan kombinasi 5 fitur sebelumnya yang berkaitan dengan penggunaan mfcc. Ketika kombinasi 4 fitur LLD yang digunakan adalah chroma, mel-spectrogram ,spectral contrast, dan spectral bandwidth, hasil akurasi dan UAR dapat dipertahankan sebesar 75% dan 74%. Namun pada kombinasi HSF, akurasi dan UAR yang didapatkan adalah yang terkecil yaitu 38% dan 39%.

Error! Reference source not found. menunjukkan hasil kombinasi dari 3 fitur LLD yang terbaik adalah mfcc, spectral contrast, dan spectral bandwidth dengan nilai akurasi sebesar 81% dan UAR sebesar 82%. Sedangkan hasil kombinasi dari 3 fitur HSF yang terbaik adalah mfcc, spectral contrast, dan tonnetz dengan nilai akurasi dan UAR sebesar 74%. Jika kita bandingkan dengan kombinasi sebelumnya, yaitu kombinasi 6 fitur, 5 fitur, dan 4 fitur, maka kombinasi 3 fitur ini memberikan nilai akurasi dan UAR yang paling besar baik kombinasi LLD maupun kombinasi HSF. Padahal jumlah kombinasi yang dilakukan lebih sedikit daripada kombinasi-kombinasi sebelumnya. Hal ini menunjukkan bahwa jumlah fitur yang banyak tidak selalu memberikan hasil yang terbaik.

Hal yang sama juga terjadi pada kombinasi 2 fitur yang ditunjukkan oleh
Tabel 6. Akurasi dan UAR tertinggi pada kombinasi 2 fitur LLD mfcc dan spectral contrast sebesar 82% dan kombinasi 2 fitur HSF mfcc dan chroma. Pada kombinasi 2 fitur ini, kombinasi LLD mampu mempertahankan nilai UAR di 82% dan meningkatkan nilai akurasi menjadi 82% meskipun fitur spectral bandwidth dihilangkan. Sedangkan pada kombinasi fitur HSF, kombinasi dari mfcc dan chroma yang meningkatkan nilai akurasi dan UAR. Padahal pada kombinasi 3 fitur sebelumnya, kombinasi dari mfcc dan chroma ini tidak menunjukkan nilai akurasi dan UAR yang tinggi.

Tabel 2: Hasil dari kombinasi 6 fitur.

No.	Kombinasi	LLD			HSF		
		Akurasi	**UAR**	**Loss**	**Akurasi**	**UAR**	**Loss**
1.	mfcc,chroma,mel,spcc,spcb,tonnetz	77%	78%	1,0585	58%	58%	1,8432

Tabel 3: Hasil dari kombinasi 5 fitur

No.	Kombinasi	LLD			HSF		
		Akurasi	**UAR**	**Loss**	**Akurasi**	**UAR**	**Loss**

1.	mfcc,chroma,mel,spcc,spcb	76%	77%	0,864	56%	57%	2,2656
2.	**mfcc,chroma,mel,spcc,tonnetz**	**77%**	**75%**	**0,9328**	56%	57%	2,2656
3.	mfcc,chroma,mel,spcb,tonnetz	72%	72%	1,2608	60%	61%	2,1852
4.	mfcc,mel,spcc,spcb,tonnetz	72%	71%	1,0871	62%	62%	2,1358
5.	**mfcc,chroma,spcc,spcb,tonnetz**	74%	71%	1,1427	**67%**	**66%**	**1,4744**
6.	chroma,mel,spcc,spcb,tonnetz	54%	56%	2,2694	45%	45%	3,3605

Tabel 4: Hasil dari kombinasi 4 fitur

No.	Kombinasi	LLD			HSF		
		Akurasi	UAR	Loss	Akurasi	UAR	Loss
1.	**mfcc,chroma,mel,spcc**	**76%**	**77%**	**1,0216**	59%	60%	2,2923
2.	**mfcc,chroma,mel,spcb**	76%	76%	1,1329	**66%**	**66%**	**1,9377**
3.	mfcc,chroma,mel,tonnetz	69%	69%	1,4147	62%	61%	2,1012
4.	mfcc,mel,spcc,spcb	75%	74%	1,0774	61%	58%	2,2741
5.	mfcc,mel,spcc,tonnetz	55%	57%	1,6224	63%	63%	1,8746
6.	mfcc,spcc,spcb,tonnetz	74%	73%	0,9642	64%	64%	1,8943
7.	**chroma,mel,spcc,spcb**	**75%**	**74%**	**1,0774**	**38%**	**39%**	**3,7328**
8.	chroma,mel,spcc,tonnetz	55%	57%	1,6224	40%	41%	3,5288
9.	chroma,mel,spcb,tonnetz	51%	54%	2,1358	45%	44%	3,4507
10.	chroma,spcc,spcb,tonnetz	43%	45%	2,4403	39%	41%	3,7348
11.	mel,spcc,spcb,tonnetz	49%	49%	2,9408	43%	45%	3,4819

Tabel 5: Hasil dari kombinasi 3 fitur

No.	Kombinasi	LLD			HSF		
		Akurasi	UAR	Loss	Akurasi	UAR	Loss
1.	mfcc-chroma-mel	78%	79%	1,0211	61%	61%	2,2441
2.	mfcc-chroma-spcc	75%	75%	1,2936	68%	68%	1,6257
3.	mfcc-chroma-spcb	72%	73%	1,1308	67%	68%	1,6974
4.	mfcc-chroma-tonnetz	74%	76%	1,3196	66%	67%	1,5419
5.	mfcc-mel-spcc	77%	79%	1,0372	54%	56%	2,698
6.	mfcc-mel-spcb	73%	73%	1,2419	63%	62%	2,2678
7.	mfcc-mel-tonnetz	76%	74%	1,1935	56%	58%	2,0101
8.	**mfcc-spcc-spcb**	**81%**	**82%**	**0,8578**	56%	58%	2,0101
9.	mfcc-spcc-tonnetz	71%	71%	1,1363	**74%**	**74%**	**1,3029**
10.	mfcc-spcb-tonnetz	69%	69%	1,5208	68%	70%	1,5222
11.	chroma-mel-spcc	53%	55%	1,688	49%	53%	3,0606
12.	chroma-mel-spcb	51%	51%	1,7606	42%	42%	3,4284
13.	chroma-mel-tonnetz	44%	45%	2,7434	40%	41%	3,7159
14.	chroma-spcc-spcb	53%	54%	1,9886	48%	49%	3,1157
15.	chroma-spcc-tonnetz	48%	47%	2,8276	39%	41%	3,6463
16.	chroma-spcb-tonnetz	31%	32%	4,107	37%	40%	4,8344
17.	mel-spcc-spcb	50%	51%	1,5472	42%	43%	3,6171
18.	mel-spcc-tonnetz	53%	55%	2,0016	43%	43%	3,2932
19.	mel-spcb-tonnetz	38%	39%	2,6795	39%	39%	3,8568
20.	spcc-spcb-tonnetz	40%	39%	2,7961	39%	37%	4,1414

Tabel 6: Hasil dari kombinasi 2 fitur

No.	Kombinasi	LLD			HSF		
		Akurasi	UAR	Loss	Akurasi	UAR	Loss
1.	mfcc,chroma	78%	79%	1,0732	**72%**	**74%**	**1,2015**
2.	mfcc,mel	79%	80%	1,0222	58%	58%	2,3231
3.	mfcc,spcc	**82%**	**82%**	**0,7883**	72%	72%	1,2482
4.	mfcc,spcb	76%	76%	1,1845	69%	70%	1,5876
5.	mfcc,tonnetz	72%	71%	1,2643	69%	68%	1,773
6.	chroma,mel	47%	47%	2,0357	35%	35%	3,9813
7.	chroma,spcc	51%	51%	2,1442	42%	43%	3,499
8.	chroma,spcb	33%	37%	3,2252	35%	38%	3,8898
9.	chroma,tonnetz	33%	34%	3,9695	36%	37%	4,0694
10.	mel,spcc	51%	54%	1,7764	41%	41%	3,5332
11.	mel,spcb	36%	36%	2,0489	37%	37%	4,057
12.	mel,tonnetz	40%	40%	2,581	39%	39%	3,5331
13.	spcc,spcb	47%	48%	1,9655	35%	35%	3,5012
14.	spcc,tonnetz	36%	36%	3,3756	31%	30%	4,4748
15.	spcb,tonnetz	24%	22%	3,6795	23%	23%	5,457

4. Kesimpulan

Kesimpulan yang didapatkan dari keseluruhan percobaan kombinasi fitur LLD dan HSF adalah fitur mfcc dan spectral contrast memberikan nilai akurasi dan UAR tertinggi. Fitur mfcc adalah fitur yang paling berpengaruh terhadap nilai akurasi dan UAR. Jika fitur mfcc ini dihilangkan maka nilai akurasi dan UAR akan turun secara signifikan. Selain itu penelitian ini memberikan bukti bahwa semakin banyak fitur yang digunakan tidak selalu memberikan hasil akurasi dan UAR yang baik. Namun yang paling mempengaruhi adalah fitur apa yang digunakan, bukan seberapa banyak fitur yang digunakan.

Daftar Rujukan

[1] Mustaqeem, M. Sajjad, and S. Kwon, "Clustering-Based Speech Emotion Recognition by Incorporating Learned Features and Deep BiLSTM," *IEEE Access*, vol. 8, pp. 79861–79875, 2020, doi: 10.1109/ACCESS.2020.2990405.

[2] E. Lieskovská, M. Jakubec, R. Jarina, and M. Chmulík, "A Review on Speech Emotion Recognition Using Deep Learning and Attention Mechanism," *Electronics*, vol. 10, no. 10, p. 1163, May 2021, doi: 10.3390/electronics10101163.

[3] K. Venkataramanan and H. R. Rajamohan, "Emotion Recognition from Speech." arXiv, Dec. 22, 2019. Accessed: Jun. 23, 2023. [Online]. Available: http://arxiv.org/abs/1912.10458

[4] D. Pravena and D. Govind, "Significance of incorporating excitation source parameters for improved emotion recognition from speech and electroglottographic signals," *Int J Speech Technol*, vol. 20, no. 4, pp. 787–797, Dec. 2017, doi: 10.1007/s10772-017-9445-x.

[5] M. Deriche and A. H. Abo Absa, "A Two-Stage Hierarchical Bilingual Emotion Recognition System Using a Hidden Markov Model and Neural Networks," *Arab J Sci Eng*, vol. 42, no. 12, pp. 5231–5249, Dec. 2017, doi: 10.1007/s13369-017-2742-5.

[6] S. Mirsamadi, E. Barsoum, and C. Zhang, "Automatic speech emotion recognition using recurrent neural networks with local attention," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA: IEEE, Mar. 2017, pp. 2227–2231. doi: 10.1109/ICASSP.2017.7952552.

[7] S. R. Bandela and T. K. Kumar, "Stressed speech emotion recognition using feature fusion of teager energy operator and MFCC," in *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Delhi: IEEE, Jul. 2017, pp. 1–5. doi: 10.1109/ICCCNT.2017.8204149.

[8] S. G. Koolagudi, Y. V. S. Murthy, and S. P. Bhaskar, "Choice of a classifier, based on properties of a dataset: case study-speech emotion recognition," *Int J Speech Technol*, vol. 21, no. 1, pp. 167–183, Mar. 2018, doi: 10.1007/s10772-018-9495-8.

[9] Tin Lay Nwe, Say Wei Foo, and L. C. De Silva, "Classification of stress in speech using linear and nonlinear features," in *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003.*

Proceedings. (ICASSP '03)., Hong Kong, China: IEEE, 2003, p. II-9–12. doi: 10.1109/ICASSP.2003.1202281.

[10] R. Xia and Y. Liu, "A Multi-Task Learning Framework for Emotion Recognition Using 2D Continuous Space," *IEEE Trans. Affective Comput.*, vol. 8, no. 1, pp. 3–14, Jan. 2017, doi: 10.1109/TAFFC.2015.2512598.

[11] S. Zhang, S. Zhang, T. Huang, and W. Gao, "Speech Emotion Recognition Using Deep Convolutional Neural Network and Discriminant Temporal Pyramid Matching," *IEEE Trans. Multimedia*, vol. 20, no. 6, pp. 1576–1590, Jun. 2018, doi: 10.1109/TMM.2017.2766843.

[12] N. Cummins, S. Amiriparian, G. Hagerer, A. Batliner, S. Steidl, and B. W. Schuller, "An Image-based Deep Spectrum Feature Representation for the Recognition of Emotional Speech," in *Proceedings of the 25th ACM international conference on Multimedia*, Mountain View California USA: ACM, Oct. 2017, pp. 478–484. doi: 10.1145/3123266.3123371.

[13] J. Lee and I. Tashev, "High-level feature representation using recurrent neural network for speech emotion recognition," in *Interspeech 2015*, ISCA, Sep. 2015, pp. 1537–1540. doi: 10.21437/Interspeech.2015-336.

[14] Z. Aldeneh and E. M. Provost, "Using regional saliency for speech emotion recognition," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA: IEEE, Mar. 2017, pp. 2741–2745. doi: 10.1109/ICASSP.2017.7952655.

[15] H. M. Fayek, M. Lech, and L. Cavedon, "Evaluating deep learning architectures for Speech Emotion Recognition," *Neural Networks*, vol. 92, pp. 60–68, Aug. 2017, doi: 10.1016/j.neunet.2017.02.013.

[16] Y. Xi, P. Li, Y. Song, Y. Jiang, and L. Dai, "Speaker to Emotion: Domain Adaptation for Speech Emotion Recognition with Residual Adapters," in *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, Lanzhou, China: IEEE, Nov. 2019, pp. 513–518. doi: 10.1109/APSIPAASC47483.2019.9023339.

[17] E. Tzinis and A. Potamianos, "Segment-based speech emotion recognition using recurrent neural networks," in *2017 Seventh International Conference on Affective Computing and Intelligent Interaction (ACII)*, San Antonio, TX: IEEE, Oct. 2017, pp. 190–195. doi: 10.1109/ACII.2017.8273599.

[18] A. Ajrana, A. Akbar, and A. Lawi, "Implementasi Algoritma Deep Artificial Neural Network Menggunakan Mel Frequency Cepstrum Coefficient Untuk Klasifikasi Audio Emosi Manusia," *Proceeding KONIK (Konferensi Nasional Ilmu Komputer)*, vol. 5, pp. 66–73, Aug. 2021.

[19] B. T. Atmaja and M. Akagi, "On The Differences Between Song and Speech Emotion Recognition: Effect of Feature Sets, Feature Types, and Classifiers," in *2020 IEEE REGION 10 CONFERENCE (TENCON)*, Nov. 2020, pp. 968–972. doi: 10.1109/TENCON50793.2020.9293852.

[20] Z. Yao, Z. Wang, W. Liu, Y. Liu, and J. Pan, "Speech emotion recognition using fusion of three multi-task learning-based classifiers: HSF-DNN, MS-CNN and LLD-RNN," *Speech Communication*, vol. 120, pp. 11–19, Jun. 2020, doi: 10.1016/j.specom.2020.03.005.

[21] S. R. Livingstone and F. A. Russo, "The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English," *PLOS ONE*, vol. 13, no. 5, p. e0196391, May 2018, doi: 10.1371/journal.pone.0196391.