[Music Genre Classification Using CNN Approach]

Speech Recognition

Kelompok: 13

Anggota Kelompok :

Janssen Mitchellano Hamaziah – 2602117525

Felix Juwono Purwoko-2602160776

Louis-2602147276

I. Latar Belakang

Topik project kami adalah *music genre classification*, yang difungsikan untuk mengklasifikasikan music berdasarkan genre-genre yang ada. Project yang kami lakukan bertujuan untuk membantu *user* / para pecinta music untuk menyesuaikan selera musik mereka berdasarkan klasifikasi genre yang ada. Belakangan Ini telah muncul beberapa aplikasi untuk mendengar music seperti spotify, joox, dll. Dalam hal ini tentu aplikasi aplikasi tersebut mengklasifikasi genre music yang ada. Hal ini untuk memudahkan user dalam mencari referensi genre yang sesuai, serta memudahkan system untuk memberikan rekomendasi kepada user (*recommender system*). Jadi sebagai contoh *user* tersebut memiliki kecenderungan untuk mendengarkan music bergenre *Blues*, maka system akan memberikan rekomendasi lagu lagu lain yang bergenre *Blues* juga. Hal ini juga berlaku untuk genre-genre yang lainnya. Karena topic *project* ini masih sangat hangat dan masih memungkinkan untuk melakukan pengembangan, maka kelompok kami memutuskan mengerjakan topik *project* ini.

II. Rumusan Masalah

1. Apakah akurasi algoritma *machine learning* cukup bagus untuk melakukan klasifikasi *genre music* ?
2. Apakah Audio yang diekstrak dapat menghasilkan fitur yang relevan yang digunakan untuk klasifikasi *genre musik* ?
3. Seberapa pengaruh antara perbedaan akurasi yang dihasilkan oleh model *CNN* dengan model *Machine Learning* yang biasa ?
4. Apa pengaruh klasifikasi genre music terhadap perkembangan industrial music yang ada ?
5. Apakah performa dari model tersebut layak untuk diaplikasi sebagai sistem rekomendasi?

III. Tujuan Project

Pada dasarnya tujuan dari project ini bertujuan untuk mengklasifikasikan atau mengelompokan genre musik secara otomatis. Dengan demikian nantinya akan memudahkan dalam memberikan rekomendasi sistem berdasarkan data musik yang telah diberi label. Dengan Music genre Classification akan memudahkan untuk memberikan preferensi music kepada user. Salah satu tujuan lain dari klasifikasi musik adalah untuk mempermudah searching dan penemuan musik, Karena dapat memberikan filter berdasarkan genre yang disukai oleh user. Terkadang di dunia nyata *user* cenderung menyetel rekomendasi playlist. Disinilah peran music genre klasifikasi digunakan. Sebagai contoh user mendengarkan lagu bergenre pop di sebuah playlist, ketika rekomendasi selanjutnya heavy metal akan menjadi masalah. Maka dari itu sangat penting untuk mengklasifikasi jenis lagu berdasarkan genre dan preferensi *user*.

IV. Ruang Lingkup Project

Ruang lingkup yang tercakup pada proyek ini adalah data musik. Selain itu ada beberapa hal lainnya yang juga termasuk pada ruang lingkup proyek, yaitu preprocessing data (pembersihan data, ekstraksi fitur menggunakan MFCC, normalisasi data, dan lainnya), pengembangan model (CNN), ada juga evaluasi metrik (Confusion Matrix, Classification Report).

V. Deskripsi Dataset

Sumber dataset : GTZAN Dataset - Music Genre Classification (kaggle.com)

Dataset yang kami gunakan dalam *project* ini total terdapat 900 *sample audio track.* yang masing-masingnya berdurasi 30 detik. Dan, terdapat 9 jenis *genre* lagu yaitu:

- Blues
- Classical
- Country
- Disco
- Hip-hop
- Metal
- Pop
- Reggae
- Rock

Tiap dari genrenya memiliki jumlah pembagian yang rata ,yaitu 100 lagu

VI. Tahap-tahap Eksperimen

Berikut pada *Figure 1* ini merupakan gambaran secara keseluruhan untuk tahapan - tahapan yang telah kami lakukan dalam melakukan eksperimen *Genre Music Classification.*

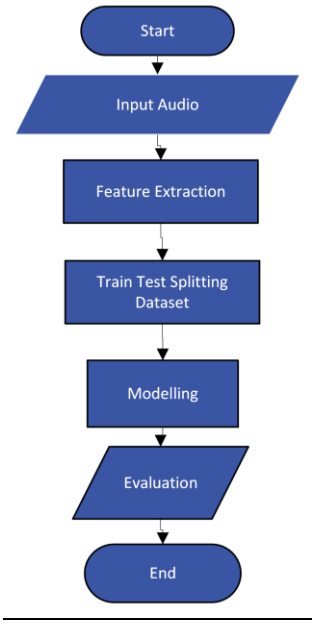

Figure 1. Tahapan Eksperimen secara keseluruhan

Berikut pada *Figure 2* ini merupakan gambaran secara keseluruhan untuk tahapan - tahapan yang telah kami lakukan dalam melakukan eksperimen *Genre Music Classification.*dengan menggunakan pendekatan *CNN*

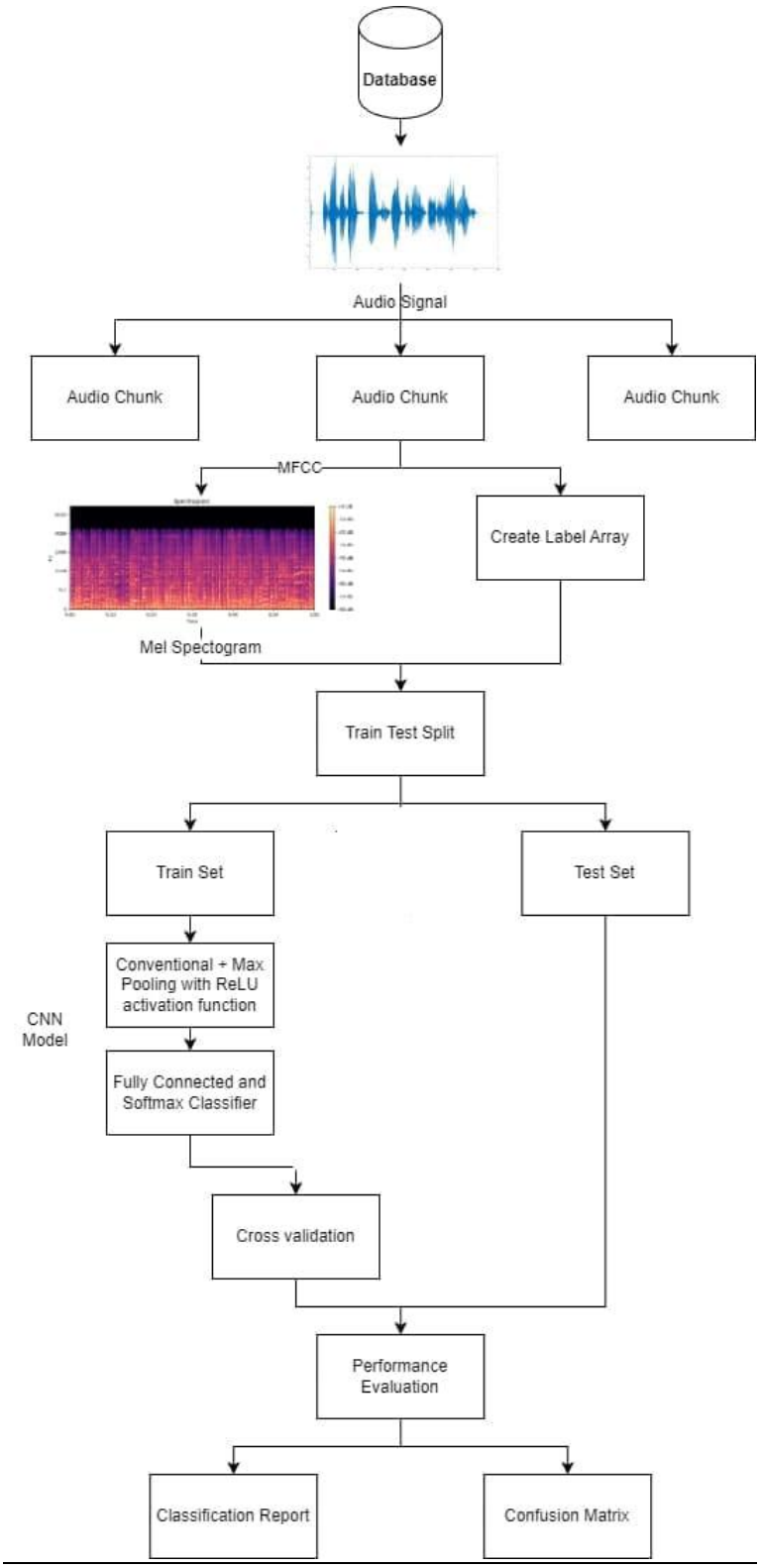

Figure 2. Tahapan Eksperimen CNN

VII. *Preprocessing*

Random Forest, XG Boost , SVM, MLP

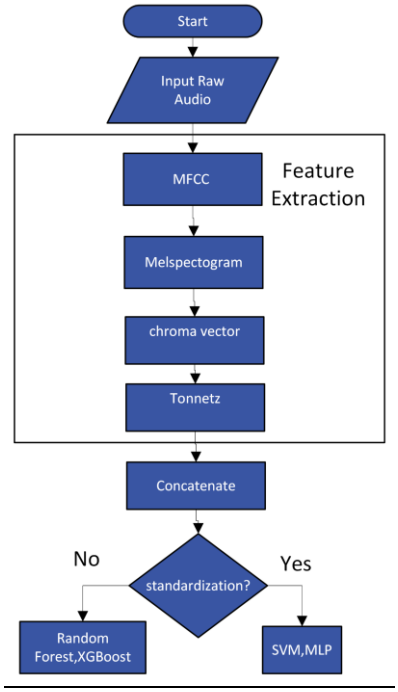

Figure 3 Flowchart .Preprossesing pada Random Forest,XGBoost,SVM,MLP

Secara mekanisme *preprocessing* pada model *Random Forest, XG Boost , SVM,* dan *MLP***,** kami menggunakan metode untuk mengekstrak beberapa fitur representatif audio nya seperti

MFCC :koefisien Cepstral yang dihitung dengan *discrete cosine transform* yang diterapkan pada spektrum daya sinyal. Pita frekuensi spektrum ini diberi jarak secara logaritmik menurut skala Mel.

Melspectogram: spektogram standar dalam skala Mel, yang merupakan skala perseptual nada yang dianggap pendengar memiliki jarak yang sama satu sama lain.

*Chroma Vector :*Vektor fitur Chroma dibentuk dengan cara memproyeksikan spektrum penuh ke 12 nampan yang mencerminkan 12 semitones unik (atau Chroma) dari oktaf musik: C, C#, D, D#, E, F, F#, G, G#, A, A#, B. Proyeksi ini memberikan representasi audio musik yang menarik dan kuat, dan sangat bergantung pada genre musik.

Tonnetz :Representasi ini dihitung dengan memproyeksikan fitur Chroma pada basis 6 dimensi yang mewakili *perfect fifth, minor third,* dan *major third* , masing-masing sebagai koordinat dua dimensi.

Kemudian, dari fitur fitur diatas di gabungkan menjadi sebuah *feature vector*. Kemudian, pada langkah berikutnya pada data yang menggunakan model *SVM,* dan *Multilayer Perceptron (MLP)* dilakukan standardidasi menggunakan *StandardScaler* pada *Library sklearn* terlebih

dahulu .Sedangkan, pada *model Ensemble Learning* seperti *Random Forest* ,dan *XG Boost* tidak dilakukan.

CNN

Pada Data preprocessing kelompok kami melakukan ekstrak audio *music* yang akan diklasifikasi. Untuk proses klasifikasi Kami membuat sebuah function untuk load model audio. Audio audio tersebut dari *class* yang berbeda akan dikonversi menjadi Mel Spectrogram, yang nantinya akan ditangkap polanya oleh model CNN.

Berikut adalah penjelasan langkah - langkah *extract audio :*

1. Menginisialisasikan sebuah list

```
def load_and_preprocess_data(data_dir, classes, target_shape=(150, 150)):
    data = []
    labels = []
```

Code snippet 1

Pada list data digunakan untuk menyimpan sebuah Mel spectrogram yang telah di ekstrak. Sedangkan list label untuk menyimpan label nama yang sesuai dengan spectrogram yang ada

2. Iterasi terhadap setiap class

```
for i_class, class_name in enumerate(classes):
    class_dir = os.path.join(data_dir, class_name)
    print("Processing--",class_name)
```

Code snippet 2

Pada proses ini digunakan untuk melakukan iterasi dan menampilkan nama nama class yang ada

3. Iterasi kedua untuk loop setiap file dalam class yang ditentukan (Nested Loop)

```
  for filename in os.listdir(class_dir):
      if filename.endswith('.wav'):
          file_path = os.path.join(class_dir, filename)
          audio_data, sample_rate = librosa.load(file_path, sr=None)
```

Code snippet 3

Iterasi tersebut melalui setiap direktori class atau file yang ingin di ekstrak. Dan dalam code tersebut terdapat validasi dimana memastikan format audio (.wav). Setelah itu akan memuat audio dan sample rate dari audio tersebut.

4. Perform preprocessing (e.g., convert to Mel spectrogram and resize)

```
chunk_duration = 4  # seconds
overlap_duration = 2  # seconds

# Convert durations to samples
chunk_samples = chunk_duration * sample_rate
overlap_samples = overlap_duration * sample_rate

# Calculate the number of chunks
num_chunks = int(np.ceil((len(audio_data) - chunk_samples) / (chunk_samples - overlap_samples))) + 1
```

Code snippet 4

Pada tahap awal audio akan ditentukan jumlah durasi setiap chunk dan tingkat overlap. dan satuan untuk chunk duration dan overlap duration adalah second. Setelah itu menghitung sampel dari chunk dan overlap. Setelah menghitung jumlah sampel maka perlu juga mengkalkulasi num of chunk.

5. Iterasi terhadap setiap chunk

```
# Iterate over each chunk
for i in range(num_chunks):
    # Calculate start and end indices of the chunk
    start = i * (chunk_samples - overlap_samples)
    end = start + chunk_samples

    # Extract the chunk of audio
    chunk = audio_data[start:end]

    # Compute the Mel spectrogram for the chunk
    mel_spectrogram = librosa.feature.melspectrogram(y=chunk, sr=sr)

#mel_spectrogram = librosa.feature.melspectrogram(y=audio_data, sr=sample_rate)
    mel_spectrogram = resize(np.expand_dims(mel_spectrogram, axis=-1), target_shape)
    data.append(mel_spectrogram)
    labels.append(i_class)
```

Code snippet 5

- Tahap pertama mengkalkulasi start dan end index untuk setiap chunk
- Extract chunk dari audio
- menghitung Mel spectrogram dari chunk
- mengubah atau resize mel spectrogram serta append data dan labels

6. return data dan labels

```
return np.array(data), np.array(labels)
```

Code snippet 6

- return data dan labels yang telah di append dalam bentuk numpy array

VIII. Rancangan Model

Arsitektur *Multilayer Perceptron(MLP)*

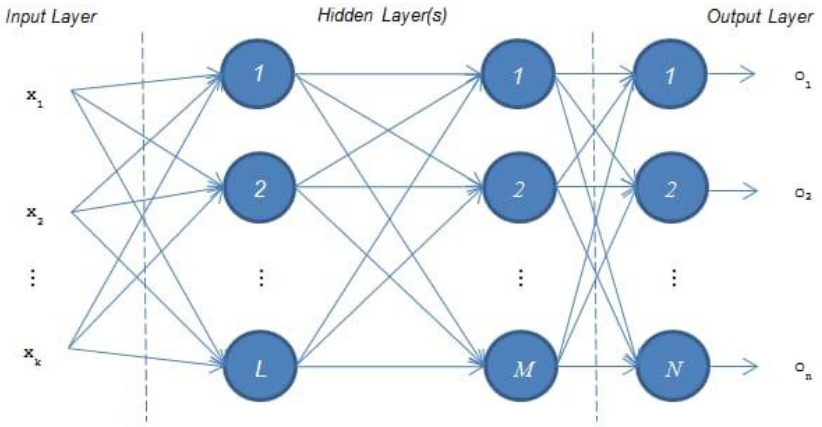

Figure 4 Arsitektur MLP

Arsitektur *CNN*

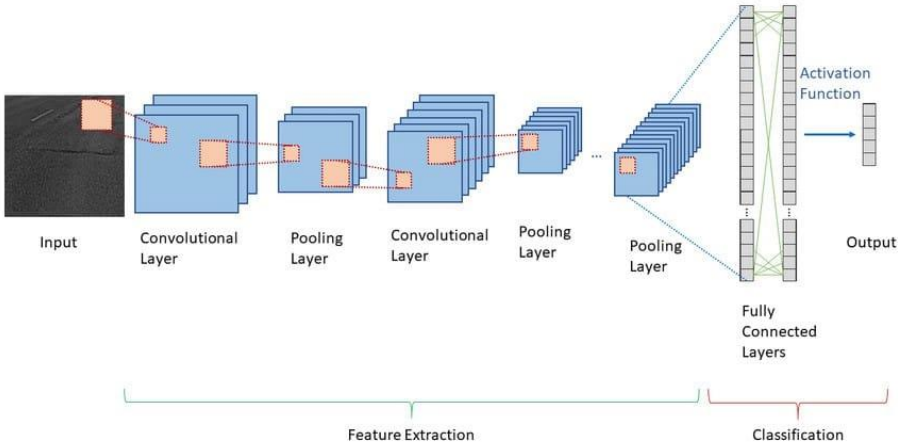

Figure 5 Arsitektur CNN

Berikut merupakan untuk gambaran arsitektur pada model CNN yang kami gunakan secara detail:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 150, 150, 32)      320

 conv2d_1 (Conv2D)           (None, 148, 148, 32)      9248

 max_pooling2d (MaxPooling2  (None, 74, 74, 32)        0
 D)

 conv2d_2 (Conv2D)           (None, 74, 74, 64)        18496

 conv2d_3 (Conv2D)           (None, 72, 72, 64)        36928

 max_pooling2d_1 (MaxPoolin  (None, 36, 36, 64)        0
 g2D)

 conv2d_4 (Conv2D)           (None, 36, 36, 128)       73856

 conv2d_5 (Conv2D)           (None, 34, 34, 128)       147584

 max_pooling2d_2 (MaxPoolin  (None, 17, 17, 128)       0
 g2D)

 dropout (Dropout)           (None, 17, 17, 128)       0

 conv2d_6 (Conv2D)           (None, 17, 17, 256)       295168

 conv2d_7 (Conv2D)           (None, 15, 15, 256)       590080
```

```
max_pooling2d_3 (MaxPoolin  (None, 7, 7, 256)         0
g2D)

conv2d_8 (Conv2D)           (None, 7, 7, 512)         1180160

conv2d_9 (Conv2D)           (None, 5, 5, 512)         2359808

max_pooling2d_4 (MaxPoolin  (None, 2, 2, 512)         0
g2D)

dropout_1 (Dropout)         (None, 2, 2, 512)         0

flatten (Flatten)           (None, 2048)              0

dense (Dense)               (None, 1200)              2458800

dropout_2 (Dropout)         (None, 1200)              0

dense_1 (Dense)             (None, 9)                 10809

=================================================================
Total params: 7181257 (27.39 MB)
Trainable params: 7181257 (27.39 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 6 Detail Arsitektur CNN

IX. Proses Pelatihan Model

Kami pada eksperimen ini menggunakan beberapa model yaitu *CNN*, *SVM* ,*Multilayer Perceptron*, *Random Forest*, dan *XG Boost* untuk mencari nilai akurasi terbaik dalam melakukan prediksi *Genre Music Classification*.Sebelum kita lakukan ke tahap *modelling* data, kami *split* secara stratify terlebih dulu berdasarkan labelnya menjadi *training*, *validation* ,dan *testing dataset* dengan *ratio* yaitu 60:20:20 pada metode *Random Forest*, *XG Boost* , *SVM*, *MLP* .

```python
from sklearn.model_selection import train_test_split

# Assuming features and labels are already numpy arrays
features = numpy.array(features)
labels = numpy.array(labels)

# First split: train and temporary (for further splitting into val and test)
features_train, features_temp, labels_train, labels_temp = train_test_split(
    features, labels, test_size=0.4, stratify=labels, random_state=42)

# Second split: validation and test
features_val, features_test, labels_val, labels_test = train_test_split(
    features_temp, labels_temp, test_size=0.5, stratify=labels_temp, random_state=42)

print(f'Train shape: {features_train.shape}, {labels_train.shape}')
print(f'Validation shape: {features_val.shape}, {labels_val.shape}')
print(f'Test shape: {features_test.shape}, {labels_test.shape}')
```

```
Train shape: (540, 498), (540,)
Validation shape: (180, 498), (180,)
Test shape: (180, 498), (180,)
```

Code snippet 7

Sedangkan, untuk CNN kami hanya melakukan splitting secara stratify berdasarkan labelsnya ke data train dan testing saja dengna ratio yaitu 80:20.

```
[25] from sklearn.model_selection import train_test_split
     X_train, X_test, y_train, y_test = train_test_split(data, labels, test_size=0.2, stratify=labels, random_state=42)
```

Code snippet 8

Berikut merupakan *hyperparameter* yang kami gunakan:

***CNN*:**

Feature Extraction

Hyperparameter pada lapisan konvolusi (Conv2D):

-*Filters*:32, 64, 128, 256, 512

-*Kernel Size* = 3

-*Padding:Same*

-*Activation Function*: ReLU

Hyperparameter pada lapisan *Pooling (MaxPool2D)*

-*Pool size* = 2

-*Strides* = 2

Hyperparameter pada lapisan *Drop Out*

-Rate:0,3 dan 0,45

Classifier

Hyperparameter pada Lapisan *Dense*

-*Unit* = 1200

-*Activation Function*= ReLU(*hidden layers*), dan *softmax* pada *output layers*

SVM :kernel=rbf, C=100

Multilayer Perceptron:

-Jumlah neuron pada layer pertama:300

-Jumlah neuron pada layer kedua:200

-*Activation Function*: ReLU(*hidden layers*), dan *softmax* pada *output layers*

-*Optimizer:RMSprop*

–*Loss function:*Sparse Categorical Crossentropy

-Jumlah Epochs:64

Random Forest : n_estimators=300, random_state=42

XG Boost : n_estimators=300, max_depth=6, learning_rate=0.25, random_state=42

X. Hasil Evaluasi dan Analisis

SVM

	precision	recall	f1-score	support
0	0.74	0.85	0.79	20
1	1.00	0.95	0.97	20
2	0.75	0.75	0.75	20
3	0.76	0.80	0.78	20
4	0.62	0.40	0.48	20
5	0.74	0.85	0.79	20
6	0.67	0.80	0.73	20
7	0.68	0.65	0.67	20
8	0.72	0.65	0.68	20
accuracy			0.74	180
macro avg	0.74	0.74	0.74	180
weighted avg	0.74	0.74	0.74	180

Figure 7. Classification Report SVM

Random Forest

	precision	recall	f1-score	support
0	0.89	0.80	0.84	20
1	1.00	0.95	0.97	20
2	0.81	0.85	0.83	20
3	0.50	0.70	0.58	20
4	0.56	0.50	0.53	20
5	0.67	0.70	0.68	20
6	0.68	0.75	0.71	20
7	0.67	0.60	0.63	20
8	0.87	0.65	0.74	20
accuracy			0.72	180
macro avg	0.74	0.72	0.73	180
weighted avg	0.74	0.72	0.73	180

Figure 8. Classification Report RF

XG Boost

	precision	recall	f1-score	support
0	0.77	0.85	0.81	20
1	1.00	0.95	0.97	20
2	0.69	0.90	0.78	20
3	0.60	0.75	0.67	20
4	0.62	0.65	0.63	20
5	0.89	0.85	0.87	20
6	0.79	0.55	0.65	20
7	0.68	0.65	0.67	20
8	0.80	0.60	0.69	20
accuracy			0.75	180
macro avg	0.76	0.75	0.75	180
weighted avg	0.76	0.75	0.75	180

Figure 9. Classification Report XGBoost

MLP

	precision	recall	f1-score	support
0	0.73	0.55	0.63	20
1	0.95	0.90	0.92	20
2	0.59	0.65	0.62	20
3	0.58	0.55	0.56	20
4	0.43	0.50	0.47	20
5	0.62	0.65	0.63	20
6	0.45	0.50	0.48	20
7	0.55	0.55	0.55	20
8	0.47	0.45	0.46	20
accuracy			0.59	180
macro avg	0.60	0.59	0.59	180
weighted avg	0.60	0.59	0.59	180

Figure 10. Classification Report MLP

CNN

	precision	recall	f1-score	support
blues	0.91	0.90	0.90	316
classical	0.96	0.97	0.96	291
country	0.91	0.77	0.84	326
disco	0.94	0.92	0.93	289
hiphop	0.90	0.98	0.94	295
metal	0.94	0.97	0.96	298
pop	0.94	0.82	0.88	318
reggae	0.87	0.94	0.91	273
rock	0.74	0.86	0.80	292
accuracy			0.90	2698
macro avg	0.90	0.90	0.90	2698
weighted avg	0.90	0.90	0.90	2698

Figure 11. Classification Report CNN

Dari hasil pengujian setiap model dari eksperimen yang telah kami lakukan, kita dapat menyimpulkan bahwa :
1. Metode *Deep Learning* seperti CNN memiliki nilai akurasi yang lebih tinggi dibandingkan dengan menggunakan metode *Machine Learning* biasa

2. Standardidasi dapat meningkatkan tingkat akurasi secara cukup signifikan pada algoritma *Machine Learning* ,seperti SVM, yaitu sebesar 74 % dari yang awal nya 50%

3 .Metode *Ensemble Learning* tanpa adanya standardidasi dapat memiliki tingkat akurasi yang cukup baik ,yaitu sebesar 72%, sedangkan pada XGBoost yaitu sebesar 75%.

XI. Kesimpulan dan Saran

Pada projek kami, telah mencapai nilai akurasi yang maksimal yaitu sebesar 90% dengan menggunakan algoritma CNN .Model ini layak diimplentasikan ke dalam *real world application* ,karena akan dapat membantu dalam mengelompokkan musik - musik yang didengar pengguna aplikasi musik secara akurat, kemudian dapat memberikan rekomendasi musik yang sesuai dengan genre yang sering didengar oleh penggunanya. Mungkin saran untuk penelitian yang dilakukan di masa depan adalah dapat mengeksplor kembali ke arah model deep learning lainnya seperti LSTM ataupun RNN lainnya agar mendapatkan akurasi yang lebih baik

XII. Lampiran

CNN

https://colab.research.google.com/drive/1L2nrm1Xq24LRLx7ovwNkpRl7E4OSnACL?usp=sharing

SVM,MLP,Random Forest,dan XGBoost

https://colab.research.google.com/drive/1KFoMdb77688-DmDKGHqx3ktqztgDxoup?usp=sharing