SIGN LANGUAGE DETECTION

Enrico Evert Nangoy, Huga Sampurna Kho, Vinsensius Christopher Nathaniel Arden

Jurusan Teknik Komputer
Fakultas Teknik dan Informatika
Universitas Multimedia Nusantara
Jl. Scientia Boulevard, Curug Sangereng, Kec. Klp. Dua, Kabupaten Tangerang, Banten 15810

***Abstract* – perkembangan teknologi pada zaman ini sangat lah pesat dan tentunya mempunyai kegunaannya masing-masing dalam membantu manusia mempermudah hidupnya serta dapat menolong banyak manusia. Salah satu nya adalah dengan menggunakan sign language detection untuk orang-orang yang tidak bisa berbicara. Sign language detection berguna untuk membantu para manusia normal jika ingin berbicara dengan manusia yang tidak bisa bicara atau yang sering kita sebut dengan tunawicara.**

I. Pendahuluan

Sign language adalah Bahasa yang menggunakan modalitas visual-manual untuk menampilkan atau menyampaikan suatu makna. Sign language dieksperikan melalui artikulasi manual dan di kombinasikan dengan elemen non-manual. Sign language dalam Bahasa alami yang lengkap dengan tata Bahasa dan leksikonnya sendiri.

II. Literature review

Parameter pembanding yang kami gunakan untuk membandingkan lambang tangan yang kami kirimkan diambil dari website Kaggle.com. [1]

III. Metodologi dan implementasi project

Isi dari datasheet yang kami gunakan adalah 2 buah file bertipe .CSV yang masin-masing file berisi beberapa baris dan 785 kolom. Dari kolom ke-2 dan seterusnya, setiap kolom mewakili nilai pixel yang terkait, dan mewakili gambar greyscale 28x28. Kolom pertama di setiap baris mewakili label dan gambar dengan jumlah 24 label.

Convolutional Neural Network (CNN) [2]adalah algoritma Deep Learing yang dapat menerima input berupa gambar, dan menetapkan importance (bobot dan bias yang dapat dipelajari) ke berbagai aspek/objext dalamgambar kemudian CNN juga dapat membedakan sebuah gambar dan gambar lainnya. Namun sebelum kami proses gambarnya kami lakukan augmentation pada gambar tersebut agar computer lebih mudah untuk melakukan proses perbandingan gambarnya.

Sehingga kami bisa memunculkan tingkat kesamaan dari hasil perbandingan diatas.

[1] https://www.kaggle.com/datasets/datamunge/sign-language-mnist

[2] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Berikut adalah proses implementasi secara singkat.

1) Import libraries.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelBinarizer
import tensorflow as tf
import tensorflow_hub as hub
import pydot
import graphviz
from PIL import Image
from keras.preprocessing.image import ImageDataGenerator
from keras_tuner import RandomSearch
from keras_tuner.engine.hyperparameters import HyperParameters
from tensorflow import keras
from keras.models import Sequential
from keras.layers import Dense, Conv2D , MaxPool2D , Flatten , Dropout , BatchNormalization
from keras.callbacks import ReduceLROnPlateau
from sklearn.metrics import accuracy_score,confusion_matrix
from keras.callbacks import ReduceLROnPlateau
```

2) Data Preprocessing.

- Mengubah array menjadi gambar (tensorflow)

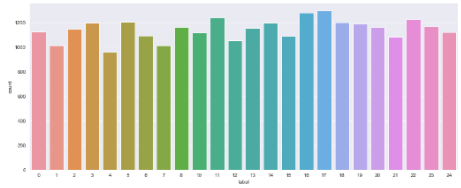
```
y_train = train_data['label']
y_test = test_data['label']
del train_data['label']
del test_data['label']
```

```
unique_labels = y_train.unique()
unique_labels = np.sort(unique_labels)
unique_labels
```

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20, 21, 22, 23, 24], dtype=int64)
```

- Memvisualisasikan label dan memastikan bahwa kumpulan data seimbang

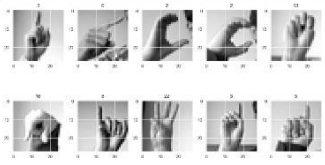
```
plt.figure(figsize=(15,6))
sns.set_style("darkgrid");
sns.countplot(y_train);
```



- Melakukan satu hot encoding untuk label

```
def show_images(images,labels):
    fig,ax = plt.subplots(2,5)
    fig.set_size_inches(10, 6)
    k = 0
    for i in range(2):
        for j in range(5):
            ax[i,j].imshow(images[k] , cmap='gray')
            ax[i,j].set_title(str(unique_labels[np.argmax(y_train[k])]))
            k += 1;
    plt.tight_layout()
```

```
show_images(train_x,y_train)
```



3) Data augmentation

Kami menggunakan augmentasi data seperti cropping, grayscale, shifting, dan horizontal flipping.

```
datagen = ImageDataGenerator(
        featurewise_center=False,  # mengatur rata-rata input menjadi 0 pada kumpulan data
        samplewise_center=False,  # mengatur setiap rata-rata sampel menjadi 0
        featurewise_std_normalization=False,  # membagi input dengan standard-deviation dari kumpulan data
        samplewise_std_normalization=False,  # membagi input dengan standard-deviationnya
        zca_whitening=False,  # mengaplikasikan ZCA whitening
        rotation_range=10,  # merotasi gambar secara random di derajat 0-180
        zoom_range = 0.1,  # mengzoom data
        width_shift_range=0.1,  # mengshift gambar secara horisontal
        height_shift_range=0.1,  # mengshift gambar secara vertikal
        horizontal_flip=False,  # memflip data secara horisontal
        vertical_flip=False)  # memflip data secara vertikal


datagen.fit(train_x)
```

4) Membuat CNN model

Tujuan kami membuat CNN model ini agar kita bisa mengklasifikasikan gambar serta mengaplikasin akurasinya.

```
def build_model(hp):
    model = keras.Sequential([
        keras.layers.Conv2D(
            filters=hp.Int('conv_1_filter', min_value=75, max_value=200, step=25),
            kernel_size=(3,3),
            activation='relu',

            input_shape=(28,28,1)
        ),
        keras.layers.BatchNormalization(),
        keras.layers.MaxPool2D(pool_size=(2,2),strides=2,padding='same'),
        keras.layers.Conv2D(
            filters=hp.Int('conv_2_filter', min_value=50, max_value=125, step=25),
            kernel_size=(3,3),
            activation='relu',
        ),

        keras.layers.Dropout(
            rate = hp.Choice('drop_1_rate', values = [0.1,0.5])
        ),
        keras.layers.BatchNormalization(),
        keras.layers.MaxPool2D(pool_size=(2,2),strides=2,padding='same'),


        keras.layers.Conv2D(
            filters=hp.Int('conv_3_filter', min_value=25, max_value=75, step=25),
            kernel_size=(3,3),
            activation='relu',
        ),
        keras.layers.BatchNormalization(),
        keras.layers.MaxPool2D(pool_size=(2,2),strides=2,padding='same'),
        keras.layers.Flatten(),
        keras.layers.Dense(
            units=hp.Int('dense_1_units', min_value=128, max_value=1024, step=32),
            activation='relu'
        ),
        keras.layers.Dropout(
            rate = hp.Choice('drop_2_rate', values = [0.1,0.3])
        ),
        keras.layers.Dense(26, activation='softmax')
    ])

    model.compile(optimizer=keras.optimizers.Adam(hp.Choice('learning_rate', values=[1e-2, 1e-3])),
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])

    return model
```

```
tuner_search=RandomSearch(build_model,
                          objective='val_accuracy',
                          max_trials=5,directory='output',project_name="ASLdetection1")
```

```
INFO:tensorflow:Reloading Oracle from existing project output\ASLdetection1\oracle.json
INFO:tensorflow:Reloading Tuner from output\ASLdetection1\tuner0.json
```

```
tuner_search.search(train_x,y_train,epochs=5,validation_data = (test_x, y_test))
```

```
INFO:tensorflow:Oracle triggered exit
```

```
model=tuner_search.get_best_models(num_models=1)[0]
```

5) Membuat callback

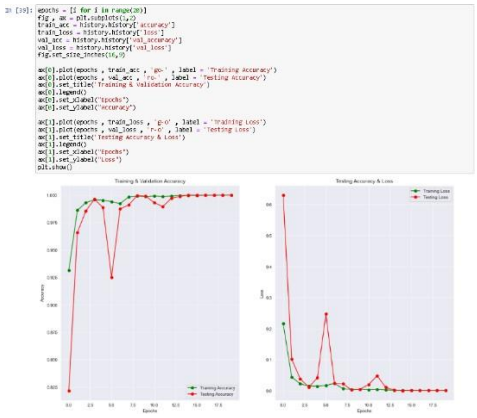
callback ini memantau kuantitas dan jika tidak ada peningkatan yang terlihat untuk jumlah epoch, maka tingkat learningnya akan dikurangkan.

```
history = model.fit(datagen.flow(train_x,y_train, batch_size = 128)
                    ,epochs = 20 # mengatur data sebanyak 20 kali
                    , validation_data = (test_x, y_test)
                    , callbacks = [lr_reduction])
```

6) Analisa model

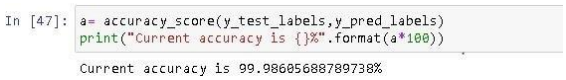
```
lr_reduction = ReduceLROnPlateau(monitor='val_accuracy', patience = 3, verbose=1,factor=0.5, min_lr=0.00001)
```

```
history = model.fit(datagen.flow(train_x,y_train, batch_size = 128)
                    ,epochs = 20
                    , validation_data = (test_x, y_test)
                    , callbacks = [lr_reduction])
```



IV. Hasil dan Analisa

Setelah dibandingkan dengan datasheet oleh CNN maka kami akan menampilkan akurasi dari perbandingan kedua gambar tersebut agar pengguna dapat mengetahui apakah tanda yang telah kita buat sudah maksimal atau belum. Namun kami belum bisa membuat agar program ini memberikan koreksi terhadap lambang atau tanda yang telah kami buat.



V. Kesimpulan

Kami mengambik data dari dataframes dan memproses gambar dengan melakukan augmentasi terlebih dahulu. Dan kami juga membuat model cnn dan menyetel hyperparameters dengan bantuan library dari keras-tuner. Lalu kami melakukan anaslisa pada model kami.

VI. Referensi

[DATAI, (2019, May) Contoh Referensi dari Internet. [Online].

(https://www.kaggle.com/code/kanncaa1/deep-learning-tutorial-for-beginners)