SIGN LANGUAGE RECOGNITION

Kelompok 3

ANGGOTA KELOMPOK

KELSTEN WUISAN - 2602080932

DERREN MALEHONG - 2602081582

KEVIN MAXWELL - 2602083524

KEANE CLEMENT - 2602084981

MASALAH

Bagi individu dengan gangguan pendengaran atau bicara, komunikasi sering kali menjadi tantangan karena keterbatasan bahasa verbal. Untuk mengatasi hal ini, *sign language* atau bahasa isyarat telah menjadi solusi yang efektif.

Namun terdapar hambatan komunikasi antara individu tunarungu dengan masyarakat umum karena tidak semua orang dapat memahami bahasa isyarat.

DATASET

Dataset yang kami gunakan untuk training terdiri dari 87000 gambar yang dimana setiap gambar memiliki ukuran 200x200 piksel.

26 kelas = A-Z
3 kelas = SPACE, DELETE, NOTHING

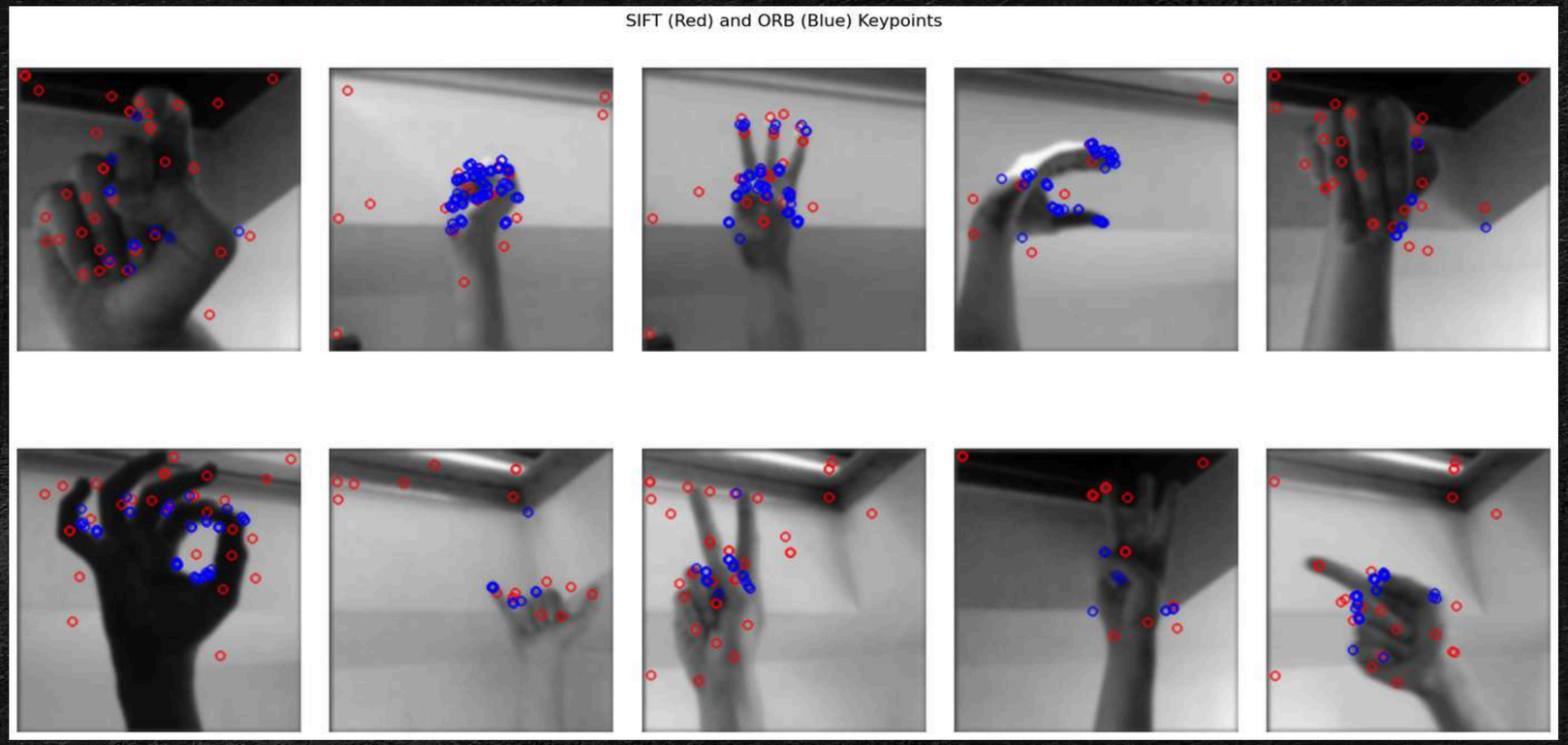
Untuk test dataset masing masing satu gambar A-Z serta space delete nothing

https://www.kaggle.com/datasets/grassknoted/asl-alphabet

Menggunakan metode BoF

1. Preprocess dengan menggunakan gaussian blur

2. Setelah kami bandingkan antara SIFT dan ORB. Kami memutuskan untuk menggunakan ORB karena menangkap fitur yang ditangkap lebih relevan.



Menggunakan metode BoF

3. Untuk visual grammar-nya kami buat dengan menggunakan K-Means, kami putuskan untuk menggunakan 1000 cluster. Karena dataset yang kami gunakan cukup besar (87.000 gambar)

4. Histogram hasil perhitungan frekuensi tiap feature kami scale-down menggunakan standard scaler untuk kemudian di-feed ke Neural Network.

```
# Preprocessing
stacked_descriptor = descriptor_list[0]
for descriptor in descriptor_list[1:]:
    stacked_descriptor = np.vstack((stacked_descriptor, descriptor))

stacked_descriptor = np.float32(stacked_descriptor)
```

✓ 7m 9.5s

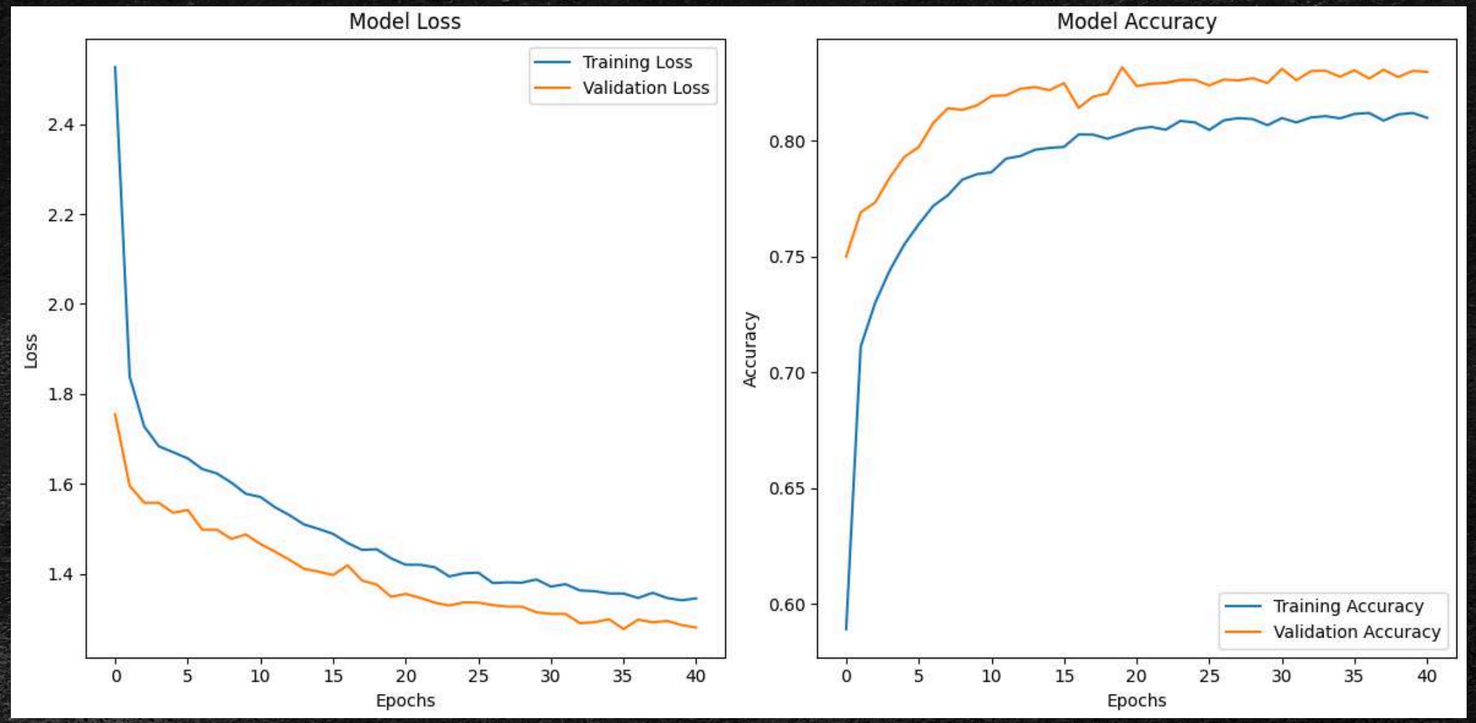
```
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans

# Clustering
kmeans = KMeans(n_clusters=1000, max_iter=20, random_state=42)
kmeans.fit(stacked_descriptor)

centroids = kmeans.cluster_centers_

image_features = np.zeros((len(images_train), len(centroids)), 'float32')

# Vector Quantization
for i in range(len(images_train)):
    words, _ = vq(descriptor_list[i], centroids)

    for w in words:
        image_features[i][w] += 1

std_scaler = StandardScaler().fit(image_features)
image_features = std_scaler.transform(image_features)
```

✓ 6m 8.6s

Menggunakan metode BoF

5. Untuk arsitektur ANN-nya kami menggunakan 3 dense layer dan output-nya menggunakan activation softmax.

6. Hasil training 50 epoch dengan early stopping (val-loss)




```python
# ANN model to process BoF

model = tf.keras.Sequential([
    tf.keras.layers.InputLayer(input_shape=(1000,)),

    tf.keras.layers.Dense(512, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.ReLU(),

    tf.keras.layers.Dense(256, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.ReLU(),
    tf.keras.layers.Dropout(0.3),

    tf.keras.layers.Dense(128, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.ReLU(),
    tf.keras.layers.Dropout(0.3),

    tf.keras.layers.Dense(29, activation='softmax')
])

# Compile the model
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

✓ 0.1s

```python
from sklearn.metrics import accuracy_score

pred = nn_pipeline.predict(images_test)
```

✓ 23.2s

```
544/544 [==============================] - 1s 2ms/step
```

```python
pred = inverse_transform_fn(pred)
```

✓ 0.0s

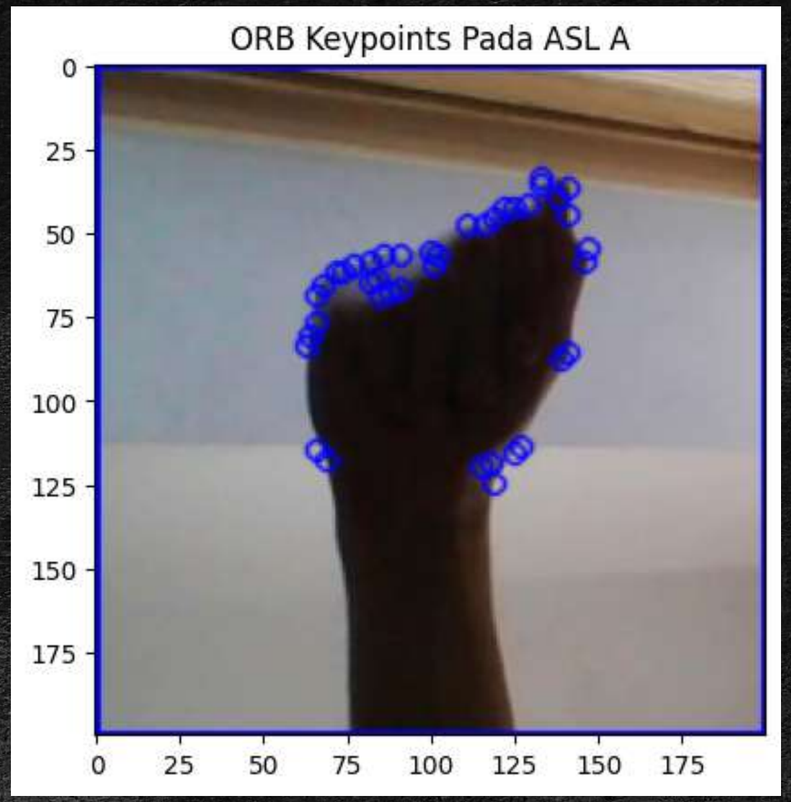
```python
accuracy_score(label_test, pred)
```

✓ 0.0s

```
0.7082758620689655
```

Menggunakan metode CNN

1. Preprocessing gambar dengan mengubah image size, dan setiap label diubah menjadi one hot encoding



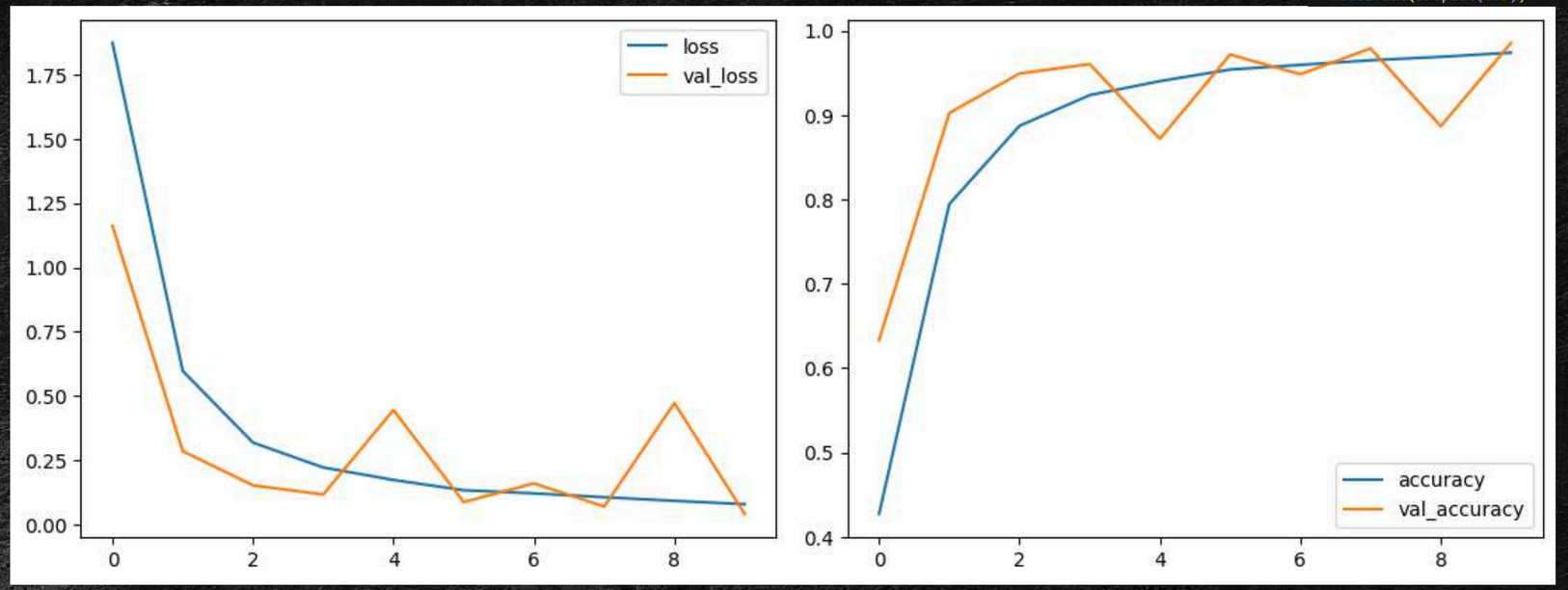

```
batch_size = 64
imageSize = 64
target_dims = (imageSize, imageSize, 3)
num_classes = 29

train_len = 87000
train_dir = 'asl_alphabet_train/asl_alphabet_train/'

def get_data(folder):
    X = np.empty((train_len, imageSize, imageSize, 3), dtype=np.float32)
    y = np.empty((train_len,), dtype=np.int32)
    cnt = 0
    for folderName in os.listdir(folder):
        if not folderName.startswith('.'):
            if folderName in ['A']:
                label = 0
            elif folderName in ['B']:
                label = 1
            elif folderName in ['C']:
                label = 2
                label = 27
            elif folderName in ['space']:
                label = 28
            else:
                label = 29
            for image_filename in os.listdir(folder + folderName):
                img_file = cv2.imread(folder + folderName + '/' + image_filename)
                if img_file is not None:
                    img_file = skimage.transform.resize(img_file, (imageSize, imageSize, 3))
                    img_arr = np.asarray(img_file).reshape((-1, imageSize, imageSize, 3))

                    X[cnt] = img_arr
                    y[cnt] = label
                    cnt += 1
    return X,y
X_train, y_train = get_data(train_dir)
print("Images successfully imported...")

X_data = X_train
y_data = y_train
print("Copies made...")
```

Menggunakan metode CNN

2. Untuk arsitektur CNN, menggunakan 4 Convolutional Layers, 2 Dense Layer dan Activation softmax

3. Berikut adalah hasil training dari 10 epochs

```
# Split Train Test
X_train, X_test, y_train, y_test = train_test_split(X_data, y_data, test_size=0.2, random_state=42,

# Split Train Validation
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.2, random_state=42,

y_cat_train = to_categorical(y_train,29)
y_cat_test = to_categorical(y_test,29)

model = Sequential()

model.add(Conv2D(128, (5, 5), input_shape=(64, 64, 3)))
model.add(Activation('relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))

model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))

model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D((2, 2)))

model.add(Conv2D(32, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D((2, 2)))

model.add(Flatten())

model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
```



VIDEO DEMO

Link Video

https://drive.google.com/file/d/1_TjSbo3q_5BtjH1k5xjxkUD_nxNnIGC1/view?usp=sharing