

Laporan Pengolahan Citra dan Video Deteksi Kartu Remi



Dibuat oleh :

Nama : Jihad Amal Farid

NRP : 5024211072

Kelas : Pengolahan Citra dan Video A

INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2023

Daftar Isi

1	Pendahuluan	2
1.1	Latar Belakang	2
1.2	Tujuan	2
2	Metodologi	2
2.1	Model Neural Network:	2
2.2	Deteksi Kartu:	2
2.3	Permainan Kartu:	2
3	Implementasi kode	3
3.1	Dataset	3
3.1.1	Penamaan Kartu	4
3.1.2	Pemilihan Region Of Interest	4
3.2	Pengambilan dan penyimpanan file Gambar	4
3.3	Training	5
3.3.1	Fungsi LoadCitraTraining	7
3.3.2	Fungsi ModelDeepLearningCNN	7
3.3.3	Fungsi TrainingCNN	7
3.3.4	Fungsi Klasifikasi	7
3.4	Deteksi Kartu	7
3.4.1	Penamaan Kartu	11
3.4.2	Deteksi Kartu	11
3.4.3	Permainan Kartu	11
4	Hasil	13
4.1	Hasil Dataset	13
4.2	Hasil Training	13
4.3	Hasil Deteksi Kartu	13
5	Kesimpulan	14

1 Pendahuluan

1.1 Latar Belakang

Pada tugas akhir mata kuliah Pengolahan Citra dan video yaitu membuat deteksi pada kartu, tidak hanya deteksi kartu saja, selain itu kartu yang dideteksi dibuatkan menjadi permainan. Permainan ini adalah implementasi sederhana dari permainan kartu di mana pemain bersaing dengan bot untuk mencapai total poin sebesar mungkin tanpa melebihi 21. Sistem deteksi kartu menggunakan model neural network untuk mengenali nilai dan jenis kartu dari input kamera.

1.2 Tujuan

Tujuan dari projek ini adalah untuk menciptakan sebuah program yang dapat mengenali jenis dan nilai kartu remi yang ditampilkan melalui kamera. Program ini memanfaatkan model machine learning untuk deteksi kartu dan memberikan pengalaman bermain kartu yang interaktif.

2 Metodologi

2.1 Model Neural Network:

- Menggunakan Keras untuk membangun dan melatih model neural network.
- Model ini dilatih untuk mengenali nilai dan jenis kartu menggunakan dataset yang sesuai.

2.2 Deteksi Kartu:

- Menggunakan OpenCV untuk mengambil frame dari kamera.
- Menentukan Region of Interest (ROI) pada frame untuk mendeteksi kartu.
- Mengubah ukuran dan normalisasi ROI untuk persiapan prediksi menggunakan model.

2.3 Permainan Kartu:

- Implementasi permainan kartu sederhana antara pemain dan bot.
- Pemain memilih untuk "stand" atau "hit" menggunakan tombol keyboard.
- Bot memilih "hit" atau "stand" berdasarkan logika tertentu.

3 Implementasi kode

Pada proyek ini saya menggunakan 3 files yaitu file untuk dataset, training, dan deteksi kartu. berikut ada implementasi kodenya:

3.1 Dataset

Berikut adalah file pemograman dari dataset

```
1 import cv2
2 import numpy as np
3 import time
4 import os
5
6 # Inisialisasi kamera
7 cap = cv2.VideoCapture(0) # Ganti angka 0 dengan indeks kamera jika memiliki lebih dari
    satu kamera
8
9 j = 1
10
11 # Tentukan ranks dan suits untuk setiap putaran
12 ranks = [str(i) for i in range(2, 11)] + ["jack", "queen", "king", "ace"]
13 suits = ["diamonds", "hearts", "spades", "clubs"]
14
15 # Pilih indeks peringkat dan jenis kartu awal
16 current_rank_index = 0
17 current_suit_index = 0
18 # Ukuran kartu remi (lebar x tinggi)
19 card_width, card_height = 240, 360
20
21 # Menghitung koordinat untuk menempatkan kotak di tengah frame
22 frame_width, frame_height = int(cap.get(3)), int(cap.get(4))
23 roi_x = (frame_width - card_width) // 2
24 roi_y = (frame_height - card_height) // 2
25
26 # Loop untuk mengambil gambar dari kamera
27 while True:
28     # Ambil frame dari kamera
29     ret, frame = cap.read()
30
31     # Pilih ROI (Region of Interest) pada tengah frame
32     roi = frame[roi_y:roi_y + card_height, roi_x:roi_x + card_width]
33
34     # Konversi ROI ke skala abu-abu dan ukuran yang lebih kecil
35     roi_gray = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
36     roi_resized = cv2.resize(roi_gray, (128, 128), interpolation=cv2.INTER_AREA)
37
38     # Tampilkan ROI yang telah diproses
39     cv2.imshow("Processed ROI", roi_resized)
40
41     # Duplikat frame untuk menggambar kotak pada frame asli
42     copy = frame.copy()
43
44     # Gambar kotak pada frame
45     cv2.rectangle(copy, (roi_x, roi_y), (roi_x + card_width, roi_y + card_height), (0, 0,
        0), 5)
46
47     # Tampilkan frame
48     cv2.imshow("Capture Images - Press 'P' to capture, 'O' to change rank", copy)
49
50     # Tangkap tombol keyboard
51     key = cv2.waitKey(1)
52
53     # Tombol 'p' untuk mengambil gambar
54     if key == ord('p'):
55         rank = ranks[current_rank_index]
56         suit = suits[current_suit_index]
57
58         # Nama file gambar
59         file_name = f"{rank}_{suit}_{j}.jpg"
60
61         # Cek apakah file dengan nama tersebut sudah ada
62         while os.path.exists(os.path.join("dataset", f"{rank}_{suit}", file_name)):
63             j += 1
64             file_name = f"{rank}_{suit}_{j}.jpg"
65
```

```

66         # Simpan gambar di dalam subfolder
67         folder_path = os.path.join("dataset", f"{rank}_{suit}")
68         image_path = os.path.join(folder_path, file_name)
69
70         # Buat subfolder jika belum ada
71         os.makedirs(folder_path, exist_ok=True)
72
73         cv2.imwrite(image_path, roi_resized)
74
75         print(f"Image {j} captured: {file_name}")
76
77     # Tombol 'o' untuk mengubah peringkat
78     elif key == ord('o'):
79         # Ganti indeks peringkat dan jenis kartu
80         current_rank_index = (current_rank_index + 1) % len(ranks)
81         if ranks[current_rank_index] == "2":
82             current_suit_index = (current_suit_index + 1) % len(suits)
83
84         j = 1
85
86     # Hentikan program ketika tombol 'q' ditekan
87     elif key == ord('q'):
88         # Tambahkan satu ke jumlah percobaan setelah break
89         break
90
91 # Bebaskan sumber daya dan tutup jendela
92 cap.release()
93 cv2.destroyAllWindows()

```

berikut adalah penjelasan dari kode ini :

3.1.1 Penamaan Kartu

```

1 ranks = [str(i) for i in range(2, 11)] + ["jack", "queen", "king", "ace"]
2 suits = ["diamonds", "hearts", "spades", "clubs"]

```

Untuk penamaan kartu diawali dari inisiasi variabel Ranks dan Suits dimana rank kartu adalah 2-10,jack,queen,king, dan ace sedangkan suits adalah lambang dari kartunya seperti diamonds,clubs(clover),hearts, dan spades.

3.1.2 Pemilihan Region Of Interest

```

1 # Pilih ROI (Region of Interest) pada tengah frame
2 roi = frame[roi_y:roi_y + card_height, roi_x:roi_x + card_width]
3
4 # Konversi ROI ke skala abu-abu dan ukuran yang lebih kecil
5 roi_gray = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
6 roi_resized = cv2.resize(roi_gray, (128, 128), interpolation=cv2.INTER_AREA)
7
8 # Tampilkan ROI yang telah diproses
9 cv2.imshow("Processed ROI", roi_resized)

```

Dari kode diatas adalah untuk pemilihan area yang akan dideteksi kartu, setelah itu gambar diubah menjadi warna abu-abu dan diperkecil gambarnya agar lebih efisien

3.2 Pengambilan dan penyimpanan file Gambar

```

1 # Tombol 'p' untuk mengambil gambar
2 if key == ord('p'):
3     rank = ranks[current_rank_index]
4     suit = suits[current_suit_index]
5
6     # Nama file gambar
7     file_name = f"{rank}_{suit}_{j}.jpg"
8
9     # Cek apakah file dengan nama tersebut sudah ada
10    while os.path.exists(os.path.join("dataset", f"{rank}_{suit}", file_name)):
11        j += 1
12        file_name = f"{rank}_{suit}_{j}.jpg"
13
14    # Simpan gambar di dalam subfolder
15    folder_path = os.path.join("dataset", f"{rank}_{suit}")
16    image_path = os.path.join(folder_path, file_name)
17

```

```

18     # Buat subfolder jika belum ada
19     os.makedirs(folder_path, exist_ok=True)
20
21     cv2.imwrite(image_path, roi_resized)
22
23     print(f"Image {j} captured: {file_name}")
24
25     # Tombol 'o' untuk mengubah peringkat
26     elif key == ord('o'):
27         # Ganti indeks peringkat dan jenis kartu
28         current_rank_index = (current_rank_index + 1) % len(ranks)
29         if ranks[current_rank_index] == "2":
30             current_suit_index = (current_suit_index + 1) % len(suits)
31
32     j = 1

```

Untuk pengambilan gambar pada kartu agar dapat dijadikan dataset yaitu menekan tombol P dan O untuk perubahan rank, jadi sistemnya adalah diawali dari ranknya 2 dan suitnya adalah diamond. Saat itu kita mengambil gambar 2 diamond dengan menekan tombol P, jika sudah cukup pengambilan gambarnya tekan tombol O untuk mengubah rank dan menjadi 3 diamond. jika diteruskan maka 2-10, jack queen, king, ace diamond, setelah ace diamond lalu kita menekan tombol P maka ranknya kembali menjadi 2 tetapi suitsnya berubah menjadi hearts. Urutan perubahannya berdasarkan line 12-13. Gambar yang sudah diambil akan otomatis tersimpan dalam file dengan nama file yang sesuai rank dan suit kartunya.

3.3 Training

Berikut adalah file pemrograman dari training

```

1 import os
2 import cv2
3 import numpy as np
4 from keras.models import Sequential
5 from keras.layers import Dense, Conv2D, Flatten, MaxPooling2D, Input
6
7
8 def LoadCitraTraining(sDir, LabelKelas):
9     JumlahKelas = len(LabelKelas)
10    TargetKelas = np.eye(JumlahKelas)
11    # Menyiapkan variabel list untuk data menampung citra dan data target
12    X = [] #Menampung Data Citra
13    T = [] #Menampung Target
14    for i in range(len(LabelKelas)):
15        #Membaca file citra di setiap direktori data set
16        DirKelas = os.path.join(sDir, LabelKelas[i])
17        files = os.listdir(DirKelas)
18        for f in files:
19            ff = f.lower()
20            print(f)
21            #memilih citra dengan ekstensi jpg, jpeg, dan png
22            if (ff.endswith('.jpg') or ff.endswith('.jpeg') or ff.endswith('.png')):
23                NmFile = os.path.join(DirKelas, f)
24                img = np.double(cv2.imread(NmFile, 1))
25                img = cv2.resize(img, (128, 128));
26                img = np.asarray(img) / 255;
27                img = img.astype('float32')
28                #Menambahkan citra dan target ke daftar
29                X.append(img)
30                T.append(TargetKelas[i])
31        #-----akhir loop :Pfor f in files-----
32    #-----akhir loop :for i in range(len(LabelKelas))----
33
34    #Mengubah List Menjadi numpy array
35    X = np.array(X)
36    T = np.array(T)
37    X = X.astype('float32')
38    T = T.astype('float32')
39    return X, T
40
41 def ModelDeepLearningCNN(JumlahKelas):
42
43     model = Sequential()
44     model.add(Input((128, 128, 3)))
45     model.add(Conv2D(32, kernel_size=3, activation='relu', padding="same"))

```

```

46     model.add(MaxPooling2D((2, 2), padding='same'))
47     model.add(Conv2D(32, kernel_size=3, activation='relu',padding="same"))
48     model.add(MaxPooling2D((2, 2), padding='same'))
49     model.add(Conv2D(32, kernel_size=3, activation='relu',padding="same"))
50
51     model.add(Flatten())
52     model.add(Dense(JumlahKelas, activation='softmax'))
53
54
55     model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy',
56 ])
57     #ModelCNN.compile(loss='mse', optimizer='adam', metrics=['accuracy'])
58     return model
59
60 def TrainingCNN(JumlahEpoh,DirektoriDataSet,LabelKelas,NamaFileBobot):
61     #Membaca Data training dan label Kelas
62     X,D=LoadCitraTraining(DirektoriDataSet,LabelKelas)
63     JumlahKelas = len(LabelKelas)
64     #Membuat Model CNN
65     ModelCNN =ModelDeepLearningCNN(JumlahKelas)
66     #Trainng
67     history=ModelCNN.fit(X, D,epochs=JumlahEpoh,shuffle=True)
68     #Menyimpan hasil learning
69     ModelCNN.save(NamaFileBobot)
70     #Mengembalikan output
71     return ModelCNN,history
72
73 def Klasifikasi(DirDataSet,DirKlasifikasi,LabelKelas,ModelCNN=[]):
74     #Menyiapkan Data input Yang akan di kasifikasikan
75     X=[]
76     ls = []
77     DirKelas = DirDataSet+"/"+DirKlasifikasi
78     print(DirKelas)
79     files = os.listdir(DirKelas)
80     n=0
81     for f in files:
82         ff=f.lower()
83         print(f)
84         if (ff.endswith('.jpg')|ff.endswith('.jpeg')|ff.endswith('.png')):
85             ls.append(ff)
86             NmFile = os.path.join(DirKelas,f)
87             img= cv2.imread(NmFile,1)
88             img=cv2.resize(img,(128,128))
89             img= np.asarray(img)/255
90             img=img.astype('float32')
91             X.append(img)
92
93
94     X=np.array(X)
95     X=X.astype('float32')
96     #Melakukan prediksi Klasifikasi
97     hs=ModelCNN.predict(X)
98
99     LKlasifikasi=[]
100     LKelasCitra =[]
101     n = X.shape[0]
102     for i in range(n):
103         v=hs[i,:]
104         if v.max()>0.5:
105             idx = np.max(np.where( v == v.max()))
106             LKelasCitra.append(LabelKelas[idx])
107         else:
108             idx=-1
109             LKelasCitra.append("-")
110         #-----akhir if
111         LKlasifikasi.append(idx)
112     #----akhir for
113     LKlasifikasi = np.array(LKlasifikasi)
114     return ls, hs, LKelasCitra
115
116
117 #Menentukan Direktori Yang menyimpan Data set
118 DirektoriDataSet="dataset"
119
120 #Label Data Set

```

```

121 LabelKelas = []
122
123 # Tentukan ranks dan suits untuk setiap putaran
124 ranks = [str(i) for i in range(2, 11)] + ["jack", "queen", "king", "ace"]
125 suits = ["diamonds", "hearts", "spades", "clubs"]
126
127 # Loop untuk menghasilkan label
128 for rank in ranks:
129     for suit in suits:
130         label = f"{rank}_{suit}"
131         LabelKelas.append(label)
132
133 # Cetak hasil label
134 print(LabelKelas)
135
136 #c. Inisialisasi parameter Training
137 JumlahEpoh = 12
138 FileBobot = "DeteksiKartu.h5"
139 #d. training
140 ModelCNN, history = TrainingCNN(JumlahEpoh, DirektoriDataSet, LabelKelas, FileBobot)
141 ModelCNN.summary()

```

berikut adalah penjelasan dari kode ini :

3.3.1 Fungsi LoadCitraTraining

Membaca dataset citra untuk training. Mengambil citra dari setiap kelas (peringkat dan jenis kartu) dan menyimpannya dalam bentuk array numpy.

3.3.2 Fungsi ModelDeepLearningCNN

Membangun model Convolutional Neural Network (CNN) menggunakan Keras untuk klasifikasi citra. Model ini memiliki lapisan konvolusi, max pooling, dan lapisan fully connected.

3.3.3 Fungsi TrainingCNN

Melakukan training pada model CNN dengan citra-citra yang telah dibaca sebelumnya. Hasil training disimpan dalam file bobot.

3.3.4 Fungsi Klasifikasi

Mengklasifikasikan citra-citra yang terdapat dalam direktori klasifikasi tertentu menggunakan model CNN yang telah dilatih sebelumnya. Hasil klasifikasi berupa label kelas untuk setiap citra.

3.4 Deteksi Kartu

Berikut adalah file pemograman dari deteksi kartu

```

1 import cv2
2 import numpy as np
3 import random
4 from keras.models import load_model
5
6 # Load model
7 model = load_model("DeteksiKartu.h5")
8
9 # Inisialisasi kamera
10 cap = cv2.VideoCapture(0) # Ganti angka 0 dengan indeks kamera jika memiliki lebih dari
    satu kamera
11 detected_player_frame = np.zeros((400, 400, 3), dtype=np.uint8)
12 detected_bot_frame = np.zeros((400, 400, 3), dtype=np.uint8)
13
14 # Tentukan ranks dan suits untuk setiap putaran
15 ranks = [str(i) for i in range(2, 11)] + ["jack", "queen", "king", "ace"]
16 suits = ["diamonds", "hearts", "spades", "clubs"]
17
18 # Ukuran kartu remi (lebar x tinggi)
19 card_width, card_height = 240, 360
20
21 # Menghitung koordinat untuk menempatkan kotak di bawah tengah frame
22 frame_width, frame_height = int(cap.get(3)), int(cap.get(4))

```



```

23 roi_x = (frame_width - card_width) // 2
24 roi_y = frame_height - card_height - 50
25
26 # Inisialisasi total poin
27 player_points = 0
28 bot_points = 0
29
30 # Dimulai dengan giliran player
31 player_turn = random.choice([True, False])
32 player_stand = False
33
34
35 # List untuk menyimpan kartu yang sudah terdeteksi
36 detected_cards = set()
37 last_detected_card = None
38
39 # Direktori tempat gambar kartu disimpan
40 card_images_dir = "PNG/"
41
42 def read_card_image(label):
43     # Dapatkan nama file gambar kartu sesuai label
44     image_path = f"{card_images_dir}{label}.png"
45     # Baca gambar kartu
46     card_image = cv2.imread(image_path)
47     # Pastikan gambar terbaca dengan benar
48     if card_image is None:
49         raise FileNotFoundError(f"Image not found: {image_path}")
50     return card_image
51
52
53 # Loop untuk mengambil gambar dari kamera
54 while True:
55     # Ambil frame dari kamera
56     key = cv2.waitKey(1)
57     ret, frame = cap.read()
58     roi = frame[roi_y:roi_y + card_height, roi_x:roi_x + card_width]
59     # Konversi ROI ke ukuran yang lebih kecil
60     roi_resized = cv2.resize(roi, (128, 128), interpolation=cv2.INTER_AREA)
61     roi_resized = roi_resized.astype("float32") / 255.0
62     roi_resized = np.expand_dims(roi_resized, axis=0)
63
64     # Prediksi kelas kartu menggunakan model
65     pred = model.predict(roi_resized)
66
67     # Loop untuk menghasilkan label
68     LabelKelas = []
69     for rank in ranks:
70         for suit in suits:
71             label = f"{rank}_{suit}"
72             LabelKelas.append(label)
73
74     # Tampilkan hasil prediksi
75     label = LabelKelas[np.argmax(pred)]
76     # Jika prediksi memiliki kepercayaan di atas ambang batas tertentu, tampilkan label
77     # jika tidak, tampilkan "Unknown"
78     confidence_threshold = 0.5
79     if np.max(pred) > confidence_threshold:
80         # Hitung poin kartu
81         rank = label.split("_")[0]
82         if rank in ["2", "3", "4", "5", "6", "7", "8", "9"]:
83             points = int(rank)
84         elif rank in ["10", "jack", "queen", "king"]:
85             points = 10
86         else:
87             points = 11
88
89     # Pemain memilih "stand" atau "hit"
90     key = cv2.waitKey(1)
91     if player_turn:
92         if key == ord('s'): # Jika tombol 's' ditekan, pemain memilih "stand"
93             player_stand = True
94         elif key == ord('h'): # Jika tombol 'h' ditekan, pemain memilih "hit"
95             if label not in detected_cards:
96                 player_points += points
97                 frame_to_display = detected_player_frame
98                 detected_cards.add(label)

```

```

98         # Tambahkan tulisan kartu ke frame detected_card_frame
99         card_image = read_card_image(label)
100         resized_card_image = cv2.resize(card_image, (75, 75))
101         detected_player_frame[len(detected_cards) * 20:len(detected_cards) *
20 + resized_card_image.shape[0], 10:10 + resized_card_image.shape[1]] =
resized_card_image
102         player_turn = False
103
104     if not player_turn:
105         # Keputusan bot hit atau stand
106         if bot_points < 17:
107             # Bot memilih "hit" dan secara acak memilih kartu
108             random_rank = random.choice(ranks)
109             random_suit = random.choice(suits)
110             label = f"{random_rank}_{random_suit}"
111
112             # Jika kartu belum terdeteksi sebelumnya, tambahkan poin ke total bot
113             if label not in detected_cards:
114                 detected_cards.add(label)
115                 # Tambahkan tulisan kartu ke frame detected_card_frame
116                 card_image = read_card_image(label)
117                 resized_card_image = cv2.resize(card_image, (75, 75))
118                 detected_bot_frame[len(detected_cards) * 20:len(detected_cards) * 20
+ resized_card_image.shape[0], 10:10 + resized_card_image.shape[1]] =
resized_card_image
119
120                 # Update total poin bot
121                 rank = label.split("_")[0]
122                 if rank in ["2", "3", "4", "5", "6", "7", "8", "9"]:
123                     points = int(rank)
124                 elif rank in ["10", "jack", "queen", "king"]:
125                     points = 10
126                 else:
127                     points = 11
128                 bot_points += points
129                 # Ganti giliran ke pemain setelah bot "hit"
130                 player_turn = True
131             else:
132                 player_turn = True
133
134     if player_stand:
135         # Keputusan bot hit atau stand
136         player_turn = False
137         if bot_points < 17:
138             # Bot memilih "hit" dan secara acak memilih kartu
139             random_rank = random.choice(ranks)
140             random_suit = random.choice(suits)
141             label = f"{random_rank}_{random_suit}"
142
143             # Jika kartu belum terdeteksi sebelumnya, tambahkan poin ke total bot
144             if label not in detected_cards:
145                 detected_cards.add(label)
146                 # Tambahkan tulisan kartu ke frame detected_card_frame
147                 card_image = read_card_image(label)
148                 resized_card_image = cv2.resize(card_image, (75, 75))
149                 detected_bot_frame[len(detected_cards) * 20:len(detected_cards) * 20
+ resized_card_image.shape[0], 10:10 + resized_card_image.shape[1]] =
resized_card_image
150
151                 # Update total poin bot
152                 rank = label.split("_")[0]
153                 if rank in ["2", "3", "4", "5", "6", "7", "8", "9"]:
154                     points = int(rank)
155                 elif rank in ["10", "jack", "queen", "king"]:
156                     points = 10
157                 else:
158                     points = 11
159                 bot_points += points
160                 # Ganti giliran ke pemain setelah bot "hit"
161                 player_turn = True
162             else:
163                 player_stand = False # Atur player_stand menjadi False di sini
164         else:
165             # Tentukan pemenang setelah loop selesai
166             if player_points == bot_points or (bot_points > 21 and player_points >
21):

```

```

167         winner = "Draw"
168     elif player_points > 21 or (bot_points <= 21 and bot_points >
player_points):
169         winner = "Bot"
170     elif bot_points > 21 or (player_points <= 21 and player_points >
bot_points):
171         winner = "Player"
172     else:
173         # Pilih pemenang berdasarkan total poin tertinggi
174         winner = "Player" if player_points > bot_points else "Bot"
175         # Tampilkan hasil permainan pada frame terakhir
176         cv2.putText(frame, f"The winner is: {winner}", (220,180), cv2.
FONT_HERSHEY_SIMPLEX, 2, (255, 255, 255), 3)
177
178     # Update last_detected_card
179     last_detected_card = label
180     cv2.putText(detected_player_frame, "Player Card:", (10, 15), cv2.
FONT_HERSHEY_SIMPLEX, 0.6, (255, 255, 255), 2)
181     cv2.putText(detected_bot_frame, "Bot Card:", (10, 15), cv2.FONT_HERSHEY_SIMPLEX,
0.6, (255, 255, 255), 2)
182     cv2.putText(frame, f"{label} ({points})", (roi_x + 5, roi_y - 10), cv2.
FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
183
184     else: # jika tidak terdeteksi kartu
185         if player_turn:
186             if key == ord('s'): # Jika tombol 's' ditekan, pemain memilih "stand"
187                 player_stand = True
188             if player_stand:
189                 if bot_points >= 17:
190                     # Tentukan pemenang setelah loop selesai
191                     if player_points == bot_points or (bot_points > 21 and player_points >
21):
192                         winner = "Draw"
193                     elif player_points > 21 or (bot_points <= 21 and bot_points >
player_points):
194                         winner = "Bot"
195                     elif bot_points > 21 or (player_points <= 21 and player_points >
bot_points):
196                         winner = "Player"
197                     else:
198                         # Pilih pemenang berdasarkan total poin tertinggi
199                         winner = "Player" if player_points > bot_points else "Bot"
200                         # Tampilkan hasil permainan pada frame terakhir
201                         cv2.putText(frame, f"The winner is: {winner}", (220,180), cv2.
FONT_HERSHEY_SIMPLEX, 2, (255, 255, 255), 3)
202
203     # Gambar kotak pada frame
204     cv2.rectangle(frame, (roi_x, roi_y), (roi_x + card_width, roi_y + card_height), (0,
0, 0), 5)
205
206     # Tampilkan player & bot poin
207     cv2.putText(frame, f"Player Points: {player_points}/21", (10, 50), cv2.
FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)
208     cv2.putText(frame, f"Bot Points: {bot_points}/21", (10, 100), cv2.
FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)
209
210     #Tampilan Text Instruksi
211     cv2.putText(frame, f"Hit press H and Stand press S", (250, 250), cv2.
FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)
212
213     # Tampilkan frame
214     cv2.imshow("Detected Cards - Player", detected_player_frame)
215     cv2.imshow("detection card", frame)
216     cv2.imshow("Detected Cards - Bot", detected_bot_frame)
217
218
219     # Hentikan program ketika tombol 'q' ditekan
220     if key == ord('q'):
221         break
222
223
224
225 # Tutup jendela
226 cap.release()
227 cv2.destroyAllWindows()

```

berikut adalah penjelasan dari kode ini :

3.4.1 Penamaan Kartu

```
1 ranks = [str(i) for i in range(2, 11)] + ["jack", "queen", "king", "ace"]
2 suits = ["diamonds", "hearts", "spades", "clubs"]
```

Untuk penamaan kartu diawali dari inisiasi variabel Ranks dan Suits dimana rank kartu adalah 2-10,jack,queen,king, dan ace sedangkan suits adalah lambang dari kartunya seperti diamonds,clubs(clover),hearts, dan spades.

3.4.2 Deteksi Kartu

```
1 roi = frame[roi_y:roi_y + card_height, roi_x:roi_x + card_width]
2 # Konversi ROI ke ukuran yang lebih kecil
3 roi_resized = cv2.resize(roi, (128, 128), interpolation=cv2.INTER_AREA)
4 roi_resized = roi_resized.astype("float32") / 255.0
5 roi_resized = np.expand_dims(roi_resized, axis=0)
6
7 # Prediksi kelas kartu menggunakan model
8 pred = model.predict(roi_resized)
9
10 # Loop untuk menghasilkan label
11 LabelKelas = []
12 for rank in ranks:
13     for suit in suits:
14         label = f"{rank}_{suit}"
15         LabelKelas.append(label)
16
17 # Tampilkan hasil prediksi
18 label = LabelKelas[np.argmax(pred)]
19 # Jika prediksi memiliki kepercayaan di atas ambang batas tertentu, tampilkan label
20 # jika tidak, tampilkan "Unknown"
21 confidence_threshold = 0.5
22 if np.max(pred) > confidence_threshold:
23     cv2.putText(frame, f"{label} ({points})", (roi_x + 5, roi_y - 10), cv2.
24     FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
```

Untuk pendeteksi kartu, diawali dari pengambilan ROI atau area mana yang akan terdeteksi kartu. Setelah itu ROI diubah ukurannya menjadi kecil dan dinormalisasi lalu memprediksi kelas kartu berdasarkan ROI yang telah diubah ukurannya. Adapun untuk loop label kelas yang digunakan untuk label sebagai penamaan kartunya. Untuk penampikan hasil prediksi kartu adalah berdasarkan label dengan probabilitas tertinggi dari hasil prediksi, jika probabilitas tertinggi melebihi ambang batas tertentu (confidence threshold), label kartu dan nilai poinnya ditampilkan pada frame utama menggunakan cv2.putText.

3.4.3 Permainan Kartu

1. Player

```
1 if player_turn:
2     if key == ord('s'): # Jika tombol 's' ditekan, pemain memilih "stand"
3         player_stand = True
4     elif key == ord('h'): # Jika tombol 'h' ditekan, pemain memilih "hit"
5         if label not in detected_cards:
6             player_points += points
7             frame_to_display = detected_player_frame
8             detected_cards.add(label)
9             # Tambahkan tulisan kartu ke frame detected_card_frame
10            card_image = read_card_image(label)
11            resized_card_image = cv2.resize(card_image, (75, 75))
12            detected_player_frame[len(detected_cards) * 20:len(
13            detected_cards) * 20 + resized_card_image.shape[0], 10:10 + resized_card_image.
14            shape[1]] = resized_card_image
15            player_turn = False
```

kode ini adalah untuk player (pemain) dimana ketika player menekan tombol S adalah untuk stand dan H adalah untuk Hit. Ketika player hit kartu maka jika kartu tidak berada list detected card player akan mendapatkan dari kartu tersebut dan kartu tersebut akan masuk list detected card yang dimana tidak akan bisa digunakan lagi.

2. Bot

```
1 if not player_turn:
2     # Keputusan bot hit atau stand
3     if bot_points < 17:
4         # Bot memilih "hit" dan secara acak memilih kartu
5         random_rank = random.choice(ranks)
6         random_suit = random.choice(suits)
```

```

7         label = f"{random_rank}_{random_suit}"
8
9         # Jika kartu belum terdeteksi sebelumnya, tambahkan poin ke total
10        bot
11        if label not in detected_cards:
12            detected_cards.add(label)
13            # Tambahkan tulisan kartu ke frame detected_card_frame
14            card_image = read_card_image(label)
15            resized_card_image = cv2.resize(card_image, (75, 75))
16            detected_bot_frame[len(detected_cards) * 20:len(detected_cards)
17            * 20 + resized_card_image.shape[0], 10:10 + resized_card_image.shape[1]] =
18            resized_card_image
19
20            # Update total poin bot
21            rank = label.split("_")[0]
22            if rank in ["2", "3", "4", "5", "6", "7", "8", "9"]:
23                points = int(rank)
24            elif rank in ["10", "jack", "queen", "king"]:
25                points = 10
26            else:
27                points = 11
28            bot_points += points
29            # Ganti giliran ke pemain setelah bot "hit"
30            player_turn = True
31        else:
32            player_turn = True

```

Kode ini adalah untuk bot, dalam permainannya bot akan memilih kartu secara acak jika point bot masih dibawah 17.

3. Point

```

1 player_points = 0
2 bot_points = 0
3     # Hitung poin kartu
4     rank = label.split("_")[0]
5     if rank in ["2", "3", "4", "5", "6", "7", "8", "9"]:
6         points = int(rank)
7     elif rank in ["10", "jack", "queen", "king"]:
8         points = 10
9     else:
10        points = 11

```

kedua point bot dan player diawali dengan nilai 0, kode diatas ini adalah point dari kartu dimana poin kartu 2-10 yaitu sesuai ranknya, jack,queen,king poinnya 10, dan ace poinnya 11.

```

1         if player_points == bot_points or (bot_points > 21 and player_points
2         > 21):
3             winner = "Draw"
4         elif player_points > 21 or (bot_points <= 21 and bot_points >
5         player_points):
6             winner = "Bot"
7         elif bot_points > 21 or (player_points <= 21 and player_points >
8         bot_points):
9             winner = "Player"
10        else:
11            # Pilih pemenang berdasarkan total poin tertinggi
12            winner = "Player" if player_points > bot_points else "Bot"
13            # Tampilkan hasil permainan pada frame terakhir
14            cv2.putText(frame, f"The winner is: {winner}", (220,180), cv2.
15            FONT_HERSHEY_SIMPLEX, 2, (255, 255, 255), 3)

```

kode diatas ini adalah untuk menentukan pemenangnya siapa, jika poin player dan bot sama atau poin player dan bot keduanya melebihi 21 maka hasilnya draw, jika poin player melebihi 21 atau poin bot kurang dari 21 dan lebih besar dari player maka pemenangnya adalah bot, jika poin bot melebihi 21 atau poin player kurang dari 21 dan lebih besar dari bot maka pemenangnya adalah player.

4 Hasil

Hasil dari deteksi kartu ini dapat dimainkan dengan baik berikut adalah hasilnya :

4.1 Hasil Dataset

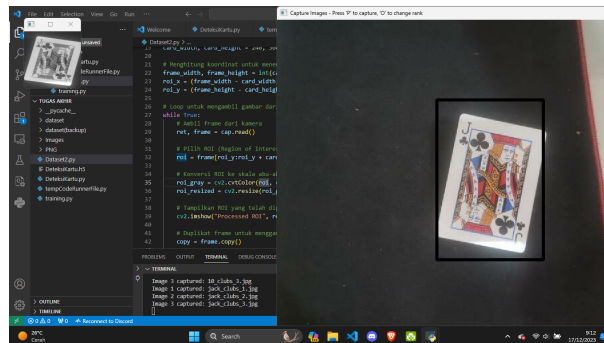


Figure 1: Ketika pengambilan Dataset

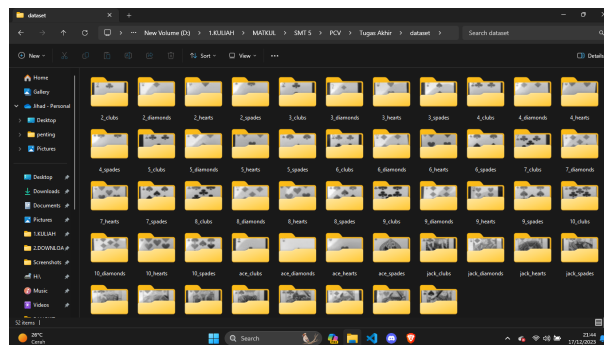


Figure 2: Hasil Dataset

4.2 Hasil Training

Hasil Training adalah berupa file dengan format .h5 yang nantinya akan diload model di deteksi kartu

4.3 Hasil Deteksi Kartu

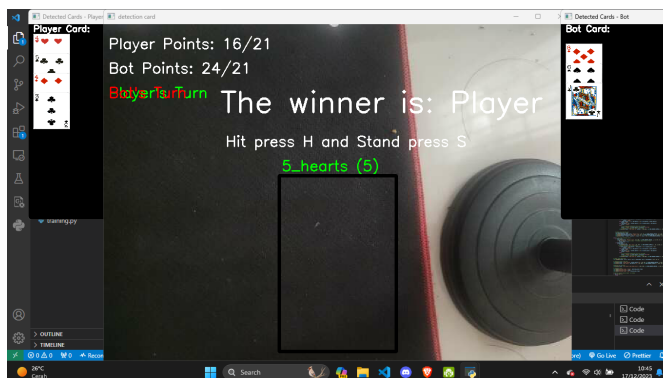


Figure 3: Ketika permainan dimenangkan player

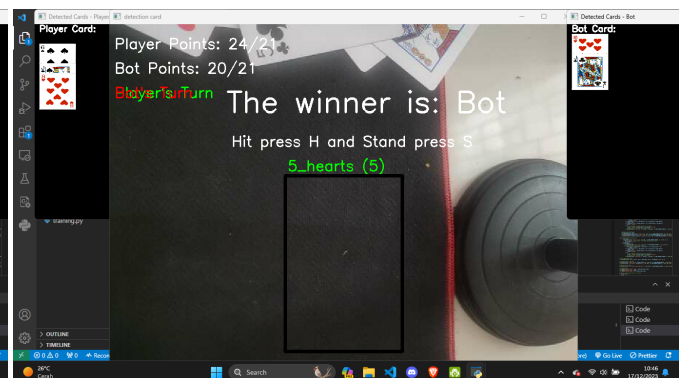


Figure 4: Ketika permainan dimenangkan player

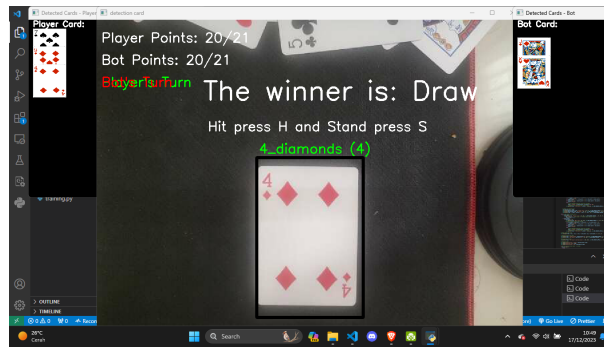


Figure 5: Ketika permainan seri

5 Kesimpulan

Projek deteksi kartu yang saya buat terdapat 3 file pemograman yaitu dataset, training, dan deteksi kartu.

1. Dataset : Ini berfungsi untuk pengambilan gambar yang gambarnya dikonversi ke warna abu-abu dan ukuran gambar diperkecil. Nantinya gambar yang sudah diambil akan digunakan untuk training.
2. Training : Training ini digunakan untuk melatih dan menerapkan model Convolutional Neural Network (CNN) dalam konteks deteksi dan klasifikasi citra
3. Deteksi Kartu : Setelah ditraining, deteksi kartu ini akan load model yang sudah ditraining oleh training agar hasil kartu yang ditraining dapat dideteksi
4. Permainan Blackjack : Cara kerja permainan ini yaitu :
 - Urutan giliran pertama diacak
 - Hit untuk mengambil kartu dan menambahkan point dan stand berarti tidak mengambil kartu
 - Bot akan hit/mengambil kartu secara random tetapi akan stand jika poinnya lebih dari 17
 - Point tidak boleh sampai melebihi 21 tetapi harus lebih besar dari bot
 - Jika poin player dan bot sama atau point keduanya melebihi dari 21 maka akan seri