TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
KHOA CƠ KHÍ CHẾ TẠO MÁY

HCMUTE

BÁO CÁO CUỐI KÌ
MÔN HỌC: TRÍ TUỆ NHÂN TẠO

Phân loại côn trùng sử dụng mạng thần kinh tích chập (CNN)

GVHD: PGS.TS Nguyễn Trường Thịnh

MHP: ARIN337629_22_2_08

Họ và tên: Nguyễn Hữu Chí

Mssv: 20146479

Lớp: Sáng thứ 2, tiết 1 - 4

Hồ Chí Minh, tháng 5 năm 2023

MỤC LỤC

CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI

1.1 Giới thiệu đề tài

Phân loại côn trùng bằng mạng thần kinh tích chập (CNN) là một lĩnh vực nghiên cứu quan trọng trong thị giác máy tính. Nó tập trung vào việc sử dụng mạng thần kinh tích chập để nhận dạng và phân loại các loài côn trùng dựa trên dữ liệu hình ảnh.

Côn trùng là một nhóm động vật đa dạng và phong phú, có vai trò quan trọng trong hệ sinh thái và quy trình sinh thái trên Trái đất. Tuy nhiên, việc phân loại côn trùng truyền thống dựa trên hình ảnh đòi hỏi sự chuyên gia hóa cao và tốn nhiều thời gian. Vì vậy, áp dụng các phương pháp tự động như mạng CNN có thể giúp tăng tốc độ và độ chính xác trong quá trình phân loại côn trùng.

Mạng CNN là một loại mạng thần kinh nhân tạo được thiết kế đặc biệt để xử lý thông tin hình ảnh. Với khả năng tự động học và trích xuất đặc trưng từ dữ liệu hình ảnh, mạng CNN đã đạt được thành công đáng kể trong nhiều ứng dụng thị giác máy tính, bao gồm cả việc phân loại côn trùng.

Trong đề tài này, chúng ta sẽ tìm hiểu về quy trình phân loại côn trùng bằng mạng CNN, bao gồm các bước chuẩn bị dữ liệu, xây dựng mạng CNN, đào tạo mô hình và kiểm tra hiệu suất. Chúng ta cũng sẽ khám phá các phương pháp tiền xử lý dữ liệu và tinh chỉnh mô hình để đạt được kết quả tốt nhất trong việc phân loại côn trùng.

Việc áp dụng mạng CNN vào phân loại côn trùng có thể mang lại nhiều lợi ích trong nghiên cứu và ứng dụng thực tế. Nó có thể hỗ trợ trong việc theo dõi và đánh giá sự biến đổi của các loài côn trùng trong tự nhiên, nghiên cứu sinh thái học, bảo vệ môi trường và phát triển nông nghiệp thông minh.

1.2 Mục đích nghiên cứu

Mục đích của nghiên cứu này là sử dụng mạng thần kinh tích chập (CNN) để phân loại côn trùng dựa trên hình ảnh một cách tự động và chính xác hơn. Nghiên cứu nhằm tạo ra một phương pháp đơn giản và hiệu quả để nhận dạng và phân loại các loài côn trùng trong quan trắc môi trường, nghiên cứu sinh thái và các lĩnh vực khác liên quan đến côn trùng.

1.3 Phương pháp nghiên cứu

Phương pháp nghiên cứu trong đề tài này là sử dụng mạng thần kinh tích chập (CNN) để phân loại côn trùng. CNN là một kiến trúc mạng thần kinh được thiết kế đặc biệt cho việc xử lý và phân loại ảnh. Quá trình nghiên cứu bao gồm các bước sau:

- Thu thập dữ liệu: Các ảnh của các loại côn trùng khác nhau được thu thập từ nguồn tài liệu hoặc bộ dữ liệu có sẵn.
- Tiền xử lý dữ liệu: Các ảnh thu thập được được chuẩn hóa và tiền xử lý để loại bỏ nhiễu

và đảm bảo chất lượng dữ liệu.

- Xây dựng mạng CNN: Mạng CNN được xây dựng với các lớp tích chập, lớp tổng hợp và lớp kết nối đầy đủ. Các lớp này giúp mạng học các đặc trưng cần thiết để phân loại côn trùng.

- Huấn luyện mạng: Dữ liệu ảnh đã được gán nhãn được sử dụng để huấn luyện mạng CNN. Quá trình huấn luyện giúp mạng học cách phân loại các loại côn trùng dựa trên các đặc trưng đã học.

- Đánh giá và điều chỉnh: Mạng CNN được đánh giá bằng cách sử dụng dữ liệu kiểm tra riêng để đo lường hiệu suất phân loại. Nếu cần thiết, mạng có thể được điều chỉnh và huấn luyện lại để cải thiện hiệu suất.

- Kiểm tra và ứng dụng: Mạng CNN đã được huấn luyện có thể được áp dụng để phân loại côn trùng trên các ảnh mới. Kết quả được đánh giá và kiểm tra để đảm bảo tính chính xác và hiệu quả của phương pháp.

1.4 Thông tin đề tài

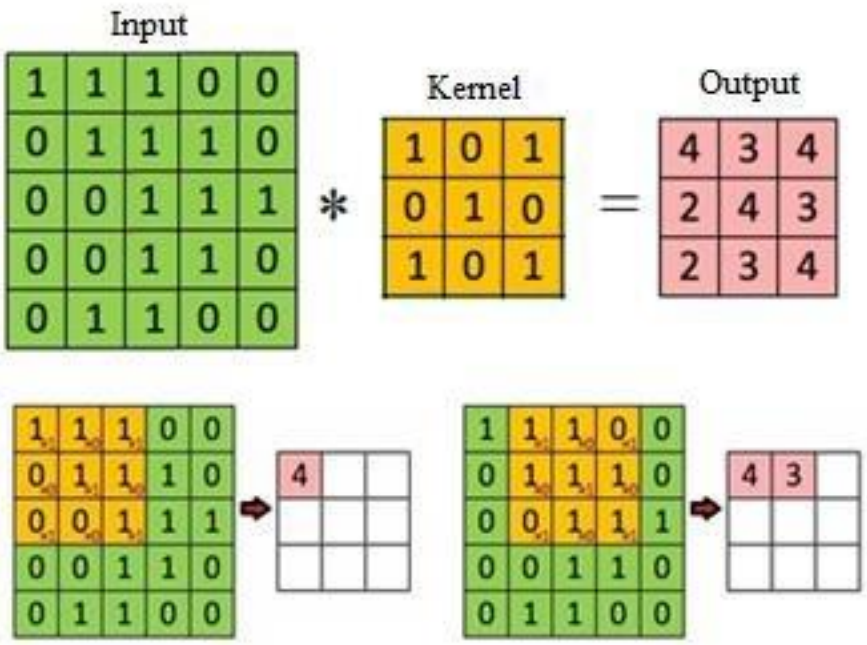
- Dữ liệu:
https://drive.google.com/drive/folders/1nbmIr0OdBbcyMHXf8VPSYAJf9FUEoo8R?usp=sharing

- Github: https://github.com/Chi68P1/AI

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 Mạng thần kinh tích chập (CNN)

Mạng thần kinh tích chập (Convolutional Neural Network - CNN) là một loại kiến trúc mạng thần kinh sử dụng chủ yếu trong việc xử lý và phân loại ảnh. Nó được thiết kế đặc biệt để học các đặc trưng cục bộ từ dữ liệu hình ảnh thông qua việc áp dụng các lớp tích chập và lớp tổng hợp.



Hình 1: Ma trận Convolutional

2.2 Các thành phần cơ bản của CNN

Các thành phần cơ bản của mạng thần kinh tích chập (CNN) bao gồm:

- Lớp đầu vào (Input Layer): Lớp này nhận dữ liệu hình ảnh đầu vào và truyền chúng đến các lớp tiếp theo.

- Lớp tích chập (Convolutional Layer): Lớp này thực hiện phép tích chập giữa ma trận đặc trưng đầu vào và các bộ lọc (kernel) để tạo ra các đặc trưng cục bộ. Mỗi bộ lọc nhận vào một phần nhỏ của đầu vào và áp dụng phép tích chập để tạo ra đầu ra.

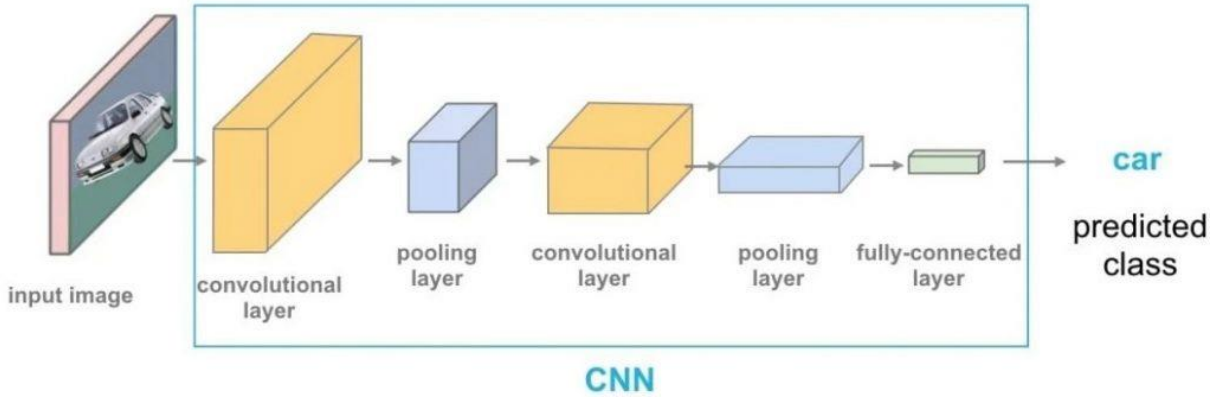
- Lớp kích hoạt (Activation Layer): Lớp này áp dụng một hàm kích hoạt phi tuyến tính lên đầu ra của lớp tích chập. Hàm kích hoạt thông thường được sử dụng là ReLU (Rectified Linear Unit) để giới thiệu tính phi tuyến vào mô hình.

- Lớp tổng hợp (Pooling Layer): Lớp này thực hiện tổng hợp (downsampling) trên các đặc trưng đã được tạo ra bởi lớp tích chập. Phương pháp tổng hợp thường sử dụng là tổng hợp theo giá trị tối đa (max pooling) hoặc tổng hợp theo giá trị trung bình (average pooling) trong một vùng cụ thể.

- Lớp kết nối đầy đủ (Fully Connected Layer): Lớp này kết nối mỗi nút đầu vào với tất cả các nút đầu ra trong lớp này. Nó thực hiện việc phân loại cuối cùng bằng cách học các quan hệ tuyến tính giữa các đặc trưng đã được trích xuất từ các lớp trước đó.

- Lớp đầu ra (Output Layer): Lớp này đưa ra dự đoán cuối cùng dựa trên đầu ra của lớp kết nối đầy đủ. Số lượng nút trong lớp này tương ứng với số lượng lớp phân loại khác nhau trong bài toán.

Bên cạnh các thành phần cơ bản này, CNN còn sử dụng các phép kích hoạt (activation functions) để đưa ra các quyết định phi tuyến tính và giúp mạng học các mô hình phức tạp hơn. Các phép kích hoạt phổ biến bao gồm ReLU (Rectified Linear Unit), sigmoid và tanh.



Hình 2: Các lớp cơ bản của mạng CNN

CHƯƠNG 3: XÂY DỰNG MÔ HÌNH CNN

3.1 Thu thập và xử lý dữ liệu

- Thu thập ảnh từ Google: chúng tôi đã tìm kiếm và tải xuống các ảnh côn trùng từ Google và các nguồn tài liệu uy tín khác. Sử dụng extension của google để quá trình này có thể thực hiện nhanh và dễ dàng hơn

- Loại bỏ những tấm hình không phù hợp: Sau khi thu thập, chúng tôi đã xem xét kỹ các tấm hình và loại bỏ những tấm không phù hợp hoặc không liên quan đến loài côn trùng mà chúng tôi muốn phân loại. Bước này giúp tạo ra một tập dữ liệu chất lượng hơn để huấn luyện mạng CNN.

- Đặt tên cho các tấm hình theo loài: Để thuận tiện cho quá trình huấn luyện, chúng tôi đã đặt tên cho mỗi tấm hình theo loài côn trùng tương ứng. Ví dụ: kien1.jpg, borua1.jpg, v.v. Điều này giúp xác định một cách rõ ràng nhãn (label) của mỗi ảnh để mạng CNN có thể học và phân loại chính xác.

Tổng cộng, chúng tôi đã thu thập hơn 1500 ảnh và xử lý dữ liệu côn trùng bằng cách tải xuống ảnh từ các nguồn tài liệu đáng tin cậy, loại bỏ những tấm không phù hợp, đặt tên theo loài và tiền xử lý dữ liệu nếu cần thiết. Tập dữ liệu này đã sẵn sàng để được sử dụng trong quá trình huấn luyện mạng CNN để phân loại côn trùng dựa trên ảnh.

3.2 Xây dựng và huấn luyện mô hình

1. Import các thư viện cần thiết:

- **listdir** và **asarray** từ module **os** và **numpy** để làm việc với các tệp và mảng.
- **save** từ module **numpy** để lưu trữ dữ liệu ảnh và nhãn vào tệp.
- **img_to_array** và load_img từ module **keras.utils.image_utils** và **keras.utils** để chuyển đổi ảnh thành mảng numpy và tải ảnh từ đường dẫn.

```
from os import listdir
from numpy import asarray
from numpy import save
from keras.utils.image_utils import img_to_array
from keras.utils import load_img
```

2. Kết nối Google Drive:

- Import module drive từ google.colab.
- Gắn kết Google Drive vào Colab để truy cập đến tệp dữ liệu.

```
from google.colab import drive
drive.mount('/content/drive')
```

3. Định nghĩa đường dẫn thư mục chứa ảnh côn trùng.

```
folder='/content/drive/MyDrive/Colab Notebooks/Insect2/'
```

4. Tạo danh sách rỗng để lưu trữ các ảnh và nhãn.

```
photos,labels = list(),list()
```

5. Lặp qua từng tệp trong thư mục:

- Xác định nhãn dựa trên tên tệp.
- Tải ảnh và chuyển đổi thành mảng numpy.
- Thêm ảnh và nhãn tương ứng vào danh sách.

```
for file in listdir(folder):

  if file.startswith('ong'):
    output=0.0
  if file.startswith('kien'):
    output=1.0
  if file.startswith('buom'):
    output=2.0
  if file.startswith('bocap'):
    output=3.0
  if file.startswith('vesau'):
    output=4.0
  if file.startswith('chuonchuon'):
    output=5.0
  if file.startswith('bohung'):
    output=6.0
  if file.startswith('chauchau'):
    output=7.0
  if file.startswith('demen'):
    output=8.0
  if file.startswith('nhen'):
    output=9.0
  if file.startswith('gian'):
    output=10.0
  if file.startswith('muoi'):
    output=11.0
  if file.startswith('borua'):
    output=12.0
  if file.startswith('bongua'):
    output=13.0
  if file.startswith('ruoi'):
    output=14.0

  photo = load_img(folder+file,target_size=(100,100))
  photo = img_to_array(photo)
  photos.append(photo)
  labels.append(output)
```

6. Chuyển đổi danh sách ảnh và nhãn thành mảng numpy.

```
photos = asarray(photos)
labels = asarray(labels)
```

7. Lưu trữ dữ liệu ảnh và nhãn vào Google Drive.

```
save('/content/drive/MyDrive/Colab Notebooks/photo_insect.npy',photos)
save('/content/drive/MyDrive/Colab Notebooks/label_insect.npy',labels)
```

8. Sử dụng train_test_split từ module sklearn.model_selection để chia dữ liệu thành tập huấn luyện và tập kiểm tra.

```
train_x, test_x, train_y,  test_y = train_test_split(photos,labels,
test_size=0.3, train_size=0.7)
```

9. Chuyển đổi kiểu dữ liệu và chuẩn hóa giá trị ảnh trong tập huấn luyện và tập kiểm tra.

```
train_x = train_x.astype('float32')
train_x = train_x/255
test_x = test_x.astype('float32')
test_x = test_x/255
```

10. Chuyển đổi nhãn thành dạng one-hot vector.

```
from keras.utils import to_categorical

y_train = to_categorical(train_y)
y_test = to_categorical(test_y)
```

11. Xây dựng mô hình mạng neural:

```
from keras import Sequential,Model,Input
from keras.layers import Dense,Flatten,Dropout, Conv2D,
MaxPooling2D,Normalization
from keras.optimizers import Adam
from keras.layers import LeakyReLU

batch_size = 64
epochs = 20
classes = 15
```

- Sử dụng Sequential từ module keras để tạo mô hình tuần tự.

```
model = Sequential()
```

- Thêm các lớp Conv2D, LeakyRelu và MaxPooling2D để thực hiện các phép convolution, giảm tắc nghẽn và max pooling.

```
model.add(Conv2D(32,kernel_size=(3,3),activation='linear',input_shape=(1
00,100,3),padding='same'))
model.add(LeakyReLU(alpha=0.1))
model.add(MaxPooling2D((2,2),padding='same'))

model.add(Conv2D(64,(3,3),activation='relu',padding='same'))
model.add(LeakyReLU(alpha=0.1))
model.add(MaxPooling2D((2,2),padding='same'))

model.add(Conv2D(128,(3,3),activation='relu',padding='same'))
model.add(LeakyReLU(alpha=0.1))
model.add(MaxPooling2D((2,2),padding='same'))
```

- Sử dụng lớp Flatten để làm phẳng đầu ra của các lớp trước.

```
model.add(Flatten())
```

- Thêm các lớp fully connected (Dense) với hàm kích hoạt tương ứng. Dropout được sử dụng để ngẫu nhiên bỏ qua một số đơn vị đầu ra trong quá trình huấn luyện, nhằm tránh overfitting.

```
model.add(Dense(700,activation='linear'))
model.add(Dropout(0.5))
model.add(Dense(classes,activation='softmax'))
```

- Biên dịch mô hình với hàm mất mát và thuật toán tối ưu.

```
from keras.backend import categorical_crossentropy

model.compile(loss='categorical_crossentropy', optimizer=Adam(),metrics
=['accuracy'])
```

12. Huấn luyện mô hình với tập huấn luyện.

```
train = model.fit(train_x,y_train,batch_size,epochs=20,verbose=1)
test_loss, test_acc = model.evaluate(test_x, y_test)
```

13. Đánh giá mô hình trên tập kiểm tra và in ra độ chính xác và hàm mất mát.

```
print("Accuracy:", test_acc)
print('Loss:', test_loss)
```

14. Lưu trữ mô hình đã huấn luyện vào Google Drive.

```
model.save('/content/drive/MyDrive/Colab Notebooks/Predict_insect.h5')
```

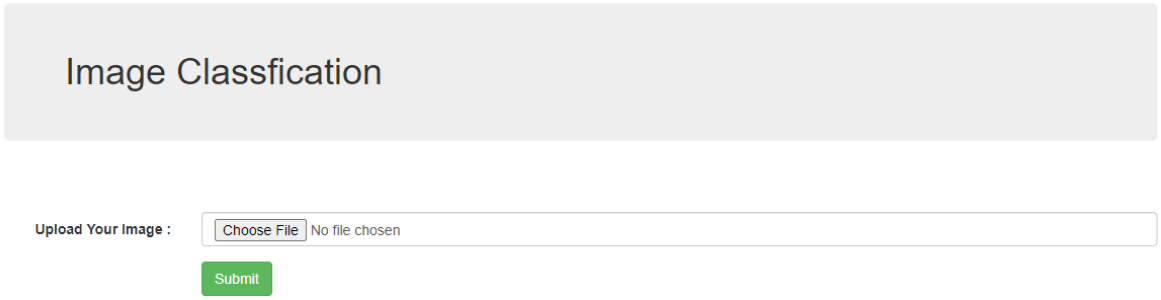
3.3 Xây dựng Web App

Xây dựng giao diện web người dùng để kiểm tra mô hình

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Insect Classification</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.
css">
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"><
/script>
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js
"></script>
</head>
<body>

<div class="container">
  <h1 class="jumbotron bg-primary">Image Classfication</h1>
  <br><br>
  <form class="form-horizontal" action="/submit" method="post"
enctype="multipart/form-data">

    <div class="form-group">
      <label class="control-label col-sm-2" for="pwd">Upload Your Image
:</label>
      <div class="col-sm-10">
        <input type="file" class="form-control" placeholder="Hours
Studied"  name="my_image" id="pwd">
      </div>
    </div>

    <div class="form-group">
      <div class="col-sm-offset-2 col-sm-10">
        <button type="submit" class="btn btn-success">Submit</button>
      </div>
    </div>
  </form>

  {% if prediction %}
  <img src="{{img_path}}" height="400px" width="400px">
  <h2>  Prediction   : <i> {{prediction}} </i></h2>

  {% endif %}

</div>
```

```
</body>
</html>
```

Web hiển thị:



Liên kết với Web App và test mô hình

```
from flask import Flask, render_template, request
from keras.models import load_model
from keras.utils import load_img, img_to_array
import numpy as np
app = Flask(__name__)

dic = {0 : 'Đây là con ong', 1 : 'Đây là con kiến', 2 : 'Đây là con
bướm', 3 : 'Đây là con bọ cạp', 4 : 'Đây là con ve sầu',
5 : 'Đây là con chuồn chuồn', 6 : 'Đây là con bọ hung', 7 : 'Đây là con
châu chấu', 8 : 'Đây là con dế mèn', 9 : 'Đây là con nhện',
10 : 'Đây là con gián',11 : 'Đây là con muỗi', 12 : 'Đây là con bọ rùa',
13 : 'Đây là con bọ ngựa', 14 : 'Đây là con ruồi'}

model = load_model('Predict_insect.h5')

model.make_predict_function()

def predict_label(img_path):
  i = load_img(img_path, target_size=(100,100))
  i = img_to_array(i)/255.0
  i = i.reshape(1, 100,100,3)
  p = model.predict(i)
  predicted_class = np.argmax(p, axis=1)
  return dic[predicted_class[0]]


# routes
@app.route("/", methods=['GET', 'POST'])
def main():
  return render_template("index.html")
```

```
@app.route("/about")
def about_page():
  return "Please subscribe  Artificial Intelligence Hub..!!!"

@app.route("/submit", methods = ['GET', 'POST'])
def get_output():
  if request.method == 'POST':
    img = request.files['my_image']
    img_path = "static/" + img.filename
    img.save(img_path)

    p = predict_label(img_path)

  return render_template("index.html", prediction = p, img_path =
img_path)

if __name__ =='__main__':
  #app.debug = True
  app.run(debug = True)
```

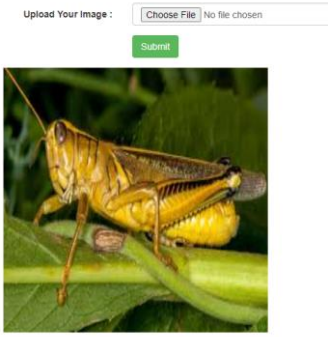
CHƯƠNG 4: KẾT QUẢ VÀ ĐÁNH GIÁ

4.1 Kết quả

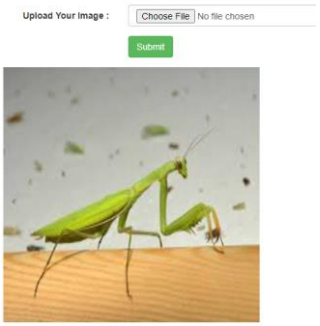
Tỉ lệ chính xác của mô hình là 66.9 %. Mất mát là 2.79

```
Epoch 13/20
17/17 [==============================] - 1s 30ms/step - loss: 0.1561 - accuracy: 0.9600
Epoch 14/20
17/17 [==============================] - 1s 31ms/step - loss: 0.1347 - accuracy: 0.9600
Epoch 15/20
17/17 [==============================] - 0s 29ms/step - loss: 0.1207 - accuracy: 0.9771
Epoch 16/20
17/17 [==============================] - 0s 29ms/step - loss: 0.0646 - accuracy: 0.9876
Epoch 17/20
17/17 [==============================] - 1s 30ms/step - loss: 0.0787 - accuracy: 0.9800
Epoch 18/20
17/17 [==============================] - 1s 30ms/step - loss: 0.0435 - accuracy: 0.9924
Epoch 19/20
17/17 [==============================] - 1s 30ms/step - loss: 0.0299 - accuracy: 0.9952
Epoch 20/20
17/17 [==============================] - 0s 29ms/step - loss: 0.0282 - accuracy: 0.9933
15/15 [==============================] - 1s 13ms/step - loss: 2.7976 - accuracy: 0.6689
Accuracy: 0.6688888669013977
Loss: 2.7976412773132324
```

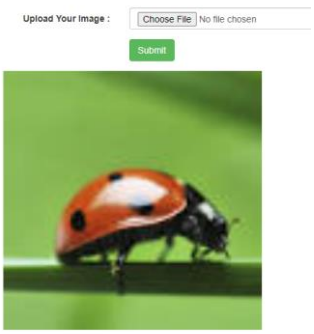
Tiến hành test mô hình với 9 hình bất kì từ tập dữ liệu train thì thu được kết quả tỉ lệ chính xác là 7/9



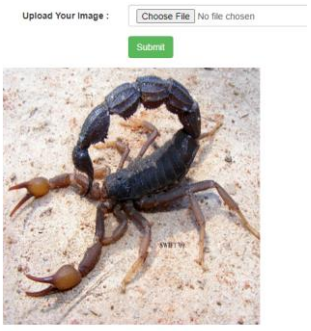
Prediction : *Đây là con châu chấu*



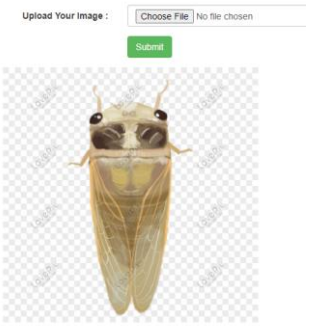
Prediction : *Đây là con bọ ngựa*



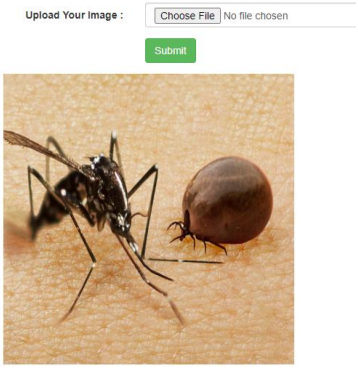
Prediction : *Đây là con bọ rùa*



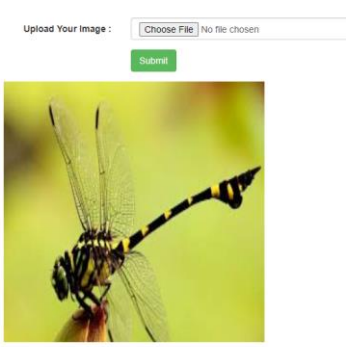
Prediction : *Đây là con bọ cạp*



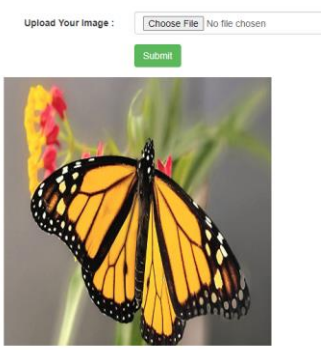
Prediction : *Đây là con ve sầu*



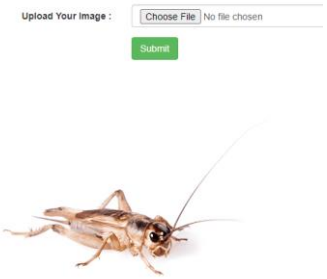
Prediction : *Đây là con gián*



Prediction : *Đây là con chuồn chuồn*

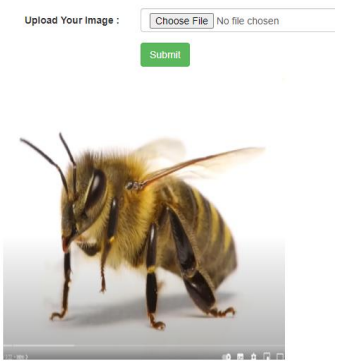


Prediction : *Đây là con kiến*

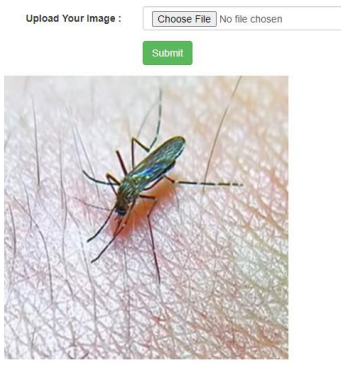


Prediction : *Đây là con dế mèn*

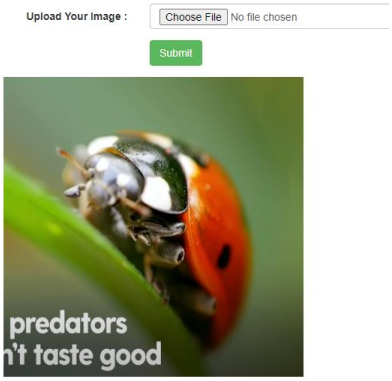
Tiến hành test mô hình với 9 hình bất kì ngoài tập dữ liệu train thì thu được tỉ lệ chính xác là 3/9



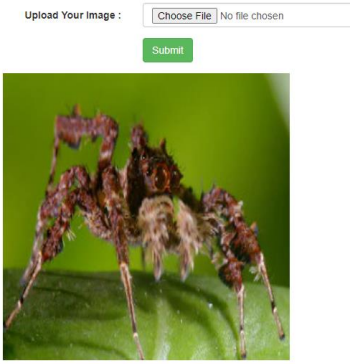
Prediction : *Đây là con dế mèn*



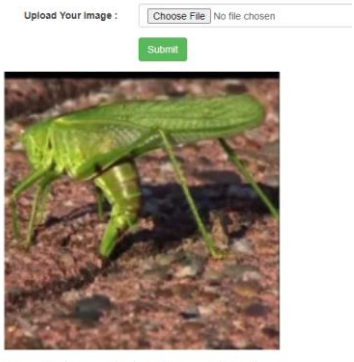
Prediction : *Đây là con muỗi*



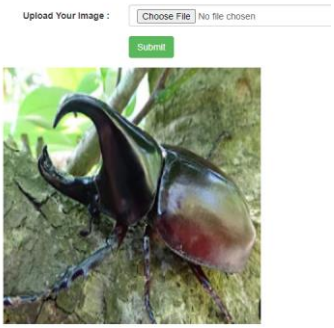
Prediction : *Đây là con bọ ngựa*



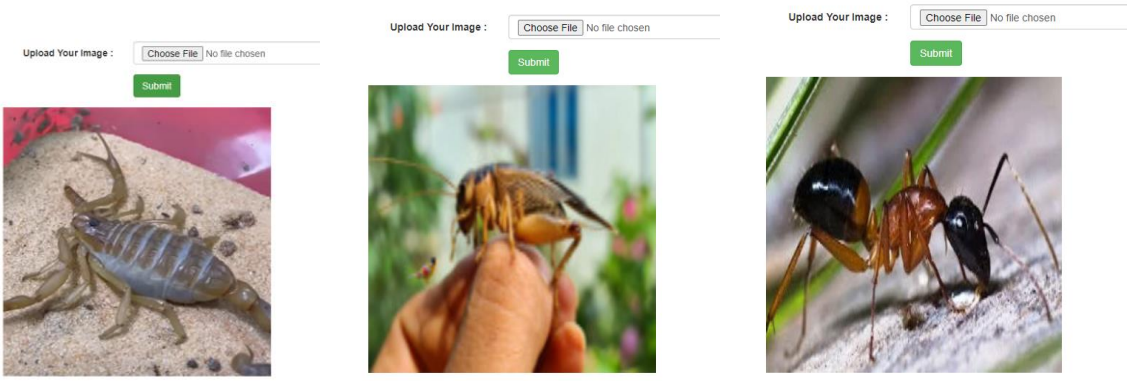
Prediction : *Đây là con ve sầu*



Prediction : *Đây là con bọ hung*



Prediction : *Đây là con bọ hung*



4.2 Đánh giá

Sau quá trình kiểm tra và đánh giá mô hình Convolutional Neural Network (CNN), chúng tôi đã nhận thấy rằng mô hình còn một số hạn chế trong khả năng phân loại côn trùng. Cần phải cải thiện tập dữ liệu và tối ưu các lớp để tăng độ chính xác của mô hình.

KẾT LUẬN

Trong nghiên cứu này, chúng tôi đã thực hiện việc phân loại côn trùng bằng cách sử dụng mạng thần kinh tích chập (CNN).

Qua quá trình thực hiện và đánh giá, chúng tôi nhận thấy mô hình CNN đã đạt được một số thành công trong việc phân loại côn trùng. Với việc sử dụng các lớp tích chập và lớp pooling, mô hình đã học được các đặc trưng quan trọng từ hình ảnh và tạo ra các biểu đồ đặc trưng phức tạp để phân loại côn trùng.

Tuy nhiên, chúng tôi cũng nhận thấy một số hạn chế trong khả năng phân loại của mô hình. Một số loài côn trùng có sự tương đồng về hình dạng hoặc màu sắc, gây khó khăn cho mô hình trong việc phân biệt chúng. Ngoài ra, mô hình cũng không đạt được độ chính xác cao đối với các hình ảnh có chất lượng thấp, nhiễu hoặc góc nhìn khác nhau.

Để cải thiện khả năng phân loại côn trùng của mô hình, chúng tôi đề xuất một số hướng nghiên cứu tiếp theo. Đầu tiên, có thể tăng cường dữ liệu bằng cách thu thập thêm hình ảnh từ nhiều nguồn và ghi nhãn chính xác để mô hình có thể học được nhiều đặc trưng khác nhau. Thứ hai, việc sử dụng mô hình CNN nâng cao hơn như ResNet hoặc InceptionNet có thể cung cấp hiệu suất tốt hơn trong việc nhận diện đặc trưng phức tạp. Cuối cùng, việc tinh chỉnh các siêu tham số và thực hiện kiểm định chéo có thể cải thiện độ chính xác của mô hình.

Tổng kết lại, mô hình CNN đã chứng tỏ khả năng phân loại côn trùng khá tốt. Tuy nhiên, vẫn còn cần thêm nỗ lực để nâng cao độ chính xác và khả năng phân loại của mô hình trong các trường hợp khó khăn. Sự tiếp tục nghiên cứu và phát triển mô hình CNN sẽ mang lại những tiến bộ đáng kể trong lĩnh vực phân loại côn trùng.

TÀI LIỆU THAM KHẢO

1. Thuật toán CNN là gì? Thông tin cấu trúc của mạng CNN (fpt.edu.vn)

2. Image Classification using CNN Keras | Full implementation (youtube.com)

3. Image Classification Web App | Flask (youtube.com)