I. Bài toán

Ở Việt Nam đã có nhiều công trình nghiên cứu các hệ thống thông minh ứng dụng Trí tuệ nhân tạo như Bệnh viện thông minh, giao thông thông minh, nhà thông minh, xe tự lái,… Trong đó, ứng dụng nhận diện biển số xe cũng được chú trọng nghiên cứu và đẩy mạnh phát triển tại thị trường xe máy lớn thứ 4 thế giới là Việt Nam. Công nghệ tự động nhận diện biển số xe được ra mắt từ năm 1976 (theo Wikipedia.org) được sử dụng tại nhiều nước phát triển trên thế giới như Australia, Canada, Denmark, United States …. Nó được chính phủ các nước sử dụng để giám sát giao thông tại các nút giao thông quan trọng, tại các trạm thu phí tự động và cũng được các tổ chức sử dụng nhằm mục đích thương mại như xây dựng bãi giữ xe thông minh.

Sau khi đã tìm hiểu qua về những ứng dụng mạnh mẽ mà công nghệ nhận diện biển số xe này mang lại, chúng em tự hỏi liệu có thể tự phát triển một hệ thống của riêng mình dựa vào những công nghệ đã có sẵn. Từ đó bài toán được đặt ra là thiết kế mô hình nhận diện và phân tích kí tự biển số xe máy dựa vào lượng dữ liệu đang có tại bãi giữ xe UIT.

II. Cách giải quyết

Input: Ảnh chụp có chứa biển số

Output: Ảnh các ký tự trên biển số xe trong ảnh đã được gán nhãn

Để giải quyết bài toán đã đưa ra bên trên, chúng em chia làm 3 bước chính sau:

- Xác định vị trí biển số xe trong khung hình.

- Tiền xử lý biển số xe thu được.
- Phân loại và gán nhãn các ký tự có trong biển số xe.

Về xác định vị trí biển số xe trong khung hình, chúng em sử dụng phương pháp Haar cascade để học những đặc trưng quan trọng của một biển số, dữ liệu được thu tại bãi giữ xe trường UIT. Ở đây chúng em sử dụng phần mềm Cascade Trainer GUI cho việc huấn luyện mô hình Haar Cascade.

Tiếp theo, đến bước tiền xử lý. Bước này được thực hiện sau khi đã phát hiện ra vùng chứa biển số xe trong bức ảnh, ta bắt đầu sử dụng các phương pháp xử lý ảnh cơ bản như: chuyển về ảnh đơn kênh, lấy ngưỡng, tìm cạnh, lấy contour để tách được từng vùng có khả năng chứa ký tự trong đó.

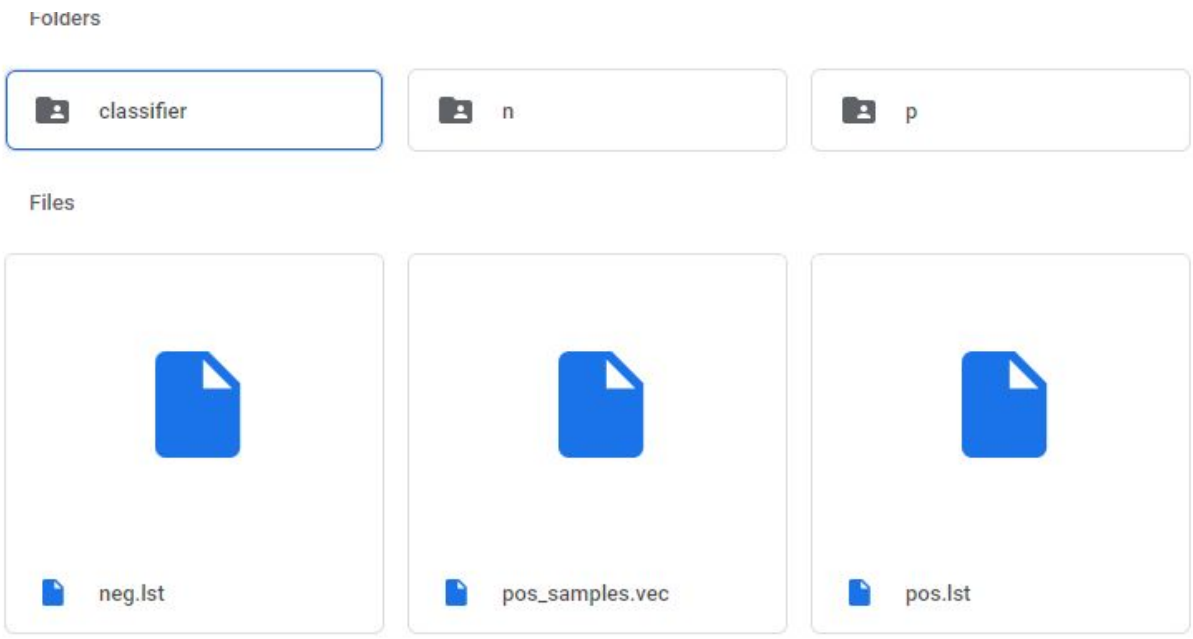
Cuối cùng, chúng em bắt đầu dự đoán xem những vùng vừa tìm được có chứa ký tự nào bằng các mô hình học sâu đã tìm hiểu từ trước. Kết quả dự đoán sẽ được lưu lại và gán nhãn cho vùng vừa dự đoán.

III. Mô tả dữ liệu

Dữ liệu chúng em sử dụng là các biển số xe tại trường UIT

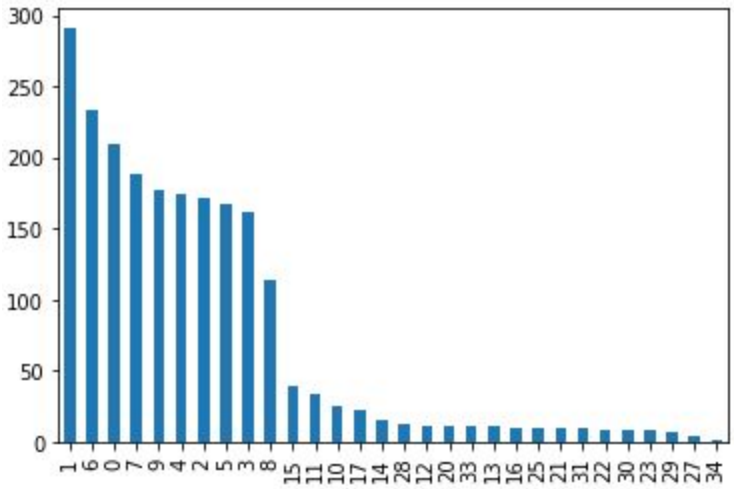
- Dữ liệu cho Haar cascade:

- Positive: 1200 mẫu (300 mẫu gốc, sau đó sử dụng hàm createSample của haar cascade trên opencv để tạo thêm mẫu mời với background khác).
- Negative: 1200 mẫu (ảnh chụp đời thường không có chứa biển số xe.



- Dữ liệu cho mô hình dự đoán ký tự trên biển số (theo hình dưới):

```
Read data done
1     291
6     233
0     210
7     188
9     177
4     174
2     172
5     167
3     162
8     114
15     39
11     34
10     26
17     22
14     16
28     13
12     11
20     11
33     11
13     11
16     10
25     10
21     10
31     10
22      9
30      8
23      8
29      7
27      4
34      2
```



Bảng thống kê số lượng dữ liệu sử dụng

- Cột bên trái là ký tự từ: 0 -> 9 và từ A-Z đã được gán số từ 0 -> 36
- Cột bên phải là số lượng ảnh của mẫu đó.

IV.Kết quả đạt được

Sau khi đã thử qua một số mô hình dự đoán ký tự sử dụng bộ dữ liệu trên, chúng em thấy rằng mô hình SVM cho ra kết quả dự đoán trên tập test set cao hơn: 94.2%.

Hàm training model bằng SVM (ở đây dùng biến thử SVC) sau đó dự đoán thử trên tập test

```
# Phân lớp bằng thuật toán SVM để huấn luyện cho mô hình nhận diện ký tự
def linear_svm(X_train, y_train ,X_test, y_test):
    print("[!] SVM data...")
    Svm = svm.SVC(kernel='linear').fit(X_train, y_train)
    y_pred = Svm.predict(X_test)
    # print(y_pred)
    print("score", metrics.accuracy_score(y_test, y_pred))
    print("[+] Finished")
    return Svm
```

Chọn training SVM

```
svmFlag=True
if svmFlag is True:
  model = linear_svm(X_train, y_train,X_test,y_test)
  # Luu trong so da huan luyen
  save_model(model, "svm", PATH_WEIGHT)
```

```
[!] SVM data...
score 0.9421296296296297
[+] Finished
[+] Saving model to file :  svm.joblib
```

Đối với phía xác định vị trí biển số xe trong khung hình sử dụng Haar Cascade cho ra kết quả còn thấp: 79.7%, điều này có thể giải thích được do số lượng dữ liệu còn ít, hình dạng biển báo còn đa dạng và góc chụp khác nhau.

```
print("Images :",countImg) #số ảnh đọc được
print("Detected images :",countDetc) #số ảnh detect được biển số xe
print("Correct images :",countAcc) #số ảnh trích xuất chính xác biển số xe
print("Score :",countAcc/countImg) #phần trăm chính xác
```

```
Images : 74
Detected images : 67
Correct images : 59
Score : 0.7972972972972973
```

Các mẫu thành công:

Một số mẫu bị lỗi:

Nhận sai kí tự


7
8
2
71-B2
3
2
8
9
9
328.99

V.So sánh

Để lựa chọn mô hình cho ra kết quả dự đoán ký tự cao nhất, chúng em sử dụng 3 mô hình sau: KNN, CNN, SVM và cho ra những kết quả trên test set sau.

KNN: với K cao nhất là 3, score : 0.9

Hàm training model bằng KNN, trước tiên sẽ tìm hệ số K tốt nhất (2,26) sau đó dự đoán thử trên tập test

```python
# Phân lớp bằng thuật toán KNN để huấn luyện cho mô hình nhận diện ký tự
def KNN(X_train, y_train,X_test,y_test):
    print("[!] KNN data...")
    # tìm K tốt nhất
    best_score=0
    best_K=1
    for k in range(2,26):
      Knn = KNeighborsClassifier(n_neighbors=k)
      Knn.fit(X_train, y_train)
      y_pred = Knn.predict(X_test)
      # print(y_pred)
      # print(len(y_test))
      Scr=metrics.accuracy_score(y_test, y_pred)

      print(k," ", Scr)
      if Scr > best_score:      #nếu score tốt hơn score tốt nhất trước đó, cập nhật hệ số K
        best_K=k
        best_score=Scr
    print("Best K is :"+str(best_K))
    Knn = KNeighborsClassifier(n_neighbors=best_K)
    Knn.fit(X_train,y_train)
    y_pred = Knn.predict(X_test)
    print("score", metrics.accuracy_score(y_test, y_pred))
    print("[+] Finished")
    return Knn
```

Chọn training KNN

```
knnFlag=True
if knnFlag is True:
  model=KNN(X_train, y_train,X_test,y_test)
  save_model(model, "knn", PATH_WEIGHT)
```

```
[!] KNN data...
2   0.8935185185185185
3   0.9004629629629629
4   0.8935185185185185
5   0.8888888888888888
6   0.8888888888888888
7   0.8865740740740741
8   0.8773148148148148
9   0.875
10   0.8587962962962963
11   0.8634259259259259
12   0.8541666666666666
13   0.8611111111111112
14   0.8518518518518519
15   0.8425925925925926
16   0.8425925925925926
17   0.8356481481481481
18   0.8333333333333334
19   0.8287037037037037
20   0.8333333333333334
21   0.8287037037037037
22   0.8287037037037037
23   0.8263888888888888
24   0.8263888888888888
25   0.8263888888888888
Best K is :3
score 0.9004629629629629
[+] Finished
[+] Saving model to file :  knn.joblib
```

CNN: Tại epochs 30, score: 0.924

Xây dựng model

> Sử dụng 5 layers, hàm loss sử dụng 'categorical_crossentropy', optimizer sử dụng 'adam'

```
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Conv2D, MaxPooling2D
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(28,28,1)))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dense(128, activation='sigmoid'))
model.add(Dense(35, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model.summary()
```

```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_5 (Conv2D)            (None, 26, 26, 32)        320
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 24, 24, 32)        9248
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 12, 12, 32)        0
_________________________________________________________________
flatten_3 (Flatten)          (None, 4608)              0
_________________________________________________________________
dense_5 (Dense)              (None, 128)               589952
_________________________________________________________________
dense_6 (Dense)              (None, 35)                4515
=================================================================
Total params: 604,035
Trainable params: 604,035
Non-trainable params: 0
```

```
[ ] His = model.fit(X_train/255.0, y_train, validation_data=(X_test/255.0, y_test), batch_size=32, epochs=30, verbose=1)
```

Kết quả thu được

```
score = model.evaluate(X_test, y_test, verbose=0)
print("epochs :",30)
print("score :", score[1])
print("[+] Finished")
```

```
epochs : 30
score : 0.9242
[+] Finished
```

SVM: score: 0.94

Hàm training model bằng SVM (ở đây dùng biến thử SVC) sau đó dự đoán thử trên tập test

```
# Phân lớp bằng thuật toán SVM để huấn luyện cho mô hình nhận diện ký tự
def linear_svm(X_train, y_train ,X_test, y_test):
    print("[!] SVM data...")
    Svm = svm.SVC(kernel='linear').fit(X_train, y_train)
    y_pred = Svm.predict(X_test)
    # print(y_pred)
    print("score", metrics.accuracy_score(y_test, y_pred))
    print("[+] Finished")
    return Svm
```

Chọn training SVM

```
svmFlag=True
if svmFlag is True:
  model = linear_svm(X_train, y_train,X_test,y_test)
  # Luu trong so da huan luyen
  save_model(model, "svm", PATH_WEIGHT)
```

```
[!] SVM data...
score 0.9421296296296297
[+] Finished
[+] Saving model to file :  svm.joblib
```

Thông qua kết quả thu được của 3 mô hình trên chúng em lựa chọn SVM cho đề tài của mình với kết quả cho ra cao nhất.

VI.Khó khăn

Do lượng dữ liệu thu được khá ít mà hình dạng biển số lại khá đa dạng nên độ chính xác của của mô hình vẫn chưa cao.

Thời gian hoàn thiện đề tài khá ngắn nến độ hoàn thiện của đề tài chưa được tốt.

VII. Kết luận

Sau những kết quả thu được, đề tài đã mang lại cho chúng em những kiến thức rất đặc biệt, và thu được những thành quả như sau

Tuy vẫn còn những vùng không nhận được và nhận sai do số lượng dữ liệu còn khá ít nhưng chúng em đã tìm những phương pháp phù hợp nhất để cho ra kết quả cao nhất có thể.

Hướng phát triển

- Có thể xây dựng thành hệ thống tự động thu thập dữ liệu biển số xe, góp phần xây dựng bộ dữ liệu biển số xe tại trường UIT thêm phong phú hơn.