

CHUYỂN ĐỔI NGÔN NGỮ KÝ HIỆU THÀNH VĂN BẢN

Nhóm 2

GVHD: Ninh Khánh Duy

NHÓM 2



Bích Quỳnh

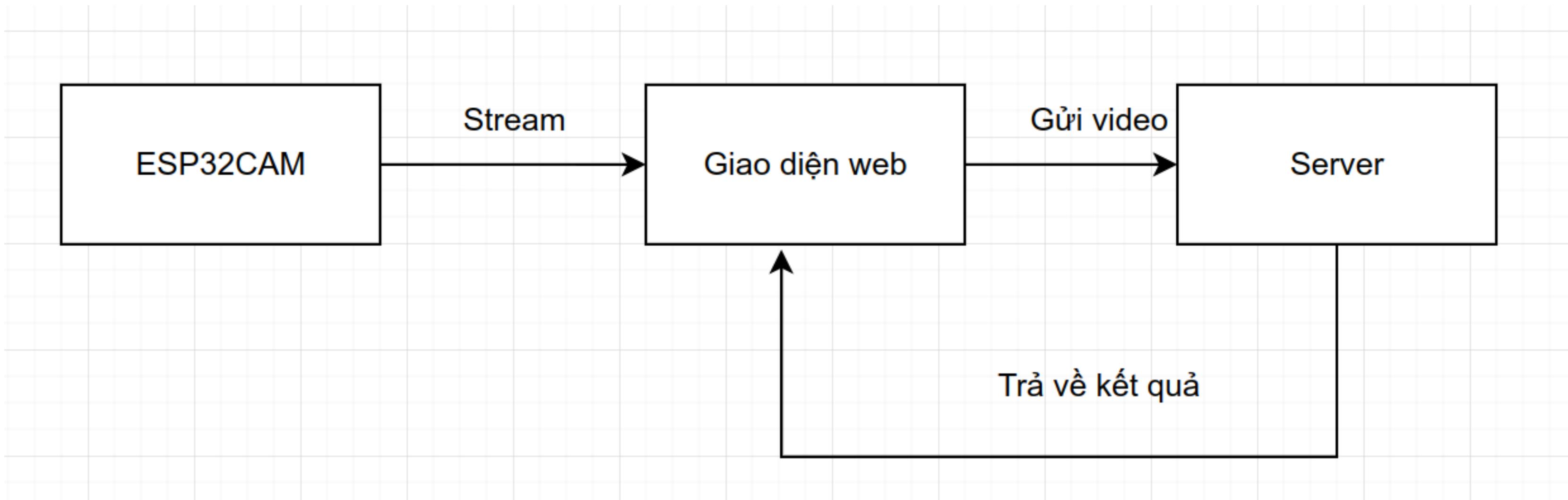


Phước Nhâm



Thanh Hiền

GIỚI THIỆU



Sơ đồ hệ thống

Dữ liệu

-Thu thập dữ liệu:

- Sử dụng esp32cam để thu thập và lưu vào thẻ nhớ.
- Sử dụng camera của laptop

-Làm giàu dữ liệu bằng cách xoay, flip, và thêm padding vào cho video. Từ 1 video gốc, sau khi làm giàu ta có được 14 video khác nhau.

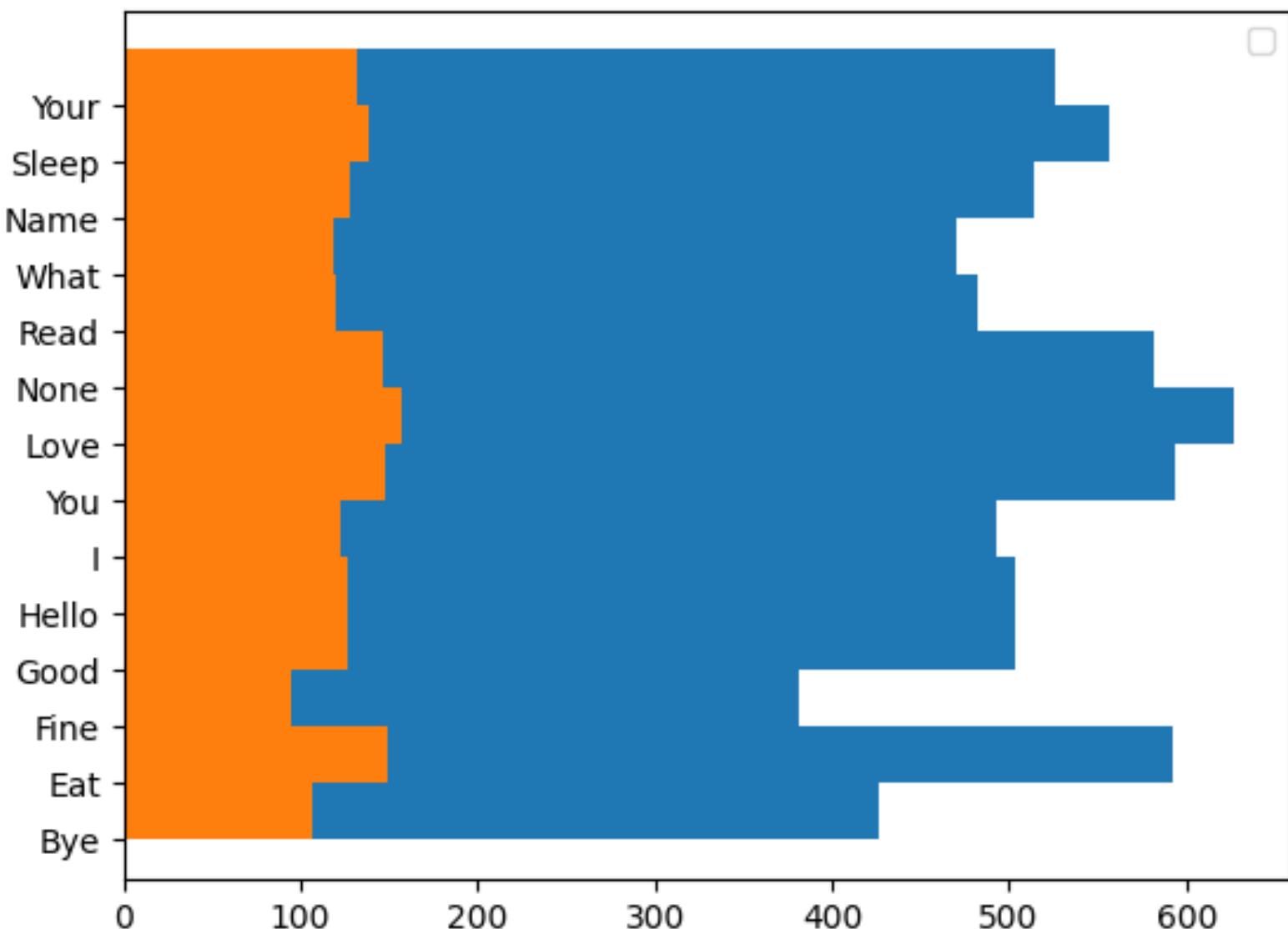
Sau khi làm giàu dữ liệu:



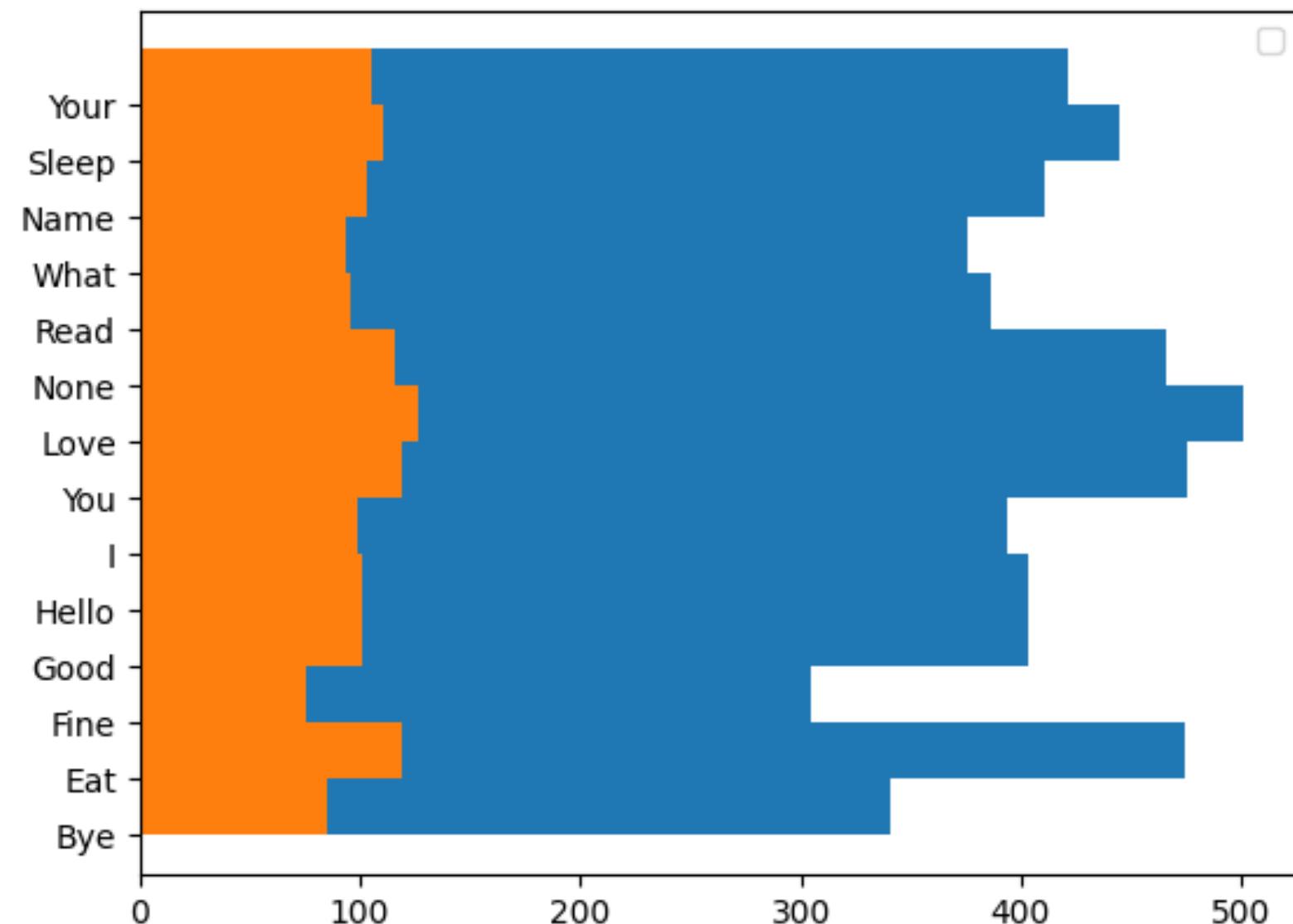
Hành động	Bye	Eat	Fine	Good	Hello	I	You	Love	None	Read	What	Name	Sleep	Your
Số lượng	560	770	546	700	658	742	742	784	728	644	658	616	700	686

PHÂN CHIA DỮ LIỆU

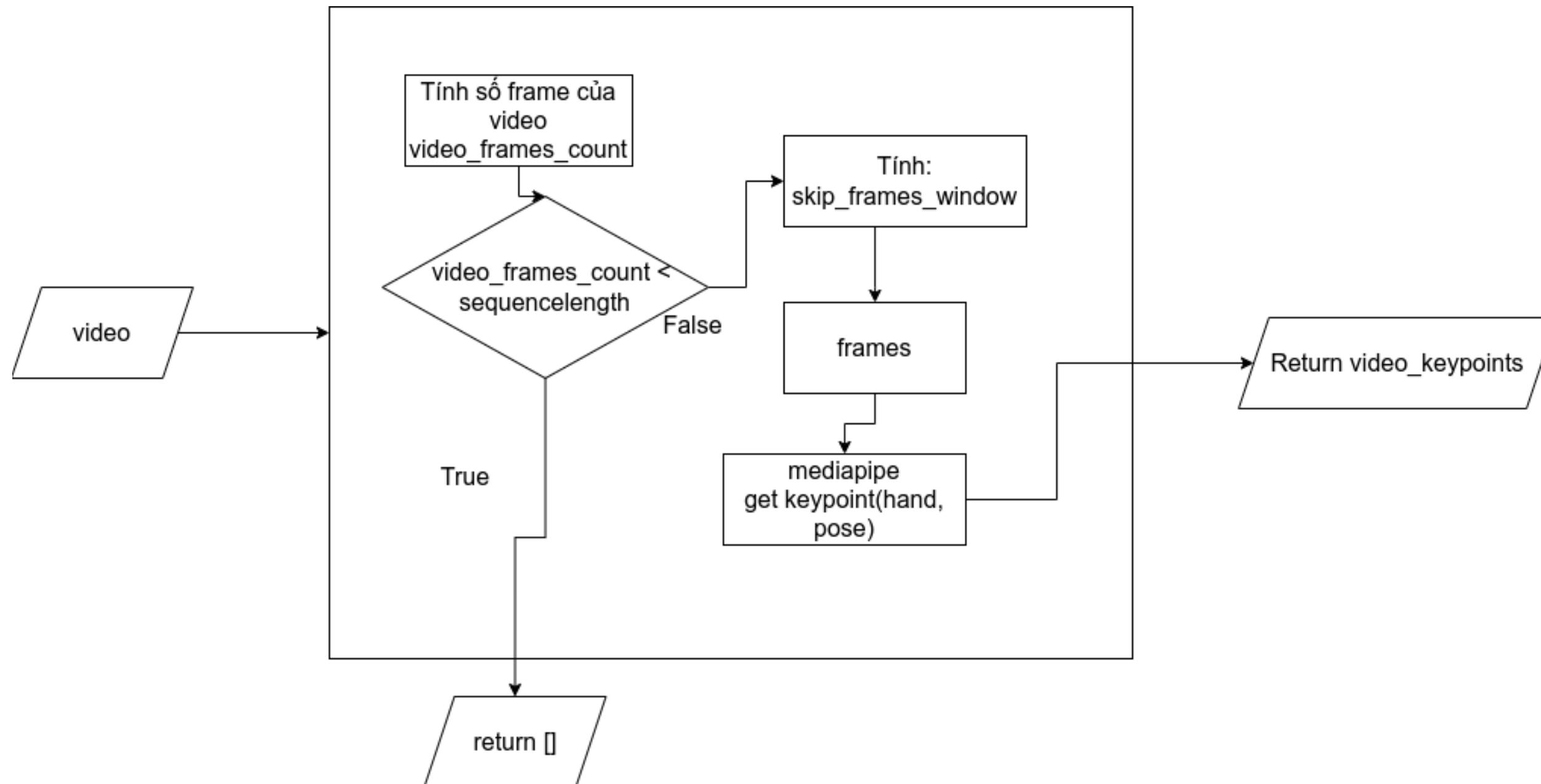
Tỷ lệ Train&val:test là 80:20



Tỷ lệ Train:val là 80:20



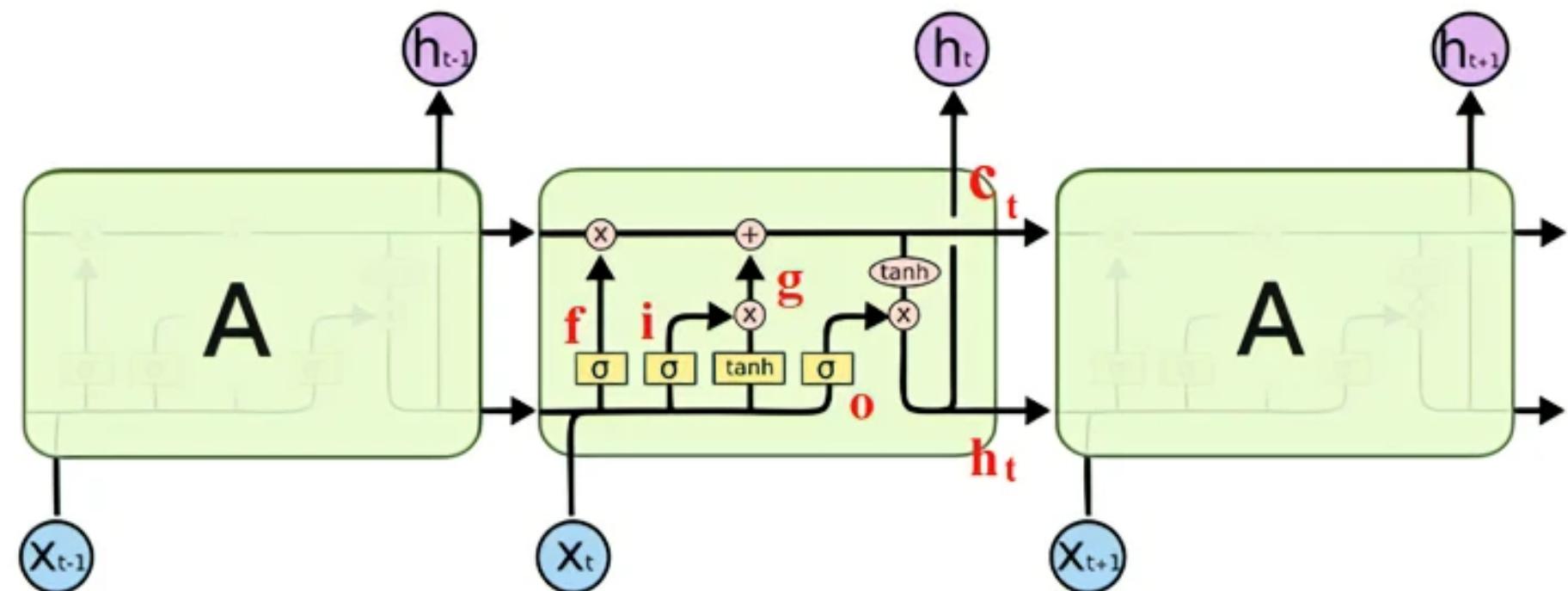
TRÍCH XUẤT ĐẶC TRƯNG



Kích thước của mỗi đặc trưng thu được là 12x300

MODEL VÀ TRAIN

KIẾN TRÚC LỚP LSTM



i là Input gate .

f là forget gate

o là output gate

Hidden state h_t Đây chính là bộ nhớ
của mạng là tổng hợp thông tin của
hidden state trước cộng với input tại
time step t

g là một trạng thái ẩn được tính
dựa trên đầu vào hiện tại x_t và trạng thái
trước h_{t-1}

c_t là bộ nhớ trong của LSTM.

MODEL VÀ TRAIN

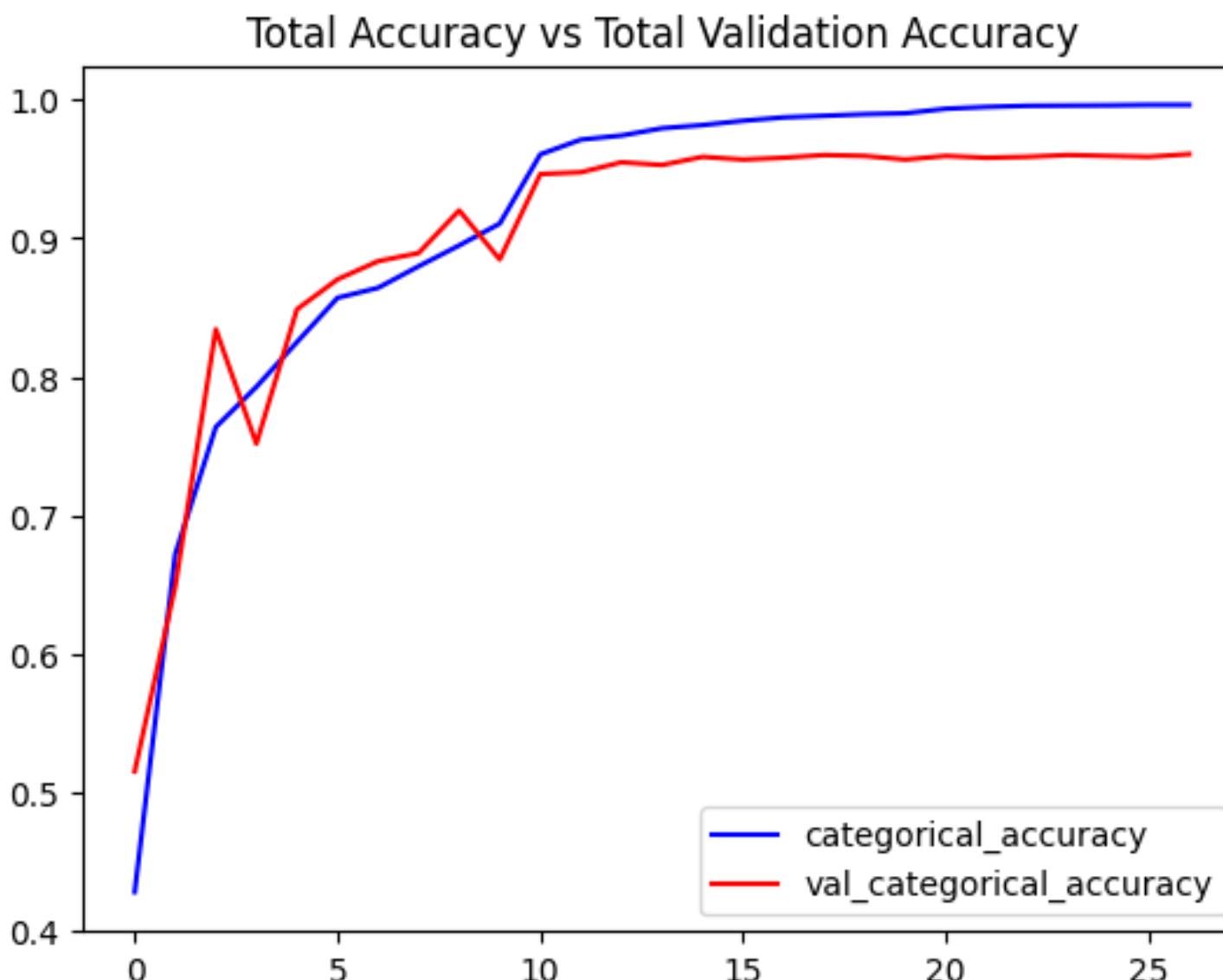
MODEL SỬ DỤNG



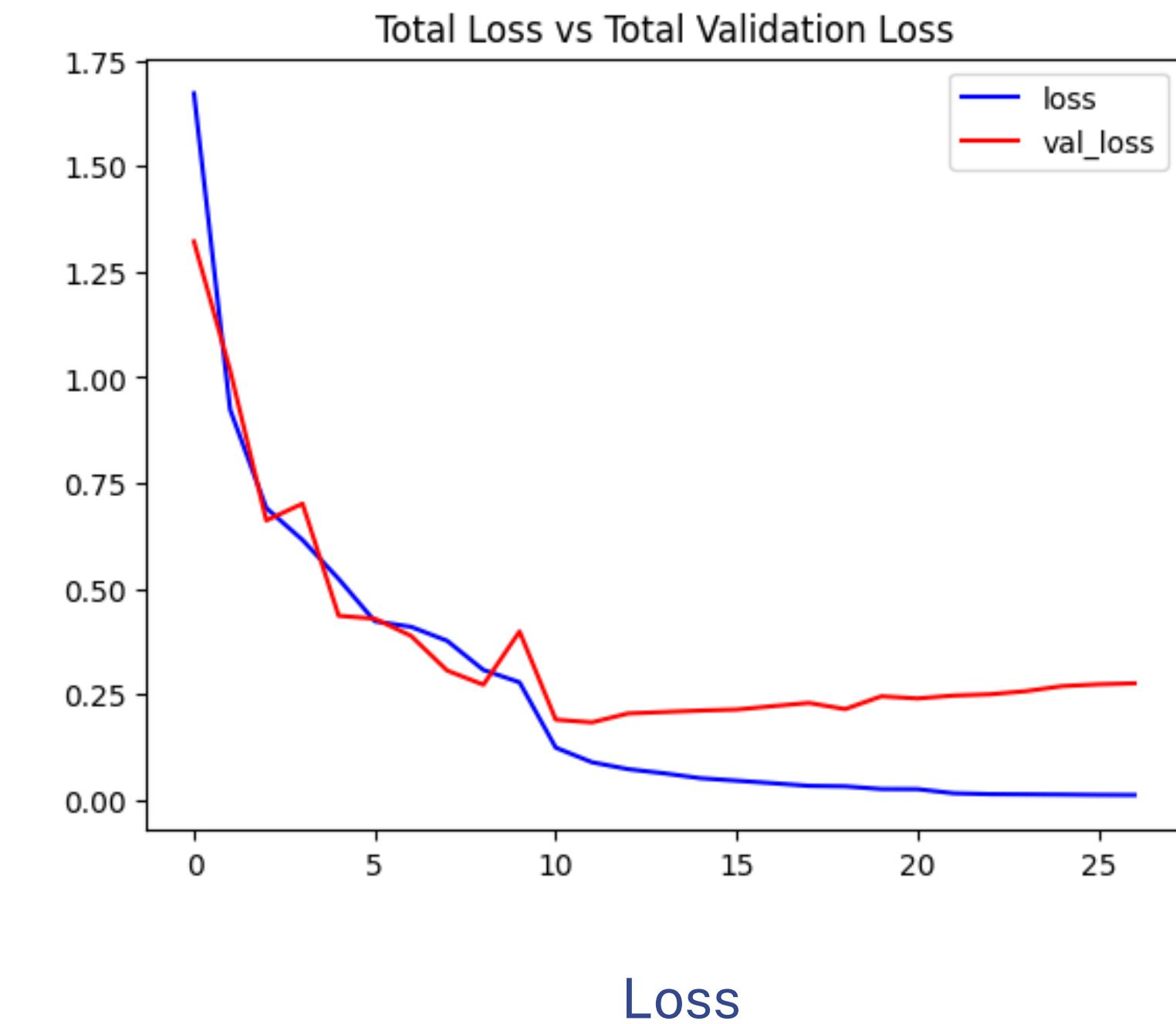
```
1 model = Sequential()
2 model.add(LSTM(64, return_sequences=True, activation='relu',
3     kernel_initializer= initializers.he_normal(),
4     input_shape=(sequence_length,X.shape[-1])))
5 model.add(LSTM(128, return_sequences=True, activation='relu',
6     kernel_initializer= initializers.he_normal()))
7 model.add(LSTM(64, return_sequences=False, activation='relu',
8     kernel_initializer= initializers.he_normal()))
9 model.add(Dense(64, activation='relu',
10    kernel_initializer= initializers.he_normal()))
11 model.add(Dense(32, activation='relu',
12    kernel_initializer= initializers.he_normal()))
13 model.add(Dense(actions.shape[0], activation='softmax',
14    kernel_initializer= initializers.he_normal()))
```

MODEL VÀ TRAIN

KẾT QUẢ



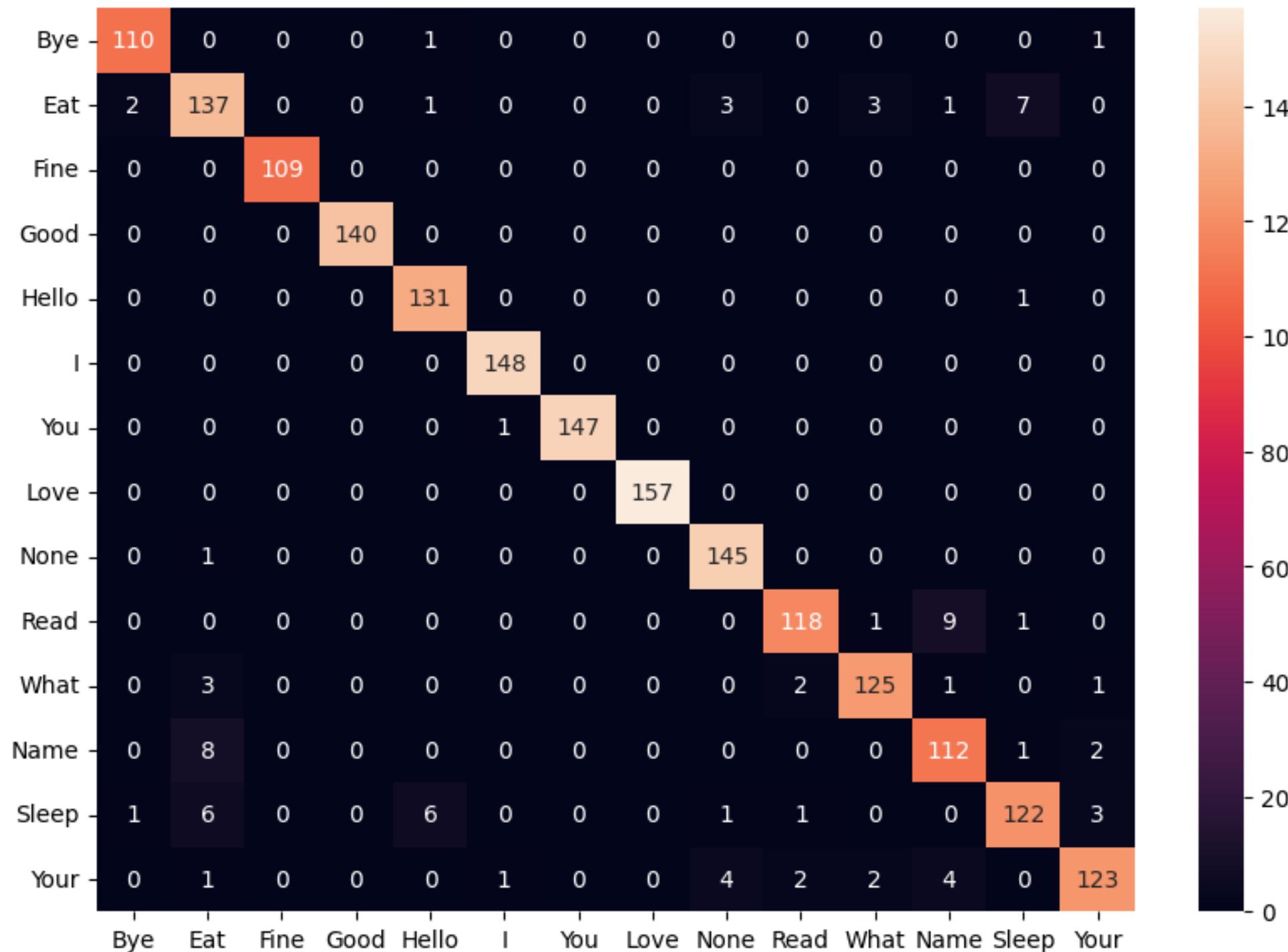
Độ chính xác



Loss

MODEL VÀ TRAIN

KẾT QUẢ



Ma trận nhầm lẫn

SERVER

Được xây dựng bằng Framework Flask

Gồm 2 API xử lí video 1 hành động và video chuỗi hành động:

- + /LSTM: đầu vào là 1 video, kết quả là chuỗi các từ tương ứng với các hành động trong video
- + /LSTM/single: đầu vào là 1 video, kết quả là hành động trong video

Ví dụ:

The screenshot shows a Postman interface with the following details:

- Method:** POST
- URL:** http://nhamcotdo.ddns.net/LSTM
- Body:** form-data (selected)
- Key:** video_file
Value: 2023-04-09_15_03_05.avi

Response:

```
[{"word": "Hello"}, {"word": "Eat"}, {"word": "Eat"}, {"word": "Bye"}, {"word": "Eat"}, {"word": "Fine"}, {"word": "Eat"}, {"word": "Read"}, {"word": "Good"}]
```

WEB FRONT END

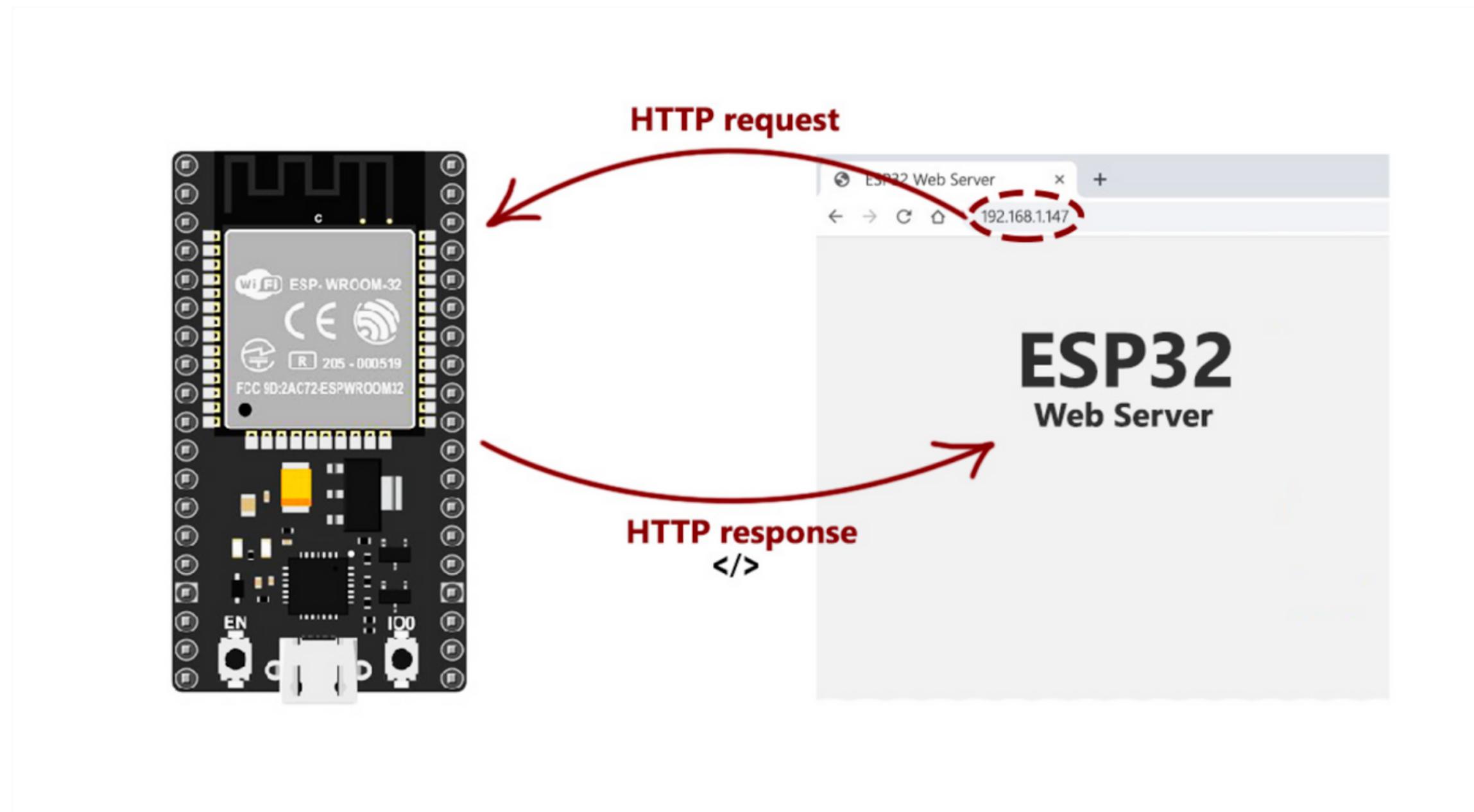
-Sử dụng thư viện DNS server để tạo một DNS server. Khi ESP32 kết nối đến WiFi, tạo ra một DNS Server với tên miền là địa chỉ IP của Wifi và port 80. Sau đó sẽ define mã html và gửi lên url trên(ví dụ <http://192.168.0.12:80>) tạo ra 1 trang FE để tương tác với người dùng.

- Chức năng: web FE sẽ có nhiệm vụ gửi video lên server và hiển thị kết quả.



QUAY VIDEO GỬI LÊN SERVER

Hệ thống sử dụng 1 ESP32CAM để ghi lại hình ảnh, sau đó stream lên giao diện web local. Trên giao diện có các nút bấm để bắt đầu ghi hình và gửi video lên server. Khi click các nút record/stop trên giao diện web , sẽ gửi yêu cầu HTTP (POST request) đến ESP32. Sau khi ghi xong, esp gửi file video lên dns server với url là ip/stop. Trang FE sẽ lấy file video từ dns server và gửi lên api, sau đó lấy kết quả về và hiển thị lên giao diện web.



KẾT QUẢ



THANK YOU
SO MUCH!

