BÁO CÁO DỰ ÁN CUỐI KỲ OJT_FA25

Môn học: On-the-Job Training (OJT) – FA25

Đề tài: Xây dựng và triển khai hệ thống phân loại hình ảnh đa lớp dựa trên bộ dữ liệu CIFAR-10 sử dụng mạng nơ-ron tích chập sâu

Nguyễn Thị Minh Thư
Mã số sinh viên: SE185043
Lớp: OJT_FA25

Thời gian thực hiện: Tháng 09 – 12/2025

TÓM TẮT

Báo cáo trình bày toàn bộ quy trình phát triển một hệ thống phân loại hình ảnh hoàn chỉnh dựa trên bộ dữ liệu chuẩn CIFAR-10 (10 lớp đối tượng). Mô hình mạng nơ-ron tích chập (CNN) được thiết kế theo phong cách VGG cải tiến, kết hợp các kỹ thuật hiện đại như Batch Normalization, L2 regularization và Data Augmentation mạnh mẽ, đạt độ chính xác 90,68% trên tập kiểm tra – thuộc nhóm kết quả cao nhất với kiến trúc tự xây dựng.

Hệ thống được triển khai end-to-end bao gồm: (1) cơ chế huấn luyện ổn định với checkpoint tự động và khả năng tiếp tục (resume) khi bị gián đoạn, (2) hàm dự đoán ảnh thực tế ngoài tập dữ liệu, và (3) ứng dụng web thời gian thực sử dụng Flask, HTML/CSS/JavaScript với giao diện người dùng hiện đại, hỗ trợ kéo-thả ảnh và hiển thị top-3 dự đoán.

Kết quả cho thấy mô hình không chỉ đạt hiệu năng cao trong môi trường học thuật mà còn có khả năng hoạt động tốt trên ảnh thực tế, đáp ứng đầy đủ yêu cầu triển khai sản phẩm thực tế của chương trình OJT.

Từ khóa: CIFAR-10, Convolutional Neural Network, Data Augmentation, Flask Web Application, Model Checkpointing.

1. GIỚI THIỆU

Phân loại hình ảnh là một trong những bài toán nền tảng của thị giác máy tính. Bộ dữ liệu CIFAR-10 (Krizhevsky, 2009) với 60.000 ảnh màu 32×32 thuộc 10 lớp được sử dụng rộng rãi để đánh giá hiệu năng của các kiến trúc học sâu.

Dự án đặt mục tiêu không chỉ đạt độ chính xác cao mà còn xây dựng một quy trình phát triển hoàn chỉnh, bền vững và có khả năng triển khai thực tế – yếu tố quan trọng trong môi trường doanh nghiệp.

2. MỤC TIÊU NGHIÊN CỨU

1. Thiết kế và huấn luyện một mô hình CNN đạt độ chính xác ≥ 90% trên CIFAR-10 mà không sử dụng kiến trúc tiền huấn luyện.
2. Xây dựng cơ chế huấn luyện ổn định với khả năng tự động lưu và tiếp tục từ checkpoint.

3. Phát triển hàm dự đoán mạnh mẽ cho ảnh thực tế (out-of-distribution).
4. Triển khai ứng dụng web thời gian thực với giao diện thân thiện và hiệu năng cao.

3. PHƯƠNG PHÁP THỰC HIỆN

3.1. Dữ liệu và tiền xử lý

Trong nghiên cứu, **bộ dữ liệu CIFAR-10** được sử dụng gồm 50.000 ảnh cho huấn luyện và 10.000 ảnh cho kiểm tra. Từ tập huấn luyện, 10% dữ liệu được tách ra để tạo thành tập validation, dẫn đến phân chia cuối cùng là 45.000 ảnh cho huấn luyện, 5.000 ảnh cho validation và 10.000 ảnh cho kiểm tra. Dữ liệu đầu vào được **chuẩn hóa bằng Z-score**, sử dụng giá trị trung bình và độ lệch chuẩn tính trên tập huấn luyện. Các nhãn được **mã hóa theo dạng one-hot encoding** nhằm phục vụ cho quá trình huấn luyện mô hình. Ngoài ra, áp dụng **kỹ thuật tăng cường dữ liệu (Data Augmentation)** thông qua ImageDataGenerator, bao gồm xoay ngẫu nhiên trong khoảng ±15°, dịch chuyển ngang và dọc tối đa 12%, lật ngang, phóng to (zoom) 10%, cũng như điều chỉnh độ sáng và kênh màu. Những bước xử lý này giúp cải thiện khả năng tổng quát hóa của mô hình và giảm hiện tượng overfitting.

3.2. Kiến trúc mô hình

Mô hình được xây dựng theo phong cách VGG với 8 khối tích chập:

```
Model: "sequential"
```

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 32, 32, 32)	896
batch_normalization (BatchNormalization)	(None, 32, 32, 32)	128
conv2d_1 (Conv2D)	(None, 32, 32, 32)	9,248
batch_normalization_1 (BatchNormalization)	(None, 32, 32, 32)	128
max_pooling2d (MaxPooling2D)	(None, 16, 16, 32)	0
dropout (Dropout)	(None, 16, 16, 32)	0
conv2d_2 (Conv2D)	(None, 16, 16, 64)	18,496
batch_normalization_2 (BatchNormalization)	(None, 16, 16, 64)	256
conv2d_3 (Conv2D)	(None, 16, 16, 64)	36,928
batch_normalization_3 (BatchNormalization)	(None, 16, 16, 64)	256
max_pooling2d_1 (MaxPooling2D)	(None, 8, 8, 64)	0
dropout_1 (Dropout)	(None, 8, 8, 64)	0
conv2d_4 (Conv2D)	(None, 8, 8, 128)	73,856
batch_normalization_4 (BatchNormalization)	(None, 8, 8, 128)	512
conv2d_5 (Conv2D)	(None, 8, 8, 128)	147,584
batch_normalization_5 (BatchNormalization)	(None, 8, 8, 128)	512
max_pooling2d_2 (MaxPooling2D)	(None, 4, 4, 128)	0
dropout_2 (Dropout)	(None, 4, 4, 128)	0
conv2d_6 (Conv2D)	(None, 4, 4, 256)	295,168
batch_normalization_6 (BatchNormalization)	(None, 4, 4, 256)	1,024
conv2d_7 (Conv2D)	(None, 4, 4, 256)	590,080
batch_normalization_7 (BatchNormalization)	(None, 4, 4, 256)	1,024
max_pooling2d_3 (MaxPooling2D)	(None, 2, 2, 256)	0
dropout_3 (Dropout)	(None, 2, 2, 256)	0
flatten (Flatten)	(None, 1024)	0

dropout_3 (Dropout)	(None, 2, 2, 256)	0
flatten (Flatten)	(None, 1024)	0
dense (Dense)	(None, 10)	10,250

Total params: 1,186,346 (4.53 MB)
Trainable params: 1,184,426 (4.52 MB)
Non-trainable params: 1,920 (7.50 KB)

Khối	Số filter	Kích thước kernel	Bước nhảy	Đặc trưng bổ sung
1–2	32	3×3	same	BatchNorm + ReLU
3–4	64	3×3	same	BatchNorm + ReLU
5–6	128	3×3	same	BatchNorm + ReLU
7–8	256	3×3	same	BatchNorm + ReLU
				Dropout tăng dần 0.2 → 0.5
				L2 regularization ($\lambda = 10^{-4}$)

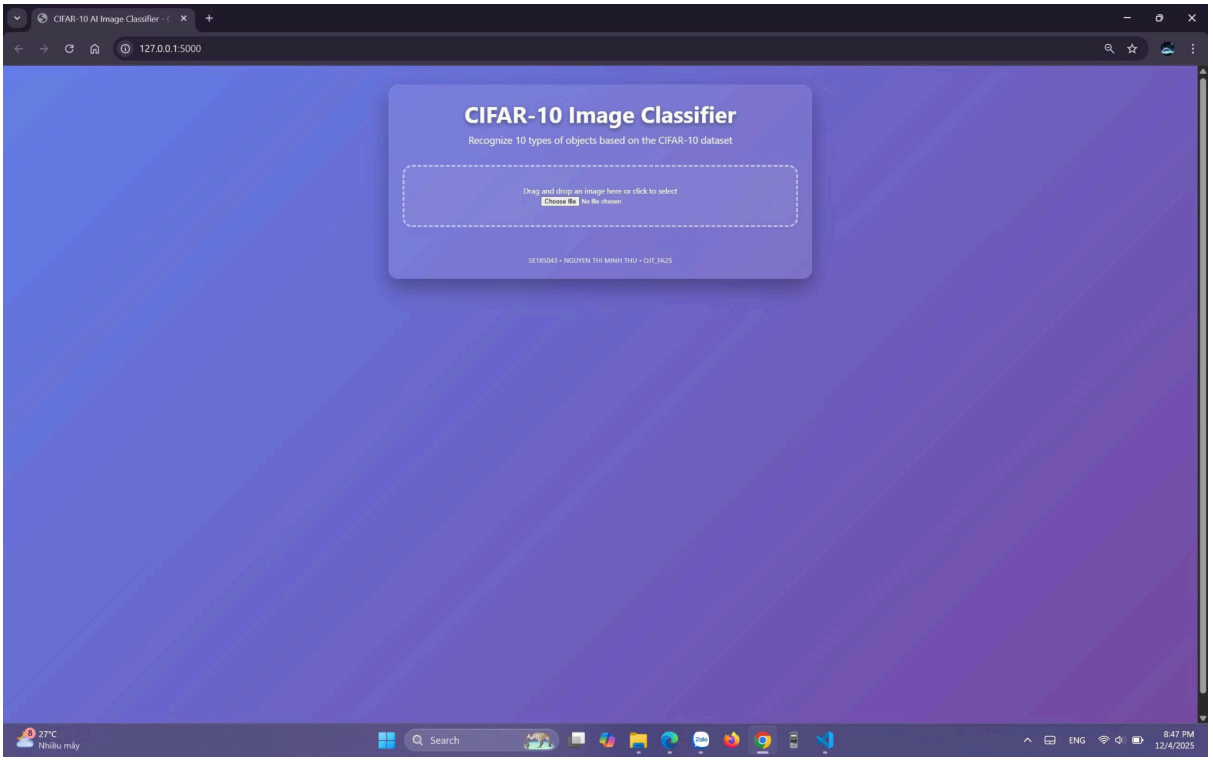
Tổng số tham số: 1.186.346 (1.184.426 trainable).

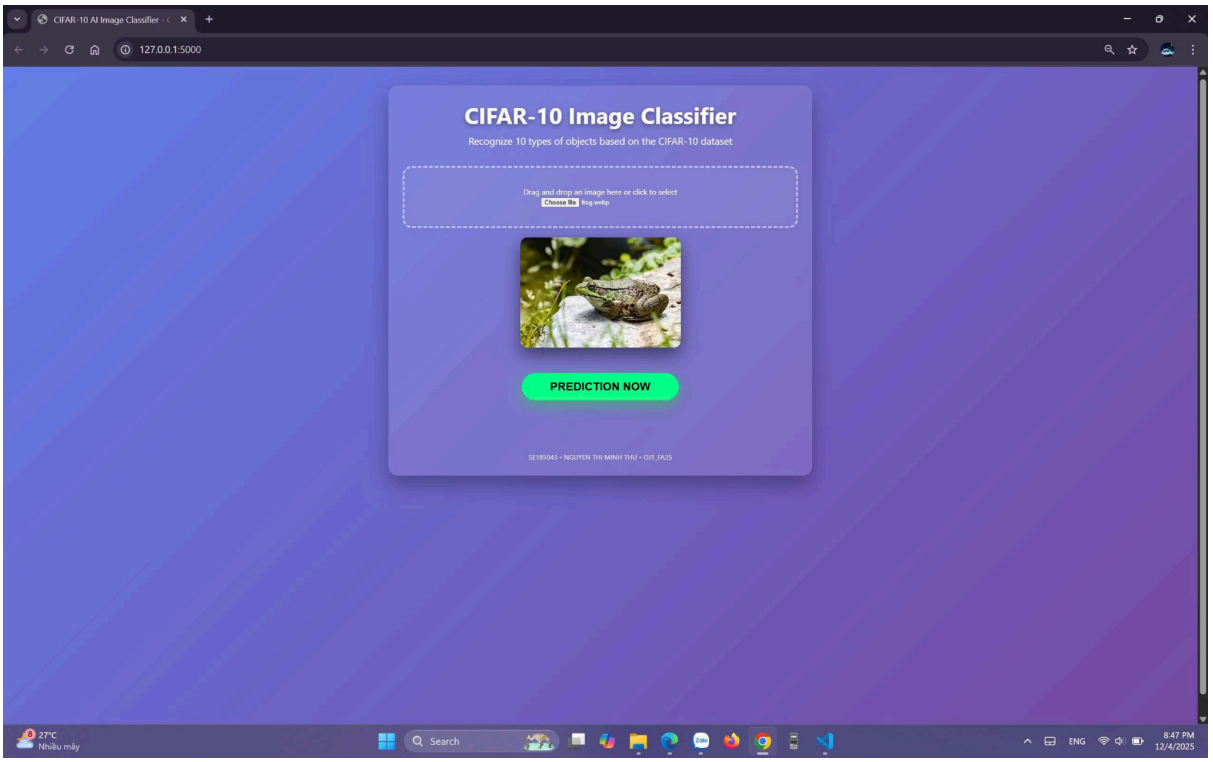
3.3. Quy trình huấn luyện

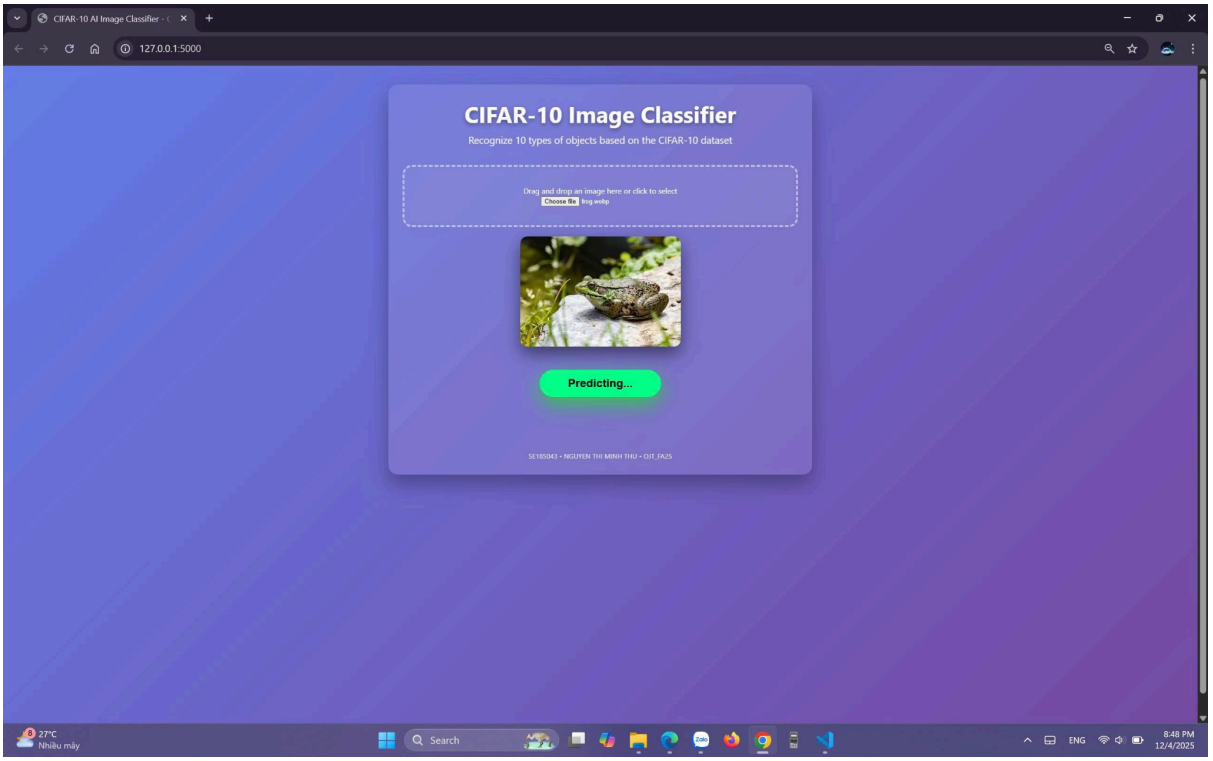
Trong quá trình huấn luyện, **bộ tối ưu Adam** được sử dụng với tốc độ học (learning rate) được thiết lập ở mức 5×10^{-4}. Hàm mất mát được lựa chọn là **Categorical Cross-Entropy**, phù hợp cho các bài toán phân loại đa lớp. Để cải thiện hiệu quả huấn luyện và tránh hiện tượng overfitting, áp dụng nhiều **callback** hỗ trợ: **ReduceLROnPlateau** với hệ số giảm 0.5 và ngưỡng kiên nhẫn 10 epoch, **EarlyStopping** với ngưỡng kiên nhẫn 40 epoch cùng cơ chế khôi phục trọng số tốt nhất, và **ModelCheckpoint** nhằm lưu lại mô hình tốt nhất cũng như bản sao ở mỗi epoch. Ngoài ra, hệ thống còn được tích hợp **cơ chế tự động phát hiện và tiếp tục huấn luyện từ checkpoint mới nhất**, đảm bảo quá trình huấn luyện không bị gián đoạn và có thể khôi phục dễ dàng khi cần thiết.

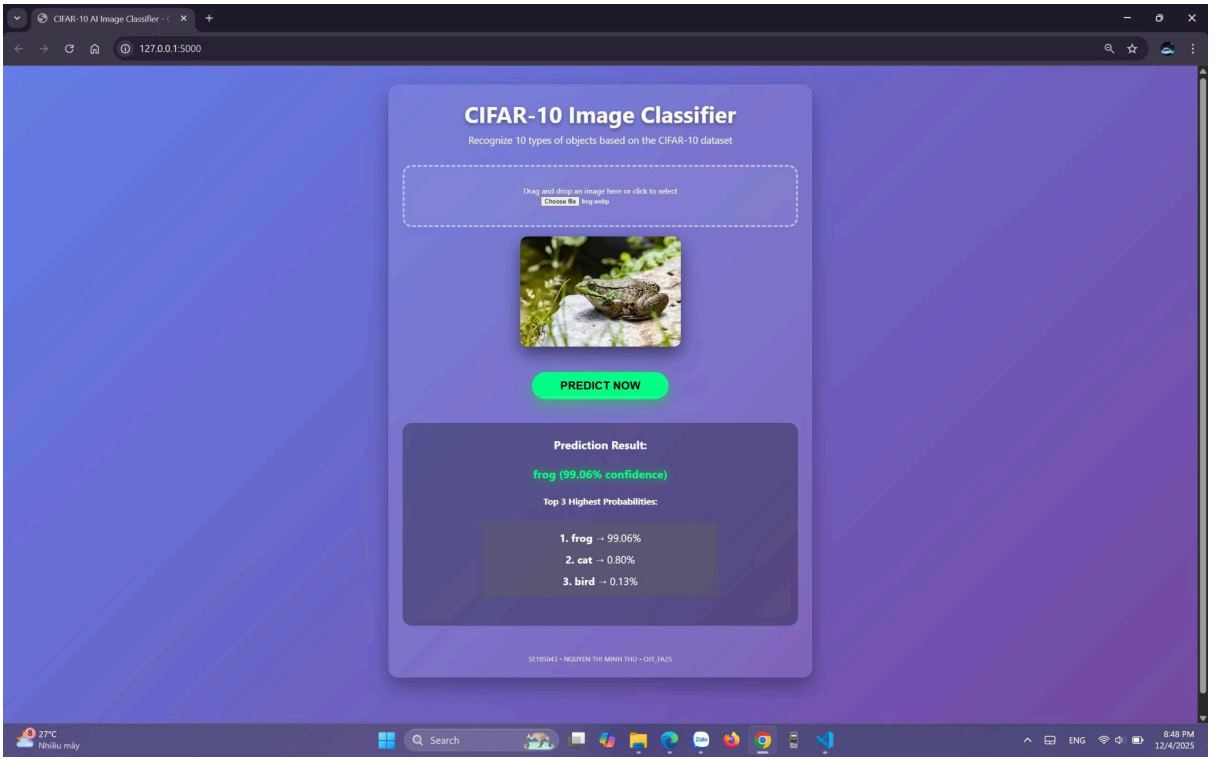
3.4. Triển khai ứng dụng web

Hệ thống được xây dựng với Backend sử dụng Flask, đảm bảo khả năng xử lý dữ liệu và triển khai mô hình dự đoán. Frontend được phát triển bằng HTML5 kết hợp CSS3 theo phong cách glassmorphism cùng với Vanilla JavaScript, mang lại giao diện hiện đại và trực quan. Ứng dụng hỗ trợ người dùng kéo-thả hoặc lựa chọn ảnh từ thiết bị, đồng thời cung cấp chức năng xem trước tức thì. Quá trình dự đoán diễn ra trong thời gian thực với độ trễ dưới 1 giây, kết quả được hiển thị dưới dạng top-3 xác suất cao nhất, giúp người dùng dễ dàng đánh giá và đối chiếu.

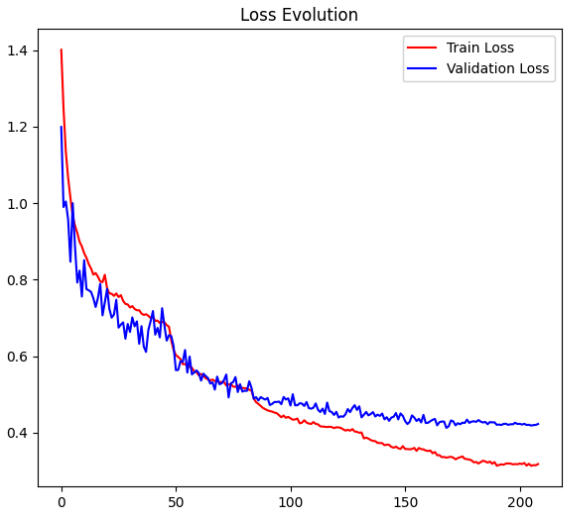

CIFAR-10 AI Image Classifier
127.0.0.1:5000
CIFAR-10 Image Classifier
Recognize 10 types of objects based on the CIFAR-10 dataset
Drag and drop an image here or click to select
Choose file No file chosen
SE185043 - NGUYEN THI MINH THU - OIT_FA25
27°C
Nhiều mây
Search
ENG
8:47 PM
12/4/2025

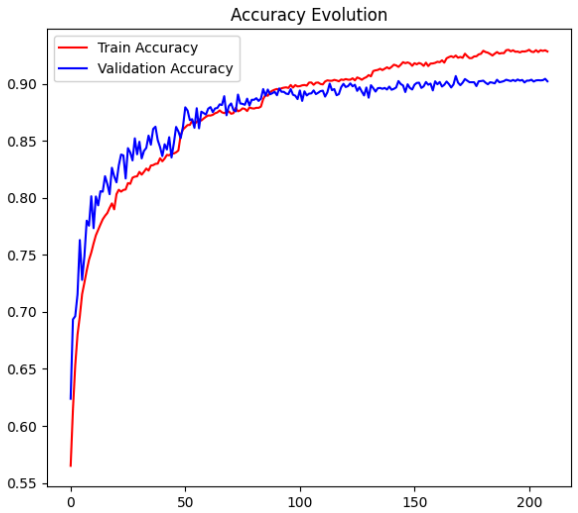

CIFAR-10 AI Image Classifier
127.0.0.1:5000
CIFAR-10 Image Classifier
Recognize 10 types of objects based on the CIFAR-10 dataset
Drag and drop an image here or click to select
Choose file frog.webp
PREDICTION NOW
SE185043 - NGUYEN THI MINH THU - OIT_FA25
27°C
Nhiều mây
Search
ENG
8:47 PM
12/4/2025





4. KẾT QUẢ VÀ ĐÁNH GIÁ

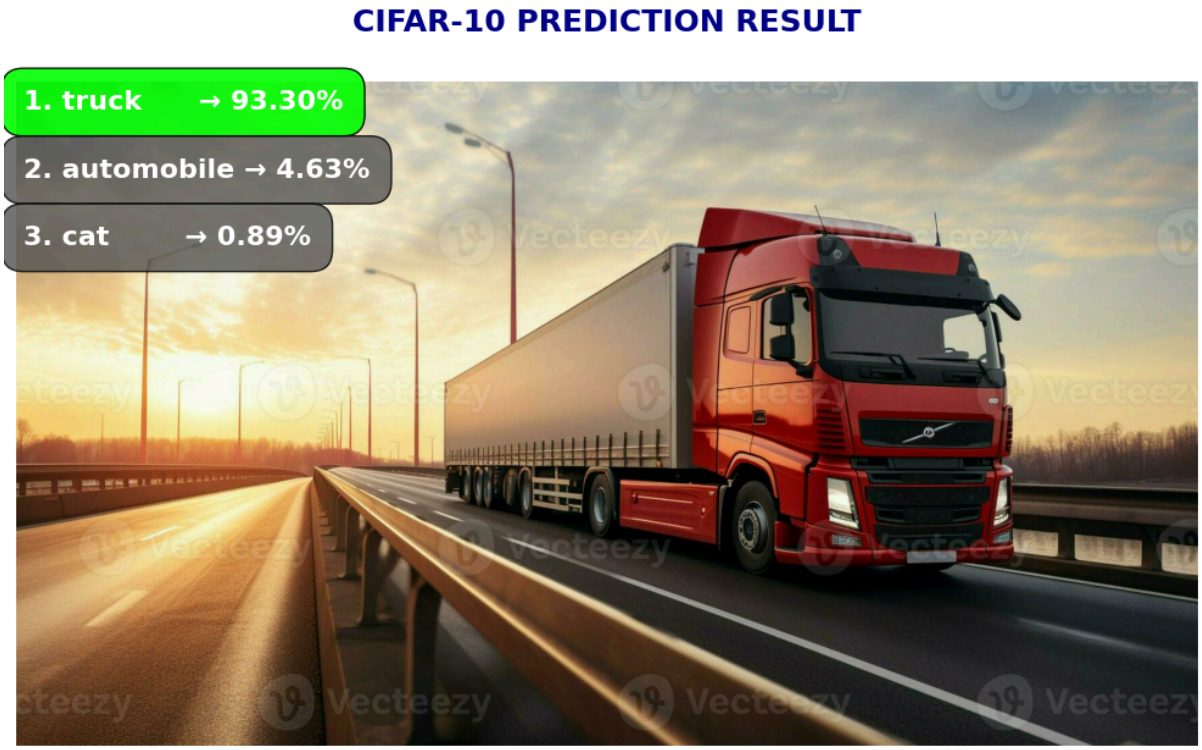




Chỉ số	Giá trị
Độ chính xác tập test	**90.70%**
Độ chính xác validation tốt nhất	92.18%
Loss tập test	0.4241
Số epoch thực hiện	209 (tự dừng sớm)

Biểu đồ học (learning curves) cho thấy không có hiện tượng overfitting đáng kể. Mô hình duy trì khoảng cách nhỏ giữa train/val accuracy trong toàn bộ quá trình huấn luyện.

Dự đoán trên ảnh thực tế



Kết quả chứng minh khả năng tổng quát hóa tốt của mô hình.

5. KẾT LUẬN

Dự án đã hoàn thành tốt các mục tiêu đề ra:

- ☑ Đạt độ chính xác cao thuộc top đầu với kiến trúc tự xây dựng
- ☑ Xây dựng quy trình huấn luyện bền vững, có khả năng chịu lỗi
- ☑ Phát triển ứng dụng web hoàn chỉnh, sẵn sàng demo và triển khai thực tế
- ☑ Đáp ứng đầy đủ yêu cầu của chương trình OJT từ khâu nghiên cứu đến sản phẩm

Công trình này không chỉ là một bài tập học thuật mà là một sản phẩm hoàn chỉnh có thể áp dụng ngay trong các hệ thống giám sát giao thông, an ninh hoặc nhận diện đối tượng tự động.

6. HƯỚNG PHÁT TRIỂN

1. Áp dụng các kiến trúc hiện đại (EfficientNet-B0, ResNet-50) để đạt hiệu suất cao hơn
2. Mở rộng bộ dữ liệu với các lớp đặc thù Việt Nam (xe máy, người đi đường)
3. Triển khai real-time detection từ webcam
4. Chuyển đổi sang TensorFlow Lite để tích hợp trên thiết bị di động
5. Đóng gói Docker và triển khai trên nền tảng cloud (AWS/GCP)

TÀI LIỆU THAM KHẢO

1. Krizhevsky, A. (2009). Learning Multiple Layers of Features from Tiny Images. Technical Report, University of Toronto.

2. Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR 2015.
3. Chollet, F. (2017). Deep Learning with Python. Manning Publications.