**📚 LLaMA-Factory Flask Backend API Documentation**

**🎯 Tổng quan**

Dự án LLaMA-Factory cung cấp một hệ thống backend Flask với **6 nhóm API chính**, phục vụ cho việc quản lý và sử dụng các mô hình ngôn ngữ lớn. Toàn bộ hệ thống API được thiết kế theo chuẩn RESTful, sử dụng các phương thức HTTP phù hợp và trả về dữ liệu dưới dạng JSON với cấu trúc thống nhất.

**🏥 1. API Kiểm tra Sức khỏe Hệ thống**

**GET /health**

Kiểm tra trạng thái hoạt động của hệ thống.

**Input:** Không có

**Output:**

{

"status": "ok",

"version": "1.0.0"

}

**🔐 2. Hệ thống Xác thực Người dùng**

**2.1 Đăng ký tài khoản**

**Endpoint:** POST /api/auth/register

**Input:**

{

"username": "string (3-50 ký tự)",

"email": "string (email hợp lệ)",

"password": "string (tối thiểu 8 ký tự)",

"confirm\_password": "string (giống password)",

"first\_name": "string (không bắt buộc)",

"last\_name": "string (không bắt buộc)"

}

**Output:**

{

"success": true,

"message": "string",

"user": {

"id": "string",

"username": "string",

"email": "string",

"first\_name": "string",

"last\_name": "string"

}

}

**2.2 Đăng nhập**

**Endpoint:** POST /api/auth/login

**Input:**

{

"username\_or\_email": "string",

"password": "string",

"remember\_me": "boolean (không bắt buộc)"

}

**Output:**

{

"success": true,

"message": "string",

"data": {

"token": "string",

"user": {

"id": "string",

"username": "string",

"email": "string"

}

}

}

**2.3 Đổi mật khẩu**

**Endpoint:** POST /api/auth/change-password

**Input:**

{

"current\_password": "string",

"new\_password": "string (tối thiểu 8 ký tự)",

"confirm\_password": "string (giống new\_password)"

}

**Output:**

{

"success": true,

"message": "string"

}

**2.4 Quên mật khẩu**

**Endpoint:** POST /api/auth/forgot-password

**Input:**

{

"email": "string (email hợp lệ)"

}

**Output:**

{

"success": true,

"message": "string"

}

**2.5 Đặt lại mật khẩu**

**Endpoint:** POST /api/auth/reset-password

**Input:**

{

"token": "string",

"new\_password": "string (tối thiểu 8 ký tự)",

"confirm\_password": "string (giống new\_password)"

}

**Output:**

{

"success": true,

"message": "string"

}

**2.6 Lấy thông tin người dùng hiện tại**

**Endpoint:** GET /api/auth/me

**Input:** Không có (yêu cầu token xác thực trong header)

**Output:**

{

"success": true,

"user": {

"id": "string",

"username": "string",

"email": "string",

"first\_name": "string",

"last\_name": "string"

}

}

**🔧 3. Hệ thống Fine-tuning Mô hình**

**3.1 Bắt đầu Fine-tuning**

**Endpoint:** POST /api/finetune

Bắt đầu một tác vụ fine-tuning mới.

**Tham số bắt buộc:**

{

"model\_name\_or\_path": "string", // Đường dẫn hoặc tên của mô hình cơ sở

"dataset": "string", // Tên dataset sử dụng để fine-tune

"template": "string", // Template sử dụng cho dữ liệu

"output\_dir": "string" // Thư mục đầu ra

}

**Tham số tùy chọn (với giá trị mặc định):**

{

"finetuning\_type": "lora", // Loại fine-tuning

"lora\_target": "all", // Target cho LoRA

"per\_device\_train\_batch\_size": 2, // Kích thước batch cho mỗi thiết bị

"gradient\_accumulation\_steps": 4, // Số bước tích lũy gradient

"lr\_scheduler\_type": "cosine", // Loại lịch trình learning rate

"logging\_steps": 5, // Số bước giữa các lần ghi log

"warmup\_ratio": 0.1, // Tỷ lệ warmup

"save\_steps": 1000, // Số bước giữa các lần lưu checkpoint

"learning\_rate": 5e-5, // Tốc độ học

"num\_train\_epochs": 3.0, // Số epoch huấn luyện

"max\_samples": 500, // Số lượng mẫu tối đa

"max\_grad\_norm": 1.0, // Giá trị chuẩn hóa gradient tối đa

"loraplus\_lr\_ratio": 16.0, // Tỷ lệ learning rate cho LoRA+

"fp16": true, // Sử dụng độ chính xác nửa

"report\_to": "none" // Báo cáo kết quả đến đâu

}

**Response:**

**✅ Thành công (200):**

{

"success": true,

"message": "<thông báo thành công>",

"task\_id": "<id của tác vụ>",

"process\_id": "<id của process>"

}

**❌ Thất bại (400):**

{

"success": false,

"message": "<thông báo lỗi>"

}

**⚠️ Lỗi xác thực (400):**

{

"success": false,

"message": "Validation error",

"errors": "<chi tiết lỗi xác thực>"

}

**3.2 Kiểm tra trạng thái Fine-tuning**

**Endpoint:** GET /api/finetune/status/<process\_id>

Lấy trạng thái của một tác vụ fine-tuning.

**Request Parameters:**

* process\_id: ID của process (trong URL)

**Response:**

**✅ Thành công (200):**

{

"success": true,

"message": "<thông báo>",

"task": [

{

"id": "<id của tác vụ>",

"process\_id": "<id của process>",

"status": "<trạng thái>", // 'pending', 'running', 'completed', 'failed'

"model\_name\_or\_path": "<đường dẫn hoặc tên mô hình>",

"dataset": "<tên dataset>",

"template": "<template>",

"output\_dir": "<thư mục đầu ra>",

"error\_message": "<thông báo lỗi nếu có>",

"created\_at": "<thời gian tạo>",

"updated\_at": "<thời gian cập nhật>",

"completed\_at": "<thời gian hoàn thành nếu có>"

}

]

}

**❌ Thất bại (404):**

{

"success": false,

"message": "<thông báo lỗi>"

}

**3.3 Lấy tất cả tác vụ Fine-tuning**

**Endpoint:** GET /api/finetune/tasks

Lấy danh sách tất cả các tác vụ fine-tuning.

**Request:** Không có tham số

**Response:**

**✅ Thành công (200):**

{

"success": true,

"tasks": [

{

"id": "<id của tác vụ>",

"process\_id": "<id của process>",

"status": "<trạng thái>",

"model\_name\_or\_path": "<đường dẫn hoặc tên mô hình>",

"dataset": "<tên dataset>",

"template": "<template>",

"output\_dir": "<thư mục đầu ra>",

"created\_at": "<thời gian tạo>",

"updated\_at": "<thời gian cập nhật>",

"completed\_at": "<thời gian hoàn thành nếu có>"

}

]

}

**3.4 Lấy tất cả mô hình đã Fine-tuned**

**Endpoint:** GET /api/finetune/models

Lấy danh sách tất cả các mô hình đã được fine-tuned.

**Request:** Không có tham số

**Response:**

**✅ Thành công (200):**

{

"success": true,

"models": [

{

"output\_dir": "<thư mục đầu ra>",

"model\_name": "<tên mô hình>",

"adapter\_type": "<loại adapter>",

"base\_model": "<mô hình cơ sở>"

}

]

}

**🎛️ 4. Quản lý Mô hình**

**4.1 Tải mô hình**

**Endpoint:** POST /api/models/load

Tải một mô hình từ tác vụ fine-tuning đã hoàn thành.

**Input:**

{

"task\_id": "string" // ID của tác vụ fine-tuning đã hoàn thành

}

**Output:**

**✅ Thành công:**

{

"success": true,

"message": "Model registered successfully: {model\_key}"

}

**❌ Lỗi:**

{

"success": false,

"message": "Error message" // ví dụ: "Fine-tuning task not found" hoặc "Task is not completed"

}

**Quy trình:**

* Kiểm tra xem tác vụ fine-tuning có tồn tại và đã hoàn thành không
* Tải mô hình từ thư mục đầu ra của tác vụ
* Thêm mô hình vào bảng model\_config trong cơ sở dữ liệu

**4.2 Gỡ bỏ mô hình**

**Endpoint:** POST /api/models/unload

Gỡ bỏ một mô hình theo ID.

**Input:**

{

"model\_id": "string" // ID của cấu hình mô hình

}

**Output:**

**✅ Thành công:**

{

"success": true,

"message": "Model unloaded successfully: {model\_key}"

}

**❌ Lỗi:**

{

"success": false,

"message": "Error message" // ví dụ: "Model not found" hoặc "Cannot unload active model"

}

**Quy trình:**

* Kiểm tra xem mô hình có tồn tại trong cơ sở dữ liệu không
* Nếu mô hình đang hoạt động, trả về lỗi yêu cầu hủy kích hoạt trước
* Nếu không, gỡ bỏ mô hình khỏi bộ nhớ và xóa khỏi cơ sở dữ liệu

**4.3 Cập nhật trạng thái hoạt động của mô hình**

**Endpoint:** POST /api/models/update-active

Kích hoạt hoặc hủy kích hoạt một mô hình.

**Input:**

{

"model\_id": "string", // ID của cấu hình mô hình

"is\_active": boolean // true để kích hoạt, false để hủy kích hoạt

}

**Output:**

**✅ Thành công:**

{

"success": true,

"message": "Model activated/deactivated successfully: {model\_key}"

}

**❌ Lỗi:**

{

"success": false,

"message": "Error message" // ví dụ: "Model not found" hoặc "Error activating model"

}

**Quy trình:**

**Nếu kích hoạt:**

* Hủy kích hoạt bất kỳ mô hình nào đang hoạt động
* Tải mô hình vào bộ nhớ nếu chưa được tải
* Cập nhật cơ sở dữ liệu để đánh dấu mô hình là hoạt động

**Nếu hủy kích hoạt:**

* Gỡ bỏ mô hình khỏi bộ nhớ
* Cập nhật cơ sở dữ liệu để đánh dấu mô hình là không hoạt động

**💬 5. API Chat**

**Chat với mô hình**

**Endpoint:** POST /api/chat

**Input:**

{

"messages": [

{

"role": "user",

"content": "string"

}

]

}

**Output:**

{

"success": true,

"message": "string",

"response": {

"role": "assistant",

"content": "string"

}

}

**📝 6. Quản lý Cuộc hội thoại**

**6.1 Tạo cuộc hội thoại**

**Endpoint:** POST /api/conversations

**Input:**

{

"title": "string (mặc định: New Conversation)",

"user\_id": "string (không bắt buộc)"

}

**Output:**

{

"success": true,

"message": "string",

"conversation\_id": "string"

}

**6.2 Xem thông tin cuộc hội thoại cụ thể**

**Endpoint:** GET /api/conversations/<conversation\_id>

**Input:** conversation\_id trong URL

**Output:**

{

"success": true,

"message": "string",

"conversation": {

"id": "string",

"title": "string",

"user\_id": "string",

"created\_at": "string (datetime)",

"updated\_at": "string (datetime)"

}

}

**6.3 Xem danh sách tất cả cuộc hội thoại**

**Endpoint:** GET /api/conversations

**Input:** user\_id (query parameter, không bắt buộc)

**Output:**

{

"success": true,

"conversations": [

{

"id": "string",

"title": "string",

"user\_id": "string",

"created\_at": "string (datetime)",

"updated\_at": "string (datetime)"

}

]

}

**6.4 Thêm tin nhắn vào cuộc hội thoại**

**Endpoint:** POST /api/conversations/<conversation\_id>/messages

**Input:**

{

"role": "string (user, assistant, hoặc system)",

"content": "string"

}

**Output:**

{

"success": true,

"message": "string"

}

**6.5 Xem lịch sử tin nhắn trong cuộc hội thoại**

**Endpoint:** GET /api/conversations/<conversation\_id>/messages

**Input:** conversation\_id trong URL

**Output:**

{

"success": true,

"message": "string",

"messages": [

{

"id": "string",

"conversation\_id": "string",

"role": "string",

"content": "string",

"created\_at": "string (datetime)"

}

]

}

**6.6 Xóa tất cả tin nhắn trong cuộc hội thoại**

**Endpoint:** POST /api/conversations/<conversation\_id>/clear

**Input:** conversation\_id trong URL

**Output:**

{

"success": true,

"message": "string"

}

**6.7 Xóa hoàn toàn cuộc hội thoại**

**Endpoint:** DELETE /api/conversations/<conversation\_id>

**Input:** conversation\_id trong URL

**Output:**

{

"success": true,

"message": "string"

}

**📊 Bảng tóm tắt API**

| **Nhóm API** | **Endpoint** | **Phương thức** | **Yêu cầu xác thực** |
| --- | --- | --- | --- |
| **🏥 Sức khỏe** | /health | GET | ❌ Không |
| **🔐 Xác thực** | /api/auth/register | POST | ❌ Không |
| **🔐 Xác thực** | /api/auth/login | POST | ❌ Không |
| **🔐 Xác thực** | /api/auth/change-password | POST | ✅ Có |
| **🔐 Xác thực** | /api/auth/forgot-password | POST | ❌ Không |
| **🔐 Xác thực** | /api/auth/reset-password | POST | ❌ Không |
| **🔐 Xác thực** | /api/auth/me | GET | ✅ Có |
| **🔧 Fine-tuning** | /api/finetune | POST | ❌ Không |
| **🔧 Fine-tuning** | /api/finetune/status/<process\_id> | GET | ❌ Không |
| **🔧 Fine-tuning** | /api/finetune/tasks | GET | ❌ Không |
| **🔧 Fine-tuning** | /api/finetune/models | GET | ❌ Không |
| **🎛️ Quản lý mô hình** | /api/models/load | POST | ❌ Không |
| **🎛️ Quản lý mô hình** | /api/models/unload | POST | ❌ Không |
| **🎛️ Quản lý mô hình** | /api/models/update-active | POST | ❌ Không |
| **💬 Chat** | /api/chat | POST | ❌ Không |
| **📝 Cuộc hội thoại** | /api/conversations | POST | ❌ Không |
| **📝 Cuộc hội thoại** | /api/conversations/<id> | GET | ❌ Không |
| **📝 Cuộc hội thoại** | /api/conversations | GET | ❌ Không |
| **📝 Cuộc hội thoại** | /api/conversations/<id>/messages | POST | ❌ Không |
| **📝 Cuộc hội thoại** | /api/conversations/<id>/messages | GET | ❌ Không |
| **📝 Cuộc hội thoại** | /api/conversations/<id>/clear | POST | ❌ Không |
| **📝 Cuộc hội thoại** | /api/conversations/<id> | DELETE | ❌ Không |

**✨ Đặc điểm chính của hệ thống API**

**🏗️ Kiến trúc**

1. **Cấu trúc JSON thống nhất**: Tất cả response đều có trường success để xác định trạng thái
2. **RESTful Design**: Tuân thủ các nguyên tắc thiết kế RESTful API
3. **HTTP Status Codes**: Sử dụng đúng mã trạng thái HTTP

**🔒 Bảo mật**

1. **JWT Authentication**: Sử dụng JWT token cho các endpoint yêu cầu xác thực
2. **Password Security**: Yêu cầu mật khẩu tối thiểu 8 ký tự
3. **Email Verification**: Hỗ trợ xác thực email và reset password

**🤖 AI/ML Features**

1. **Fine-tuning linh hoạt**: Hỗ trợ nhiều tham số cấu hình cho việc tinh chỉnh mô hình
2. **Model Management**: Cho phép tải, kích hoạt và quản lý nhiều mô hình
3. **LoRA Support**: Hỗ trợ fine-tuning với LoRA (Low-Rank Adaptation)

**💬 Chat System**

1. **Conversation Management**: Hệ thống quản lý cuộc hội thoại hoàn chỉnh
2. **Message History**: Lưu trữ và truy xuất lịch sử tin nhắn
3. **Multi-role Support**: Hỗ trợ các vai trò user, assistant, và system

**📈 Monitoring & Management**

1. **Health Check**: API kiểm tra sức khỏe hệ thống
2. **Task Tracking**: Theo dõi trạng thái các tác vụ fine-tuning
3. **Error Handling**: Xử lý lỗi chi tiết và thông báo rõ ràng