MLOps Project Specification for Product Classification Application

Vitalij Merenics, Fares Naem, Sebastián Mantilla-Serrano

1. Context and Objectives

1.1 Project Context

The primary goal of this project is to create an application that enables users to interact with a machine-learning model designed to classify products based on images and text descriptions. This system addresses the need for efficient categorization of products, an essential process for various industries, such as e-commerce, inventory management, and digital marketplaces. This classification allows businesses to organize products, making it easier for customers to find items and improving overall operational efficiency.

The application will be built using **FastAPI** and will expose a REST API that allows different types of users—both administrators and general users—to interact with the machine learning model. The application will be flexible, offering prediction functionalities, allowing new data to be ingested, training the model on the newly ingested data, and evaluating the model's performance over time.

1.2 Problem Statement

The main problem the application aims to address is the efficient classification of products into categories based on both textual descriptions and images. Manual classification can be time-consuming, error-prone, and resource-intensive. Our solution will automate this process through a machine learning model that can process both structured text and unstructured image data. This will not only save time but also increase the accuracy and consistency of product categorization.

The specific challenges include:

- The need to accommodate different types of users (e.g., admin vs. general user) with different permission levels.
- The ingestion of new data (images and text) that can evolve the model over time.
- Continuous monitoring and retraining of the model as new products are added.

1.3 Stakeholders

- **Sponsor:** The project sponsor is a company operating in the e-commerce domain, looking to automate product classification.
- **Users:**
    - **General Users:** These are everyday users or employees of the e-commerce company who need to classify new products or query the system to classify a product based on its image and text description.
    - **Administrators:** Admins will have more control over the system. They can upload new training data (images and text), trigger model retraining, and monitor the model's performance.

1.4 Application Usage Context

The application will be deployed locally or on a **cloud server** (AWS) and exposed as a **REST API**. The application is not constrained to a graphical user interface (GUI), but can later integrate with other platforms such as a web frontend, a mobile app, or other backend systems via the API. The system will provide role-based authentication and will be integrated with the organization's existing infrastructure for user management.

1.5 Medium of Interaction

The main interaction with the system will be through a **REST API**, allowing different applications (either graphical interfaces or other backends) to make use of the classification service. The general users will interact with API endpoints to classify products, while admins will have access to additional endpoints to retrain the model and manage data.

2. Model Overview

2.1 Model Type and Functionality

The model utilized for this project is a **multimodal machine learning model** capable of processing both images and text descriptions. The model is designed to predict the category of a product using:

- **Image data**: A convolutional neural network (CNN) pre-trained on a large dataset, then fine-tuned for the specific product categorization task.
- **Text data**: A transformer-based language model that processes textual descriptions and extracts relevant features for categorization.

These two modalities (image and text) are combined in the final layers of the model to make the product category predictions.

2.2 Evaluation Metrics

The performance of the model is measured using the following metrics:

- **Accuracy**: Percentage of correct classifications out of the total classifications.
- **Precision and Recall**: To measure the model's ability to correctly identify categories and handle imbalanced datasets.
- **F1-Score**: The harmonic mean of precision and recall, useful for understanding performance across different product categories. This is the principal metric used to evaluate the performance of the model.
- **Training Time**: The time it takes to retrain the model with new data.
- **Prediction Time**: The speed at which the model can classify a product during an API call.

2.3 Model Constraints

Given the nature of the application, certain constraints need to be considered:

- **Robustness**: The model should handle both high-quality and low-quality images.

- **Prediction Time**: For real-time predictions, especially in customer-facing scenarios, the model should return results within a set time limit (e.g., less than 1 second per prediction).

3. Database

Although optional, this project will likely use a **relational database** (e.g., PostgreSQL) to store:

- **Product Images**: Metadata such as image URL paths.
- **Text Descriptions**: Associated product descriptions.
- **Product Categories**: Both predicted categories and ground-truth labels for training data.
- **User Information**: Storing role-based authentication details (e.g., admin or general user).

To handle image data, we may also use an object storage service (e.g., AWS EC2) to store and access large image datasets.

The database schema will evolve as new data is ingested and new products are classified. The system will need to accommodate new data and efficiently query for predictions.

4. API Design

The API serves as the core interface between the model, the database, and the users. Below is an outline of the API structure, with different endpoints categorized by their functionality.

4.1 Authentication

- **POST /auth/login**: Users (both admin and general) log in with credentials to get an authentication token.
- **POST /auth/signup**: New users can create an account with basic information (admin functionality).

4.2 Prediction

- **POST /predict**: Takes an image and a text description as input and returns the predicted category. This will call the machine learning model.
    - **Input**: Image file and product description.
    - **Output**: Predicted category.

4.3 Data Management (Admin Only)

- **POST /admin/add-product-data**: Allows an admin to upload new product data (images, text descriptions, and categories) to the database for future training.
    - **Input**: Image files, text descriptions, and product categories.
    - **Output**: Status of data upload.
- **POST /admin/retrain-model**: Initiates a retraining process for the model using the updated data in the database.

    - **Input**: (Optional) Configuration for model training parameters (e.g., epochs, batch size).
    - **Output**: Status of the training process.
- **GET /admin/evaluate-model**: Returns the current performance metrics of the model on a validation/test dataset.
    - **Output**: Model accuracy, precision, recall, F1-Score, etc.

4.4 Monitoring and Logs (Admin Only)

- **GET /admin/logs**: Retrieves logs of the API's operation, including data ingestion, model retraining, and errors.

5. Testing and Monitoring

Testing and monitoring are critical components to ensure the system's robustness and that it meets the expected performance.

5.1 Unit Testing

Each individual component of the application should be tested:

- **Model Training**: Test that the model can be retrained with new data and produces reasonable predictions.
- **Prediction**: Test that the model returns correct predictions when given valid image and text inputs.
- **API Endpoints**: Ensure all endpoints return correct responses, both for valid and invalid inputs.
- **Data Ingestion**: Test that new product data (images and descriptions) can be uploaded and properly saved to the database.

5.2 Monitoring

Ongoing monitoring is required to ensure that the model continues to perform well:

- **Model Performance Monitoring**: After deployment, the model's performance will be continuously tracked. Metrics like accuracy and F1-score will be periodically evaluated on both the existing dataset and new incoming data.
- **Retraining Criteria**: The system will trigger retraining when performance drops below a certain threshold, or periodically (e.g., monthly), depending on new product data availability.
- **Alerting System**: When the model's performance degrades, an email alert or notification will be sent to the admins, prompting them to retrain the model or investigate potential issues.

6. Flow Process Overview and Implementation Scheme

The flow can be categorized into **User Interactions** and **Admin Interactions**, with each role having specific endpoints and functionalities. The model lifecycle, data management, and API interactions are the core components of this flow.

6.1. User Interaction (General User)

- **Login/Authentication**:
    - User accesses the application and logs in using credentials via POST /auth/login.
    - User receives an authentication token to access the prediction endpoints.
- **Product Classification**:
    - User uploads an image and text description of a product via POST /predict.
    - The system sends this data to the machine learning model.
    - The model predicts the product category by processing both the image (CNN) and text (Transformer) data.
    - The predicted product category is returned to the user.

6.2. Admin Interaction

- **Login/Authentication**:
    - Admin logs in using credentials via POST /auth/login and receives an authentication token.
- **Data Ingestion**:
    - Admin uploads new product data (images, text, and categories) via POST /admin/add-product-data.
    - Data is saved in the database (PostgreSQL for metadata and object storage for images).
- **Model Retraining**:
    - Admin triggers retraining of the model using the new data via POST /admin/retrain-model.
    - The system retrains the model, updates the weights, and the new model is deployed.
    - Admin receives a status update of the training process.
- **Model Evaluation**:
    - Admin can request current model performance metrics via GET /admin/evaluate-model (Accuracy, Precision, F1-Score, etc.).
- **Monitoring Logs**:
    - Admin can retrieve operational logs to monitor data ingestion, retraining history, and potential errors via GET /admin/logs.

6.3. Model and Data Flow

- **Data Storage**:

    - Product images and text descriptions uploaded by users or admins are saved in the database.
    - The database also stores user information and product categories (predicted and ground truth for training).
- **Prediction Flow**:
    - The system fetches data from the database and feeds it to the machine learning model.
    - The model outputs a category based on the processed text and image data.
- **Retraining Process**:
    - When retraining is triggered by the admin, the system pulls new product data from the database.
    - The model is retrained, tested, and its performance metrics are logged and evaluated.

6.4. Monitoring and Testing

- **Unit Testing**: Each component is tested individually (prediction, model training, data ingestion, etc.).
- **Performance Monitoring**:
    - Model performance is tracked over time and alerts are sent when performance degrades (via email or notification).
- **Retraining Trigger**:
    - If the model's performance drops below a threshold or after a certain time, retraining can be triggered either automatically or manually.

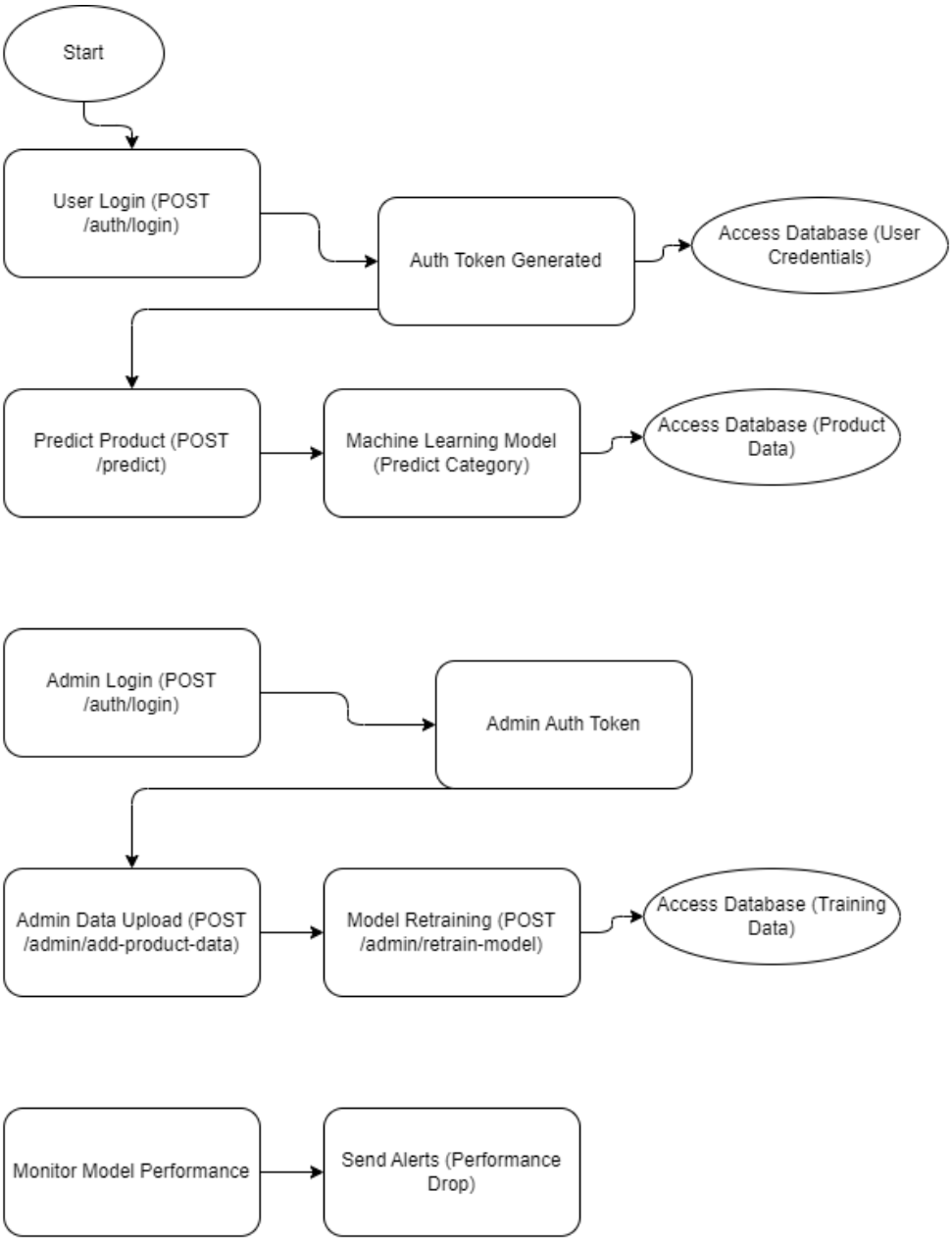

Figure 1: Product Classification Application scheme diagram

Conclusion

This specification outlines a structured approach to building an application for product classification system in the context of e-commerce using FastAPI and a multimodal machine learning model. The key components—model performance, database management, API structure, and testing/monitoring—are essential to ensuring that the system remains robust, and efficient in addressing the core problem of automating product categorization. With a clear specification, the development process can proceed in a structured manner, ensuring that the project remains on track and fulfills the customer expectations of a fast and precise automated classification system.