

Projet AIA4 – 2025

Réseaux de Neurones et Deep Learning II

Application mobile de recherche basée sur l'Image et le Texte, connectée à une plateforme de vente en ligne

1. Contexte et Présentation du projet

Ce projet a pour objectif de concevoir une **application mobile intelligente** permettant aux utilisateurs d'effectuer des recherches de produits en utilisant **du texte** ou **des images**. L'application sera connectée à une plateforme de e-commerce et exploitera des modèles d'Intelligence Artificielle pour fournir les résultats les plus pertinents.

Le système repose sur :

- un **pipeline de collecte automatisée (scraping)**,
- une **architecture de Feature Extraction multimodale**,
- un **moteur de scoring** combinant vision et NLP,
- une **API backend** optimisée,
- et une **application mobile cross-platform** (Flutter).

2. Collecte automatisée des données (Web Scraping) – M1

Les données seront extraites automatiquement depuis la plateforme de vente.

Données à scraper

- id du produit
- nom
- description
- prix
- réduction
- URL image
- **catégorie / sous-catégorie** (ajout utile pour classification)
- **popularité / rating** (si disponible)

Gestion des images

- Toutes les images seront téléchargées dans un **répertoire structuré** :
 - /dataset/
 - /images/

- id_12345.jpg
- Vérification automatique :
 - Format (JPEG/PNG)
 - Taille minimale
 - Suppression des images corrompues

Aspects techniques

- Scraping automatique via BeautifulSoup, Selenium, ou Playwright
- Tâches planifiées (CRON ou Airflow)
- CSV + stockage dans une base NoSQL (MongoDB) ou **SQL (PostgreSQL)**

3. Feature Extraction (Textuelle & Visuelle) – M1

Objectif : convertir texte et images en **vecteurs numériques** pour permettre la recherche sémantique.

3.1 Extraction Textuelle

- Utilisation de **Universal Sentence Encoder (USE)**
- Embedding de la description et du nom du produit
- Format : vecteur 512 dimensions

3.2 Extraction Visuelle

- Utilisation du modèle **Xception pré-entraîné sur ImageNet**
- Extraction du vecteur du dernier layer global pooling
- Format : vecteur de 4096 dimensions

3.3 Normalisation et stockage

- Normalisation L2 sur les embeddings :
 - $x = \text{np.array}([3, 4])$
 - $x_norm = x / \text{np.linalg.norm}(x)$
- Stockage dans une base pour la similarité :
 - Utiliser l'API **FAISS** Pour le calcul de similarité

4. Moteur de Scoring Multimodal – M2

Le moteur compare l'input utilisateur à l'ensemble des produits.

4.1 Similarité textuelle

- Métrique : **cosine similarity**
- Score $\in [0,1]$

4.2 Similarité visuelle

- Cosine ou Euclidean distance
- Normalisation obligatoire

4.3 Fusion des scores

Un scoring final multimodal peut être défini :

$$\text{FinalScore} = \alpha * \text{TextScore} + \beta * \text{ImageScore}$$

Où α et β sont réglables selon l'importance donnée à chaque modalité. Ces valeurs doivent être guidées par la préférence de l'utilisateur.

4.4 Optimisations possibles

- PCA pour réduire les dimensions
- Index FAISS (IVF, HNSW) pour accélérer les recherches
- Threshold dynamique pour filtrer les résultats faibles

5. Architecture Backend – M2

- API REST développée en **Spring Boot** ou **FastAPI/FlaskAPI**
- Endpoints possible :
 - /search/text
 - /search/image
 - /search/multimodal
 - /product/{id}
 - /cart/add
 - /cart/list
- Chargement des modèles en mémoire (pickle / SavedModel)
- ORMs : Hibernate ou SQLAlchemy
- Gestion du cache (Redis)

6. Application Mobile – M3

Développée en **Flutter** (recommandé pour Android + iOS).

Fonctionnalités principales

Affichage Produit

- Par catégorie
- Nom et image du produit
- Prix et Réduction

Recherche

- Champ texte
- Upload ou capture d'image via caméra
- Affichage des résultats par pertinence

Panier

- Ajout rapide au panier
- Mise à jour des quantités
- Soumission de commande (Prévoir juste le bouton)

Design & UI

- Interface intuitive (Material Design)
- Filtre : prix, réduction, catégorie
- Historique de recherche

7. Sécurité & Protection des données

- API sécurisée par JWT
- Stockage des images sur serveur isolé ou S3
- Poisoned Data Checking (vérification des données scrappées)
- Filtrage anti-robots sur la plateforme cible

8. Déploiement

- Backend sur **Docker** + Kubernetes
- App mobile distribuée via Play Store
- CI/CD : GitHub Actions