Développement d’un moteur de recherche multimodal

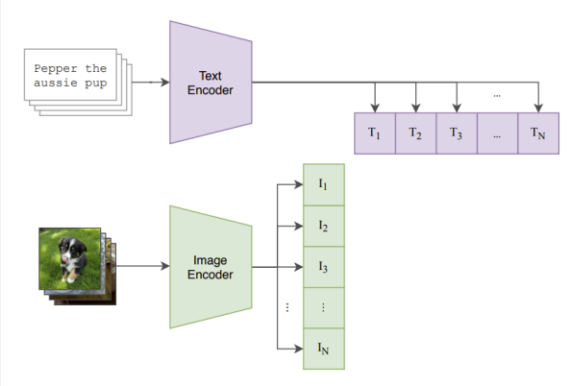
BILAL EL BIYADI ET YVAN LOUAMBA

CY TECH - ING 3 IA B

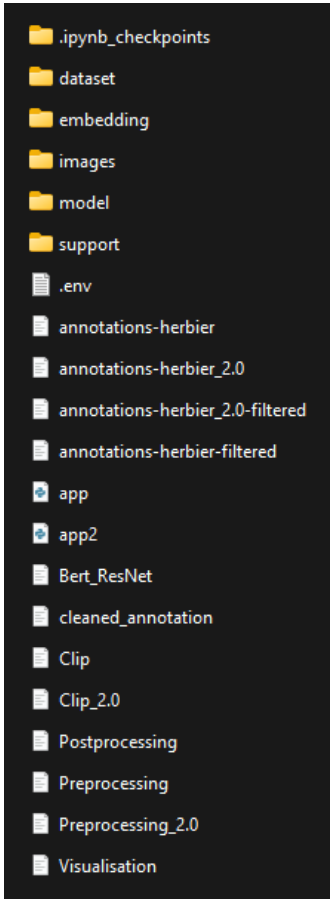
6 JANVIER 2025

Introduction et contexte

Ce projet vise à la mise au point d'un moteur de recherche multimodal visant à associer des images d'herbier à des descriptions textuelles. L'objectif est de permettre à un utilisateur de rechercher efficacement une plante par mots-clés ou descriptions en langage naturel, et de lui présenter les images les plus pertinentes.



Arborescence du projet



Exemple d’image du dataset

Figure 1: Exemple d’image de feuille issue du répertoire d’images d’herbier numérisé

Présentation des datasets

Dataset 1 : Structure simplifiée (300 échantillons)

```
code;epines;feuille_ext_acuminee
ANG 419,00;1.0;1.0
ANG 430,00;1.0;0.0
ANG 2056,00;1.0;0.0
ANG 5088,00;1.0;0.0
...
```

Il comporte :

• code : identifiant unique (ex. ANG 419,00).

• epines : variable binaire (0/1) indiquant la présence ou l'absence d'épines.

• feuille_ext_acuminee : variable binaire (0/1) précisant la forme de la feuille.

Ce dataset est relativement *basique* : chaque plante est décrite par uniquement deux attributs seulement.

Présentation des datasets

Dataset 2 : Structure riche avec des attributs additionnels

```
T;code;E;BF;TF
0.0;CLF 289236,0000;1.0;1.0;1.0
0.0;P03327766;1.0;1.0;1.0
0.0;P04681621;0.0;0.0;1.0
1.0;P03550384;0.0;0.0;1.0
...
```

On y trouve :

- T : Tige (Herbacée, ligneuse) encodée en 0/1.
- code : identifiant unique (P03327766, CLF 289236,0000, etc.).
- E : Épine (0/1 : absence ou présence).
- BF : Bordure des feuilles (0/1 : lisse ou dentée).
- TF : Taille des feuilles (ex. 0/1/2, Moins d’1 cm, Entre 1 cm et 10 cm, Plus de 10 cm).

Préparation du dataset multimodal

Étapes pour chaque dataset :

1. Lecture du fichier CSV et vérification des images (code associé).
2. Génération des descriptions textuelles via un prompt adapté.
3. Stockage final au format JSON ({image_filename, code, description}).

Modèles utilisés pour la génération :

Google Gemini 1.5 Flash : Bonne qualité mais lent.
OpenAI GPT-4 : Qualité similaire à Gemini, plus rapide, adopté pour la plupart des cas.

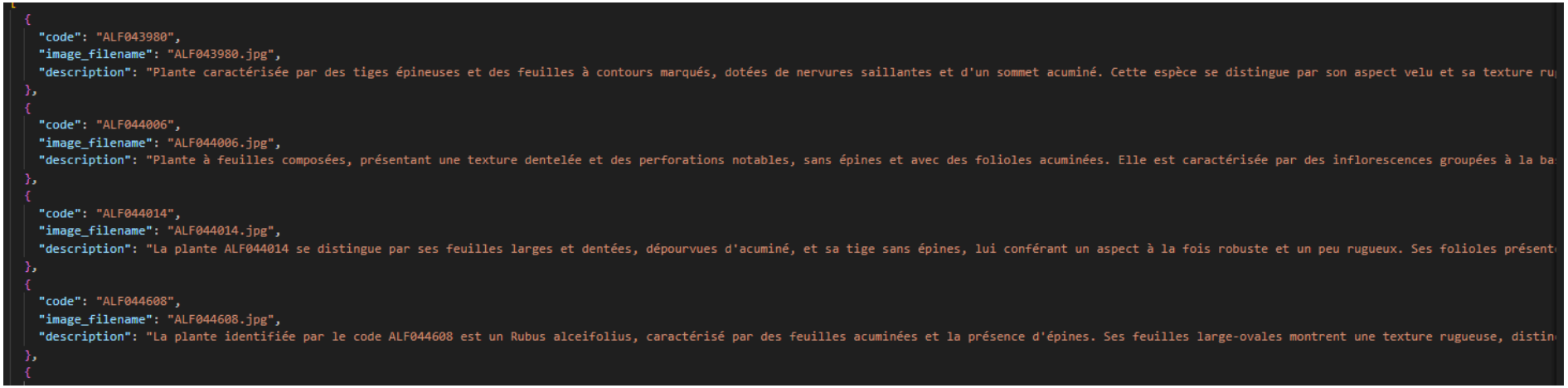
Prompt dataset1

```
prompt = (
    f"Tu es un botaniste expert.\n"
    f"À partir de l'image ci-dessous et des informations complémentaires, produis une description botanique "
    f"concise (1 ou 2 phrases maximum) qui ne s'applique qu'à cette plante.\n\n"

    f"Code de la plante : {code}\n"
    f"Caractéristique 'épines' : {epines}\n"
    f"Caractéristique 'feuille_ext_acuminee' : {feuille_acuminee}\n\n"

    f"- N'intègre que les détails dérivés de l'image et des caractéristiques listées.\n"
    f"- Utilise un vocabulaire accessible tout en restant suffisamment précis.\n"
    f"- La description doit permettre de distinguer clairement l'image associée de toute autre image.\n\n"
    f"Décris uniquement la plante correspondant à ce code et ces attributs, sans mentionner d'autres plantes.\n"
    f"---\n"
    f"Voici l'image : "
)
```

Prompt dataset2

```
# Prompt adapté pour la nouvelle structure:
prompt = (
    f"Tu es un botaniste expert.\n"
    f"À partir de l'image ci-dessous et des informations complémentaires, produis une description botanique "
    f"concise (1 phrase et demi maximum) qui ne s'applique qu'à cette plante.\n\n"
    f"Code de la plante : {code}\n"
    f"Caractéristique 'Tige' : {t}\n"
    f"Caractéristique 'Épine' : {e}\n"
    f"Caractéristique 'BF' : {bf}\n"
    f"Caractéristique 'TF' : {tf}\n\n"
    f"- N'intègre que les détails dérivés de l'image et des caractéristiques listées.\n"
    f"- Utilise un vocabulaire accessible tout en restant suffisamment précis.\n"
    f"- La description doit permettre de distinguer clairement l'image associée de toute autre image.\n\n"
    f"Décris uniquement la plante correspondant à ce code et ces attributs, sans mentionner d'autres plantes.\n"
    f"---\n"
    f"Voici l'image : "
)
```

Dataset_multimodal.json



Filtrage et nettoyage

Après génération des paires :

Suppression des images introuvables.

Élimination des descriptions vides.

Division des données :

- **Train** : 70%
- **Validation** : 15%
- **Test** : 15%

Environ **300 échantillons** par dataset final.

Analyse du modèle CLIP

Développé par OpenAI

Modèle multimodal associant texte et images.
Entraîné sur des données web massives (texte + images).

Principe clé :

Apprentissage contrastif : Maximiser la similarité entre des paires texte-image associées et minimiser celle des paires non associées.

Architecture :

Deux encodeurs distincts :

Texte : Basé sur un transformeur.
Image : Basé sur un réseau neuronal convolutif ou un Vision Transformer (ViT).

Fonctionnalités :

Génération d'un espace latent partagé pour le texte et les images.
Capable de recherches croisée (texte → image, image → texte).

Difficultés rencontrées

Problème 1 : NaN pendant l'entraînement

Cause : Overflow numérique (fp16).
Solution :

- Passage en float32 (model.float()).
- Utilisation accrue de mémoire GPU.

Problème 2 : Limitation du tokenizer

Maximum : 77 tokens.
Impact : Aucun problème pour les prompts concis (descriptions botaniques).

Hyperparamètres

num_epochs = 5
temperature = 0.1
lr = 1e-5
optimizer : Adam
batch_size = 4

Remarque :

Pour chaque minibatch, le but est d'associer correctement chaque image à une phrase unique.

Lorsqu'on calcule l'association entre une image et une phrase, il faut maximiser le lien entre l'image et sa phrase associée tout en minimisant les liens avec les autres phrases.

Résultats CLIP sans fine-tuning (Baseline)

Dataset 1 (simplifié)	Train Acc.	Val Acc.	Test Acc.
Baseline CLIP (prétr.)	29.52%	–	28.89%

Table 1: Performances initiales (dataset 1).

Dataset 2 (plus riche)	Train Acc.	Val Acc.	Test Acc.
Baseline CLIP (prétr.)	29.52%	28.89%	22.22%

Table 2: Performances initiales (dataset 2).

On note que le test accuracy du dataset 2 (22.22 %) est plus bas que celui du dataset 1 (28.89 %).

La plus grande richesse en features textuelles ne se retrouve pas exploitée sans fine-tuning.

Premier Fine-Tuning : Projection Heads

Dans ce premier fine-tuning, toutes les couches du modèle CLIP ont été gelées sauf :

visual.proj : C'est la couche de projection finale des caractéristiques visuelles extraites par la partie vision du modèle

CLIP.text_projection : C'est la couche de projection finale des caractéristiques textuelles extraites par la partie texte du modèle CLIP.

Résultats Fine-tuning 1 de CLIP

Approche FT1 (dataset 1)	Train Acc.	Val Acc.	Test Acc.
Finetuning partiel (1er essai)	25.24%	26.67%	26.67%
Finetuning partiel (2e essai)	72.86%	46.67%	48.89%

Table 3: Approche de fine-tuning partiel pour dataset 1.

On note qu'avec le 2ème essai la test accuracy a augmenté de 20% (28,89 → 48,89).

Approche FT1 (dataset 2)	Train Acc.	Val Acc.	Test Acc.
Finetuning partiel	62.38%	37.78%	31.11%

On gagne environ 9 points (22.22% → 31.11%) sur le test.

Deuxième Fine-Tuning : Déblocage de Plus de Couches

Dans ce second fine-tuning, toutes les couches ont été gelées sauf :

visual.proj et **text_projection** : Couche de projection finale, comme dans le premier fine-tuning.

visual.transformer.resblocks.11. **et transformer.resblocks.11.** : Derniers blocs de résidus des encodeurs visuel et textuel.

visual.ln_post et **ln_final** : Couches de normalisation finales pour améliorer la qualité des sorties des encodeurs.

Résultats Fine-tuning 2 de CLIP

Approche FT2 (dataset 1)	Train Acc.	Val Acc.	Test Acc.
Finetuning + DataAug Profond	72.86%	46.67%	48.89%

Table 5: Approche de fine-tuning plus profond pour dataset 1.

Les performances sur le dataset 1 montrent que l'approche de finetuning avec augmentation des données suggère une amélioration des performances.

Approche FT2 (dataset 2)	Train Acc.	Val Acc.	Test Acc.
Finetuning + DataAug Profond	69.52%	44.44%	35.56%

Table 6: Approche de fine-tuning plus profond pour dataset 2.

Dans chaque cas, la *test accuracy* est encore améliorée, aboutissant à 48.89% pour *dataset 1* et 35.56% pour *dataset 2*. L'écart (13 points) reflète possiblement la complexité supérieure du second dataset.

Analyse du modèle BERT+ResNet

Fusion de deux architectures puissantes :

BERT (Bidirectional Encoder Representations from Transformers) :

- Encodeur de texte basé sur un transformeur.
- Connu pour sa compréhension fine des relations linguistiques.

ResNet (Residual Neural Network) :

- Réseau convolutionnel profond pour le traitement d'images.
- Permet d'extraire des caractéristiques visuelles riches.

Principe clé :

- Extraction séparée des caractéristiques texte et image via BERT et ResNet.
- Combinaison des embeddings texte et image dans un espace commun.

Fonctionnalités :

- Recherche croisée (texte → image, image → texte).
- Adaptable à des tâches spécifiques via fine-tuning.

Difficultés rencontrées

Gestion séparée des modèles pré-entraînés :

BERT : Tokenisation générant des tenseurs de dimension **768**.

ResNet : Encodage produisant des tenseurs de dimension **1000**.

Alignement artificiel nécessaire :

Utilisation de couches de projection (512) pour rendre texte et image comparables dans un espace commun.

Étape essentielle pour des embeddings compatibles.

Optimisation de l'entraînement :

Gestion du gel/dégel sélectif des couches pour :

Réduire le nombre de paramètres.

Stabiliser l'entraînement.

Résultats BERT+ ResNet sans fine-tuning (Baseline)

Modèle	Configuration	Train Acc.	Val Acc.	Test Acc.
BERT+ResNet	Baseline (No FT)	24.00%	33.33%	24.44%
	Fine-tuning (10 époques)	34.76%	42.22%	47.33%

Résultats Baseline (Sans Fine-tuning):

Utilisation des poids pré-entraînés (BERT : anglais général, ResNet : ImageNet).
Scores faibles, non adaptés à la botanique ou au dataset Herbier (300 échantillons).

Mise en place du Fine-tuning:

Approche : Fine-tuning partiel (10 époques, dernières couches dégelées).

Étapes clés :

- Projections des embeddings (768 et 1000 → 512, normalisation).
- Similarité calculée dans l'espace latent 512 avec une perte contrastive.
- Optimisation avec AdamW (learning rate initial : 5e-5, scheduler).

Ressources : GPU T4 (Google Colab), durée : 40 minutes.

Résultats Après Fine-tuning

Gain notable en accuracy test (de **24,44 %** à **47,33 %**).
Adaptation efficace des dernières couches aux données spécifiques de l'herbier.

Postprocessing : développement du moteur de recherche

Postprocessing après entraînement :

Sauvegarde des poids fine-tunés (.pth).
Génération et stockage des embeddings d'images (format JSON).
Interface utilisateur (Streamlit) pour choisir et tester les modèles.

Sauvegarde des poids :

1.Charge le modèle choisi (CLIP ou BERT+ResNet).
2.Transforme les images en embeddings via preprocessing.
3.Normalise les embeddings (L2) et les stocke en JSON

Postprocessing : développement du moteur de recherche

Fonctionnalités principales :

Champ de saisie pour une **query textuelle**.

Sélecteur pour choisir un modèle :

- CLIP (fine-tuné)
- BERT+ResNet (fine-tuné)

Résultats : Calcul de la **similarité cosinus** et affichage des images **top-k** les plus pertinentes.

Postprocessing : développement du moteur de recherche

Étapes de gestion :

L’utilisateur choisit un modèle spécifique.

Chargement des fichiers associés :

- Poids finetunés (**finetuned_xxx.pth**).
- Embeddings (**image_embeddings_xxx.json**).

L’interface est lancée avec ces paramètres, permettant de tester différentes configurations sans modifier le code.

Postprocessing : développement du moteur de recherche

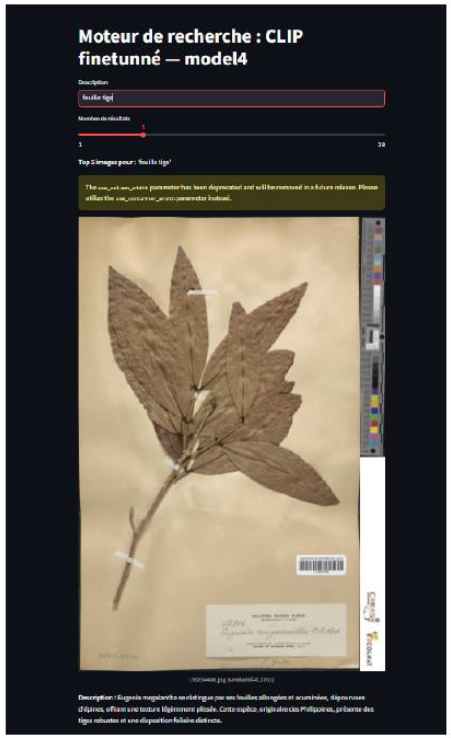


Figure 2: Exemple d'interface graphique (Streamlit) pour la recherche.

Postprocessing : développement du moteur de recherche

Observations sur les similarités texte-image :

CLIP (fine-tuné) : Similarité moyenne ≈ 0.25.

BERT+ResNet (fine-tuné) : Similarité moyenne ≈ 0.07.

Modèle	Sim. Moyenne Observée
CLIP (finetuned)	≈ 0.25
BERT+ResNet (finetuned)	≈ 0.07

Table 7: Scores de similarité moyens observés (embeddings).

Tableau final : récapitulatif complet des résultats

Modèle	Configuration	Train Acc.	Val Acc.	Test Acc.
CLIP - DS1	Baseline (prétr.)	29.52%	–	28.89%
	Fine-tune Partiel (1er)	25.24%	26.67%	26.67%
	Fine-tune Partiel (2e)	72.86%	46.67%	48.89%
CLIP - DS2	Baseline (prétr.)	29.52%	28.89%	22.22%
	Fine-tune Partiel	62.38%	37.78%	31.11%
	Fine-tune Profond + DataAug	69.52%	44.44%	35.56%
BERT+ResNet	Baseline (No FT)	24.00%	33.33%	24.44%
	Fine-tuning (10 époques)	34.76%	42.22%	47.33%

Table 8: Tableau récapitulatif final des résultats (tous modèles et configurations).

Conclusion et perspectives

Pertinence démontrée :

- Transfert de connaissances efficace pour la recherche image-texte botanique.
- Travail sur un volume limité (**300 échantillons**).

Réalisations clés :

- Deux datasets d’annotations (simplifié et riche).
- Génération de fichiers JSON via prompts adaptés.
- **CLIP** : Fine-tuning améliore la précision top-1 (+10 à +20 points).
- **BERT+ResNet** : Test accuracy doublée (~24% → ~47%).

Conclusion et perspectives

Observations :

- Compression des scores de similarité (0.07–0.27) pour CLIP et BERT/ResNet.
- Interface Streamlit fonctionnelle, démontrant la faisabilité du moteur multimodal.

Perspectives :

- Entraînement prolongé avec plus de données.
- Amélioration des performances globales.