Institut National Polytechnique
Félix Houphouët-Boigny

École Nationale Supérieure
de Statistique et d'Économie Appliquée

Logo du Projet
Doppelganger

Rapport de Projet

Système de Reconnaissance Faciale « Doppelganger »

Encadré par :
Monsieur MOHAMED ABBAS

Membres du Groupe :

COULIBALY Foungnigué Adèle
MABIALA Bergin
KONAN N'guessan Esaie
KOUAKOU Marcel Junior

Date : 16 Janvier 2026

Projet : Développement d'une application de reconnaissance faciale basée sur ArcFace

1. Introduction et Contexte

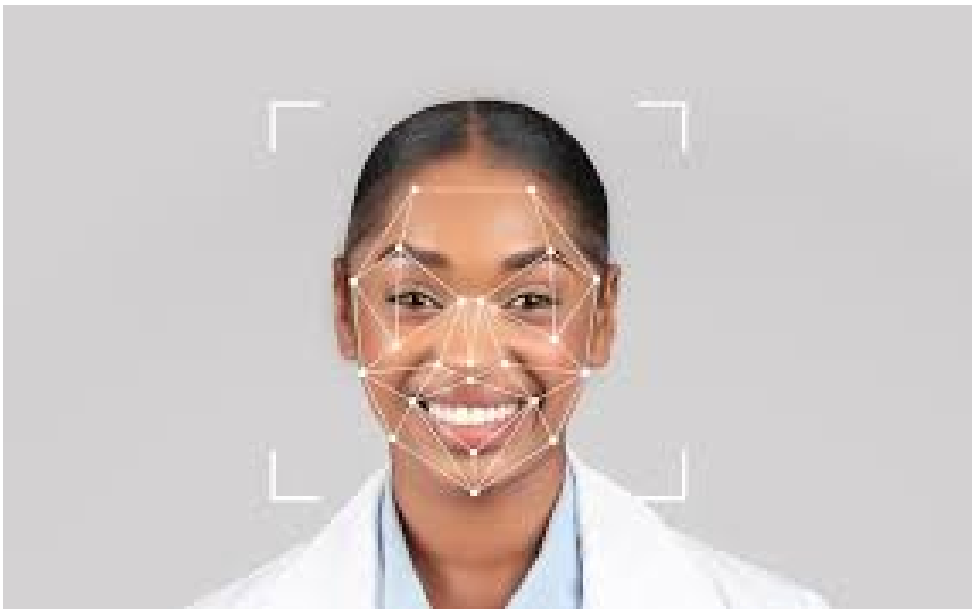

FIGURE 1 – Illustration du concept de reconnaissance faciale

Ce projet vise à développer un système de reconnaissance faciale permettant d'identifier, parmi une base de visages connus, la personne la plus ressemblante à une photo test fournie par l'utilisateur. L'application, baptisée **Doppelganger**, utilise le modèle **ArcFace** via la bibliothèque **InsightFace**, et est déployée sous forme d'interface web interactive grâce à **Streamlit**.

Objectif principal

Comparer un visage inconnu à une base de visages pré-enregistrés et retourner les meilleures correspondances avec un score de similarité.

2. Démarche et Méthodologie

2.1. Choix technologiques

FIGURE 2 – InsightFace - Bibliothèque pour la reconnaissance faciale

— **InsightFace (ArcFace)** : Pour l'extraction robuste des *embeddings* faciaux. Cette bibliothèque offre des modèles pré-entraînés de haute précision.

FIGURE 3 – Streamlit - Framework pour applications web interactives

— **Streamlit** : Pour créer une interface utilisateur simple, interactive et déployable. Permet un développement rapide d'applications data science.

FIGURE 4 – Scikit-learn - Bibliothèque pour le machine learning

— **Scikit-learn** : Pour le calcul de similarité cosinus entre les embeddings faciaux.

FIGURE 5 – OpenCV - Bibliothèque pour le traitement d'images

— **OpenCV & PIL** : Pour le prétraitement des images avant l'extraction des features.

2.2. Architecture du système

1. **Phase d'enregistrement** :
   - Les images des personnes connues sont stockées dans `data/amis/[NOM]/`.
   - Chaque image est traitée par InsightFace pour générer un vecteur d'embedding.
   - Les embeddings sont sauvegardés dans un fichier `.npy` pour accélérer les futures requêtes.

2. **Phase de reconnaissance** :
   - L'utilisateur téléverse une photo test.
   - Le système extrait l'embedding du visage détecté.
   - Une similarité cosinus est calculée entre l'embedding test et tous ceux de la base.
   - Les 3 meilleurs matches sont retournés avec leur score.

2.3. Interface utilisateur

FIGURE 6 – Interface utilisateur complète de l'application Doppelganger

L'interface de l'application est divisée en deux parties principales :

1. **Panier de contrôle (gauche) :**
   - **Configuration de la Base** : Affiche l'état de la base de données (13 visages chargés)
   - **Bouton de génération** : Permet de générer ou mettre à jour les embeddings
   - **Zone de téléversement** : Permet de sélectionner une image test (formats JPG, JPEG, PNG, limite 200MB)
2. **Interface principale (droite) :**
   - **Titre et description** : Présente l'application "Doppelganger - Reconnaissance Faciale ArcFace"
   - **Zone de résultats** : Affichera les résultats de la ressemblance après analyse
   - **Section interactive** : Lance la comparaison et montre le top 3 des ressemblances

Caractéristiques de l'interface

- Interface responsive et moderne grâce à Streamlit
- Barre de progression lors du traitement des images
- Gestion intuitive des fichiers par drag-and-drop
- Affichage en temps réel des résultats
- Design épuré avec une ergonomie optimisée

3. Difficultés Rencontrées et Solutions

3.1. Gestion des embeddings et performances

— **Problème** : La génération des embeddings pour une base importante pouvait ralentir l'application.

— **Solution** : Utilisation du cache Streamlit (`@st.cache_resource`, `@st.cache_data`) pour ne charger le modèle et les embeddings qu'une seule fois.

3.2. Détection faciale sur images variées

— **Problème** : Certaines photos ne contenaient pas de visage détectable (angles, luminosité).

— **Solution** : Ajout de messages d'erreur explicites et conseils à l'utilisateur.

3.3. Conversion des scores pour l'affichage

— **Problème** : Le score de similarité (numpy float) causait une erreur dans `st.progress()`.

— **Solution** : Conversion explicite en `float()` Python avant l'affichage.

3.4. Structure des dossiers

— **Problème** : Nécessité d'une arborescence stricte pour associer un nom à chaque visage.

— **Solution** : Organisation en sous-dossiers nommés (`data/amis/Nom/photo1.jpg`).

4. Résultats Obtenus

4.1. Fonctionnalités implémentées

Fonctionnalités principales

- Extraction d'embeddings avec ArcFace
- Calcul de similarité cosinus
- Interface Streamlit interactive et intuitive
- Gestion d'une base de visages modulable
- Affichage du top 3 avec scores et exemples visuels
- Barre de progression lors de la génération des embeddings

4.2. Exemple de sortie

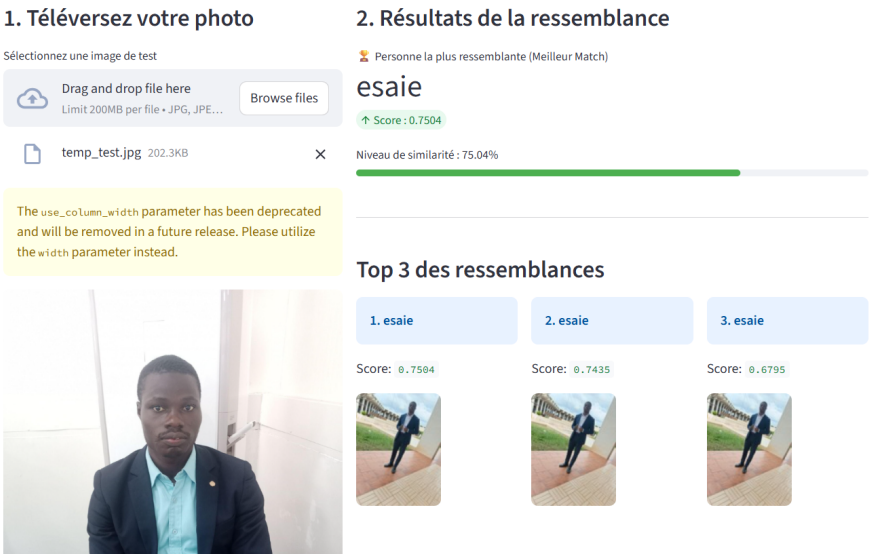

FIGURE 7 – Exemple 1 : Match avec "Esaie" (Score : 0.7504)

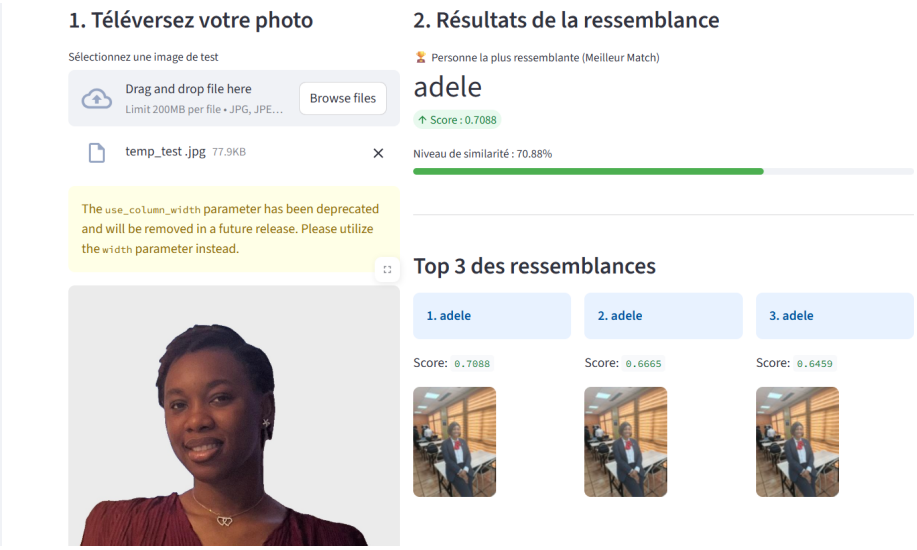

FIGURE 8 – Exemple 2 : Match avec "Adèle" (Score : 0.7088)

Pour une photo test de l'utilisateur :

Exemple	Meilleur match	Score
Exemple 1	"Esaie"	0.7504
Exemple 2	"Adèle"	0.7088

TABLE 1 – Résultats des tests de reconnaissance

4.3. Limites identifiées

⚠ Limites du système

- Sensible à la qualité et à l'orientation des visages
- Nécessite un visage unique par image test
- Performance réduite sur CPU pour de grandes bases

5. Conclusion et Perspectives

Le projet **Doppelganger** répond aux objectifs initiaux : il permet de trouver efficacement les ressemblances faciales dans une base prédéfinie. L'utilisation d'ArcFace assure une bonne robustesse, et Streamlit rend l'outil accessible sans compétences techniques.

Améliorations envisageables

- **Ajout de la détection multi-visages**
- **Intégration d'une base de données externe (SQLite)**
- **Option de seuillage de similarité**
- **Version déployée en ligne via Streamlit Cloud**

Perspectives d'évolution : Le système pourra être enrichi avec des fonctionnalités avancées de traitement d'images et d'apprentissage profond pour améliorer sa précision et son adaptabilité à différents contextes d'utilisation.

6. Références Techniques

— **InsightFace** : `https://github.com/deepinsight/insightface`

— **Streamlit** : `https://streamlit.io`

— **ArcFace paper** : Deng et al., "ArcFace : Additive Angular Margin Loss for Deep Face Recognition", 2019

— **OpenCV Documentation** : `https://docs.opencv.org`

— **Scikit-learn Documentation** : `https://scikit-learn.org`