RAPPORT DE SYNTHÈSE — CHALLENGE 2 (RECONNAISSANCE FACIALE, ESTIMATION D'ÂGE, OCR & FRAUDE)

Ce rapport présente une synthèse des trois tâches du Challenge 2 (répertoire `n2/`) : - Tâche 1 — Reconnaissance faciale robuste - Tâche 2 — Estimation d'âge - Tâche 3 — OCR & détection de fraude

Le document décrit les jeux de données, l'architecture des modèles, les résultats (checkpoints et soumissions présents dans `n2/outputs/`), les choix techniques, et des recommandations opérationnelles pour déploiement ou amélioration.

1. DONNÉES ET ORGANISATION

Les jeux de données sont organisés sous `n2/data/` : - `dataset_tache_1/` : images train/test nommées `XXXX_0.jpg`, `XXXX_1.jpg` (identités multiples par sujet) - `dataset_tache_2/` : images avec l'âge encodé dans le nom de fichier (suffixe numérique) - `dataset_tache_3/` : dossiers pays contenant sous-dossiers `normal`, `forgery_1..4`, et `gt` pour les étiquettes

Sorties déjà présentes : - Checkpoints pour les tâches (ex. `n2/outputs/tache1/checkpoints/ckpt_ep10.pth`, `tache2` ckpt_ep6.pth, `tache3` ckpt_ep1.pth) - Submissions CSV : - `n2/outputs/tache1/submissions/task1_matching_submission.csv` - `n2/outputs/tache2/submissions/task2_age_submission.csv` - `n2/outputs/tache3/submissions/task3_ocr_fraud_submission.csv` (généré lors d'évaluations)

2. ARCHITECTURES ET CHOIX MÉTHODOLOGIQUES

Résumé des architectures et choix présents dans les scripts :

Tâche 1 — Reconnaissance faciale - Backbone : ResNet18 (facilement remplaçable par ResNet50 ou modèles spécialisés comme ArcFace) - Embeddings L2-normalisés (taille configurable, défaut 512) - Option : classification head pour fine■tuning - Matching : Faiss si disponible, sinon sklearn NearestNeighbors - Prétraitement : augmentation (RandomResizedCrop, Flip, ColorJitter) et normalisation ImageNet - Stratégie d'inférence : moyenne des embeddings par PID pour construire la « galerie » puis nearest neighbor pour chaque image test

Tâche 2 — Estimation d'âge - Backbone : ResNet18 - Double■tête : classification sur bins 0..100 + régression continue - Loss combinée : CrossEntropy + MSE (pondération configurable) - Prétraitement : crop et normalisation; stratégie de prédiction : combinaison 50/50 entre prédiction par bins et régression

Tâche 3 — OCR & Détection de fraude - Backbone : ResNet18 + tête classification (5 classes: normal, forgery_1..4) - Extraction de texte : EasyOCR pour récupérer le contenu textuel des documents - Pipeline : classification visuelle + OCR pour enrichir les features (texte + visuel)

3. RÉSULTATS ET ÉTAT ACTUEL

Les checkpoints trouvés dans `n2/outputs/` indiquent que des entraînements ont été menés pour chaque tâche. - Tâche 1 : checkpoints jusqu'à `ckpt_ep10.pth` (indique 10 epochs sauvegardés) - Tâche 2 : checkpoints jusqu'à `ckpt_ep6.pth` - Tâche 3 : checkpoint initial `ckpt_ep1.pth`

Des fichiers de soumission existent (ex. `task1_matching_submission.csv`, `task2_age_submission.csv`) — ceux-ci peuvent servir de base pour évaluation et benchmark.

Remarque : Aucun journal d'évaluation (metrics détaillées) n'est inclus dans le dépôt; il est recommandé d'ajouter des fichiers `metrics_epoch_*.json` lors de l'entraînement pour suivi.

4. ÉVALUATION RAPIDE — QUALITÉ ET AMÉLIORATIONS POSSIBLES

Observations générales : - Les architectures sont standards et faciles à reproduire. Elles sont adaptées pour prototypage mais peuvent être améliorées pour production. - Pour la reconnaissance faciale, l'utilisation d'ArcFace ou de losses metric-learning (triplet, cosface) augmenterait la séparation des embeddings. - Pour l'estimation d'âge, la discrétisation actuelle est raisonnable ; toutefois, l'utilisation de réseaux plus larges ou d'ensembles peut réduire le MAE. - Pour l'OCR, EasyOCR est pratique mais peut être lent; pour la production, envisager l'utilisation de moteurs plus rapides ou de prétraitements (deskew, contrast) pour améliorer la lecture.

Risques et limites : - Données déséquilibrées par pays/âge/type de falsification → biais potentiel

des modèles. - Manque d'annotations riches (bounding boxes, segmentation) limitant l'amélioration de la détection de fraude. - Performances GPU non vérifiées ici; les checkpoints suggèrent entraînements mais pas forcément performances optimales.

5. RECOMMANDATIONS ACTIONNABLES

Priorités immédiates (court terme) 1. Mettre en place scripts d'évaluation et métriques (MAE pour âge, précision/R@k pour matching, F1/AUC pour fraude) et sauvegarder les métriques par epoch. 2. Standardiser les chemins et ajouter un `run_experiment.sh` pour lancer entraînement/éval de manière reproductible. 3. Échantillonner et inspecter manuellement quelques soumissions CSV pour vérifier la qualité (ex. correspondances proposées dans task1).

Améliorations techniques (moyen terme) 1. Remplacer ResNet18 par un backbone plus performant (ResNet50, EfficientNet, ou MV2) et/ou utiliser des poids pré-entrainés robustes. 2. Pour la reconnaissance faciale : intégrer une loss metric-learning (ArcFace, CosFace) et augmenter les stratégies d'augmentation (random erasing, mixup). 3. Pour l'estimation d'âge : expérimenter l'approche ordinal regression (considérer l'ordre des classes d'âge) et calibrer la combinaison classification+régression. 4. Pour l'OCR/fraude : combiner features visuels + features textuels extraits par OCR dans un modèle multimodal (fusion late ou via embeddings BERT pour le texte).

Mise en production (long terme) 1. Dockeriser le pipeline d'inférence (image légère + dépendances) et créer une API REST pour inference (FastAPI + torchscript ou ONNX runtime). 2. Mettre en place un pipeline CI qui exécute tests rapides (smoke tests) et une validation sur un petit jeu d'évaluation. 3. Surveillance post■déploiement : monitorer dérive des données (data drift) et performances (A/B evaluation si possible).

6. PLAN D'EXPÉRIMENTATION PROPOSÉ (3–6 MOIS)

Phase 1 (T1) — Reproductibilité & métriques - Ajout de scripts d'évaluation et sauvegarde métriques. - Baseline: reproduire run avec les checkpoints existants et calculer MAE/accuracy/R@1.

Phase 2 (T2) — Amélioration modèles - Reconnaissance: ArcFace + augmentation avancée. - Estimation: expérimentations ordinal regression vs. multi-task. - OCR: pipeline multimodal texte+image.

Phase 3 (T3) — Production - Dockerisation, API, tests automatisés, déploiement pilote sur serveur avec GPU.

7. ANNEXES — COMMANDES UTILES

Exemples d'exécution (depuis `n2/`):

Train tâche 1 : ``` python Tache_1_Reconnaissance_Faciale.py --mode train --data_dir data/dataset_tache_1/dataset_tache_1 --out_dir outputs/tache1 --epochs 10 ```

Eval tâche 2 : ``` python Tache_2_Estimation_Age.py --mode eval --data_dir data/dataset_tache_2/dataset_tache_2 --out_dir outputs/tache2 --resume outputs/tache2/checkpoints/ckpt_ep6.pth --cpu ```

Train tâche 3 : ``` python Tache_3_OCR_Fraude.py --mode train --data_dir data/dataset_tache_3/dataset_tache_3/train --out_dir outputs/tache3 --epochs 5 ```

Si vous le souhaitez, je peux : - Lancer localement (dans cet environnement) une exécution d'évaluation rapide sur un sous-échantillon pour produire metrics et vérifier les checkpoints ; - Générer un rapport PDF similaire à celui du Challenge 1 ; - Créer un commit proposant ce rapport et les changements liés (README amélioré, scripts d'éval).

Indiquez quelle action vous souhaitez que je fasse ensuite (par ex. exécution d'un test d'inférence rapide pour l'une des tâches).