Rapport d'avancement 1 - Classification de Documents Administratifs

1. Composition du trinôme

Responsable d'équipe : Zaynab ER-RGHAY

Membres :

- Zaynab ER-RGHAY
- Bilal LAHFARI
- Sami MALEK

---

2. Lien vers le projet GIT

Profils des développeurs :

- https://github.com/zaykats
- https://github.com/SamiMalek10
- https://github.com/D0esN0tM1tter

Repository : `[ l'URL du repository Github]`

(Invitation envoyée à l'adresse du professeur : reda.chefira@ensam.um5.ac.ma)

---

3. Architecture du modèle - Version 1

📥 INPUT

- **Format :** Fichiers PDF (mono ou multi-pages)
- **Catégories cibles :** 5 classes
  - Pièce d'identité (CNIE Recto/Verso)
  - Relevé bancaire
  - Facture d'électricité
  - Facture d'eau
  - Document employeur (bulletins de paie + attestations)

🔄 PIPELINE DÉTAILLÉ

Phase 1 : Prétraitement des Documents

Module : `pdf_processor.py`

1. **Conversion PDF → Images**
   - Utilisation de `pdf2image` avec résolution 300 DPI
   - Support multi-pages
   - Gestion des PDFs corrompus
2. **Amélioration de la qualité d'image**
   - Débruitage (fastNlMeansDenoising)
   - Amélioration du contraste (CLAHE)
   - Correction d'inclinaison (deskewing)
   - Binarisation adaptative pour l'OCR
3. **Double prétraitement**

- Optimisation pour OCR (binarisation, contraste élevé)
- Normalisation pour CNN (ImageNet standards)

Phase 2 : Module Computer Vision

Modules : `template_detector.py` + `hybrid_classifier.py`

A. Détection de Gabarits Structurels

Extraction de features visuelles sans dépendre du contenu textuel :

Feature	Méthode	Utilité
Ratio d'aspect	Calcul H/W	Distinction carte (1.5-1.7) vs A4 (1.3-1.5)
Détection photo	Haar Cascade (OpenCV)	Identification CNIE
Structure tabulaire	Transformée de Hough	Détection relevés/factures
Densité de texte	Analyse binarisation	Caractérisation documents
Zone signature	Analyse variance locale	Identification documents employeur

B. Classification Hybride CV

- **Backbone :** ResNet50 pré-entraîné (ImageNet)
- **Architecture :**

```
Image → ResNet50 (features 2048D) ─┐
                                   ├─→ Concaténation → Dense → Softmax (5 classes)
Features Gabarits (64D) ───────────┘
```

- **Stratégie :** Fusion des features CNN deep learning + features structurelles explicites

---

Phase 3 : Module NLP

Modules : `ocr_extractor.py` + `pattern_matcher.py` + `camembert_classifier.py`

A. Extraction de Texte (OCR)

- **Moteur :** Tesseract OCR (langue française)
- **Configuration :** LSTM engine (--oem 3) + mode bloc uniforme (--psm 6)
- **Post-traitement :**
  - Filtrage par score de confiance (> 60%)
  - Correction d'erreurs contextuelles (O→0, I→1 dans chiffres)
  - Calcul confiance moyenne par document

B. Classification par Motifs Sémantiques (Baseline)

Dictionnaires de mots-clés spécifiques par classe :

- **Identité :** "identité", "nationale", "CIN", "né(e)", "nationalité", "date émission"
- **Bancaire :** "solde", "débit", "crédit", "compte", "RIB", "opération"
- **Électricité :** "kWh", "électricité", "puissance", "ONE", "LYDEC", "consommation"
- **Eau :** "m³", "eau", "index", "RADEEMA", "AMENDIS"
- **Employeur :** "salaire", "employeur", "cotisations", "bulletin", "attestation"

Scoring :

- Pondération par longueur de mot (mots longs = plus spécifiques)
- Comptage des occurrences
- Normalisation sur échelle 0-1

C. Extraction de Patterns Spécifiques

Détection par regex :

- Montants (DH, MAD, €)

- Unités de mesure (kWh, m^3)
- Dates (formats DD/MM/YYYY)
- Identifiants (CIN, RIB)

D. Classification CamemBERT (Prévu)

- Modèle : CamemBERT-base (français)
- Méthode : Extraction embeddings [CLS] → classification
- Fine-tuning : Sur corpus documents administratifs marocains

---

Phase 4 : Fusion Multimodale Intelligente

Module : `multimodal_fusion.py`

Système expert combinant les prédictions CV et NLP selon des stratégies hiérarchiques :

Stratégie 1 : Accord Parfait (priorité maximale)

- Si CV et NLP prédisent la même classe avec confiance > 0.8
- Confiance finale = moyenne des deux
- Validation immédiate si règles métier respectées

Stratégie 2 : CV Fort + Validation Gabarits

- Si CV confiance > 0.9 ET score gabarit > 0.7
- Applicable aux documents très visuels (CNIE, relevés structurés)
- Confiance combinée = 70% CV + 30% gabarits

Stratégie 3 : NLP Fort + Motifs Textuels

- Si NLP confiance > 0.9 ET patterns textuels forts
- Applicable aux documents textuels (attestations, factures)
- Confiance combinée = 80% NLP + 20% patterns

Stratégie 4 : Vote Pondéré (en cas de désaccord)

- Score CV final = 60% confiance CNN + 40% score gabarits
- Score NLP final = 70% confiance + 30% force patterns
- Décision selon score maximal

Règles Métier (Post-validation)

Vérifications de cohérence par classe :

Classe	Contraintes obligatoires
CNIE	Photo détectée + ratio 1.5-1.7
Relevé bancaire	Structure tabulaire + montants
Facture électricité	Présence "kWh" + montants
Facture eau	Présence "m^3" + montants
Document employeur	Montants salariaux + signature

Système de Rejet

- Seuil de rejet : confiance < 0.6
- Documents ambigus → dossier "À vérifier"
- Violations règles métier → réduction confiance 30-50%

Phase 5 : Post-traitement et Sortie

Module : `main.py` (pipeline principal)

1. **Tri automatique** dans dossiers par classe

2. **Gestion des rejets** (dossier "a_verifier")
3. **Génération rapport JSON** avec :
    - Classe prédite + confiance globale
    - Scores CV et NLP individuels
    - Chemin de décision (quelle stratégie utilisée)
    - Features extraites (gabarits, patterns)
    - Temps de traitement
    - Métriques OCR

📤 OUTPUT

Structure des dossiers de sortie :

```
data/output/
├── identite/
├── releve_bancaire/
├── facture_electricite/
├── facture_eau/
├── document_employeur/
├── a_verifier/              # Documents ambigus (confiance < 0.6)
└── classification_report.json  # Rapport détaillé
```

Contenu du rapport JSON :

json

```
{
 "document.pdf": {
   "results": [{
     "page_number": 1,
     "predicted_class": "identite",
     "confidence": 0.87,
     "decision_path": "perfect_agreement",
     "rejected": false,
     "cv_prediction": "identite",
     "cv_confidence": 0.85,
     "nlp_prediction": "identite",
     "nlp_confidence": 0.89,
     "template_scores": {...},
     "pattern_scores": {...}
   }],
   "processing_time": 4.32,
   "pages_count": 1
 }
}
```

---

4. État d'Avancement Technique

✅ Réalisations Complètes

Infrastructure (100%)

- ☑ ~~Structure modulaire du projet créée~~
- ☑ ~~Système de gestion offline des modèles (`OfflineModelManager`)~~
- ☑ ~~Configuration centralisée (`config.py`)~~
- ☑ ~~Script d'installation automatique (`setup_offline.py`)~~
- ☑ ~~Documentation complète (README.md)~~
- ☑ ~~Gestion des dépendances (requirements.txt)~~
- ☑ ~~Système de logging intégré~~

Module Prétraitement (100%)

- ☑ ~~Conversion PDF → Images (pdf2image)~~

- [x] ~~Amélioration qualité d'image (débruitage, contraste, CLAHE)~~
- [x] ~~Correction d'inclinaison (deskewing)~~
- [x] ~~Double pipeline (OCR + CNN)~~
- [x] ~~Gestion erreurs et PDFs corrompus~~

Module Computer Vision (85%)

- [x] ~~**Détecteur de gabarits complet :**~~
  - [x] ~~Calcul ratio d'aspect~~
  - [x] ~~Détection photo (Haar Cascade)~~
  - [x] ~~Détection structure tabulaire (Hough)~~
  - [x] ~~Calcul densité de texte~~
  - [x] ~~Détection zones signature~~
  - [x] ~~Scoring par correspondance de gabarit~~
- [x] ~~Prétraitement images pour CNN (normalisation ImageNet)~~
- [x] ~~Chargement ResNet50 offline~~
- [ ] **Fine-tuning ResNet50** (non commencé - prévu phase 2)
- [ ] **Architecture hybride entraînée** (non commencé - prévu phase 2)

Note : Version actuelle utilise les scores de gabarits comme baseline pour CV

Module NLP (75%)

- [x] ~~**Extracteur OCR complet :**~~
  - [x] ~~Configuration Tesseract optimale~~
  - [x] ~~Extraction avec scores de confiance~~
  - [x] ~~Correction erreurs post-OCR~~
  - [x] ~~Détection langue~~
- [x] ~~**Pattern Matcher fonctionnel :**~~
  - [x] ~~Dictionnaires mots-clés 5 classes~~
  - [x] ~~Scoring pondéré~~
  - [x] ~~Extraction patterns spécifiques (montants, dates, unités)~~
- [x] ~~Chargement CamemBERT offline~~
- [ ] **Fine-tuning CamemBERT** (non commencé - prévu phase 2)
- [ ] **Architecture RNN + Attention** (optionnel - prévu phase 3)

Note : Version actuelle utilise pattern matching comme baseline pour NLP

Module Fusion (100%)

- [x] ~~Stratégie accord parfait~~
- [x] ~~Stratégie CV fort + gabarits~~
- [x] ~~Stratégie NLP fort + patterns~~
- [x] ~~Vote pondéré en cas de désaccord~~
- [x] ~~Système de règles métier par classe~~
- [x] ~~Gestion rejets et seuils~~
- [x] ~~Calcul confiance combinée~~
- [x] ~~Logging des décisions~~

Pipeline Principal (100%)

- ✅ ~~Orchestration complète des modules~~
- ✅ ~~Traitement par lots de PDFs~~
- ✅ ~~Gestion multi-pages~~
- ✅ ~~Tri automatique par dossiers~~
- ✅ ~~Génération rapport JSON détaillé~~
- ✅ ~~Interface CLI avec arguments~~
- ✅ ~~Barre de progression (tqdm)~~
- ✅ ~~Gestion erreurs et timeouts~~

🔄 En Cours / Prochaines Étapes

Phase Actuelle : Collecte de Données et Tests

1. Constitution du Dataset (Priorité 1 - Semaine en cours)

Objectif : 20-30 exemples par classe (100-150 documents au total)

Classe	Exemples Requis	Sources	Statut
CNIE	20-30 (R+V)	Simulation + exemples anonymisés	🔄 En cours
Relevés bancaires	20-30	BMCE, Attijariwafa, CIH, BP...	🔄 En cours
Factures électricité	20-30	ONE, LYDEC, RADEEMA, REDAL	🔄 En cours
Factures eau	20-30	AMENDIS, RADEEMA, REDAL	🔄 En cours
Docs employeur	20-30	Bulletins paie + attestations	🔄 En cours

Critères de qualité du dataset :

- Variété de qualités (excellente, moyenne, dégradée)
- Différentes régies/banques
- Formats variés (scan, photo, PDF natif)
- Documents récents (2023-2025)

2. Tests du Pipeline Baseline (Priorité 1 - Cette semaine)

- ☐ Test sur 5 documents par classe (validation rapide)
- ☐ Vérification fonctionnement offline
- ☐ Mesure temps de traitement initial
- ☐ Identification bugs critiques
- ☐ Ajustement seuils si nécessaire

3. Évaluation des Performances Baseline (Priorité 2 - Semaine prochaine)

Métriques à calculer :

- ☐ Accuracy globale
- ☐ Précision/Rappel/F1-score par classe
- ☐ Matrice de confusion
- ☐ Temps moyen de traitement
- ☐ Taux de rejet
- ☐ Analyse des erreurs

Objectifs minimums Version 1 :

- Accuracy globale > 70%
- Temps traitement < 10s/document
- Taux rejet < 20%

📋 Roadmap des Améliorations Prévues

Phase 2 : Optimisation des Modèles (Semaines 3-4)

A. Fine-tuning ResNet50

- Entraînement sur dataset collecté
- Freeze des premières couches
- Learning rate faible (1e-4)
- Augmentation de données
- Objectif : +10-15% accuracy CV

B. Fine-tuning CamemBERT

- Adaptation au domaine administratif marocain
- Entraînement sur textes extraits
- Classification directe ou extraction features
- Objectif : +10-15% accuracy NLP

C. Optimisation Fusion

- Ajustement pondérations selon résultats réels
- Enrichissement règles métier
- Calibration seuils de confiance
- Objectif : +5-10% accuracy globale

Phase 3 : Extensions et Robustesse (Semaines 5-6)

A. Robustesse

- Gestion documents dégradés
- Amélioration prétraitement
- Multi-scale detection
- Ensembling de modèles

B. Interface Utilisateur

- Interface web Streamlit
- Upload et visualisation
- Feedback utilisateur
- Statistiques en temps réel

C. Optimisation Performance

- Support GPU (CUDA)
- Batch processing optimisé
- Modèles légers (MobileNet, DistilBERT)
- Cache intelligent

Phase 4 : Validation Finale (Semaine 7)

- Tests exhaustifs sur dataset complet

- Validation croisée
- Benchmarking final
- Documentation technique complète
- Préparation présentation

---

5. Stack Technique Implémentée

Langages et Frameworks

- **Python 3.8+** : Langage principal
- **PyTorch 1.9+** : Deep Learning framework
- **Transformers 4.20+** : Modèles NLP (HuggingFace)
- **OpenCV 4.5+** : Computer Vision

Bibliothèques Principales

Domaine	Bibliothèque	Version	Usage
PDF	pdf2image	1.16+	Conversion PDF→Images
CV	torchvision	0.10+	ResNet50, transforms
CV	opencv-python	4.5+	Traitement images, gabarits
NLP	pytesseract	0.3.8+	OCR Tesseract
NLP	transformers	4.20+	CamemBERT
Utilitaires	numpy	1.21+	Calculs numériques
Utilitaires	tqdm	4.62+	Barres de progression
Logging	logging	Built-in	Gestion logs

Modèles Pré-entraînés

- **ResNet50** : ImageNet (85.5M paramètres) - Classification images
- **CamemBERT-base** : Corpus français (110M paramètres) - NLP français
- **Haar Cascade** : OpenCV - Détection visages

Environnement

- Environnement virtuel Python (venv)
- Fonctionnement 100% offline après setup
- Tesseract OCR avec langue française
- Compatible Linux/macOS/Windows

---

6. Organisation du Travail en Trinôme

Répartition des Tâches

Membre	Modules Principaux	Responsabilités
Zaynab ER-RGHAY(Chef)	Fusion + Coordination	·Fusion multimodale .Intégration modules .Tests globaux . Coordination équipe
Bilal LAHFARI	Computer Vision	·Détection gabarits .Fine-tuning ResNet50 . Pipeline CV
Sami MALEK	NLP	·OCR + Pattern matching .Fine-tuning CamemBERT .Pipeline NLP

Modules Communs :

- Configuration : Zaynab
- Prétraitement : Bilal
- Pipeline principal : Zaynab
- Documentation : Tous
- Collecte données : Tous

Organisation Git

Branches :

- `main` : Version stable
- `develop` : Développement actif
- `feature/cv-module` : Module CV (Bilal)
- `feature/nlp-module` : Module NLP (Sami)
- `feature/fusion` : Module Fusion (Zaynab)

Workflow :

1. Développement sur branches feature
2. Pull requests avec revue de code
3. Merge sur develop après validation
4. Merge develop → main pour releases

Communication

- **Réunions quotidiennes** : 15 min (standup)
- **Points hebdomadaires** : 1h (revue progrès + planification)
- **Outils** : Discord/WhatsApp + GitHub Issues
- **Documentation** : Wiki GitHub + commentaires code

---

7. Métriques d'Évaluation Prévues

Métriques de Classification

Métrique	Description	Objectif
Accuracy globale	% documents bien classés	> 90%
Précision par classe	TP / (TP + FP)	> 90%
Rappel par classe	TP / (TP + FN)	> 90%
F1-Score par classe	Moyenne harmonique P/R	> 90%
Matrice confusion	Analyse erreurs croisées	Visualisation

Métriques de Performance

Métrique	Description	Objectif
Temps traitement	Moyenne par document	< 5s
Temps extraction OCR	OCR seul	< 2s
Temps classification	CV + NLP	< 3s
Usage RAM	Mémoire moyenne	< 2GB
Taux rejet	% documents "À vérifier"	< 10%

Métriques de Robustesse

- Performance sur documents dégradés (bruit, flou)

- Consistance inter-régies (différentes banques/régies)
- Stabilité sur plusieurs exécutions
- Gestion documents multi-pages

Plan de Test

Dataset de Test (15% du total) :

- Séparation stricte entraînement/validation/test
- 3-5 documents par classe en test
- Variété de qualités et sources

Tests de Stress :

- Documents très dégradés
- PDFs complexes (50+ pages)
- Texte manuscrit
- Images avec arrière-plan complexe

8. Risques et Mitigation

Risque	Impact	Probabilité	Mitigation
Dataset insuffisant	Élevé	Moyen	Augmentation données + génération synthétique
OCR faible qualité	Moyen	Élevé	Amélioration prétraitement + correction post-OCR
Overfitting modèles	Moyen	Moyen	Early stopping + régularisation + augmentation
Temps traitement long	Faible	Faible	Optimisation + GPU + modèles légers
Divergence CV/NLP	Moyen	Moyen	Fusion robuste + règles métier

9. Livrables Finaux Prévus

Code et Documentation

- [x] ~~Repository Git structuré et documenté~~
- [x] ~~Code modulaire et commenté~~
- [x] ~~README complet avec instructions~~
- [ ] Tests unitaires (pytest)
- [ ] Documentation technique (Sphinx)

Modèles

- [x] ~~Modèles pré-entraînés téléchargés~~
- [ ] Modèles fine-tunés sur dataset
- [ ] Fichiers de poids sauvegardés
- [ ] Rapports de performance

Interface

- [x] ~~Interface CLI fonctionnelle~~
- [ ] Interface web Streamlit (optionnel)
- [ ] Visualisation des résultats

Rapports

- [ ] Rapport technique détaillé
- [ ] Analyse des performances
- [ ] Matrice de confusion
- [ ] Comparaison des approches

---

10. Conclusion et Prochains Jalons

Résumé de l'Avancement

État actuel : ~80% de l'infrastructure complète

✅ **Forces :**

- Architecture solide et modulaire
- Pipeline fonctionnel bout-en-bout
- Fusion multimodale intelligente
- Système offline opérationnel
- Documentation exhaustive

⚠️ **Points critiques restants :**

- Collecte du dataset (en cours)
- Tests du pipeline baseline
- Fine-tuning des modèles deep learning

Prochains Jalons

Semaine	Objectif Principal	Livrables
S2 (actuelle)	Dataset + Tests baseline	·100-150 documents Rapport tests initiaux Métriques baseline
S3-S4	Fine-tuning modèles	·ResNet50 entraîné CamemBERT entraîné Métriques améliorées
S5	Optimisation + Interface	·Fusion optimisée Interface web Tests robustesse
S6	Validation finale	·Tests exhaustifs Documentation finale Rapport complet

Engagement de l'Équipe

Notre trinôme s'engage à :

- Maintenir une communication régulière
- Respecter les délais fixés
- Produire un code de qualité
- Documenter rigoureusement notre travail
- Collaborer efficacement via Git
- Atteindre les objectifs de performance (> 90% accuracy)

Date du rapport : 30 Novembre 2025

Version : 1.0

Statut : Infrastructure complète - Phase collecte données en cours