Équipe Implémentation — Tâches et Sous-tâches

Projet : Fader Networks

---

1. Objectif général

L'équipe Implémentation est responsable de :

- la reconstruction complète du modèle Fader Networks à partir du papier original,
- la création des modules Encoder, Decoder et Discriminator,
- l'intégration des fonctions de perte et du mécanisme adversarial,
- l'entraînement du modèle en utilisant le pipeline fourni par l'équipe Data,
- l'évaluation et la comparaison avec le modèle officiel préentraîné,
- la documentation technique et la production de résultats visuels.

2. 1. Mise en place de l'environnement

2.1. 1.1 Installation des dépendances

- Installation de PyTorch, torchvision, numpy, matplotlib, tqdm.
- Vérification de la compatibilité GPU (CUDA).
- Installation d'outils optionnels : TensorBoard, Pillow, sklearn.

2.2. 1.2 Création de l'architecture du projet

- Création de la structure suivante :

```
FaderNetworks/
  models/
      encoder.py
      decoder.py
      discriminator.py
  training/
      losses.py
      trainer.py
  utils/
      helpers.py
```

```
outputs/
main.py
```

— Mise en place d'un environnement de tests local ou Google Colab.

3. 2. Implémentation du modèle Fader Networks

3.1. 2.1 Conception de l'Encoder

— Implémenter un CNN descendant :

— C16 → C32 → C64 → C128 → C256 → C512.

— Utiliser LeakyReLU et BatchNorm.

— Produire un latent code z de dimension $512 \times 2 \times 2$.

3.2. 2.2 Conception du Decoder

— Implémenter un réseau de transposed convolutions symétrique à l'encoder.

— Injecter les attributs y à chaque couche (concaténation de canaux).

— Utiliser ReLU dans les couches internes et une activation Tanh en sortie.

3.3. 2.3 Conception du Discriminator

— Réseau : 1 couche convolutionnelle + 2 couches fully-connected.

— Sortie : probabilité pour chaque attribut.

— Ajouter Dropout (0.3) pour stabiliser l'entraînement.

4. 3. Mise en place des fonctions de perte

4.1. 3.1 Reconstruction loss

$$L_{AE} = \|D(E(x), y) - x\|^2$$

4.2. 3.2 Discriminator loss

$$L_{dis} = -\log P(y|E(x))$$

4.3. 3.3 Encoder adversarial loss

$$L_{enc} = L_{AE} - \lambda \log P(1 - y|E(x))$$

— Implémentation du paramètre λ croissant (scheduling).

— Gestion des gradients séparés pour chaque module.

5. 4. Entraînement du modèle

5.1. 4.1 Structure de la boucle d'entraînement

— Pour chaque batch :

1. **Étape 1 :** Mise à jour du Discriminator sur (z, y).
2. **Étape 2 :** Mise à jour Encoder + Decoder pour tromper le Discriminator.
3. **Étape 3 :** Mise à jour du Decoder pour la reconstruction correcte.

— Sauvegarde des checkpoints à intervalles réguliers.

5.2. 4.2 Monitoring et logs

— Imprimer périodiquement :

— L_{AE} (Reconstruction),

— L_{dis} (Prédiction attributs),

— L_{enc} (Objectif adversarial).

— Suivi via TensorBoard ou courbes matplotlib.

6. 5. Évaluation du modèle

6.1. 5.1 Reconstruction des images

— Comparer image originale x et image reconstruite $\hat{x}$.

— Sauvegarder des exemples visuels.

6.2. 5.2 Modification d'attributs

— Modifier un ou plusieurs attributs (ex : lunettes, sourire).

— Générer des images modifiées x' via :

$$x' = D(E(x), y')$$

- Analyser :
  - Pertinence du changement,
  - Réalisme de l'image,
  - Préservation de l'identité.

6.3. 5.3 Comparaison avec le modèle officiel

- Charger le modèle préentraîné (Facebook AI Research).
- Comparer :
  - Reconstruction,
  - Qualité des modifications d'attributs,
  - Naturalité,
  - Cohérence visuelle.
- Construire un tableau comparatif.

7. 6. Documentation et production des résultats

7.1. 6.1 Documentation technique

- Décrire l'architecture exacte (dimension des couches).
- Documenter le mécanisme adversarial.
- Détailler les hyperparamètres utilisés.

7.2. 6.2 Archivage des résultats

- Sauvegarde des reconstructions.
- Sauvegarde des modifications d'attributs.
- Sauvegarde des checkpoints du modèle.

7.3. 6.3 Contribution au rapport final

- Rédaction de la section "Méthodologie / Implémentation".
- Production des figures, schémas et tableaux.
- Explication des différences avec le modèle officiel.

Résumé des responsabilités de l'équipe Implémentation

— Construire un modèle Fader Networks fidèle au papier original.

— Implémenter correctement l'entraînement adversarial.

— Produire des résultats reproductibles et de haute qualité.

— Comparer la réimplémentation au modèle officiel.

— Assurer la documentation complète du travail.