INF7370 Apprentissage automatique – Hiver 2025

Professeur : Mohamed Bouguessa

Travail pratique 3

Envoyé le : 26 mars 2025

Date de remise : 3 mai 2025 à 9h30

Important

1. Le TP est noté sur 15 points.
2. Le travail peut être fait individuellement ou en équipe de deux personnes.
3. Vous devez travailler seulement avec les deux fichiers : 1_Modele_TP3.py ET 2_Evaluation_TP3.py
4. Ne changez pas les noms de ces deux fichiers.
5. Inclure votre code dans les endroits spécifiés dans ces deux fichiers.
6. Indiquer votre nom dans les deux fichiers.
7. Remettre quatre fichiers :
    a. 1_Modele_TP3.py
    b. 2_Evaluation_TP3.py
    c. Le fichier du modèle Keras. Donc il faut sauvegarder et récupérer votre modèle avec **le minimum loss (perte)** en format keras.
    d. Votre rapport **en format PDF**. Le rapport doit suivre les directives mentionnées dans l'énoncé.
8. Remettre les fichiers dans **un seul fichier zip identifié à votre nom**. N'utilisez pas un nom générique (exemple : TP3).
9. Ne pas remettre les fichiers des images.
10. La date de remise est **le 3 mai 2025 à 9h30. Moodle ferme à 09h30**. **Aucun travail ne sera accepté à partir de cette heure et ce peu importe les raisons techniques.**

Énoncé

Le but de ce travail est de développer un autoencodeur convolutif pour encoder et reconstruire les images de deux espèces marines: dauphins et requins. Par la suite, appliquer un SVM sur le embedding (l'espace de représentation/descripteurs) généré par l'encodeur afin de classer les images des dauphins et des requins.

Vous devez utiliser le langage Python avec la librairie Keras pour l'implémentation de l'autoencodeur. Il est recommandé d'utiliser Google Colab comme environnement de développement, car ce dernier offre la possibilité d'utiliser un GPU.

Ensemble de données

Pour entrainer votre modèle, il faut utiliser un ensemble de 4 200 images de deux espèces marines. Le tableau suivant résume la répartition des images en deux classes.

	Dauphins	Requins
Données d'entrainement	1 800	1 800
Données de test	300	300

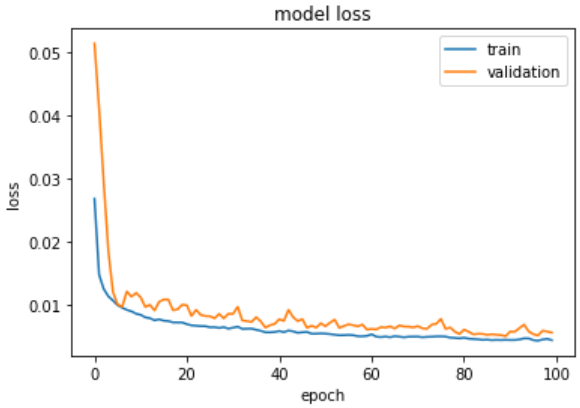
Tâche 1 : Montage de l'architecture et entrainement du modèle

En premier lieu il faut élaborer **votre propre** architecture de l'autoencodeur (le modèle) et l'entrainer sur les images fournies dans le dossier `donnees/entrainement`.

Il est à noter que les données d'entrainement (3 600 images au total) doivent être divisées en deux parties : Training et Validation. Vous avez la liberté de choisir les proportions de chaque ensemble. Ici, je souhaite mentionner que le but derrière la division des données d'entrainement en deux sous-ensembles (Training et Validation) est pour travailler avec un échantillon qui permet de trouver la meilleure architecture selon la perte (*loss*) obtenue via les données de validation.

Dans votre rapport, vous devez indiquer les informations suivantes :

- La taille de l'ensemble de Training ainsi que la taille de l'ensemble de Validation.
- Est-ce que vous avez effectué un prétraitement des données (*data augmentation* par exemple).
- L'optimisateur utilisé avec les paramètres associés à cet optimisateur.
- La taille du lot (*batch size*) d'entrainement.
- Le nombre d'époques (*number of Epochs*) et l'arrêt précoce s'il y a lieu.
- Le nombre de couches utilisées avec le type les paramètres de chaque couche. Par exemple :
    - Le nombre de couches de convolution avec la taille et le nombre des filtres utilisés.
    - Le nombre de couches d'échantillonnage (*pooling*) ainsi que la taille du filtre.
- Dropout : Oui/Non?
- Le type des fonctions d'activations.
- Le temps total d'entraînement en minutes.
- L'erreur minimale commise lors de l'entrainement (*Minimum Loss*).
- La courbe de perte (*loss*) par époque (Training vs Validation). À titre d'exemple, la figure suivante illustre nos résultats :



- Justifier vos choix en indiquant les facteurs ayant contribué à l'amélioration de l'entrainement.

IMPORTANT

Il faut utiliser le code du fichier 1_Modele_TP3.py (de l'exemple MNIST **associé au TP3** – disponible dans moodle – Section TP3) comme point de départ pour développer votre modèle. Insérer votre code dans les endroits indiqués à cette fin dans le fichier 1_Modele_TP3.py. Il faut ajuster l'architecture de votre autoencodeur dans les deux fonctions **encoder** et **decoder**. Insérer des commentaires dans le code afin d'expliquer vos choix.

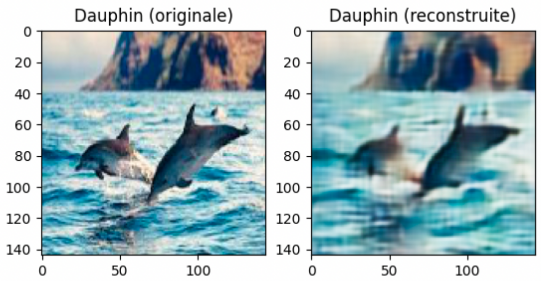
Tâche 2 : Évaluation du modèle

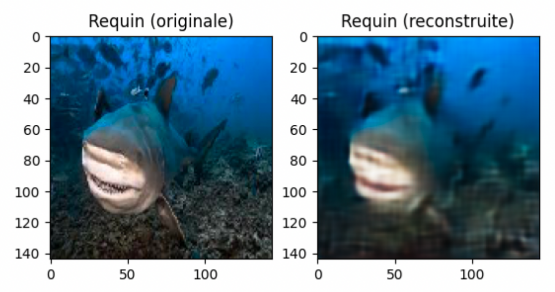
Le but de cette partie est d'évaluer le modèle élaboré à la Tâche 1. À cette fin, il faut **utiliser les données de test** disponible dans `donnees/test`. C'est un ensemble de 600 images contenant 300 images de chaque classe.

Premièrement, appliquer l'autoencodeur réalisé à la Tâche 1 sur les données de test afin de produire la reconstruction des images originales.

Dans votre rapport, vous devez :

- Extraire une image **de chaque classe** et montré sa forme originale et reconstruite. Voici un exemple.





Deuxièmement, prendre la partie encodeur de l'autoencodeur entraîné à la Tâche 1 et puis faire passer toutes les images des données de test. Ensuite, récupérer la sortie de l'encodeur qui représente le vecteur qui encode les données de test (dans ce qui suit, on va nommer ce vecteur de représentation associé données de test *le embedding*).

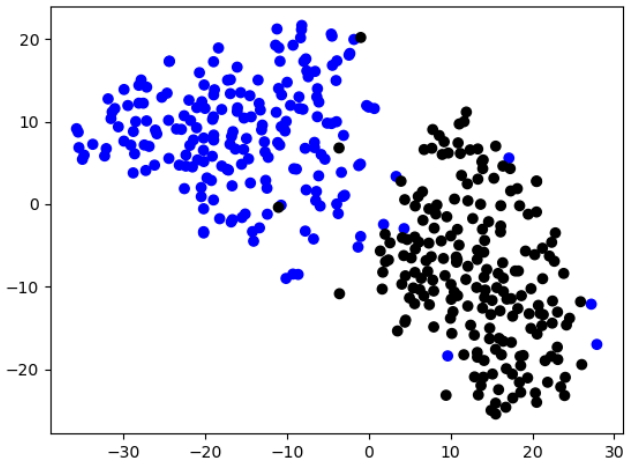
Troisièmement :

- Appliquer un **SVM[*] linaire** avec validation croisée sur le embedding.
- Afficher le résultat de l'entrainement du SVM avec validation croisée en utilisant l'exactitude (Accuracy).

Important : Votre but est d'avoir le meilleur résultat possible à la suite de l'application du SVM sur le embedding. Bien évidemment, plus la qualité des descripteurs (embedding) est meilleure, plus l'exactitude du SVM sera élevée. **Vous devez donc trouver l'architecture qui vous permet d'atteindre le meilleur résultat.**

Quatrièmement :

- Visualiser le embedding en deux dimensions : Afficher dans un scatter plot du embedding associé aux données tests en utilisant TSNE-2D[†]. Voici un exemple d'un scatter plot qui visualise en deux dimensions l'encodage des données de test MNIST (associées au TP3):



[*] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
[†] https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html

IMPORTANT

Il faut utiliser le code du fichier 2_Evaluation_TP3.py (de l'exemple MNIST) comme point de départ. Dans le code fourni, il manque quelques éléments comme l'entrainement du SVM et l'extraction des images reconstruites par espèce et le scatter plot. Insérer votre code dans les endroits indiqués à cette fin dans le fichier 2_Evaluation_TP3.py.

Tâche 3 : Conclusion

Conclure votre rapport en discutant, d'une façon générale, les problèmes rencontrés ainsi que les démarches possibles qui peuvent considérés pour améliorer votre modèle.

Squelettes de code

Afin de vous faciliter la tâche, nous avons préparé le code sur le dataset MNIST. Il y a deux fichiers : un pour la création du modèle et un autre pour l'évaluation du modèle. Le code est associé à l'élaboration et l'évaluation d'un autoencodeur convolutif pour l'encodage la reconstruction des chiffres manuscrits. Spécifiquement, le modèle est entrainé sur deux classes : les images associées au chiffre 2 et les images associées au chiffre 7. Ces images ont été extraites de l'ensemble de données MINST.

Nous avons élaboré le code pour bâtir un autoencodeur pour l'encodage la reconstruction des images à niveau de gris pour deux classes : les images représentant le chiffre 2 et les images représentant le chiffre 7. Ce code va vous aider à réaliser le TP3. Bien évidemment, il faut, l'adapter pour le problème du TP3 (images colorées de deux espèces marines). Veuillez passer en revue les deux fichiers de code et exécutez-les sur Colab.