

Université du Québec à Montréal

Rapport

Travail pratique #2

INF7370-Apprentissage automatique – Hiver 2025

Professeur Mohamed, Bouguessa

Préparer par

Bellune Tabitha Megane
BAGOU A. Ewoenam Gracia

7 avril 2025

Table des matières

1.Introduction.....	3
2. Montage de l'architecture et entraînement du modèle.....	4
2.1 Ensemble de données.....	4
2.2 Traitement de données.....	4
2.3 Paramètres et Hyperparamètres.....	4
2.4 Architecture.....	4
2.5 Affichage des résultats d'entraînement.....	10
2.6 Justification du choix de l'architecture.....	11
3. Évaluation du Modèle.....	11
4. Conclusion.....	14

1.Introduction

Dans ce projet, nous implémentons un modèle de réseau de neurones à convolution (CNN) capable de distinguer six espèces marines : baleine, dauphin, morse, phoque, requin et requin-baleine. Les données sont analysées puis divisées en trois ensembles : entraînement, validation et test. Nous justifions ensuite les choix architecturaux du modèle, en mettant en avant leur contribution à l'amélioration de l'apprentissage. Enfin, nous évaluons les performances du modèle en mesurant son exactitude et en identifiant les espèces les plus souvent mal classées. Nous présentons également les courbes d'apprentissage, la matrice de confusion et un graphe illustrant le comportement du modèle obtenu.

2. Montage de l'architecture et entraînement du modèle

2.1 Ensemble de données

Pour ce travail, le jeu de données est un ensemble de 30 000 images de six espèces marines (baleine, dauphin, morse, phoque, requin et requin-baleine) réparti comme suit : 18 000 images pour les données d'entraînement dont 3 000 pour chaque espèce; 6 000 images pour les données de validation dont 1 000 pour chaque espèce; et 6000 images pour les données de test.

2.2 Traitement de données

Les images ont été prétraitées en étant redimensionnées à une taille de 200x200 pixels et normalisées pour améliorer la convergence du modèle.

L'augmentation des données a été appliquée sur l'ensemble d'entraînement. Des transformations telles que la rotation, le déplacement, le zoom et l'ajustement de luminosité ont été utilisées pour augmenter la diversité des échantillons.

2.3 Paramètres et Hyperparamètres

L'entraînement du modèle a été réalisé avec l'optimiseur *Adam* et une fonction de perte catégorielle cross-entropie pour la classification multiclasse. Nous avons choisi Adam en ce qu'il permet une convergence plus rapide et de bonne qualité.

Un early stopping *EarlyStopping()* a été mis en place afin de stopper l'entraînement en cas de non amélioration des performances sur l'ensemble de validation. Une réduction du taux d'apprentissage *ReduceLROnPlateau* a été appliquée lorsque la perte de validation cessait de diminuer.

Le modèle a été entraîné sur 60 époques avec un batch size de 32. L'évaluation a été réalisée en calculant l'accuracy sur l'ensemble de validation à chaque époque, permettant ainsi de suivre l'évolution des performances du modèle au fil du temps.

2.4 Architecture

Notre architecture de réseau de neurones à convolution (CNN), conçue pour distinguer les six espèces marines, est composée de 5 blocs de couches. Chaque couche renferme les couches suivantes : **Input**, **Convolution (Conv)**, **Batch Normalization**, **ReLU**, **Pooling**. La **Fully Connected (FC)**, quant à elle, évolue avec l'apprentissage. Afin de prévenir le surapprentissage (*overfitting*), nous avons appliqué un **dropout** après la troisième et la quatrième couche.

La **couche d'entrée** traite les images sous forme de matrices tridimensionnelles (3D), représentant les trois canaux de couleur : rouge, vert et bleu (RGB).

Les **couches de convolution** exploitent cette matrice d'entrée à l'aide de **filtres de petite taille** pour extraire les caractéristiques les plus importantes, en produisant une matrice de dimensions réduites. Nous avons utilisé successivement **32, 64, 128, 256 et 512 filtres**, avec une taille de noyau (**kernel size**) de **3x3**. Le **padding** a été défini sur "same" afin de conserver les dimensions d'entrée après chaque convolution.

La **normalisation par lot** (*Batch Normalization*) permet de stabiliser et d'accélérer l'apprentissage du modèle en normalisant les activations de chaque couche.

La **fonction d'activation ReLU** (Rectified Linear Unit) a été utilisée pour introduire de la non-linéarité. Elle permet au réseau d'apprendre des motifs complexes en remplaçant toutes les valeurs négatives par zéro.

Les **couches de pooling** réalisent un échantillonnage spatial (souvent appelé "sous-échantillonnage") en conservant les caractéristiques les plus significatives. Nous avons utilisé une **fenêtre de pooling de 2x2**.

La **couche Fully Connected** est constituée d'un perceptron multicouche. Tous les neurones en sortie sont connectés à tous les neurones en entrée. Nous avons utilisé **Global Average Pooling** afin de réduire le nombre de paramètres du modèle par rapport à une approche avec **Flatten**, ce qui le rend plus léger. Cette technique aide également à limiter le risque de surapprentissage en simplifiant la structure du modèle.

De plus, dans cette couche, la couche dense contient **32 000 neurones**, suivie d'un **Batch Normalization**, d'un **Dropout**, et d'une **fonction d'activation ReLU**. En sortie, une **seconde couche dense** composée de **6 neurones** correspond aux **6 espèces à classer**, avec une **fonction d'activation Softmax**, appropriée pour les tâches de classification multi-classes.

Une illustration graphique accompagne cette description pour représenter l'architecture complète de notre CNN.

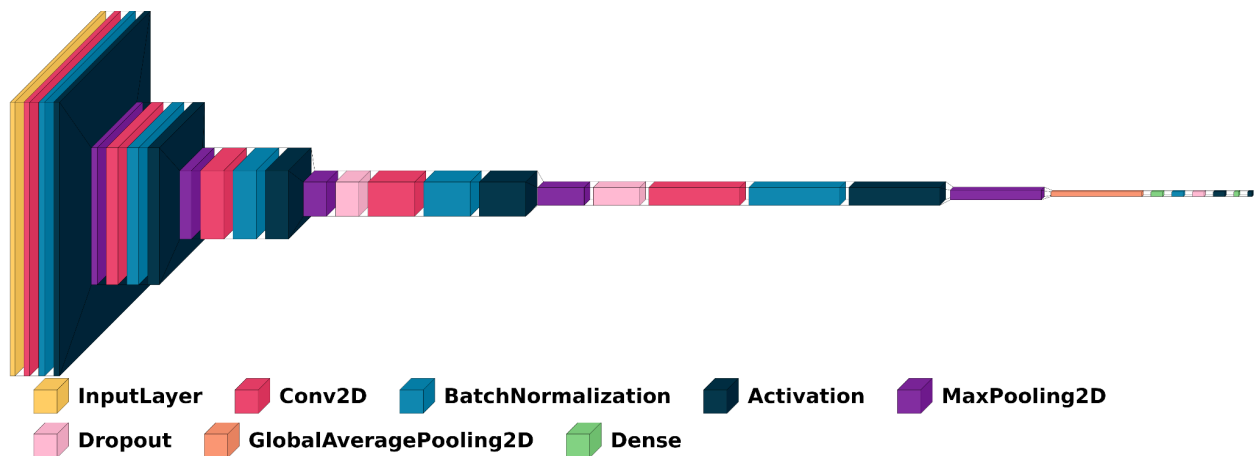
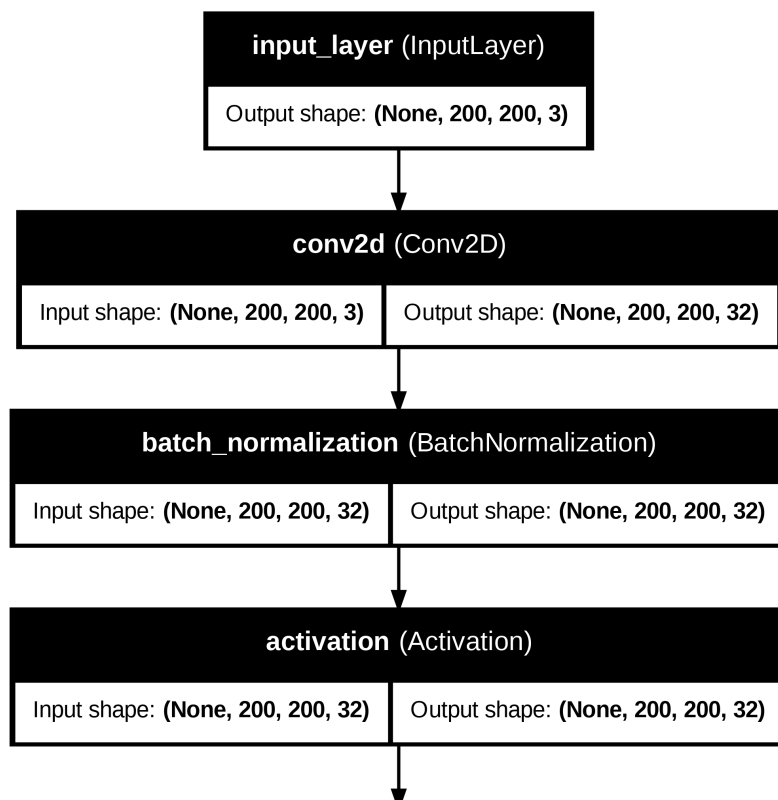
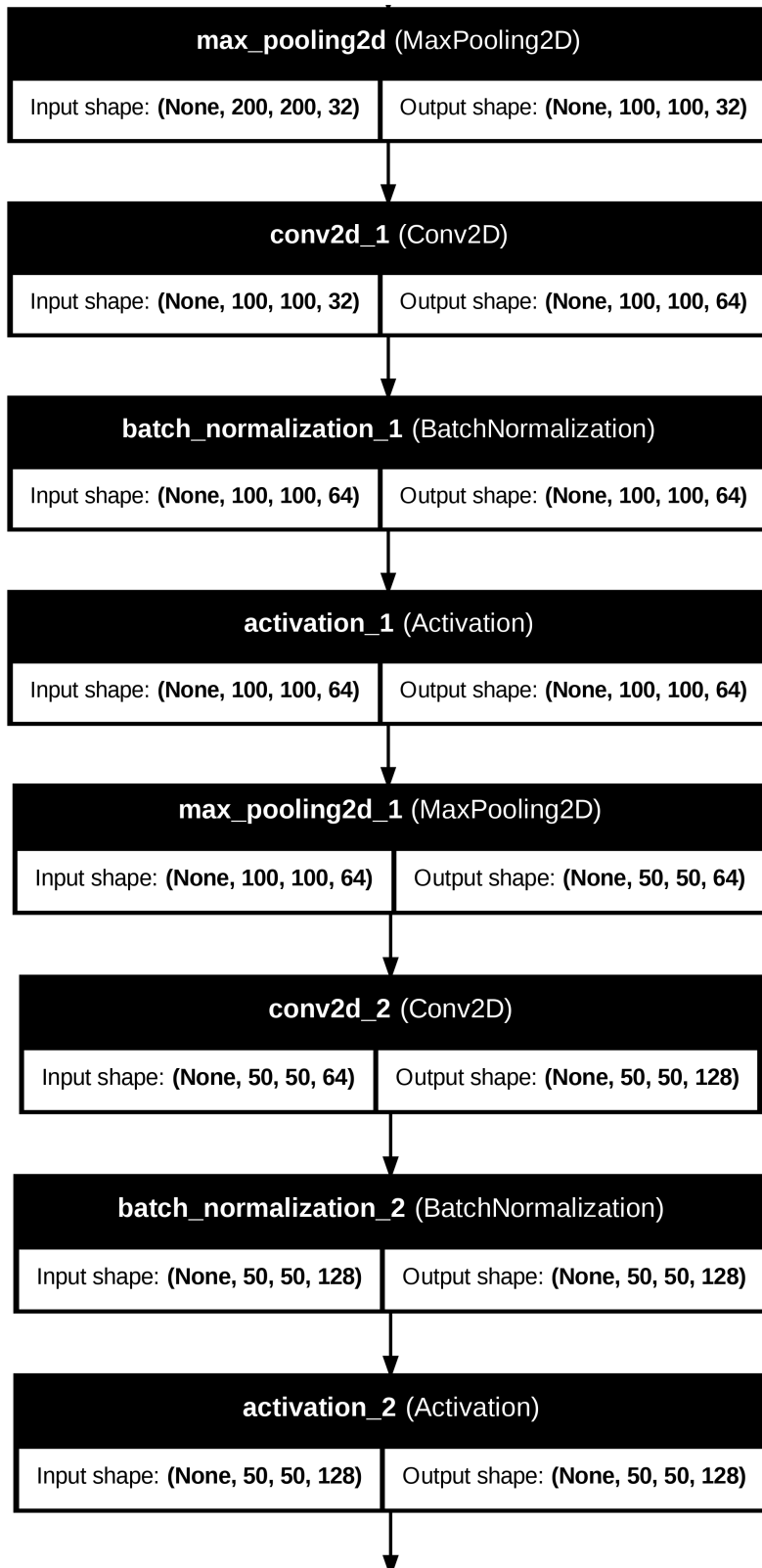
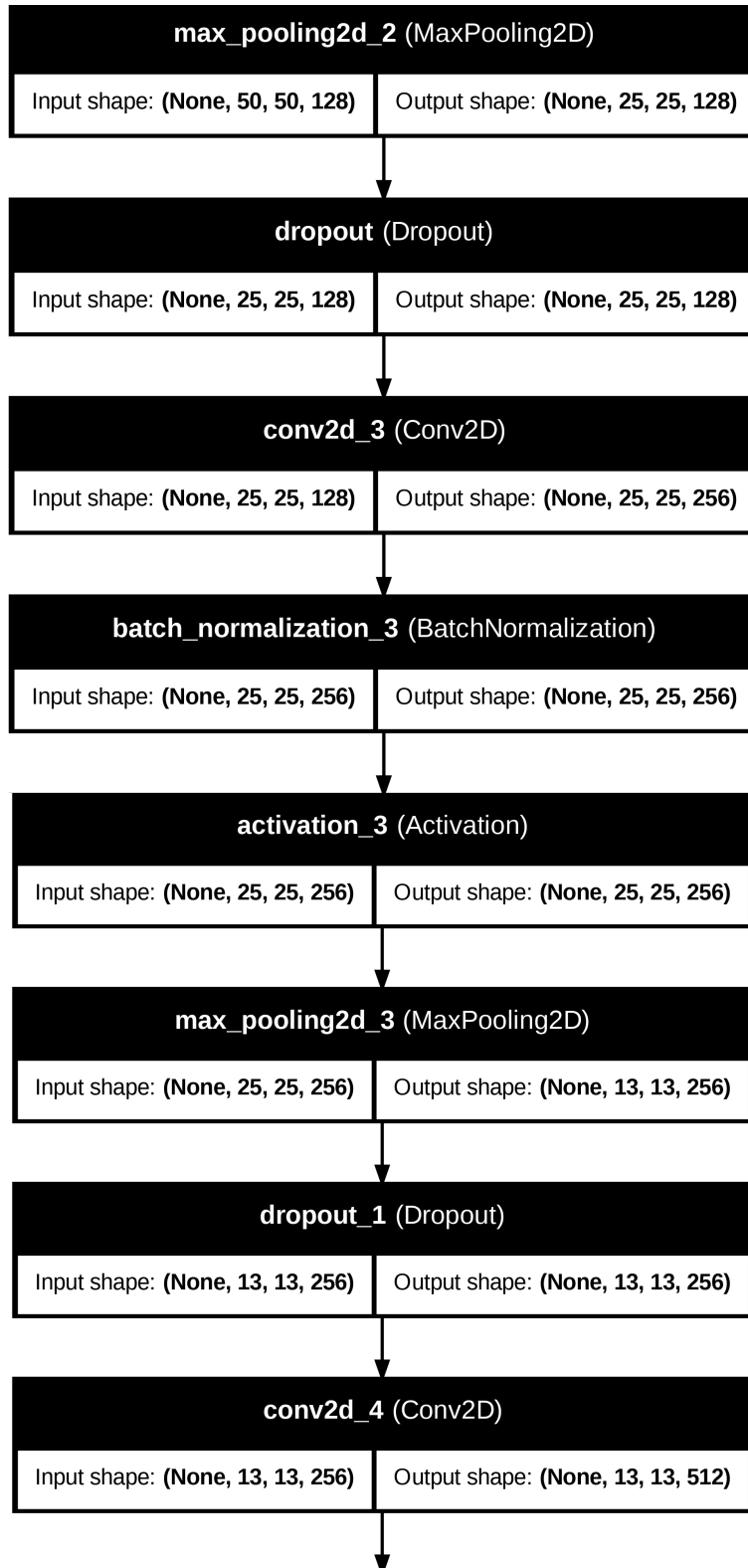


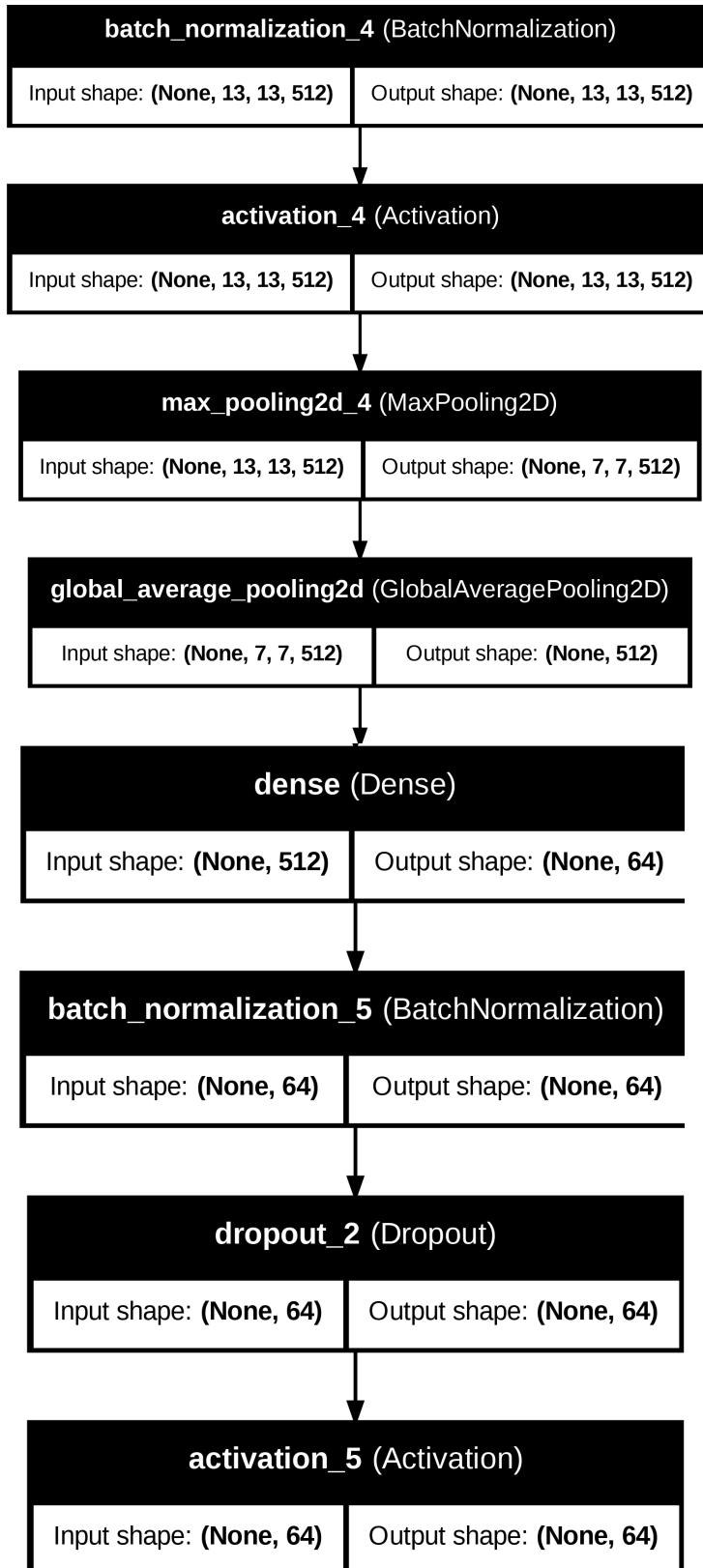
Figure 1: Visualisation 3D de l'architecture du réseau de neurons à convolution (CNN)

Une illustration graphique détaillée avec les informations de chaque couche









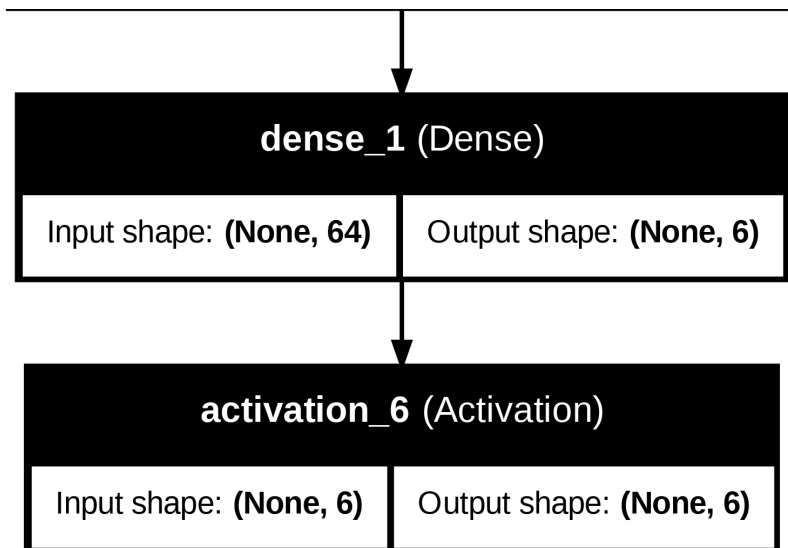


Figure 2 : Visualisation plus détaillée de l'architecture du réseau de neurones à convolution.

2.5 Affichage des résultats d'entraînement

Le temps total d'entraînement est de **570,26 secondes** (soit environ **9,5 minutes**). L'**erreur minimale** observée lors de l'entraînement est de **19,18 %**, tandis que l'**exactitude maximale** atteinte est de **85,67 %**.

Cette illustration montre l'évolution de l'exactitude sur l'entraînement et la validation.

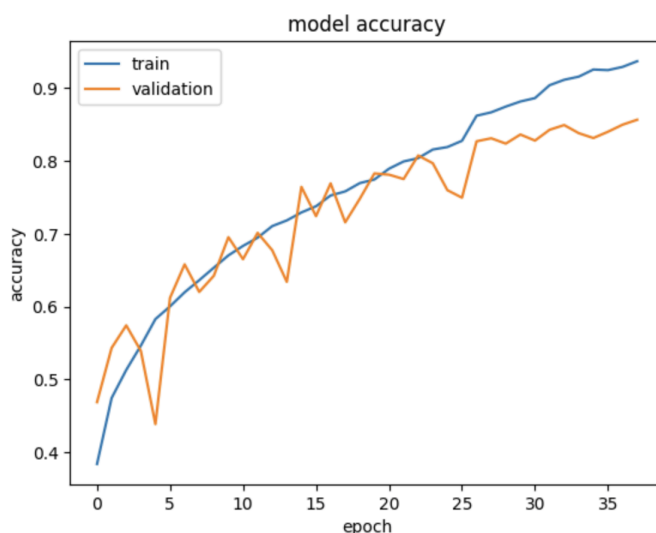


Figure 3 : Courbe d'exactitude (accuracy) par époque.

Remarques : Cette courbe montre que le taux de l'entraînement augmente régulièrement jusqu'à **90%**. La courbe qui représente la validation est instable, mais se stabilise autour de **85%**. Ce qui montre une bonne performance générale du modèle.

Cette figure illustre la diminution de l'erreur au fil des époques.

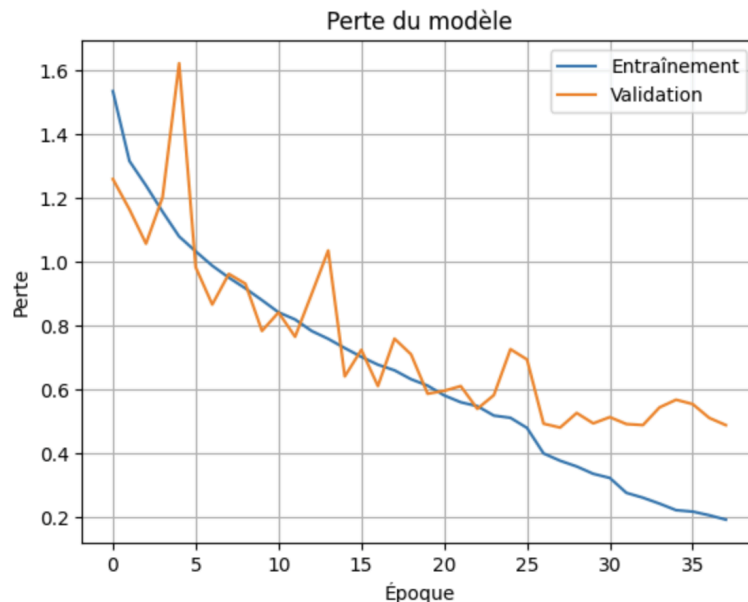


Figure 4 : Courbe de perte (loss) par époque.

Remarques: Cette courbe indique une diminution progressive du taux de perte. Ce qui indique un bon apprentissage du modèle.

2.6 Justification du choix de l'architecture

Le réseau de neurones à convolution (CNN) a été initialement conçu pour la reconnaissance d'images. Cette architecture a été mise en place dans le but de construire un modèle à la fois robuste et stable. Nous avons pris en compte plusieurs éléments clés, tel que le nombre de blocs de couches, le taux de dropout, pour que le modèle puisse bien apprendre sans être trop collé aux données. Nous avons également tenu compte du nombre de paramètres à implémenter, au choix du nombre de neurones, ainsi qu'aux fonctions d'activation. L'ensemble de ces éléments ont permis de concevoir une architecture capable de produire les meilleurs résultats.

3. Évaluation du Modèle

Pour évaluer le modèle développé nous avons utilisé les données de test (ensemble de 6000 images dont 1000 images pour chaque classe). Nous avons chargé les images de test, puis utilisé le modèle entraîné pour prédire les classes, ce qui nous a permis de calculer les métriques de performance comme la précision et l'exactitude et de générer les visualisations d'évaluation.

L'exactitude (accuracy) obtenue est de **83,72%** sur les données de test et la précision est de **86,83%**

Cette figure ci-dessous présente les performances du modèle en visualisant les prédictions correctes et incorrectes pour chaque classe.

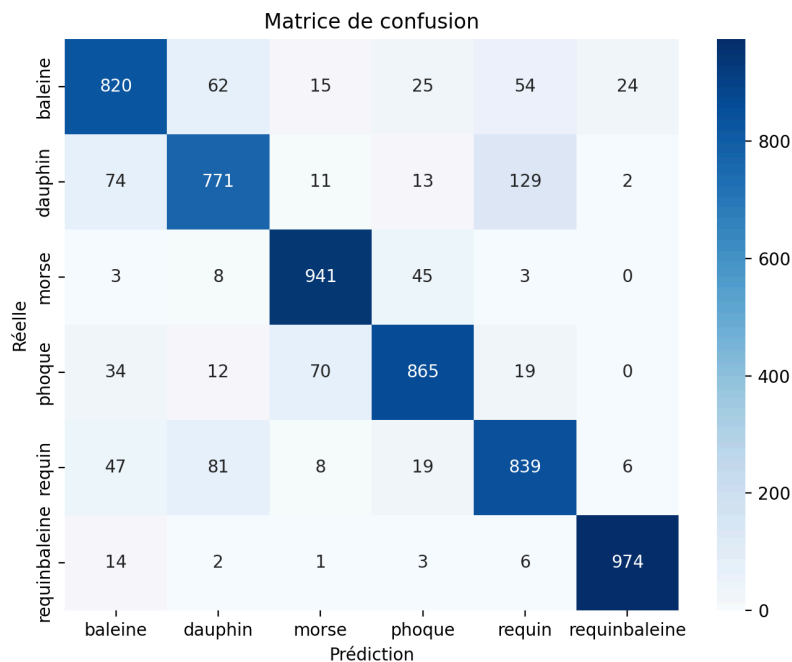


Figure 5 : Matrice de confusion.

Remarques : La matrice de confusion montre que la majorité des images sont bien classées soit 5210 images bien classées et 790 mal classées. La classe “requinbaleine” a eu le plus d’images bien classées soit **974 sur 1000** et la classe dauphin le moins d’images bien classées soit **771 sur 1000**. Cela peut être dû à la similarité des images et aux bruits.

Cette figure ci-dessous montre des images test avec leurs prédictions, selon le même format d’affichage utilisé précédemment.


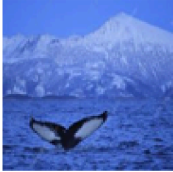



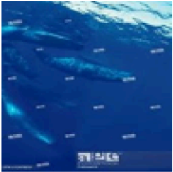




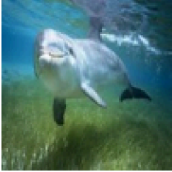




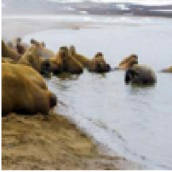
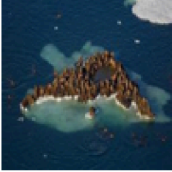
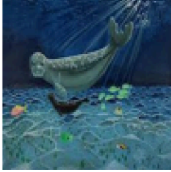

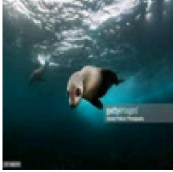

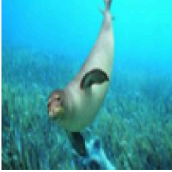
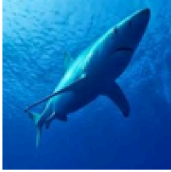
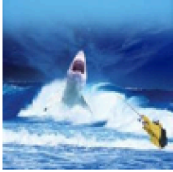










Images mal classées : Réel (lignes) vs Prédit (colonnes)						
	baleine	dauphin	morse	phoque	requin	requinbaleine
baleine						
dauphin						
morse						Aucune
phoque						Aucune
requin						
requinbaleine						

Figure 6 : Exemples d'images classées.

Remarques : Parmi les 790 images mal classées, nous avons prélevé 36 d'entre elles représentées ci dessus. La colonne représente les classes prédites par le modèle et la ligne, les classes réelles. Nous constatons que sur la plupart de ces images l'espèce marine n'est pas clairement visible (seule la queue apparaît ou l'image est floue). On note également d'autres éléments présents qui ne font pas partie des caractéristiques descripteurs de l'espèce.

Le modèle généralise bien mais reste sensible à la qualité des images.

L'exactitude globale dépasse le minimum attendu de 82%, ce qui indique que le modèle est performant sur les données de test.

4. Conclusion

Ce travail a permis d'implémenter un modèle CNN performant pour la classification d'images de six espèces marines. Nous avons rencontré des problèmes de performance avec l'utilisation initiale de seulement deux couches convolutionnelles, ce qui a nécessité l'ajout de deux couches supplémentaires afin d'améliorer l'extraction des caractéristiques. La couche flatten, a généré un très grand nombre de paramètres, et donc nous l'avons remplacée par une couche de **global average pooling** afin de réduire la complexité du modèle tout en conservant ses performances de généralisation. L'architecture finale, bien que présentant quelques signes de surapprentissage, offre de bonnes performances de classification, et peut être améliorée par l'optimisation des hyperparamètres, l'exploration d'autres architectures CNN et l'intégration de techniques de régularisation avancées.