



**HAL**  
open science

# Apprentissage actif profond pour la reconnaissance visuelle à partir de peu d'exemples

Sébastien Deschamps

► **To cite this version:**

Sébastien Deschamps. Apprentissage actif profond pour la reconnaissance visuelle à partir de peu d'exemples. Intelligence artificielle [cs.AI]. Sorbonne Université, 2023. Français. NNT : 2023SORUS199 . tel-04205380

**HAL Id: tel-04205380**

**<https://theses.hal.science/tel-04205380>**

Submitted on 12 Sep 2023

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

**THÈSE DE DOCTORAT DE SORBONNE UNIVERSITÉ**

Spécialité **Informatique**

École Doctorale Informatique, Télécommunications et Électronique (Paris)

**Deep Active Learning for Visual Recognition  
with Few Examples**

**Apprentissage actif profond pour la reconnaissance visuelle  
à partir de peu d'exemples**

Présentée par

**Sébastien DESCHAMPS**

Dirigée par

**Hichem SAHBI**

Pour obtenir le grade de

**DOCTEUR de SORBONNE UNIVERSITÉ**

Présentée et soutenue publiquement en Juin 2023

Devant le jury composé de :

**Mme Anissa MOKRAOUI**

*Professeure, Université Sorbonne Paris Nord*

Présidente du jury

**M. Michel CRUCIANU**

*Professeur, CNAM Paris*

Rapporteur

**M. Chaabane DJERABA**

*Professeur, Université de Lille*

Rapporteur

**M. Clément MALLET**

*Professeur/chercheur, Université Gustave Eiffel & IGN*

Examineur

**M. Andrei STOIAN**

*Docteur, Zama AI (précédemment chez Thales)*

Co-encadrant de thèse

**M. Hichem SAHBI**

*Chercheur CNRS & HDR, Sorbonne Université - LIP6*

Directeur de thèse



## RÉSUMÉ

L'analyse automatique d'images a permis d'améliorer l'exploitation des capteurs d'image, avec des données qui proviennent de différents capteurs tels que des caméras de téléphone, des caméras de surveillance, des imageurs satellites ou encore des drones. L'apprentissage profond obtient d'excellents résultats dans les applications d'analyse d'images où de grandes quantités de données annotées sont disponibles, mais apprendre un nouveau classifieur d'images à partir de zéro est une tâche difficile. La plupart des méthodes de classification d'images sont supervisées, nécessitant des annotations, ce qui représente un investissement important. Différentes solutions d'apprentissage frugal (avec peu d'exemples annotés) existent, notamment l'apprentissage par transfert, l'apprentissage actif, l'apprentissage semi-supervisé ou bien le méta-apprentissage.

L'objectif de cette thèse est d'étudier ces solutions d'apprentissage frugal pour des tâches de reconnaissance visuelle, notamment la classification d'images et la détection des changements dans des images satellites. Ainsi, le classifieur est entraîné de façon itérative en commençant avec très peu de données, et en demandant à l'utilisateur d'annoter le moins possible de données pour obtenir des performances satisfaisantes. L'apprentissage actif profond a été étudié initialement avec d'autres méthodes et nous a semblé le plus adapté à notre problématique métier, nous avons donc privilégié cette solution.

Nous avons développé dans cette thèse une première approche interactive, où nous posons les questions les plus informatives sur la pertinence des données à un oracle (annotateur). En fonction de ses réponses, une fonction de décision est mise à jour itérativement. Nous modélisons la probabilité que les échantillons soient pertinents, en minimisant une fonction objectif capturant la représentativité, la diversité et l'ambiguïté des données. Les données avec une probabilité élevée sont ensuite sélectionnées pour annotation. Nous avons fait évoluer cette approche, en utilisant l'apprentissage par renforcement pour pondérer dynamiquement et précisément l'importance de la représentativité, l'ambiguïté et la diversité des données à chaque cycle d'apprentissage actif. Finalement, notre dernière approche consiste en un modèle d'affichage qui sélectionne des exemples virtuels les plus représentatifs et divers, qui remettent en question le modèle appris, de sorte à obtenir un modèle très discriminatoire dans les itérations suivantes de l'apprentissage actif. Les bons résultats obtenus face aux différentes baselines et l'état de l'art, en détection de changements dans des images satellites et en classification d'images, ont permis de démontrer la pertinence des modèles d'apprentissage frugal proposés,



et ont donné lieu à diverses publications (SAHBI et al. 2021; DESCHAMPS et SAHBI 2022b; DESCHAMPS et SAHBI 2022a; SAHBI et DESCHAMPS 2022).

## ABSTRACT

Automatic image analysis has improved the exploitation of image sensors, with data coming from different sensors such as phone cameras, surveillance cameras, satellite imagers or even drones. Deep learning achieves excellent results in image analysis applications where large amounts of annotated data are available, but learning a new image classifier from scratch is a difficult task. Most image classification methods are supervised, requiring annotations, which is a significant investment. Different frugal learning solutions (with few annotated examples) exist, including transfer learning, active learning, semi-supervised learning or meta-learning.

The goal of this thesis is to study these frugal learning solutions for visual recognition tasks, namely image classification and change detection in satellite images. The classifier is trained iteratively by starting with only a few annotated samples, and asking the user to annotate as little data as possible to obtain satisfactory performance. Deep active learning was initially studied with other methods and suited our operational problem the most, so we chose this solution.

In this thesis, we have developed an interactive approach, where we ask the most informative questions about the relevance of the data to an oracle (annotator). Based on its answers, a decision function is iteratively updated. We model the probability that the samples are relevant, by minimizing an objective function capturing the representativeness, diversity and ambiguity of the data. Data with high probability are then selected for annotation. We have improved this approach, using reinforcement learning to dynamically and accurately weight the importance of representativeness, diversity and ambiguity of the data in each active learning cycle. Finally, our last approach consists of a display model that selects the most representative and diverse virtual examples, which adversely challenge the learned model, in order to obtain a highly discriminative model in subsequent iterations of active learning. The good results obtained against the different baselines and the state of the art in the tasks of satellite image change detection and image classification have demonstrated the relevance of the proposed frugal learning models, and have led to various publications (Sahbi et al. 2021; Deschamps and Sahbi 2022b; Deschamps and Sahbi 2022a; Sahbi and Deschamps 2022).



## REMERCIEMENTS

*“Celui qui n’a pas d’objectifs ne risque pas de les atteindre.” Sun Tzu*

Ce manuscrit de thèse est l’aboutissement de plusieurs années de travail, et je souhaiterais remercier toutes les personnes qui ont pu aider à sa réalisation. Pour commencer, je remercie chaleureusement mon directeur de thèse **Hichem Sahbi** pour son éclairage scientifique et sa disponibilité tout au long de cette thèse.

Je souhaite également remercier **Michel Crucianu** et **Chaabane Djeraba** pour le temps qu’ils ont consacré sur ce manuscrit en tant que rapporteurs, ainsi que **Anissa Mokraoui**, **Clément Mallet** et **Andrei Stoian** pour leur participation à mon jury de soutenance.

Je remercie aussi mes collègues du laboratoire Theresis de Thales, et notamment **Andrei Stoian** ainsi que **Jean-Emmanuel Haugeard** pour l’encadrement industriel de la thèse, et les doctorants de l’équipe MLIA du LIP6 pour le bon environnement de travail.

Enfin, je remercie ma compagne **Delphine** pour son soutien inconditionnel lors de ces années de doctorat, ma mère **Claude** et mon frère **Nahuel** ainsi que mes amis, pour leur support au quotidien.

Sébastien



# TABLE DES MATIÈRES

|   |      |
|---|------|
| RÉSUMÉ  | i    |
| ABSTRACT  | iii  |
| REMERCIEMENTS   | v    |
| TABLE DES MATIÈRES  | vii  |
| TABLE DES FIGURES   | ix   |
| Liste des tableaux  | xi   |
| ACRONYMES   | xiii |
| 1 INTRODUCTION  | 1    |
| 1.1 Définition du problème et motivation . . . . .                      | 1    |
| 1.2 Contexte du problème . . . . .                                      | 2    |
| 1.3 Aperçu de l'apprentissage actif . . . . .                           | 7    |
| 1.4 Démarche, contributions et leur positionnement . . . . .            | 9    |
| 1.5 Organisation du manuscrit . . . . .                                 | 12   |
| 2 RÉSEAUX DE NEURONES PROFONDS  | 13   |
| 2.1 Introduction aux réseaux de neurones . . . . .                      | 14   |
| 2.2 Les réseaux de neurones profonds . . . . .                          | 17   |
| 2.3 Les réseaux de neurones convolutifs . . . . .                       | 20   |
| 2.4 Transformeurs pour la vision (ViT) . . . . .                        | 41   |
| 3 APPRENTISSAGE FRUGAL  | 45   |
| 3.1 Apprentissage automatique à partir de peu de données . . . . .      | 46   |
| 3.2 L'apprentissage actif profond pour l'image . . . . .                | 59   |
| 3.3 Positionnement des travaux et contributions . . . . .               | 75   |
| 3.4 Bases de données et métriques d'évaluation . . . . .                | 76   |
| 4 APPRENTISSAGE PROFOND ACTIF POUR LA RECONNAISSANCE VI-<br>SUELLE      | 81   |
| 4.1 Introduction . . . . .  | 82   |
| 4.2 Méthode proposée . . . . .  | 83   |
| 4.3 Expériences en classification d'images . . . . .                    | 89   |
| 4.4 Expériences en détection de changements . . . . .                   | 103  |
| 4.5 Conclusion . . . . .  | 107  |
| 5 SÉLECTION D'AFFICHAGE POUR L'APPRENTISSAGE FRUGAL PAR<br>RENFORCEMENT | 109  |
| 5.1 Introduction . . . . .  | 110  |
| 5.2 Méthode proposée . . . . .  | 111  |
| 5.3 Expériences . . . . .   | 117  |
| 5.4 Conclusion . . . . .  | 128  |

|     |   |     |
|-----|---|-----|
| 6   | APPRENTISSAGE PROFOND ACTIF POUR L'IMAGE À L'AIDE D'EXEMPLES VIRTUELS | 131 |
| 6.1 | Introduction  | 132 |
| 6.2 | Méthode proposée  | 133 |
| 6.3 | Expériences de détections des changements dans des images satellites  | 138 |
| 6.4 | Expériences de classification d'images                                | 140 |
| 6.5 | Conclusion  | 143 |
| 7   | CONCLUSION  | 145 |
| 7.1 | Synthèse  | 145 |
| 7.2 | Perspectives  | 146 |
|     | BIBLIOGRAPHIE   | 149 |

## TABLE DES FIGURES

|   |           |
|---|-----------|
| <b>CHAPITRE 1 : INTRODUCTION</b>  | <b>1</b>  |
| Figure 1.1 Schéma de l'apprentissage actif . . . . .                        | 5         |
| <b>CHAPITRE 2 : RÉSEAUX DE NEURONES PROFONDS</b>                            | <b>14</b> |
| Figure 2.1 Le perceptron . . . . .  | 15        |
| Figure 2.2 Le problème du ou-exclusif . . . . .                             | 16        |
| Figure 2.3 Le perceptron multi-couches. . . . .                             | 16        |
| Figure 2.4 Différentes fonctions d'activation non-linéaires . . . . .       | 17        |
| Figure 2.5 Diagramme d'un Deep Neural Network (DNN) . . . . .               | 19        |
| Figure 2.6 Représentation graphique de la fonction softmax . . . . .        | 20        |
| Figure 2.7 Une architecture classique d'un CNN . . . . .                    | 21        |
| Figure 2.8 Différents noyaux appliqués à une image . . . . .                | 22        |
| Figure 2.9 Illustration d'une couche de convolution . . . . .               | 23        |
| Figure 2.10 Algorithme de normalisation par lot . . . . .                   | 25        |
| Figure 2.11 Illustration de la régularisation par dropout . . . . .         | 26        |
| Figure 2.12 Transformation CenterCrop . . . . .                             | 28        |
| Figure 2.13 Transformation Grayscale . . . . .                              | 28        |
| Figure 2.14 Transformation ColorJitter . . . . .                            | 28        |
| Figure 2.15 Transformation RandomRotation . . . . .                         | 28        |
| Figure 2.16 Transformation RandomCrop . . . . .                             | 29        |
| Figure 2.17 Transformation RandomHorizontalFlip . . . . .                   | 29        |
| Figure 2.18 Transformation RandomVerticalFlip . . . . .                     | 29        |
| Figure 2.19 Descente de gradient illustrée . . . . .                        | 33        |
| Figure 2.20 Architecture du CNN LeNet-5 . . . . .                           | 35        |
| Figure 2.21 Architecture du CNN AlexNet . . . . .                           | 38        |
| Figure 2.22 Architecture du CNN VGG16 . . . . .                             | 39        |
| Figure 2.23 Un bloc de Residual Network du papier original . . . . .        | 41        |
| Figure 2.24 Architecture originale du transformeur . . . . .                | 43        |
| Figure 2.25 Architecture de Vision Transformer (ViT) . . . . .              | 44        |
| <b>CHAPITRE 3 : APPRENTISSAGE FRUGAL</b>                                    | <b>46</b> |
| Figure 3.1 Illustration de l'apprentissage few-shot . . . . .               | 47        |
| Figure 3.2 Les trois scénarios d'apprentissage actif . . . . .              | 50        |
| Figure 3.3 Carte de chaleur des différentes mesures d'incertitude . . . . . | 56        |
| Figure 3.4 Exemple entropie de vote . . . . .                               | 57        |
| Figure 3.5 Diagramme de l'apprentissage actif profond . . . . .             | 60        |



|   |   |            |
|---|---|------------|
| Figure 3.6  | Illustration de l'intuition derrière GAAL . . . . .                                   | 65         |
| Figure 3.7  | Algorithme de (TRAN et al. 2019) . . . . .  | 67         |
| Figure 3.8  | Le jeu de données de chiffres manuscrits MNIST . . . . .                              | 76         |
| Figure 3.9  | Le jeu de données multi-classes CIFAR-10 . . . . .                                    | 77         |
| Figure 3.10   | Le jeu de données DOTA avant son adaptation en Object-DOTA . . . . .                  | 78         |
| Figure 3.11   | Paire d'images de Jefferson . . . . .   | 79         |
| <b>CHAPITRE 4 : APPRENTISSAGE PROFOND ACTIF POUR LA RECONNAISSANCE VISUELLE</b>           |   | <b>82</b>  |
| Figure 4.1  | Illustration du partitionnement multiple des données . . .                            | 88         |
| Figure 4.2  | Vue d'ensemble des architectures de CNNs populaires . .                               | 96         |
| Figure 4.3  | Imagettes sélectionnées par notre algorithme ou l'aléatoire sur CIFAR-10 . . . . .    | 103        |
| Figure 4.4  | Imagettes sélectionnées par notre algorithme ou l'aléatoire sur Object-DOTA . . . . . | 104        |
| Figure 4.5  | Test de référence détection de changements . . . . .                                  | 106        |
| <b>CHAPITRE 5 : SÉLECTION D'AFFICHAGE POUR L'APPRENTISSAGE FRUGAL PAR RENFORCEMENT</b>    |   | <b>110</b> |
| Figure 5.1  | Dynamique des hyper-paramètres RL . . . . .   | 120        |
| Figure 5.2  | Test de référence RL sur Jefferson . . . . .  | 121        |
| Figure 5.3  | Test de référence RL sur Object DOTA . . . . .  | 122        |
| Figure 5.4  | Exemples d'images annotées sur Object-DOTA . . . . .                                  | 130        |
| <b>CHAPITRE 6 : APPRENTISSAGE PROFOND ACTIF POUR L'IMAGE À L'AIDE D'EXEMPLES VIRTUELS</b> |   | <b>132</b> |
| Figure 6.1  | Jefferson comparaison de notre modèle virtuel avec les baselines . . . . .            | 139        |

## LISTE DES TABLEAUX

|   |     |
|---|-----|
| CHAPITRE 2 : RÉSEAUX DE NEURONES PROFONDS   | 14  |
| CHAPITRE 3 : APPRENTISSAGE FRUGAL   | 46  |
| Table 3.1 Exemple incertitude par moindre confiance . . . . .   | 54  |
| Table 3.2 Exemple incertitude par entropie . . . . .  | 54  |
| Table 3.3 Exemple incertitude échantillonnage avec marges . . . . .   | 55  |
| CHAPITRE 4 : APPRENTISSAGE PROFOND ACTIF POUR LA RECONNAISSANCE VISUELLE  | 82  |
| Table 4.1 CIFAR-10, comparaison des distances avec faible régime d'annotation . . . . .                               | 92  |
| Table 4.2 CIFAR-10, comparaison des distances avec la régularisation entropique (haut régime d'annotation) . . . . .  | 92  |
| Table 4.3 CIFAR-10, comparaison des distances avec la régularisation quadratique (haut régime d'annotation) . . . . . | 93  |
| Table 4.4 CIFAR-10 comparaison avec les baselines (ShuffleNet v2) .   | 93  |
| Table 4.5 CIFAR-10 comparaison avec les baselines (MobileNet-v2) .  | 94  |
| Table 4.6 CIFAR-10 comparaison avec les baselines (ResNet-18) . . .   | 95  |
| Table 4.7 CIFAR-100 comparaison avec les baselines (ResNet-18) . .  | 95  |
| Table 4.8 Object-DOTA comparaison avec les baselines . . . . .  | 96  |
| Table 4.9 MNIST, étude d'ablation 100 à 1000 échantillons . . . . .   | 97  |
| Table 4.10 CIFAR-10, étude d'ablation 2000 à 10000 échantillons . . .   | 98  |
| Table 4.11 CIFAR-10, étude d'ablation 100 à 1000 échantillons . . . . .   | 98  |
| Table 4.12 CIFAR-10, nombre de "victoires" des critères . . . . .   | 99  |
| Table 4.13 Object-DOTA, étude d'ablation 100 à 1000 échantillons . .  | 100 |
| Table 4.14 Comparaison avec l'état de l'art sur MNIST . . . . .   | 100 |
| Table 4.15 Comparaison avec l'état de l'art sur CIFAR-10 . . . . .  | 101 |
| Table 4.16 Comparaison avec l'état de l'art sur CIFAR-100 . . . . .   | 102 |
| Table 4.17 Etude d'ablation en détection de changements . . . . .   | 106 |
| Table 4.18 Comparaison mono et multi-partitionnement en détection de changements . . . . .                            | 107 |
| CHAPITRE 5 : SÉLECTION D'AFFICHAGE POUR L'APPRENTISSAGE FRUGAL PAR RENFORCEMENT                                       | 110 |
| Table 5.1 Object-DOTA étude d'ablation AL+RL . . . . .  | 119 |
| Table 5.2 Jefferson étude d'ablation AL+RL . . . . .  | 119 |

|           |   |     |
|-----------|---|-----|
| Table 5.3 | Object-DOTA sans préentraînement du réseau . . . . .      | 124 |
| Table 5.4 | Object-DOTA avec préentraînement du réseau . . . . .      | 125 |
| Table 5.5 | Object-DOTA comparaison avec ou sans préentraînement .    | 125 |
| Table 5.6 | Object-DOTA expérience de rééquilibrage des classes . . . | 126 |
| Table 5.7 | CIFAR-10 comparaison de RL-C avec différentes récompenses | 126 |
| Table 5.8 | CIFAR-10 tests additionnels AL Flat . . . . .             | 127 |
| Table 5.9 | CIFAR-10 tests additionnels AL+RL bis . . . . .           | 128 |

**CHAPITRE 6 : APPRENTISSAGE PROFOND ACTIF POUR L'IMAGE  
À L'AIDE D'EXEMPLES VIRTUELS** 132

|           |   |     |
|-----------|---|-----|
| Table 6.1 | Jefferson étude d'ablation modèle d'exemples virtuels . . .                       | 138 |
| Table 6.2 | Object-DOTA. Comparaison des performances selon la taille<br>des images . . . . . | 141 |
| Table 6.3 | Paramètres pour la classification d'images . . . . .                              | 142 |
| Table 6.4 | Object-DOTA. Comparaison avec les baselines . . . . .                             | 142 |

## ACRONYMES

|        |   |
|--------|---|
| AL     | Active Learning                                   |
| cGAN   | Conditional Generative Adversarial Network        |
| GAN    | Generative Adversarial Network                    |
| GCN    | Graph Convolutional Network                       |
| DCGAN  | Deep Convolutional Generative Adversarial Network |
| DNN    | Deep Neural Network                               |
| CNN    | Convolutional Neural Network                      |
| DL     | Deep Learning                                     |
| RVB    | Rouge Vert Bleu                                   |
| SVM    | Support Vector Machine                            |
| SVMfr  | Machine à vecteurs de support                     |
| GPU    | Graphics Processing Unit                          |
| GPUfr  | Processeur graphique                              |
| ILSVRC | ImageNet Large Scale Visual Recognition Challenge |
| VRAM   | Video Random Access Memory                        |
| VRAMfr | Mémoire vive vidéo                                |
| LRN    | Local Response Normalization                      |
| LRNfr  | Normalisation par réponse locale                  |
| PCA    | Principal Component Analysis                      |
| PCAfr  | Analyse en composantes principales                |
| VGG    | Visual Geometry Group                             |
| ResNet | Residual Network                                  |
| SGD    | Stochastic Gradient Descent                       |
| SGDfr  | Descente de gradient stochastique                 |
| Adam   | Adaptive Moment Estimation                        |
| VAE    | Variational Auto-Encoder                          |
| BNN    | Bayesian Neural Network                           |
| ViT    | Vision Transformer                                |
| MLP    | Multi-Layer Perceptron                            |



## INTRODUCTION

### 1.1 Définition du problème et motivation

Motivée par l'augmentation du taux de production des données image, l'analyse automatique de celles-ci a permis de faire un grand pas en avant dans l'exploitation des capteurs d'image. Ces données proviennent d'une gamme éclectique de capteurs tels que des caméras de téléphone, des caméras de surveillance, des imageurs basés sur satellite et des drones avec caméras. Les tâches typiques d'analyse d'images incluent la classification de scènes, d'objets, d'actions, la détection et la localisation d'objets, le suivi, la détection de changements et la segmentation sémantique.

L'apprentissage profond (*Deep Learning*) a donné des résultats remarquables dans les applications d'analyse d'images où de grandes quantités de données annotées sont disponibles. Ceci est en partie dû à la capacité des réseaux de neurones convolutifs d'intégrer ces énormes quantités de données dans de puissants a priori qui modélisent efficacement la distribution des données. Cela est particulièrement vrai dans le cas des images naturelles, où des collections allant jusqu'à des dizaines de millions d'exemples sont disponibles (RUSSAKOVSKY et al. 2015, KRASIN et al. 2017). Ces a priori sont intrinsèquement intégrés aux extracteurs de caractéristiques appris, au-delà desquelles divers modèles peuvent être spécialisés, tels que la classification d'images (HE et al. 2016), la localisation d'objets à l'aide de la régression (Wei LIU et al. 2016, GIRSHICK 2015) et l'apprentissage métrique (YI et al. 2014).

Cependant, étant donné le nombre important d'images avec une grande variabilité intra-classe, apprendre un nouveau classifieur d'images à partir de zéro est toujours une tâche difficile. La plupart des méthodes de classification d'images existantes sont supervisées, c'est-à-dire qu'elles nécessitent des annotations. L'annotation d'un grand corpus de données représente un investissement important et la manière dont les images sont annotées est souvent liée au fonctionnement des méthodes d'apprentissage supervisé. Dans ce cas, le praticien doit avoir des bonnes connaissances en apprentissage pour faire des bons choix de données mais aussi pour en collecter en quantité et diversité suffisantes, afin de bien définir des

classes tout en anticipant les problèmes induits par des biais statistiques, pour nettoyer les données et pour choisir des modèles et des hyper-paramètres appropriés. D'autre part, l'opérateur ou l'utilisateur final d'un système de création de classification d'image aura besoin d'une assistance automatique pour surmonter ces problèmes, et, au minimum, d'un retour rapide sur ses annotations afin de pouvoir découvrir des solutions par le biais d'expérimentations.

Pour la classification, l'apprentissage avec peu d'exemples annotés a été étudié via l'apprentissage par transfert (YOSINSKI et al. 2014), l'apprentissage actif (SENER et SAVARESE 2017), l'apprentissage semi-supervisé (OLIVER et al. 2018) et le méta-apprentissage (SANTORO et al. 2016). Afin que les classifieurs apprennent progressivement, demander aux utilisateurs de fournir un retour sur la pertinence est une approche intéressante, mais qui présente ses propres difficultés, telles que l'oubli catastrophique (KIRKPATRICK et al. 2017) ou le fait de devoir conserver de nombreux échantillons précédents pour de nouvelles itérations d'entraînement.

Selon le jeu de données utilisé, la classification d'images peut connaître un déséquilibre de classes élevé et une gestion multi-résolution. Certains travaux obtiennent de bons résultats grâce à un apprentissage actif (RADOSAVOVIC et al. 2018, CONTARDO et al. 2017) et à un apprentissage auto-supervisé (K. WANG et al. 2018). Des résultats intéressants sont affichés lorsque le nombre d'exemples annotés est aussi bas que 10% du jeu de données d'origine. Toutefois, cela peut toujours être un nombre important car la plupart des jeux de données contiennent des milliers d'instances d'objets.

L'objectif de cette thèse est d'étudier des méthodes d'apprentissage incrémentiel de classifieurs d'images en commençant par très peu d'instances et en demandant à l'utilisateur d'annoter, de corriger ou de valider le moins possible d'annotations automatiques pour obtenir des performances satisfaisantes. L'apprentissage actif profond (*deep active learning*) ainsi que d'autres méthodes telles que l'apprentissage avec peu d'essais (*few-shot learning*) ou encore le méta-apprentissage (*meta-learning*) ont été considérées initialement. L'apprentissage actif profond nous a rapidement semblé plus adapté au problème opérationnel que nous cherchions à résoudre, nous avons donc privilégié cette solution.

## 1.2 Contexte du problème

Une grande quantité de données non étiquetées, notamment des images satellites contenant plusieurs objets d'intérêts, tels que des bateaux, avions ou piscines, est récupérée chaque jour. Ces données peuvent être stratégiques, notamment lors de catastrophes naturelles, ou dans un contexte de défense, il est donc impor-

tant de pouvoir entraîner un classifieur à reconnaître les objets d'intérêt de façon automatique. L'apprentissage profond est tout indiqué pour ce faire, néanmoins une grande quantité de données est insuffisante pour entraîner un modèle, il est nécessaire d'avoir des annotations de qualité pour chacune des images / objets. Ainsi, un annotateur humain (par exemple un ingénieur) va étiqueter les données images une à une à l'aide d'un logiciel dédié, ce qui prend beaucoup de temps et l'empêche de travailler sur des problèmes nécessitant ses compétences. De fait, il peut être intéressant de ne donner à cet annotateur que les images / objets les plus pertinents, c'est-à-dire les plus susceptibles d'influencer positivement la qualité de généralisation du modèle d'apprentissage profond entraîné sur ces données. C'est là qu'intervient l'apprentissage actif, dont le but est de réduire le coût d'annotation des données, tout en conservant une bonne qualité de généralisation du modèle entraîné à partir de ces données annotées. Cette annotation est interactive et se fait donc par l'intermédiaire d'un oracle qui peut donner l'étiquette correcte pour chaque donnée. Cela correspond au cas d'usage réel, où un utilisateur expert, par exemple un ingénieur, annoté les données images une à une à l'aide d'un logiciel dédié.

Il existe plusieurs variantes d'apprentissage actif, la version dite "en ligne" consiste à recevoir une image par une image, et l'algorithme d'apprentissage actif détermine si la donnée doit être étiquetée par l'oracle ou non. La version que nous avons choisie utilise un ensemble de données non étiquetées appelé *pool*, l'algorithme d'apprentissage actif va cette fois-ci déterminer quel paquet d'images de ce *pool* sélectionner pour être annotées par l'oracle. Ensuite, un classifieur est entraîné à partir de ces données étiquetées et sa performance est évaluée sur l'ensemble de données test, par exemple avec une métrique d'*accuracy*. Ce processus est répété plusieurs itérations, ou cycles, avec un nombre fixe de données sélectionnées à chaque cycle, et ce jusqu'à épuisement d'un budget défini à l'avance. Il existe également une variante, où le but n'est pas d'épuiser le budget d'annotations, mais d'atteindre une performance d'*accuracy* cible à partir du moins d'annotations possible.

### 1.2.1 Formulation du problème de l'apprentissage actif

Soit  $\mathcal{X}$  l'ensemble de toutes les images possibles tirées d'une distribution existante mais inconnue  $P(X, Y)$ . La variable aléatoire  $X$  se réfère à une image d'entrée et  $Y$  à son étiquette de classe inconnue. En considérant  $nc$  classes visuelles (c'est-à-dire des labels ou catégories), et  $\mathcal{U}$  un grand sous-ensemble de  $\mathcal{X}$  dont les labels sont inconnus initialement, notre objectif est de modéliser des classifieurs  $\{g_c\}_{c=1}^{nc}$  grâce à une annotation interactive d'une petite fraction de  $\mathcal{U}$ , et d'entraîner les paramètres de  $\{g_c\}_c$ .



Soit  $\mathcal{D}_t$  un *affichage* (défini comme sous-ensemble de  $\mathcal{U}$ ) montré à un oracle (un annotateur expert qui fournit des annotations pour n'importe quel sous-ensemble d'images) à n'importe quelle itération  $t$  de l'apprentissage actif, et soit  $\mathcal{Y}_t$  les étiquettes sous-jacentes. L'affichage initial  $\mathcal{D}_t$  (avec  $t = 0$ ) est aléatoirement échantillonné de façon uniforme, et utilisé afin d'entraîner les classifieurs suivants en répétant la boucle d'apprentissage actif suivante, jusqu'à atteindre un seuil de performances ou en épuisant un budget d'annotations prédéfini :

- Récupérer les étiquettes de  $\mathcal{D}_t$  avec  $\mathcal{Y}_t \leftarrow \text{oracle}(\mathcal{D}_t)$ , cette fonction oracle peut dépendre d'une vérité terrain connue seule de l'utilisateur.
- Entraîner  $\{g_{c,t}\}_c$  grâce à  $\bigcup_{\tau=1}^t (\mathcal{D}_\tau, \mathcal{Y}_\tau)$  où le deuxième indice dans  $g_{c,t}$  fait référence à la fonction de décision à l'itération  $t$ . Différents modèles d'apprentissage peuvent être considérés, notamment des réseaux profonds convolutifs.
- Sélectionner le prochain affichage  $\mathcal{D} \subset \mathcal{U} - \bigcup_{\tau=1}^t \mathcal{D}_\tau$  qui augmente possible-ment les performances de généralisation des classifieurs suivants,  $\{g_{c,t+1}\}_c$ . Comme les étiquettes de l'affichage  $\mathcal{D}$  sont inconnues et coûteuses, on ne peut pas essayer toutes les combinaisons des différents affichages  $\mathcal{D}$  possibles, entraîner les classifieurs associés, et choisir le meilleur affichage. C'est l'algorithme de sélection d'apprentissage actif qui va sélectionner l'affichage le plus optimal selon une fonction d'acquisition définie en amont : la contribution principale de nos travaux.

Cette procédure est présentée de façon informelle sur la [figure 1.1](#).

## 1.2.2 Diverses contraintes opérationnelles

Il existe diverses contraintes opérationnelles dans l'application réelle de l'apprentissage actif, pour en citer quelques-uns : il est possible que l'oracle soit imparfait, ou qu'il faille en utiliser plusieurs. Certaines données peuvent être plus coûteuses à annoter que d'autres, ce qui n'est pas pris en compte dans le budget lorsque l'on fait de l'apprentissage actif en recherche pure, etc. Voici un rapide tour d'horizon des différentes contraintes opérationnelles que l'on peut rencontrer en apprentissage actif dans l'industrie.

### 1.2.2.1 Comment choisir l'oracle

Il est tout à fait possible d'avoir plusieurs oracles à disposition, avec des compétences, expertise et motivation variables, par exemple avec Amazon Mechanical Turk. Une façon de modéliser ce problème est de choisir parmi deux oracles :

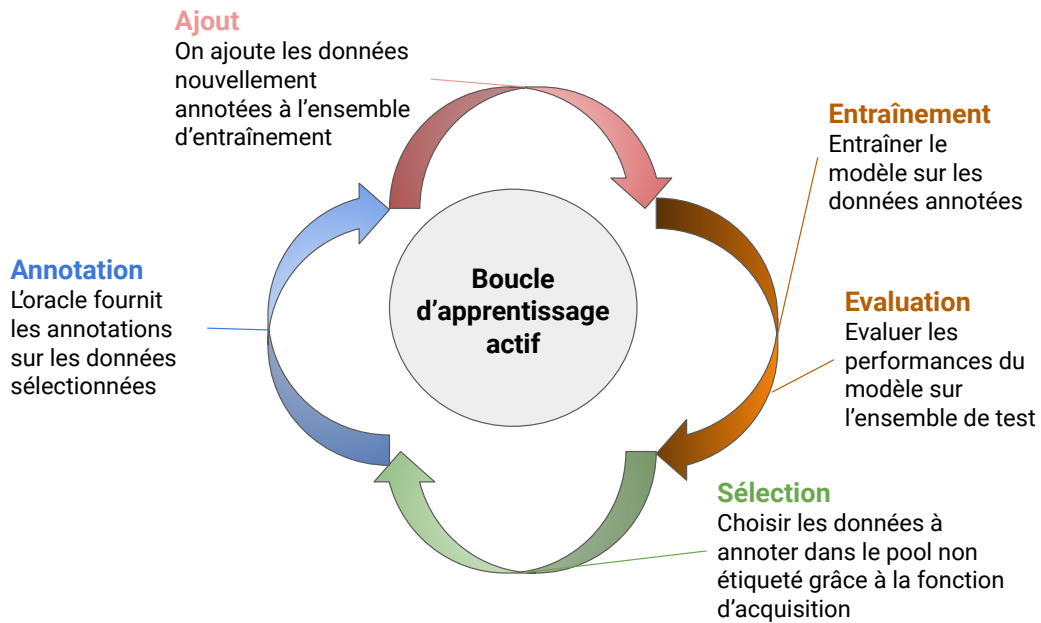


FIGURE 1.1 – **Schéma de l'apprentissage actif.** La première étape est celle de sélection, où l'algorithme choisit les données que l'oracle doit annoter à l'étape suivante. Ces données sont ajoutées à l'ensemble étiqueté qui sert ensuite pour l'étape d'entraînement du modèle, et enfin les performances sont évaluées sur l'ensemble de test.

un oracle expert qui produira des étiquettes fiables mais à un coût plus élevé (par exemple un ingénieur ou docteur du domaine), ou bien un oracle qui produit des étiquettes parfois fausses mais qui coûte moins cher (par exemple une personne lambda). Ce problème est notamment étudié par (Chicheng ZHANG et CHAUDHURI 2015).

### 1.2.2.2 Données parfois difficiles à annoter

En pratique, toutes les données ne sont pas aussi propices à être annotées : certaines données sont très ambiguës ou difficiles à étiqueter. Il est ainsi nécessaire de prendre en compte la probabilité que l'oracle puisse fournir une étiquette, au moment de choisir un oracle, comme le montre (YAN et al. 2016). Ce problème se pose notamment lorsque l'on utilise des images générées artificiellement pour faire de l'apprentissage actif, notamment des GANs : les images peuvent être de mauvaise qualité voire pas ressemblantes à une classe en particulier, et donc très difficiles à annoter pour l'oracle.

### 1.2.2.3 Coût d'annotation des données

Comme indiqué précédemment, les données peuvent coûter plus ou moins cher à étiqueter : certaines données pour lesquelles le classifieur est incertain peuvent également être ambiguës pour l'oracle. Cela peut arriver notamment pour les petits objets tirés d'images satellites avec lesquels nous avons expérimentés : la résolution n'est pas forcément suffisamment grande pour obtenir un objet net, et la petite taille rend parfois difficile de distinguer la classe cible. Ainsi, cela peut prendre plus de temps d'annotation pour une telle image en comparaison d'une image plus simple : ce problème est notamment traité par (SETTLES et al. 2008; VIJAYANARASIMHAN et GRAUMAN 2011). Il est également possible d'avoir des étiquettes gratuites, et de ne payer que pour les exemples négatifs, notamment en détection des fraudes (SABATO et al. 2013).

### 1.2.2.4 Apprentissage par transfert

Il est possible d'utiliser l'apprentissage par transfert en amont de l'apprentissage actif, afin de démarrer ce dernier plus efficacement. L'apprentissage par transfert (WEISS et al. 2016) permet d'utiliser les connaissances d'un domaine initial où l'on dispose d'un grand nombre de données annotées, et un domaine cible avec peu ou pas de données annotées qui est similaire au domaine initial. Ainsi, appliquer directement le modèle entraîné sur le domaine initial ne donne pas de bons résultats sur le domaine cible, mais en entraînant une partie du modèle (notamment les parties entièrement connectées) sur le nouveau domaine, il est possible d'obtenir de bons résultats avec peu d'étiquettes. Cela se combine donc particulièrement bien avec l'apprentissage actif (CHATTOPADHYAY et al. 2013; GAVVES et al. 2015; X. WANG et al. 2014).

### 1.2.2.5 Les limitations de l'apprentissage actif

L'apprentissage actif peut être difficile à appliquer en pratique, notamment car la plupart des méthodes nécessitent de réentraîner le modèle après chaque nouvel échantillon ou paquet d'échantillons. Cela a un coût notamment pour les modèles d'apprentissage profond, ce qui amène les chercheurs à utiliser d'autres méthodes pour l'apprentissage actif profond, comme décrit dans la section 1.3. Le biais d'échantillonnage existe en apprentissage actif (DASGUPTA 2011; SETTLES 2011), et plus particulièrement avec les CNNs (SENER et SAVARESE 2017). Il est également important de noter que les oracles ne sont pas toujours parfaits dans leur annotation dans la réalité, et certains labels sont plus coûteux que d'autres, comme discuté précédemment. Un des autres problèmes courants en apprentissage actif est corroboré par les résultats de nos expériences sur différentes tâches :

les performances d'un algorithme d'apprentissage actif ne sont pas toujours aussi bonnes qu'espérées, et parfois pires que l'échantillonnage aléatoire. Nous avons en effet remarqué avec les résultats d'expériences que notre méthode était relativement plus efficace pour la détection de changements dans les images satellites, que sur les bases de classification d'images (CIFAR10, CIFAR100, Object-DOTA...) que nous avons testées. Divers articles de la littérature ont ainsi des conclusions similaires (Cawley 2011; Ebert et al. 2012).

## 1.3 Aperçu de l'apprentissage actif

### 1.3.1 Méthodes d'apprentissage actif et leur évolution

L'apprentissage actif étant utilisé depuis des dizaines d'années, de nombreuses stratégies ont vu le jour, et la plupart des méthodes avant l'ère de l'apprentissage profond sont décrites dans la revue de littérature de (Settles 2009).

#### 1.3.1.1 Les premières méthodes : des heuristiques

Les premières méthodes étaient des heuristiques choisies par des humains, comme par exemple l'échantillonnage aléatoire où les données à être annotées par l'oracle sont choisies aléatoirement, ou bien les méthodes de réduction de la variance présentées ci-après. De fait, elles reposent sur des intuitions humaines, par exemple le postulat "les données les plus incertaines vont probablement le plus aider le classifieur à être performant", ou bien "réduire la variance peut aider à réduire la fonction de coût". Nous décrivons ces méthodes de façon plus précise dans le chapitre d'état de l'art de l'apprentissage actif, néanmoins cela reste important d'en introduire quelques-unes dès maintenant afin de bien situer le contexte de la thèse.

**Echantillonnage par incertitude.** L'objectif de l'échantillonnage par incertitude est d'étiqueter les données pour lesquelles le classifieur est le plus incertain, et donc les plus proches de la frontière de décision du classifieur. Il existe plusieurs façons de définir l'incertitude d'une donnée, notamment l'entropie maximum des probabilités a posteriori sur les classes, les données dont la classe sélectionnée ont les probabilités "maximum" les plus minimales, et enfin la technique de la marge où l'on sélectionne les données avec la marge minimale entre les deux classes les plus probables. Il est également possible de combiner ces stratégies, (cf. Körner et Wrobel 2006).

**Méthodes par densité.** Les méthodes pondérées par densité visent à prendre en compte l'ensemble du jeu de données non étiquetées  $\mathcal{U}$  lors de la sélection des données. Il est ainsi possible d'éviter de sélectionner des valeurs extrêmes, notamment grâce à l'échantillonnage par représentativité (LI et GUO 2013), où le but est de sélectionner les échantillons les plus représentatifs de la distribution des données. Cette méthode est parfois combinée avec d'autres, telles que l'échantillonnage par incertitude, afin de sélectionner les données incertaines les plus représentatives. La stratégie d'échantillonnage par diversité permet également de s'assurer de sélectionner des échantillons différents des précédents, et est beaucoup utilisée dans le scénario basé sur un *pool* de données où un paquet de données est sélectionné à chaque cycle (YANG et al. 2015).

**Requête par vote.** Les méthodes de requête par vote (*Query-By-Committee* ou *QBC*) utilisent plusieurs classifieurs et sélectionne les données pour lesquelles ce comité de classifieurs a été le plus en désaccord dans ses prédictions. L'intuition est que les régions où les classifieurs sont en désaccord sont les plus informatives, et l'un des avantages de cette méthode est de ne pas nécessiter d'estimations des probabilités de classes, au contraire des méthodes d'échantillonnage par incertitude par exemple.

### 1.3.2 L'apprentissage actif profond pour l'image

L'apprentissage actif profond consiste en l'application de l'apprentissage actif, avec les réseaux profonds tels que les réseaux de neurones convolutifs (CNNs). Du fait que ces réseaux demandent généralement beaucoup de données annotées pour être efficaces, que leur entraînement se fait par paquets de données, les méthodes classiques d'apprentissage actif ne sont pas aussi performantes qu'escompté (SENER et SAVARESE 2017). Ainsi, de nouvelles méthodes d'apprentissage actif adapté aux réseaux profonds ont vu le jour (REN et al. 2021).

Afin de répondre à notre problématique, nous avons principalement utilisé des CNNs qui sont état de l'art dans la classification d'images, et les différentes méthodes d'apprentissage actif que nous avons élaborées sont ainsi adaptées aux réseaux profonds. Nous voyons plus en détail les différentes problématiques de l'apprentissage actif profond dans le chapitre suivant, cependant une courte présentation est nécessaire pour introduire nos travaux.

La reconnaissance et la classification d'images sont très importantes pour pouvoir appliquer l'apprentissage actif à d'autres tâches de vision par ordinateur, comme la détection d'objets, qui se fait grâce à des réseaux convolutionnels dits *backbone* pour classifier les objets après qu'ils aient été localisés. Ainsi, avoir un al-

gorithme d'apprentissage actif performant pour la classification d'images permet d'adapter ce dernier à la tâche de détection d'objets. L'un des principaux problèmes en apprentissage actif pour l'image concerne la sélection d'échantillons pertinents lorsque les données sont de très grande dimension, avec de nombreuses caractéristiques.

Différentes solutions ont été adoptées par la littérature, comme (K. WANG et al. 2016) utilise des pseudo-étiquettes aux échantillons pour lesquels le classifieur est confiant, afin de les ajouter à l'ensemble de données très incertaines et ainsi avoir des étiquettes gratuites pour entraîner le classifieur. Le papier (P. LIU et al. 2016) incorpore la représentativité et l'incertitude des données dans sa méthode pour sélectionner des images hyperspectrales pertinentes : la mesure d'incertitude seule n'est pas suffisante pour l'apprentissage actif profond, la diversité ou la représentativité des données est d'autant plus importante lorsque l'on prend les données par paquets.

L'apprentissage actif profond est également très utilisé pour l'analyse d'images médicales, notamment dans (FOLMSBEE et al. 2018). En effet, les jeux de données d'images médicales sont très difficiles à obtenir et requièrent des pathologistes experts, ce qui rend l'utilisation de l'apprentissage actif pertinente. La plupart des méthodes d'apprentissage actif profond pour les images médicales sont détaillées dans la revue de littérature (BUDD et al. 2021).

D'autres utilisations de l'apprentissage actif profond existent, comme par exemple la reconnaissance de captchas (STARK et al. 2015), ou bien les tâches de classification d'images en général (RANGANATHAN et al. 2017).

## 1.4 Démarche, contributions et leur positionnement

Nos différentes contributions visent à résoudre deux tâches : la détection de changements dans les images satellites, et la classification d'images dans des bases académiques (MNIST, CIFAR-10, CIFAR-100) et opérationnelles (Object-DOTA, tiré du jeu d'images satellites DOTA, cf. XIA et al. 2018). Ainsi, nous utilisons pour la détection de changements un classifieur SVM, ainsi que des réseaux convolutifs pour graphe (GCN). Pour la classification d'images, nous avons utilisé des CNNs et transformeurs pour la vision afin d'extraire les caractéristiques et classifier, ainsi que des forêts aléatoires pour la classification dans certains cas.

Notre méthode se situe dans le scénario d'apprentissage actif basé sur les *pools* de données, et nous combinons plusieurs stratégies classiques d'apprentissage actif dans un critère probabiliste unifié. En effet, notre fonction d'acquisition va sélectionner les données présentant le plus d'ambiguïté (à l'instar de l'échantillonnage par incertitude), de façon diversifiée et représentative. Ces trois critères

d’ambiguïté, diversité et représentativité sont respectivement obtenus grâce à la sortie *softmax* des CNNs utilisés, ou bien des scores de probabilités obtenus avec les forêts aléatoires; par l’entropie d’un sac de mots visuels obtenu grâce à l’algorithme k-means, en séparant les données en diverses partitions; et enfin en prenant les données les plus représentatives des différentes partitions, grâce à une mesure de distance telle que la distance euclidienne.

Cette méthode initiale a évolué par la suite, et a été améliorée en utilisant notamment l’apprentissage par renforcement afin de pondérer dynamiquement nos trois critères d’ambiguïté, diversité et représentativité au fur et à mesure des cycles d’apprentissage actif. La littérature s’est généralement cantonnée à n’utilise que l’incertitude et éventuellement la diversité ou la représentativité en parallèle, mais très rarement les trois, et encore moins dans un critère unifié. De plus, l’évolution dynamique de notre méthode au fur et à mesure des itérations est novatrice en comparaison des méthodes de la littérature.

Notre dernière contribution se situe cependant plutôt dans le scénario de génération de données synthétiques, car nous produisons des images synthétiques qui vont chercher à maximiser une nouvelle fonction objectif alliant l’incertitude et la représentativité des données. Les étiquettes de ces données générées seront alors celles des images réelles les plus proches en termes de distance, ce qui nous permet d’entraîner le réseau avec plus de données que l’on demande d’étiquettes à l’oracle.

## 1.4.1 Publications

### 1.4.1.1 Apprentissage actif pour la reconnaissance visuelle

Cette première contribution a donné lieu à une publication pour la conférence International Geoscience and Remote Sensing Symposium (IGARSS) de 2021, ainsi qu’à une présentation orale en visio-conférence (SAHBI et al. 2021). Nous avons présenté dans cet article un algorithme d’apprentissage actif novateur pour la détection de changements dans les images satellites, grâce à un modèle de questions et de réponses. Nous posons à un oracle (annotateur) les questions les plus informatives sur la pertinence des paires d’images satellites échantillonnées, et selon ses réponses, la fonction de décision est mise à jour itérativement. Nous modélisons ainsi la probabilité que les échantillons soient pertinents; cette dernière est obtenue en minimisant une fonction objectif qui capture la représentativité, la diversité et l’ambiguïté des données. Les données ayant la probabilité la plus élevée sont ainsi sélectionnés et soumises à l’oracle pour annotation, et servent à entraîner le modèle d’apprentissage automatique.



En plus de cet article, nous avons également effectué des expériences de classification d’images sur plusieurs jeux de données (MNIST, CIFAR10, CIFAR100 ainsi qu’une version adaptée à la classification d’images du jeu de données DOTA, que nous avons nommée Object-DOTA), et démontrons l’efficacité de notre méthode sur cette tâche.

#### 1.4.1.2 Apprentissage par renforcement pour la pondération des critères

Avec notre algorithme d’apprentissage actif basé sur les critères de diversité, représentativité et ambiguïté des données, nous utilisons des hyperparamètres de pondération de ces critères, respectivement  $\alpha, \eta, \beta$ . Ces paramètres ne changeaient pas au fur et à mesure des itérations, néanmoins les différentes études d’ablations que nous avons menées ont clairement démontré qu’un critère seul ou deux à deux n’était pas le meilleur pour chacune des itérations. Ainsi, nous avons eu l’idée de changer dynamiquement le poids des critères de diversité, représentativité et ambiguïté au fil des cycles d’apprentissage actif. Pour ce faire, nous utilisons l’apprentissage par renforcement et notamment l’algorithme du *Q-learning* sans états, et nous obtenons de bons résultats en mettant plus ou moins d’emphase sur les divers critères au fur et à mesure des itérations. Cela a donné lieu à deux publications, pour ICPR et IGARSS 2022 (DESCHAMPS et SAHBI 2022b; DESCHAMPS et SAHBI 2022a).

#### 1.4.1.3 Exemples virtuels informatifs pour l’apprentissage actif

Nous avons également conçu un nouvel algorithme interactif de classification d’images et de détection de changements dans les images satellites, basé sur l’apprentissage actif profond. Ainsi notre approche est itérative et s’appuie encore une fois sur un modèle par question et réponse avec la sélection du sous-ensemble d’images à annoter le plus pertinent à donner à l’oracle. La contribution principale de notre algorithme réside dans la sélection d’exemples virtuels les plus représentatifs et divers, lesquels remettent en cause le modèle appris, de sorte à obtenir un modèle très discriminatoire dans les cycles ultérieurs de l’apprentissage actif. Nous avons obtenu de bons résultats que ça soit en classification d’images ou en détection de changements, et ces travaux ont donné lieu à une publication dans IGARSS 2022 (SAHBI et DESCHAMPS 2022).



## 1.5 Organisation du manuscrit

Ce manuscrit de thèse est organisé en plusieurs chapitres : l'introduction qui présente les motivations ainsi que l'apprentissage actif classique mais aussi profond, le positionnement général de nos travaux par rapport à l'état de l'art, et nos diverses contributions. Ensuite, l'état de l'art est séparé en deux chapitres : un premier chapitre qui présente l'apprentissage profond et notamment les réseaux de neurones convolutifs ainsi que les transformeurs pour la vision, et un second pour présenter l'apprentissage actif profond et notamment pour l'image.

Ensuite, notre partie contribution est séparée en trois chapitres : notre première contribution avec la méthode probabiliste combinant les critères d'ambiguïté, diversité et représentativité des données, la deuxième contribution qui améliore la 1ère grâce notamment à l'apprentissage par renforcement, et enfin la dernière contribution à partir d'exemples virtuels pertinents pour l'apprentissage actif. Ce manuscrit se termine par un chapitre de conclusion et perspectives.

## RÉSEAUX DE NEURONES PROFONDS

### *Résumé chapitre*

*Dans ce chapitre, nous présentons l'évolution des réseaux de neurones : d'abord le perceptron et sa variante multi-couches, qui sont ensuite devenus les réseaux de neurones profonds dont nous présentons l'architecture typique. Ces derniers ont pour particularité d'avoir un grand nombre de couches cachées ainsi que des fonctions d'activation non linéaires. Ensuite, nous présentons les réseaux de neurones convolutifs, qui sont des réseaux de neurones profonds spécialisés pour les problèmes de vision par ordinateur. De fait, l'architecture de ces réseaux de neurones convolutifs ainsi que la manière d'entraîner un tel réseau sont abordées dans ce chapitre, avec en plus des modèles courants dont le ResNet-18 que nous avons utilisé dans la majorité de nos expériences pour la tâche de classification d'images. Finalement, les transformeurs pour la vision (ViT) sont présentés dans la dernière section de ce chapitre : issus du domaine du langage naturel, ils ont été adaptés aux tâches de reconnaissance visuelle et nous les avons utilisés pour certaines expériences récentes. Nous présentons l'architecture originale du transformeur avec notamment le mécanisme d'attention, la partie encodeur et décodeur, avant d'explicitement comment exploiter ces réseaux pour la vision.*

## Sommaire

---

|       |  |    |
|-------|--|----|
| 2.1   | Introduction aux réseaux de neurones . . . . .                   | 14 |
| 2.1.1 | Perceptron . . . . .   | 14 |
| 2.1.2 | Perceptron multi-couches. . . . .                                | 15 |
| 2.2   | Les réseaux de neurones profonds . . . . .                       | 17 |
| 2.2.1 | Fonctions d'activation non-linéaires . . . . .                   | 17 |
| 2.2.2 | Exemple d'architecture d'un réseau de neurones profond . . . . . | 18 |
| 2.2.3 | Fonctions d'activation des couches de sortie . . . . .           | 19 |
| 2.3   | Les réseaux de neurones convolutifs . . . . .                    | 20 |
| 2.3.1 | Architecture des réseaux de neurones convolutifs . . . . .       | 21 |
| 2.3.2 | Entraînement d'un réseau de neurones convolutif . . . . .        | 27 |
| 2.3.3 | Modèles de réseaux de neurones convolutifs courants . . . . .    | 34 |
| 2.4   | Transformeurs pour la vision (ViT) . . . . .                     | 41 |
| 2.4.1 | Encodeur et décodeur . . . . .                                   | 41 |
| 2.4.2 | Le mécanisme d'attention . . . . .                               | 42 |
| 2.4.3 | Architecture originale du transformeur . . . . .                 | 43 |
| 2.4.4 | Les transformeurs pour la vision par ordinateur . . . . .        | 43 |

---

## 2.1 Introduction aux réseaux de neurones

Dans ce chapitre nous décrivons les modèles d'apprentissage automatique que nous avons utilisés en majorité : les réseaux de neurones profonds ou Deep Neural Networks (DNNs) dans la littérature, et plus particulièrement les réseaux de neurones convolutifs plus connus sous le nom Convolutional Neural Networks (CNNs) ainsi que les transformeurs. Ces réseaux étant très performants dans beaucoup de domaines, il est important de bien les décrire et notamment leurs spécificités car cela influence la façon dont on fait de l'apprentissage actif / Active Learning (AL) conjointement avec de l'apprentissage profond / Deep Learning (DL). Nous introduirons également le perceptron original ainsi que le perceptron multi-couches, avant de passer aux réseaux de neurones profonds à proprement parler.

### 2.1.1 Perceptron

Le perceptron est un algorithme d'apprentissage supervisé de classifieurs binaires (ROSENBLATT 1958), il s'agit d'un neurone formel, c'est-à-dire une représentation mathématique d'un neurone biologique. On l'associe à un critère

d'apprentissage automatique afin de déterminer les poids qui multipliés aux entrées permettront de séparer un problème d'apprentissage supervisé. Il s'agit d'un réseau de neurones à une seule couche qui comprend des valeurs d'entrée, des poids et biais, une somme pondérée et enfin une fonction d'activation. L'algorithme consiste à prendre les valeurs d'entrées et les multiplier par leurs poids associés, avant de sommer ces valeurs pour obtenir une somme pondérée. Une fonction d'activation est alors appliquée sur cette somme afin de produire la sortie du perceptron, ce qui assure d'avoir une sortie entre 0 et 1 ou  $-1$  et  $1$  selon la fonction. Nous présenterons les différentes fonctions d'activation plus en détail par la suite; la [figure 2.1](#) illustre le fonctionnement du perceptron.

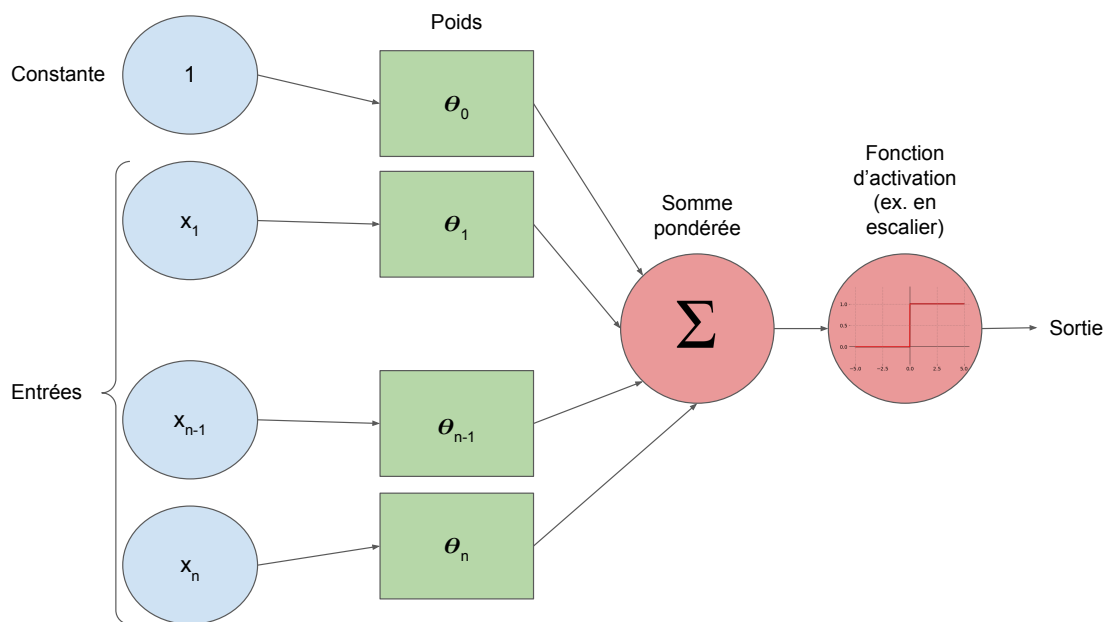


FIGURE 2.1 – **Le perceptron.** Les entrées  $x_i$  sont multipliées par leurs poids associés  $\theta_i$  ( $i = 0, \dots, n$ ) avant d'être sommées, puis une fonction d'activation est appliquée.

### 2.1.2 Perceptron multi-couches.

Le perceptron multi-couches / Multi-Layer Perceptron (MLP) définit une famille de fonctions qui vise à améliorer le perceptron classique. En effet, ce dernier est utile pour la classification de données linéairement séparables, mais est beaucoup moins efficace dans le cas contraire, comme notamment le problème du ou-exclusif (XOR); ce problème met en lumière l'existence d'un ensemble non linéairement séparable pour toute classification de quatre points. (cf. [figure 2.2](#))

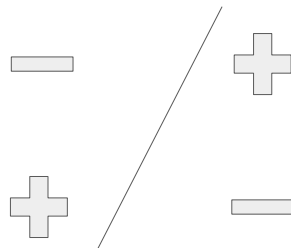


FIGURE 2.2 – **Le problème du ou-exclusif.** On remarque que ce problème en apparence très simple est pourtant composé de données non linéairement séparables.

Ainsi, le perceptron multi-couches permet de classifier des jeux de données non linéairement séparables, grâce à l'utilisation d'une architecture plus complexe afin d'apprendre des modèles de régression et classification. En plus des couches d'entrée et sortie du perceptron classique, la version multi-couches contient des couches cachées entre les deux comme l'on peut voir sur la [figure 2.3](#). L'algorithme

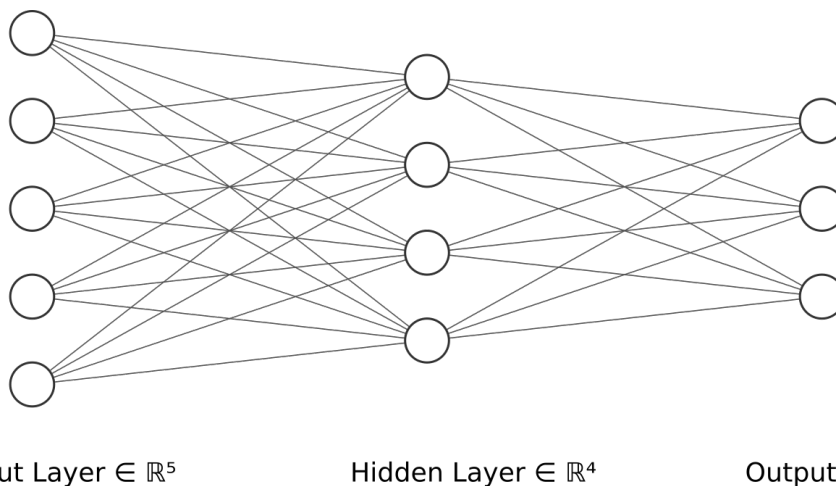


FIGURE 2.3 – **Le perceptron multi-couches.** Son architecture comprend des couches d'entrée, cachées et sortie entièrement connectées.

fonctionne de façon similaire au perceptron, le produit scalaire entre les entrées et les poids entre la couche d'entrée et cachée ( $\theta_h$ ) donne une valeur qui est ensuite transmise à une fonction d'activation puis envoyée vers la prochaine couche, encore une fois grâce au produit scalaire et les poids associés. Ce processus est répété jusqu'à ce que la couche de sortie soit atteinte, avant que tous ces calculs ne soient utilisés directement selon la sortie en phase de test, ou utilisés dans le cadre de l'algorithme de rétropropagation lors de la phase d'entraînement. Le perceptron multi-couches est ainsi l'élément de base de tous les réseaux de neurones qui ont suivi, et nous verrons les réseaux plus contemporains dans les sections suivantes.

## 2.2 Les réseaux de neurones profonds

Les DNNs proviennent des réseaux de neurones, et notamment des perceptrons multi-couches que l'on a vus précédemment ; la principale différence vient du nombre de couches cachées qui est très important dans le cas des réseaux de neurones profonds. Nous présenterons leur architecture, en commençant par les différentes fonctions d'activation non-linéaires qui peuvent être appliquées, comme par exemple dans la figure 2.4 tirée de (JIA et al. 2014 ; SZE et al. 2017), la liste est non exhaustive.

### 2.2.1 Fonctions d'activation non-linéaires

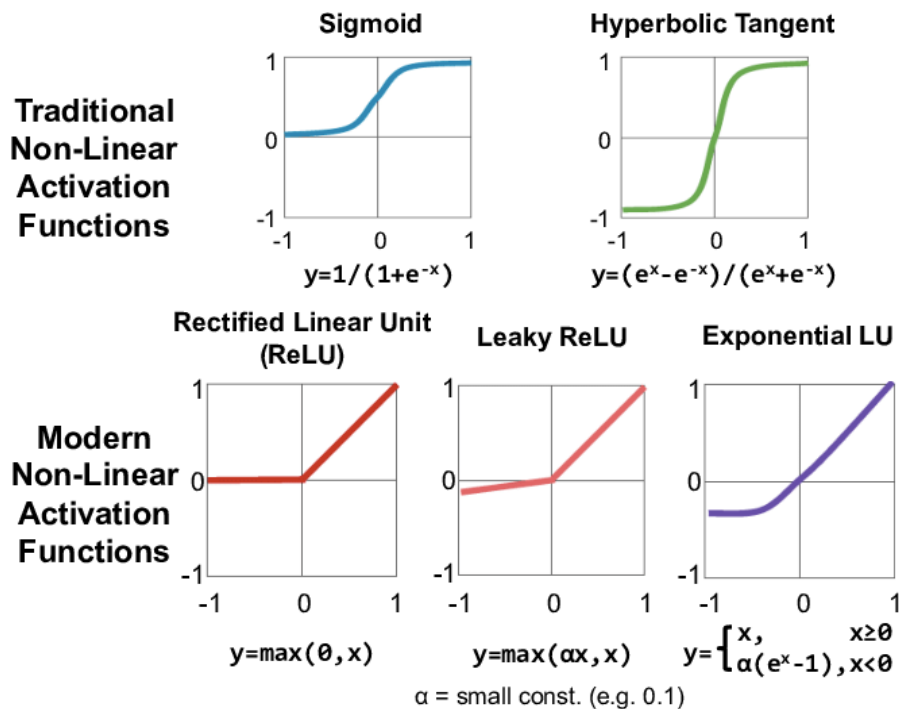


FIGURE 2.4 – Différentes fonctions d'activation non-linéaires. La sigmoïde et la tangente hyperbolique étaient en vogue auparavant, mais la fonction ReLU ou ses variantes sont très utilisées en apprentissage profond actuellement. (crédits : SZE et al. 2017)

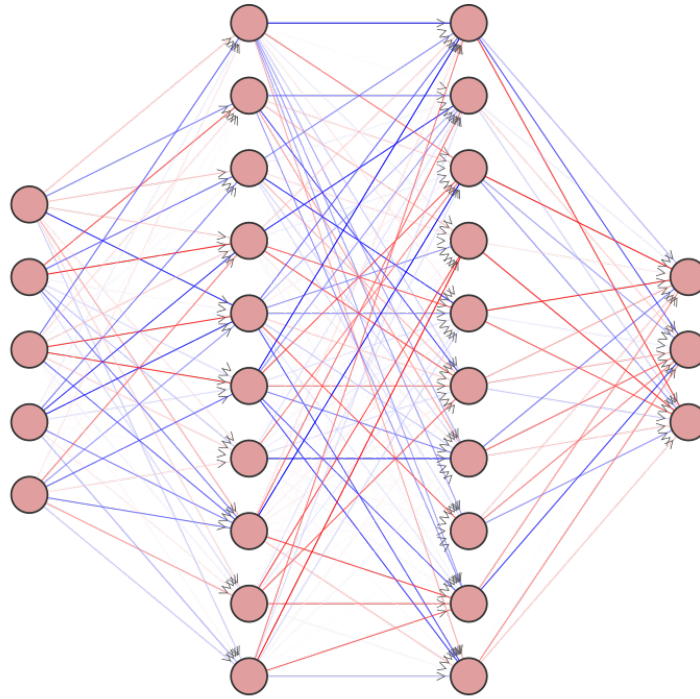
- **Sigmoid** : la fonction *sigmoïde* était très utilisée traditionnellement comme fonction d'activation non linéaire, en donnant une valeur entre 0 et 1 qui peut être utilisée comme probabilité notamment dans le cadre de la classification binaire avec seulement deux classes.

- **Hyperbolic Tangent** : la fonction *tangente hyperbolique* ( $\tanh$ ) est similaire à la fonction *sigmoïde* mais elle donne un résultat entre  $-1$  et  $1$  plutôt qu'entre  $0$  et  $1$ . De fait, les entrées négatives seront bien discriminées par rapport aux valeurs proches de  $0$ , contrairement à la fonction *sigmoïde* qui pourra les confondre. Utilisée aussi plutôt dans la classification binaire, la fonction  $\tanh$  est plus efficace en pratique que la fonction *sigmoïde* comme l'on peut voir dans le papier (KARLIK et OLGAC 2011).
- **Rectified Linear Unit (ReLU)** : la fonction *ReLU* est la fonction d'activation moderne la plus simple et la plus utilisée en pratique (on pourrait la traduire par unité linéaire redressée). Lorsque  $x > 0$  elle renvoie  $x$ , et renvoie  $0$  dans le cas contraire : il s'agit du maximum entre  $x$  et  $0$ , ce qui permet d'avoir un filtre sur les données en laissant passer les valeurs positives dans les couches suivantes du réseau. On l'utilise généralement dans les couches cachées; *ReLU* permet d'entraîner les réseaux très rapidement, ce qui participe à sa popularité dans l'apprentissage profond (NAIR et HINTON 2010). Un des problèmes avec cette fonction est que certains neurones peuvent ne jamais être activés, car leurs poids et biais ne sont pas mis à jour avec une valeur de gradient nulle pour  $x \leq 0$ .
- **Leaky ReLU** : la fonction *Leaky ReLU* que l'on pourrait traduire par "ReLU qui fuit" est une variante de ReLU qui permet d'augmenter les performances en précision (MAAS et al. 2013). On remarque ainsi sur la [figure 2.4](#) que la sortie aura des valeurs négatives non nulles pour  $x < 0$ .
- **Exponential LU** : la fonction *Exponential LU* (*ELU*) est également une variante de ReLU, plus accentuée encore que Leaky ReLU qui est utilisée afin d'obtenir des gains de performances marginaux (CLEVERT et al. 2015). Les valeurs sont lisses lorsque  $x < 0$ , ainsi *ELU* aura des valeurs négatives non nulles, contrairement à *ReLU*. Cela permet ainsi d'éviter le problème des neurones non activés de la fonction ReLU en aidant le réseau à déplacer les poids et biais dans la bonne direction, cependant le coût de calcul est plus élevé.

## 2.2.2 Exemple d'architecture d'un réseau de neurones profond

De manière plus informelle, les réseaux de neurones profonds sont composés de couches de noeuds, qui contiennent une couche d'entrée, plusieurs couches cachées, et une couche de sortie. Chaque noeud est connecté à un autre avec un poids et un seuil associé pour être activé ou non, ainsi si la sortie d'un noeud est au dessus de ce seuil, ce noeud est activé et enverra des données au prochain

noeud. Si ce n'est pas le cas, les données ne seront pas envoyées. De plus amples détails peuvent être trouvés dans les papiers suivants : (Weibo LIU et al. 2017; SZE et al. 2017). Un réseau de neurones dit *feedforward* comprenant une couche d'entrée, des couches cachées et une couche de sortie est présenté figure 2.5.



Input Layer  $\in \mathbb{R}^5$  Hidden Layer  $\in \mathbb{R}^{10}$  Hidden Layer  $\in \mathbb{R}^{10}$  Output Layer  $\in \mathbb{R}^3$

FIGURE 2.5 – **Diagramme d'un DNN.** On a ici 5 entrées et deux couches cachées de taille 10, ainsi qu'une couche de sortie à 3 classes. (crédits : outil fourni par LENAÏL 2019)

### 2.2.3 Fonctions d'activation des couches de sortie

Les couches de sortie utilisent également des fonctions d'activation, différentes de celles utilisées dans les couches intermédiaires. Voici une liste non exhaustive des fonctions d'activation pour les couches de sortie les plus courantes, ainsi que leur figures associées.

- **Softmax** : la fonction *softmax* présentée en figure 2.6 transforme un vecteur réel en vecteur de probabilités (les différentes valeurs sont positives ou nulles et leur somme est égale à 1). Cette fonction est donc très utilisée dans les problèmes multiclassés. On peut l'interpréter comme une probabilité d'appartenance aux classes, plus la valeur s'approche de 1 pour une classe  $c$  et plus le classifieur estime que la donnée appartient à la classe  $c$ , et inverse-



ment pour 0. Elle tire son nom de la fonction *argmax* (GOODFELLOW et al. 2016; WIKIPEDIA 2022b), tandis que *softmax* est une version plus douce (*soft*) d'*argmax*, continue et dérivable qui permet d'avoir une sortie sous forme de probabilité. Son équation est la suivante :

$$\text{softmax}(z)_j = \frac{\exp(z_j)}{\sum_{k=1}^K \exp(z_k)} \quad \forall j \in \{1, \dots, K\} \quad (2.1)$$

Ici, on a  $\mathbf{z} = (z_1, \dots, z_K)$  un vecteur de  $K$  nombres réels strictement positifs et de somme égale à 1.

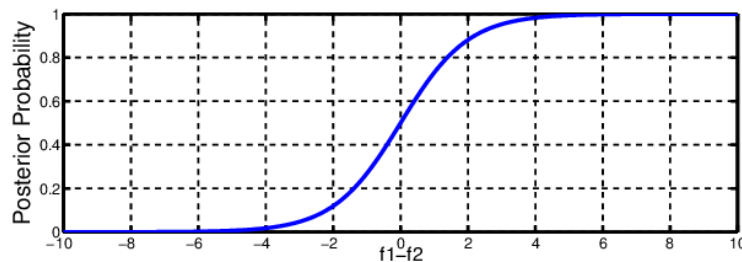


FIGURE 2.6 – **Représentation graphique de la fonction softmax.** Cette fonction transforme la sortie d'une couche en vecteur de probabilités et est très utilisée pour la classification multi-classes en dernière couche. (crédits : B. CHEN et al. 2017)

- **Linéaire** : la fonction *linéaire* correspond à l'identité : la valeur d'entrée est retournée directement en sortie, son équation est donc très simple. Généralement, lors de l'entraînement d'un modèle avec une fonction d'activation linéaire pour la couche de sortie, les données cibles sont mises à l'échelle en amont grâce à des transformations de normalisation.

$$\text{linear}(z)_j = z_j \quad \forall j \in \{1, \dots, K\} \quad (2.2)$$

Nous avons principalement utilisé des réseaux de neurones profonds convolutifs, nous expliquerons dans la section suivante leur particularité par rapport aux DNNs ainsi que la façon de les entraîner, les différentes fonctions de coût, etc.

## 2.3 Les réseaux de neurones convolutifs

Les réseaux de neurones convolutifs (CNNs), ou parfois nommés *ConvNet*, sont un type de réseaux artificiels acycliques (*feed-forward*), inspiré par le cortex visuel. Les CNNs sont utilisés dans de nombreux domaines, notamment la reconnaissance d'image et vidéo, le traitement du langage naturel, les systèmes de recommanda-

tion, etc. Nous verrons leur fonctionnement plus en détail, notamment à partir des articles suivants : (Weibo LIU et al. 2017; SZE et al. 2017; J. WU 2017; O'SHEA et NASH 2015; ROBINSON 2017).

### 2.3.1 Architecture des réseaux de neurones convolutifs

Les CNNs sont composés de plusieurs couches de convolution qui servent à l'extraction de caractéristiques, ces dernières sont ensuite données en entrée à des couches entièrement connectées pour la partie de classification. Nous verrons plus en détail les différents éléments qui composent un CNN, et comme notre application cible est la classification d'images, les données d'entrée correspondent à une image (2D en couleurs ou bien avec des niveaux de gris). On peut voir sur la figure 2.7 une illustration d'une architecture de CNN classique.

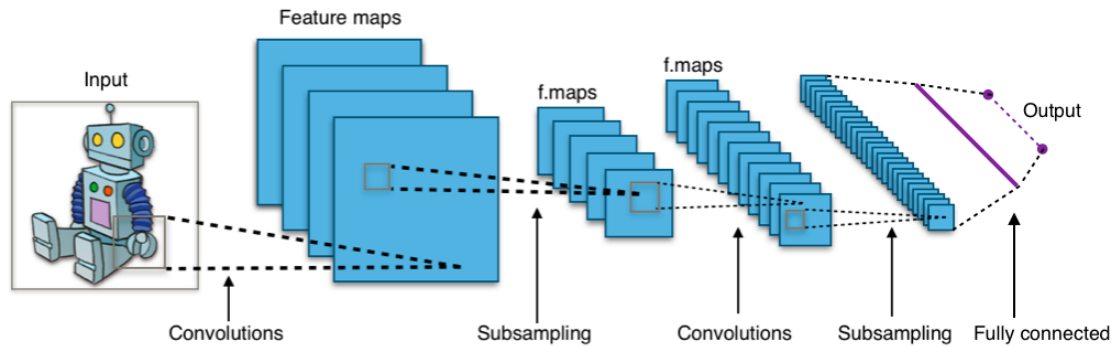


FIGURE 2.7 – Une architecture classique d'un CNN. On obtient des cartes des caractéristiques après les opérations de convolution, le sous-échantillonnage est effectué avec des couches de pooling ce qui réduit la taille de ces cartes, et des couches entièrement connectées sont utilisées à la fin du réseau. (crédits : WIKIPEDIA 2022a)

#### 2.3.1.1 Tenseurs

Les tenseurs sont une généralisation d'ordre supérieur des matrices et vecteurs. Ainsi,  $x \in \mathbb{R}^{H \times W \times D}$  est un tenseur d'ordre 3 (du troisième ordre), qui contient ainsi  $HWD$  éléments. Un tenseur d'ordre 3 peut être considéré comme contenant un certain nombre de matrices de taille  $H \times W$ , et l'on peut appeler  $D$  le nombre de canaux. Par rapport aux notations habituelles, un tenseur d'ordre 0 correspond à un scalaire, un vecteur est un tenseur d'ordre 1, une matrice est un tenseur d'ordre 2. Les images en entrée que nous utilisons sont généralement des tenseurs d'ordre 3 : l'image a  $H$  lignes et  $W$  colonnes, ce qui donne un tenseur de taille  $H \times W \times 3$  pour une image en couleur au format Rouge Vert Bleu (RVB). Les CNNs utilisent des tenseurs d'ordre 4, mais un processus d'aplatissement /

vectorisation (*flattening / vectorization*) est souvent utilisé ; la vectorisation consistera à transformer le tenseur d'ordre  $n$  en un vecteur / tenseur d'ordre 1 de façon récursive. Ainsi, on vectorise le premier canal, puis le deuxième, etc. jusqu'à ce que tous les canaux soient vectorisés, puis on concatène le tout. Un exemple de vectorisation d'une matrice / tenseur d'ordre 2 serait :

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \text{vec}(A) = (a, c, b, d)^T = \begin{bmatrix} a \\ c \\ b \\ d \end{bmatrix} \quad (2.3)$$

Ainsi, un CNN prend en entrée un tenseur d'ordre 3 (une image RVB avec  $H$  lignes et  $W$  colonnes), avant de passer par plusieurs étapes appelées couches comme décrit précédemment. Il en existe différents types : une couche de convolution, de pooling, de normalisation, entièrement connectée, etc.

### 2.3.1.2 Noyaux

Les noyaux (*kernels*) sont de petites matrices ayant diverses fonctions, par exemple le floutage, l'amélioration de la netteté de l'image, la détection de contours, etc. Dans notre cas nous utilisons ces noyaux pour le processus de convolution : on rajoute chaque élément de l'image à ses voisins, pondéré par les entrées du noyau. Voici un exemple de ce que peuvent produire différents noyaux sur une image (cf. figure 2.8).

| Opération                  | Noyau   | Résultat  |
|----------------------------|---|---|
| Identité                   | $\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$     |  |
| Amélioration de la netteté | $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ |  |

FIGURE 2.8 – **Différents noyaux appliqués à une image.** Le noyau identité appliqué à une image donnera la même image, tandis que l'amélioration de la netteté aura pour effet visible d'avoir des contours plus nets. (crédits : WIKIPEDIA 2022d)

### 2.3.1.3 Couche d'entrée

L'image d'entrée est placée dans cette couche, généralement en 2D 1-canal (niveaux de gris) ou 2D 3-canaux (RVB en couleurs) ou bien en 3D. Cela a une importance pour l'apprentissage des noyaux, ils doivent avoir la même profondeur que l'entrée. Ainsi, les couches d'entrée d'un CNN nécessitent une dimension précise pour éviter d'obtenir une mauvaise taille selon le pas (*stride* dans la littérature, un des paramètres de noyau qui contrôle la quantité de mouvement de ce noyau sur une image *ie.* le nombre de pixels / unités à chaque itération) du noyau et des couches de pooling : on redimensionne les images d'entrée en une taille commune adaptée à l'architecture du CNN que l'on utilise.

### 2.3.1.4 Couche de convolution

Cette couche est composée de convolutions de grande dimension, comprenant des cartes des caractéristiques (*feature maps*) d'entrée 2D qui forment un canal. Chacun de ces canaux est associé à un noyau 2D de la pile de filtres pour effectuer l'opération de convolution (cf. figure 2.9). Cette pile de filtres est souvent désignée

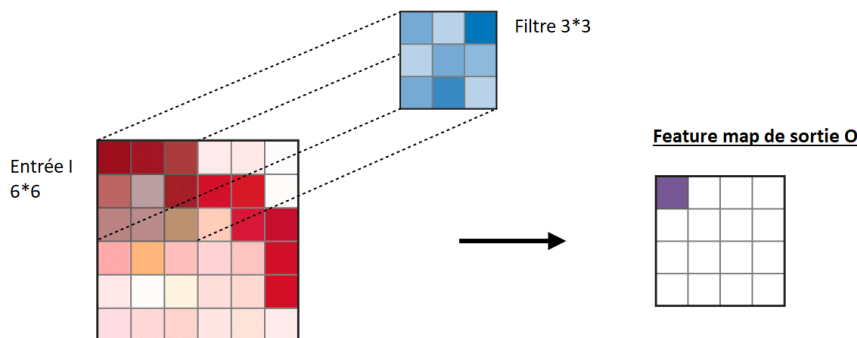


FIGURE 2.9 – **Illustration d'une couche de convolution.** La couche de convolution utilise un filtre  $3 \times 3$  pour effectuer les opérations de convolution sur l'entrée  $I$ , ce qui donne la couche de sortie  $O$  que l'on nomme carte des caractéristiques (ou attributs). La marge est de 0 et le pas de 1, (crédits : A. AMIDI et S. AMIDI 2018)

comme un seul filtre 3D, et les résultats des convolutions sur chaque point sont additionnés sur tous les canaux, et un biais de dimension 1 est ajouté. On obtient en sortie un ensemble de cartes qui vont indiquer où sont localisées les différentes caractéristiques de l'image, sachant que différents filtres permettent d'extraire des caractéristiques différentes, par exemple la détection de bord, des formes géométriques, etc. De plus amples informations sur le calcul d'une couche de convolution à plusieurs canaux peuvent être trouvées dans (SZE et al. 2017),

tandis que l'équation 2.4 montre le calcul plus simple du produit de convolution entre deux images  $f$  et  $g$  en niveau de gris,  $f * g$  (PERRET 2017) :

$$\forall (x, y) \in \mathbb{Z}^2, (f * g)(x, y) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f(i, j)g(x - i, y - j) \quad (2.4)$$

Pour éviter de perdre de l'information sur les bords avec le noyau, on ajoute à l'entrée une couche de marge (*padding*). De plus, on ajuste le déplacement des fenêtres glissantes grâce au paramètre de pas.

On parle souvent pour les couches de convolution de champ réceptif (*receptive field*), mais cela existe également pour les couches de pooling. Il s'agit dans le cadre de l'apprentissage profond de la taille de la région d'entrée qui produit la caractéristique (ARAUJO et al. 2019; WIKIPEDIA 2022e), sachant que les couches de convolution utilisent des bouts d'images (*patch*) avec une taille de fenêtre prédéfinie, et n'ont donc pas accès à toute la région d'entrée contrairement aux couches entièrement connectées pour lesquelles le concept de champ réceptif n'a donc pas lieu d'être.

### 2.3.1.5 Couche de pooling

La couche de pooling effectue une opération de sous-échantillonnage spatial, réduisant la dimension des données d'entrée, et donc le nombre de paramètres à apprendre du réseau. Cela permet notamment de faire en sorte que les couches suivantes du CNN peuvent appréhender des détails de haut niveau. La plupart du temps, on utilise des couches dites max-pooling avec des noyaux  $2 \times 2$  et un pas de 2, ce qui réduit l'entrée à 25% de sa taille originale, tout en maintenant la profondeur originale. On applique le calcul sur chaque carte des caractéristiques indépendamment, en les parcourant avec une fenêtre glissante pour y réaliser l'opération de pooling.

### 2.3.1.6 Couche entièrement connectée

Cette couche est équivalente aux couches que l'on trouve dans les DNNs présentés précédemment : c'est un ensemble de fonctions non-linéaires où chaque fonction est constituée d'un neurone ou bien d'un perceptron, une transformation linéaire est ensuite appliquée au vecteur d'entrée grâce à la matrice de poids associée à chaque neurone. Cela permet de produire des scores de classes grâce aux fonctions d'activation pour la classification. Ainsi, contrairement aux couches convolutionnelles, chaque entrée du vecteur d'entrée influence chacune des sorties du vecteur de sortie ; cependant les poids n'affectent pas forcément toutes

les sorties. Les couches entièrement connectées sont disposées après les autres couches des CNNs (convolution, pooling, etc.), et des fonctions ReLU sont utilisés entre ces couches afin de prendre en compte les non linéarités et aussi améliorer les performances.

### 2.3.1.7 Couche de normalisation

L'entraînement des réseaux de neurones profonds est complexe notamment du fait que la distribution des entrées de chaque couche change lors de l'entraînement, car les paramètres des couches précédentes ont changé. Cela a pour effet de ralentir l'entraînement de par la nécessité d'utiliser un taux d'apprentissage plus faible et de faire plus de réglages de paramètres. Le papier (IOFFE et SZEGEDY 2015) décrit ce phénomène comme "décalage interne des covariables" (*internal covariate shift*), et utilisent donc une couche de normalisation pour résoudre le problème : la normalisation par lot (*batch normalization* ou *BN*) dont nous présentons l'algorithme en figure 2.10.

|   |                        |
|---|------------------------|
| <b>Input:</b> Values of $x$ over a mini-batch: $\mathcal{B} = \{x_{1\dots m}\}$ ;               |                        |
| Parameters to be learned: $\gamma, \beta$   |                        |
| <b>Output:</b> $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$                                       |                        |
| $\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i$                                     | // mini-batch mean     |
| $\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2$        | // mini-batch variance |
| $\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$ | // normalize           |
| $y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i)$                 | // scale and shift     |

FIGURE 2.10 – **Algorithme de normalisation par lot.** La moyenne et variance de chaque valeur du mini-lot sont calculées pour permettre la normalisation. La dernière étape autorise les valeurs normalisées à être décalées vers une nouvelle moyenne et variance grâce aux facteurs  $\gamma, \beta$  qui sont appris directement par le réseau pour offrir les meilleures prédictions. (crédits : IOFFE et SZEGEDY 2015)

### 2.3.1.8 Couche de dropout

Le *dropout* que l'on pourrait traduire par "abandon", est une technique de régularisation très populaire en apprentissage profond, pas seulement pour les CNNs.

Les réseaux profonds ont beaucoup de paramètres, et cette grande quantité de neurones qui apprennent peut amener à un phénomène de sur-apprentissage (*overfitting*), ainsi les performances sur l'ensemble d'entraînement seront très bonnes mais la généralisation en souffrira et les performances qui comptent le plus (*ie.* sur le jeu de test) ne seront donc pas optimales. Le concept du dropout est donc de désactiver aléatoirement différents neurones à chaque itération d'apprentissage (SRIVASTAVA et al. 2014). Avant la démocratisation du dropout dans les réseaux de neurones profonds, d'autres techniques de régularisation étaient employées pour pallier le problème de sur-apprentissage, notamment les techniques de régularisation L<sub>1</sub>/L<sub>2</sub>. Ces techniques ne régularisaient pas suffisamment à cause du phénomène de co-adaptation, en effet lorsque tous les neurones apprennent ensemble certains seront plus efficaces que d'autres dans leur capacité à prédire, et au fur et à mesure les neurones moins prédictifs seront délaissés. Les techniques de régularisation classique comme L<sub>1</sub>/L<sub>2</sub> accentuaient le phénomène car elles régularisent selon les capacités prédictives des connexions, de fait les neurones les plus efficaces étaient davantage privilégiés et les moins efficaces d'autant plus abandonnés. Ce cercle vicieux empêchait de simplement augmenter la taille des réseaux pour améliorer la précision, les performances étaient donc plafonnées par la profondeur limitée des réseaux. Plus de détails sur la technique du dropout sont trouvables dans l'article (BALDI et SADOWSKI 2013), et le principe général du dropout est illustré en figure 2.11. On observe ainsi qu'il

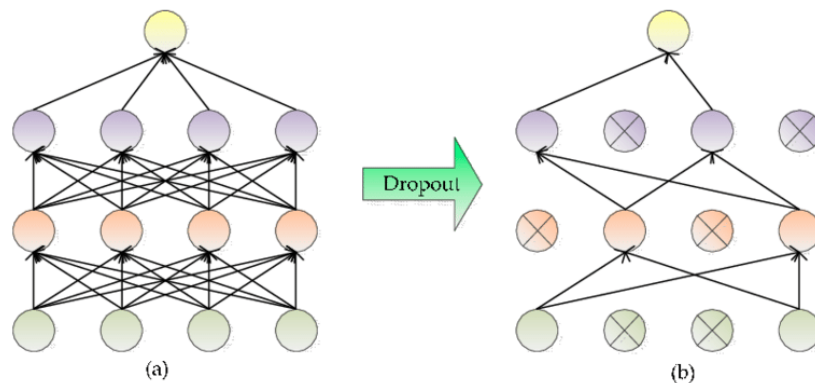


FIGURE 2.11 – **Illustration de la régularisation par dropout.** On remarque ainsi que le nombre de synapses a été grandement réduit avec la moitié des noeuds désactivés. (crédits : X. ZHANG et al. 2019)

y a beaucoup moins de connexions entre les neurones, ce qui force chacun des neurones actifs à apprendre pleinement sans être délaissés au profit de neurones avec de meilleures capacités prédictives. Comme les neurones désactivés seront différents à chaque passe, cela permettra au fil du temps d'avoir désactivé la grande majorité des neurones ; le paramètre contrôlant la proportion de neurones à désactiver à chaque itération est généralement fixé à 0.5 empiriquement. Il est à



noter que la plupart des CNNs ont des couches entièrement connectées dans leur architecture, et on applique le dropout sur ces dernières.

### 2.3.2 Entraînement d'un réseau de neurones convolutif

L'entraînement d'un CNN se fait de la même façon que pour un DNN et nous décrivons la procédure ici. Le *framework* (la librairie Python) d'apprentissage profond que nous avons employé est PyTorch (PASZKE et al. 2019; SUBRAMANIAN et al. 2021) qui est particulièrement adapté à la recherche par sa modularité et facilité de prise en main, mais il en existe d'autres comme TensorFlow, MxNet, Keras... Cette sous-section est principalement inspirée du livre de l'apprentissage profond (GOODFELLOW et al. 2016) et nous décrivons plusieurs concepts importants comme l'augmentation des données (*data augmentation*), les fonctions de coût, la rétropropagation du gradient, etc.

#### 2.3.2.1 Jeux et chargeurs de données

Afin de pouvoir séparer le code du modèle et le code de gestion des données, les différentes librairies d'apprentissage profond permettent l'utilisation de classes pour charger les données et définir son propre jeu de données ou utiliser l'un des jeux standards déjà disponibles comme CIFAR-10 ou MNIST. En apprentissage actif, nous avons besoin de sélectionner des données précises et il est donc important de bien prendre en compte la façon dont la librairie d'apprentissage profond que nous utilisons gère les données, leur ordonnancement, etc. De plus, nous avons utilisé un jeu de données personnalisé *ObjectDOTA* adapté de (XIA et al. 2018), ainsi nous avons implémenté un *Dataset* personnalisé.

#### 2.3.2.2 Augmentation des données grâce aux transformations

Il existe différentes techniques utilisées avec les CNNs afin d'augmenter leur capacité de généralisation et leurs performances, et les différentes librairies d'apprentissage profond permettent donc d'appliquer des transformations aux images notamment pour l'augmentation des données. Avant d'entraîner le réseau sur les données, on procède donc aux transformations, et voici une liste non exhaustive de transformations populaires (cf. TORCHCONTRIBUTORS 2017) :

- *CenterCrop* : Cette transformation découpe l'image d'entrée au centre (cf. figure 2.12).





FIGURE 2.12 – Transformation CenterCrop.

- *Grayscale* : Cette transformation convertit l'image d'entrée en niveaux de gris (cf. figure 2.13).

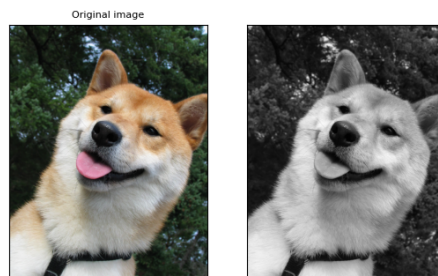


FIGURE 2.13 – Transformation Grayscale.

- *ColorJitter* : Cette transformation change aléatoirement la luminosité, saturation et d'autres propriétés d'une image (cf. figure 2.14).

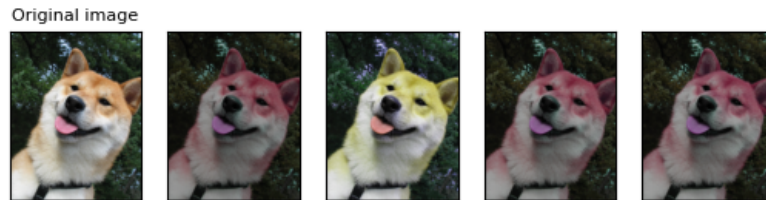


FIGURE 2.14 – Transformation ColorJitter.

- *RandomRotation* : Cette transformation effectue une rotation de l'image d'entrée avec un angle aléatoire (cf. figure 2.15).

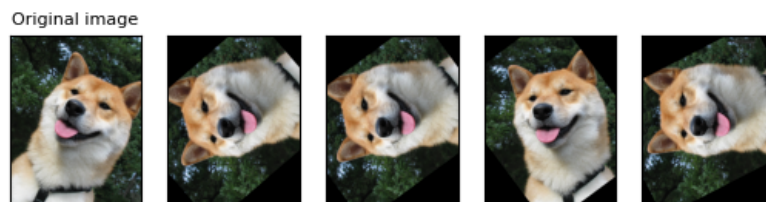


FIGURE 2.15 – Transformation RandomRotation.

- *RandomCrop* : Cette transformation découpe l'image à un emplacement aléatoire (cf. figure 2.16).



FIGURE 2.16 – Transformation RandomCrop.

- *RandomHorizontalFlip* : Cette transformation effectue un basculement horizontal de l'image d'entrée, selon une probabilité donnée (cf. figure 2.17).

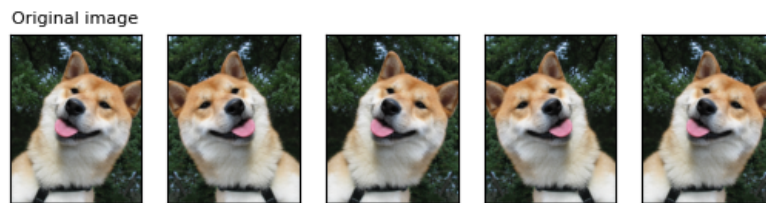


FIGURE 2.17 – Transformation RandomHorizontalFlip.

- *RandomVerticalFlip* : Cette transformation effectue un basculement vertical de l'image d'entrée, selon une probabilité donnée (cf. figure 2.18).

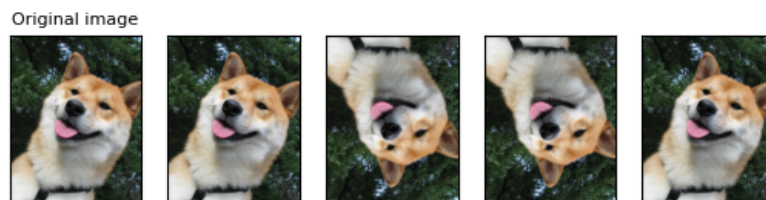


FIGURE 2.18 – Transformation RandomVerticalFlip.

Ainsi, en plus des données originales, le réseau apprend grâce à des données supplémentaires qui sont des transformations des images originales. Cela permet d'obtenir plus de données mais également d'améliorer les *features* afin qu'elles soient plus invariantes aux différentes transformations appliquées.

### 2.3.2.3 Fonction de coût

La fonction de coût (*loss function*) la plus utilisée est l'entropie croisée ou *cross-entropy* dans la littérature. On définit ainsi une distribution  $p(\mathbf{y}|\mathbf{x}; \theta)$  où  $\mathbf{y}$  sont les

labels,  $x$  les données d'entrée et  $\theta$  les paramètres du modèle avant d'appliquer l'estimateur du maximum de vraisemblance. On utilise ainsi l'entropie croisée entre les données d'entraînement et les prédictions du modèle, et on peut ajouter un terme de régularisation à notre fonction de coût. L'équation de l'entropie croisée dans le cas de la classification multi-classes est :

$$-\sum_{i=1}^N y_{x,i} \log(\hat{y}_{x,i}) \quad (2.5)$$

Avec  $N$  le nombre de classes,  $y_{x,i} = (y_{x,1}, \dots, y_{x,N})$  le vecteur de vérité terrain pour chaque donnée  $x$  et  $\hat{y}_{x,i} = (\hat{y}_{x,1}, \dots, \hat{y}_{x,N})$  le vecteur prédit par le modèle pour chaque donnée  $x$ , la somme est donc effectuée sur toutes les classes  $i \in \{1, \dots, N\}$ .

### 2.3.2.4 Dérivation automatique

L'algorithme principal utilisé pour entraîner les réseaux de neurones (profonds) est l'algorithme de rétropropagation (*back propagation* RUMELHART et al. 1986) où les poids du modèle / paramètres sont mis à jour en fonction du gradient de la fonction de coût (*loss function* ou *loss*) par rapport à ces paramètres. Le but est donc de réduire l'erreur de chaque paramètre, pour avoir différents paramètres qui produisent de bonnes prédictions. Néanmoins, la rétropropagation est utilisée pour calculer le gradient tandis que l'apprentissage en lui-même est effectué par ce qu'on appelle un optimiseur (*optimizer*), le plus couramment utilisé pour les DNNs étant la Stochastic Gradient Descent (SGD) (Descente de gradient stochastique), mais il en existe plusieurs autres, notamment Adam (KINGMA et BA 2014) qui est également un optimiseur populaire avec les CNNs. Nous verrons les optimiseurs plus en détail dans le paragraphe suivant dédié, après avoir expliqué l'algorithme de rétropropagation du gradient. Afin de trouver les dérivées de la fonction de

---

**Procédure 2.1** Algorithme de la passe avant d'un réseau de neurones basique.

---

- 1: **procedure** NET\_FORWARD( $x, \theta_h, \theta_o$ ) :
  - 2: **Entrée** :  $x$  : donnée d'entrée,  $\theta_h$  : poids couche cachée,  $\theta_o$  : poids couche de sortie
  - 3: **Sortie** :  $\hat{y}, Z_o, Z_h$  : la prédiction du réseau, calculs couches.
  - 4:      $Z_h = \langle x, \theta_h \rangle$  // CALCUL COUCHE CACHÉE
  - 5:      $H = \text{ReLU}(Z_h)$
  - 6:      $Z_o = \langle H, \theta_o \rangle$  // CALCUL COUCHE DE SORTIE
  - 7:      $\hat{y} = \text{ReLU}(Z_o)$
  - 8: **return**  $\hat{y}, Z_o, Z_h$
  - 9: **end procedure**
- 

coût en fonction des paramètres, on utilise la règle de la chaîne (*chain rule*). La

rétropropagation fonctionne ainsi pour les DNNs (GOODFELLOW et al. 2016; LECUN et al. 2015; SCHMIDHUBER 2015) :

- Règle de la chaîne : soit une fonction *forward*  $f(x) = a_1(a_2(a_3(x)))$  avec  $a_1, a_2, a_3$  des fonctions d'activation pour différentes couches de notre réseau. La règle de la chaîne permet de calculer les dérivées de  $f(x)$  en fonction de  $x$  par  $f'(x) = f'(a_1).a'_1(a_2).a'_2(a_3).a'_3(x)$ .
- Application sur un réseau avec un seul neurone :  $z = \theta.x$  l'entrée pondérée dont la dérivée est  $z'(x) = \theta$  et  $z'(\theta) = x$ ; une fonction d'activation ReLU  $r = \max(0, z)$  dont la dérivée est  $r'(z) = \begin{cases} 0 & \text{if } z \leq 0 \\ 1 & \text{if } z \geq 0 \end{cases}$ ; une fonction de coût  $c = \frac{1}{2}(\hat{y} - y)^2$  dont la dérivée est :  $c'(\hat{y}) = (\hat{y} - y)$ . On aura alors un coût total  $c(r(z(\theta x)))$  et on peut donc trouver la dérivée du coût par rapport au poids  $\theta$  grâce à la règle de la chaîne vue précédemment. Ainsi  $c'(\theta) = c'(r).r'(z).z'(\theta) = (\hat{y} - y).r'(z).x$ , et on peut remonter de la sortie jusqu'à l'entrée en appliquant la règle de la chaîne récursivement, il faudra donc garder en mémoire les dérivées déjà calculées lorsque le réseau est profond.
- La rétropropagation en elle-même : on calcule l'erreur de la couche de sortie et on passe le résultat à la couche cachée précédente dont on calcule le gradient ou dérivée également, que l'on passe à la couche précédente, etc. La dérivée du coût par rapport aux poids de la couche nous indique donc la direction dans laquelle faire tendre les poids afin de réduire le coût total (RUMELHART et al. 1986).

Les bibliothèques d'apprentissage profond disposent d'un moteur de dérivation automatique appelé *torch.autograd* (pour celle que nous utilisons) qui permet le calcul automatique du gradient pour n'importe quel graphe de calcul, que nous utilisons afin d'éviter de calculer manuellement les gradients.

### 2.3.2.5 Optimisation

L'optimisation des paramètres se fait de façon itérative : à chaque itération (*epoch*) le modèle prédit une sortie, calcule l'erreur associée à sa prédiction en comparaison des données de vérité terrain (calcul de la *loss*), récupère les dérivées associées aux différents poids comme vu précédemment, puis optimise les paramètres grâce à l'algorithme de descente de gradient ou autre. Le but est de réduire l'erreur du modèle à chaque itération d'entraînement, et lorsque l'erreur commence à converger, ou bien si on a un jeu de données de validation à dis-

position, l'entraînement du modèle est stoppé afin d'évaluer ses performances sur l'ensemble de test. Il existe ainsi divers algorithmes d'optimisation pour ajuster les paramètres du modèle afin de réduire son erreur, nous en présenterons quelques-uns, inspiré notamment de (RUDER 2016).

- *Notations* : soit  $J(\theta)$  la fonction objectif à minimiser et  $\theta \in \mathbb{R}^d$  les paramètres du modèle,  $\eta$  le taux d'apprentissage (*learning rate*) qui détermine la taille du pas que l'on fait afin d'atteindre un minimum.
- **Vanilla gradient descent** : cette variante originale de la descente de gradient, aussi nommée *batch gradient descent*, calcule le gradient de la fonction de coût par rapport aux paramètres  $\theta$  pour le jeu de données d'entraînement entier et met à jour les paramètres via  $\theta = \theta - \eta \cdot \nabla_{\theta} J(\theta)$ . L'idée de la descente de gradient est de suivre la direction de la pente de la surface créée par la fonction de coût à optimiser jusqu'à atteindre une vallée / un minimum local le plus proche possible du minimum global. L'inconvénient principal de cette version de la descente de gradient est sa lenteur : en effet, il est nécessaire de calculer les gradients sur le jeu de données entier pour faire une seule mise à jour, ce qui peut se révéler très problématique sur de gros jeux de données, et le modèle ne peut pas être mis à jour avec de nouveaux exemples à la volée. On met ensuite à jour les paramètres dans la direction opposée des gradients, comme vue dans le paragraphe sur la rétropropagation. Cet algorithme permet donc de converger vers le minimum global lorsque la surface de la fonction objectif est convexe, et vers un minimum local autrement.
- **Stochastic gradient descent** : la descente de gradient stochastique met à jour les paramètres pour chacun des exemples  $x_i$  et étiquettes  $y_i$  associées :  $\theta = \theta - \eta \cdot \nabla_{\theta} J(\theta; x_i; y_i)$ . Cela permet un calcul beaucoup plus rapide et permet l'apprentissage à la volée, néanmoins ces mises à jour très fréquentes induisent une grande variance dans l'erreur de la fonction objectif. Cela permet potentiellement d'aller vers de meilleurs minimum locaux que la version vanilla, mais la convergence vers le meilleur minimum peut être lente à cause d'un phénomène de dépassement (*overshooting*). On peut jouer sur une mise à jour dynamique du taux d'apprentissage afin de régler ces problèmes de convergence.
- **Mini-batch gradient descent** : la descente de gradient par mini-lot met à jour les gradients pour chaque mini-lot de  $n$  exemples d'entraînement de la façon suivante :  $\theta = \theta - \eta \cdot \nabla_{\theta} J(\theta; x_{i:i+n}; y_{i:i+n})$ . Cela permet de rendre la convergence potentiellement plus stable en réduisant la variance lors de la mise à jour des paramètres car on n'influence pas tous les paramètres d'un seul coup. De plus, les bibliothèques d'apprentissage profond permettent de faire

les calculs de gradients par mini-lot de façon très efficace ce qui accélère grandement les choses en comparaison de la descente de gradient *vanilla*. Il est à noter que l'optimiseur appelé communément **SGD** est généralement appliqué pour faire une descente de gradient par mini-lot. On peut en effet appliquer les trois algorithmes principaux de descente de gradient selon la taille du lot que l'on utilise, et on utilise généralement une taille de lot fixe inférieure à la taille du jeu de données entier mais supérieure à 1.

- *Optimisation de la descente de gradient* : comme le choix du taux d'apprentissage est compliqué (trop bas et la convergence devient très lente, trop grand et cela peut diverger ou rester bloqué près du minimum, voir [figure 2.19](#)), diverses stratégies sont adoptées. Des ordonnanceurs de taux d'apprentissage

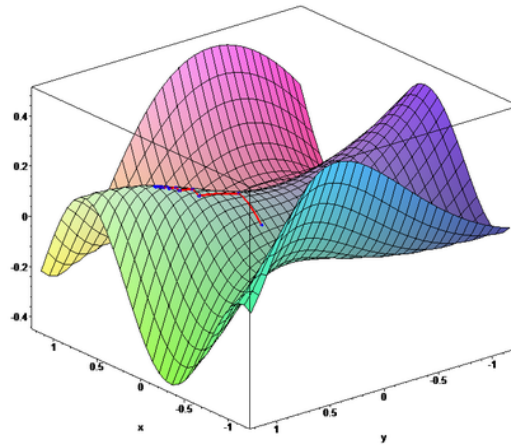


FIGURE 2.19 – **Descente de gradient illustrée.** On remarque ici les difficultés que peut avoir l'algorithme de descente de gradient sur les zones plates, avec une oscillation du gradient en boucle près des points de selle. (crédits : WIKIPEDIA 2022C)

(*learning rate schedulers*) permettent notamment de changer dynamiquement ce dernier, en le réduisant progressivement après un certain nombre d'itérations, ou lorsque l'erreur est en dessous d'un certain seuil, etc. Néanmoins, cette technique doit être faite avant l'apprentissage donc les réglages peuvent être imparfaitement choisis. Ainsi, des algorithmes sont utilisés en parallèle pour tenter de résoudre les différents problèmes de la descente de gradient, par exemple le *momentum* (que l'on pourrait traduire par l'élan ou la dynamique) en ajoutant une fraction de la mise à jour précédente à la mise à jour actuelle :  $v_t = \gamma v_{t-1} + \eta \nabla_{\theta} J(\theta)$ ;  $\theta = \theta - v_t$ .

- **Adam** : l'algorithme d'estimation adaptative des moments / Adaptive Moment Estimation, provient de l'article (KINGMA et BA 2014) et permet d'avoir des taux d'apprentissages adaptés à chacun des paramètres et permet de garder



l'information sur les anciens gradients de façon similaire à la technique du momentum. C'est un optimiseur très utilisé en apprentissage profond avec de bons résultats empiriques sur divers jeux de données.  $m_t$  et  $v_t$  sont des estimations de la moyenne et de la variance décentrée des gradients (premier et second moments). Adam utilise ces estimations du premier et second moments des gradients afin d'adapter le taux d'apprentissage pour chacun des poids du réseau (cf. [Algorithme 2.2](#)).

---

**Procédure 2.2** Algorithme Adam.
 

---

```

1: procédure ADAM( $\gamma, \beta_1, \beta_2, \theta_0, f(\theta), \lambda$ ) :
2: Entrée :  $\gamma$  : taux d'apprentissage,  $\beta_1, \beta_2$  : bêtas,  $\theta_0$  : paramètres,  $f(\theta)$  : objectif,
    $\lambda$  : dégradation des pondérations.  $\beta_1^t, \beta_2^t$  désignent  $\beta_1, \beta_2$  à la puissance  $t$ .
3: Sortie :  $\theta_t$  : paramètres optimisés.
4:    $m_0 \leftarrow 0, v_0 \leftarrow 0$ 
5:   for  $t = 1$  to ... do
6:      $g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$  // RÉCUPÉRATION GRADIENTS À L'INSTANT  $t$ 
7:     if  $\lambda \neq 0$  then
8:        $g_t \leftarrow g_t + \lambda \theta_{t-1}$  // OPTIONNEL
9:     end if
10:     $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$  // MISE À JOUR ESTIMATION 1ER MO-
      MENT BIAISÉ
11:     $v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$  // MISE À JOUR ESTIMATION 2ND MO-
      MENT BIAISÉ
12:     $\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$  // CALCUL ESTIMATION 1ER MOMENT AVEC
      BIAIS CORRIGÉ
13:     $\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$  // CALCUL ESTIMATION 2ND MOMENT AVEC
      BIAIS CORRIGÉ
14:     $\theta_t \leftarrow \theta_{t-1} - \gamma \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$  // MISE À JOUR DES PARAMÈTRES
15:  end for
16: return  $\theta_t$  // PARAMÈTRES RÉSULTANTS
17: end procédure

```

---

### 2.3.3 Modèles de réseaux de neurones convolutifs courants

Depuis l'avènement de l'apprentissage profond avec notamment le succès du CNN *AlexNet* au Image Net (Large Scale Visual Recognition Challenge, ILSVRC), de nombreux modèles différents de CNNs ont vu le jour. Nous en présenterons quelques-uns : tout d'abord *LeNet*, l'un des premiers réseaux de ce genre qui n'a cependant pas obtenu le succès escompté. Ensuite nous présenterons *AlexNet*, qui a grandement participé à l'essor de l'apprentissage profond avec sa victoire au concours de vision par ordinateur susmentionné, *VGG* qui a été une des

premières grosses améliorations après *AlexNet* et dont l'architecture est souvent utilisée à des fins éducatives lorsque l'on enseigne le fonctionnement des CNNs. Nous présenterons finalement *ResNet*, ce réseau a été une avancée significative en termes de performances et d'architecture, et c'est également le CNN que nous avons le plus utilisé dans nos travaux.

### 2.3.3.1 LeNet

Le réseau *LeNet* proposé par (LECUN et al. 1998, tout du moins dans sa version la plus connue, *LeNet-5*, mais l'approche a été introduite bien plus tôt par LECUN et al. 1989 également) est un des premiers CNNs. La première version était utilisée pour la classification de chiffres en images noir et blanc  $28 \times 28$  pixels, tandis que *LeNet-5* contient plusieurs couches de convolution et entièrement connectées, entraîné par rétropropagation pour la reconnaissance de caractères manuscrits cette fois-ci. Néanmoins, son manque de compétitivité face à l'état de l'art et notamment les algorithmes utilisant les Support Vector Machines (SVMs) (Machine à vecteurs de support) empêche les CNNs de vraiment se démocratiser. Son architecture est visible dans la figure 2.20, il contient environ 60000 paramètres.

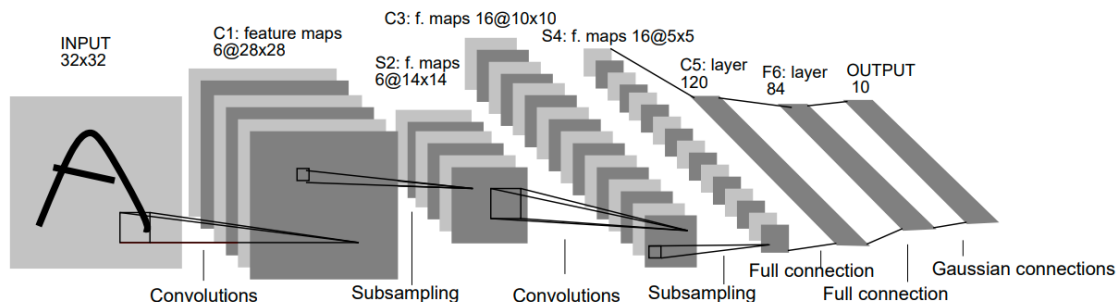


FIGURE 2.20 – Architecture du CNN LeNet-5. Ici appliqué à la reconnaissance de chiffres manuscrits, avec des couches de convolution aux cartes des caractéristiques de différentes tailles, des couches de pooling et des couches entièrement connectées. (crédits : LECUN et al. 1998)

### 2.3.3.2 AlexNet

Le réseau *AlexNet* proposé par (KRIZHEVSKY et al. 2012) est le premier CNN à avoir remporté la compétition ImageNet Large Scale Visual Recognition Challenge (ILSVRC), en 2012. Ce réseau a propulsé les CNNs sur le devant de la scène en vision par ordinateur, et a grandement aidé à motiver les recherches sur l'apprentissage profond. L'erreur *top-5* d'*AlexNet* était d'environ 15%, un énorme gap comparé aux autres papiers en compétition qui plafonnaient à environ 25%. Ce réseau a repris le principe de *LeNet* en l'adaptant à un réseau large et profond : le nombre



de paramètres est ainsi bien supérieur aux 60k de LeNet, avec environ 61 millions de paramètres. Ce réseau est alors innovant sur plusieurs points :

- Il utilise des Graphics Processing Units (GPUs) (Processeur graphique) pour accélérer les calculs, ce qui permet notamment d'avoir un réseau aussi profond. Par manque de mémoire avec les cartes graphiques de l'époque (3Go de Video Random Access Memory (VRAM) / Mémoire vive vidéo alors que les cartes grand public actuelles ont une dizaine de Go de mémoire vidéo...). De fait, le réseau nécessite deux cartes graphiques Nvidia GTX 580 pour effectuer les calculs de convolutions.
- AlexNet utilise la fonction d'activation non-linéaire *ReLU* que l'on a présentée dans la [section 2.2](#), tandis que la fonction *tangente hyperbolique* était plus populaire précédemment, mais beaucoup moins rapide à converger. De plus, l'utilisation de la fonction ReLU permet d'utiliser un réseau très profond en minimisant le problème du gradient qui disparaît.
- AlexNet n'utilise pas la technique de normalisation par lot présentée précédemment, pas encore populaire à ce moment, mais une technique de régularisation est tout de même appliquée : la Local Response Normalization (LRN) (Normalisation par réponse locale) qui permet d'améliorer la capacité de généralisation du modèle.

$$b_{x,y}^i = a_{x,y}^i / \left( k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a_{x,y}^j)^2 \right)^\beta \quad (2.6)$$

Ici,  $i$  correspond à la sortie du filtre  $i$ ,  $a_{x,y}^i, b_{x,y}^i$  sont les valeurs des pixels aux positions  $(x, y)$  avant et après normalisation et  $N$  le nombre de canaux. Il y a également des hyper-paramètres :  $(k, \alpha, \beta, n)$  :  $k$  évite qu'il y ait division par zéro,  $\alpha$  sert de constante de normalisation,  $\beta$  de constante de contraste et  $n$  est utilisé pour définir le nombre de valeurs de pixels consécutifs à considérer lors de la normalisation. Les auteurs KRIZHEVSKY et al. ont déterminé ces hyper-paramètres avec un ensemble de validation et ont utilisé les valeurs :  $k = 2, n = 5, \alpha = 10^{-4}, \beta = 0.75$

- AlexNet utilise la technique du max-pooling plutôt que l'average-pooling qui était utilisé précédemment dans les CNNs ce qui améliore légèrement les résultats. De plus, les auteurs utilisent du chevauchement afin d'améliorer les résultats de l'ordre de 0.4% et 0.3% pour l'erreur top-1 et top-5 respectivement. Pour obtenir du chevauchement, ils choisissent un pas  $s$  de taille inférieure à la taille du noyau  $z$ . Les valeurs qu'ils ont utilisées en pratique sont :  $s = 2$  et  $z = 3$  (KRIZHEVSKY et al. 2012).

- AlexNet utilise des techniques d'augmentation des données très efficaces :
  1. La première forme d'augmentation des données qu'il utilise est la mise en miroir : ils génèrent des translations d'images et des reflets horizontaux. Pour ce faire, ils récupèrent des bouts d'images aléatoirement de taille  $224 \times 224$  pixels et leurs reflets horizontaux depuis les images originales de taille  $256 \times 256$  pixels. Cela leur permet d'augmenter la taille de leur ensemble d'entraînement d'un facteur 2048 : les translations ajoutent  $(256 - 224)^2 = 32^2 = 1024$  possibilités, que l'on multiplie par deux avec les reflets horizontaux :  $1024 \times 2 = 2048$ . Cela leur permet d'éviter le sur-apprentissage ce qui les autorise à garder un réseau très profond. Au moment du test, le réseau fait une prédiction après extraction de cinq bouts d'image  $224 \times 224$  (les quatre coins et le centre du bout d'image) ainsi que leurs reflets horizontaux pour arriver à dix bouts d'images. Ils prennent ensuite la moyenne des prédictions du réseau sur ces dix bouts d'images.
  2. La deuxième forme d'augmentation des données qu'il utilise est d'altérer les intensités des canaux RVB dans les images d'entraînement. Ils appliquent la technique Principal Component Analysis (PCA) (Analyse en composantes principales) sur les valeurs des pixels RVB de l'ensemble d'entraînement. Pour chaque image d'entraînement, ils ajoutent des multiples des composantes principales, dont les magnitudes sont proportionnelles aux valeurs propres (*eigenvalues*) correspondantes multipliées par une variable aléatoire tirée d'une loi gaussienne de moyenne 0 et d'écart type 0.1. Ainsi, ils ajoutent à chaque pixel d'image RVB  $I_{xy} = [I_{xy}^R, I_{xy}^V, I_{xy}^B]^T$  la valeur suivante (cf. KRIZHEVSKY et al. 2012) :

$$[\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3][\alpha_1 \lambda_1, \alpha_2 \lambda_2, \alpha_3 \lambda_3]^T \quad (2.7)$$

Ici,  $p_i$  et  $\lambda_i$  sont les  $i$ -ème vecteur et valeur propres de la matrice de covariance  $3 \times 3$  des valeurs des pixels RVB respectivement,  $\alpha_i$  est la variable aléatoire susmentionnée. Cette technique leur permet d'approximativement simuler une propriété des images naturelles d'avoir une identité d'objet invariante aux changements d'intensité et de couleur de l'illumination. Cela leur permet de réduire l'erreur top-1 d'un peu plus que 1%, ce qui est non négligeable, l'augmentation des données deviendra une technique courante dans l'utilisation des CNNs par la suite.

- AlexNet utilise également la technique du dropout (cf. figure 2.11) afin de réduire le phénomène de sur-apprentissage (HINTON et al. 2012). Ils utilisent une probabilité de 0.5 de désactiver les neurones (càd que la sortie des neurones sera de 0 avec une probabilité de 0.5) et appliquent le dropout sur les deux premières couches entièrement connectées, ce qui réduit le



encore plus en avant. Les différences majeures avec AlexNet sont les suivantes : la couche LRN a été supprimée car jugée avoir un impact limité, et de plus VGG utilise des noyaux de convolution de taille  $3 \times 3$  plutôt que  $5 \times 5$ . Le fait d'avoir des noyaux plus petits permet de garder un champ réceptif similaire tout en ayant plus de variations non linéaires, et en réduisant le nombre de paramètres. Son architecture pour la version VGG-16 est disponible figure 2.22.

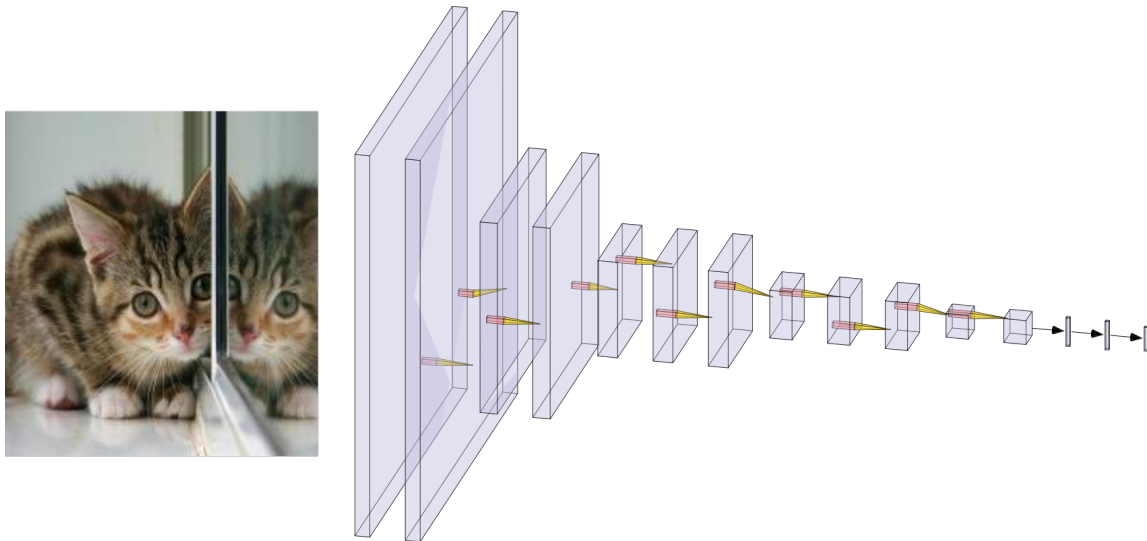


FIGURE 2.22 – Architecture du CNN VGG16.

- La couche d'entrée avec une image de taille  $224 \times 224$  pixels (RVB donc 3 canaux), en utilisant des *crops* (recadrage d'image) au centre si besoin pour avoir la même taille d'image en entrée quelle que soit l'image originale.
- Plusieurs couches de convolution avec des fonctions d'activation ReLU (cf. section 2.2), la profondeur / nombre de filtres double à travers chaque pile de couches tandis que la taille des cartes des caractéristiques est divisée par 4, notamment avec les couches de max-pooling appliquées.
- Trois couches entièrement connectées avec également la couche de softmax pour la classification.

Un des avantages principaux du réseau, en dépit de son nombre important de paramètres, est donc la simplicité de son architecture.

#### 2.3.3.4 ResNet

Le réseau *ResNet* ou encore *Residual Network* provient du papier (HE et al. 2016) et utilise des connexions résiduelles ce qui permet d'avoir un réseau encore plus

profond que ceux présentés précédemment (SZEGEDY et al. 2017). Ce réseau est le premier DNN du concours ImageNet à avoir surpassé la précision d'un humain avec une erreur top-5 inférieure à 5%.

Ce réseau a pour but de pallier le problème principal de VGG, à savoir qu'à force d'avoir un réseau de plus en plus profond, la capacité de généralisation commence à diminuer, notamment à cause du problème du gradient qui disparaît (phénomène du *vanishing gradient*) lors de la rétropropagation du gradient. ResNet considère donc un module de raccourci utilisant la fonction identité afin de sauter des couches (cf. figure 2.23). Ainsi, partant du principe que l'identité  $f(x) = x$  est facile à apprendre pour le réseau, le but est de contourner l'entrée de la première couche du modèle pour être la sortie de la dernière couche, et de prédire la fonction que le réseau était en train d'apprendre avant l'entrée :  $f(x) + x = h(x)$  avec au départ,  $f(x) = 0$ . Ainsi, les gradients peuvent passer par les connexions sautées des dernières couches aux premières lors de la rétropropagation, et ne pas disparaître : les Residual Networks (ResNets) peuvent donc être beaucoup plus profonds et on a ainsi ResNet-18, ResNet-34, ResNet-50, ResNet-101 et même ResNet-152, ce qui est énorme comparé au plus grand VGG comprenant 19 couches.

Il existe ainsi deux versions du réseau : la version sans goulot d'étranglement (*bottleneck*) et celle avec ; afin de réduire le nombre de paramètres ResNet utilise des filtres de taille  $1 \times 1$ , ainsi on passe de deux couches dans le module de raccourci à trois. Les filtres  $1 \times 1$  réduisent le nombre de paramètres avant de les restaurer. ResNet-18 et ResNet-34 n'utilisent pas de goulot d'étranglement, tandis que ResNet-50, 101 et 152 l'utilisent. La version qui a gagné le concours ImageNet est ainsi ResNet-152 avec 60 millions de paramètres pour environ 1% d'erreur top-5 en moins comparé à ResNet-50 qui a lui environ 24 millions de paramètres.

### 2.3.3.5 Après les CNNs

Depuis le succès des transformeurs dans le domaine du traitement du langage naturel, plusieurs chercheurs ont tenté de les adapter au domaine de la vision par ordinateur. Ainsi sont nés les transformeurs ViT qui obtiennent de bonnes performances lorsque préentraînées avec beaucoup de données. Nous les avons utilisés notamment pour l'extraction de caractéristiques en combinaison avec l'apprentissage actif et les décrivons ci-après.

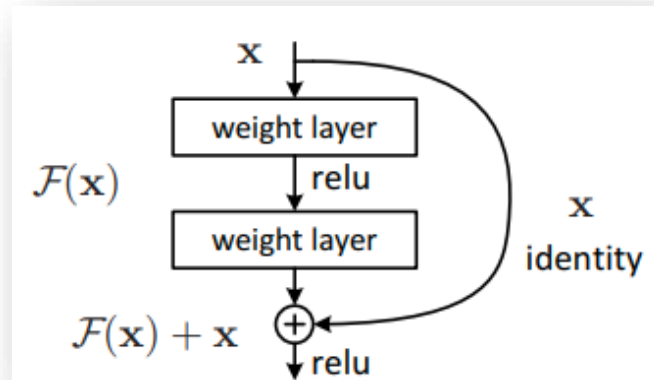


FIGURE 2.23 – Un bloc de Residual Network du papier original. On remarque que l'on obtient  $\mathcal{F}(x) + x$  à la fin grâce à la connexion directe de l'identité. (crédits : HE et al. 2016)

## 2.4 Transformeurs pour la vision (ViT)

Les transformeurs n'ont pas d'invariance aux translations ou de champs réceptifs restreints localement, contrairement aux CNNs, mais ils sont plus robustes aux permutations spatiales (NASEER et al. 2021); ils nécessitent toutefois des séquences plutôt que des grilles. Ainsi, afin de pouvoir utiliser les transformeurs pour la vision, la solution est de transformer les données non séquentielles que sont les images en séquences. Nous verrons cela plus en détail dans les prochaines sous-sections, en commençant par les concepts d'encodeur et décodeur, nécessaires pour bien comprendre le fonctionnement des transformeurs.

### 2.4.1 Encodeur et décodeur

**Encodeur.** Il sert à encoder une séquence source, par exemple un mot ou une phrase en français, vers une représentation en mémoire. Un encodeur est composé de deux blocs :

- Un bloc d'attention qui compare des positions de la séquence source afin d'estimer leurs importances relatives grâce à des poids.
- Un bloc dit *feed-forward* qui sert à apprendre la représentation en mémoire. L'entrée d'un encodeur correspond aux données originales ou à la sortie d'un encodeur précédent.

**Décodeur.** Il sert à comparer ce qu'il y a en mémoire avec une séquence cible, qui pourrait être une phrase correspondante en finnois afin d'engranger de l'expérience pour le futur. Un décodeur est composé de trois blocs :

- Un bloc d'auto-attention pour la mémoire (la sortie d'un encodeur), du fait que le décodeur a accès à toute la séquence mémoire d'un seul coup, ce qui pourrait l'amener à casser la causalité en considérant que les étapes futures contribuent aux étapes passées. Afin de régler ce problème, un masque est utilisé pour éviter les fuites dans le passé.
- Un bloc d'attention à cible mémoire qui compare la mémoire à la cible dans le but d'obtenir les importances relatives.
- Un bloc feed-forward qui apprend à gagner de l'expérience pour le futur.

## 2.4.2 Le mécanisme d'attention

Le mécanisme d'attention est une des plus importantes contributions des transformeurs, l'attention sert à relier une séquence à une autre afin de déterminer l'importance relative entre chaque paire d'éléments. Il est possible d'avoir de l'auto-attention, si la deuxième séquence est la même que la première, ce qui est utile notamment pour la structure syntaxique et contextuelle au sein d'une même phrase. L'algorithme de l'attention fonctionne ainsi :

- Un vecteur de requête est dérivé de la première séquence.
- Une paire clé - valeur de vecteurs sont dérivés de la deuxième séquence.
- La similarité entre la clé et la requête est mesurée et utilisée comme poids d'attention.
- Les attentions sont ensuite appliquées au vecteur de valeur, afin d'obtenir en sortie un vecteur mémoire.

Il est également possible d'obtenir une attention à têtes multiples (*multi-head attention*) en séparant les clé / valeur / requête en plusieurs vecteurs plus petits travaillant en parallèle. Cela permet une plus grande performance qu'une seule tête d'attention, en ayant une attention locale plus performante sur les différentes têtes.

### 2.4.3 Architecture originale du transformeur

L'architecture se compose également de couches de normalisation afin de faciliter l'apprentissage du réseau, et des blocs résiduels comme dans ResNet sont utilisés (cf. figure 2.24).

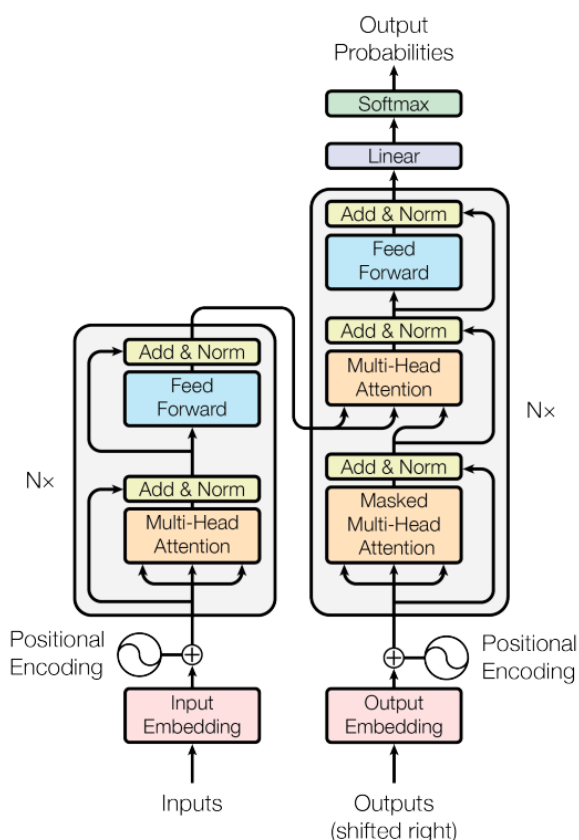


FIGURE 2.24 – Architecture originale du transformeur. (crédits : VASWANI et al. 2017)

### 2.4.4 Les transformeurs pour la vision par ordinateur

Afin de transformer les images en données séquentielles qui peuvent être utilisées par un transformeur, plusieurs étapes sont nécessaires. Cette architecture est appelée Vision Transformer (ViT) dans la littérature (DOSOVITSKIY et al. 2020), et fonctionne de cette façon : on commence par séparer les images en petits bouts que l'on aplatit, avant de produire des *embeddings* linéaires de dimension inférieure depuis ces bouts d'images aplatits. Des *embeddings* positionnels sont ensuite ajoutés, avant de donner cette séquence en entrée à un encodeur de transformeur classique ; le modèle est ensuite préentraîné avec des étiquettes d'images



sur un grand jeu de données de façon complètement supervisée, avant d'être *fine-tuné* sur le jeu de données cible (cf. figure 2.25). ViT nécessite un préentraînement sur un énorme nombre de données pour rivaliser avec les meilleurs CNNs, ce n'est donc pas une solution à utiliser dans tous les cas. Cela peut tout de même être utile dans le cadre de l'apprentissage actif, notamment pour l'extraction de bonnes cartes des caractéristiques du jeu cible en entraînant le modèle sur un autre jeu de données au préalable.

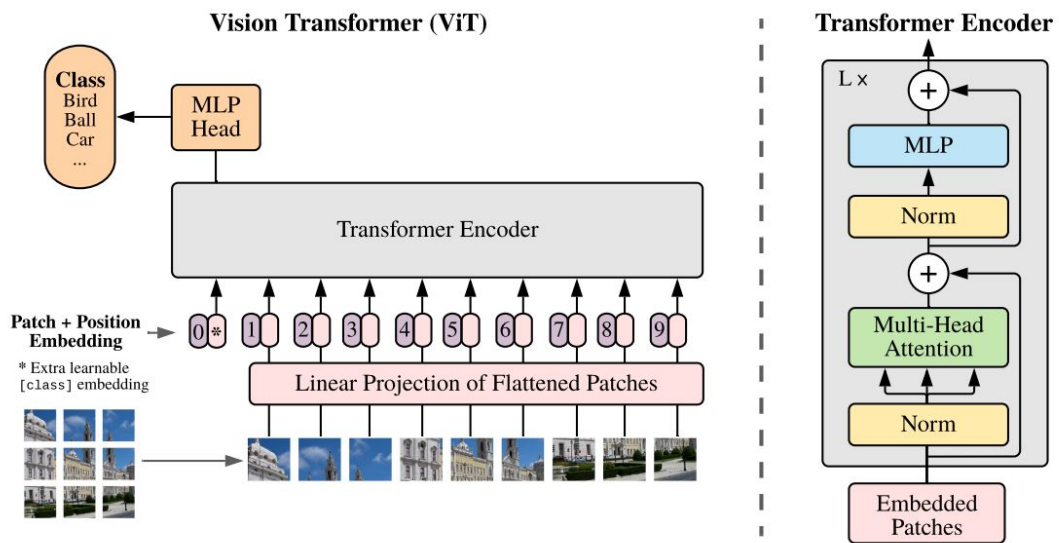


FIGURE 2.25 – Architecture de ViT. (crédits : DOSOVITSKIY et al. 2020)

## APPRENTISSAGE FRUGAL

### *Résumé chapitre*

*Dans ce chapitre, nous présentons un aperçu général de l'apprentissage frugal et ses méthodes : l'apprentissage auto-supervisé, semi-supervisé, actif ou encore few-shot. Nous expliquons pourquoi nous avons choisi l'apprentissage actif profond pour résoudre notre problématique de thèse, avant de présenter l'état de l'art de ce dernier avec des données images. De plus, nous portons une attention particulière aux méthodes d'apprentissage actif profond basées sur une base de données non étiquetées où le but est de sélectionner les instances les plus informatives de cette base. Ces méthodes sont généralement focalisées sur la diversité des données, l'incertitude, ou bien une combinaison des deux. Par ailleurs, les méthodes génératives à base de réseaux adverses génératifs (GANs) ou auto-encodeurs, consistant à générer des données images pertinentes pour l'apprentissage supervisé sont également étudiées. Finalement, nous prenons soin de positionner nos différents travaux dans l'état de l'art en expliquant le cheminement qui nous a conduit à privilégier une méthode avec base de données initialement puis sur la synthèse de données.*

## Sommaire

|       |   |    |
|-------|---|----|
| 3.1   | Apprentissage automatique à partir de peu de données . . . . .            | 46 |
| 3.1.1 | Apprentissage <i>few-shot</i> . . . . .                                   | 47 |
| 3.1.2 | Apprentissage semi-supervisé et auto-supervisé . . . . .                  | 48 |
| 3.1.3 | Introduction à l'apprentissage actif . . . . .                            | 49 |
| 3.1.4 | Premières méthodes en apprentissage actif . . . . .                       | 52 |
| 3.2   | L'apprentissage actif profond pour l'image . . . . .                      | 59 |
| 3.2.1 | Combiner apprentissage actif et apprentissage profond . . . . .           | 61 |
| 3.2.2 | Méthodes génératives d'apprentissage actif . . . . .                      | 64 |
| 3.2.3 | Méthodes par diversité sur une base de données . . . . .                  | 68 |
| 3.2.4 | Méthodes par incertitude sur une base de données . . . . .                | 72 |
| 3.2.5 | Méthodes hybrides sur une base de données . . . . .                       | 74 |
| 3.3   | Positionnement des travaux et contributions . . . . .                     | 75 |
| 3.3.1 | Diversité, représentativité et incertitude des données . . . . .          | 75 |
| 3.3.2 | Apprentissage par renforcement pour la pondération des critères . . . . . | 75 |
| 3.3.3 | Exemples virtuels informatifs pour l'apprentissage actif . . . . .        | 76 |
| 3.4   | Bases de données et métriques d'évaluation . . . . .                      | 76 |
| 3.4.1 | MNIST . . . . .   | 76 |
| 3.4.2 | CIFAR-10 et CIFAR-100 . . . . .   | 77 |
| 3.4.3 | Object-DOTA . . . . .   | 77 |
| 3.4.4 | Données satellitaires en détection de changements . . . . .               | 78 |
| 3.4.5 | Métriques utilisées . . . . .   | 79 |

### 3.1 Apprentissage automatique à partir de peu de données

Le problème principal dans l'entreprise est d'avoir une multitude de capteurs fournissant des données images en grande quantité, qui ne sont cependant pas annotées. L'apprentissage frugal peut prendre plusieurs formes : un faible nombre de données disponibles, un grand nombre de données disponibles mais avec peu d'annotations, des données qui arrivent au fur et à mesure par petites quantités... La problématique d'apprentissage frugal que nous cherchons à résoudre est plutôt celle du grand nombre de données disponibles mais avec des annotations manquantes et qui ont un coût en temps et en moyens humains élevé.

Avec l'apprentissage actif, étant donné un certain nombre de données non labellisées, le but sera de sélectionner les données qui devraient être annotées

afin d'apprendre un modèle performant avec le moins de données possible. Les décisions sur les données à labelliser en premier sont prises itérativement et adaptativement. C'est un cas spécifique de l'apprentissage semi-supervisé, où on a une faible quantité de données labellisées avec une grosse quantité de données non labellisées, ce qui correspond bien à notre motivation première. Cependant, d'autres approches frugales existent et elles ont donc été étudiées afin de déterminer si elles pouvaient convenir à notre application cible.

### 3.1.1 Apprentissage *few-shot*

Le but de l'apprentissage *few-shot* est de reconnaître des classes, à partir d'un très faible nombre d'exemples labellisés. En classification d'images, on va donner quelques exemples d'images de classes différentes de façon répétée afin d'entraîner le classifieur à attribuer les classes correctes à partir d'un faible nombre de données annotées. Cette procédure d'évaluation classique est nommée la classification *n-way, k-shot* (cf. figure 3.1).



FIGURE 3.1 – Illustration de l'apprentissage *few-shot*. Ici les tâches sont de la classification 3-way, 2-shot où le jeu de support a trois classes différentes avec deux exemples pour chacune. La performance est évaluée sur le jeu cible grâce au jeu support associé afin d'entraîner le modèle. Il n'y a pas de classe identique d'une tâche d'entraînement à l'autre ou avec la tâche de test : on a les classes {Fox, Falco, Toad}, {Marth, Kirby, Mario} en entraînement et {Falcon, Yoshi, Link} en test. L'algorithme doit apprendre à classifier des images plutôt qu'un ensemble de données en particulier.

L'apprentissage few-shot ne se limite pas aux données non annotées comme source d'information, mais utilise des connaissances préalables telles que des données supervisées d'autres domaines, des modèles préentraînés, etc. (Y. WANG et YAO 2019)

Il peut être utilisé pour faire de l'apprentissage frugal, avec quelques exemples annotés d'une classe cible, et en utilisant les images d'autres classes comme connaissance préalable ainsi qu'un modèle préentraîné sur un autre jeu de données, à la manière de l'apprentissage par transfert.

### 3.1.2 Apprentissage semi-supervisé et auto-supervisé

Les méthodes qui vont tirer parti d'un ensemble de données étiquetées et non étiquetées sont dites semi-supervisées. On peut considérer cela comme un ensemble qui inclut l'apprentissage actif, mais cela reste un ensemble de techniques en dehors de ce dernier et donc un sujet de recherche connexe, l'apprentissage actif restant un sujet à part entière.

L'apprentissage auto-supervisé concerne les méthodes qui n'ont pas besoin d'étiquettes pour apprendre : le modèle utilise ses propres prédictions comme labels. Ainsi, la supervision provient des données en elle-mêmes, notamment grâce à leur structure sous-jacente. Ces méthodes sont surtout efficaces dans le domaine du traitement du langage naturel, en entraînant des modèles tels que BERT ou RoBERTa sur de grands jeux de données non étiquetés avant d'utiliser ces modèles sur d'autres tâches. Il est cependant difficile d'utiliser l'apprentissage auto-supervisé dans le domaine de la vision, notamment à cause de la difficulté à représenter l'incertitude des prédictions pour les images. Par exemple, l'incertitude est compliquée à représenter lorsque la tâche consiste à prédire les images manquantes dans une vidéo, car il y a trop de possibilités. Certains papiers comme (K. WANG et al. 2018) combinent l'apprentissage auto-supervisé avec l'apprentissage actif, en assignant des pseudo-labels aux exemples jugés faciles et en demandant à l'oracle d'annoter les exemples jugés difficiles.

L'apprentissage auto et semi-supervisé peuvent donc répondre à notre problématique de départ de tirer parti d'un grand nombre de données non étiquetées, cependant nous avons privilégié l'apprentissage actif qui permet à des utilisateurs humains (notamment les ingénieurs travaillant sur des problématiques utilisant l'apprentissage profond) d'annoter plus efficacement.

### 3.1.3 Introduction à l'apprentissage actif

L'apprentissage actif répond le mieux à notre problématique de limiter le nombre de données à étiqueter tout en gardant des performances satisfaisantes, l'état de l'art se concentre donc dessus et nous présenterons en premier lieu les différents scénarios en apprentissage actif. L'ouvrage de référence pour les premières méthodes ainsi que les définitions précises de ces scénarios est la revue de littérature (SETTLES 2009). Ces scénarios considèrent le cas courant où les requêtes sont des données non étiquetées qui doivent être annotées par un oracle, mais il existe des cas où des *sacs* de données qui doivent être étiquetés sont considérés plutôt qu'une donnée.

Il est ainsi expliqué dans cette revue de littérature qu'un sac contenant uniquement des instances négatives sera considéré négatif et ne sera pas utilisé pour l'entraînement, mais un sac avec au moins une instance de donnée positive sera considéré positif et sera utilisée pour l'entraînement d'un modèle, même si ce sac contient également des instances négatives par ailleurs. Un intérêt possible à ces algorithmes, que nous aborderons dans les perspectives lors du dernier chapitre, serait d'utiliser nos algorithmes d'apprentissage actif pour la tâche de classification d'images, et de les adapter à une tâche voisine qui est la détection d'objets, en considérant chaque image contenant plusieurs objets comme un sac, et ainsi choisir d'annoter une image si elle contient au moins un objet sélectionné par notre algorithme d'apprentissage actif pour la classification d'images.

Cependant, dans le cas le plus courant, les trois catégories sus-mentionnées sont la génération d'échantillons, l'échantillonnage de flux de données et enfin l'apprentissage actif sur une base de données. Nous verrons ces scénarios plus en détail dans les sous-sections qui suivent, et la [figure 3.2](#) illustre leurs fonctionnements respectifs.

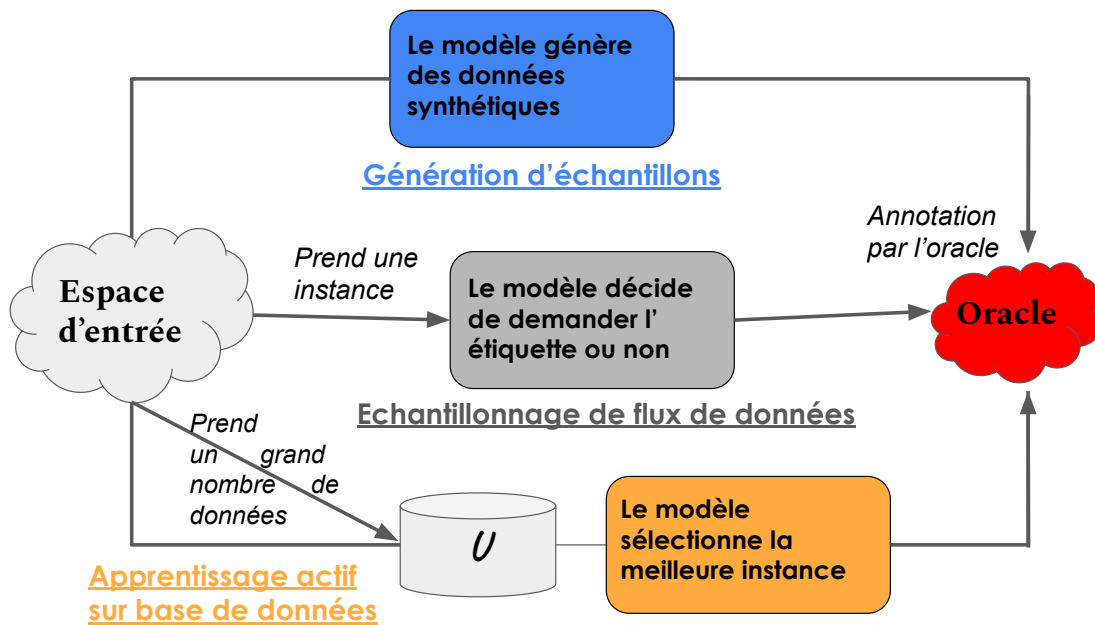


FIGURE 3.2 – Les trois scénarios d'apprentissage actif. (SETTLES 2009)

### 3.1.3.1 Génération d'échantillons

D'après Burr Settles dans son ouvrage de référence, ce scénario est l'un des premiers en apprentissage actif, il a en effet été étudié notamment dans (ANGLUIN 1988). En plus de pouvoir demander à l'oracle d'annoter des données présentes dans l'espace d'entrée, le modèle d'apprentissage actif peut également générer lui-même des données et faire une requête à l'oracle afin que ce dernier les annote. Ce scénario a été utilisé plusieurs fois, ainsi (ANGLUIN 2001) a démontré son efficacité pour des problèmes se déroulant dans des espaces finis de manière théorique.

Ce scénario est encore utilisé dans la littérature contemporaine, avec néanmoins une évolution dans la manière de générer les données : les Generative Adversarial Networks (GANs) ont ainsi été utilisés avec un succès mitigé en apprentissage actif. Malgré l'évolution des techniques, un problème récurrent de ce scénario persiste actuellement comme dans les premières méthodes : la difficulté pour un oracle humain d'annoter des données générées artificiellement qui sont parfois difficiles à reconnaître. Ainsi, (BAUM et LANG 1992) ont utilisé le scénario "apprentissage par requête d'adhésion" avec des oracles humains afin d'entraîner un réseau de neurones dans le but de classer des caractères manuscrits. Cependant, beaucoup des images générées afin d'être annotées contenaient des symboles non reconnaissables par l'être humain, sans signification réelle. Un problème similaire a été rencontré par les auteurs de GAAL (J.-J. ZHU et BENTO 2017).

Ce scénario a du potentiel si les données générées sont plus réalistes et compréhensibles par l'oracle tout en restant informatives du point de vue de l'apprentissage actif. Nous proposons ainsi un algorithme de génération d'échantillons virtuels dans le dernier chapitre de cette thèse.

### 3.1.3.2 Apprentissage actif en ligne

Ce scénario d'échantillonnage basé sur les flux a notamment été présenté dans (ATLAS et al. 1989, D. COHN et al. 1994). Un des postulats de départ est que ça ne coûte rien d'obtenir une donnée non étiquetée, on peut donc directement le présenter au modèle d'apprentissage actif afin qu'il décide de demander l'étiquette associée ou non. De fait, cette présentation des données se fait généralement donnée par donnée.

Diverses stratégies pour sélectionner les données à étiqueter ont été utilisées avec ce scénario : ainsi, dans le papier (DAGAN et ENGELSON 1995) les auteurs utilisent une mesure d'informativité de sorte à demander les étiquettes des données les plus informatives. Par exemple, en définissant un seuil minimum d'informativité pour sélectionner toutes les données dépassant ce seuil. Des approches récentes existent (ŽLIOBAITĚ et al. 2013; CASTELLANI et al. 2022) mais le domaine d'application ne correspond pas à notre problématique de reconnaissance visuelle dans des données images.

Le fait de présenter les données une par une au modèle était pertinent dans plusieurs applications d'apprentissage actif, cependant ce scénario n'est pas adapté aux techniques modernes d'apprentissage actif qui fonctionnent avec des réseaux de neurones profonds. En effet, l'entraînement de ces réseaux est beaucoup plus rapide et performant avec des lots de données.

### 3.1.3.3 Apprentissage actif hors ligne

L'apprentissage actif avec un échantillonnage par lots est le scénario le plus courant et peut facilement être utilisé dans le cadre d'apprentissage actif profond. Ce scénario considère que l'on a deux ensembles de données : un grand nombre de données non annotées  $\mathcal{U}$  (pour *unlabeled*, non étiqueté) et un ensemble potentiellement vide au départ ou tout du moins réduit :  $\mathcal{L}$  (pour *labeled*, étiqueté). Habituellement, on sélectionne les données à présenter à l'oracle selon une mesure d'informativité, et on peut ainsi choisir de présenter les données une par une ou en prenant des paquets à chaque cycle.

Du fait qu'il s'agisse du scénario le plus courant, la littérature regorge de papiers utilisant ce scénario que ce soit au début de l'apprentissage actif ou actuellement. On a ainsi divers papiers de classification de texte (LEWIS et GALE



1994; TONG et KOLLER 2001; HOI et al. 2006), des papiers d'extraction de l'information (THOMPSON et al. 1999; SETTLES et CRAVEN 2008), de classification d'images (TONG et KOLLER 2001; Cha ZHANG et T. CHEN 2002), recherche d'images par le contenu (FERECATU et al. 2005), etc.

Contrairement à l'échantillonnage par flux qui évalue l'informativité de chaque donnée de façon séquentielle, ce scénario mesure l'informativité de l'ensemble des données afin de choisir (en fonction généralement d'un budget, nombre de cycles, labels disponibles...) un certain nombre de données à étiqueter.

### 3.1.4 Premières méthodes en apprentissage actif

La plupart des premiers algorithmes d'apprentissage actif sont essentiellement des heuristiques (DASGUPTA 2004; SETTLES 2009) définies sur des critères d'incertitude où un apprenant actif échantillonne les données dont les étiquettes sont les moins fiables (LEWIS et GALE 1994) en utilisant différentes mesures d'incertitude telles que la marge et l'entropie. Les stratégies basées sur la marge sélectionnent les données présentant la plus petite différence entre le meilleur et le deuxième meilleur score de prédiction (JOSHI et al. 2009) tandis que les approches basées sur l'entropie échantillonnent les données les plus pertinentes en maximisant l'entropie des scores sous-jacents. Des variantes de ces travaux (LI et GUO 2013) abordent le problème de l'incertitude dans l'interrogation des données<sup>1</sup> en combinant les mesures de densité, de représentativité et d'incertitude afin de sélectionner les instances à étiqueter pour différentes tâches, notamment la reconnaissance des objets et des scènes.

D'autres méthodes s'appuient sur plusieurs modèles complémentaires – formés avec des hypothèses concurrentes – afin de sélectionner les données présentant les désaccords les plus importants, comme la méthode de la requête par votes (SEUNG et al. 1992) ou s'appuient sur la modification prévue du modèle/de l'erreur (SETTLES et al. 2007; Nicholas ROY et MCCALLUM 2001; HOULSBY et al. 2011) en sélectionnant et en étiquetant les instances qui réduisent le plus l'erreur par rapport aux modèles d'entraînement utilisés (SETTLES et CRAVEN 2008; X. ZHU et al. 2003). La minimisation de l'erreur de généralisation de ces modèles peut également être réalisée à l'aide de la réduction de la variance prévue; cependant, cette dernière est généralement coûteuse en calcul par rapport aux mesures d'échantillonnage d'incertitude susmentionnées (D. A. COHN et al. 1996), et son optimisation ne peut pas toujours être réalisée avec une solution analytique. Ces premières méthodes sont présentées plus en détail dans les sections suivantes.

---

1. ne tenant pas compte des informations sur la grande quantité d'instances non étiquetées, ainsi que la propension à interroger les valeurs aberrantes.

### 3.1.4.1 Echantillonnage par incertitude

La méthode d'échantillonnage par incertitude (*uncertainty sampling*) est très simple et très utilisée (LEWIS et GALE 1994), c'est ainsi une *baseline* de référence dans la littérature, aux côtés de l'échantillonnage aléatoire (*random sampling*). Le modèle demande ici à l'oracle d'annoter les données pour lesquelles il est le plus incertain des étiquettes, ce qui est parfait pour les modèles d'apprentissage probabilistes qui peuvent utiliser leurs prédictions et les probabilités associées comme mesure d'incertitude. Les modèles d'apprentissage profonds peuvent ainsi utiliser leur couche de *softmax* pour obtenir les probabilités associées à chaque classe, même si la précision du *softmax* des réseaux profonds comme mesure d'incertitude est contestée (GAL et al. 2016).

Dans le cas de classification binaire, les papiers (LEWIS et GALE 1994; LEWIS et CATLETT 1994) vont choisir les données avec les probabilités les plus proches de 0.5, ce qui correspondrait en classification multi-classes à la mesure dite de moindre confiance. Les mesures d'incertitude les plus couramment utilisées dans la méthode d'échantillonnage par incertitude sont : la moindre confiance, l'entropie et l'échantillonnage avec marges que nous présentons ci-après.

**Moindre confiance.** Cette mesure compare les probabilités maximum sur chacune des données (et donc les probabilités des labels prédits par le modèle), et où l'on donne à annoter les données ayant ces probabilités les plus basses. Formellement, cette mesure est :

$$x_{LC}^* = \arg \max_x \{1 - P_\theta(\hat{y}|x)\}, \quad (3.1)$$

où  $\hat{y} = \arg \max_y P_\theta(y|x)$ , ce qui correspond au label de la classe avec la plus haute probabilité à posteriori avec le modèle  $\theta$ . Cf. [tableau 3.1](#) pour un exemple sur un jeu de données avec les classes *chat*, *chien*, *pingouin*.

**Entropie.** La mesure d'incertitude d'entropie est très utilisée et en voici la définition formelle :

$$x_H^* = \arg \max_x \left\{ - \sum_i^{nc} P_\theta(y_i|x) \log P_\theta(y_i|x) \right\}, \quad (3.2)$$

où  $y_i$  boucle sur toutes les étiquettes possibles ( $nc$  le nombre de classes du jeu de données). Ici, plus l'entropie d'un échantillon sera grande, plus son incertitude sera grande : on sélectionnera alors les  $K$  (la taille d'un paquet à chaque cycle) exemples avec la plus grande entropie. Dans le cas de la classification binaire, cette

TABLE 3.1 – **Exemple incertitude par moindre confiance.** On aura ici pour  $x_1$  :  $1 - 0.9 = 0.1$ , pour  $x_2$  :  $1 - 0.43 = 0.57$  et enfin pour  $x_3$  :  $1 - 0.5 = 0.5$ . De fait, l'image choisie sera donc  $x_2$  qui a obtenu le score le plus élevé, car sa classe avec la plus haute probabilité avait une probabilité plus faible que les deux autres images ( $0.43 < 0.5 < 0.9$ ).

| Image | Chat | Chien | Pingouin |
|-------|------|-------|----------|
| $x_1$ | 0.9  | 0.09  | 0.01     |
| $x_2$ | 0.19 | 0.38  | 0.43     |
| $x_3$ | 0.3  | 0.5   | 0.2      |

mesure est équivalente aux mesures de moindre confiance et d'échantillonnage avec marges présentées respectivement avant et après l'entropie. Dans le cas multi-classes cependant, l'entropie permet de généraliser à des modèles probabilistes telles que les arbres (HWA 2004). Le tableau 3.2 montre l'exemple précédent lorsque l'on utilise l'entropie.

TABLE 3.2 – **Exemple incertitude par entropie.** On aura ici pour  $x_1$  : environ 0.155, pour  $x_2$  : environ 0.454 et enfin pour  $x_3$  : environ 0.447. De fait, l'image choisie sera  $x_2$ .

| Image | Chat | Chien | Pingouin |
|-------|------|-------|----------|
| $x_1$ | 0.9  | 0.09  | 0.01     |
| $x_2$ | 0.19 | 0.38  | 0.43     |
| $x_3$ | 0.3  | 0.5   | 0.2      |

**Echantillonnage avec marges.** La mesure d'incertitude d'échantillonnage avec marges vise à corriger un des défauts de la stratégie de moindre confiance : seule l'information sur le label le plus probable est considérée, et les informations sur le reste de la distribution des étiquettes ne sont donc pas prises en compte. Pour l'échantillonnage avec marges cependant, les deux classes les plus probables sont considérées : ainsi les données avec une différence importante entre leur label prédit et le second label le plus probable, sont considérées comme faciles à classifier et ne sont donc pas présentées à l'oracle. Par contre, les labels où le classifieur accorde presque la même importance (avec des probabilités très proches) au label le plus probable et au second, sont considérés difficiles et très incertains. Cependant, une grande partie de la distribution est tout de même ignorée, contrairement à

la mesure d'incertitude par entropie. Le papier (SCHEFFER et al. 2001) présente l'échantillonnage avec marges dont voici l'équation associée :

$$x_M^* = \arg \min_x (P_\theta(\hat{y}_1|x) - P_\theta(\hat{y}_2|x)), \quad (3.3)$$

où  $\hat{y}_1$  et  $\hat{y}_2$  sont les étiquettes des 1ère et 2ème classe les plus probables d'après le modèle. Le tableau 3.3 montre finalement les résultats de l'échantillonnage avec marges.

TABLE 3.3 – **Exemple incertitude échantillonnage avec marges.** On aura ici pour  $x_1 : 0.81$ , pour  $x_2 : 0.05$  et enfin pour  $x_3 : 0.2$ . De fait, l'image choisie sera  $x_2$ .

| Image | Chat | Chien | Pingouin |
|-------|------|-------|----------|
| $x_1$ | 0.9  | 0.09  | 0.01     |
| $x_2$ | 0.19 | 0.38  | 0.43     |
| $x_3$ | 0.3  | 0.5   | 0.2      |

**Comparaison de ces mesures d'incertitude.** Divers articles ont étudié les différences empiriques entre les différentes mesures d'incertitudes utilisées dans le cadre de l'échantillonnage par incertitude. Ainsi, (SETTLES et CRAVEN 2008) ont trouvé qu'il n'existe pas de mesure supérieure dans toutes les applications, mais que les différentes stratégies ont en général de meilleurs résultats que la *baseline* d'échantillonnage aléatoire. (SETTLES 2009) explique que l'entropie semble plus appropriée si la fonction objectif essaie de minimiser une *log-loss*, tandis que les autres sont plus appropriées lorsque le but est de réduire l'erreur de classification, en aidant le modèle à mieux discriminer entre des classes précises, et tout particulièrement la mesure *margin*. La figure 3.3 illustre les différences entre les différentes mesures d'incertitude (SETTLES 2009).

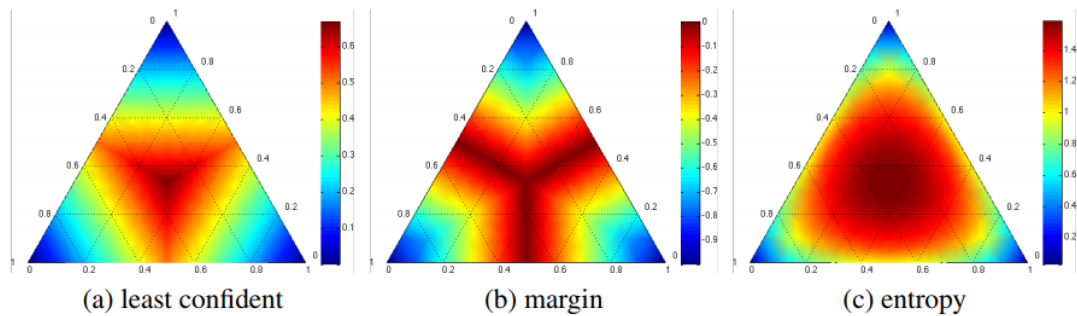


FIGURE 3.3 – **Carte de chaleur des différentes mesures d’incertitude.** Dans le contexte d’un problème de classification à trois classes, les coins représentent là où le modèle assigne une très haute probabilité pour une classe. La zone considérée la plus informative est au centre. (crédits : SETTLES 2009)

La stratégie d’échantillonnage par incertitude se prête bien aux modèles probabilistes, mais il est possible d’utiliser des classifieurs non probabilistes, par exemple (TONG et KOLLER 2001) utilisent des SVMs et sélectionnent les données les plus proches de la frontière de décision.

### 3.1.4.2 Requête par votes

Cette méthode considère un comité  $\mathcal{C} = \{\theta^{(1)}, \dots, \theta^{(C)}\}$  de modèles qui sont entraînés sur l’ensemble étiqueté  $\mathcal{L}$  tout en représentant des hypothèses concurrentes (SETTLES 2009; SEUNG et al. 1992). Ainsi, chaque modèle membre du comité va pouvoir voter pour choisir l’étiquette des différentes données, et on considère les données avec le plus de désaccords comme les plus informatives.

Cela nécessite donc de pouvoir obtenir des modèles qui représentent différentes régions de l’espace d’entrée mais également de pouvoir mesurer les désaccords entre ces différents modèles. Dans la littérature, on retrouve ainsi (ABE 1998) qui utilisent le *boosting* et *bagging* (des méthodes d’apprentissage par ensembles, FREUND et SCHAPIRE 1997) sous la forme *query-by-boosting* et *query-by-bagging*, ou bien (BREIMAN 1996) qui utilise seulement le *bagging* afin de construire des comités de modèles représentant différentes régions de l’espace d’entrée. Un petit nombre de modèles comme comité peut se montrer performant en pratique (SETTLES et CRAVEN 2008).

En ce qui concerne la mesure du niveau de désaccord, plusieurs approches sont possibles comme présenté ci-après.

**Entropie de vote.** L'entropie de vote utilise un comité de différents classifieurs qui vont donner des prédictions potentiellement différentes, ce qui permet d'obtenir une distribution de probabilité pour chacune des données : la distribution des étiquettes lorsque l'on prend la prédiction d'un classifieur choisi aléatoirement (cf. figure 3.4).

$$x_{VE}^* = \arg \max_x \left\{ - \sum_i \frac{V(y_i(x))}{C} \log \frac{V(y_i(x))}{C} \right\}, \quad (3.4)$$

où  $y_i(x)$  boucle sur toutes les étiquettes possibles ( $nc$  le nombre de classes du jeu de données), et  $V(y_i(x))$  correspond au nombre de votes qu'une étiquette reçoit des prédictions des membres du comité, pour chaque donnée  $x$ . Enfin,  $C$  correspond à la taille du comité, et donc au nombre de classifieurs. Cette mesure du niveau de désaccord est l'équivalent pour la méthode de requête par comité de l'entropie utilisée dans l'échantillonnage par incertitude.










|  |   |   |  | <u>Distributions de probabilités associées</u> |          |                 |   |   |   |  |  |  |
|--|---|---|--|--|----------|-----------------|---|---|---|--|--|--|
|  |   |   |  | Classe 0                                       | Classe 1 | Classe 2        |   |   |   |  |  |  |
| <table border="1"> <thead> <tr> <th>Classe 0</th> <th>Classe 1</th> <th>Classe 2</th> </tr> </thead> <tbody> <tr> <td></td> <td></td> <td></td> </tr> </tbody> </table> |   |   |  | Classe 0                                       | Classe 1 | Classe 2        |  |  |  |  |  |  |
| Classe 0   | Classe 1  | Classe 2  |  |  |          |                 |   |   |   |  |  |  |
|   |  |  |  |  |          |                 |   |   |   |  |  |  |
|  |   |   |  | Donnée 1                                       | 2/3      | 1/3             | 0   |   |   |  |  |  |
|  |   |   |  | Donnée 2                                       | 0        | 2/3             | 1/3   |   |   |  |  |  |
|  |   |   |  | Donnée 3                                       | 1/3      | 1/3             | 1/3   |   |   |  |  |  |
|  |   |   |  | Donnée 4                                       | 0        | 0               | 1   |   |   |  |  |  |
|  |   |   |  | Donnée 5                                       | 0        | 1/3             | 2/3   |   |   |  |  |  |
|  |   |   |  | <u>Résultat entropie de vote</u>               |          |                 |   |   |   |  |  |  |
|  |   |   |  | Donnée 1                                       | Donnée 2 | <b>Donnée 3</b> | Donnée 4  | Donnée 5  |   |  |  |  |
|  |   |   |  | 0.6365   | 0.6365   | <b>1.0986</b>   | 0   | 0.6365  |   |  |  |  |

FIGURE 3.4 – Exemple entropie de vote. On a 3 classifieurs, 5 données de trois classes différentes {0, 1, 2}, la première étape consiste à récupérer les prédictions de chaque classifieur sur toutes les données. Ensuite, on calcule la probabilité d'obtenir chacune des classes en choisissant aléatoirement un classifieur. Finalement, on obtient l'entropie de vote de chacune des données et l'on choisit la donnée avec la plus grande entropie.

**Divergence de Kullback-Leibler.** Une autre méthode pour mesurer le niveau de désaccord est la moyenne de la divergence de Kullback-Leibler.

$$x_{KL}^* = \arg \max_x \left\{ \frac{1}{C} \sum_{c=1}^C D(P_{\theta^{(c)}} \| P_C) \right\}, \quad (3.5)$$

où :

$$D(P_{\theta^{(c)}} \| P_C) = \sum_i P_{\theta^{(c)}}(y_i|x) \log \frac{P_{\theta^{(c)}}(y_i|x)}{P_C(y_i|x)}. \quad (3.6)$$

Ici,  $\theta^{(c)}$  représente un modèle particulier du comité, et  $C$  représente le comité entier, ce qui donne  $P_C(y_i|x) = \frac{1}{C} \sum_{c=1}^C P_{\theta^{(c)}}(y_i|x)$  comme la probabilité consensuelle que  $y_i$  est le label correct. Cette mesure basée sur la divergence de Kullback-Leibler (kullback and leibler, 1951) est utilisée pour mesurer les différences entre deux distributions de probabilités. Avec cette mesure, les données les plus informatives seront celles avec la plus grande différence moyenne entre les distributions des labels d'un membre du comité et les labels les plus consensuels.

### 3.1.4.3 Autres méthodes de la littérature

L'échantillonnage par incertitude est une *baseline* classique de l'apprentissage actif que nous avons utilisée dans nos expériences, et nos contributions l'utilisent en partie. Nous avons également utilisé une variante de l'entropie de vote dans nos expériences d'apprentissage par renforcement, ainsi avons-nous présenté en détail ces deux méthodes classiques d'apprentissage actif. Il en existe cependant d'autres (SETTLES 2009), comme le changement de modèle prévu qui va sélectionner les données qui modifieraient le plus le modèle si l'on connaissait leurs étiquettes (SETTLES et al. 2007). On peut également citer la réduction de l'erreur prévue qui vise à estimer à quel point l'erreur de généralisation sera probablement réduite grâce à diverses méthodes d'estimation telles que la minimisation de l'erreur 0/1 attendue ou l'erreur logarithmique attendue (Nicholas ROY et McCALLUM 2001; X. ZHU et al. 2003).

Il y a aussi les méthodes basées sur la réduction de la variance prévue où le but est de réduire l'erreur de généralisation de façon indirecte, en minimisant la variance de sortie, par exemple grâce au ratio d'information de Fisher (T. ZHANG et OLES 2000). Les méthodes de densité pondérée visent à répondre à la problématique des valeurs extrêmes, où certaines données peuvent être proches de la frontière de décision, tout en étant peu représentatives des autres données. Les méthodes précédentes de réduction de l'erreur et variance prévues peuvent éviter ces problèmes mais elles sont particulièrement lentes. Ainsi, (SETTLES et CRAVEN 2008) propose une technique généraliste de densité pondérée qui vise



à résoudre ce problème en temps raisonnable, tout comme (MCCALLUMZY et NIGAMY 1998; XU et al. 2007).

## 3.2 L'apprentissage actif profond pour l'image

Les réseaux de neurones convolutifs (CNNs) peuvent capturer les informations spatiales dans les images et sont devenus état de l'art pour les tâches de classifications d'images, et les solutions d'apprentissage actif sont maintenant développées de manière à être compatibles avec les CNNs : on parle alors d'apprentissage actif profond.

Avec la résurgence des réseaux de neurones profonds, d'autres catégories de solutions d'apprentissage actif sont apparues récemment. Une catégorie de méthodes (par exemple, (SENER et SAVARESE 2017)) redéfinit l'apprentissage actif comme un problème de sélection de *core-sets* (CAMPBELL et BRODERICK 2019) suite à l'observation que les techniques classiques d'apprentissage actif ne sont pas adaptées lorsqu'elles sont appliquées à des réseaux convolutifs profonds par lots. Les *core-sets* sont des sous-ensembles informatifs de données pondérées qui servent d'approximation des données originales (FELDMAN 2020). Le papier (SENER et SAVARESE 2017) propose ainsi une méthode d'apprentissage actif utilisant les *core-sets* plus performante que les *baselines* d'échantillonnage aléatoire, par k-médian et incertitude.

Au lieu d'échantillonner des données réelles, d'autres méthodes d'apprentissage actif considèrent des réseaux adverses génératifs (GAN, DCGAN, etc.) afin d'échantillonner des données synthétiques (J.-J. ZHU et BENTO 2017; MAYER et TIMOFTE 2020). En particulier, l'échantillonnage adverse pour l'apprentissage actif (ASAL, MAYER et TIMOFTE 2020) combine l'échantillonnage par incertitude, la génération et l'appariement d'échantillons adverses afin de synthétiser des instances de données incertaines (sans effectuer une recherche exhaustive sur un ensemble de données non étiquetées), avec une résilience supposée plus élevée au biais d'échantillonnage par rapport aux autres approches d'apprentissage actif génératives.

Cependant, le gain de toutes ces approches synthétiques basées sur les GANs n'a pas été établi de manière formelle par rapport à d'autres stratégies, notamment l'échantillonnage aléatoire et l'entropie maximale/distance minimale, ainsi que l'apprentissage auto-supervisé. Nous décrivons plus en détail ces différentes méthodes dans les sections suivantes. La figure 3.5 illustre l'architecture d'un modèle d'apprentissage actif profond.

Les paramètres du modèle d'apprentissage peuvent être initialisés à partir d'un ensemble d'entraînement étiqueté de départ, généralement choisi aléatoirement.



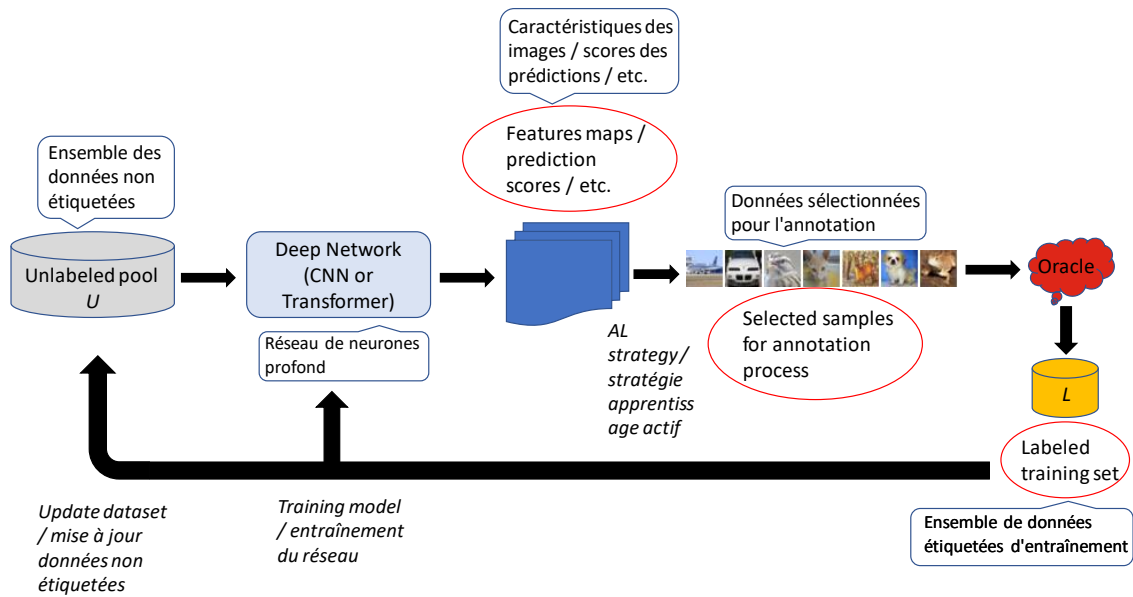


FIGURE 3.5 – **Diagramme de l’apprentissage actif profond.** Le diagramme est similaire à l’apprentissage actif classique, mais le CNN permet d’obtenir les features ou scores de prédiction pour la stratégie d’apprentissage actif. (inspiré par REN et al. 2021)

Dans notre cas, nous n’avons pas besoin d’étiqueter aléatoirement des données pour commencer notre premier algorithme d’apprentissage actif, néanmoins nous devons utiliser le modèle préentraîné sur un autre jeu de données afin d’effectuer une étape d’inférence sur les données pour obtenir les probabilités des prédictions avec la couche de *softmax*. Les échantillons de l’ensemble non étiqueté  $\mathcal{U}$  sont souvent utilisés pour extraire leurs *features* grâce au modèle d’apprentissage profond. La stratégie d’apprentissage actif est ensuite utilisée pour choisir les échantillons à annoter, qui seront alors donnés à l’oracle et ajoutés à l’ensemble des données étiquetées  $\mathcal{L}$ . L’ensemble des données non annotées  $\mathcal{U}$  est ensuite mis à jour en retirant les données déjà sélectionnées, et le modèle d’apprentissage profond est entraîné sur l’ensemble des données étiquetées (puis utilisé sur l’ensemble de test pour évaluer la performance, selon la métrique utilisée). Cette boucle est répétée jusqu’à ce que le budget d’annotation soit épuisé, il existe cependant diverses conditions d’arrêt d’après (REN et al. 2021). Ces critères sont par exemple, un nombre fixe d’itérations maximum, un seuil minimum de précision de classification à atteindre, un nombre minimum d’échantillons étiquetés, une valeur cible attendue de précision, etc. (BUDD et al. 2021 ; FOLMSBEE et al. 2018 ; P. LIU et al. 2016 ; MALDONADO et HARABAGIU 2019 ; SCHRÖDER et NIEKLER 2020)

Dans le papier (BLOODGOOD et VIJAY-SHANKER 2014), les auteurs détaillent les différentes stratégies d’arrêt en apprentissage actif et proposent leur solution

basée sur la prédiction de stabilité, en tirant parti de la capacité d'un modèle bien entraîné à être stable dans ses prédictions. Leur stratégie a été pensée pour de l'apprentissage actif traditionnel, mais est également pertinente pour l'apprentissage actif profond.

### 3.2.1 Combiner apprentissage actif et apprentissage profond

On a vu précédemment que l'apprentissage actif permettait de résoudre certaines problématiques de l'apprentissage profond qui se posent notamment en entreprise, où l'accès à une quantité importante de données non étiquetées est courant, mais où la phase d'annotation est coûteuse en moyens humains. Nous détaillerons plus longuement l'intérêt de combiner l'apprentissage actif et profond dans un premier temps, avant d'explicitier la manière de le faire dans un second temps.

#### 3.2.1.1 Pourquoi les combiner ?

Nous avons déjà évoqué les très bonnes performances dans de nombreux domaines de l'apprentissage profond, qui peut avoir cependant divers désavantages :

1. L'explicabilité des réseaux de neurones profonds n'est pas aussi bonne que des modèles d'apprentissage automatique tels que les arbres de décision.
2. Les réseaux de neurones profonds nécessitent une grande quantité de données afin d'être plus performantes que les autres techniques d'apprentissage automatique, et notamment de données annotées.
3. Il est coûteux en moyens matériels, notamment GPU (cartes graphiques mais pas que) permettant d'entraîner des modèles complexes.
4. D'autres points qui ne concernent pas directement nos travaux, comme l'oubli catastrophique lorsque l'on apprend un réseau déjà entraîné sur une tâche différente. Le domaine de l'apprentissage continu a pour but de résoudre ce problème (DELANGE et al. 2021).

Nos travaux n'ont pas pour but de répondre au premier point sur l'explicabilité des réseaux de neurones profonds, cependant ils se portent sur les points 2 et potentiellement 3. En effet, nos études préliminaires qui comparaient les différentes méthodes d'apprentissage frugal nous ont confirmé que l'apprentissage actif permettait de répondre à notre problématique : beaucoup de données non étiquetées à disposition et leur annotation est coûteuse.

L'apprentissage actif permet de réduire le coût d'annotation grâce à des stratégies pour sélectionner les données les plus informatives pour un modèle d'apprentissage automatique, afin de maximiser les performances pour un budget donné,

ou bien d'atteindre un seuil de performances cible (par exemple la précision d'un classifieur) avec le moins de données étiquetées possible.

Le potentiel de l'apprentissage actif combiné à l'apprentissage profond a été remarqué, et de nombreux chercheurs ont entrepris de faire de l'apprentissage actif profond. Cependant, les techniques utilisées habituellement dans l'apprentissage actif ne fonctionnent pas forcément aussi bien qu'escompté lorsque l'on tente de les combiner avec l'apprentissage profond, ce que nous verrons dans la sous-section suivante.

### 3.2.1.2 Comment les combiner ?

La revue de littérature sur l'apprentissage actif profond (REN et al. 2021) a recensé plusieurs difficultés pour combiner l'apprentissage actif avec l'apprentissage profond, ainsi que les solutions adoptées par la recherche pour tenter de pallier ces problèmes.

- **Comment évaluer l'incertitude avec l'apprentissage profond ?** On a vu dans la section 3.1.4.1 que l'incertitude occupe une place importante dans les stratégies d'apprentissage actif. Il est possible dans le cadre de la classification d'utiliser la couche de *softmax* afin d'obtenir une probabilité sur les différentes classes, mais des travaux ont montré que les réseaux se montraient trop confiants. Ainsi, dans le papier (K. WANG et al. 2016), ils mettent en exergue les imprécisions de la sortie après softmax comme mesure de confiance, ce qui conduit à de mauvais résultats pour les *baselines* d'échantillonnage par incertitude en comparaison à l'échantillonnage aléatoire, contrairement aux méthodes d'apprentissage actif classiques (D. WANG et SHANG 2014).

L'apprentissage profond bayésien a été utilisé par (GAL et GHARAMANI 2015) afin d'obtenir des algorithmes utilisant des CNNs plus robustes lorsque utilisés avec un faible nombre de données. Comme les CNNs ont tendance au sur-apprentissage, les auteurs proposent d'utiliser une distribution de probabilité sur les noyaux des CNNs grâce à des distributions variationnelles de Bernoulli pour approximer la probabilité conditionnelle de leur modèle. Cette méthode a donc été utilisée dans la recherche en apprentissage actif afin de diminuer le problème des modèles d'apprentissage profonds trop confiants dans leurs prédictions de sortie, afin d'augmenter les performances des approches basées sur l'incertitude. Ainsi, (GAL et al. 2017; KIRSCH et al. 2019; POP et FULOP 2018; TRAN et al. 2019) ont appliqué la technique d'apprentissage profond bayésien susmentionnée à l'apprentissage actif avec un certain succès, et l'apprentissage actif avec des réseaux profonds bayésiens est utilisé par plusieurs papiers.

- **Méthodes d'apprentissage actif habituelles pas adaptées à l'apprentissage profond.** L'apprentissage actif utilise habituellement une petite quantité de données annotées pour l'apprentissage du modèle, tandis que l'apprentissage profond utilise de grandes quantités de données. De fait, il est habituel dans les stratégies d'apprentissage actif d'annoter une donnée à la fois et de mettre à jour le modèle à chaque itération, ce qui ne peut pas être fait avec l'apprentissage profond (ZH DANOV 2019). Même en annotant plusieurs données à la fois avec l'apprentissage actif, le fait de devoir ajouter seulement quelques données au modèle *deep* et de le réentraîner à chaque fois est problématique car très chronophage. Ainsi, le nombre de cycles d'apprentissage actif et la taille de l'ensemble de données étiquetées à chaque itération sont des paramètres qu'il faut choisir avec précaution.

Une des solutions de la littérature par rapport à ce problème est de tirer parti de la technique d'augmentation des données, notamment en générant des données (TRAN et al. 2019) ou en utilisant de l'apprentissage auto-supervisé grâce à des pseudo-étiquettes sur certains échantillons jugés faciles (K. WANG et al. 2016). D'autres papiers ont choisi de combiner l'ensemble de données étiquetées avec celui des données non annotées, en ajoutant une composante semi-supervisée en plus de la classification supervisée initiale (SIMÉONI et al. 2021; HOSSAIN et Nirmalya ROY 2019). Les méthodes traditionnelles d'apprentissage actif recensées par (SETTLES 2009) dans sa revue de littérature ne fonctionnent pas lorsque appliquées à l'apprentissage profond d'après (SENER et SAVARESE 2017). Une des solutions afin d'adapter les stratégies traditionnelles donnée par donnée à l'apprentissage profond est d'utiliser la stratégie d'échantillonnage de lots, qui considère la diversité des échantillons de chaque lot en plus de leur informativité. Plusieurs papiers utilisent cette solution, notamment (ZH DANOV 2019; KIRSCH et al. 2019; GISSIN et SHALEV-SHWARTZ 2019; ASH et al. 2019).

- **Entraînement du classifieur et apprentissage des features : séparés en apprentissage actif, joints en apprentissage profond.** Les méthodes traditionnelles d'apprentissage actif utilisent des modèles d'apprentissage automatique précédant l'ère de l'apprentissage profond, ainsi il était courant d'avoir une partie extraction de caractéristiques séparée de l'entraînement des classifieurs. Les réseaux de neurones profonds, et notamment les réseaux de neurones convolutifs pour la classification d'images, optimisent l'entraînement du classifieur de façon conjointe avec l'apprentissage des caractéristiques. Ainsi, (K. WANG et al. 2016) remarquent que des problèmes de divergence peuvent apparaître si l'on considère l'apprentissage actif et l'apprentissage profond comme deux problèmes séparés, ou en se contentant de faire du *fine-tuning* des réseaux profonds dans le cadre de l'apprentissage actif. Dans

nos travaux, nous avons ainsi parfois utilisé notre algorithme d'apprentissage actif avec des architectures profondes complètes, et dans d'autres cas utilisé les réseaux profonds afin de récupérer les caractéristiques que nous avons utilisées comme entités fixes comme dans les approches d'apprentissage actif traditionnelles.

Afin de trouver une solution au problème, (YOO et KWEON 2019) proposent une méthode d'apprentissage actif qui est simple mais agnostique aux applications, et fonctionne avec les réseaux profonds. Pour ce faire, ils attachent un petit module paramétrique qu'ils nomment "module de prédiction de l'erreur" à un réseau cible, et lui apprennent à prédire les erreurs des données non étiquetées, afin de suggérer des données que le modèle cible va probablement mal prédire.

### 3.2.2 Méthodes génératives d'apprentissage actif

Les méthodes génératives d'apprentissage actif ont pour but de synthétiser des échantillons informatifs à ajouter à l'ensemble d'apprentissage, généralement en utilisant des réseaux adverses génératifs ou encore des auto-encodeurs variationnels (VAEs). Elles utilisent généralement le principe d'incertitude pour ce faire, mais d'autres principes d'apprentissage actif pourraient être utilisés.

#### 3.2.2.1 Méthodes à base de GAN

Parmi les méthodes qui utilisent des réseaux adverses génératifs, on peut citer l'article de GAAL (J.-J. ZHU et BENTO 2017) comme une des premières à faire de l'apprentissage actif à partir de tels réseaux. L'approche proposée par J.-J. ZHU et BENTO consiste à générer des données sur lesquelles le modèle d'apprentissage courant sera incertain, grâce au principe de l'échantillonnage par incertitude. Ainsi, pour un classifieur ayant comme fonction de décision  $f(x) = W\phi(x) + b$ , la distance approximative vers la frontière de décision est  $\|W\phi(x) + b\|$ . A partir d'un générateur  $G$  entraîné ils formulent la génération synthétique de données comme le problème d'optimisation suivant :

$$\min_z \|W\phi(G(z)) + b\|, \quad (3.7)$$

où  $z$  est la variable latente et  $G$  obtenu par le GAN. Minimiser cette fonction de coût leur permet d'orienter les échantillons qu'ils génèrent vers la frontière de décision, comme l'illustre la figure 3.6. Ils espèrent ainsi générer des exemples plus informatifs pour leur modèle, car plus incertains que les données déjà existantes, et utilisent ces exemples générés pour être annotés par l'oracle et ajoutés

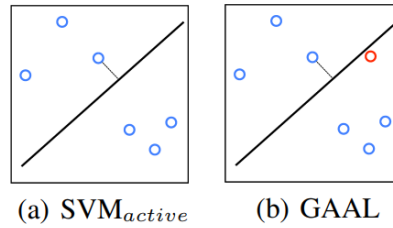


FIGURE 3.6 – **Illustration de l'intuition derrière GAAL.** L'algorithme (a) sélectionne les données proches de la frontière : ici le cercle bleu avec le trait, tandis que l'algorithme GAAL (b) génère des données potentiellement plus informatives pour le modèle : ici le cercle rouge. (crédits : J.-J. ZHU et BENTO 2017)

à l'ensemble d'entraînement. Comme leur contribution est un cadre de travail général, il est possible d'utiliser d'autres critères d'apprentissage actif comme la diversité dans leur fonction objectif afin d'obtenir des échantillons plus variés.

En pratique, ils utilisent des Support Vector Machines (SVMs) linéaires et des réseaux adverses génératifs convolutifs (DCGANs). Ils obtiennent de moins bons résultats que l'échantillonnage aléatoire sur MNIST et CIFAR-10 (réduit à deux classes, cheval et automobile), en raison d'un biais d'échantillonnage : leurs échantillons sont trop redondants et manquent de diversité. Ils sont également confrontés au problème de la génération d'échantillons difficiles à annoter car difficilement reconnaissables par un humain.

L'article (MAHAPATRA et al. 2018) utilise un réseau adverse génératif conditionnel (cGAN, MIRZA et OSINDERO 2014) : les auteurs y conditionnent leur GAN à un ensemble de données d'images réelles afin de générer des images radiographiques (rayons x) réalistes de poitrine dans le domaine de la classification et la segmentation d'images médicales. Contrairement aux méthodes utilisant l'incertitude pour sélectionner des échantillons informatifs au risque d'avoir un biais d'échantillonnage, cette méthode vise à créer des échantillons diversifiés. Pour cela, ils utilisent principalement trois composants : la génération des échantillons, leur modèle de classification et segmentation et un moyen de calculer l'informativité des données qu'ils ont générées. Ils utilisent ainsi des réseaux de neurones bayésiens (BNNS) pour capturer les incertitudes statistique et épistémique (KENDALL et GAL 2017). L'incertitude statistique est l'information que les données ne peuvent expliquer : plus de données ne va pas forcément réduire cette incertitude, mais augmenter la précision des mesures pourrait. L'incertitude épistémique est l'incertitude du modèle lui-même, on peut la réduire en augmentant le nombre de données d'entraînement.

Dans le papier (MAYER et TIMOFTE 2020), les auteurs combinent l'échantillonnage par incertitude et les réseaux adverses génératifs afin de synthétiser des instances de données incertaines. Ils n'effectuent pas de recherche exhaustive sur un ensemble de données non étiquetées, et leur méthode a supposément une résilience plus importante au biais d'échantillonnage par rapport à d'autres approches génératives d'apprentissage actif. Leur méthode nommée ASAL (*Adversarial Sampling for Active Learning*) génère de nouveaux échantillons grâce à une stratégie basée sur l'incertitude, mais ces échantillons générés par GAN ne sont pas utilisés pour entraîner le classifieur. Les échantillons synthétiques peuvent être difficiles à annoter par un oracle humain car ce sont des données incertaines et produites par un GAN, donc moins réalistes que les images d'origine. De fait, la solution du papier est d'annoter les exemples réels qui sont les plus similaires aux échantillons générés précédemment.

Pour mesurer la similarité, ils utilisent dans un premier temps la valeur des pixels en niveaux de gris ou en RVB, et un auto-encodeur dans un second temps. Les caractéristiques produites par l'encodeur servent à comparer les échantillons, après compression. Contrairement à d'autres méthodes sur bases de données, les scores d'incertitude de leurs échantillons ne sont pas recalculés à chaque cycle d'apprentissage actif. Le principal défaut de leur méthode est que seule l'incertitude est utilisée pour sélectionner les données, ce qui peut introduire de la redondance dans les groupes de données à annoter.

La méthode ARAL (*Adversarial Representation Active Learning*, MOTTAGHI et YEUNG 2019) s'inspire de certaines méthodes de cette section (P. LIU et al. 2016; TRAN et al. 2019; J.-J. ZHU et BENTO 2017) pour entraîner le réseau avec les données étiquetées et non étiquetées, tout en intégrant directement l'apprentissage génératif adverse dans le réseau afin d'obtenir plus de données pour l'améliorer. Ils observent que les données non étiquetées permettent de bien identifier la topologie des données, surtout dans les premiers cycles d'apprentissage actif lorsque le nombre d'exemples étiquetés est faible et que cette topologie est donc difficile à obtenir. Les auteurs sélectionnent des exemples peu représentés dans l'ensemble d'entraînement en étendant la méthode du papier (SINHA et al. 2019). Ainsi, ils vont entraîner leur modèle avec à la fois des données étiquetées et non étiquetées, en utilisant un GAN pour modéliser la structure sous-jacente des données non étiquetées.

### 3.2.2.2 Méthodes bayésiennes ou à base de VAE

Des travaux ont étendu l'approche introduite par (J.-J. ZHU et BENTO 2017) avec de l'apprentissage bayésien : (TRAN et al. 2019) développent BGDAL (*Bayesian Generative Active Deep Learning*), où ils combinent plusieurs méthodes généra-



tives et bayésiennes : (J.-J. ZHU et BENTO 2017; TRAN et al. 2017; ODENA et al. 2017; KINGMA et WELLING 2013). Leur but est de générer des échantillons de régions de désaccord où différentes classes s'entremêlent, et leur algorithme est illustré figure 3.7.

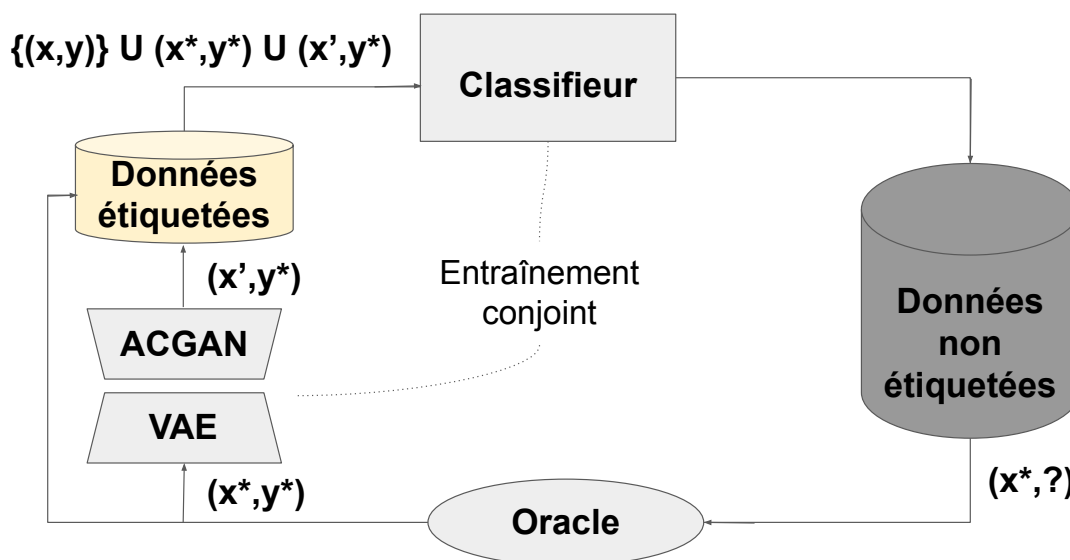


FIGURE 3.7 – **Algorithme de (Tran et al. 2019)**. On remarque ainsi qu'il y a un ACGAN, un auto-encodeur variationnel (VAE) en plus du classifieur habituel, entraînés de façon conjointe. Soit  $\{(x, y)\}$  les données étiquetées,  $x^*$  les données à étiqueter par l'oracle ( $y^*$  l'étiquette de l'oracle), et les données générées par le modèle VAE / ACGAN sont  $x'$ .

Ils modifient l'augmentation de données bayésienne en conditionnant l'étape de génération d'un échantillon  $x$  et d'une étiquette  $y$  à la place d'une variable latente  $u$  et d'un label  $y$  dans (TRAN et al. 2017). L'échantillon  $x$  le plus informatif est sélectionné à partir d'une estimation d'équation, avant de passer par l'auto-encodeur variationnel qui va servir à générer l'échantillon  $x'$ . Son étiquette est la même que pour  $x^* : y^*$  donnée par l'oracle. Cela permet d'avoir plus de données avec  $(x^*, y^*)$  et  $(x', y^*)$  qui sont utilisés pour l'entraînement du classifieur.

On peut également citer la méthode VAAL (*Variational Adversarial Active Learning*, SINHA et al. 2019) qui apprend le mécanisme d'échantillonnage d'une manière adverse, en utilisant un VAE et un réseau adverse entraîné à discriminer entre les données étiquetées et non étiquetées, afin d'apprendre un espace latent. Dans cet espace, ils utilisent un réseau adverse afin de correctement classifier les données étiquetées par rapport aux données non étiquetées. De plus, leur VAE est entraîné à apprendre un espace de features de sorte à tromper le réseau adverse pour qu'il prédise que toutes les données viennent de l'ensemble étiqueté, tan-



dis que le discriminateur apprend à faire la distinction. Leur intuition est qu'ils ne vont pas mesurer explicitement l'incertitude, mais choisir des données qui vont produire une grande incertitude et ne sont donc pas bien représentés dans l'ensemble non étiqueté.

Le papier (KIM et al. 2021) étend la méthode VAAL précédente avec TA-VAAL (Task-Aware Variational Adversarial Active Learning). La méthode VAAL étant agnostique aux tâches, considérant la distribution de données des ensembles étiquetés et non étiquetés, les auteurs de TA-VAAL modifient cette méthode en remplaçant une des *loss*, de façon à être à la fois agnostique aux tâches et attentif aux tâches. Ils combinent donc (SINHA et al. 2019; YOO et KWEON 2019), pour modéliser la façon dont l'ajout d'étiquettes aux données sélectionnées influence l'ensemble complet, mais également pour exploiter la distribution conditionnelle  $P(y|x)$  qui modélise la dépendance de la sortie par rapport à l'entrée.

### 3.2.2.3 Difficultés

Jusqu'à récemment, le gain de toutes ces approches synthétiques basées sur le GAN n'a pas été établi de manière cohérente par rapport à d'autres stratégies, notamment l'échantillonnage aléatoire et les *baselines* d'entropie maximale/distance minimale, ainsi que l'apprentissage auto-supervisé. Néanmoins, les méthodes se sont améliorées en corrigeant les divers problèmes pour obtenir de meilleurs résultats, notamment en prenant la diversité en compte ainsi qu'en utilisant pleinement les données non étiquetées. Un problème subsiste néanmoins, ces méthodes sont assez lourdes et compliquées à mettre en place, et l'annotation par un humain pourra rester difficile sur certains jeux de données plus spécifiques que les jeux publics de recherche.

## 3.2.3 Méthodes par diversité sur une base de données

Les méthodes basées sur la diversité tentent de sélectionner des échantillons de l'ensemble des données qui représentent de manière adéquate la topologie des données.

### 3.2.3.1 Approches basées sur la représentativité des données

La plupart des approches basées sur la représentativité des données, ou la densité des données, tentent de sélectionner les données par construction d'un *core-set* (PHILLIPS 2016) bien représentatif des données afin de réduire le coût d'annotation. L'idée est donc d'utiliser le *core-set* pour représenter la distribution du jeu

de données original avec un nombre moins important d'annotations. Plusieurs papiers utilisent cette approche et nous en présenterons quelques-uns.

L'article (SENER et SAVARESE 2017) présente l'apprentissage actif comme un problème de sélection de *core-sets*, minimisant la distance euclidienne dans l'espace des caractéristiques du modèle entre les points de données sélectionnés et non sélectionnés. L'objectif est de choisir un sous-ensemble de données non étiquetées de telle sorte qu'un modèle entraîné sur celui-ci ait des performances similaires à un modèle entraîné sur l'ensemble des données. Ils s'intéressent au risque réel du modèle entraîné à partir d'un petit ensemble étiqueté. Ce risque réel dépend de l'erreur d'entraînement du modèle sur le sous-ensemble étiqueté, et de l'erreur de généralisation sur le jeu de données entier ainsi qu'un terme qu'ils appellent l'erreur *core-set*. Ce terme est la différence entre l'erreur empirique moyenne sur l'ensemble étiqueté et l'erreur empirique moyenne sur le jeu de données entier qui inclue donc des données non étiquetées.

**Définition du problème de (Sener et Savarese 2017).** Les auteurs définissent le problème formellement de la façon suivante : soit  $C$  le nombre de classes d'un problème de classification multiclassés défini sur un espace compact  $\mathcal{X}$  et un espace d'étiquettes  $\mathcal{Y} = \{1, \dots, C\}$ .  $l(\cdot) : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{R}$  désigne une fonction de coût paramétrée sur les paramètres  $w$  de l'algorithme d'apprentissage profond. L'ensemble des données sont échantillonnées de façon indépendantes et identiquement distribuées (*i.i.d.*) sur l'espace  $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$  avec  $\{\mathbf{x}_i, y_i\}_{i \in [n]} \sim p_Z$  où  $[n] = \{1, \dots, n\}$ .  $s^0 = \{s^0(j) \in [n]\}_{j \in [m]}$  constitue l'ensemble initial de données choisies aléatoirement de façon uniforme. La taille du budget d'étiquettes à demander à l'oracle est de  $b$ , et  $A_s$  est un algorithme d'apprentissage qui retourne un ensemble de paramètres  $w$  à partir d'un ensemble de données étiquetées  $s$ . Ils définissent ainsi l'apprentissage actif comme :

$$\min_{s^1: |s^1| \leq b} E_{\mathbf{x}, y \sim p_Z} [l(\mathbf{x}, y; A_{s^0 \cup s^1})] \quad (3.8)$$

Néanmoins, le budget  $b$  va être séparé en plusieurs cycles d'apprentissage actif, ce qui transforme le problème en :

$$\min_{s^{k+1}: |s^{k+1}| \leq b} E_{\mathbf{x}, y \sim p_Z} [l(\mathbf{x}, y; A_{s^0 \cup \dots \cup s^{k+1}})] \quad (3.9)$$

**Adaptation de l'apprentissage actif comme couverture d'ensemble.** Afin d'obtenir une stratégie d'apprentissage actif adaptée à l'entraînement par lots, les auteurs proposent dans l'équation 3.10 une borne supérieure de la fonction de coût définie précédemment (*cf.* équation 3.8).

$$\begin{aligned}
E_{\mathbf{x}, y \sim p_Z} [l(\mathbf{x}, y; A_s)] &\leq \underbrace{\left| E_{\mathbf{x}, y \sim p_Z} [l(\mathbf{x}, y; A_s)] - \frac{1}{n} \sum_{i \in [n]} l(\mathbf{x}_i, y_i; A_s) \right|}_{\text{Erreur de généralisation}} + \underbrace{\frac{1}{|s|} \sum_{j \in s} l(\mathbf{x}_j, y_j; A_s)}_{\text{Erreur d'entraînement}} \\
&+ \underbrace{\left| \frac{1}{n} \sum_{i \in [n]} l(\mathbf{x}_i, y_i; A_s) - \frac{1}{|s|} \sum_{j \in s} l(\mathbf{x}_j, y_j; A_s) \right|}_{\text{coût } core\text{-set}} \quad (3.10)
\end{aligned}$$

Le risque de population est ainsi contrôlé par l'erreur d'entraînement du modèle entraîné sur l'ensemble étiqueté  $s$ , l'erreur de généralisation sur le jeu de données entier  $[n]$  et un terme qu'ils nomment le coût *core-set*. Ce dernier terme est calculé avec la différence entre le coût empirique moyen sur l'ensemble de points qui ont des étiquettes et le coût empirique moyen sur le jeu de données complet, ce qui inclut donc des données non étiquetées. On observe dans l'équation 3.11 de quelle façon les auteurs redéfinissent leur problème d'apprentissage actif.

$$\min_{s^1: |s^1| \leq b} \frac{1}{n} \sum_{i \in \{1, \dots, n\}} l(\mathbf{x}_i, y_i; A_{s^0 \cup s^1}) - \frac{1}{|s^0| + |s^1|} \sum_{j \in s^0 \cup s^1} l(\mathbf{x}_j, y_j; A_{s^0 \cup s^1}) \quad (3.11)$$

De façon plus informelle, à partir d'un ensemble annoté initial ( $s^0$ ) et d'un budget ( $b$ ), ils essaient de trouver un ensemble de points à annoter ( $s^1$ ). Le but étant que les performances soient les plus proches des performances d'un modèle entraîné sur le jeu de données entier lorsqu'ils apprennent un modèle à partir de ce sous-ensemble étiqueté ( $s^1$ ). De plus amples détails sur leur implémentation des *core-set* pour les CNNs, la résolution du problème k-centre ainsi que leurs performances sont disponibles dans leur papier (SENER et SAVARESE 2017).

Le papier (GEIFMAN et EL-YANIV 2017) s'inspire des idées de compression de *core-set* et présentent un algorithme d'apprentissage actif qui sélectionne des points consécutifs de la base de données. Pour cela, ils utilisent l'algorithme glouton de résolution du problème k-centre (GONZALEZ 1985), dans l'espace d'une activation neurale d'une couche de représentation. Ils utilisent des *core-sets* afin de compresser les données d'entrée en fonction de leur représentation interne, l'algorithme glouton de (GONZALEZ 1985) est leur moteur de *core-set* mais les auteurs indiquent que d'autres peuvent être utilisés, potentiellement plus performants.

### 3.2.3.2 Autres approches par diversité

Dans cet article de Yong Cheng Wu (Y. C. WU 2013), l'auteur tente de maximiser la diversité en utilisant des méthodes non profondes car elles n'étaient pas très répandues en 2013. Il utilise deux classifieurs sur le même ensemble et essaie de sélectionner les instances qui ont été classées différemment par chaque classifieur. C'est une variation du co-testing (MUSLEA et al. 2006) où l'auteur Y. C. WU utilise deux classifieurs comme indiqué précédemment, en lieu et place de deux vues. Afin de mesurer l'informativité des données, ils utilisent la confiance de sa prédiction selon les données étiquetées.

L'article (AGARWAL et al. 2020) tente d'utiliser la diversité contextuelle au lieu de la simple diversité visuelle comme les autres méthodes. Ils observent que le vecteur de probabilité des CNNs pour une région d'intérêt contient généralement des informations provenant d'un champ réceptif plus large. Ils utilisent cette mesure de diversité contextuelle dans deux cadres : un ensemble de base et une politique d'apprentissage par renforcement pour la sélection active des images, et parviennent à obtenir des résultats de pointe sur la segmentation sémantique, la détection d'objets et la classification d'images. La récompense qu'ils utilisent pour leur partie d'apprentissage par renforcement est l'équation 3.12.

$$R = \alpha R_{cd} + (1 - \alpha)(R_{vr} + R_{sr}) \quad (3.12)$$

Le détail des équations des sous-récompenses  $R_{cd}$ ,  $R_{vr}$ ,  $R_{sr}$  sont dans le papier, mais  $R_{cd}$  mesure la diversité contextuelle qui est agrégée sur un sous-ensemble d'images ;  $R_{vr}$  mesure la représentation visuelle en utilisant les caractéristiques des images sur l'ensemble non étiqueté. Ils indiquent que cette sous-récompense préfère choisir des images étalées sur l'espace des features, de façon similaire aux approches k-medoides (approche plus robuste aux valeurs aberrantes que k-moyennes). Finalement,  $R_{sr}$  est la sous-récompense de la représentation sémantique afin de s'assurer d'avoir des images équilibrées sur l'ensemble des classes.

### 3.2.3.3 Difficultés

Les méthodes basées sur la distance, dont notamment celles utilisant les *core-sets*, ont souvent le problème des distances entre les points de données qui peuvent sembler identiques dans un espace à haute dimension. De plus, la diversité offre généralement de bonnes performances dans les premiers cycles d'apprentissage actif, mais finissent par manquer d'informativité au fur et à mesure, ce qui peut être résolu avec l'incertitude.

### 3.2.4 Méthodes par incertitude sur une base de données

Il existe plusieurs sous-catégories parmi les méthodes d'apprentissage actif basées sur l'incertitude : celles avec ou sans ensemble. Les méthodes ensemblistes utilisent plusieurs modèles, ou plusieurs phases d'inférence avec un seul modèle, afin d'obtenir des incertitudes prédictives plus calibrées, tandis que les méthodes non-ensemblées n'utilisent qu'un seul modèle avec une seule inférence et *dropout* afin de calculer les valeurs d'incertitude. Généralement, les méthodes par incertitude tentent de sélectionner des échantillons jugés ambigus, en partant du principe que plus un modèle est incertain quant à sa prédiction, plus l'échantillon est informatif pour le modèle.

#### 3.2.4.1 Méthodes ensemblistes

Certains articles, comme (GAL et GHARAMANI 2016), abordent l'incertitude en utilisant l'apprentissage bayésien : ils utilisent le dropout pendant l'inférence pour modéliser l'incertitude dans les réseaux profonds. Plusieurs inférences sont effectuées pour estimer l'incertitude des prédictions en utilisant le dropout sur leur modèle pour chaque échantillon. Cette technique, connue sous le nom de *dropout* de Monte Carlo (MC-Dropout), leur permet d'approcher l'incertitude avec plus de précision qu'une sortie unique de *softmax*, en approximant une distribution de probabilité sur les paramètres.

L'article (GAL et al. 2017) utilise spécifiquement cette technique pour l'apprentissage actif, ce qui leur permet d'acquérir des échantillons informatifs accélérant l'apprentissage du modèle. Le papier a ainsi pour but d'adapter l'apprentissage actif à un usage avec de l'apprentissage profond, ce qui n'était pas forcément commun à l'époque du papier en 2017. Leur utilisation des réseaux de neurones bayésiens convolutifs leur permet de modéliser l'incertitude de façon plus efficace qu'en utilisant un seul modèle, et donc d'obtenir de bonnes performances en apprentissage actif profond.

Le papier (BELUCH et al. 2018) n'utilise pas MC-Dropout pour approximer la distribution mais plutôt un ensemble complet de modèles CNNs, avec des performances améliorées par rapport aux méthodes utilisant MC-Dropout, grâce à une incertitude mieux étalonnée. Ils attribuent leurs meilleures performances que les méthodes MC-Dropout au manque de diversité de ces dernières et de leur capacité de modélisation diminuée.

### 3.2.4.2 Méthodes non-ensemblistes

Dans le papier (K. WANG et al. 2016), les auteurs sélectionnent les échantillons les plus informatifs en utilisant une stratégie d'échantillonnage par incertitude. Pour ce faire, ils utilisent les probabilités de classe obtenues avec la sortie *softmax* pour calculer l'incertitude par la technique de moindre confiance, marge minimale ou entropie. Ils formulent leur algorithme *CEAL* (Cost-Effective Active Learning) dans l'équation 3.13. A partir d'un jeu de données  $D$  contenant  $m$  catégories et  $n$  échantillons tel que  $D = \{x_i\}_{i=1}^n$ ,  $D^L$  correspond aux données étiquetées et  $D^U$  les données non étiquetées. L'étiquette de  $x_i$  est  $y_i = j$  avec  $j \in \{1, \dots, m\}$ , sachant que la plupart des données du problème sont non étiquetées.

$$\min_{\{\mathcal{W}, y_i, i \in D^U\}} -\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^m \mathbf{1}\{y_i = j\} \log p(y_i = j | x_i; \mathcal{W}), \quad (3.13)$$

Ici,  $\mathbf{1}$  est la fonction indicatrice,  $\mathcal{W}$  les paramètres du CNN,  $p(y_i = j | x_i; \mathcal{W})$  la sortie softmax du CNN pour la catégorie  $j$ , ce qui représente la probabilité de l'échantillon  $x_i$  d'appartenir au classifieur  $j$ . Leur algorithme est conçu de sorte à mettre à jour les exemples pseudo-étiquetés  $y_i$  et les paramètres du réseau  $\mathcal{W}$  alternativement.

L'article de Yoo et Kweon (YOO et KWEON 2019) n'utilise ni ensembles ni softmax, mais parvient tout de même à obtenir de bonnes performances d'apprentissage actif. Ils utilisent un modèle de prédiction de *loss* supplémentaire à leur CNN, qui essaie d'apprendre les *loss* des points de données non étiquetés. Ensuite, ils sélectionnent les échantillons dont la *loss* prédite est la plus élevée pour les étiqueter : ils peuvent suggérer des données pour lesquelles le modèle cible est susceptible de produire une prédiction erronée. Un article similaire de Hemmer et al. (HEMMER et al. 2022) tente d'estimer l'incertitude avec une distribution de Dirichlet sur les probabilités de classe. L'estimation de l'incertitude uniquement avec les probabilités softmax est parfois sous-optimale, puisque le modèle est souvent incertain dans ses prédictions même avec une sortie softmax élevée (GAL et al. 2016). Ils considèrent aussi que les méthodes d'ensemble prennent trop de temps puisqu'il y a plusieurs passes avec le *dropout*, et que l'attache d'un autre modèle au réseau est trop coûteuse en termes d'implémentation.

### 3.2.4.3 Difficultés

Les principaux problèmes des approches basées sur l'incertitude sont le biais d'échantillonnage, où un manque de diversité des données bride les performances de ces stratégies. C'est d'autant plus le cas avec l'apprentissage profond dans le cas où on fait de l'apprentissage actif par lots, il y a un risque d'avoir beaucoup

de redondance dans les images choisies ce qui peut amener à de moins bonnes performances que l'échantillonnage aléatoire.

### 3.2.5 Méthodes hybrides sur une base de données

Il existe plusieurs papiers qui combinent les approches par incertitude et par diversité dans le cadre de l'apprentissage actif. Du fait que les CNNs apprennent par lots, on a vu précédemment que les méthodes par incertitude pouvaient sélectionner des données redondantes ce qui affecte les performances. Ainsi, il faut faire attention à sélectionner des données informatives de façon jointe, et pas seulement individuellement. De la même façon, les méthodes par diversité peuvent manquer d'informativité, il est donc intéressant de combiner les deux approches.

#### 3.2.5.1 Présentation de divers articles

La variante BatchBALD (KIRSCH et al. 2019) qui s'inspire de BALD (HOULSBY et al. 2011) est une approximation *tractable* de l'information mutuelle entre un lot de points et les paramètres du modèle. Cette méthode est utilisée comme fonction d'acquisition pour l'apprentissage actif bayésien profond afin d'échantillonner conjointement plusieurs données informatives. Tout en ayant des performances comparables à celles d'autres approches basées sur des lots, cette méthode présente l'avantage supplémentaire de réduire la redondance des échantillons sélectionnés. Le papier (DEPEWEG et al. 2018) utilise également les réseaux neuronaux bayésiens pour l'apprentissage actif. Les auteurs soutiennent que ces réseaux, avec des variables latentes, sont des modèles probabilistes à la fois évolutifs et flexibles, qui peuvent tenir compte de l'incertitude dans l'estimation des poids du réseau et, en utilisant les variables latentes, capturer des modèles de bruit complexes dans les données. Ils décomposent l'incertitude en composantes épistémiques et aléatoires afin d'identifier les points informatifs pour l'apprentissage actif, et ils utilisent également l'apprentissage par renforcement.

Dans le papier BADGE (ASH et al. 2019) (*Batch Active Learning by Diverse Gradient Embeddings*), ils utilisent également à la fois l'incertitude et la diversité et échantillonnent des groupes de points disparates et de grande ampleur lorsqu'ils sont représentés dans un espace de gradient utilisant des étiquettes hallucinées. Ils font un compromis entre l'incertitude et la diversité sans utiliser d'hyperparamètres ajustés manuellement, contrairement à d'autres approches qui réussissent parfois avec une taille de lot ou une architecture particulière. Ils calculent l'incertitude en utilisant l'amplitude du gradient à l'aide des paramètres de la couche de sortie finale, et ils utilisent l'algorithme K-Means++ pour la diversité.

### 3.2.5.2 Difficultés

Les principales difficultés des approches qui combinent l'incertitude et la diversité, sont la nécessité de bien pondérer l'un et l'autre. En effet, l'un pourra s'avérer plus utile dans les premiers cycles d'apprentissage actif et l'autre plutôt vers la fin. C'est pourquoi nous avons fait évoluer notre méthode basée sur une combinaison des deux, en partant d'une pondération fixée par hyperparamètres initialement, à une pondération dynamique au fur et à mesure des itérations d'apprentissage actif.

## 3.3 Positionnement des travaux et contributions

### 3.3.1 Diversité, représentativité et incertitude des données

Notre première contribution appartient aux approches hybrides d'apprentissage actif basées sur la combinaison de la diversité et de l'incertitude, et nous utilisons également un troisième critère de représentativité. Nous n'utilisons pas d'ensembles et nous utilisons la sortie *softmax* des probabilités de classe comme mesure d'incertitude, en combinant tous nos critères dans une seule fonction objective unifiée. Notre critère de diversité nous permet d'éviter l'inconvénient des méthodes basées sur l'incertitude qui peuvent avoir des points de données redondants dans leurs lots, tandis que le critère d'incertitude nous permet de trouver des échantillons informatifs. De plus, le critère de représentativité nous permet d'éviter les valeurs aberrantes, en prenant des échantillons plus proches des centroïdes de leur cluster.

### 3.3.2 Apprentissage par renforcement pour la pondération des critères

Notre deuxième contribution utilisant l'apprentissage par renforcement est une approche sur une base de données qui combine la diversité, représentativité et l'incertitude des données de la même manière que notre première contribution. La différence principale avec cette dernière réside dans l'optimisation dynamique des hyperparamètres de pondération de ces critères. On note ces hyperparamètres  $\alpha, \beta, \eta$  et ils pondèrent respectivement la diversité, l'incertitude et la représentativité des données. La fonction de récompense que l'on utilise est variable, et dans certains cas sera considérée comme faisant partie de l'approche de requête



par votes présentée précédemment, ou bien de réduction de l'erreur prévue voire changement de modèle prévu.

### 3.3.3 Exemples virtuels informatifs pour l'apprentissage actif

Finalement, notre 3ème contribution se situe à la fois dans le scénario sur une base de données, mais également dans le scénario de génération d'échantillons. En effet, nous utilisons les données de l'ensemble non labellisé  $\mathcal{U}$  tout en générant des données artificielles grâce à une fonction objectif. Notre fonction objectif utilise la diversité et représentativité des données, avec une composante d'incertitude, il s'agit donc d'une méthode hybride.

## 3.4 Bases de données et métriques d'évaluation

Nous avons effectué des expériences sur quatre jeux de données : MNIST, CIFAR-10 et CIFAR-100 ainsi que Object DOTA. Les trois premiers sont des jeux de données classiques de la littérature en apprentissage actif, tandis que le dernier est l'adaptation d'un jeu de détection d'objets à la tâche de classification d'images.

### 3.4.1 MNIST



FIGURE 3.8 – Le jeu de données de chiffres manuscrits MNIST.

MNIST est un jeu de données de reconnaissance de chiffres manuscrits, largement utilisé mais considéré comme facile. Ce n'est pas notre cible principale puisqu'il est peu utilisé dans les articles sur l'apprentissage actif, mais il est tout de même intéressant de voir si nous pouvons battre les baselines sur ce jeu de

données. Il comprend 60000 images d'entraînement et 10000 images de test. Les classes sont les chiffres [0 – 9].

### 3.4.2 CIFAR-10 et CIFAR-100

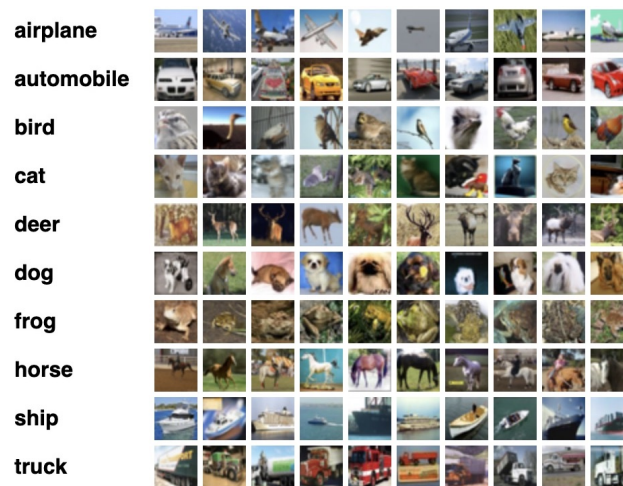


FIGURE 3.9 – Le jeu de données multi-classes CIFAR-10.

Le jeu de données CIFAR-10 est un sous-ensemble étiqueté du jeu de données de 80 millions de petites images. Il est composé de 60000 images couleur  $32 \times 32$  réparties en 10 classes, avec 6000 images par classe. Il y a 50000 images d'entraînement et 10000 images de test. Les classes sont : avion, automobile, oiseau, chat, cerf, chien, grenouille, cheval, bateau, camion.

Le jeu de données CIFAR-100 quant à lui est exactement comme CIFAR-10, excepté le fait qu'il comporte 100 classes contenant 600 images chacune. Il y a ainsi 500 images d'entraînement et 100 de test, les 100 classes sont regroupées en 20 superclasses. Chaque image a ainsi un label "fin" (sa classe) et un label "grossier" (sa superclasse).

### 3.4.3 Object-DOTA

Nous avons un jeu de données cible pour notre application : Object-DOTA, qui est le jeu de données DOTA (XIA et al. 2018) mais transformé pour être un jeu de données de classification d'images.

Nous avons utilisé les boîtes englobantes de la vérité terrain, afin d'extraire de chaque image les objets et ainsi obtenir une banque d'images que nous utiliserons comme jeu de données de classification d'images.

Puisque DOTA peut être évalué de plusieurs façons, par exemple en utilisant



FIGURE 3.10 – Le jeu de données DOTA avant son adaptation en Object-DOTA.

des boîtes englobantes tournées, les coordonnées sont fournies comme suit :  $x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4$ . Pour faciliter l'extraction d'objets ainsi que la classification, nous ne récupérerons pas les polygones originaux, mais des rectangles non orientés à la place :



Il y a 98990 images d'entraînement et 28853 images de test pour un total de 15 classes différentes : avion, navire, réservoir de stockage, losange de baseball, court de tennis, terrain de basket-ball, terrain d'athlétisme, port, pont, grand véhicule, petit véhicule, hélicoptère, rond-point, terrain de football et piscine.

#### 3.4.4 Données satellitaires en détection de changements

Lors de notre première contribution, la base de données de détection de changements dans des images satellites consiste en : 2200 paires d'échantillons sans chevauchement ( $30 \times 30$  pixels en RVB) pris depuis deux grandes images satellites GeoEye-1 (une comme référence et une de test) de  $2400 \times 1652$  pixels avec une résolution spatiale de 1.65m/pixel. Ces images correspondent à la même zone de Jefferson en Alabama et ont été prises en 2010 et 2011, où de nombreux changements sont présents notamment dûs à des tornades, ainsi que des non changements (sont inclus des changements non pertinents comme les nuages). La vérité terrain contient 2161 paires d'images négatives (*ie.* pas de changements; des changements non pertinents) et seulement 39 paires d'images positives (des changements pertinents), ce qui rend la base très déséquilibrée : moins de 2% des

images correspondent à des changements pertinents. La moitié de cet ensemble de données est utilisée pour l'apprentissage et l'autre moitié pour l'évaluation.



FIGURE 3.11 – **Paire d'images de Jefferson.** Cette paire d'images montre un zoom sur une partie de la région de Jefferson ayant subi des changements suite à une tornade.

### 3.4.5 Métriques utilisées

#### 3.4.5.1 Classification d'images : *accuracy score*

La métrique que nous utilisons pour nos contributions en apprentissage actif sur la tâche de classification d'images est *l'accuracy score* ou score de précision. Il s'agit d'une métrique de performances d'un modèle de classification, calculé par le nombre de prédictions correctes divisé par le nombre total de prédictions. Sa facilité de calcul et de compréhension en fait une métrique classique. Dans certains cas, nous utilisons également sa version équilibrée dite *balanced accuracy score*, où une pondération sera appliquée en fonction du nombre d'exemples dans chaque classe, ce qui est particulièrement adapté pour les jeux de données déséquilibrés.

#### 3.4.5.2 Détection de changements : taux d'erreur EER

La métrique utilisée pour la détection des changements dans les images satellites est le taux d'erreur EER (pour *equal error rate*), qui correspond à une erreur de généralisation équilibrée avec autant d'importance pour des erreurs dans les classes "changements" et "pas de changements". Ainsi,  $EER = (ERP + ERN) / 2$  avec  $ERP = \# \text{ changements faux positifs} / \# \text{ changements vrais négatifs}$  et  $ERN = \# \text{ changements faux négatifs} / \# \text{ changements vrais positifs}$ . Les faux positifs et les faux négatifs sont pénalisés de la même manière, afin d'éviter que l'erreur soit dominée par le taux de faux positifs. En effet, la classe des non changements est dominante, et l'EER est adaptée au fait qu'il y ait beaucoup de déséquilibre des

classes en détection de changements. La performance est meilleure lorsque l'EER est plus basse.

## APPRENTISSAGE PROFOND ACTIF POUR LA RECONNAISSANCE VISUELLE

### *Résumé chapitre*

*Nous présentons dans ce chapitre un nouvel algorithme d'apprentissage actif pour la détection des changements dans les images satellites ainsi que la classification d'images, néanmoins notre méthode est utilisable avec d'autres applications. La solution proposée est interactive et basée sur un modèle de questions et de réponses, qui pose à un oracle (annotateur) les questions les plus informatives sur la pertinence des images échantillonnées. En fonction des réponses de l'oracle, la fonction de décision est mise à jour itérativement. Nous étudions un nouveau cadre qui modélise la probabilité que les échantillons soient pertinents; cette probabilité est obtenue en minimisant une fonction objectif capturant la représentativité, la diversité et l'ambiguïté des données. Seules les données ayant une probabilité élevée selon ces critères sont sélectionnées et soumises à l'oracle afin d'être annotées. De nombreuses expériences sur entre autres la tâche de la détection de changements sur des images satellites après des catastrophes naturelles (notamment des tornades) montrent la pertinence de la méthode proposée par rapport aux travaux connexes. Nous proposons également des expériences de classification d'images sur plusieurs jeux de données (MNIST, CIFAR-10, CIFAR-100 ainsi qu'Object-DOTA) et démontrons l'efficacité de notre méthode sur cette tâche.*

## Sommaire

---

|       |   |     |
|-------|---|-----|
| 4.1   | Introduction . . . . .  | 82  |
| 4.2   | Méthode proposée . . . . .  | 83  |
| 4.2.1 | Apprentissage interactif . . . . .  | 83  |
| 4.2.2 | Fonction d’acquisition de données . . . . .   | 84  |
| 4.2.3 | Optimisation de notre fonction objectif . . . . .                                     | 86  |
| 4.2.4 | Partitionnement multiple de données . . . . .   | 86  |
| 4.2.5 | Différentes distances et régularisations . . . . .                                    | 88  |
| 4.3   | Expériences en classification d’images . . . . .                                      | 89  |
| 4.3.1 | Implémentation et <i>baselines</i> . . . . .  | 90  |
| 4.3.2 | Le choix de la distance a un impact sur les performances . . . . .                    | 91  |
| 4.3.3 | L’impact du choix du CNN sur la hiérarchie des méthodes . . . . .                     | 93  |
| 4.3.4 | L’impact individuel et joint de la diversité, représentativité et ambiguïté . . . . . | 96  |
| 4.3.5 | Performances compétitives avec l’état de l’art . . . . .                              | 100 |
| 4.3.6 | L’analyse qualitative confirme la pertinence de notre méthode . . . . .               | 102 |
| 4.4   | Expériences en détection de changements . . . . .                                     | 103 |
| 4.4.1 | Détails d’implémentation et jeux de données . . . . .                                 | 103 |
| 4.4.2 | Net avantage de notre méthode face aux baselines . . . . .                            | 105 |
| 4.4.3 | Impact de chacun des termes selon le cycle d’apprentissage actif . . . . .            | 105 |
| 4.5   | Conclusion . . . . .  | 107 |

---

## 4.1 Introduction

L’objet de notre première contribution est d’obtenir de bonnes performances avec des modèles d’apprentissage automatique pour les tâches de classification d’images, sur divers jeux de données – CIFAR-10, MNIST ou une variante de DOTA que nous appelons Object-DOTA – ainsi que la détection de changement sur des images satellites, en utilisant le moins d’annotations possibles. La solution retenue est l’apprentissage actif décrit précédemment en [section 3.1.3](#), et notamment profond de par l’utilisation de CNNs.

Nous introduisons un algorithme pour apprendre l’informativité des données en utilisant un modèle probabiliste basé sur des questions & réponses à un oracle (utilisateur en mesure d’étiqueter les données, *cf.* [section 4.2.1](#)). Ce modèle capture l’informativité des échantillons à l’aide d’une fonction objectif qui mélange les critères de représentativité, de diversité et d’ambiguïté des données, ainsi qu’un terme de régularisation (*cf.* [section 4.2.2](#)). Le critère de diversité nous permet

d'échantillonner les données à travers plusieurs clusters tandis que la représentativité ne considère que les échantillons proches de leurs centroïdes, évitant ainsi les données significativement éloignées des autres (aussi appelées *outliers*). Le critère d'ambiguïté sélectionne quant à lui les données proches des frontières de décision des classificateurs entraînés. Enfin, le terme de régularisation permet de contrôler la taille de l'échantillon. La fonction objectif proposée admet une solution analytique sous la forme d'une distribution de Gibbs (cf. section 4.2.3).

## 4.2 Méthode proposée

En considérant  $\mathcal{S} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{X}$  comme une collection d'images d'apprentissage non étiquetées tirées d'une distribution de probabilité existante mais inconnue  $P(X, Y)$ ; ici  $X$  et  $Y$  sont deux variables aléatoires représentant respectivement les données d'entrée dans  $\mathcal{X}$  et leurs étiquettes de vérité terrain inconnues (c'est-à-dire les classes ou les catégories, prises dans un ensemble  $\mathcal{Y}$ ). Notre objectif est d'apprendre une fonction de correspondance  $f : \mathcal{X} \rightarrow \mathcal{Y}$  qui assigne un échantillon donné  $\mathbf{x} \in \mathcal{X}$  à une catégorie  $\mathbf{y} \in \mathcal{Y}$ . L'apprentissage de  $f$  nécessite un sous-ensemble de données d'apprentissage annotées par un oracle (c'est-à-dire un utilisateur ou un expert). Comme ces annotations sont généralement très coûteuses, la conception de  $f$  doit tenir compte du fait qu'il faut utiliser le moins d'étiquettes possible tout en minimisant l'erreur de généralisation  $P(f(X) \neq Y)$ .

### 4.2.1 Apprentissage interactif

Notre algorithme est itératif et repose sur un modèle question-réponse, qui suggère l'affichage le plus informatif à un oracle, recueille les annotations et met à jour la fonction de décision  $f$  en conséquence. Soit  $\mathcal{D}_t \subset \mathcal{S}$  un sous-ensemble d'images présentées à un oracle à l'itération  $t$  et soit  $\mathcal{Y}_t$  les étiquettes inconnues de  $\mathcal{D}_t$ ; en pratique  $|\mathcal{D}_t|$  est fixé en fonction d'un budget cible d'annotations : par exemple 10 cycles (donc  $0 \leq t < 10$ ) de 100 étiquettes ( $|\mathcal{D}_t| = 100$ ). Nous construisons notre fonction d'apprentissage  $f$  itérativement en posant à l'oracle des "questions" sur les étiquettes de  $\mathcal{D}_t$  selon les étapes suivantes.

**Page zéro.** Sélectionner une page  $\mathcal{D}_0$  comprenant des représentants de  $\mathcal{S}$ . Dans la pratique,  $\mathcal{D}_0$  correspond à des données aléatoires ou aux centroïdes d'un cluster  $\{h_1, \dots, h_K\}$  de  $\mathcal{S}$  obtenu avec l'algorithme des K-moyennes.



Exécuter les étapes suivantes pour  $t = 0, \dots, T - 1$  ( $T$  peut également dépendre d'un budget d'annotation prédéfini) :

- **Oracle.** Etiqueter la page  $\mathcal{D}_t$  avec une fonction oracle notée  $\mathcal{C}(\cdot)$  et affecter  $\mathcal{C}(\mathcal{D}_t)$  à  $\mathcal{Y}_t$ .
- **Modèle d'apprentissage.** Entraîner une fonction de décision  $f_t(\cdot)$  sur les données étiquetées jusqu'à présent,  $\cup_{k=0}^t (\mathcal{D}_k, \mathcal{Y}_k)$ , et l'utiliser pour prédire les étiquettes sur les données de test en fonction de  $f_t(\cdot)$ . En pratique, nous utilisons différents classifieurs, y compris des réseaux de neurones convolutifs, afin de construire et mettre à jour  $f_t(\cdot)$ .
- **Modèle d'affichage.** Sélectionner le prochain affichage  $\mathcal{D}_{t+1} \subset \mathcal{S} \setminus \cup_{k=0}^t \mathcal{D}_k$  (à montrer à l'oracle) qui minimise  $P(f_{t+1}(X) \neq Y)$ . Il est clair qu'une stratégie de force brute qui (i) considère tous les affichages possibles  $\mathcal{D} \subset \mathcal{S} \setminus \cup_{k=0}^t \mathcal{D}_k$ , (ii) apprend les classifieurs sous-jacents  $f_{t+1}(\cdot)$  sur  $\mathcal{D} \cup_{k=0}^t \mathcal{D}_k$  et (iii) évalue leur erreur de généralisation est hautement combinatoire et hors de portée.

Des heuristiques de sélection d'affichage, liées à l'apprentissage actif, sont généralement utilisées à la place, mais il convient d'être prudent dans l'utilisation de ces heuristiques car nombre d'entre elles peuvent donner de moins bons résultats que la stratégie de sélection basique consistant à choisir les données de manière uniformément aléatoire (HUIJSER et GEMERT 2017). Notre modèle d'affichage alternatif proposé dans ce chapitre combine la diversité, l'ambiguïté ainsi que la représentativité des données d'entrée; la diversité vise à sélectionner les données afin de découvrir différents modes de  $f_{t+1}(\cdot)$  tandis que l'ambiguïté cherche à affiner localement la frontière de décision de  $f_{t+1}(\cdot)$ .

## 4.2.2 Fonction d'acquisition de données

Notre algorithme est probabiliste et attribue à chaque échantillon  $\mathbf{x}_i$  un degré d'informativité  $\mu_i$  qui mesure la probabilité que  $\mathbf{x}_i$  appartienne à l'affichage suivant  $\mathcal{D}_{t+1}$ ; par conséquent,  $\mathcal{D}_{t+1}$  correspondra aux  $\{\mathbf{x}_i\}_i$  non étiquetés ayant les plus fortes informativités  $\{\mu_i\}_i$ . En considérant  $\mu \in \mathbb{R}^n$  comme un vecteur de ces  $\{\mu_i\}_i$ , nous proposons de trouver  $\mu$  comme l'optimum du problème de minimisation sous contrainte suivant :

$$\begin{aligned} \min_{\mu \geq 0, \|\mu\|_1=1} & \eta \operatorname{tr}(\operatorname{diag}(\mu'[\mathbf{C} \circ \mathbf{D}])) + \alpha [\mathbf{C}'\mu]' \log[\mathbf{C}'\mu] \\ & + \beta \operatorname{tr}(\operatorname{diag}(\mu'[\mathbf{F} \circ \log \mathbf{F}])) + \gamma \mu' \log \mu, \end{aligned} \quad (4.1)$$

Ici  $\circ, '$  sont respectivement le produit d'Hadamard et la transposition matricielle,  $\|\cdot\|_1$  est la norme  $\ell_1$ ,  $\log$  est appliqué entrée par entrée et  $\text{diag}$  fait correspondre un vecteur à une matrice diagonale. Dans la fonction objectif ci-dessus, (i)  $\mathbf{D} \in \mathbb{R}^{n \times K}$  et  $\mathbf{D}_{ik} = d_{ik}^2$  est la distance euclidienne entre  $\mathbf{x}_i$  et le centroïde de cluster  $k$ , (ii)  $\mathbf{C} \in \mathbb{R}^{n \times K}$  est la matrice indicatrice dont chaque entrée  $\mathbf{C}_{ik} = 1$  si  $\mathbf{x}_i$  appartient au cluster  $k$  (0 sinon), et (iii)  $\mathbf{F} \in \mathbb{R}^{n \times nc}$  est une matrice de scores de classification avec  $\mathbf{F}_{ic} = f_t^c(\mathbf{x}_i)$  et  $f_t^c(\cdot)$  étant une fonction de classification proportionnelle à la probabilité que  $\mathbf{x}_i$  appartienne à la  $c^{\text{ème}}$  classe <sup>1</sup>.

Le premier terme de cette fonction objectif mesure la *représentativité* des échantillons sélectionnés dans  $\mathcal{D}$  (réécrit sous la forme  $\sum_i^n \sum_k^K 1_{\{\mathbf{x}_i \in h_k\}} \mu_i d_{ik}^2$ ); en d'autres termes, il capte la proximité de chaque  $\mathbf{x}_i$  par rapport au centroïde de son groupe, de sorte que ce terme atteint sa plus petite valeur lorsque tous les échantillons sélectionnés coïncident avec ces centroïdes. Le deuxième terme (réécrit comme  $\sum_k [\sum_{i=1}^n 1_{\{\mathbf{x}_i \in h_k\}} \mu_i] \log[\sum_{i=1}^n 1_{\{\mathbf{x}_i \in h_k\}} \mu_i]$ ) mesure la *diversité* des échantillons sélectionnés en tant qu'entropie de la distribution de probabilité des clusters sous-jacents; cette mesure est minimisée lorsque les échantillons sélectionnés appartiennent à des clusters différents et vice-versa. L'algorithme utilisé pour le clustering en pratique est k-means avec initialisation k-means++ (ARTHUR et VASSILVITSKII 2007), sur les caractéristiques extraites des images par un ResNet-18 préentraîné sur ImageNet pour la classification d'images, et sur les caractéristiques extraites par un CNN basé sur les graphes pour la détection de changements sur la base Jefferson. Le troisième critère (équivalent à  $\sum_i \sum_c^{nc} \mu_i \mathbf{F}_{ic} \log \mathbf{F}_{ic}$ ) modélise l'*ambiguïté* ou l'*incertitude* dans  $\mathcal{D}$  mesurée comme l'entropie des probabilités de classes obtenues avec le classifieur; ce terme atteint sa plus petite valeur lorsque les données ont des scores identiques pour chacune des classes. Enfin, le quatrième terme est lié à la *cardinalité* de  $\mathcal{D}$ , mesurée par l'entropie de la distribution  $\mu$ ; ce terme agit également comme un régularisateur et aide à obtenir une solution analytique (comme on le verra par la suite). L'impact de ces termes est contrôlé par  $\alpha, \beta, \gamma, \eta \geq 0$ .

Nous formulons le problème de minimisation en ajoutant une contrainte d'égalité et une borne inférieure sur les  $\{\mu_i\}_i$  qui assurent une normalisation des valeurs d'informativité et nous permettent de considérer  $\{\mu_i\}_i$  comme une distribution de probabilité sur  $\mathcal{S}$ .

---

1. Ces scores d'ambiguïté peuvent être obtenus grâce à l'opérateur softmax dans les Convolutional Neural Networks (CNNs).

### 4.2.3 Optimisation de notre fonction objectif

**Proposition 4.1.** Soit  $\mathbb{1}_{nc}$ ,  $\mathbb{1}_K$  deux vecteurs de  $nc$  et  $K$  uns respectivement. Les conditions d'optimalité de l'équation 4.1 conduisent à la solution :

$$\mu^{(\tau+1)} := \frac{\hat{\mu}^{(\tau+1)}}{\|\hat{\mu}^{(\tau+1)}\|_1}, \quad (4.2)$$

avec :

$$\hat{\mu}^{(\tau+1)} = \exp \left( -\frac{1}{\gamma} [(\mathbf{D} \circ \mathbf{C})\mathbb{1}_K + \alpha \mathbf{C}(\log[\mathbf{C}'\mu^{(\tau)}] + \mathbb{1}_K) + \beta(\mathbf{F} \circ \log \mathbf{F})\mathbb{1}_{nc}] \right). \quad (4.3)$$

Les détails de la preuve sont omis et résultent des conditions d'optimalité du gradient de l'équation 4.1.

En considérant la proposition ci-dessus, la solution optimale est obtenue itérativement comme un point fixe de l'équation 4.2 et l'équation 4.3 avec  $\hat{\mu}^{(0)}$  initialement fixé à des valeurs aléatoires. Notons que la convergence est observée en pratique en quelques itérations, et que le point fixe sous-jacent, noté  $\tilde{\mu}$ , correspond aux échantillons les plus informatifs dans l'affichage  $\mathcal{D}_{t+1}$  (selon le critère de l'équation 4.1) utilisé pour entraîner le classifieur subséquent  $f_{t+1}$  (Algorithme 4.1).

### 4.2.4 Partitionnement multiple de données

Nous introduisons une légère variante de la fonction objectif susmentionnée qui repose sur le partitionnement multiple de données (*multi-clustering*). Ce dernier permet de surmonter la sensibilité de notre modèle par rapport aux valeurs surestimées ou sous-estimées de  $K$  qui peuvent ne pas refléter la dispersion réelle des données. En effet, si l'on considère un partitionnement grossier (avec  $K$  sous-estimé), alors le second terme de l'équation 4.1 peut ne pas être suffisant pour capturer des détails subtils dans la diversité des données non explorées. De plus, une co-pénalisation peut apparaître dans les valeurs de  $\mu$  des données appartenant aux mêmes clusters à granularité grossière. Alors que le partitionnement à granularité fine (avec un  $K$  surestimé) empêche la co-pénalisation, il ne permet pas de distinguer les clusters inexplorés proches et lointains. Par conséquent, il faut mesurer la diversité en prenant en compte non seulement les partitions à granularité fine mais aussi celles à granularité grossière, afin de capturer la diver-

**Procédure 4.1** Mécanisme de sélection d’affichage.

---

```

1: procédure AL_FLAT( $\mathcal{S}, \mathcal{D}_0, T, B$ ) :
2: Entrée :  $\mathcal{S}$  : images,  $\mathcal{D}_0 \subset \mathcal{S}$  : affichage initial,  $T$  : budget,  $B$  : nombre d’étiquettes par cycle.
3: Sortie :  $\cup_{t=0}^{T-1} (\mathcal{D}_t, \mathcal{Y}_t)$  et  $\{f_t\}_t$ .
4:   for  $t := 1$  to  $T - 1$  do
5:      $\mathcal{Y}_t \leftarrow \mathcal{C}(\mathcal{D}_t)$  // ORACLE
6:      $f_t \leftarrow \arg \min_f P(f(X) \neq Y)$  // MODÈLE D’APPRENTISSAGE
       (CONSTRUIT AVEC  $\cup_{k=0}^t (\mathcal{D}_k, \mathcal{Y}_k)$ )
7:      $\hat{\mu}^{(0)} \leftarrow$  aléatoire ;  $\mu^{(0)} \leftarrow \frac{\hat{\mu}^{(0)}}{\|\hat{\mu}^{(0)}\|_1}$  ;  $\tau \leftarrow 0$ 
8:     while ( $\|\mu^{(\tau+1)} - \mu^{(\tau)}\|_1 \geq \epsilon \wedge \tau < \text{maxiter}$ ) do
9:        $\mu^{(\tau+1)}$  (cf. équation 4.2 et équation 4.3) // MODÈLE D’AFFI-
       CHAGE
10:       $\tau \leftarrow \tau + 1$ 
11:    end while
12:     $\tilde{\mu} \leftarrow \mu^{(\tau)}$ 
13:     $\mathcal{D}_{t+1} \leftarrow \{\mathbf{x}_i \in \mathcal{S} \setminus \cup_{k=0}^t \mathcal{D}_k : \tilde{\mu}_i \in \mathcal{L}_B(\tilde{\mu})\}$  //  $\mathcal{L}_B(\tilde{\mu})$  LES  $B$  PLUS
       GRANDES VALEURS DE  $\tilde{\mu}$ 
14:  end for
15: return  $\cup_{t=0}^{T-1} (\mathcal{D}_t, \mathcal{Y}_t), \{f_t\}_t$ 
16: end procédure

```

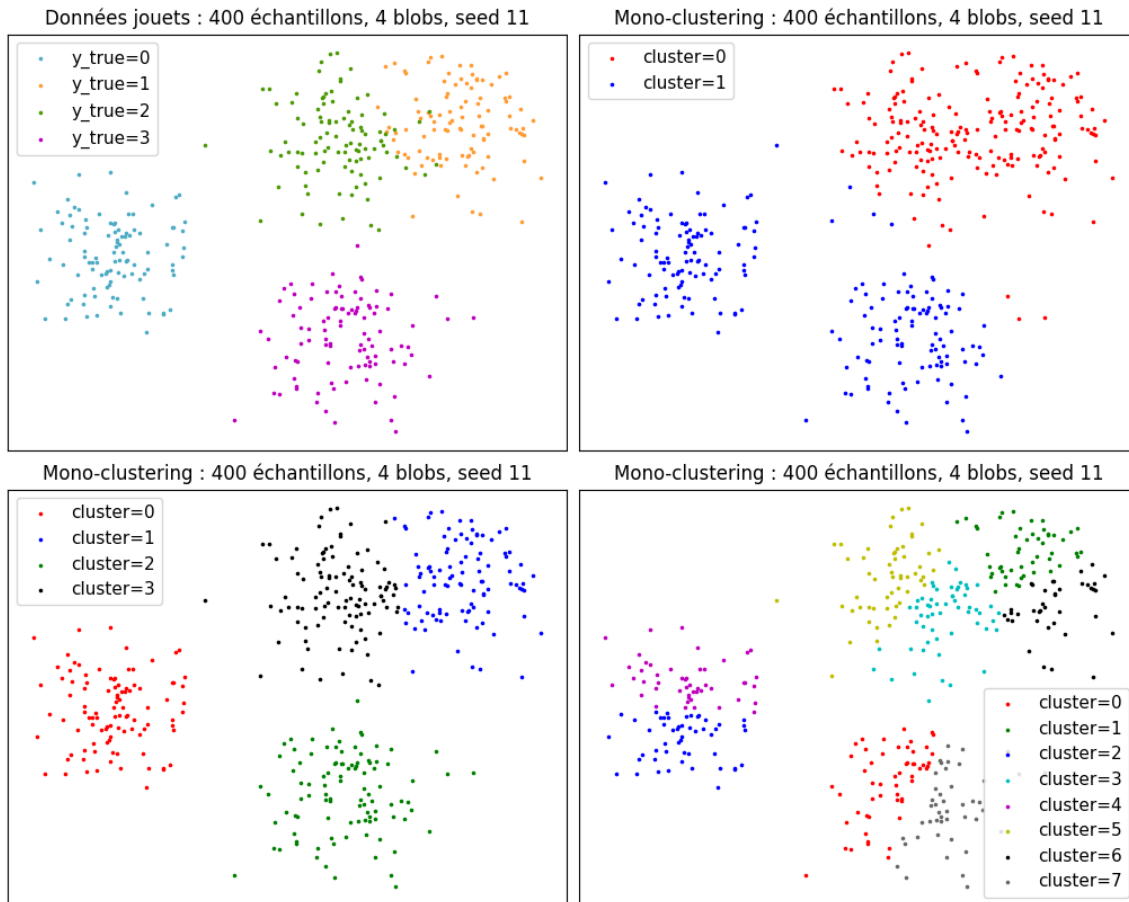
---

sité à différentes granularités. De fait, nous introduisons une variante de notre fonction objectif (équation 4.4).

$$\min_{\mu \geq 0, \|\mu\|_1=1} \sum_{s=1}^S \eta \operatorname{tr}(\operatorname{diag}(\mu'[\mathbf{C}_s \circ \mathbf{D}_s])) + \alpha [\mathbf{C}'\mu]' \log[\mathbf{C}'\mu] + \beta \operatorname{tr}(\operatorname{diag}(\mu'[\mathbf{F} \circ \log \mathbf{F}])) + \gamma \mu' \log \mu, \quad (4.4)$$

Nous observons sur la figure 4.1 une illustration du mono-partitionnement pour mieux comprendre l’intérêt du partitionnement multiple : les données synthétiques appartiennent à 4 classes et l’algorithme K-moyennes partitionne les données en respectivement 2, 4 et 8 lots. On a ainsi une granularité grossière avec 2 lots, moyenne avec 4 et fine avec 8 lots. Lorsque nous utilisons le partitionnement multiple dans notre fonction objectif, nous pouvons ainsi prendre en compte différents niveaux de granularité.

FIGURE 4.1 – **Illustration du partitionnement multiple des données.** En haut à gauche (resp. droite), les données originales avec 4 classes distinctes (resp. droite le mono-partitionnement avec  $K = 2$ ). En bas à gauche (resp. droite), le mono-partitionnement avec  $K = 4$  (resp.  $K = 8$ ).



## 4.2.5 Différentes distances et régularisations

### 4.2.5.1 Comparaison de distances pour le calcul de la représentativité

En plus de l'ajout du partitionnement de données multiples, nous avons également essayé une variante de notre fonction objectif qui utiliserait une autre distance que la distance euclidienne dans le calcul de la matrice  $\mathbf{D}$  où chaque élément de la matrice représente la distance entre une donnée  $x_i$  et le  $k^{\text{ème}}$  centroïde de partition. Ainsi, nous avons testé les distances de Manhattan, euclidienne, Chebyshev et Mahalanobis.

- **Distance de Manhattan** :  $\left( \sum_{i=1}^n |x_i - y_i| \right)$  cette distance est également appelée distance *City Block* ou *Taxicab*, car elle correspond au plus court chemin que prendrait un taxi pour parcourir des pâtés de maisons organisés en grille.
- **Distance Euclidienne** :  $\sqrt{\sum_{i=1}^n (x_i - y_i)^2}$  la distance la plus couramment utilisée comme mesure de similarité, il s'agit de la longueur du segment reliant deux points pour un espace à deux dimensions. Les données doivent être normalisées pour plus d'efficacité, et l'espace des données ne doit pas être de trop grande dimension.
- **Distance de Chebyshev** :  $\left( \max_i |x_i - y_i| \right)$  il s'agit de la différence maximum entre deux vecteurs selon n'importe quelle dimension, donc la distance maximum pour un axe donné.
- **Distance de Mahalanobis** :  $\sqrt{(x - y)^t V^{-1} (x - y)}$  avec  $V$  la matrice de covariance de  $x$  et  $y$ . Cette distance a l'avantage d'accorder moins d'importance aux composantes des vecteurs les plus dispersées, et est donc plus résiliente aux valeurs aberrantes.

#### 4.2.5.2 Régularisation quadratique plutôt qu'entropique.

La régularisation quadratique, que l'on a utilisée à la place de la régularisation entropique, sur le terme de diversité, qui appliquait  $\log$  à la matrice  $C'$  permet également de trouver une solution itérative à notre fonction objectif, nous étudions donc l'impact que cela a sur les performances. Ici, on aura comme solution de la fonction objectif :

$$\hat{\mu}^{(t+1)} = \exp \left( -\frac{1}{\gamma} \left[ \eta(\mathbf{D} \circ \mathbf{C}) \mathbf{1}_K + \alpha \mathbf{C}(\mathbf{C}' \mu^{(t)}) + \beta(\mathbf{F} \circ \log \mathbf{F}) \mathbf{1}_{nc} \right] \right) \quad (4.5)$$

Avec :

$$\mu^{(t+1)} = \frac{\hat{\mu}^{(t+1)}}{\|\hat{\mu}^{(t+1)}\|_1} \quad (4.6)$$

## 4.3 Expériences en classification d'images

Nous avons réalisé les expériences de notre algorithme en utilisant quatre jeux de données : MNIST, CIFAR-10, CIFAR-100 et Object-DOTA. Une description détaillée de ces jeux de données est disponible dans la (section 3.4).

## 4.3.1 Implémentation et *baselines*

### 4.3.1.1 Détails d'implémentation

Les performances sont rapportées sur les données de test en utilisant la précision moyenne par rapport à des pourcentages différents et croissants de données d'entraînement étiquetées. Ces performances sont obtenues en calculant la moyenne de la précision au cours de plusieurs exécutions (c'est-à-dire des sessions d'entraînement d'un CNN), chacune d'entre elles durant au moins 30 époques (*epochs*) avant d'observer la convergence. Notez que chaque session correspond à une division aléatoire des données d'entraînement originales en 10% pour la validation et 90% pour l'apprentissage. L'ensemble de validation est utilisé afin de rechercher les meilleurs paramètres de notre modèle d'apprentissage actif, à savoir  $\alpha$ ,  $\beta$ ,  $\eta$  et  $\gamma$ . Nous considérons dans nos expériences différentes instances ResNet-18 entraînées à partir de zéro en utilisant soit l'optimiseur *Adam*, soit *SGD*. (basé sur l'ordonnanceur Cosine-Annealing-LR avec  $T_{\max} = 200$ ). Le taux d'apprentissage pour *Adam* est de 0,001 et de 0,01 pour *SGD*, avec un momentum fixé à 0,9 et une décroissance des poids du modèle égale à  $5e-4$ . Dans ces expériences, les meilleurs paramètres (observés) de notre modèle d'apprentissage actif, sur l'ensemble de validation, correspondent à  $\alpha = 0.01$ ,  $\beta = 0.01$  et  $\gamma = \eta = 1$ .

### 4.3.1.2 Présentation des différentes baselines

Afin d'étudier la pertinence de notre méthode, nous comparons le modèle d'affichage proposé à deux stratégies de référence différentes, à savoir les échantillonnages par incertitude et aléatoire. Nous comparons également les performances par rapport à l'apprentissage entièrement supervisé, c'est-à-dire en utilisant le jeu de données entièrement étiqueté.

**Échantillonnage aléatoire.** Cette stratégie consiste à prélever des échantillons aléatoires dans le pool de données d'apprentissage non étiquetées. En plus d'être simple, cette stratégie est également difficile à battre dans l'apprentissage actif et particulièrement dans l'apprentissage actif profond dépendant du jeu de données (MAYER et TIMOFTE 2020). Malgré l'utilisation de modèles pré-entraînés, des performances décentes sont généralement observées sur des ensembles de données avec une fraction relativement faible de données étiquetées. Cependant, en pratique, ces performances dépendent fortement de la distribution des données choisies.

**Échantillonnage par incertitude.** Cette stratégie consiste à choisir les données dont les échantillons non étiquetés sont les plus ambigus ; par exemple, les données proches de la frontière de décision sont souvent sélectionnées, et jugées plus précieuses que les données qui en sont éloignées. L'un des inconvénients de cette stratégie est que la plupart des données incertaines peuvent être très similaires les unes aux autres, ce qui nuit à la diversité.

**Apprentissage entièrement supervisé.** Cette stratégie nous fournit une limite supérieure des performances d'une tâche de classification donnée, lorsque toutes les données d'apprentissage avec leurs étiquettes sont utilisées.

### 4.3.2 Le choix de la distance a un impact sur les performances

Nous expérimentons l'utilisation de différentes distances pour le calcul de la matrice  $\mathbf{D}$  de notre fonction objectif ( $\mathbf{D} \in \mathbb{R}^{n \times K}$  et  $\mathbf{D}_{ik} = d_{ik}^2$  est la distance {manhattan, euclidienne, chebyshev, mahalanobis} entre  $\mathbf{x}_i$  et  $k^{\text{ème}}$  centroïde de partition) avec régularisation entropique ou quadratique. Ainsi nous observons dans le [tableau 4.1](#) que la distance de Mahalanobis est plus performante que les autres en utilisant de 100 à 1000 étiquettes avec la régularisation entropique, mais à un niveau de performances similaire aux distances de Manhattan et euclidienne pour la régularisation quadratique. A noter, les résultats reportés dans le [tableau 4.1](#) proviennent d'un seul et même essai. De 1100 à 2000 étiquettes, la distance de Mahalanobis est plus performante que les autres à chaque cycle d'apprentissage actif pour la régularisation quadratique, et à tous les cycles excepté le 8<sup>e</sup> avec la régularisation entropique. Une comparaison du nombre de cycles d'apprentissage actif où la régularisation entropique est plus performante que la régularisation quadratique (en considérant la distance avec le meilleur score d'accuracy pour ce cycle) nous permet d'observer que la régularisation quadratique est meilleure (6 fois sur 10) entre 100 et 1000 étiquettes. La régularisation entropique domine cependant pour chacun des 10 cycles d'apprentissage actif entre 1100 et 2000 étiquettes face à la régularisation quadratique. Avec la régularisation entropique pour 2000 à 20000 étiquettes, le [tableau 4.2](#) montre que la distance euclidienne est la plus performante avec une légère marge, et la distance de Manhattan est plus performante entre 5000 et 45000 étiquettes. Les résultats sont néanmoins très proches et nous privilégions ainsi la distance euclidienne par défaut. Le [tableau 4.3](#) montre quant à lui que la distance de Manhattan est presque aussi performante que la distance euclidienne entre 2000 et 20000 étiquettes, et plus souvent supérieure aux autres distances entre 5000 et 45000 étiquettes, encore une fois avec une faible marge.



TABLE 4.1 – CIFAR-10, comparaison des distances avec faible régime d’annotation. Avec régularisation entropique (*entr.*) ou quadratique (*quad.*). ResNet-18, accuracy (%) pour 1 essai, en fonction du nombre de données annotées.  $\alpha = \beta = \eta = \gamma = 1$

|             |       |              |              |              |              |              |              |              |              |              |              |
|-------------|-------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| Distance    | Regu. | 100<br>0.2%  | 200<br>0.4%  | 300<br>0.6%  | 400<br>0.8%  | 500<br>1%    | 600<br>1.2%  | 700<br>1.4%  | 800<br>1.6%  | 900<br>1.8%  | 1000<br>2%   |
| Manhattan   | entr. | 33.65        | <b>42.9</b>  | 46.07        | 36.94        | 52.62        | 53.0         | 56.0         | 50.19        | 27.42        | 49.22        |
| Euclidean   | entr. | 32.47        | 42.8         | 45.21        | 37.01        | 51.94        | 53.17        | 54.89        | 51.33        | 26.01        | 47.1         |
| Chebyshev   | entr. | 32.62        | 42.21        | 45.6         | 37.78        | 52.15        | 53.22        | 55.12        | <b>52.04</b> | 23.26        | 46.04        |
| Mahalanobis | entr. | <b>34.01</b> | 42.21        | <b>46.37</b> | <b>39.28</b> | <b>53.42</b> | <b>53.6</b>  | <b>56.05</b> | 51.08        | <b>33.09</b> | <b>55.01</b> |
| Distance    | Regu. | 1100<br>2.2% | 1200<br>2.4% | 1300<br>2.6% | 1400<br>2.8% | 1500<br>3%   | 1600<br>3.2% | 1700<br>3.4% | 1800<br>3.6% | 1900<br>3.8% | 2000<br>4%   |
| Manhattan   | entr. | 50.45        | 51.69        | 53.69        | 58.16        | 58.89        | 59.1         | 58.93        | 52.31        | 62.32        | 63.13        |
| Euclidean   | entr. | 49.19        | 49.93        | 52.53        | 56.15        | 57.26        | 57.71        | 57.52        | <b>52.39</b> | 60.78        | 61.98        |
| Chebyshev   | entr. | 48.55        | 50.91        | 53.11        | 56.91        | 58.04        | 58.79        | 58.6         | 48.69        | 62.03        | 62.42        |
| Mahalanobis | entr. | <b>55.27</b> | <b>56.68</b> | <b>58.49</b> | <b>61.19</b> | <b>61.93</b> | <b>62.23</b> | <b>61.38</b> | 51.57        | <b>64.03</b> | <b>64.56</b> |
| Distance    | Regu. | 100<br>0.2%  | 200<br>0.4%  | 300<br>0.6%  | 400<br>0.8%  | 500<br>1%    | 600<br>1.2%  | 700<br>1.4%  | 800<br>1.6%  | 900<br>1.8%  | 1000<br>2%   |
| Manhattan   | quad. | 33.3         | 42.87        | 45.98        | 35.57        | 53.46        | <b>54.31</b> | <b>56.04</b> | <b>53.34</b> | 21.89        | 42.51        |
| Euclidean   | quad. | <b>34.04</b> | 42.7         | 45.88        | <b>38.68</b> | <b>53.53</b> | 54.11        | 56.02        | 52.7         | 25.05        | 46.59        |
| Chebyshev   | quad. | 33.86        | 42.52        | <b>46.49</b> | 38.67        | 52.52        | 53.11        | 55.18        | 52.0         | 26.0         | 45.57        |
| Mahalanobis | quad. | 33.65        | <b>43.03</b> | 46.06        | 37.03        | 53.21        | 53.43        | 56.01        | 51.42        | <b>28.19</b> | <b>50.88</b> |
| Distance    | Regu. | 1100<br>2.2% | 1200<br>2.4% | 1300<br>2.6% | 1400<br>2.8% | 1500<br>3%   | 1600<br>3.2% | 1700<br>3.4% | 1800<br>3.6% | 1900<br>3.8% | 2000<br>4%   |
| Manhattan   | quad. | 44.76        | 46.87        | 47.86        | 52.34        | 53.25        | 53.34        | 51.92        | 48.82        | 57.47        | 57.97        |
| Euclidean   | quad. | 48.24        | 51.99        | 53.56        | 57.16        | 58.15        | 59.06        | 58.6         | 47.59        | 61.37        | 62.54        |
| Chebyshev   | quad. | 46.65        | 47.71        | 52.21        | 54.94        | 55.92        | 56.35        | 56.1         | 49.98        | 59.2         | 60.3         |
| Mahalanobis | quad. | <b>51.9</b>  | <b>53.08</b> | <b>55.26</b> | <b>58.65</b> | <b>58.89</b> | <b>59.41</b> | <b>59.57</b> | <b>51.7</b>  | <b>62.12</b> | <b>63.13</b> |

TABLE 4.2 – CIFAR-10, comparaison des distances avec la régularisation entropique (haut régime d’annotation). Avec régularisation entropique (*entr.*). ResNet-18, accuracy (%) pour 5 essais, en fonction du nombre de données annotées.  $\alpha = \beta = \eta = \gamma = 1$

|             |       |              |              |              |              |              |              |              |              |              |              |
|-------------|-------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| Distance    | Regu. | 2000<br>4%   | 4000<br>8%   | 6000<br>12%  | 8000<br>16%  | 10000<br>20% | 12000<br>24% | 14000<br>28% | 16000<br>32% | 18000<br>36% | 20000<br>40% |
| Manhattan   | entr. | 65.01        | 71.11        | 73.8         | 72.75        | 74.14        | <b>77.17</b> | 78.79        | 76.29        | 80.45        | 80.37        |
| Euclidean   | entr. | 65.37        | 71.36        | <b>74.25</b> | <b>73.42</b> | 74.68        | 76.95        | 78.78        | <b>77.14</b> | 80.54        | <b>80.84</b> |
| Chebyshev   | entr. | 64.94        | 70.78        | 73.6         | 72.33        | <b>74.97</b> | 77.0         | <b>79.07</b> | 76.37        | 80.15        | 80.78        |
| Mahalanobis | entr. | <b>65.45</b> | <b>71.38</b> | 73.84        | 73.1         | 74.14        | 77.07        | 78.98        | 76.83        | <b>80.56</b> | 80.51        |
| Distance    | Regu. | 5000<br>10%  | 10000<br>20% | 15000<br>30% | 20000<br>40% | 25000<br>50% | 30000<br>60% | 35000<br>70% | 40000<br>80% | 45000<br>90% | full<br>100% |
| Manhattan   | entr. | 66.5         | 75.37        | 76.85        | 80.4         | <b>82.95</b> | <b>83.02</b> | 81.35        | 84.53        | 81.85        | 95.03        |
| Euclidean   | entr. | <b>67.49</b> | 75.05        | 77.01        | 80.73        | 82.88        | 83.0         | <b>82.01</b> | 84.53        | 82.12        | 95.03        |
| Chebyshev   | entr. | 66.12        | <b>75.65</b> | 77.19        | 80.74        | 82.92        | 82.67        | 81.84        | 84.61        | <b>82.21</b> | 95.03        |
| Mahalanobis | entr. | 67.13        | 75.28        | <b>77.27</b> | <b>80.83</b> | 82.79        | 82.89        | 81.59        | <b>84.73</b> | 81.86        | 95.03        |

TABLE 4.3 – CIFAR-10, comparaison des distances avec la régularisation quadratique (haut régime d'annotation). Avec régularisation quadratique (quad.). ResNet-18, accuracy (%) pour 5 essais, en fonction du nombre de données annotées.  $\alpha = \beta = \eta = \gamma = 1$

| Distance    | Regu. | 2000<br>4%   | 4000<br>8%   | 6000<br>12%  | 8000<br>16%  | 10000<br>20% | 12000<br>24% | 14000<br>28% | 16000<br>32% | 18000<br>36% | 20000<br>40% |
|-------------|-------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| Manhattan   | quad. | 65.1         | 71.2         | 74.0         | <b>72.95</b> | 74.82        | <b>77.58</b> | 78.91        | 76.64        | <b>80.54</b> | 80.68        |
| Euclidean   | quad. | <b>65.58</b> | <b>71.6</b>  | <b>74.02</b> | 72.72        | 74.68        | 76.92        | <b>78.95</b> | 76.6         | 80.31        | <b>80.73</b> |
| Chebyshev   | quad. | 65.2         | 71.23        | 73.63        | 72.66        | 74.67        | 77.07        | 78.93        | <b>77.05</b> | 80.27        | 80.57        |
| Mahalanobis | quad. | 65.17        | 71.37        | 73.75        | 72.72        | <b>75.03</b> | 77.22        | 78.87        | 76.73        | 80.08        | <b>80.73</b> |
| Distance    | Regu. | 5000<br>10%  | 10000<br>20% | 15000<br>30% | 20000<br>40% | 25000<br>50% | 30000<br>60% | 35000<br>70% | 40000<br>80% | 45000<br>90% | full<br>100% |
| Manhattan   | quad. | <b>67.92</b> | <b>75.89</b> | 76.49        | <b>80.73</b> | 82.85        | <b>83.25</b> | <b>81.87</b> | 84.58        | <b>82.45</b> | 95.03        |
| Euclidean   | quad. | 66.4         | 74.93        | <b>77.17</b> | 80.61        | 82.9         | 83.01        | 81.79        | 84.42        | 82.11        | 95.03        |
| Chebyshev   | quad. | 65.05        | 75.41        | 76.76        | 80.58        | <b>83.04</b> | 82.9         | 81.67        | <b>84.62</b> | 82.11        | 95.03        |
| Mahalanobis | quad. | <b>67.92</b> | 75.04        | 76.83        | 80.62        | 82.92        | 82.9         | 81.45        | <b>84.62</b> | 81.9         | 95.03        |

### 4.3.3 L'impact du choix du CNN sur la hiérarchie des méthodes

TABLE 4.4 – CIFAR-10 comparaison avec les baselines (ShuffleNet v2). Accuracy moyenne pour 5 essais avec différents nombres de données annotées. (%)

*nb classes* : 10, *optimizer* : *sgd*, *scheduler* : *cosine*, *dropout* : *False*, *lr* : 0.01, *momentum* : 0.9, *weight decay* : 0.0005, *model choice* : *shuffle net v2*, *input size* = 224, *vec length* = 512, *nb clusters list* = [2, 4, 8, 16, 32, 64], *nb epochs* = 30,  $\alpha = \beta = \eta = \gamma = 1$

| Method                  | 200<br>0.4%                       | 400<br>0.8%   | 600<br>1.2%   | 800<br>1.6%   | 1000<br>2%   | 1200<br>2.4%  | 1400<br>2.8%  | 1600<br>3.2%  | 1800<br>3.6%  | 2000<br>4%   |
|-------------------------|-----------------------------------|---------------|---------------|---------------|--------------|---------------|---------------|---------------|---------------|--------------|
| AL (our)                | <b>13.91</b>                      | <b>14.85</b>  | 16.74         | 27.95         | 49.65        | <b>65.66</b>  | 74.72         | 79.24         | 80.95         | 83.27        |
| Uncertainty sampling    | 12.52                             | 12.17         | <b>20.94</b>  | <b>29.17</b>  | <b>51.04</b> | 65.12         | <b>75.3</b>   | 79.76         | <b>81.81</b>  | <b>83.71</b> |
| Random sampling         | 10.4                              | 11.06         | 17.03         | 28.86         | 49.0         | 64.6          | 74.79         | <b>79.77</b>  | 80.99         | 83.61        |
| Method                  | 8200<br>16.4%                     | 8400<br>16.8% | 8600<br>17.2% | 8800<br>17.6% | 9000<br>18%  | 9200<br>18.4% | 9400<br>18.8% | 9600<br>19.2% | 9800<br>19.6% | 10000<br>20% |
| AL (our)                | <b>87.32</b>                      | 87.77         | <b>87.86</b>  | <b>86.46</b>  | <b>87.29</b> | <b>87.68</b>  | 87.71         | 84.79         | <b>85.67</b>  | 85.76        |
| Uncertainty sampling    | 87.02                             | 87.67         | 87.71         | 86.43         | 87.12        | 87.56         | <b>87.86</b>  | <b>85.15</b>  | 85.23         | 85.56        |
| Random sampling         | 87.17                             | <b>87.86</b>  | 87.56         | 86.11         | 87.11        | 87.55         | 87.8          | 84.92         | 85.39         | <b>86.05</b> |
| <b>Fully supervised</b> | Acc. : 93.81 50000 données (100%) |               |               |               |              |               |               |               |               |              |

Nous avons étudié l'impact des différents réseaux sur les performances de notre méthode en comparaison avec les baselines. Avec le réseau ShuffleNet v2, le [tableau 4.4](#) montre que notre méthode est plus performante dans les premiers cycles d'apprentissage actif entre 0.4% et 0.8% de la base, c'est-à-dire avec 200 et 400 images. Lorsque nous utilisons beaucoup plus de données, entre 8200 et

10000 images, toutes les méthodes obtiennent des résultats très proches les uns des autres, à environ 7 points d'écart de l'entièrement supervisé utilisant toutes les étiquettes en un seul cycle d'entraînement du réseau. Les résultats obtenus sur

TABLE 4.5 – **CIFAR-10 comparaison avec les baselines (MobileNet-v2)**. Accuracy moyenne pour 5 essais avec différents nombres de données annotées. (%)

nb classes : 10, optimizer : sgd, scheduler : cosine, dropout : False, lr : 0.01, momentum : 0.9, weight decay : 0.0005, model choice : MobileNet-v2, input size = 224, vec length = 512, nb clusters list = [2, 4, 8, 16, 32, 64], nb epochs = 30,  $\alpha = \beta = \eta = \gamma = 1$

|                         |                                   |               |               |               |              |               |               |               |               |              |
|-------------------------|-----------------------------------|---------------|---------------|---------------|--------------|---------------|---------------|---------------|---------------|--------------|
| Method                  | 200<br>0.4%                       | 400<br>0.8%   | 600<br>1.2%   | 800<br>1.6%   | 1000<br>2%   | 1200<br>2.4%  | 1400<br>2.8%  | 1600<br>3.2%  | 1800<br>3.6%  | 2000<br>4%   |
| AL (our)                | 56.72                             | 68.69         | 68.97         | 76.79         | 78.27        | 75.54         | 76.87         | 77.91         | 77.11         | 82.8         |
| Random sampling         | <b>58.82</b>                      | <b>69.85</b>  | <b>70.34</b>  | <b>77.95</b>  | <b>79.13</b> | <b>76.93</b>  | <b>78.29</b>  | <b>78.49</b>  | <b>78.15</b>  | <b>83.35</b> |
| Method                  | 8200<br>16.4%                     | 8400<br>16.8% | 8600<br>17.2% | 8800<br>17.6% | 9000<br>18%  | 9200<br>18.4% | 9400<br>18.8% | 9600<br>19.2% | 9800<br>19.6% | 10000<br>20% |
| AL (our)                | 86.9                              | 87.74         | 87.77         | 84.85         | 87.02        | <b>87.53</b>  | <b>87.83</b>  | 83.66         | 84.15         | 83.69        |
| Random sampling         | <b>87.0</b>                       | <b>87.84</b>  | <b>88.0</b>   | <b>85.67</b>  | <b>87.03</b> | 87.38         | 87.8          | <b>84.16</b>  | <b>84.66</b>  | <b>84.7</b>  |
| <b>Fully supervised</b> | Acc. : 94.66 50000 données (100%) |               |               |               |              |               |               |               |               |              |

CIFAR-10 avec le réseau MobileNet-v2 sont reportés dans le [tableau 4.5](#), où nous comparons les performances de notre méthode d'apprentissage actif avec l'échantillonnage aléatoire. On remarque que les performances avec peu de données sont d'emblée bien meilleures qu'avec le réseau ShuffleNet v2 utilisé précédemment, et bien que notre méthode ait des performances légèrement moins bonnes que la baseline, les scores d'accuracy restent très proches.

Notre méthode est très performante quel que soit le nombre d'étiquettes avec le ResNet-18 (*cf.* [tableau 4.6](#)), mais les résultats sont encore une fois très proches les uns des autres lorsque nous dépassons 2000 étiquettes. Cela explique l'intérêt de notre méthode pour des cycles d'apprentissage actif avec un faible nombre d'étiquettes de l'ordre de la centaine de données. Nous avons également effectué des comparaisons avec les baselines sur la base CIFAR-100, disponibles dans le [tableau 4.7](#). La baseline d'échantillonnage d'incertitude est la plus performante, et les résultats sont parfois proches les uns des autres, même avec peu d'étiquettes. Dans cette comparaison, les résultats sont très proches de l'échantillonnage aléatoire.

Nous avons choisi d'utiliser un ResNet-18 pour plusieurs raisons : les performances sont très satisfaisantes pour des jeux de données comme CIFAR-10 ou bien Object-DOTA. L'implémentation est très simple grâce à la librairie torchvision de Pytorch qui propose des modèles déjà sérialisés avec la possibilité de charger le modèle déjà pré-entraîné sur ImageNet ou de l'entraîner à partir de

TABLE 4.6 – **CIFAR-10 comparaison avec les baselines (ResNet-18).** *Accuracy moyenne pour 5 essais avec différents nombres d’annotations. (%)*  
*nb classes : 10, optimizer : sgd, scheduler : cosine, dropout : False, lr : 0.01, momentum : 0.9, weight decay : 0.0005, model choice : ResNet-18, input size = 224, vec length = 512, nb clusters list = [2, 4, 8, 16, 32, 64], nb epochs = 30,  $\alpha = \beta = \eta = \gamma = 1$*

| Method                  | 200                               | 400          | 600          | 800          | 1000         | 1200         | 1400         | 1600         | 1800         | 2000         |
|-------------------------|-----------------------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
|                         | 0.4%                              | 0.8%         | 1.2%         | 1.6%         | 2%           | 2.4%         | 2.8%         | 3.2%         | 3.6%         | 4%           |
| AL (our)                | <b>61.31</b>                      | <b>69.99</b> | 71.24        | 78.0         | <b>79.64</b> | 77.83        | 78.64        | <b>80.36</b> | 79.51        | 83.19        |
| Uncertainty sampling    | 59.15                             | 68.83        | 71.09        | 77.56        | 78.7         | 77.6         | 78.45        | 78.76        | 79.14        | <b>83.4</b>  |
| Random sampling         | 60.47                             | 69.01        | 71.74        | <b>78.04</b> | 79.23        | <b>78.14</b> | <b>79.2</b>  | 79.43        | <b>79.67</b> | 82.98        |
| Method                  | 8200                              | 8400         | 8600         | 8800         | 9000         | 9200         | 9400         | 9600         | 9800         | 10000        |
| AL (our)                | <b>87.28</b>                      | 87.53        | <b>87.59</b> | 85.8         | 86.83        | 87.58        | 87.65        | 84.8         | 84.39        | 85.29        |
| Uncertainty sampling    | 87.06                             | <b>87.68</b> | 87.48        | 85.61        | <b>86.84</b> | 87.25        | 87.61        | 84.64        | 84.9         | 85.41        |
| Random sampling         | 87.25                             | <b>87.68</b> | 87.57        | <b>86.09</b> | 86.7         | <b>87.59</b> | <b>87.74</b> | <b>84.94</b> | <b>85.45</b> | <b>85.73</b> |
| <b>Fully supervised</b> | Acc. : 95.03 50000 données (100%) |              |              |              |              |              |              |              |              |              |

TABLE 4.7 – **CIFAR-100 comparaison avec les baselines (ResNet-18).** *Accuracy moyenne pour 5 essais avec différents nombres de données annotées. (%)*  
*nb classes : 100, optimizer : sgd, scheduler : cosine, dropout : False, lr : 0.01, momentum : 0.9, weight decay : 0.0005, model choice : ResNet-18, input size = 224, vec length = 512, nb clusters list = [2, 4, 8, 16, 32, 64], nb epochs = 30,  $\alpha = \beta = \eta = \gamma = 1$*

| Method                  | 100                               | 200          | 300          | 400          | 500          | 600          | 700          | 800          | 900          | 1000         |
|-------------------------|-----------------------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
|                         | 0.2%                              | 0.4%         | 0.6%         | 0.8%         | 1%           | 1.2%         | 1.4%         | 1.6%         | 1.8%         | 2%           |
| AL (our)                | 8.83                              | 12.8         | 14.46        | 17.32        | 23.48        | <b>25.05</b> | 27.6         | 26.94        | 27.4         | 31.71        |
| Uncertainty sampling    | 9.0                               | <b>13.16</b> | <b>15.12</b> | <b>17.51</b> | <b>23.54</b> | 24.85        | <b>28.19</b> | <b>27.81</b> | <b>28.18</b> | <b>32.92</b> |
| Random sampling         | <b>9.38</b>                       | 13.04        | 14.43        | 16.73        | 22.64        | 24.35        | 27.16        | 26.99        | 27.23        | 31.88        |
| <b>Fully supervised</b> | Acc. : 79.51 50000 données (100%) |              |              |              |              |              |              |              |              |              |

zéro. On remarque ainsi sur la figure 4.2 que ResNet-18 est significativement moins complexe que VGG-16, tout en offrant des performances similaires sur ImageNet. Cela nous permet donc de faire plus d’expériences et de nous concentrer sur l’apport de notre méthode en tant que fonction d’acquisition de données, plutôt que sur la partie calcul / deep learning. Un autre modèle que nous avons considéré est MobileNet-v2, cependant ce modèle est beaucoup moins utilisé en pratique dans les autres articles sur l’apprentissage actif. Nous avons utilisé le réseau ResNet-18 sur Object-DOTA pour comparer les performances de notre méthode avec les baselines. Le tableau 4.8 montre que notre méthode a des performances proches de l’échantillonnage aléatoire mais légèrement inférieures de 100 à 1000 échantillons, ce qui correspond à un très faible pourcentage de la base totale, de 0.1% à 1% des données.

FIGURE 4.2 – **Vue d'ensemble des architectures de CNNs populaires.** L'axe des abscisses indique la complexité du modèle en nombre d'opérations flottantes par seconde (1 gigaflops = 1 milliard de flops). Une valeur moins élevée implique un temps de calcul GPU moins élevé. L'axe des ordonnées indique la précision top-1 (%) sur ImageNet, une valeur plus élevée implique une meilleure performance du modèle. Le rayon du cercle indique le nombre de paramètres du modèle. Crédits : (BIANCO et al. 2018)

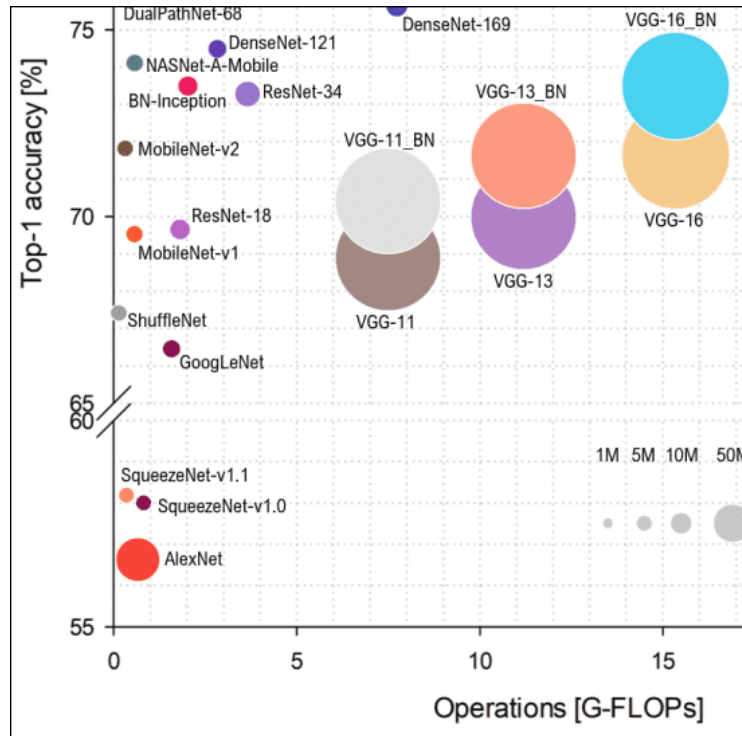


TABLE 4.8 – **Object-DOTA comparaison avec les baselines.** Accuracy moyenne pour 5 essais. *nb classes* : 15, *optimizer* : *sgd*, *scheduler* : *cosine*, *dropout* : *False*, *lr* : 0.01, *momentum* : 0.9, *weight decay* : 0.0005, *model choice* : *ResNet-18*, *input size* = 224, *vec length* = 512, *nb clusters list* = [2, 4, 8, 16, 32, 64], *nb epochs* = 30,  $\alpha = \beta = \eta = \gamma = 1$

| Method                  | 100<br>0.1%                        | 200<br>0.2%  | 300<br>0.3%  | 400<br>0.4%  | 500<br>0.5% | 600<br>0.6%  | 700<br>0.7% | 800<br>0.8%  | 900<br>0.9% | 1000<br>1%   |
|-------------------------|------------------------------------|--------------|--------------|--------------|-------------|--------------|-------------|--------------|-------------|--------------|
| AL (our)                | 50.03                              | 64.71        | 67.99        | 53.55        | 76.85       | 77.0         | 81.37       | 76.0         | 74.17       | 82.68        |
| Uncertainty sampling    | 49.52                              | 66.79        | 70.37        | <b>56.58</b> | 78.05       | 74.05        | <b>81.9</b> | 76.96        | <b>75.8</b> | 83.57        |
| Random sampling         | <b>53.33</b>                       | <b>68.85</b> | <b>71.12</b> | 53.13        | <b>78.8</b> | <b>78.74</b> | 81.78       | <b>77.69</b> | 75.57       | <b>83.67</b> |
| <b>Fully supervised</b> | Acc. : 91.26% 98990 données (100%) |              |              |              |             |              |             |              |             |              |

#### 4.3.4 L'impact individuel et joint de la diversité, représentativité et ambiguïté

Nous étudions l'impact de chaque terme pris séparément et conjointement, sachant que le terme de cardinalité pondéré par  $\gamma$  est toujours conservé car il permet la régularisation et sert à définir la solution analytique (cf. section 4.2.3).

TABLE 4.9 – **MNIST, étude d'ablation 100 à 1000 échantillons.** *Accuracy (%) selon le nombre de données annotées avec différentes valeurs des paramètres de diversité, représentativité ou ambiguïté sur MNIST. Ici, Rep., Amb. et Div. représentent respectivement la représentativité ( $\eta = 1$ ), l'ambiguïté ( $\beta = 1$ ) et la diversité ( $\alpha = 1$ ). La valeur de  $\gamma$  est la même pour toutes les expériences. Ces résultats sont présentés pour différents nombres (ou fractions équivalentes) d'échantillons étiquetés.*

| Composante | 100   | 200          | 300   | 400   | 500   | 600   | 700   | 800   | 900          | 1000         |
|------------|-------|--------------|-------|-------|-------|-------|-------|-------|--------------|--------------|
|            | 0.17% | 0.34%        | 0.51% | 0.68% | 0.85% | 1.02% | 1.19% | 1.36% | 1.53%        | 1.70%        |
| Div.       | 11.35 | 76.35        | 81.96 | 87.19 | 91.41 | 92.21 | 93.1  | 93.56 | 92.76        | 96.08        |
| Amb.       | 11.35 | 77.74        | 82.61 | 87.21 | 90.83 | 91.95 | 92.89 | 93.47 | 88.97        | 96.23        |
| Rep.       | 11.35 | 77.26        | 83.1  | 81.86 | 91.1  | 91.93 | 93.46 | 93.97 | 92.45        | 96.14        |
| Div.+Amb.  | 11.35 | 75.81        | 81.84 | 82.84 | 91.43 | 92.08 | 93.41 | 93.82 | 92.87        | <b>96.28</b> |
| Div.+Rep.  | 11.35 | 77.6         | 82.43 | 86.3  | 90.61 | 91.46 | 93.03 | 93.47 | 92.52        | 96.16        |
| Amb.+Rep.  | 11.35 | 76.59        | 81.73 | 85.97 | 90.75 | 91.92 | 93.34 | 93.84 | <b>94.71</b> | 96.18        |
| All        | 11.35 | <b>78.14</b> | 83.19 | 64.39 | 91.27 | 91.91 | 93.25 | 93.55 | 93.35        | <b>96.28</b> |

Nous observons sur le [tableau 4.9](#) les résultats de cette étude sur la base de données MNIST : notre modèle complet (appelé "All") a connu une baisse de performances lors de deux essais sur cinq au régime de 400 étiquettes, abaissant le score d'accuracy à 64%, alors que dans les autres régimes d'étiquetage, il dépasse les autres configurations plus souvent (trois fois alors que rep. et div.+amb. sont meilleurs que le reste trois fois, et les autres une seule fois). Par conséquent, à la fin du processus itératif, le réglage complet (en particulier lorsque les termes de diversité et d'ambiguïté sont exploités) est clairement avantageux.

Nous réalisons également une étude d'ablation sur CIFAR-10, et les résultats (dans le [tableau 4.10](#)) nous permettent d'observer à nouveau que la diversité et l'ambiguïté donnent de bons résultats, en particulier à la fin du processus itératif (c'est-à-dire à 10000); ces résultats sont conformes à la distribution de CIFAR-10 qui est plus complexe que MNIST, et nécessite donc un effort d'étiquetage plus important afin de capturer précisément la frontière de décision. Nous avons aussi mené une étude complémentaire, en utilisant différentes valeurs de paramètres  $\alpha, \beta, \eta$  allant de 0, 1 à 10000 afin de vérifier si les composantes les plus performantes le restent à ces différentes pondérations. En observant le [tableau 4.11](#), nous constatons que la représentativité n'est pas performante, sauf pour une instance particulière de valeurs de paramètres, ( $\alpha = 1000$  ou  $\beta = 1000$  ou  $\eta = 1000$ ). La diversité apparaît comme la plus performante dans l'ensemble, étant la meilleure dans 24 cas sur 60, contre 21 fois pour l'ambiguïté et 15 fois pour la représentativité. Précision de calcul : nous considérons  $\alpha = a, \beta = a, \eta = a$  avec  $a \in \mathcal{A}$  ( $\mathcal{A} = \{0.1, 1, 10, 100, 1000, 10000\}$ ) comme un seul et même cas, ce qui donne donc pour 10 cycles d'apprentissage actif  $6 * 10 = 60$  cas possibles. Bien que les scores d'accuracy soient relativement proches, il y a encore souvent un

TABLE 4.10 – **CIFAR-10, étude d’ablation 2000 à 10000 échantillons.** Accuracy (%) selon le nombre de données annotées avec différentes valeurs des paramètres de diversité, représentativité ou ambiguïté sur CIFAR-10. Les pondérations sont binaires, 1 si le critère est activé et 0 sinon (donc pour diversité,  $\alpha = 1, \beta = \eta = 0$ , cf. [tableau 4.9](#)). ResNet-18, 5 essais.

| Composante | 2000         | 4000  | 6000  | 8000  | 10000        |
|------------|--------------|-------|-------|-------|--------------|
|            | 4%           | 8%    | 12%   | 16%   | 20%          |
| Div.       | 53.74        | 67.05 | 74.34 | 73.38 | 76.02        |
| Amb.       | 55.71        | 67.49 | 74.85 | 72.2  | 75.2         |
| Rep.       | <b>57.97</b> | 67.99 | 75.0  | 73.69 | 76.36        |
| Div.+Amb.  | 51.68        | 66.02 | 73.61 | 73.65 | <b>77.52</b> |
| Div.+Rep.  | 52.81        | 65.56 | 74.17 | 72.62 | 76.86        |
| Amb.+Rep.  | 55.1         | 66.01 | 73.57 | 66.4  | 76.26        |
| All        | 55.35        | 67.19 | 75.21 | 68.89 | 75.78        |

TABLE 4.11 – **CIFAR-10, étude d’ablation 100 à 1000 échantillons.** Accuracy (%) selon le nombre de données annotées avec différentes valeurs des paramètres de diversité (div.  $\alpha$ ) / ambiguïté (amb.  $\beta$ ) / représentativité (rep.  $\eta$ ). ResNet-18, 5 essais.

| Composante | valeur           | 100          | 200          | 300          | 400          | 500          | 600          | 700          | 800          | 900          | 1000         |
|------------|------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
|            |                  | 0.2%         | 0.4%         | 0.6%         | 0.8%         | 1%           | 1.2%         | 1.4%         | 1.6%         | 1.8%         | 2%           |
| Div.       | $\alpha = 0.1$   | 32.35        | 41.95        | <b>45.56</b> | 35.44        | 51.3         | 51.66        | 54.44        | 49.14        | <b>30.23</b> | <b>50.5</b>  |
| Amb.       | $\beta = 0.1$    | 32.56        | <b>42.86</b> | 45.32        | <b>39.0</b>  | <b>53.13</b> | <b>53.6</b>  | <b>55.12</b> | <b>51.79</b> | 23.88        | 46.87        |
| Rep.       | $\eta = 0.1$     | <b>32.93</b> | 42.27        | 45.45        | 35.98        | 51.79        | 52.06        | 54.27        | 51.15        | 23.29        | 47.37        |
| Div.       | $\alpha = 1$     | 32.13        | <b>42.97</b> | <b>45.87</b> | <b>38.2</b>  | 51.93        | 52.97        | <b>55.7</b>  | 51.42        | 26.01        | 47.11        |
| Amb.       | $\beta = 1$      | <b>32.74</b> | 42.04        | 45.07        | 38.03        | <b>52.54</b> | <b>53.1</b>  | 55.57        | 50.83        | 20.43        | 42.25        |
| Rep.       | $\eta = 1$       | 31.69        | 41.74        | 44.44        | 36.48        | 51.92        | 52.31        | 54.78        | <b>51.73</b> | <b>26.84</b> | <b>47.71</b> |
| Div.       | $\alpha = 10$    | 34.11        | 42.61        | <b>46.8</b>  | <b>38.98</b> | <b>53.51</b> | <b>54.11</b> | <b>56.29</b> | <b>54.02</b> | 24.73        | 47.85        |
| Amb.       | $\beta = 10$     | <b>34.85</b> | <b>43.89</b> | 46.36        | 37.07        | 52.38        | 52.33        | 54.86        | 50.67        | <b>33.58</b> | <b>53.07</b> |
| Rep.       | $\eta = 10$      | 32.19        | 42.1         | 45.04        | 36.16        | 51.5         | 51.77        | 54.28        | 51.21        | 26.18        | 45.78        |
| Div.       | $\alpha = 100$   | <b>34.17</b> | 41.98        | 45.75        | <b>40.49</b> | <b>53.13</b> | 53.51        | 55.78        | 53.09        | <b>30.28</b> | <b>51.51</b> |
| Amb.       | $\beta = 100$    | 32.28        | <b>42.72</b> | <b>46.06</b> | 38.27        | 52.84        | 53.67        | <b>55.99</b> | 52.06        | 27.02        | 48.34        |
| Rep.       | $\eta = 100$     | 32.25        | 41.55        | 45.35        | 38.08        | 52.58        | <b>53.79</b> | 55.87        | <b>53.43</b> | 24.29        | 46.64        |
| Div.       | $\alpha = 1000$  | 33.64        | 42.26        | <b>46.44</b> | 35.28        | 52.19        | 52.85        | 55.6         | 51.5         | 28.14        | 51.04        |
| Amb.       | $\beta = 1000$   | 31.65        | 41.68        | 46.09        | <b>36.61</b> | 52.69        | 53.32        | 55.6         | <b>52.78</b> | 31.8         | 52.37        |
| Rep.       | $\eta = 1000$    | <b>33.89</b> | <b>42.47</b> | 45.92        | 36.43        | <b>53.79</b> | <b>54.7</b>  | <b>56.59</b> | 51.58        | <b>34.14</b> | <b>54.58</b> |
| Div.       | $\alpha = 10000$ | <b>33.91</b> | 42.3         | 46.05        | 38.5         | <b>53.75</b> | <b>54.44</b> | 55.85        | 52.65        | <b>28.31</b> | <b>49.43</b> |
| Amb.       | $\beta = 10000$  | 32.9         | <b>43.19</b> | 45.68        | <b>39.91</b> | 52.58        | 52.96        | 55.57        | <b>53.81</b> | 23.64        | 45.99        |
| Rep.       | $\eta = 10000$   | 33.04        | 42.38        | <b>46.53</b> | 36.48        | 53.09        | 53.5         | <b>56.05</b> | 52.49        | 27.5         | 49.07        |

réel écart de performance entre les différentes composantes, ce qui tend à confirmer que la pondération a un impact. Lorsqu’elles sont prises conjointement, les scores d’accuracy sont généralement dans la marge d’erreur et les performances sont similaires pour toutes les valeurs des paramètres. Additionnellement, le ta-



TABLE 4.12 – **CIFAR-10, nombre de "victoires" des critères.** Nombre de fois où chaque critère a eu le meilleur score d'accuracy selon les valeurs des paramètres (ex. pour  $a = 0.1$  la diversité a eu le meilleur score d'accuracy 3 fois sur 10, l'ambiguïté 6 fois sur 10 et la représentativité une fois sur dix), afin d'observer si chaque critère garde la même importance relative avec différents poids.

| Composante | 0.1            | 1               | 10              | 100             | 1000           | 10000           |
|------------|----------------|-----------------|-----------------|-----------------|----------------|-----------------|
| Div.       | $\frac{3}{10}$ | $\frac{4}{10}$  | $\frac{6}{10}$  | $\frac{5}{10}$  | $\frac{1}{10}$ | $\frac{5}{10}$  |
| Amb.       | $\frac{6}{10}$ | $\frac{3}{10}$  | $\frac{4}{10}$  | $\frac{3}{10}$  | $\frac{2}{10}$ | $\frac{3}{10}$  |
| Rep.       | $\frac{1}{10}$ | $\frac{10}{10}$ | $\frac{10}{10}$ | $\frac{10}{10}$ | $\frac{7}{10}$ | $\frac{10}{10}$ |

bleau 4.12 montre pour les différentes valeurs de  $a$  le nombre de fois où chacune des composantes (diversité, représentativité et ambiguïté) a obtenu le meilleur score d'accuracy. Ainsi, si les composantes gardaient la même supériorité relative quelle que soit leur pondération associée (nommons cela l'hypothèse  $h_0$ ), on aurait par exemple :

- Soit  $D_a, R_a, A_a (\forall a \in \mathcal{A})$  le nombre de fois où respectivement la diversité, représentativité et ambiguïté ont obtenu un score d'accuracy supérieur aux autres composantes.  $h_0 \implies D_i = D_j, R_i = R_j, A_i = A_j \forall i, j \in \mathcal{A} \times \mathcal{A}$ . Ce n'est empiriquement pas le cas, néanmoins une tendance se dessine avec la diversité et l'ambiguïté plus dominantes que la représentativité à l'exception de  $a = 1000$ . Etant donné la nature stochastique du modèle d'apprentissage profond utilisé, il est difficile en pratique de s'assurer d'une reproductibilité parfaite (MUNJAL et al. 2022), et les scores d'accuracy reportés sont parfois dans la marge d'erreur. Cette expérience nous a donc amenés à considérer différentes options de normalisation des composantes. Nous obtenons de bonnes performances avec moins de 1% des étiquettes sur Object-DOTA, notre jeu de données de classification d'images obtenu à partir d'objets du jeu de données DOTA, et globalement notre méthode utilisant une seule composante est la plus performante (cf. tableau 4.13) dans la moitié des cas, tandis que plusieurs critères conjoints ont les meilleures performances dans l'autre moitié des cas. Au début, l'ambiguïté + la représentativité sont meilleures, puis l'ambiguïté seule avant que cela soit partagé entre chacune des méthodes. Les résultats sont encore possiblement dans la marge de variance des réseaux, mais c'est encourageant, car cela signifie que nous pouvons obtenir des performances encore plus élevées en calibrant le poids des composantes dynamiquement et plus précisément, ce que nous faisons dans le prochain chapitre avec succès.



TABLE 4.13 – **Object-DOTA, étude d’ablation 100 à 1000 échantillons.** *Accuracy (%) avec différents nombres d’échantillons annotés sur Object-DOTA selon le critère choisi : diversité (Div.), ambiguïté (Amb.), représentativité (Rep.) ou les trois (All). Les pondérations sont binaires, 1 si le critère est activé et 0 sinon (donc pour diversité,  $\alpha = 1, \beta = \eta = 0$ , cf. tableau 4.9). ResNet-18, 5 essais, 50 epochs.*

| Composante | 100          | 200         | 300          | 400          | 500          | 600          | 700          | 800          | 900          | 1000         |
|------------|--------------|-------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
|            | 0.1%         | 0.2%        | 0.3%         | 0.4%         | 0.5%         | 0.6%         | 0.7%         | 0.8%         | 0.9%         | 1%           |
| Div.       | 54.39        | 65.19       | 73.82        | 76.0         | 77.61        | 76.91        | <b>77.41</b> | 76.24        | 53.17        | 79.68        |
| Amb.       | 56.9         | 69.16       | <b>75.13</b> | <b>77.85</b> | <b>78.07</b> | 78.28        | 76.66        | 77.12        | 55.24        | 79.58        |
| Rep.       | 57.86        | 68.53       | 73.07        | 76.18        | 77.58        | 78.12        | 76.02        | 78.08        | 54.43        | <b>80.23</b> |
| Div.+Amb.  | 58.19        | 66.74       | 73.16        | 75.55        | 77.22        | 77.0         | 76.38        | <b>79.66</b> | 46.81        | 77.01        |
| Div.+Rep.  | 57.51        | 67.39       | 73.06        | 76.58        | 77.36        | 76.1         | 77.11        | 75.16        | 54.94        | 79.43        |
| Amb.+Rep.  | <b>58.51</b> | <b>69.3</b> | 74.17        | 76.47        | 77.83        | <b>78.58</b> | 73.93        | 77.68        | 51.09        | 77.62        |
| All        | 55.05        | 65.85       | 73.17        | 76.43        | 77.52        | 76.63        | 74.27        | 74.79        | <b>55.96</b> | 80.02        |

### 4.3.5 Performances compétitives avec l’état de l’art

Sur MNIST, notre méthode d’apprentissage actif dépasse trois fois celle d’ASAL (MAYER et TIMOFTE 2020), bien que par de petites marges (cf. tableau 4.14), mais ASAL et la nôtre sont dépassées par Core-Set (SENER et SAVARESE 2017). Les

TABLE 4.14 – **Comparaison avec l’état de l’art sur MNIST.** *Nous comparons les performances de notre méthode avec ASAL (MAYER et TIMOFTE 2020) et Core-Set (SENER et SAVARESE 2017) sur MNIST.*

| Method   | 2000      | 4000  | 6000   | 8000   | 10000        |
|----------|-----------|-------|--------|--------|--------------|
|          | 3.40%     | 6.80% | 10.20% | 13.60% | 17%          |
| Our      | 97.32     | 98.12 | 98.68  | 98.84  | 98.91        |
| ASAL     | 97.8      | 98.4  | 98.65  | 98.75  | 98.9         |
| Core-set | <b>98</b> | 98.5  | 98.8   | 98.95  | <b>98.95</b> |

résultats d’ASAL et de Core-set sont tirés de l’article d’ASAL, où ils ont implémenté Core-Set pour leurs comparaisons. Notez que l’architecture CNN utilisée dans ces travaux connexes n’est pas spécifiée dans leur article, mais les faibles performances obtenues (avec la stratégie d’échantillonnage aléatoire) suggèrent que leurs CNNs pourraient être entraînés de manière sous-optimale par rapport aux nôtres. Sur CIFAR-10, notre méthode dépasse ASAL d’une marge significative (cf. tableau 4.15) pour des taux d’étiquetage suffisamment importants (c’est-à-dire lorsque  $\# \text{ labels} \geq 15000$ ), et ce gain est à nouveau perceptible à la fin du processus d’apprentissage itératif. Suite à toutes ces observations préliminaires (en particulier l’étude sur l’ablation), on peut suggérer une procédure adaptative

TABLE 4.15 – **Comparaison avec l’état de l’art sur CIFAR-10.** Nous comparons les performances de notre méthode avec ASAL (MAYER et TIMOFTE 2020) entre 5000 et 30000 étiquettes ainsi qu’avec le fully supervised de 50000 étiquettes. De plus, nous comparons également notre méthode entre 2000 et 12000 étiquettes avec DBAL (GAL et al. 2017), Core-Set (SENER et SAVARESE 2017), ASAL2018 (MAYER et TIMOFTE 2018) et WAAL (SHUI et al. 2020) sur CIFAR-10, d’après les résultats implémentés par WAAL pour toutes les méthodes.

| Method | 5000<br>10% | 10000<br>20% | 15000<br>30% | 20000<br>40% | 25000<br>50% | 30000<br>60% | 50000<br>100% |
|--------|-------------|--------------|--------------|--------------|--------------|--------------|---------------|
| Our    | 61.22       | 71.97        | 78.41        | 86.47        | 88.85        | <b>89.35</b> | 90.71         |
| ASAL   | <b>67</b>   | 74           | 78           | 81.5         | 83.5         | 85           | 88            |

---

| Method   | 2000<br>4% | 4000<br>8% | 6000<br>12% | 8000<br>16% | 10000<br>20% | 12000<br>24% |
|----------|------------|------------|-------------|-------------|--------------|--------------|
| Our      | 55.35      | 67.19      | 75.21       | 68.89       | 75.78        | /            |
| DBAL     | 46         | 56         | 63          | 66          | 69           | 72           |
| Core-Set | 46         | 56         | 59.8        | 66          | 69           | 72.5         |
| ASAL2018 | 46         | 54.5       | 61          | 65          | 69.5         | 71           |
| WAAL     | 55         | 62.5       | 67          | 69.5        | 72.5         | 75           |

qui rend le réglage de  $\alpha, \beta, \eta, \gamma$  “dépendant de l’itération”, de sorte que l’on puisse mettre davantage l’accent sur la “diversité et la représentativité” pendant les premières itérations, puis sur l’ambiguïté lors des itérations ultérieures. Cette extension est étudiée dans le chapitre suivant.

De plus, nous observons dans le [tableau 4.15](#) une comparaison des résultats de notre méthode avec celles de divers papiers sur la base de données CIFAR-10. Le CNN utilisé par WAAL (SHUI et al. 2020) pour l’implémentation des autres méthodes de ce tableau est un VGG-16 qui sert d’extracteur de caractéristiques, tandis qu’un MLP à 2 couches est utilisé comme classifieur et fonction critique. Cette fonction critique sert à évaluer dans (SHUI et al. 2020) la probabilité que l’échantillon provienne d’une partie étiquetée ou non des données. Ils stoppent l’entraînement après 80 epochs (avec des techniques d’arrêt prématuré), et utilisent 64 images par lot lors de l’entraînement du modèle optimisé par SGD, en utilisant la recherche par quadrillage pour le réglage des hyper-paramètres.

Finalement, nous avons également comparé notre méthode d’apprentissage actif avec Core-Set (SENER et SAVARESE 2017) sur la base CIFAR-100, et le [tableau 4.16](#) montre que nous obtenons des performances nettement supérieures à leur méthode couplée au réseau VGG-16.

TABLE 4.16 – **Comparaison avec l’état de l’art sur CIFAR-100.** Nous comparons les performances de notre méthode avec Core-Set (SENER et SAVARESE 2017) sur CIFAR-100.

| Method              | 5000         | 10000 | 15000        | 20000        |
|---------------------|--------------|-------|--------------|--------------|
|                     | 10%          | 20%   | 30%          | 40%          |
| AL (our, ResNet-18) | <b>59.05</b> | 64.94 | <b>67.46</b> | <b>71.42</b> |
| Core-Set (VGG-16)   | 28           | 45    | 50           | 54.5         |

### 4.3.6 L’analyse qualitative confirme la pertinence de notre méthode

La procédure pour sélectionner les images des [figure 4.3](#) et [figure 4.4](#) est la suivante : à chaque itération  $t$  d’apprentissage actif, les images sources correspondant à 4 éléments (arbitrairement les 4 derniers) de l’affichage  $\mathcal{D}_t$  sont récupérées. La partie gauche de la [figure 4.3](#) montre que notre algorithme d’apprentissage actif sélectionne des images provenant de classes diverses du jeu de données CIFAR-10. Certaines imageries d’animaux (par exemple *cycle1\_img3\_cat*, *cycle5\_img1\_cat*, *cycle5\_img2\_bird*, *cycle3\_img0\_deer*, *cycle7\_img3\_dog*) me semblent difficiles à reconnaître pour un humain, ce qui tend à souligner que l’incertitude des données a été capturée par notre algorithme. Nous observons sur la partie droite de la [figure 4.3](#) que l’échantillonnage aléatoire permet également d’obtenir des images de classes diverses, cependant la plupart des images sélectionnées me semblent plus simples à reconnaître pour un humain en moyenne. Cela souligne que l’incertitude des données n’est pas particulièrement capturée par cette *baseline* sinon par hasard (par exemple les images *cycle3\_img1\_cat* *cycle6\_img2\_bird* me semblent difficiles à classifier).

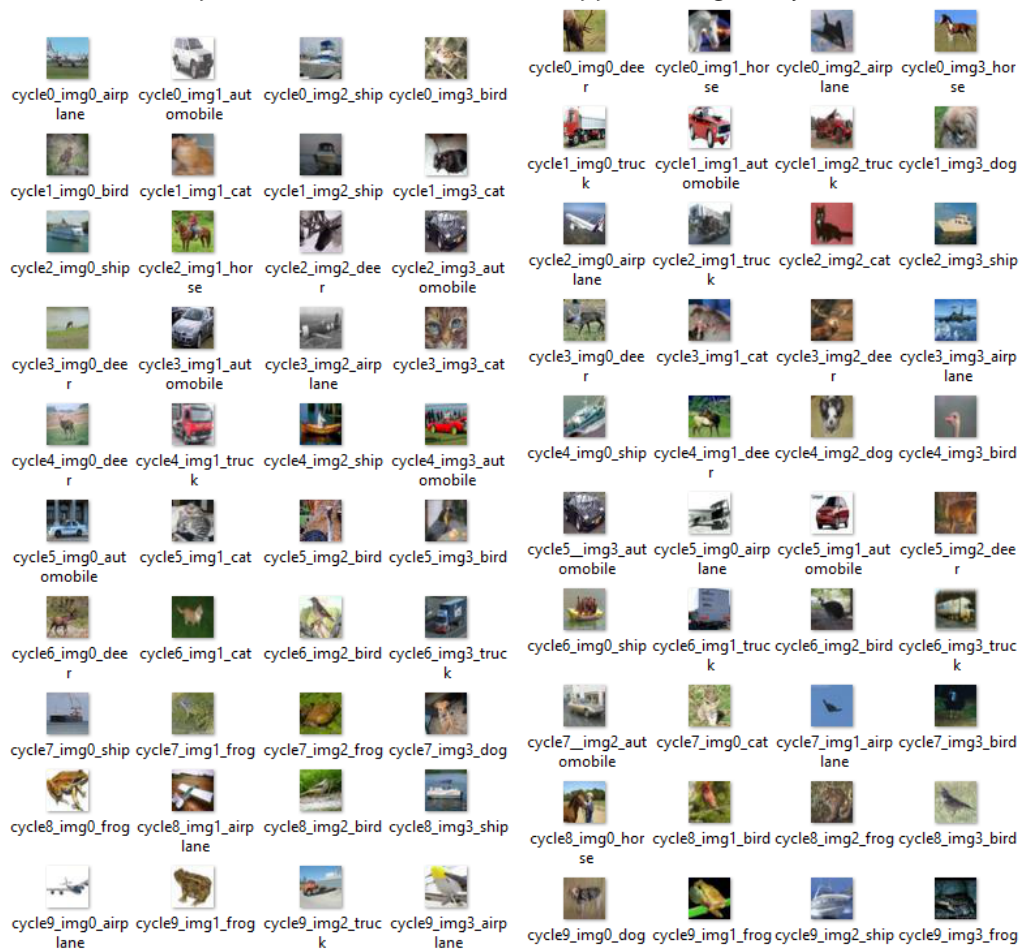
Nous observons sur la partie gauche de la [figure 4.4](#) que sur le jeu de données Object-DOTA, notre algorithme sélectionne des images qui me semblent difficiles à classifier, par exemple *cycle1\_img3\_ship*, *cycle1\_img2\_small-vehicle*, et plus généralement la majorité des classes *ship* me semblent difficiles à distinguer. On remarque la diversité intra-classe avec la présence d’objets de même classe distincts les uns des autres : par exemple, des véhicules de tailles ou couleurs différentes, des bateaux plus ou moins flous, de formes différentes, etc. Nous observons sur la partie droite de la [figure 4.4](#) que l’algorithme d’échantillonnage aléatoire a également sélectionné des images de façon diversifiée, cependant la majorité des images me semblent subjectivement plus simples à classifier.

## 4.4 Expériences en détection de changements

### 4.4.1 Détails d'implémentation et jeux de données

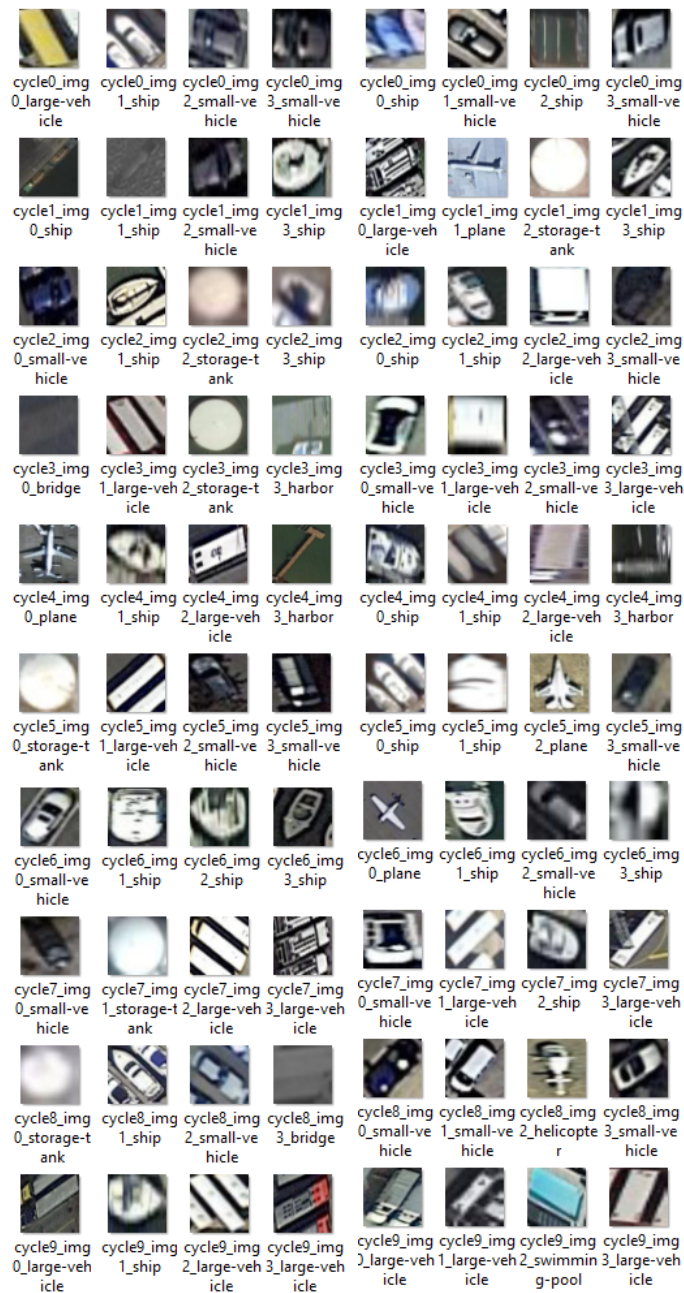
Nous évaluons également les performances de notre méthode sur la tâche particulière de détection des changements en utilisant un jeu de données de 2200 paires de patches non chevauchants (de  $30 \times 30$  pixels en RVB) provenant de deux images satellites GeoEye-1 alignées (référence et test) de  $2400 \times 1652$  pixels avec une résolution spatiale de 1,65m/pixel. Ces images correspondent à la même zone de Jefferson (Alabama) et ont été prises respectivement en 2010 et en 2011, avec de

FIGURE 4.3 – **Imagettes sélectionnées par notre algorithme ou l'aléatoire sur CIFAR-10.** Cette figure montre des exemples d'échantillons tirés des affichages 0 à 9 grâce à notre méthode basée sur l'apprentissage actif (gauche) ou avec l'échantillonnage aléatoire (droite). Chaque rangée correspond ainsi à une itération d'apprentissage actif.



nombreux changements dus aux tornades (destruction de bâtiments, etc.) et sans changement (y compris des changements non pertinents comme les nuages). La vérité terrain sous-jacente contient 2 161 paires de patches négatifs (sans changement ou avec des changements non pertinents) et seulement 39 paires de patches

FIGURE 4.4 – **Imagettes sélectionnées par notre algorithme ou l'aléatoire sur Object-DOTA.** Cette figure montre également des échantillons tirés des pages 0 à 9 grâce à notre méthode basée sur l'apprentissage actif (gauche) ou avec l'échantillonnage aléatoire (droite).





positifs (changements pertinents), donc  $< 2\%$  de ces patchs correspondent à des changements pertinents; la moitié de cet ensemble est utilisée pour construire les modèles d'affichage et d'apprentissage et l'autre moitié pour l'évaluation. Les performances sont rapportées en utilisant un taux d'erreur équilibré (EER) sur l'ensemble d'évaluation de  $\mathcal{S}$ . L'EER est l'erreur de généralisation équilibrée qui pondère de manière égale les erreurs dans les classes "changement" et "pas de changement". Un EER plus petit implique une meilleure performance. En pratique, un CNN basé sur les graphes prend une paire d'images alignées en entrée, et renvoie la représentation de cette paire en sortie. Cette représentation est ensuite utilisée par un classifieur SVM pour déterminer s'il y a un changement ou non : si le score est  $> 0$  il y a un changement pertinent, si le score est  $\leq 0$  il n'y a pas de changement pertinent.

#### 4.4.2 Net avantage de notre méthode face aux baselines

La figure 4.5 montre l'EER en fonction des différentes itérations (et les taux d'échantillonnage sous-jacents dans le tableau 4.17). Nous observons à partir de ces résultats l'impact positif de notre modèle d'affichage par rapport aux autres stratégies. Comme la tâche est fortement déséquilibrée, toutes ces stratégies comparatives ne sont pas en mesure de repérer suffisamment la classe rare (changements); l'aléatoire permet de capturer la diversité des données sans être capable de minimiser l'EER aux dernières itérations. L'incertitude affine localement les fonctions de décision mais souffre du manque de diversité. Alors que toutes ces stratégies (pour  $t \leq 1$ ) ont des EER élevés, notre méthode proposée (lorsqu'elle est combinée avec notre modèle d'affichage) réduit rapidement l'EER et dépasse toutes les autres stratégies, à la fin du processus itératif. Cela provient de l'adaptation rapide des fonctions de décision  $\{f_t(\cdot)\}_t$  à l'oracle lors de l'apprentissage frugale à partir des échantillons les plus pertinents.

#### 4.4.3 Impact de chacun des termes selon le cycle d'apprentissage actif

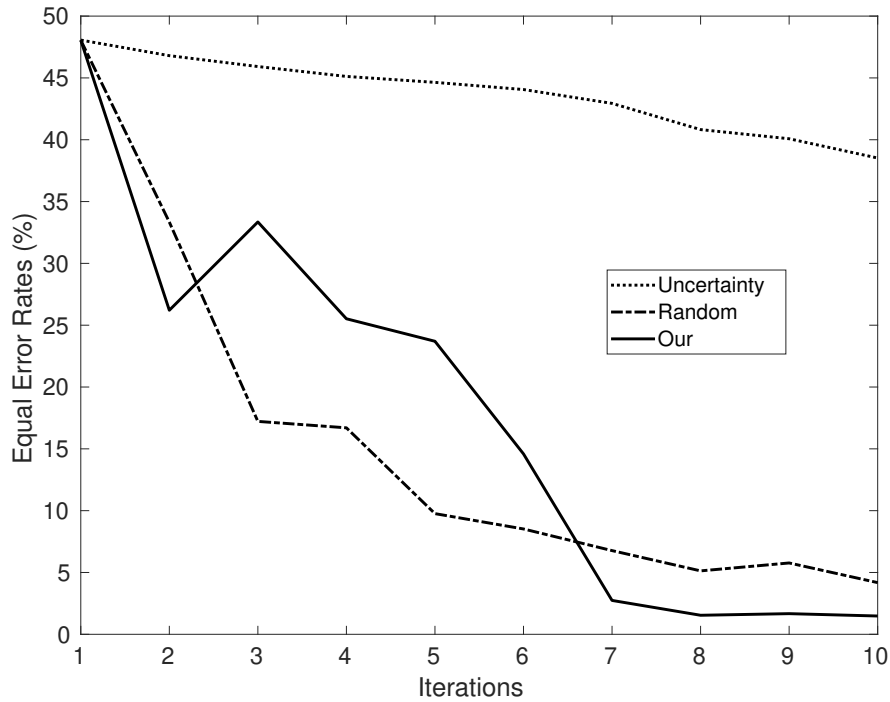
Comme pour les expériences précédentes, nous étudions également l'impact de chaque terme de notre fonction objectif lorsqu'il est pris séparément et conjointement. Le tableau 4.17 montre l'impact de chacun de ces termes pris individuellement, par paire et tous conjointement. Nous observons que la représentativité+diversité sont les critères les plus importants aux premiers stades du processus d'apprentissage itératif, tandis que l'impact du terme d'ambiguïté intervient plus tard, lors des dernières itérations (lorsque les modes de données

TABLE 4.17 – **Etude d’ablation en détection de changements.** Ces résultats sont présentés pour différentes itérations  $t = 0, \dots, T - 1$  (Iter) et les taux d’échantillonnage sous-jacents (Samp) définis comme suit  $(\sum_{k=0}^{t-1} |\mathcal{D}_k| / (|\mathcal{S}|/2)) \times 100$ .  $\gamma = 1$ .

| Iter                                | 1     | 2     | 3     | 4     | 5     | 6     | 7           | 8           | 9     | 10          |
|-------------------------------------|-------|-------|-------|-------|-------|-------|-------------|-------------|-------|-------------|
| Samp%                               | 1.45  | 2.90  | 4.36  | 5.81  | 7.27  | 8.72  | 10.18       | 11.63       | 13.09 | 14.54       |
| Div ( $\alpha = 1$ )                | 48.05 | 31.24 | 23.45 | 30.41 | 44.81 | 24.12 | 13.22       | 17.02       | 6.88  | 7.98        |
| Amb ( $\beta = 1$ )                 | 48.05 | 46.68 | 38.73 | 29.91 | 14.74 | 20.11 | 8.33        | 7.41        | 7.37  | 5.53        |
| Rep ( $\eta = 1$ )                  | 48.05 | 26.21 | 12.72 | 10.48 | 9.88  | 9.70  | 8.52        | 8.85        | 8.61  | 8.82        |
| Div+Amb ( $\alpha = \beta = 1$ )    | 48.05 | 41.69 | 28.82 | 23.08 | 23.41 | 23.42 | 19.82       | 13.10       | 8.16  | 6.97        |
| Div+Rep ( $\alpha = \eta = 1$ )     | 48.05 | 26.21 | 33.35 | 25.10 | 21.55 | 11.71 | 2.84        | 1.65        | 1.59  | <b>1.43</b> |
| Amb+Rep ( $\beta = \eta = 1$ )      | 48.05 | 26.21 | 12.62 | 10.81 | 9.82  | 9.70  | 8.53        | 9.23        | 8.60  | 8.82        |
| All ( $\alpha = \beta = \eta = 1$ ) | 48.05 | 26.21 | 33.35 | 25.52 | 23.70 | 14.59 | <b>2.74</b> | <b>1.54</b> | 1.67  | <b>1.48</b> |

sont tous explorés) afin d’affiner localement les fonctions de décision. Ces observations sont illustrées par l’EER, et pour différents taux d’échantillonnage définis à chaque itération  $t$  comme  $(\sum_{k=0}^{t-1} |\mathcal{D}_k| / (|\mathcal{S}|/2)) \times 100$  avec  $|\mathcal{D}_k|$  fixé à 16 en pra-

FIGURE 4.5 – **Test de référence détection de changements.** Cette figure montre une comparaison de différentes stratégies d’échantillonnage à différentes itérations et leurs taux d’échantillonnage associés (cf. tableau 4.17). L’apprentissage entièrement supervisé obtient un EER de 0.94%.



tique et encore  $|\mathcal{S}| = 2, 200$ . Nous montrons également les performances avec et sans partitionnement multiple dans le [tableau 4.18](#).

TABLE 4.18 – **Comparaison mono et multi-partitionnement en détection de changements.** *Ces résultats sont présentés à l'itération finale et tous les termes de notre fonction objectif sont utilisés.*

| # de partitions | Mono-partition |       |      |             |      |      |      |      |      | Multi-partition<br>All combined |
|-----------------|----------------|-------|------|-------------|------|------|------|------|------|---------------------------------|
|                 | 2              | 4     | 8    | 16          | 32   | 64   | 128  | 256  | 512  |                                 |
| EER (%)         | 12.73          | 10.59 | 9.55 | <b>6.08</b> | 9.09 | 6.42 | 9.13 | 9.68 | 8.78 | <b>1.48</b>                     |

## 4.5 Conclusion

Nous présentons dans ce chapitre un nouvel algorithme d'apprentissage actif interactif pour la classification d'images. La méthode proposée est basée sur un modèle de requête et de réponse qui suggère les données les plus informatives à un oracle et selon les réponses de ce dernier, met à jour un critère de décision qui capture son intention. Nous proposons un modèle d'affichage probabiliste obtenu par la minimisation d'une fonction objectif contrainte mélangeant (i) la diversité qui explore les différents modes de distribution des données, (ii) la représentativité qui se concentre sur les échantillons les plus informatifs dans ces modes et (iii) l'ambiguïté qui renvoie les échantillons avec les classifications les plus ambiguës. Nous considérons également un terme de régularisation qui lisse les probabilités apprises et permet d'obtenir des solutions analytiques. Les expériences menées sur différents jeux de données montrent les performances prometteuses de notre approche.





## SÉLECTION D’AFFICHAGE POUR L’APPRENTISSAGE FRUGAL PAR RENFORCEMENT

### *Résumé chapitre*

*Nous avons introduit dans le chapitre précédent un nouvel algorithme d’apprentissage actif basé sur un modèle de questions et réponses, posant à un annotateur oracle les questions les plus informatives sur la pertinence des échantillons. Nous avons mis à jour notre fonction de décision de manière itérative en fonction des réponses de l’oracle, en minimisant une fonction objectif pour déterminer la probabilité que les échantillons soient pertinents, en tenant compte de la représentativité, de la diversité et de l’ambiguïté des données. Nous présentons maintenant une nouvelle façon d’améliorer notre algorithme, en utilisant l’apprentissage par renforcement pour pondérer dynamiquement et précisément l’importance de la représentativité, de l’ambiguïté et de la diversité à chaque cycle d’apprentissage actif. Les expériences que nous avons effectuées montrent l’efficacité de cette version améliorée de notre algorithme d’apprentissage actif, sur CIFAR-10 et Object-DOTA pour la tâche de classification d’images et sur Jefferson pour la détection des changements dans des images satellites.*

## Sommaire

|       |  |     |
|-------|--|-----|
| 5.1   | Introduction . . . . .   | 110 |
| 5.2   | Méthode proposée . . . . .   | 111 |
| 5.2.1 | Modèle de sélection d’affichage . . . . .                              | 111 |
| 5.2.2 | Optimisation . . . . .   | 113 |
| 5.2.3 | Sélection d’affichage basée sur l’apprentissage par renforcement . . . | 113 |
| 5.2.4 | Différentes fonctions de récompense . . . . .                          | 117 |
| 5.3   | Expériences . . . . .  | 117 |
| 5.3.1 | Détails d’implémentation . . . . .                                     | 118 |
| 5.3.2 | Performances améliorées de notre méthode avec le RL . . . . .          | 118 |
| 5.3.3 | Comparaison avec les baselines . . . . .                               | 120 |
| 5.3.4 | Expériences additionnelles pour affiner les réglages . . . . .         | 122 |
| 5.4   | Conclusion . . . . .   | 128 |

## 5.1 Introduction

Dans le précédent chapitre, nous avons présenté un nouvel algorithme interactif de détection des changements dans les images satellites qui pose à l'utilisateur les questions les plus informatives sur la pertinence des classifications et, en fonction des réponses de l'utilisateur, met à jour une fonction de décision. Nous avons également appliqué cet algorithme à la tâche de la classification d'images sur divers jeux de données comme CIFAR (KRIZHEVSKY et al. 2010), MNIST (LECUN 1998) ou bien l'adaptation de DOTA (XIA et al. 2018) à la classification d'images : Object-DOTA.

Notre modèle est probabiliste et attribue à chaque échantillon d'entraînement non étiqueté une mesure de probabilité qui capture l'importance de cet échantillon pour la mise à jour des critères de décision. Nous obtenons cette probabilité comme l'optimum d'une fonction objectif sous contrainte mélangeant représentativité, diversité, ambiguïté et un terme de cardinalité et régularisation. La plupart des autres solutions d'apprentissage actif sont essentiellement des heuristiques (PINSLER et al. 2019; PRUDÊNCIO et LUDERMIR 2008; RANGANATHAN et al. 2017; DASGUPTA 2004; ARAUJO et al. 2019; SETTLES 2009), tandis que la nôtre est probabiliste et unifie tous les termes susmentionnés en une seule fonction objectif dont la solution modélise la probabilité de pertinence des échantillons lors de l'apprentissage des fonctions de décision.

Dans ce chapitre, nous appliquons à nouveau notre algorithme à la tâche de classification d'images et de détection de changement d'images satellites, mais

nous améliorons notre méthode. En effet, nous utilisons l'apprentissage par renforcement (parfois abrégé RL pour *Reinforcement Learning*) et en particulier une version Q-learning sans état pour choisir dynamiquement et avec précision les paramètres  $\alpha$ ,  $\beta$  et  $\eta$  qui contrôlent respectivement l'importance de la diversité, de l'ambiguïté et de la représentativité des données à chaque itération d'apprentissage actif.

Des expériences approfondies menées sur les jeux de données Object-DOTA pour la classification d'images et Jefferson pour la détection des changements dans des images satellites, montrent l'efficacité de notre modèle proposé par rapport à plusieurs baselines. L'apprentissage par renforcement pour optimiser dynamiquement nos paramètres à chaque étape d'apprentissage actif nous permet d'atteindre de meilleures performances, notamment par rapport aux paramètres statiques ou aux méthodes de recherche aléatoire.

## 5.2 Méthode proposée

Notre méthode est une évolution de notre précédent algorithme, voici donc un bref rappel des notations utilisées dans le cadre de l'apprentissage actif. Soit  $\mathcal{X}$  l'ensemble de toutes les images possibles tirées d'une distribution de probabilité existante mais inconnue  $P(X, Y)$ . La variable aléatoire  $X$  représente une image d'entrée et  $Y$  son étiquette de vérité terrain inconnue. Considérant  $nc$  le nombre de classes visuelles (c'est-à-dire des étiquettes ou catégories), et  $\mathcal{U}$  un large sous-ensemble de  $\mathcal{X}$  dont les étiquettes sont inconnues initialement, notre objectif est de concevoir des classifieurs  $\{g_c\}_{c=1}^{nc}$  en étiquetant de façon interactive une très petite portion de  $\mathcal{U}$ , et d'entraîner les paramètres de  $\{g_c\}_c$ . Cet étiquetage interactif et l'entraînement subséquent correspond à l'apprentissage actif.

Soit  $\mathcal{D}_t$  un affichage ou une page (défini comme sous-ensemble de  $\mathcal{U}$ ) montré à un oracle, l'annotateur expert qui fournit les étiquettes de vérité terrain pour n'importe quel sous-ensemble d'images, à n'importe quelle itération (ou cycle)  $t$  de l'apprentissage actif, et soit  $\mathcal{Y}_t$  les étiquettes associées. L'affichage initial  $\mathcal{D}_0$  est échantillonné aléatoirement uniformément, et utilisé afin d'entraîner les classifieurs suivants en répétant les étapes décrites précédemment (cf. [section 4.2.1](#)).

### 5.2.1 Modèle de sélection d'affichage

Pour rappel, nous considérons un cadre probabiliste qui attribue à chaque échantillon  $\mathbf{x}_i$  un degré d'appartenance  $\mu_i$  qui mesure la probabilité d'appartenance de  $\mathbf{x}_i$  à l'ensemble étiqueté  $D_{t+1}$ ; par conséquent,  $D_{t+1}$  correspondra aux

$\{\mathbf{x}_i\}_i$  non étiquetés ayant les plus fortes appartenances  $\{\mu_i\}_i$ . En considérant  $\mu \in \mathbb{R}^n$  comme un vecteur de ces appartenances  $\{\mu_i\}_i$ , nous proposons de trouver  $\mu = \{\mu_i\}_i$  comme le minimum du problème d’optimisation sous contrainte suivant :

$$\begin{aligned} \min_{\mu \geq 0, \|\mu\|_1=1} \quad & \eta \operatorname{tr}(\operatorname{diag}(\mu'[\mathbf{C} \circ \mathbf{D}])) + \alpha [\mathbf{C}'\mu]' \log[\mathbf{C}'\mu] \\ & + \beta \operatorname{tr}(\operatorname{diag}(\mu'[\mathbf{F} \circ \log \mathbf{F}])) + \gamma \mu' \log \mu, \end{aligned} \quad (5.1)$$

ici,  $\circ, '$  sont respectivement le produit de Hadamard et la transposition de matrice,  $\|\cdot\|_1$  est la norme  $l_1$  et le logarithme est appliqué élément par élément dans la matrice associée. Dans la fonction objectif ci-dessus :

1.  $\mathbf{D} \in \mathbb{R}^{n \times K}$  et  $\mathbf{D}_{ik} = d_{ik}^2$  est la distance euclidienne entre  $\mathbf{x}_i$  et  $k^{\text{ème}}$  centroïde de la partition des données en clusters.
2.  $\mathbf{C} \in \mathbb{R}^{n \times K}$  est la matrice indicatrice dont chaque entrée  $\mathbf{C}_{ik} = 1$  si  $\mathbf{x}_i$  appartient au cluster  $k$ . (0 sinon)
3.  $\mathbf{F} \in \mathbb{R}^{n \times nc}$  est une matrice de scoring des données avec  $\mathbf{F}_{ic} = f_c(\mathbf{x}_i)$ .

Le premier terme de cette fonction objectif (réécrit sous la forme  $\sum_i \sum_k \mathbf{1}_{\{\mathbf{x}_i \in h_k\}} \mu_i d_{ik}^2$ ) mesure la *représentativité* des échantillons sélectionnés dans l’affichage  $\mathcal{D}_t$ ; en d’autres termes, il capture la proximité de chaque  $\mathbf{x}_i$  par rapport au centroïde de son cluster, de sorte que ce terme atteint sa plus petite valeur lorsque tous les échantillons sélectionnés coïncident avec ces centroïdes.

Le deuxième terme (réécrit comme  $\sum_k [\sum_{i=1}^n \mathbf{1}_{\{\mathbf{x}_i \in h_k\}} \mu_i] \log[\sum_{i=1}^n \mathbf{1}_{\{\mathbf{x}_i \in h_k\}} \mu_i]$ ) mesure la *diversité* des échantillons sélectionnés comme l’entropie de la distribution de probabilité des clusters sous-jacents; cette mesure est minimisée lorsque les échantillons sélectionnés appartiennent à des clusters différents et vice-versa.

Le troisième critère (équivalent à  $\sum_i \sum_c \mu_i \mathbf{F}_{ic} \log \mathbf{F}_{ic}$ ) capture l’*ambiguïté* de  $\mathcal{D}_t$  mesurée comme l’entropie de la fonction de scoring; ce terme atteint sa plus petite valeur lorsque les données sont classées de manière égale par rapport aux différentes catégories. Enfin, le quatrième terme est lié à la *cardinalité* de  $\mathcal{D}_t$ , mesurée par l’entropie de la distribution  $\mu$ ; ce terme agit également comme un régularisateur et aide à obtenir une solution analytique. L’impact de ces termes est contrôlé par  $\alpha, \beta, \gamma, \eta \geq 0$ . Nous utilisons un paramètre  $\gamma$  dont le calcul se trouve dans l’équation 5.2.

$$\gamma \propto \|\eta (\mathbf{D} \circ \mathbf{C}) \mathbf{1}_K + \alpha \mathbf{C} (\log[\mathbf{C}'\mu^{(\tau)}] + \mathbf{1}_K) + \beta (\mathbf{F} \circ \log \mathbf{F}) \mathbf{1}_{nc}\|_1 \quad (5.2)$$

## 5.2.2 Optimisation

Les détails d'optimisation de notre fonction objectif se trouvent dans la [section 4.2.3](#), un rappel de l'équation 5.3 et l'équation 5.4 est cependant nécessaire pour la suite.

$$\mu^{(\tau+1)} := \frac{\hat{\mu}^{(\tau+1)}}{\|\hat{\mu}^{(\tau+1)}\|_1}, \quad (5.3)$$

avec :

$$\hat{\mu}^{(\tau+1)} = \exp\left(-\frac{1}{\gamma}[\eta(\mathbf{D} \circ \mathbf{C})\mathbb{1}_K + \alpha\mathbf{C}(\log[\mathbf{C}'\mu^{(\tau)}] + \mathbb{1}_K) + \beta(\mathbf{F} \circ \log \mathbf{F})\mathbb{1}_{nc}]\right). \quad (5.4)$$

Avec  $\mathbb{1}_{nc}$ ,  $\mathbb{1}_K$  deux vecteurs de  $nc$  et  $K$  uns respectivement.  $\eta$  n'est pas nécessaire dans cette équation, néanmoins nous le conservons pour plus de clarté et aussi pour la suite de notre contribution.

Les expériences menées précédemment nous ont aiguillé sur le fait que le réglage des hyper-paramètres  $\alpha, \beta, \eta$  était capital au succès de notre modèle d'affichage. Par exemple, mettre plus d'accent sur la diversité (*ie.* un  $\alpha$  plus élevé) permet une plus grande exploration des modes de classe, tandis que se concentrer sur l'ambiguïté (*ie.* avec une valeur de  $\beta$  plus élevée) permet de mieux affiner localement les fonctions de décision.

Ainsi, un bon équilibre entre exploration et un affinement local des fonctions de décision apprises devrait être obtenu en sélectionnant les meilleures configurations de ces hyper-paramètres. De plus, le paramétrage de ces derniers devrait dépendre du cycle d'apprentissage actif, car les premières itérations  $t$ , celles intermédiaires ou les dernières nécessitent potentiellement des stratégies de sélection d'affichage différentes. En outre, l'étiquetage est fait avec parcimonie à la volée, il est donc impossible d'avoir à l'avance des ensembles de validation supplémentaires pour régler ces hyper-paramètres de façon optimale.

Dans le cas où de tels ensembles de validation seraient disponibles, le réglage de ces hyper-paramètres pour chaque itération  $t$  est très combinatoire et insoluble (*intractable*). En effet, le nombre d'hyper-paramètres augmente linéairement en fonction du nombre maximum d'itérations  $T$ , et le nombre de configurations possibles de ces hyper-paramètres augmente de façon polynomiale en  $\mathcal{O}(p^T)$  où  $p$  est le nombre de configurations possibles testées pour chaque hyper-paramètre.

## 5.2.3 Sélection d'affichage basée sur l'apprentissage par renforcement

Par souci de clarté, nous réécrivons les classifieurs  $\{g_{c,t}\}_c$  entraînés à chaque itération  $t$  comme  $g_t$ . Soit  $\Lambda_\alpha, \Lambda_\beta, \Lambda_\eta$  les espaces de paramètres associés respecti-

vement à  $\alpha, \beta, \eta$ , et soit  $\Lambda$  le produit cartésien sous-jacent. Pour chaque itération suivante  $t+1$ , et pour n’importe quelle instance  $\lambda_{t+1} \in \Lambda$  (noté de façon raccourcie  $\lambda$ ), il est possible d’obtenir un affichage (réécrit  $\mathcal{D}_{t+1}^\lambda$ ) en résolvant l’équation 5.1, et la meilleure configuration  $\lambda^*$  qui donne un affichage optimal peut être définie comme :

$$\lambda^* \leftarrow \arg \min_{\lambda \in \Lambda} \mathcal{R}_{\text{emp}}(g_{t+1}; \mathcal{V}_t) \quad (5.5)$$

où  $\mathcal{V}_t \subset \cup_{\tau=1}^t \mathcal{D}_\tau$  un ensemble de validation obtenu grâce aux anciennes annotations de l’oracle. Il est important de noter que les classifieurs  $\{g_t\}_t$  sont entraînés sur  $\{\cup_{\tau=1}^t \mathcal{D}_\tau\}_t$ , mais ces ensembles d’entraînement ne contiennent pas  $\{\mathcal{V}_t\}_t$  qui sont seulement utilisés pour la validation.  $\mathcal{R}_{\text{emp}}(g_{t+1}; \mathcal{V}_t)$  désigne le risque empirique de  $g_{t+1}$  sur  $\mathcal{V}_t$ .

La résolution de l’équation 5.5 nécessite de générer et annoter de multiples pages  $\mathcal{D}_{t+1}^\lambda$  pour différents  $\lambda$ , et entraîner les classifieurs sous-jacents. Cette équation ne permet pas de trouver la meilleure configuration  $\lambda$  en temps raisonnable, et le régime d’apprentissage frugal implique que le budget pour annoter plusieurs affichages n’est pas toujours suffisant. Enfin, les ensembles  $\{\mathcal{V}_t\}_t$  ne sont pas suffisamment grands en pratique pour avoir un paramétrage de  $\lambda^*$  fiable ce qui peut amener à une mauvaise capacité de généralisation du classifieur.

Afin de passer outre les limitations suscitées, nous considérons un cadre basé sur l’apprentissage par renforcement afin d’entraîner ces hyper-paramètres, en ne considérant pas seulement la récompense immédiate (c’est-à-dire l’accuracy du classifieur actuel) mais les estimations futures de ces récompenses.

Nous modélisons la tâche de réglage des paramètres comme un Markov Decision Process (MDP), et une introduction à ce concept est trouvable dans le livre de (SUTTON et BARTO 2018). Un cadre d’apprentissage par renforcement basé sur un MDP est représenté par un tuple  $\langle S, A, R, q, \delta \rangle$ .

- $S$  – l’ensemble de tous les états.
- $A$  – l’ensemble des actions.
- $R$  – la fonction de récompense immédiate  $R : S \times A \mapsto \mathbb{R}$ .
- $q$  – la fonction de transition  $q : S \times A \times \mathbb{R} \mapsto S$ .
- $\delta$  – le facteur de remise.

L’environnement avec lequel l’agent RL interagit est  $\mathcal{E}$ , en exécutant une action parmi  $A = \{1, \dots, |A|\}$  dans le but de maximiser la récompense escomptée actualisée. L’agent suit une politique stochastique,  $\pi : S \mapsto A$ , qui calcule la vraie valeur état-action comme :

$$Q(s, a) = E_\pi \left[ \sum_{t=0}^{\infty} \delta^t r_t \mid S_0 = s, A_0 = a \right], \quad (5.6)$$

où  $r_t = R(s, a)$  est une récompense immédiate à l'itération  $t$  de l'apprentissage par renforcement,  $S_0$  un état initial,  $A_0$  une action initiale et  $\delta \in [0, 1]$  est un facteur d'actualisation qui équilibre les récompenses immédiates et futures. L'objectif de la politique optimale est de sélectionner les actions qui maximiseront la récompense cumulative actualisée ; ie.  $\pi_*(s) \leftarrow \arg \max_a Q(s, a)$  avec  $Q(s, \pi_*(s))$  la valeur optimale de l'action. L'une des méthodes les plus utilisées pour résoudre ce type de problèmes d'apprentissage par renforcement est le Q-learning (JIN et al. 2018), qui estime directement la fonction de valeur optimale et obéit à l'identité fondamentale, l'équation de Bellman :

$$Q(s, a) \leftarrow (1 - \gamma_{rl})Q(s, a) + \gamma_{rl}(r_t + \delta \max_{a'} Q(s', a')), \quad (5.7)$$

avec  $\gamma_{rl}$  et  $\delta$  respectivement le taux d'apprentissage et le facteur de remise que l'on a fixés en pratique à 0.1 et 0.9 et  $s' = q(s, a)$ . Nous utilisons une version sans état de cet algorithme, donc  $Q(s, a) = Q(a)$  et  $R(s, a) = R(a)$ . Nous considérons deux modes pour  $\Lambda$  explicités ci-après, et l'ensemble  $A$  des actions possibles dépend donc du choix de  $\Lambda$  dans le domaine continu ou discret.

1. *Discret* : Dans ce mode,  $\Lambda$  est égal à  $\{0, 1\}^3 \setminus (0, 0, 0)$ , ainsi l'ensemble  $A$  des actions sous-jacentes correspond aux 7 configurations binaires possibles de  $\alpha, \beta, \eta$ .
2. *Continu* : Dans ce mode,  $\Lambda$  est égal à  $\mathbb{R}^3 \setminus (0, 0, 0)$  et l'ensemble  $A$  des actions sous-jacentes correspond aux 27 possibilités de mise à jour incrémentales de  $\alpha, \beta, \eta$  par trois facteurs multiplicatifs, arbitrairement choisis en pratique parmi  $\{1, 0.95, (0.95)^{-1}\}^3$ .

A chaque itération  $t$ , la récompense  $R(a)$  de chaque action  $a \in A$  est évaluée une fois que l'action aura été exécutée, et l'affichage  $\mathcal{D}_{t+1}$  et classifieur  $g_{t+1}$  sous-jacents auront été entraînés. La récompense  $R(a)$  est mesurée grâce à l'équation 5.5 en utilisant l'accuracy  $1 - \mathcal{R}_{\text{emp}}(g_{t+1}; \mathcal{V}_t)$  de la fonction de décision apprise  $g_{t+1}$  évaluée sur l'ensemble de test : cela est plutôt un ensemble de validation  $\mathcal{V}_t \subset \cup_{\tau=1}^t \mathcal{D}_\tau$  dont la cardinalité n'excède pas (en pratique) 10% des affichages étiquetés par l'oracle. Cet ensemble de validation n'est utilisé que pour l'estimation de la récompense (et donc les mises à jour des hyper-paramètres  $\alpha, \beta, \eta$ ) et non pas pour l'entraînement du classifieur. La fonction de récompense est calculée sur une ou plusieurs itérations appelées épisodes, et la récompense avec la plus grande valeur est conservée. La description détaillée de notre algorithme de sélection d'affichage basé sur l'apprentissage par renforcement se trouve [Algorithme 5.1](#).



---

**Procédure 5.1 Sélection de données grâce à l’apprentissage par renforcement.**  
Optimisation de nos paramètres d’apprentissage actif grâce à l’apprentissage par renforcement.

---

```

1: procédure AL_RL( $\mathcal{U}, \mathcal{D}_0 \subset \mathcal{U}, T, B$ ) :
2: Entrée :  $\mathcal{U}$  : ensemble des images,  $\mathcal{D}_0$  : ensemble annoté initial,  $T$  : budget,  $B$  :
   nombre d’étiquettes par cycle,  $E$  : nombre d’épisodes.
3: Sortie :  $\cup_{t=0}^{T-1} (\mathcal{D}_t, \mathcal{Y}_t)$  : ensembles annotés obtenus,  $\{g_t\}_t$  classifieurs entraînés
   à partir des données annotées.
4:    $\forall$ action,  $Q(\text{action}) \leftarrow \text{rand}$  // RAND  $\in [0, 1]$ 
5:   for  $t = 0$  to  $T - 1$  do
6:      $\mathcal{V}_t \leftarrow \text{RSubset}(\mathcal{D}_t)$  // SOUS-ENSEMBLE ALÉATOIRE
7:      $\mathcal{Y}_t \leftarrow \text{oracle}(\mathcal{D}_t)$  // ANNOTATIONS DE L’ORACLE
8:      $g_t \leftarrow \arg \min_g P(g(X) \neq Y), r_t \leftarrow 0$ 
9:     // MODÈLE D’APPRENTISSAGE (CONSTRUIT SUR  $\cup_{k=0}^t \mathcal{D}_k \setminus \mathcal{V}_k$ )
10:    if  $t \geq 1$  then
11:      for  $e = 0$  to  $E - 1$  do
12:         $r_e \leftarrow \text{calcul\_recompense}(e)$  // RÉCOMPENSE
13:        if  $r_e > r_t$  then
14:           $r_t \leftarrow r_e$ 
15:        end if
16:      end for
17:      Définir  $Q(\text{action\_precedente})$  grâce à l’équation 5.7.
18:    end if
19:     $v \leftarrow \text{rand}$ 
20:    if  $v \leq \exp(-t)$  then
21:      meilleure_action  $\leftarrow$  action_aleatoire de  $A$ 
22:      // EXPLORATION AVEC PROBABILITÉ  $\exp(-t)$ 
23:    else
24:      meilleure_action  $\leftarrow \arg \max_{\text{action}} Q(\text{action})$ 
25:      // SINON PRENDRE LA MEILLEURE VALEUR-Q
26:    end if
27:    action_precedente  $\leftarrow$  meilleure_action
28:     $(\alpha, \beta, \eta) \leftarrow \text{maj}(\text{meilleure\_action}, \alpha, \beta, \eta)$ 
29:     $\hat{\mu}^{(0)} \leftarrow \text{rand}; \mu^{(0)} \leftarrow \frac{\hat{\mu}^{(0)}}{\|\hat{\mu}^{(0)}\|_1}; \tau \leftarrow 0$ 
30:    while  $(\|\mu^{(\tau+1)} - \mu^{(\tau)}\|_1 \geq \epsilon \wedge \tau < \text{maxiter})$  do
31:      Définir  $\mu^{(\tau+1)}$  grâce à l’équation 5.3 et l’équation 5.4
32:      // MODÈLE D’AFFICHAGE
33:       $\tau \leftarrow \tau + 1$ 
34:    end while
35:     $\tilde{\mu} \leftarrow \mu^{(\tau)}$ 
36:     $\mathcal{D}_{t+1} \leftarrow \{\mathbf{x}_i \in \mathcal{U} \setminus \cup_{k=0}^t \mathcal{D}_k : \tilde{\mu}_i \in \mathcal{L}_B(\tilde{\mu})\}$ 
37:    //  $\mathcal{L}_B(\tilde{\mu})$  LES  $B$  PLUS GROSSES VALEURS DE  $\tilde{\mu}$ 
38:  end for
39: return  $\cup_{t=0}^{T-1} (\mathcal{D}_t, \mathcal{Y}_t), \{g_t\}_t$ 
40: end procédure

```

---

## 5.2.4 Différentes fonctions de récompense

Le choix de la fonction de récompense est très important, car il est nécessaire que cette fonction représente le plus fidèlement possible l'influence du choix de l'action, et donc du facteur d'incrémentation des hyper-paramètres  $\alpha, \beta, \eta$ . Ainsi, nous avons modélisé différentes fonctions de récompense lors de nos expériences.

### 5.2.4.1 Récompense basée sur la diversité

La récompense diversity est celle qui a été retenue pour nos expériences, et les résultats se sont montrés convaincants en pratique comme le montre le [tableau 5.7](#). Cette récompense utilise des classifieurs randomisés, et chaque point de l'affichage est comparé avec tous les autres points de l'affichage, par l'intermédiaire des codes binaires entrée par entrée. Pour un problème à deux classes, chaque échantillon non étiqueté a un code binaire de taille égale au nombre de classifieurs randomisés : par exemple 1011001 signifie qu'il y a 8 classifieurs randomisés (à noter que 8 n'est pas le nombre de classes de la tâche à résoudre) et 1 ou 0 signifie que sa classification pour une catégorie donnée est + ou -. Pour des problèmes multi-classes, le code devient décimal, et le reste est appliqué de la même façon. La diversité des erreurs prend donc chaque point de l'affichage  $\mathcal{D}$ , et regarde le nombre de fois où il y a des désaccords avec les autres points.

### 5.2.4.2 Récompense nfold

La récompense nfold permet de ne pas utiliser d'étiquettes supplémentaires si le nombre d'épisodes est strictement égal à 1. L'idée de cette récompense est de partitionner l'affichage  $\mathcal{D}_t$  de l'itération courante  $t$  en  $NP$  sous-ensembles d'entraînement et de validation, et d'entraîner le modèle courant  $g_t$  sur le sous-ensemble d'entraînement associé, et évaluer les performances sur le sous-ensemble de validation associé. Nous obtenons ainsi  $NP$  valeurs d'accuracy, et la récompense finale  $R_{\text{nfold}}$  est calculée en prenant la moyenne arithmétique de ces dernières.

## 5.3 Expériences

Nous étudions la pertinence de notre méthode sur la tâche de classification d'images en utilisant les jeux de données Object-DOTA (XIA et al. 2018) ainsi que CIFAR-10 (KRIZHEVSKY et al. 2010) pour les réglages. De plus, nous utilisons le jeu de données Jefferson (SAHBI et al. 2021) pour la tâche de détection des

changements dans des images satellites. Plus de détails sur ces jeux de données utilisés sont trouvables dans la [section 3.4](#).

### 5.3.1 Détails d’implémentation

Les images de Jefferson et Object-DOTA / CIFAR sont encodées grâce à deux réseaux préentraînés : le GCN (réseau de graphes convolutionnels) pour le premier, et ViT (DOSOVITSKIY et al. 2020) pour le second. Le GCN a plusieurs blocs de couches d’agrégation et de produits scalaires, mais également des couches de *pooling* et entièrement connectées. Nous n’utilisons pas de label des jeux de données cibles pour le préentraînement : dans le cas de Jefferson, le GCN est entraîné sur Jefferson mais avec une fonction de perte prétexte auto-supervisée, similaire à celle dans (DOERSCH et al. 2015), tandis que ViT est préentraîné sur ImageNet (RUSSAKOVSKY et al. 2015) avec une fonction de perte supervisée. Les classifieurs utilisés sur Jefferson sont des SVMs, tandis que pour la classification d’images, il s’agit de forêts aléatoires, dont le réglage des paramètres (*ex.* profondeur maximale, nombre d’estimateurs, etc.) a été effectué sur CIFAR-10 séparément des autres bases, et les paramètres ainsi optimisés ont ensuite été utilisés sur Object-DOTA.

### 5.3.2 Performances améliorées de notre méthode avec le RL

Afin d’évaluer l’impact des différents termes de notre fonction objectif, nous les étudions de façon individuelle, par paire ou tous utilisés. Cependant, nous gardons toujours le critère de cardinalité pondéré par  $\gamma$  qui sert de régularisateur et permet d’obtenir l’équation 5.3 sous forme analytique. L’impact des différents termes et combinaisons est visible sur le [tableau 5.2](#) ainsi que le [tableau 5.1](#). Nous observons que la combinaison de représentativité et diversité a un très fort impact, surtout dans les premières itérations d’apprentissage actif. Cependant, le terme d’ambiguïté apporte un impact dans les dernières itérations de sorte à affiner localement les fonctions de décision, donc quand les modes de la distribution des données sont mieux explorés. Ces performances sont reportées pour différents pourcentages d’échantillonnage à chaque itération  $t$ .

Nous constatons cependant qu’aucun de ces paramètres n’obtient les meilleures performances à chaque itération (*cf.* [figure 5.1](#)). Ainsi, cela confirme notre intuition initiale qu’un meilleur réglage des paramètres de pondération  $\alpha, \beta, \eta$  doit dépendre des itérations. Pour ce faire, nous utilisons l’apprentissage par renforcement comme décrit précédemment, et les résultats sont visibles sur le [tableau 5.2](#) sous l’appellation RL-D et RL-C : la première étant discrète et la seconde continue.

TABLE 5.1 – **Object-DOTA étude d’ablation AL+RL.** Comparaison des performances de notre fonction objectif avec ses différents critères de pondération sur Object-DOTA. Div., Rep. et Amb. correspondent respectivement à la diversité, représentativité et ambiguïté. Les résultats sont présentés à différentes itérations  $t$  (Iter) et les taux d’échantillonnage sous-jacents (Samp%). RL-D et RL-C correspondent aux versions discrète et continue de notre modèle basé sur l’apprentissage par renforcement. **Une valeur d’accuracy plus haute implique de meilleures performances.**

| Iter        | 2     | 3     | 4     | 5     | 6     | 7            | 8            | 9            | 10           |
|-------------|-------|-------|-------|-------|-------|--------------|--------------|--------------|--------------|
| Samp%       | 0.2   | 0.3   | 0.4   | 0.5   | 0.6   | 0.7          | 0.8          | 0.9          | 1.0          |
| Rep.        | 25.77 | 27.91 | 29.81 | 30.69 | 31.96 | 33.51        | 34.08        | 34.53        | 35.05        |
| Div.        | 26.16 | 34.04 | 37.97 | 41.33 | 41.85 | 44.44        | 45.57        | 48.12        | 48.72        |
| Amb.        | 38.44 | 48.86 | 54.96 | 58.21 | 59.51 | 61.03        | 61.14        | 62.61        | 62.66        |
| Rep. + Div. | 49.57 | 51.38 | 53.60 | 54.44 | 54.74 | 55.49        | 55.47        | 55.87        | 56.25        |
| Rep. + Amb. | 42.04 | 49.43 | 53.49 | 57.23 | 59.73 | 61.88        | 62.78        | 63.42        | 64.16        |
| Div. + Amb. | 41.76 | 48.54 | 53.11 | 56.21 | 57.32 | 58.12        | 59.05        | 60.07        | 61.10        |
| All (flat)  | 47.18 | 56.80 | 59.73 | 61.03 | 63.70 | 63.85        | 64.34        | 64.74        | 65.23        |
| RL-D        | 35.42 | 41.19 | 43.44 | 46.28 | 51.29 | 53.43        | 54.09        | 53.47        | 56.59        |
| RL-C        | 46.29 | 55.36 | 57.82 | 60.72 | 63.08 | <b>64.43</b> | <b>64.88</b> | <b>66.20</b> | <b>66.61</b> |

TABLE 5.2 – **Jefferson étude d’ablation AL+RL.** Comparaison des performances de notre fonction objectif avec ses différents critères de pondération sur Jefferson. Div., Rep. et Amb. correspondent respectivement à la diversité, représentativité et ambiguïté. Les résultats sont présentés à différentes itérations  $t$  (Iter) et les taux d’échantillonnage sous-jacents (Samp%). RL-D et RL-C correspondent aux versions discrète et continue de notre modèle basé sur l’apprentissage par renforcement. **Une valeur EER plus faible implique de meilleures performances.**

| Iter        | 2           | 3     | 4           | 5           | 6           | 7           | 8           | 9           | 10          |
|-------------|-------------|-------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| Samp%       | 2.90        | 4.36  | 5.81        | 7.27        | 8.72        | 10.18       | 11.63       | 13.09       | 14.54       |
| Rep.        | 26.21       | 12.72 | 10.48       | 9.88        | 9.70        | 8.52        | 8.85        | 8.61        | 8.82        |
| Div.        | 31.24       | 23.45 | 30.41       | 44.81       | 24.12       | 13.22       | 17.02       | 6.88        | 7.98        |
| Amb.        | 46.68       | 38.73 | 29.91       | 14.74       | 20.11       | 8.33        | 7.41        | 7.37        | 5.53        |
| Rep. + Div. | 26.21       | 33.35 | 25.10       | 21.55       | 11.71       | 2.84        | 1.65        | 1.59        | 1.43        |
| Rep. + Amb. | 26.21       | 12.62 | 10.81       | 9.82        | 9.70        | 8.53        | 9.23        | 8.60        | 8.82        |
| Div. + Amb. | 41.69       | 28.82 | 23.08       | 23.41       | 23.42       | 19.82       | 13.10       | 8.16        | 6.97        |
| All (flat)  | 26.21       | 33.35 | 25.52       | 23.70       | 14.59       | 2.74        | 1.54        | 1.67        | 1.48        |
| RL-D        | 31.75       | 10.36 | 14.83       | 13.36       | 14.70       | <b>1.06</b> | <b>1.06</b> | <b>1.10</b> | <b>1.01</b> |
| RL-C        | <b>9.91</b> | 21.29 | <b>9.95</b> | <b>6.54</b> | <b>4.65</b> | 2.63        | 2.44        | 2.95        | 1.80        |

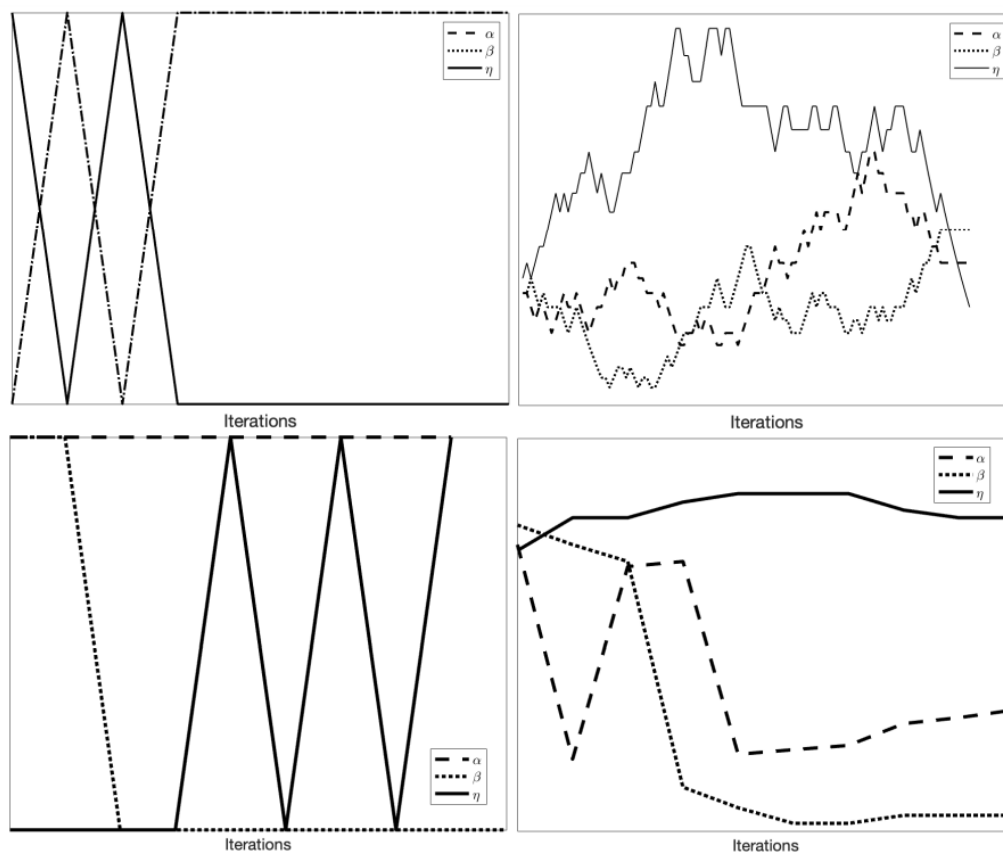


FIGURE 5.1 – **Dynamique des hyper-paramètres RL.** Cette figure montre la dynamique des hyper-paramètres aux différentes itérations d’apprentissage actif, respectivement sur Jefferson (haut) et Object-DOTA (bas). La version discrète (gauche) et continue (droite) présentent des différences visibles : une variation graduelle pour la version continue, et plus abrupte dans le cas de la version discrète.

Ces versions RL obtiennent de meilleures performances que les autres combinaisons, dont la version "all" / flat avec tous les paramètres  $\alpha, \beta, \eta$  activés. Les performances deviennent particulièrement meilleures lors des dernières itérations pour la version discrète (RL-D) et les premières itérations pour la version continue (RL-C) sur Jefferson (tableau 5.2), et les dernières itérations sur Object DOTA (tableau 5.1).

### 5.3.3 Comparaison avec les baselines

La figure 5.2 montre plusieurs comparaisons de notre méthode basée sur l’apprentissage par renforcement face à différentes stratégies d’échantillonnage telles que l’aléatoire, maxmin et l’incertitude (respectivement *random*, *maxmin* et *uncer-*

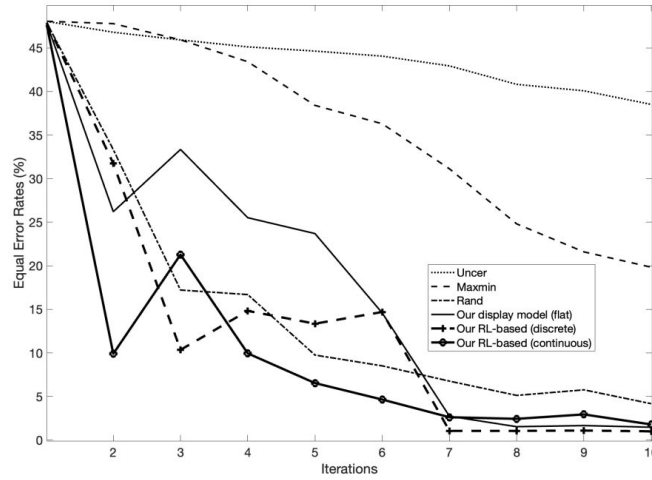


FIGURE 5.2 – **Test de référence RL sur Jefferson.** Cette figure montre une comparaison de différentes stratégies d'échantillonnage à différentes itérations et leurs taux d'échantillonnage associés sur Jefferson (cf. [tableau 5.2 : Samp](#)), avec notamment notre stratégie basée sur le RL. L'apprentissage entièrement supervisé obtient un EER de 0.94%.

*tainty*). La stratégie aléatoire sélectionne des données de  $\mathcal{U} \setminus \cup_{k=0}^t \mathcal{D}_k$  tandis que MaxMin, de façon similaire à (SENER et SAVARESE 2017), sélectionne de façon gloutonne un échantillon  $x_i$  dans  $\mathcal{D}_{t+1}$  depuis l'ensemble d'images  $\mathcal{U} \setminus \cup_{k=0}^t \mathcal{D}_k$  en maximisant la distance minimum par rapport à  $\cup_{k=0}^t \mathcal{D}_k$ . Nous comparons également notre méthode avec l'échantillonnage par incertitude (CULOTTA et MCCALLUM 2005) qui sélectionne les échantillons avec les scores les plus ambigus (*ie.* proches de zéro). La version entièrement supervisée utilisant toutes les annotations nous sert de borne supérieure des performances, le modèle d'apprentissage est entraîné d'un seul coup.

Les performances sur les [figure 5.2](#) et [figure 5.3](#) montrent l'impact positif de notre méthode basée sur l'apprentissage par renforcement, que ce soit les versions discrète et continue, face aux stratégies d'échantillonnage de référence. Hormis le modèle flat utilisé également dans (SAHBI et al. 2021), la plupart de ces méthodes ne classifient pas les données suffisamment bien. Ces dernières sont soit efficaces dans les premières itérations (par exemple MaxMin et aléatoire qui capturent correctement la diversité des données sans pouvoir affiner les fonctions de décision), ou bien dans les dernières itérations (comme l'incertitude qui affine localement les fonctions de décision mais souffre d'un manque de diversité).

La stratégie d'affichage flat (SAHBI et al. 2021) présentée au chapitre précédent a les avantages de l'aléatoire, MaxMin et l'incertitude, mais est limitée par les pondérations fixes des critères de représentativité, diversité et ambiguïté. Ap-

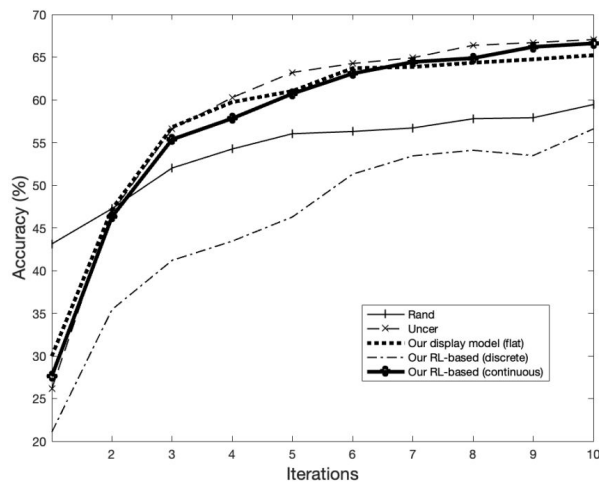


FIGURE 5.3 – **Test de référence RL sur Object DOTA.** Cette figure montre une comparaison de différentes stratégies d’échantillonnage à différentes itérations et leurs taux d’échantillonnage associés sur Object DOTA (cf. [tableau 5.1 : Samp](#)), avec notamment notre stratégie basée sur le RL. L’apprentissage entièrement supervisé obtient une accuracy de 74.92%.

prendre ces critères au fur et à mesure des itérations, comme le fait notre méthode basée sur l’apprentissage par renforcement, permet de s’adapter et de tirer parti au maximum des avantages de chacun des critères. La version RL-C étant efficace également aux premières itérations, cela est particulièrement utile lorsque beaucoup d’annotations par cycle sont utilisées. On peut voir sur la [figure 5.4](#) des exemples d’ensembles d’images annotées obtenus avec notre méthode RL en comparaison des autres stratégies de sélection. Notre méthode ainsi que les expériences présentées ci-avant ont donné lieu à deux publications dans des conférences internationales, à savoir IAPR ICPR’2022 (DESCHAMPS et SAHBI 2022a) et IGARSS’2022 (DESCHAMPS et SAHBI 2022b).

### 5.3.4 Expériences additionnelles pour affiner les réglages

#### 5.3.4.1 Accélération du temps de calcul de notre méthode

Le temps de calcul est un paramètre important dès lors que l’on utilise des réseaux profonds, même en utilisant de puissants GPUs, l’entraînement des modèles et même l’inférence prennent beaucoup de temps. Nous sommes intéressés par les performances de nos méthodes d’apprentissage actif en comparaison d’autres stratégies, nous avons donc cherché à minimiser le temps de calcul afin de produire plus d’expériences, tout en gardant les mêmes capacités de comparaison

équitable entre les différentes méthodes. La solution que nous avons adoptée est d'utiliser un classifieur peu coûteux en ressources, l'algorithme de la forêt aléatoire (BREIMAN 2001), sur les cartes de caractéristiques obtenues avec un modèle de type ViT (DOSOVITSKIY et al. 2020) qui s'est montré le plus efficace.

En effet, afin d'évaluer la faisabilité d'utiliser nos algorithmes avec cette méthode, nous avons testé à la fois la rapidité et la précision, en utilisant toutes les étiquettes et cartes de caractéristiques d'entraînement. L'évaluation sur l'ensemble de test a été effectuée en utilisant les cartes de caractéristiques récupérées de la même façon que pour l'ensemble d'entraînement. Un ResNet-18 préentraîné sur ImageNet et entraîné ensuite sur Object-DOTA, permet d'obtenir des caractéristiques qui donnent au maximum une accuracy d'environ 40% en test, ce qui est insuffisant : nous avons donc utilisé un réseau capable de produire de meilleures cartes de caractéristiques.

Avec le modèle ViT (DOSOVITSKIY et al. 2020), préentraîné sur ImageNet, et entraîné ensuite sur un certain nombre  $n$  de données d'Object-DOTA choisies aléatoirement, nous avons obtenu des résultats insuffisants lorsque  $n = 100$  ou  $n = 1000$ , mais de très bons résultats lorsque  $n = 10000$ . Ainsi, nous utilisons un peu plus de 10% du jeu de données pour l'entraînement du réseau ViT, nous l'utilisons simplement comme extracteur de cartes de caractéristiques, cartes que nous sérialisons afin d'être réutilisées de façon quasi instantanée dans nos expériences. Ensuite, les méthodes d'apprentissage actif sont comparées en utilisant le classifieur de forêt aléatoire, ce qui est extrêmement rapide.

#### 5.3.4.2 Influence du préentraînement des réseaux sur les résultats

Nous avons mené des expériences supplémentaires afin d'observer si la hiérarchie des méthodes sur Object-DOTA restait la même avec un CNN préentraîné ou avec des paramètres initiaux aléatoires (*from scratch*). Le [tableau 5.3](#) montre les résultats des baselines d'échantillonnage aléatoire et par incertitude ainsi que notre méthode initiale d'apprentissage actif n'utilisant pas l'apprentissage par renforcement, séparée en fonction du nombre de composantes qu'elle utilise : toutes les composantes (All), juste la diversité (Div.), représentativité (Rep.), ambiguïté (Amb.), la paire diversité et ambiguïté (Div.+Amb.), etc. Notre méthode initiale avec la diversité et l'ambiguïté est ainsi la plus performante pour 100, 200, 600 étiquettes, la diversité et la représentativité pour 300, 800 étiquettes, la diversité pour 700, 1000 étiquettes et enfin l'ambiguïté pour 900 étiquettes. Du côté des baselines, l'échantillonnage aléatoire est plus performant pour 400, 500 étiquettes.

Le [tableau 5.4](#) montre quant à lui les résultats sur Object-DOTA d'un ResNet-18 cette fois-ci préentraîné. De plus, il a été entraîné sur un plus grand nombre d'epochs, ce nombre étant celui à partir duquel le réseau converge. Cette fois-ci,



TABLE 5.3 – **Object-DOTA sans préentraînement du réseau.** Object DOTA : 100 labels / cycle et 10 cycles, 10 essais, ResNet-18 modèle "from scratch", 100 epochs. **Une accuracy (%) plus haute implique de meilleures performances.**

| Method               | 100                 | 200                 | 300                 | 400                 | 500                 |
|----------------------|---------------------|---------------------|---------------------|---------------------|---------------------|
| Random sampling      | 37.91 ± 5.73        | 57.0 ± 1.86         | 48.24 ± 9.66        | <b>51.79 ± 5.38</b> | <b>63.28 ± 2.82</b> |
| Uncertainty sampling | 36.95 ± 8.76        | 57.75 ± 1.66        | 39.75 ± 10.7        | 42.78 ± 12.41       | 62.09 ± 2.47        |
| AL (All)             | 40.92 ± 8.46        | 57.45 ± 2.87        | 44.91 ± 6.07        | 46.24 ± 8.47        | 62.6 ± 2.25         |
| AL (Div.)            | 39.41 ± 8.48        | 58.04 ± 1.79        | 46.63 ± 8.35        | 49.13 ± 6.55        | 62.73 ± 2.85        |
| AL (Rep.)            | 34.87 ± 8.32        | 53.45 ± 1.61        | 43.11 ± 6.49        | 49.26 ± 6.39        | 60.8 ± 1.83         |
| AL (Amb.)            | 40.62 ± 7.6         | 58.26 ± 2.1         | 44.84 ± 9.32        | 44.3 ± 11.51        | 61.98 ± 3.23        |
| AL (Div.+Rep.)       | 31.87 ± 8.5         | 58.19 ± 1.79        | <b>48.34 ± 7.29</b> | 51.09 ± 5.08        | 59.74 ± 4.35        |
| AL (Div.+Amb.)       | <b>43.19 ± 4.92</b> | <b>58.7 ± 1.68</b>  | 45.57 ± 10.84       | 41.87 ± 12.34       | 63.13 ± 2.66        |
| AL (Amb.+Rep.)       | 40.74 ± 5.95        | 56.87 ± 3.3         | 38.45 ± 9.01        | 43.88 ± 7.69        | 61.1 ± 2.28         |
| Method               | 600                 | 700                 | 800                 | 900                 | 1000                |
| Random sampling      | 66.15 ± 1.95        | 62.51 ± 2.48        | 60.3 ± 5.44         | 62.18 ± 5.22        | 69.88 ± 1.24        |
| Uncertainty sampling | 66.09 ± 1.72        | 62.78 ± 4.97        | 61.67 ± 5.34        | 61.61 ± 7.97        | 69.73 ± 1.44        |
| AL (All)             | 66.17 ± 1.46        | 62.49 ± 4.13        | 61.42 ± 4.05        | 60.04 ± 5.75        | 69.55 ± 1.2         |
| AL (Div.)            | 66.72 ± 0.9         | <b>63.34 ± 2.78</b> | 59.65 ± 5.47        | 64.07 ± 2.57        | <b>70.68 ± 0.78</b> |
| AL (Rep.)            | 64.22 ± 1.05        | 59.94 ± 5.14        | 61.65 ± 4.34        | 61.07 ± 3.11        | 67.88 ± 0.79        |
| AL (Amb.)            | 66.91 ± 1.13        | 61.83 ± 3.93        | 61.53 ± 4.93        | <b>64.71 ± 3.41</b> | 70.39 ± 1.04        |
| AL (Div.+Rep.)       | 65.35 ± 0.84        | 61.79 ± 3.31        | <b>62.03 ± 3.23</b> | 63.46 ± 2.42        | 69.32 ± 1.12        |
| AL (Div.+Amb.)       | <b>67.04 ± 1.13</b> | 62.0 ± 2.0          | 60.96 ± 7.12        | 62.89 ± 3.51        | 69.89 ± 0.61        |
| AL (Amb.+Rep.)       | 65.24 ± 1.32        | 60.73 ± 4.24        | 55.53 ± 4.38        | 62.97 ± 3.1         | 68.92 ± 1.35        |

la diversité et l’ambiguïté est la plus performante pour 800 étiquettes, l’ambiguïté pour 300, 400, 500 étiquettes, la représentativité pour 1000 étiquettes, l’ambiguïté et la représentativité pour 100, 200, 600 étiquettes et enfin toutes les composantes pour 900 étiquettes. Du côté des baselines, l’échantillonnage aléatoire est plus performant pour 200 étiquettes et l’échantillonnage par incertitude pour 700 étiquettes. Finalement, le [tableau 5.5](#) montre que selon que l’on ait préentraîné le réseau ou non, la méthode la plus performante pour chaque itération (*ie.* nombre d’images étiquetées utilisées pour entraîner le réseau) est différente à chaque fois. Cela est à nuancer car les résultats sont souvent très proches, mais cela confirme l’intuition qu’il n’y a pas de méthode supérieure aux autres dans toutes les conditions en apprentissage actif. Nos autres expériences utilisent plutôt des réseaux préentraînés pour la tâche de classification d’images, car cela correspond aux conditions réelles de l’application industrielle.

### 5.3.4.3 Rééquilibrage des classes

Le jeu de données Object-DOTA est très déséquilibré, ainsi nous avons expérimenté l’utilisation de rééquilibrage des classes dans la fonction de perte, afin d’assigner un poids différent à chaque exemple ; en fonction de la proportion du

TABLE 5.4 – **Object-DOTA avec préentraînement du réseau.** Object DOTA : 100 labels / cycle et 10 cycles, 5 essais, ResNet-18 **modèle préentraîné**, 50 epochs. **Une accuracy (%) plus haute implique de meilleures performances.**

| Method               | 100                 | 200                 | 300                 | 400                  | 500                 |
|----------------------|---------------------|---------------------|---------------------|----------------------|---------------------|
| Random sampling      | 55.88 ± 7.59        | <b>69.3</b> ± 2.62  | 73.37 ± 1.82        | 76.19 ± 1.5          | 77.11 ± 1.35        |
| Uncertainty sampling | 57.18 ± 5.15        | 68.38 ± 1.94        | 73.99 ± 1.19        | 77.22 ± 1.11         | 77.9 ± 1.05         |
| AL (All)             | 55.05 ± 9.35        | 65.85 ± 3.03        | 73.17 ± 2.21        | 76.43 ± 1.82         | 77.52 ± 1.78        |
| AL (Div.)            | 54.39 ± 0.91        | 65.19 ± 3.25        | 73.82 ± 1.37        | 76.0 ± 1.01          | 77.61 ± 1.12        |
| AL (Rep.)            | 57.86 ± 1.92        | 68.53 ± 2.3         | 73.07 ± 1.2         | 76.18 ± 0.55         | 77.58 ± 0.59        |
| AL (Amb.)            | 56.9 ± 3.06         | 69.16 ± 2.24        | <b>75.13</b> ± 1.22 | <b>77.85</b> ± 0.83  | <b>78.07</b> ± 0.71 |
| AL (Div.+Rep.)       | 57.51 ± 3.22        | 67.39 ± 0.83        | 73.06 ± 2.4         | 76.58 ± 1.2          | 77.36 ± 1.47        |
| AL (Div.+Amb.)       | 58.19 ± 3.69        | 66.74 ± 4.73        | 73.16 ± 2.49        | 75.55 ± 2.43         | 77.22 ± 3.11        |
| AL (Amb.+Rep.)       | <b>58.51</b> ± 1.07 | <b>69.3</b> ± 4.27  | 74.17 ± 1.27        | 76.47 ± 1.35         | 77.83 ± 1.63        |
| Method               | 600                 | 700                 | 800                 | 900                  | 1000                |
| Random sampling      | 75.84 ± 3.81        | 77.0 ± 3.69         | 76.03 ± 9.34        | 45.59 ± 4.06         | 78.93 ± 5.34        |
| Uncertainty sampling | 77.68 ± 2.01        | <b>77.79</b> ± 2.22 | 76.71 ± 2.33        | 43.41 ± 8.59         | 75.71 ± 4.25        |
| AL (All)             | 76.63 ± 1.68        | 74.27 ± 6.58        | 74.79 ± 5.42        | <b>55.96</b> ± 10.43 | 80.02 ± 2.03        |
| AL (Div.)            | 76.91 ± 2.41        | 77.41 ± 1.24        | 76.24 ± 2.35        | 53.17 ± 11.84        | 79.68 ± 1.35        |
| AL (Rep.)            | 78.12 ± 0.94        | 76.02 ± 2.23        | 78.08 ± 3.06        | 54.43 ± 8.08         | <b>80.23</b> ± 1.59 |
| AL (Amb.)            | 78.28 ± 1.29        | 76.66 ± 2.42        | 77.12 ± 2.78        | 55.24 ± 9.91         | 79.58 ± 3.39        |
| AL (Div.+Rep.)       | 76.1 ± 2.06         | 77.11 ± 1.83        | 75.16 ± 3.19        | 54.94 ± 3.08         | 79.43 ± 0.73        |
| AL (Div.+Amb.)       | 77.0 ± 1.21         | 76.38 ± 0.58        | <b>79.66</b> ± 1.51 | 46.81 ± 10.03        | 77.01 ± 3.3         |
| AL (Amb.+Rep.)       | <b>78.58</b> ± 1.48 | 73.93 ± 5.56        | 77.68 ± 1.55        | 51.09 ± 8.65         | 77.62 ± 3.72        |

TABLE 5.5 – **Object-DOTA comparaison avec ou sans préentraînement.** Comparaison des méthodes qui ont obtenu les meilleures performances à chaque cycle d’apprentissage actif, de 100 à 1000 étiquettes, selon que le réseau ait été préentraîné ou non.

| Method       | 100       | 200         | 300       | 400   | 500   |
|--------------|-----------|-------------|-----------|-------|-------|
| From scratch | Div.+Amb. | Div.+Amb.   | Div.+Rep. | Rand. | Rand. |
| Pretrained   | Amb.+Rep. | Rand.       | Amb.      | Amb.  | Amb.  |
| Method       | 600       | 700         | 800       | 900   | 1000  |
| From scratch | Div.+Amb. | Div.        | Div.+Rep. | Amb.  | Div.  |
| Pretrained   | Amb.+Rep. | Uncertainty | Div.+Amb. | All   | Rep.  |

nombre d’échantillons de même classe par rapport au nombre total d’échantillons dans chaque affichage  $\mathcal{D}_t$  avec  $t$  l’itération d’apprentissage actif. Le [tableau 5.6](#) montre que le rééquilibrage des classes dans la fonction de perte donne des performances nettement inférieures lorsque le réseau est préentraîné, cependant l’accuracy s’améliore grandement avec un modèle non préentraîné à partir de 300 étiquettes. Il est probable que le modèle préentraîné soit déjà capable de recon-

naître les classes rares de notre jeu de données. De plus, le fait de choisir un poids minimum à 1 plutôt que 0 est très bénéfique sur le modèle non préentraîné, avec une différence de quasiment 10 points.

TABLE 5.6 – **Object-DOTA expérience de rééquilibrage des classes.** *Object-DOTA : 100 labels / cycle et 3 cycles, 5 essais, ResNet-18 modèle "from scratch", 30 epochs, nb workers = 8. Une accuracy (%) plus haute implique de meilleures performances..*

| Rééquilibrage     | Réseau       | 100          | 200          | 300           |
|-------------------|--------------|--------------|--------------|---------------|
| Non               | From scratch | 29.52 ± 5.37 | 35.96 ± 7.98 | 32.24 ± 8.96  |
| Oui, $\min_w = 0$ | From scratch | 32.59 ± 7.89 | 42.59 ± 5.21 | 40.75 ± 10.96 |
| Oui, $\min_w = 1$ | From scratch | 32.31 ± 4.75 | 35.81 ± 9.9  | 50.35 ± 2.32  |
| Non               | Pretrained   | 58.61 ± 3.23 | 65.85 ± 2.52 | 70.79 ± 2.34  |
| Oui, $\min_w = 0$ | Pretrained   | 47.71 ± 3.44 | 59.59 ± 2.17 | 63.95 ± 2.8   |
| Oui, $\min_w = 1$ | Pretrained   | 46.42 ± 3.01 | 58.78 ± 2.44 | 63.07 ± 4.48  |

#### 5.3.4.4 Comparaison des différentes fonctions de récompense

Le [tableau 5.7](#) montre une comparaison de notre méthode Flat avec notre méthode RL-C, en utilisant différentes récompenses : diversity et nfold, et leurs

TABLE 5.7 – **CIFAR-10 comparaison de RL-C avec différentes récompenses.** *Comparaison des performances de RL-C avec les fonctions de récompense diversity et nfold face à Flat sur CIFAR-10. Une valeur d'accuracy plus haute implique de meilleures performances.*

| Method           | 200<br>0.4%         | 400<br>0.8%         | 600<br>1.2%         | 800<br>1.6%         | 1000<br>2%          |
|------------------|---------------------|---------------------|---------------------|---------------------|---------------------|
| Flat             | <b>41.68 ± 5.98</b> | 44.78 ± 3.5         | 46.27 ± 2.44        | 47.87 ± 1.28        | 48.0 ± 1.21         |
| RL-C (diversity) | 39.34 ± 7.92        | <b>46.07 ± 4.56</b> | <b>48.47 ± 2.91</b> | <b>49.28 ± 2.35</b> | <b>49.11 ± 1.47</b> |
| RL-C (nfold)     | 38.95 ± 8.15        | 45.03 ± 3.32        | 46.23 ± 2.46        | 47.53 ± 1.76        | 46.18 ± 2.05        |

performances entre 200 et 1000 étiquettes sur CIFAR-10 (ce qui fait entre 0.4% et 2% de la base). Nous observons que l'écart type est légèrement plus élevé pour notre méthode RL-C par rapport à Flat, quelle que soit la fonction de récompense associée. Flat obtient une meilleure accuracy à la première itération, mais c'est RL-C avec la fonction de récompense diversity qui obtient ensuite les meilleures performances entre 400 et 1000 étiquettes. RL-C combinée à la fonction de récompense nfold obtient des performances proches de Flat. Les performances et la solidité théorique de la récompense "diversity" (cette récompense permet d'obtenir des configurations qui donnent des affichages avec des points qui ne font

pas consensus parmi des classifieurs randomisés, et sont donc potentiellement informatifs) expliquent pourquoi nous avons opté pour cette récompense pour nos publications (DESCHAMPS et SAHBI 2022b; DESCHAMPS et SAHBI 2022a).

#### 5.3.4.5 Comparaison du nombre de clusters

Le [tableau 5.8](#) montre les performances de notre algorithme d'apprentissage actif Flat lorsque l'on fait varier le nombre de clusters, sur le jeu de données

TABLE 5.8 – **CIFAR-10 tests additionnels AL Flat.** *Comparaison des performances de notre méthode Flat avec différentes listes de nombres de clusters sur CIFAR-10. Une valeur d'accuracy plus haute implique de meilleures performances.*

| # clusters | 200<br>0.4%          | 400<br>0.8%        | 600<br>1.2%         | 800<br>1.6%         | 1000<br>2%          |
|------------|----------------------|--------------------|---------------------|---------------------|---------------------|
| 8          | 39.28 ± 6.52         | 45.24 ± 3.75       | 47.27 ± 2.68        | 49.3 ± 1.57         | 49.34 ± 1.04        |
| 16         | 25.3 ± 10.32         | 31.88 ± 6.06       | 34.9 ± 3.23         | 37.09 ± 2.1         | 38.42 ± 1.55        |
| 32         | 38.25 ± 7.94         | 46.3 ± 3.75        | 48.96 ± 2.4         | 50.84 ± 2.0         | 51.13 ± 1.11        |
| 64         | 40.11 ± 7.68         | 46.3 ± 4.02        | 50.12 ± 2.21        | 52.68 ± 0.95        | 50.83 ± 0.9         |
| 128        | 32.06 ± 8.84         | 37.94 ± 4.7        | 41.67 ± 2.31        | 43.79 ± 1.11        | 44.01 ± 1.5         |
| 256        | 36.97 ± 8.54         | 42.59 ± 3.2        | 46.59 ± 2.83        | 48.03 ± 1.78        | 48.26 ± 0.95        |
| 2, 4       | <b>40.34 ± 10.78</b> | <b>48.4 ± 5.16</b> | <b>51.23 ± 2.53</b> | <b>52.75 ± 0.96</b> | <b>51.66 ± 0.35</b> |
| 2, 32      | 33.42 ± 8.46         | 39.75 ± 3.68       | 42.55 ± 2.41        | 44.24 ± 1.0         | 45.96 ± 0.95        |
| 2, 256     | 34.58 ± 10.51        | 44.27 ± 4.26       | 47.92 ± 2.69        | 49.63 ± 0.4         | 49.35 ± 0.25        |
| 32, 64     | 39.31 ± 6.27         | 44.67 ± 3.2        | 48.22 ± 2.25        | 50.02 ± 0.8         | 49.69 ± 0.29        |
| 128, 256   | 39.45 ± 7.48         | 46.5 ± 3.42        | 49.56 ± 2.6         | 50.62 ± 0.98        | 49.97 ± 0.9         |

CIFAR-10 entre 200 et 1000 étiquettes (0.4% à 2% de la base). Nous observons que la combinaison avec 2, 4 clusters est systématiquement la plus performante, suivie de près par les combinaisons mono-cluster à 32 et 64 clusters. Parmi les autres combinaisons de multi-clusters, la combinaison à 128, 256 clusters obtient également une accuracy proche de la meilleure combinaison. La difficulté d'anticiper correctement le nombre de clusters idéal nous a cependant fait utiliser en pratique une combinaison avec beaucoup de nombres de clusters différents : 2, 4, 8, 16, 32, 64.

#### 5.3.4.6 Variation des paramètres de l'algorithme Q-learning

Le [tableau 5.9](#) montre les différentes performances que l'on obtient sur CIFAR-10 lorsque l'on fait varier certains paramètres de notre algorithme de Q-learning, notamment le facteur de remise  $\delta$  et le taux d'apprentissage  $\gamma_{rl}$  (cf. [équation 5.7](#)). Les performances sont proches les unes des autres, néanmoins la version avec

$\delta = 0.1$  et  $\gamma_{rl} = 0.1$  par défaut obtient la meilleure accuracy à chaque itération. Lorsque l’on utilise le  $\delta = 0.99$  par défaut et que l’on fait varier  $\gamma_{rl}$ , la version avec  $\gamma_{rl} = 0.001$  est la plus performante pour 200 et 1000 étiquettes (0.4% à 2% de la base CIFAR-10), tandis que la version  $\gamma_{rl} = 0.5$  est la plus performante pour 400, 600 et 800 étiquettes (soit 0.8% à 1.6% de la base). Nous constatons que le taux d’apprentissage  $\gamma_{rl}$  influence peu les résultats, contrairement au facteur de remise  $\delta$  dont la meilleure version  $\delta = 0.1$  obtient parfois jusqu’à 5 points d’accuracy en plus. Cela signifie donc qu’accorder beaucoup plus d’importance à la récompense actuelle plutôt qu’aux récompenses futures (un facteur de 10 entre 0.1 et  $0.1^2$ , beaucoup plus élevé qu’entre 0.99 et  $0.99^2$ ) donne de meilleures performances avec notre fonction objectif sur CIFAR-10.

TABLE 5.9 – **CIFAR-10 tests additionnels AL+RL bis.** *Comparaison des performances de notre fonction objectif avec ses différents critères de pondération sur CIFAR-10. Une valeur d’accuracy plus haute implique de meilleures performances.*

| Valeur $\delta$ | Valeur $\gamma_{rl}$  | 200<br>0.4%  | 400<br>0.8%  | 600<br>1.2%  | 800<br>1.6%  | 1000<br>2%   |
|-----------------|-----------------------|--------------|--------------|--------------|--------------|--------------|
| $\delta = 0.01$ | $\gamma_{rl} = 0.1$   | 27.35        | 41.08        | 42.38        | 47.75        | 49.69        |
| $\delta = 0.1$  | $\gamma_{rl} = 0.1$   | <b>32.32</b> | <b>42.26</b> | <b>47.4</b>  | <b>50.42</b> | <b>50.35</b> |
| $\delta = 0.5$  | $\gamma_{rl} = 0.1$   | 23.4         | 29.12        | 38.42        | 40.46        | 43.07        |
| $\delta = 2$    | $\gamma_{rl} = 0.1$   | 28.78        | 39.84        | 43.43        | 45.09        | 46.97        |
| $\delta = 0.99$ | $\gamma_{rl} = 0.001$ | <b>29.36</b> | 40.25        | 44.77        | 48.19        | <b>50.13</b> |
| $\delta = 0.99$ | $\gamma_{rl} = 0.01$  | 26.98        | 37.53        | 44.44        | 46.78        | 47.82        |
| $\delta = 0.99$ | $\gamma_{rl} = 0.5$   | 25.01        | <b>40.61</b> | <b>46.44</b> | <b>48.86</b> | 48.76        |

## 5.4 Conclusion

Nous avons présenté dans ce chapitre un modèle d’apprentissage actif se basant sur l’optimisation d’une fonction objectif mêlant des critères de représentativité, diversité et ambiguïté des données. Cette approche probabiliste assigne des mesures d’appartenance aux échantillons non étiquetés, et sélectionne comme ensemble d’entraînement les échantillons avec les mesures les plus élevées. Cet algorithme est aussi basé sur l’apprentissage par renforcement, qui va sélectionner les meilleures combinaisons (discrète ou continue) de représentativité, diversité et ambiguïté au fur et à mesure des itérations d’apprentissage actif, ce qui permet d’obtenir de meilleures performances. Les diverses expériences menées sur les tâches de détection de changements dans les images satellites ainsi que la classification d’images, sur le jeu de données Object-DOTA, montrent l’amélioration des

performances de notre méthode par rapport à la version du chapitre précédent, les tests de référence ainsi que l'état de l'art.



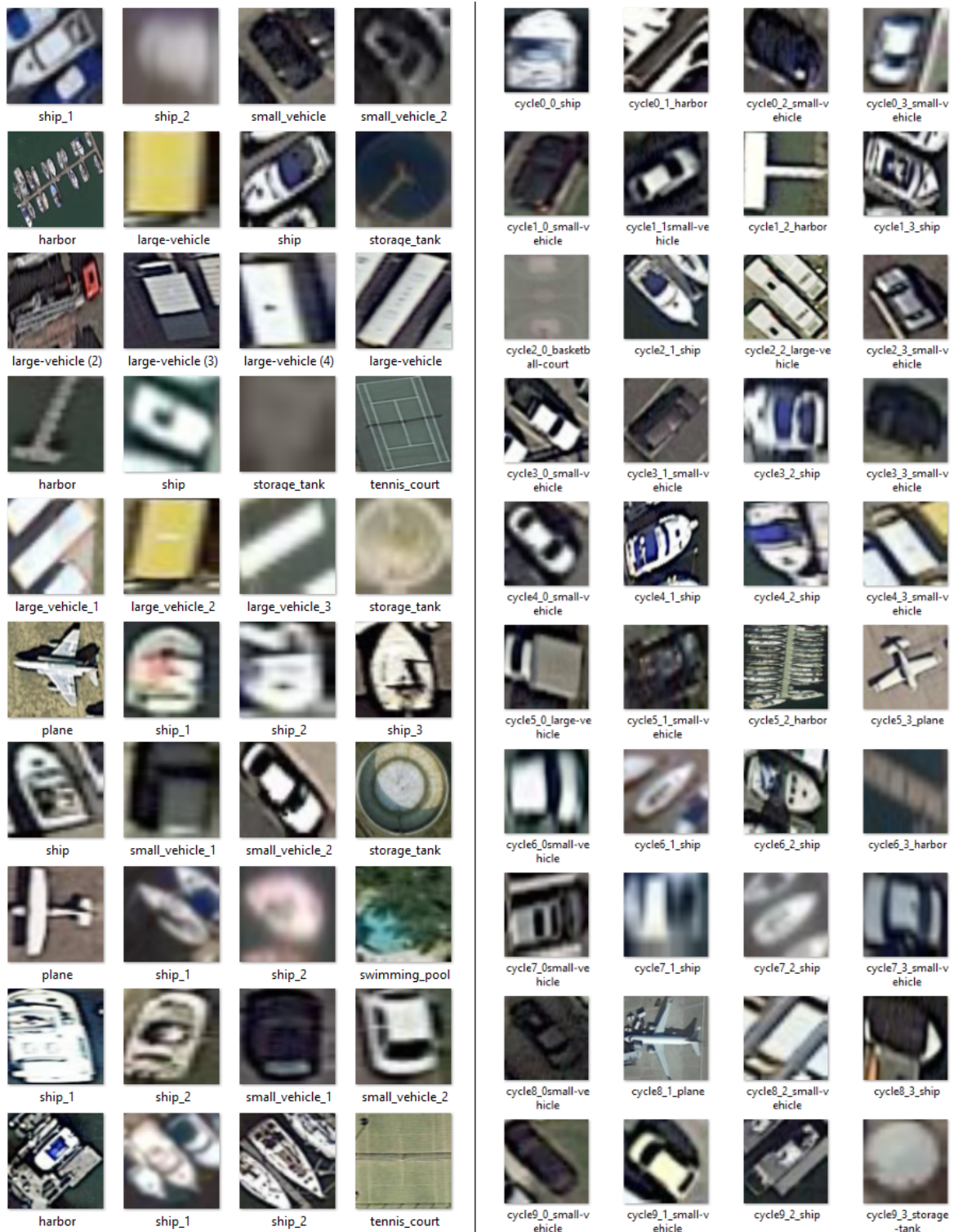


FIGURE 5.4 – Exemples d’images annotées sur Object-DOTA. Cette figure montre des images sélectionnées pour annotation par notre stratégie AL+RL (gauche) et par la stratégie d’échantillonnage aléatoire (droite) sur Object-DOTA.

## APPRENTISSAGE PROFOND ACTIF POUR L'IMAGE À L'AIDE D'EXEMPLES VIRTUELS

### *Résumé chapitre*

*Dans ce chapitre, nous concevons un nouvel algorithme interactif de classification d'images basé sur l'apprentissage actif. Le cadre proposé est itératif et s'appuie sur un modèle par question & réponse qui demande à un oracle (utilisateur expert) des questions sur l'affichage (un sous-ensemble d'images à annoter) le plus informatif et met à jour un modèle d'apprentissage automatique en fonction des réponses de l'utilisateur. La contribution de notre algorithme réside dans le nouveau modèle d'affichage qui va sélectionner des exemples virtuels les plus représentatifs et divers, lesquels remettent en cause le modèle appris, ce qui conduit à un modèle très discriminatoire dans les itérations ultérieures de l'apprentissage actif. Des expériences approfondies conduites sur la tâche difficile de la classification d'images, démontrent l'efficacité et la supériorité de notre modèle d'affichage virtuel en comparaison de la littérature de référence.*



## Sommaire

---

|       |  |     |
|-------|--|-----|
| 6.1   | Introduction . . . . .   | 132 |
| 6.2   | Méthode proposée . . . . .   | 133 |
| 6.2.1 | Classification d'images de façon interactive : rappel . . . . .                | 134 |
| 6.2.2 | Un nouveau modèle d'affichage virtuel . . . . .                                | 135 |
| 6.2.3 | Optimisation . . . . .   | 136 |
| 6.2.4 | Inversion de réseaux profonds . . . . .  | 136 |
| 6.3   | Expériences de détections des changements dans des images satellites . . . . . | 138 |
| 6.3.1 | Etude d'ablation . . . . .   | 138 |
| 6.3.2 | Comparaison avec les baselines . . . . .                                       | 139 |
| 6.4   | Expériences de classification d'images . . . . .                               | 140 |
| 6.4.1 | Détails d'implémentation . . . . .   | 140 |
| 6.4.2 | Performances de notre méthode . . . . .  | 141 |
| 6.5   | Conclusion . . . . .   | 143 |

---

## 6.1 Introduction

La méthode que nous proposons dans ce chapitre n'est pas une itération de notre méthode initiale, mais une méthode entièrement nouvelle qui sélectionne des exemples virtuels les plus représentatifs et diversifiés, qui remettent en question les fonctions de classification apprises jusqu'alors, ce qui conduit à obtenir des fonctions hautement discriminantes dans les itérations suivantes de l'apprentissage actif. Nous appliquons cette nouvelle méthode à la tâche de détection des changements dans des images satellites, dont nous faisons un bref rappel ci-après pour faciliter la lecture, et de la classification d'images, sur le jeu de données Object-DOTA (XIA et al. 2018).

Pour rappel, la détection de changements dans des images satellites a pour but de localiser des instances de changements *pertinents* dans une scène donnée à des moments différents. Il existe beaucoup d'applications de ce problème, notamment l'évaluation des dommages aux infrastructures lors de catastrophes naturelles (tornades, tremblements de terres, etc.), ainsi ce problème est présent dans la littérature (BRUNNER et al. 2010; GOKON et al. 2015). Cette tâche est un défi du fait que les changements pertinents sont diversifiés et les scènes sont sujettes à un grand nombre de changements non pertinents, à cause des artefacts des capteurs, erreurs d'acquisition des images, variations de luminosité, occlusions, conditions météorologiques, etc. La classification d'images a pour but d'attribuer la bonne

catégorie, ou classe, pour des images contenant un seul objet d'intérêt prenant la majorité de l'espace de l'image.

Dans ce chapitre, nous introduisons un nouvel algorithme de détection de changements dans des images satellites et de classification d'images basé sur un modèle de questions & réponses, qui interroge les intentions d'un utilisateur (oracle), et met à jour un modèle d'apprentissage automatique en conséquence. L'oracle fournit ces annotations de façon frugale seulement sur les affichages (données) les plus informatifs, qui sont *appris* plutôt que d'être directement échantillonnés de l'ensemble *fixe* de données non étiquetées. Une distribution de probabilités conditionnelles est définie et mesure l'importance de chacun des exemplaires dans les affichages appris à partir de l'ensemble de données non étiquetées. Cette distribution conditionnelle est apprise en utilisant un nouveau critère adversaire qui trouve les exemplaires les plus divers, représentatifs et incertains qui remettent (le plus) en cause le modèle d'apprentissage automatique précédemment entraîné.

Il est à noter que, malgré le fait qu'il soit adversaire, le cadre que nous proposons dans notre modèle d'affichage est conceptuellement différent des GANs (CRESWELL et al. 2018). En effet, ces derniers cherchent à générer de fausses données qui induisent en erreur les discriminateurs entraînés tandis que le modèle que nous proposons vise à générer les données les plus critiques pour les annotations futures ; c'est-à-dire les exemplaires les plus représentatifs et divers qui augmentent l'incertitude, et remettent (le plus) en cause le discriminateur courant, ce qui conduit finalement à des classifieurs plus précis dans les itérations suivantes de classification d'images. Les expériences menées sur la tâche difficile de classification d'images, montrent la pertinence de notre modèle *d'apprentissage d'exemplaires et d'affichage* en comparaison de la littérature.

## 6.2 Méthode proposée

Soit  $\mathcal{X}$  l'ensemble de toutes les images possibles tirées d'une distribution de probabilité existante mais inconnue  $P(X, Y)$ . Ici, la variable aléatoire  $X$  se réfère à une image d'entrée et  $Y$  à son étiquette de classe inconnue. Soit  $nc$  classes visuelles (c'est-à-dire des étiquettes ou catégories), et  $\mathcal{U}$  un grand sous-ensemble de  $\mathcal{X}$  où les étiquettes sont initialement inconnues, notre objectif est de concevoir des classifieurs  $\{g_c\}_{c=1}^{nc}$  en étiquetant de façon interactive une petite portion de  $\mathcal{U}$ , et d'entraîner les paramètres de  $\{g_c\}_c$ . Cette annotation interactive et l'entraînement se nomme apprentissage actif.

### 6.2.1 Classification d'images de façon interactive : rappel

L'apprentissage actif a été décrit dans les chapitres précédents, néanmoins un rappel des notations et de l'algorithme s'impose afin de faciliter la lecture de ce chapitre. Notre algorithme de classification d'images et de détection de changements est interactif; il est basé sur un modèle de questions & réponses qui montre un sous-ensemble informatif d'images (appelé "affichage") à un oracle, récupère ses annotations et entraîne un critère de décision  $f$  grâce à ces annotations. Considérant  $\mathcal{D}_t \subset \mathcal{I}$  comme un affichage montré à l'oracle à l'itération  $t$ , et  $\mathcal{Y}_t$  comme les étiquettes inconnues de  $\mathcal{D}_t$ ; en pratique  $|\mathcal{D}_t|$  et le nombre maximum d'itérations (noté  $T$ ) sont définis en fonction d'un budget d'annotation fixé au préalable. A partir d'un affichage aléatoire  $\mathcal{D}_0$ , nous construisons notre critère de classification d'images et de détection de changements itérativement tout en exécutant les étapes ci-après pour  $t = 0, \dots, T - 1$  :

1. Interroger l'oracle à propos des étiquettes de  $\mathcal{D}_t$  et entraîner une fonction de décision  $f_t(\cdot)$  sur  $\cup_{k=0}^t (\mathcal{D}_k, \mathcal{Y}_k)$ ; dans nos expériences ces fonctions de décisions  $\{f_t(\cdot)\}_t$  sont des classifieurs de marge construits par dessus les caractéristiques de convolution.
2. Sélectionner le prochain affichage  $\mathcal{D}_{t+1} \subset \mathcal{I} \setminus \cup_{k=0}^t \mathcal{D}_k$  à montrer à l'oracle; il est clair qu'une stratégie de force brute qui considèrerait tous les affichages possibles  $\mathcal{D} \subset \mathcal{I} \setminus \cup_{k=0}^t \mathcal{D}_k$ , apprendrait les classifieurs  $f_{t+1}(\cdot)$  sous-jacents sur  $\mathcal{D} \cup_{k=0}^t \mathcal{D}_k$  et évaluerait leur précision serait très combinatoire. Les stratégies de sélection d'affichage, liées à l'apprentissage actif, sont privilégiées et permettent l'apprentissage d'affichage plus facile à traiter. Cependant, la conception des stratégies de sélection d'affichage doit être réalisée avec soin car beaucoup de ces heuristiques sont équivalentes (ou pires) aux stratégies d'affichage de base qui choisissent les données de manière uniformément aléatoire (HUIJSER et GEMERT 2017).

Le modèle d'affichage que nous proposons dans cet article est différent des stratégies d'échantillonnage habituelles (voir SAHBI et al. 2021) et repose sur la synthèse d'exemples (également appelés exemples virtuels) qui maximisent la diversité, la représentativité ainsi que l'incertitude. La diversité vise à concevoir des exemples qui permettent d'explorer différents modes de  $f_{t+1}(\cdot)$  (c'est-à-dire des exemples bien répartis dans l'espace, de sorte à bien modéliser la fonction de décision dans les différentes parties de l'espace) alors que la représentativité fait que ces exemplaires ressemblent le plus possible aux données d'entrée, ce qui permet aussi d'obtenir des annotations plus fiables de l'oracle. Enfin, l'ambiguïté cherche à affiner localement la fonction de décision apprise  $f_{t+1}(\cdot)$ . Tous les détails de notre modèle d'affichage proposé sont présentés dans la section suivante qui constitue la principale contribution de ce chapitre.

### 6.2.2 Un nouveau modèle d'affichage virtuel

Nous considérons un cadre qui attribue pour chaque échantillon  $\mathbf{x}_i$  une distribution conditionnelle  $\{\mu_{ik}\}_{k=1}^K$  mesurant la probabilité d'attribuer  $\mathbf{x}_i$  à chacun des  $K$ -exemples virtuels, et ces derniers constituent l'affichage ultérieur  $\mathcal{D}_{t+1}$ . Contrairement aux chapitres précédents (cf. chapitre 4 et chapitre 5), la méthode proposée dans ce chapitre ne nécessite pas de seuillage dur ni de classement des appartenances  $\{\mu_{ik}\}_{k=1}^K$  afin de définir  $\mathcal{D}_{t+1}$ ; au lieu de cela, les entrées de  $\mathcal{D}_{t+1}$ , également désignées par  $\mathbf{V}$ , ainsi que  $\{\mu_{ik}\}_{k=1}^K$ , sont trouvées en minimisant une nouvelle fonction objectif définie par :

$$\min_{\mathbf{V}; \mu \geq 0; \mu \mathbf{1}_K = \mathbf{1}_n} \mathbf{tr}(\mu d(\mathbf{V}, \mathbf{X})') + \alpha [\mathbf{1}'_n \mu] \log[\mathbf{1}'_n \mu]' + \beta \mathbf{tr}(f(\mathbf{V})' \log f(\mathbf{V})) + \gamma \mathbf{tr}(\mu' \log \mu), \quad (6.1)$$

ici  $'$  est l'opérateur de transposition de matrice,  $\mathbf{1}_K, \mathbf{1}_n$  désignent deux vecteurs de  $K$  et  $n$  uns respectivement,  $\mu \in \mathbb{R}^{n \times K}$  est une matrice apprise qui fournit l'appartenance de chaque échantillon  $\mathbf{x}_i$  au  $k^{\text{ème}}$  exemple virtuel et  $\log$  est appliqué entrée par entrée. Dans la fonction objectif ci-dessus,  $d(\mathbf{V}, \mathbf{X}) \in \mathbb{R}^{K \times n}$  est la matrice de distance euclidienne entre les exemples virtuels dans  $\mathbf{V}$  et les données originales dans  $\mathbf{X}$  tandis que  $f(\mathbf{V}) \in \mathbb{R}^{2 \times K}$  est une matrice de scores dont les colonnes donnent la réponse de la fonction de décision apprise  $f_t(\cdot) \in [0, 1]$  et son complémentaire  $1 - f_t(\cdot)$  sur les  $K$  exemples virtuels. Le premier terme de la fonction objectif (réécrit comme  $\sum_i \sum_k \mu_{ik} \|\mathbf{x}_i - \mathbf{V}_k\|_2^2$ ) mesure la *représentativité* des exemples synthétiques  $\{\mathbf{V}_k\}_k$ ; cela modélise à quel point chaque échantillon d'entraînement  $\mathbf{x}_i$  est proche du plus proche (ou plus représentatif)  $\mathbf{V}_k$ , et s'annule lorsque tous les échantillons d'entraînement coïncident avec leurs exemples virtuels. Le second terme (équivalent à  $\sum_k [\frac{1}{n} \sum_{i=1}^n \mu_{ik}] \log[\frac{1}{n} \sum_{i=1}^n \mu_{ik}]$ ) capture la *diversité* des données virtuelles générées comme l'entropie de la distribution de probabilité des appartenances sous-jacentes; cette mesure est minimale lorsque les échantillons d'entraînement sont assignés à des exemples virtuels différents, et vice-versa. Le troisième critère (réécrit comme  $\sum_k \sum_c [f_c(\mathbf{V})]'_k [\log f_c(\mathbf{V})]_k$ ) mesure l'*ambiguïté* (ou incertitude) dans  $\mathcal{D}_{t+1}$  comme l'entropie de la fonction de scores; sa valeur la plus petite est atteinte lorsque les exemples virtuels dans  $\mathcal{D}_{t+1}$  ont des scores égaux pour les différentes classes. Finalement, le quatrième terme considère, sans aucun a priori sur les trois autres termes, la distribution d'appartenance de chaque donnée d'entrée est uniforme, il agit donc comme régularisateur et aide également à obtenir une solution analytique. Tous ces termes sont pondérés par les coefficients  $\alpha, \beta, \gamma \geq 0$ . Nous considérons des contraintes d'égalité et d'inégalité ce qui garantit que l'appartenance de chaque échantillon  $\mathbf{x}_i$  aux exemples virtuels forme une distribution de probabilité.

### 6.2.3 Optimisation

**Proposition 6.1.** *Les conditions d'optimalité de l'équation 6.1 conduisent à la solution :*

$$\begin{aligned}\mu^{(\tau+1)} &:= \mathbf{diag}(\hat{\mu}^{(\tau+1)} \mathbf{1}_K)^{-1} \hat{\mu}^{(\tau+1)} \\ \mathbf{V}^{(\tau+1)} &:= \hat{\mathbf{V}}^{(\tau+1)} \mathbf{diag}(\mathbf{1}'_n \mu^{(\tau)})^{-1},\end{aligned}\tag{6.2}$$

avec  $\hat{\mu}^{(\tau+1)}, \hat{\mathbf{V}}^{(\tau+1)}$  qui sont respectivement :

$$\begin{aligned}\exp\left(-\frac{1}{\gamma}[d(\mathbf{X}, \mathbf{V}^{(\tau)}) + \alpha(\mathbf{1}_n \mathbf{1}'_K + \mathbf{1}_n \log \mathbf{1}'_n \mu^{(\tau)})]\right) \\ \mathbf{X} \mu^{(\tau)} + \beta \sum_c \nabla_v f_c(\mathbf{V}^{(\tau)}) \circ (\mathbf{1}_d [\log f_c(\mathbf{V}^{(\tau)})]' + \mathbf{1}_d \mathbf{1}'_K),\end{aligned}\tag{6.3}$$

ici  $\circ$  est le produit matriciel d'Hadamard et  $\mathbf{diag}(\cdot)$  construit une matrice diagonale à partir d'un vecteur.

Les détails de la preuve résultent des conditions d'optimalité du gradient et lagrangien de l'équation 6.1. En considérant la proposition ci-dessus,  $\hat{\mu}^{(0)}$  et  $\hat{\mathbf{V}}^{(0)}$  sont initialement fixés à des valeurs aléatoires et, en pratique, la solution converge vers les points fixes (notés  $\tilde{\mu}, \tilde{\mathbf{V}}$ ) en quelques itérations. Ces points fixes définissent les exemples virtuels les plus *pertinents* de l'affichage  $\mathcal{D}_{t+1}$  (d'après le critère 1) qui sont utilisés pour entraîner le classifieur  $f_{t+1}$  (cf. [Algorithme 6.1](#)).

### 6.2.4 Inversion de réseaux profonds

Nous proposons également une version de notre algorithme utilisant l'apprentissage profond : les exemples virtuels sont ici obtenus par une inversion du réseau profond grâce à une fonction de perte adverse qui permet la synthèse des exemples les plus représentatifs et diversifiés, avec les scores de classification les plus ambigus. L'objectif est donc de générer des exemples critiques pour les annotations futures, avec l'impact le plus important sur les classifieurs appris par la suite.

**Inversion du réseau profond.** Nous reprenons les mêmes notations que précédemment, mais cette fois-ci le classifieur  $f$  est un CNN, entraîné sur un sous-ensemble des exemples virtuels les plus critiques (noté  $\{\mathbf{V}_k\}_k$ ) dont l'étiquetage est obtenu de façon frugale par l'oracle. Nous définissons une distribution  $\{\mu_{ik}\}_{i=1}^n$

**Procédure 6.1 Mécanisme de sélection d’affichage.**


---

```

1: procédure VIRTUAL_EXEMPLARS( $\mathcal{I}, \mathcal{D}_0 \subset \mathcal{I}, T$ ) :
2: Entrée :  $\mathcal{I}$  : ensemble des images,  $\mathcal{D}_0 \subset \mathcal{I}$  : ensemble annoté initial,  $T$  : budget.
3: Sortie :  $\cup_{t=0}^{T-1}(\mathcal{D}_t, \mathcal{Y}_t)$  : ensembles annotés obtenus,  $\{f_t\}_t$  classifieurs entraînés
   à partir des données annotées.
4:   for  $t = 0$  to  $T - 1$  do
5:      $\mathcal{Y}_t \leftarrow \text{oracle}(\mathcal{D}_t)$ 
6:      $f_t \leftarrow \arg \min_f \text{Loss}(f, \cup_{k=0}^t(\mathcal{D}_k, \mathcal{Y}_k))$ 
7:      $\tau \leftarrow 0$ ;  $\hat{\mu}^{(0)} \leftarrow \text{random}$ ;  $\hat{\mathbf{V}}^{(0)} \leftarrow \text{random}$ 
8:     Définir  $\mu^{(0)}$  et  $\mathbf{V}^{(0)}$  en utilisant l’équation 6.2 et l’équation 6.3
9:     while ( $\|\mu^{(\tau+1)} - \mu^{(\tau)}\|_1 + \|\mathbf{V}^{(\tau+1)} - \mathbf{V}^{(\tau)}\|_1 \geq \epsilon \wedge \tau < \text{maxiter}$ ) do
10:       Définir  $\mu^{(\tau+1)}$  et  $\mathbf{V}^{(\tau+1)}$  grâce à l’équation 6.2 et l’équation 6.3
11:        $\tau \leftarrow \tau + 1$ 
12:     end while
13:      $\tilde{\mu} \leftarrow \mu^{(\tau)}$ ;  $\tilde{\mathbf{V}} \leftarrow \mathbf{V}^{(\tau)}$ 
14:      $\mathcal{D}_{t+1} \leftarrow \{\mathbf{x}_i \in \mathcal{I} \setminus \cup_{k=0}^t \mathcal{D}_k : \mathbf{x}_i \leftarrow \arg \min_{\mathbf{x}} \|\mathbf{x} - \mathbf{V}_k\|_2^2\}_{k=1}^K$ 
15:   end for
16: return  $\cup_{t=0}^{T-1}(\mathcal{D}_t, \mathcal{Y}_t), \{f_t\}_t$ 
17: end procédure

```

---

qui mesure la probabilité conditionnelle d’assigner les  $n$  échantillons non étiquetés de  $\mathcal{I}$  à  $\mathbf{V}_k$ . Avec cette définition,  $\{\mathbf{V}_k\}_k$  et  $\{\mu_{ik}\}_{i,k}$  sont obtenus avec :

$$\begin{aligned}
\min_{\{\mathbf{V}_k; \mu_{ik} \geq 0; \sum_i \mu_{ik} = 1\}_{i,k}} & \sum_{i=1}^n \sum_{k=1}^K \mu_{ik} \|\mathbf{x}_i - \mathbf{V}_k\|_2^2 + \alpha \sum_{k=1}^K \left[ \frac{1}{n} \sum_{i=1}^n \mu_{ik} \right] \log \left[ \frac{1}{n} \sum_{i=1}^n \mu_{ik} \right] \\
& + \beta \sum_{k=1}^K f_{\text{sm}}(\mathbf{V}_k) \log f_{\text{sm}}(\mathbf{V}_k) + (1 - f_{\text{sm}}(\mathbf{V}_k)) \log(1 - f_{\text{sm}}(\mathbf{V}_k))
\end{aligned} \tag{6.4}$$

Les contraintes d’égalité et inégalité garantissent le caractère stochastique, cette fois-ci, des appartenances  $\{\mu_{ik}\}_i$  pour chaque exemple virtuel  $\mathbf{V}_k$ . Le premier terme de l’équation 6.4 mesure à quel point les  $\{\mathbf{V}_k\}_k$  sont représentatifs des échantillons non étiquetés dans  $\mathcal{I}$ ; de par la stochasticité de  $\mu$ , ce terme atteint plus facilement de petites valeurs, ce qui permet la production d’exemples virtuels plus proches des données d’entrée et donc d’hériter d’étiquettes plus précises lorsque l’oracle annote leurs données les plus proches. Le second terme capture la diversité de ces exemples comme l’entropie de la distribution de probabilité des appartenances sous-jacentes. Finalement, le troisième critère mesure l’ambiguïté (ou incertitude) des  $\{\mathbf{V}_k\}_k$  comme l’entropie du softmax  $f_{\text{sm}}$  de  $f$ ; il atteint sa plus petite valeur lorsque les exemples virtuels ont le même score sur toutes les classes.

## 6.3 Expériences de détections des changements dans des images satellites

Les expériences de détection de changements sont à nouveau effectuées sur le jeu de données Jefferson, comme dans les chapitres précédents.

### 6.3.1 Etude d'ablation

Afin d'étudier l'impact des différents termes de notre fonction objectif, nous menons à nouveau une étude qui les considère individuellement, pairs à pairs et tous utilisés conjointement. Dans cette étude, le dernier terme de l'équation 6.1 est toujours gardé comme régularisateur et il permet d'obtenir la solution analytique de l'équation 6.2. L'impact de chacun de ces termes et leur combinaison est montré sur le tableau 6.1. Nous observons ainsi que la combinaison repré-

TABLE 6.1 – **Jefferson étude d'ablation modèle d'exemples virtuels.** Ce tableau montre une étude d'ablation de notre modèle d'affichage. Ici, Rep., Div. et Amb. représentent respectivement la représentativité, la diversité et l'ambiguïté. Ces résultats sont montrés à différentes itérations  $t = 0, \dots, T - 1$  (Iter) et leurs taux d'échantillonnage sous-jacents (Samp), définis comme  $(\sum_{k=0}^{t-1} |\mathcal{D}_k| / (|\mathcal{I}/2|)) \times 100$ . La valeur AUC (Area Under Curve) correspond à la moyenne arithmétique des EERs sur l'ensemble des itérations. Une valeur AUC plus basse implique de meilleures performances.

| Rep.  | Div. | Amb. | 1     | 2     | 3     | 4    | 5    | 6    | 7     | 8     | 9     | 10    | AUC          |
|-------|------|------|-------|-------|-------|------|------|------|-------|-------|-------|-------|--------------|
| ×     | ×    | ✓    | 47.81 | 27.29 | 11.15 | 7.97 | 8.18 | 7.31 | 7.97  | 7.94  | 7.50  | 7.90  | 14.10        |
| ×     | ✓    | ×    | 47.81 | 18.72 | 11.24 | 7.97 | 8.18 | 7.29 | 7.59  | 7.88  | 7.50  | 7.90  | 13.21        |
| ✓     | ×    | ×    | 47.81 | 35.98 | 16.86 | 6.52 | 4.98 | 2.67 | 2.03  | 1.80  | 1.45  | 1.30  | 12.14        |
| ✓     | ×    | ✓    | 47.81 | 40.40 | 23.86 | 9.56 | 7.65 | 5.75 | 5.47  | 6.12  | 4.40  | 5.72  | 15.67        |
| ×     | ✓    | ✓    | 47.81 | 27.29 | 11.15 | 7.97 | 8.18 | 7.31 | 7.97  | 7.94  | 7.50  | 7.90  | 14.10        |
| ✓     | ✓    | ×    | 47.81 | 29.84 | 17.63 | 6.21 | 4.40 | 2.70 | 1.98  | 1.92  | 1.65  | 1.52  | 11.57        |
| ✓     | ✓    | ✓    | 47.81 | 27.61 | 11.76 | 5.74 | 2.95 | 2.39 | 1.89  | 1.61  | 1.55  | 1.34  | <b>10.47</b> |
| Samp% |      |      | 1.45  | 2.90  | 4.36  | 5.81 | 7.27 | 8.72 | 10.18 | 11.63 | 13.09 | 14.54 | -            |

sentativité+diversité a le plus grand impact, surtout aux premières itérations de la détection de changements, tandis que l'impact du terme d'ambiguïté s'élève plus tard de sorte à affiner localement les fonctions de décision (*ie.* une fois que les modes de la distribution des données deviennent bien explorés). Ces performances EERs sont montrées pour différents taux d'échantillonnage définis – à

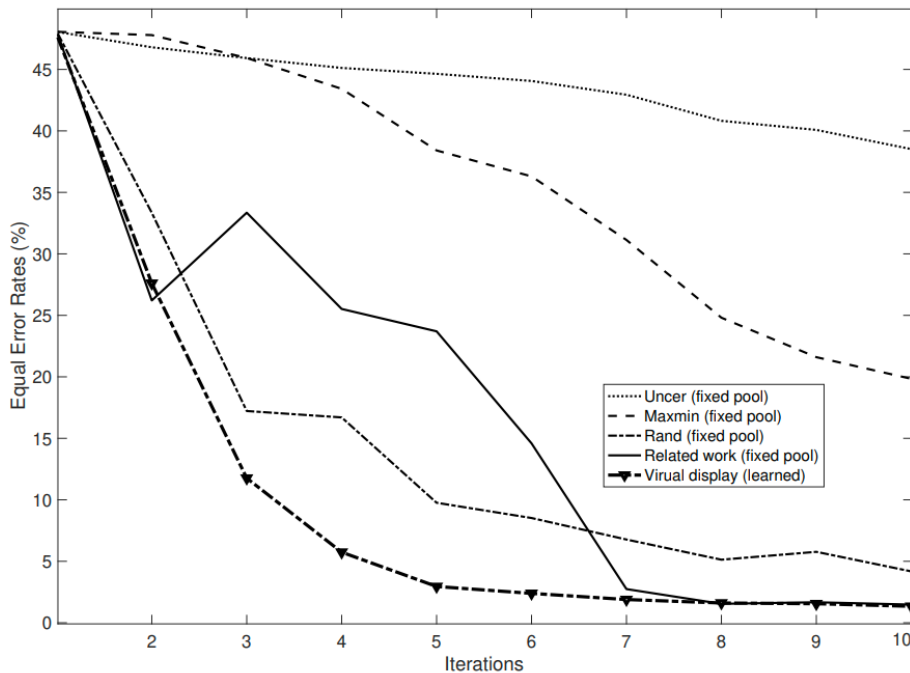


chaque itération  $t$  – comme  $(\sum_{k=0}^{t-1} |\mathcal{D}_k| / (|\mathcal{I}/2|)) \times 100$  avec  $|\mathcal{I}| = 2200$  et  $|\mathcal{D}_k|$  fixé à 16.

### 6.3.2 Comparaison avec les baselines

Nous étudions plus en détail la force de notre modèle d’affichage contre d’autres stratégies d’échantillonnage telles que la *recherche aléatoire*, *maxmin* et *l’incertitude*. L’aléatoire sélectionne les échantillons du pool de données d’entraînement non étiquetées tandis que l’incertitude sélectionne, depuis le même pool, l’affichage dont les scores de classification sont les plus ambigus (*ie.* les plus proches de zéro). La baseline maxmin consiste à échantillonner de façon glou-tonne les données dans  $\mathcal{D}_{t+1}$ ; chaque échantillon dans  $\mathbf{x}_i \in \mathcal{D}_{t+1} \subset \mathcal{I} \setminus \cup_{k=0}^t \mathcal{D}_k$  est choisi en maximisant sa distance minimum par rapport à  $\cup_{k=0}^t \mathcal{D}_k$ , ce qui conduit aux échantillons les plus distincts dans  $\mathcal{D}_{t+1}$ . Nous comparons notre modèle d’af-

FIGURE 6.1 – **Jefferson comparaison de notre modèle virtuel avec les baselines.** Cette figure montre une comparaison des différentes stratégies d’échantillonnage à différentes itérations (*Iter*) et les taux d’échantillonnage sous-jacents du tableau 6.1 (*Samp*). Ici *Uncer* et *Rand* se réfèrent à l’échantillonnage par incertitude et aléatoire respectivement. A noter que l’apprentissage *fully supervised* obtient un *EER* de 0.94%. *Related work* se réfère à la méthode du chapitre 4.



fichage avec notre méthode du premier chapitre des contributions, qui consiste à



assigner une mesure de probabilité à l'ensemble entier des données non étiquetées et de sélectionner l'affichage avec les plus hautes probabilités. Nous montrons également les performances avec le mode fully supervised comme borne supérieure, qui consiste à construire un unique classifieur grâce au jeu de données entier avec les annotations récupérées de la vérité terrain. Les performances EERs visibles sur la [figure 6.1](#) (pour différentes itérations et taux d'échantillonnage) montrent l'impact positif du modèle d'affichage virtuel que nous proposons face aux stratégies d'échantillonnage susmentionnées. Hormis le modèle du [chapitre 4](#), la plupart de ces méthodes comparatives ne parviennent pas à trouver les classes rares avec du changement de façon satisfaisante. En effet, les baselines random et maxmin capturent la diversité dans la phase initiale de la recherche interactive sans être capables d'affiner la fonction de décision aux dernières itérations. En revanche, l'incertitude affine bien la fonction de décision mais manque de diversité. La stratégie d'affichage du premier chapitre des contributions réunit les avantages de random et maxmin ainsi que l'incertitude, mais souffre d'une rigidité dans l'affichage sélectionné (surtout aux premières itérations), qui est pris d'un ensemble fixe de données d'entraînement tandis que le modèle d'affichage virtuel (proposé dans ce chapitre) est appris et donc plus flexible et efficace dans des régimes hautement frugaux.

## 6.4 Expériences de classification d'images

Nous avons également mené des expériences sur la tâche de classification d'images, sur la base de données Object-DOTA, avec la version profonde de notre algorithme. L'implémentation de notre algorithme profond a nécessité un travail d'ingénierie et plusieurs réglages détaillés dans la [section 6.4.1](#); ces résultats récents, même préliminaires montrent déjà l'efficacité de notre méthode.

### 6.4.1 Détails d'implémentation

Les expériences ont été faites sur un GPU Nvidia GTX 1070 avec 8GB de RAM. Il existe peu de GPUs grand public avec plus de RAM, certains ayant 11GB ou 12GB de RAM mais rarement plus. Plusieurs étapes ont nécessité des accommodations afin de ne pas dépasser la capacité mémoire de la carte graphique, notamment la sauvegarde des caractéristiques de chacune des données. En effet, la matrice de représentativité nécessite les caractéristiques de toutes les données d'entrée afin de calculer la distance entre les exemples virtuels et ces données d'entrée, ce qui représente une très grosse matrice pour le jeu de données Object-DOTA avec quasiment 100000 images.

La taille initiale des images était fixée à  $224 \times 224$ , ce qui correspond à la taille d'entrée du CNN ResNet-18 que nous utilisons. Il a fallu grandement réduire cette taille d'entrée, ce qui nuit aux performances ; il est possible de trouver un moyen de contourner le problème afin de ne pas réduire les performances outre-mesure, néanmoins en utilisant la même taille d'entrée réduite pour chacune des méthodes d'apprentissage actif, la comparaison des méthodes reste pertinente. Ainsi, en pratique, nous divisons la taille des images par 8, on passe donc d'images RVB à  $3 \times 224 \times 224 = 150528$  caractéristiques, à  $3 \times 28 \times 28 = 2352$  caractéristiques. De plus, plutôt que d'utiliser la simple précision (float32), nous convertissons ces images en semi-précision (float16), afin de réduire encore plus l'empreinte mémoire. Le [tableau 6.2](#) montre la différence relative de performances selon la taille des images en entrée. De plus, certaines opérations intermédiaires, notamment dans le calcul

TABLE 6.2 – **Object-DOTA. Comparaison des performances selon la taille des images.** Une valeur d'accuracy plus haute implique de meilleures performances.

| Method        | Img     | 100          | 200   | 300   | 400   | 5000  | 600   | 700   | 800          | 900          | 1000         |
|---------------|---------|--------------|-------|-------|-------|-------|-------|-------|--------------|--------------|--------------|
|               |         | 0.1%         | 0.2%  | 0.3%  | 0.4%  | 0.5%  | 0.6%  | 0.7%  | 0.8%         | 0.9%         | 1%           |
| Virtual (our) | $28^2$  | <b>30.71</b> | 17.5  | 36.81 | 43.48 | 45.88 | 49.9  | 50.5  | <b>50.97</b> | <b>46.22</b> | <b>52.44</b> |
| Random        | $28^2$  | 24.18        | 32.45 | 39.25 | 46.17 | 47.92 | 51.06 | 50.79 | 49.67        | 44.71        | 46.18        |
| Random        | $56^2$  | 31.01        | 37.43 | 50.35 | 59.63 | 62.4  | 65.05 | 65.71 | 65.98        | 62.08        | 65.35        |
| Random        | $112^2$ | 39.88        | 49.55 | 59.46 | 69.15 | 67.98 | 72.04 | 72.97 | 73.58        | 73.31        | 72.91        |
| Random        | $224^2$ | 36.51        | 44.62 | 57.35 | 66.36 | 68.37 | 69.82 | 70.68 | 70.36        | 66.97        | 69.94        |

de  $\mu$  ou pour la fonction de perte de  $V$ , créaient des matrices trop grosses pour tenir en mémoire, il a donc fallu séparer certaines matrices en petites fractions, effectuer les calculs sur ces dernières puis concaténer les résultats une fois les opérations terminées. Notamment,  $\mu$ , la matrice de représentativité et  $\mu * \mathbf{1}_K$  ont dû être fractionnés de la sorte. Il est à noter que les données virtuelles sont utilisées comme features, dont les données réelles les plus proches (selon ces features) sont encore une fois utilisées pour l'annotation par l'oracle et l'entraînement des différents réseaux.

Les paramètres que nous avons utilisés pour nos expériences sont disponibles dans le [tableau 6.3](#), nous avons gardé le même réseau et optimiseur, respectivement ResNet-18 et SGD, pour la base Object-DOTA.

## 6.4.2 Performances de notre méthode

Le [tableau 6.4](#) montre que notre méthode obtient de bonnes performances sur Object-DOTA, très proches des meilleures baselines random, maxmin et margin. L'accuracy moyenne n'est pas toujours la meilleure, cependant l'écart type est net-

TABLE 6.3 – **Paramètres pour la classification d’images.** Une description détaillée des paramètres que nous avons utilisés pour nos CNNs pour les expériences de classification d’images.

| Paramètre         | Valeur            | Optimiseur SGD | Valeur |
|-------------------|-------------------|----------------|--------|
| nb_cycles         | 10                | initial_lr     | 0.01   |
| nb_labels         | 100               | lr             | 0.01   |
| nb_samples        | 98906             | maximize       | False  |
| nb_classes        | 15                | momentum       | 0.9    |
| nb_features       | 2352              | nesterov       | False  |
| test_dataset_size | 28853             | weight_decay   | 0.0005 |
| model             | ResNet-18         |                |        |
| criterion         | CrossEntropyLoss  |                |        |
| scheduler         | CosineAnnealingLR |                |        |

tement inférieur aux baselines à partir de la troisième itération d’apprentissage actif. Le nombre d’exemples virtuels utilisés a été limité à 10 pour des soucis d’empreinte mémoire, et les paramètres de pondération  $\alpha, \beta$  n’ont pas été modifiés; une marge de progression est donc possible avec notre méthode.

TABLE 6.4 – **Object-DOTA. Comparaison avec les baselines.** Une valeur d’accuracy plus haute implique de meilleures performances.

| Method        | 100<br>0.1%   | 200<br>0.2%   | 300<br>0.3%         | 400<br>0.4%   | 500<br>0.5%   |
|---------------|---------------|---------------|---------------------|---------------|---------------|
| Virtual (our) | 36.97 ± 12.40 | 45.20 ± 11.71 | 44.53 ± 6.91        | 51.14 ± 4.77  | 54.65 ± 2.97  |
| Flat (our)    | 24.1 ± 12.22  | 28.87 ± 15.24 | 30.79 ± 13.09       | 34.86 ± 9.18  | 38.62 ± 1.55  |
| Entropy       | 25.16 ± 9.7   | 26.29 ± 14.67 | 31.99 ± 11.49       | 33.45 ± 10.3  | 35.14 ± 10.31 |
| Maxmin        | 36.77 ± 16.46 | 50.48 ± 12.57 | 54.03 ± 13.07       | 56.65 ± 10.82 | 57.85 ± 10.93 |
| Least conf.   | 23.03 ± 8.67  | 32.11 ± 3.46  | 35.97 ± 7.22        | 36.34 ± 6.33  | 37.75 ± 8.96  |
| Margin        | 36.25 ± 13.57 | 50.09 ± 9.56  | 52.66 ± 11.03       | 54.9 ± 10.18  | 56.16 ± 9.44  |
| Random        | 38.85 ± 13.81 | 49.45 ± 9.92  | 51.2 ± 11.16        | 54.55 ± 10.2  | 56.98 ± 9.84  |
| Method        | 600<br>0.6%   | 700<br>0.7%   | 800<br>0.8%         | 900<br>0.9%   | 1000<br>1%    |
| Virtual (our) | 57.02 ± 2.77  | 59.43 ± 3.53  | <b>61.09 ± 4.83</b> | 58.54 ± 6.01  | 61.70 ± 7.64  |
| Flat (our)    | 35.3 ± 9.04   | 41.94 ± 0.91  | 40.65 ± 1.81        | 40.08 ± 1.36  | 40.29 ± 1.7   |
| Entropy       | 41.19 ± 6.49  | 42.1 ± 5.79   | 42.18 ± 8.68        | 40.79 ± 9.15  | 42.39 ± 5.58  |
| Maxmin        | 59.64 ± 9.8   | 61.08 ± 8.9   | 60.77 ± 7.72        | 59.13 ± 10.81 | 63.39 ± 8.76  |
| Least conf.   | 37.17 ± 8.65  | 39.32 ± 7.93  | 40.26 ± 7.2         | 36.11 ± 11.34 | 39.77 ± 6.83  |
| Margin        | 56.52 ± 8.76  | 56.47 ± 8.34  | 58.19 ± 7.77        | 59.83 ± 8.57  | 59.61 ± 7.07  |
| Random        | 59.14 ± 9.46  | 59.44 ± 9.01  | 59.41 ± 7.89        | 58.97 ± 10.61 | 61.13 ± 9.88  |

## 6.5 Conclusion

Nous présentons dans ce chapitre une nouvelle méthode interactive de classification d'images basée sur l'apprentissage actif. La force de la méthode proposée réside dans la flexibilité du modèle d'affichage appris qui permet d'entraîner des exemples virtuels. Ces derniers sont trouvés tout en maximisant leur diversité et leur représentativité ainsi que leur ambiguïté, créant ainsi un contexte adversaire qui met à l'épreuve le classifieur actuel et améliore le suivant. Des expériences, menées sur la tâche de détection de changements dans des image satellites, montrent la performance supérieure de notre modèle d'affichage virtuel par rapport à différents modèles et stratégies d'échantillonnage pour l'apprentissage actif.



## CONCLUSION

Les méthodes d'apprentissage profond obtiennent des résultats très impressionnants dans de nombreux domaines, néanmoins leur efficacité dépend grandement du nombre de données annotées disponibles pour entraîner leurs algorithmes. L'annotation des données a un réel coût dans diverses industries, où des ingénieurs ou experts vont l'effectuer manuellement durant des heures voire des jours. Diverses solutions existent pour pallier ce problème ou tout du moins de minimiser les coûts de ce processus d'annotation des données.

Ainsi, les travaux présentés dans ce manuscrit de thèse visent à réduire le nombre d'annotations nécessaires pour l'entraînement d'un modèle d'apprentissage automatique performant dans deux tâches de vision par ordinateur : la détection de changements dans les images satellites, et la classification d'images. Pour ce faire, nous avons étudié diverses méthodes telles que le *few-shot learning* ou l'apprentissage auto-supervisé, avant de conclure que l'apprentissage actif était la méthode d'apprentissage frugal (avec peu de données et/ou d'annotations disponibles) la plus adaptée à notre problématique métier. L'apprentissage actif a pour but de sélectionner les données les plus informatives afin de les annoter en priorité pour qu'un modèle d'apprentissage automatique soit le plus performant possible, grâce à une fonction d'acquisition des données qui va déterminer lesquelles présenter à un oracle qui connaît la vérité terrain. La difficulté réside dans la façon d'identifier quelles données seront les plus informatives avant d'avoir accès à leurs étiquettes.

### 7.1 Synthèse

Nos premiers travaux ont porté sur la conception d'une méthode d'apprentissage actif adaptée aux réseaux profonds, en réunissant des critères classiques d'apprentissage actif, tels que l'ambiguïté mais également la diversité et représentativité des données, dans une même fonction objectif probabiliste. Nous avons obtenu de bons résultats en détection de changements dans les images satellites, cependant pour l'application de classification d'images les performances étaient relativement moins bonnes. Néanmoins, nous avons observé des résultats inté-

ressants en faisant varier les différents paramètres de pondération de nos trois critères principaux que sont l'ambiguïté, la diversité et la représentativité des données. En effet, au fur et à mesure des cycles d'apprentissage actif, différentes stratégies se sont révélées plus ou moins efficaces, ce qui nous a donné l'idée de faire varier ces pondérations au cours du temps. Ces différents résultats ainsi que la réflexion qui nous a poussé à faire évoluer notre stratégie initiale ont été présentés dans le [chapitre 4](#), et ont donné lieu à une publication dans IGARSS (SAHBI et al. 2021).

Cette idée nous a amené aux travaux du [chapitre 5](#), où nous avons utilisé l'apprentissage par renforcement pour faire varier nos paramètres de pondération de l'ambiguïté, diversité et représentativité des données au fur et à mesure des itérations d'apprentissage actif. Ainsi, grâce à une fonction de récompense habilement choisie, nous déterminons si nous devons plutôt mettre l'accent sur l'ambiguïté, la diversité ou la représentativité des données pour les prochaines itérations. Les résultats obtenus sont très encourageants et ont donné lieu à deux publications (DESCHAMPS et SAHBI 2022a; DESCHAMPS et SAHBI 2022b), que ce soit dans la détection de changements dans les images satellites ou pour la classification d'images.

Finalement, après la conception de notre méthode dans le scénario d'apprentissage actif basé sur les *pools* de données, notre dernière contribution utilise également la génération de données afin d'augmenter le nombre de données et étiquettes disponibles pour entraîner le modèle à moindre coût. En effet, notre algorithme interactif du [chapitre 6](#) sélectionne des exemples virtuels les plus représentatifs et divers, qui remettent en cause le modèle appris afin d'obtenir un modèle de plus en plus discriminatoire au fur et à mesure des itérations d'apprentissage actif. Les expériences en détection de changements ont donné lieu à une publication (SAHBI et DESCHAMPS 2022), et des expériences avec de l'apprentissage profond ont été effectuées sur le jeu de classification d'images Object-DOTA dans le [chapitre 6](#).

## 7.2 Perspectives

Malgré les bons résultats obtenus par nos différentes méthodes d'apprentissage actif, il n'existe toujours pas de stratégie supérieure pour tous les jeux de données ou applications et chacune des méthodes de l'état de l'art a divers avantages selon le besoin en précision, vitesse de calcul, etc. De plus, il existe beaucoup d'applications dans la vision pour ordinateur, et l'on pourrait ainsi adapter notre méthode qui est pour l'instant appliquée à deux tâches, à d'autres tâches telles que la détection d'objets.

**Application de nos méthodes à d'autres tâches de détection.** Il est tout à fait possible d'adapter nos méthodes pour la détection d'objets qui fonctionne avec des réseaux de neurones profonds. En effet, les réseaux de l'état de l'art utilisent un *backbone* comme extracteur de caractéristiques, c'est-à-dire un CNN standard entraîné pour la tâche de classification d'images. Ainsi, il sera possible d'obtenir les probabilités de classe sur les images d'entraînement, nécessaires à nos méthodes d'apprentissage actif. Bien que les images en détection d'objets peuvent contenir une multitude d'objets, tandis qu'un seul objet d'intérêt est considéré en classification d'images, une première version naïve pour adapter notre méthode à la détection d'objets serait d'utiliser la moyenne des scores de probabilité obtenus par notre fonction objectif afin de décider si une image doit être donnée à l'oracle pour annotation.

De plus, un autre cas d'usage pertinent pour l'entreprise dans laquelle j'ai effectué ma thèse CIFRE, est de n'annoter que certains des objets de l'image considérés comme plus difficiles pour le réseau. Là encore, notre méthode permet d'obtenir des scores de pertinence pour chacun des objets individuellement afin d'indiquer à l'annotateur lesquels doivent être annotés dans une ou plusieurs images données.

Enfin, une autre application possible de nos méthodes serait la détection et la reconnaissance d'actions dans les vidéos et la recherche d'information (cf. L. WANG et SAHBI 2013; FERECATU et al. 2005; STOIAN et al. 2015; MAZARI et SAHBI 2020). Ces tâches ont des données encore plus difficiles et coûteuses à annoter, ainsi l'utilisation de l'apprentissage actif serait pertinente.

**Meilleure explicabilité des résultats.** Une des problématiques dans l'apprentissage actif, qui est également présente pour nos méthodes, est qu'il est parfois difficile d'expliquer clairement pourquoi une méthode fonctionne mieux qu'une autre dans un cas ou jeu de données précis, mais potentiellement moins bien sur une autre base. De fait, cela rend difficile d'anticiper les réglages à effectuer (notamment au niveau des hyperparamètres) avant d'appliquer l'algorithme sur un jeu de données différent. Toutes les méthodes ne se valent pas sur le plan de la rapidité, utiliser une méthode plus simple comme l'échantillonnage aléatoire qui a souvent des performances correctes voire meilleures que certaines méthodes de l'état de l'art peut être une bonne option.

Une meilleure explicabilité des résultats obtenus par les différentes méthodes sur les différentes tâches et jeux de données, permettrait de mieux anticiper la méthode à utiliser, ou de procéder à des réglages initiaux plus pertinents, lorsque l'on veut utiliser l'apprentissage actif sur une nouvelle tâche ou jeu de données.





## BIBLIOGRAPHIE

- ABE, Naoki (1998). "Query learning strategies using boosting and bagging". In : *International Conference on Machine Learning (ICML)* (cf. p. 56).
- AGARWAL, Sharat, Himanshu ARORA, Saket ANAND et Chetan ARORA (2020). "Contextual diversity for active learning". In : *European Conference on Computer Vision (ECCV)* (cf. p. 71).
- AMIDI, Afshine et Shervine AMIDI (2018). *CS230 - Convolutional Neural Networks*. <https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks> (cf. p. 23).
- ANGLUIN, Dana (1988). "Queries and concept learning". In : *Machine Learning* (cf. p. 50).
- ANGLUIN, Dana (2001). "Queries revisited". In : *International Conference on Algorithmic Learning Theory (ALT)* (cf. p. 50).
- ARAUJO, André, Wade NORRIS et Jack SIM (2019). "Computing receptive fields of convolutional neural networks". In : *Distill* (cf. p. 24, 110).
- ARTHUR, David et Sergei VASSILVITSKII (2007). "K-means++ the advantages of careful seeding". In : *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms* (cf. p. 85).
- ASH, Jordan T, Chicheng ZHANG, Akshay KRISHNAMURTHY, John LANGFORD et Alekh AGARWAL (2019). "Deep batch active learning by diverse, uncertain gradient lower bounds". In : *arXiv preprint library* (cf. p. 63, 74).
- ATLAS, Les, David COHN et Richard LADNER (1989). "Training connectionist networks with queries and selective sampling". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 51).
- BALDI, Pierre et Peter J SADOWSKI (2013). "Understanding dropout". In : *neurips* (cf. p. 26).
- BAUM, Eric B et Kenneth LANG (1992). "Query learning can work poorly when a human oracle is used". In : *International Joint Conference on Neural Network (IJCNN)* (cf. p. 50).
- BELUCH, William H, Tim GENEWEIN, Andreas NÜRNBERGER et Jan M KÖHLER (2018). "The power of ensembles for active learning in image classification". In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cf. p. 72).
- BIANCO, Simone, Remi CADENE, Luigi CELONA et Paolo NAPOLETANO (2018). "Benchmark analysis of representative deep neural network architectures". In : *IEEE access* 6, p. 64270-64277 (cf. p. 96).

- BLOODGOOD, Michael et K VIJAY-SHANKER (2014). "A method for stopping active learning based on stabilizing predictions and the need for user-adjustable stopping". In : *arXiv preprint library* (cf. p. 60).
- BREIMAN, Leo (1996). "Bagging predictors". In : *Machine Learning* (cf. p. 56).
- BREIMAN, Leo (2001). "Random forests". In : *Machine learning* (cf. p. 123).
- BRUNNER, Dominik, Guido LEMOINE et Lorenzo BRUZZONE (2010). "Earthquake damage assessment of buildings using VHR optical and SAR imagery". In : *IEEE Transactions on Geoscience and Remote Sensing (GRSS)* (cf. p. 132).
- BUDD, Samuel, Emma C ROBINSON et Bernhard KAINZ (2021). "A survey on active learning and human-in-the-loop deep learning for medical image analysis". In : *Medical Image Analysis* (cf. p. 9, 60).
- CAMPBELL, Trevor et Tamara BRODERICK (2019). "Automated scalable Bayesian inference via Hilbert coresets". In : *Journal of Machine Learning Research (JMLR)* (cf. p. 59).
- CASTELLANI, Andrea, Sebastian SCHMITT et Barbara HAMMER (2022). "Stream-based Active Learning with Verification Latency in Non-stationary Environments". In : *arXiv preprint library* (cf. p. 51).
- CAWLEY, Gavin C (2011). "Baseline methods for active learning". In : *International Conference on Artificial Intelligence and Statistics (AISTATS)* (cf. p. 7).
- CHATTOPADHYAY, Rita, Wei FAN, Ian DAVIDSON, Sethuraman PANCHANATHAN et Jieping YE (2013). "Joint transfer and batch-mode active learning". In : *International Conference on Machine Learning (ICML)* (cf. p. 6).
- CHEN, Binghui, Weihong DENG et Junping DU (2017). "Noisy softmax : Improving the generalization ability of dcnn via postponing the early softmax saturation". In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cf. p. 20).
- CLEVERT, Djork-Arné, Thomas UNTERTHINER et Sepp HOCHREITER (2015). "Fast and accurate deep network learning by exponential linear units (elus)". In : *arXiv preprint library* (cf. p. 18).
- COHN, David, Les ATLAS et Richard LADNER (1994). "Improving generalization with active learning". In : *Machine Learning* (cf. p. 51).
- COHN, David A, Zoubin GHAHRAMANI et Michael I JORDAN (1996). "Active learning with statistical models". In : *Journal of Artificial Intelligence Research (JAIR)* (cf. p. 52).
- CONTARDO, Gabriella, Ludovic DENOYER et Thierry ARTIÈRES (2017). "A meta-learning approach to one-step active learning". In : *arXiv preprint library* (cf. p. 2).
- CRESWELL, Antonia, Tom WHITE, Vincent DUMOULIN, Kai ARULKUMARAN, Biswa SENGUPTA et Anil A BHARATH (2018). "Generative adversarial networks : An overview". In : *IEEE signal processing magazine* (cf. p. 133).

- CULOTTA, Aron et Andrew McCALLUM (2005). "Reducing labeling effort for structured prediction tasks". In : *AAAI* (cf. p. 121).
- DAGAN, Ido et Sean P ENGELSON (1995). "Committee-based sampling for training probabilistic classifiers". In : *Machine Learning Proceedings 1995 (MLP1995)* (cf. p. 51).
- DASGUPTA, Sanjoy (2004). "Analysis of a greedy active learning strategy". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 52, 110).
- DASGUPTA, Sanjoy (2011). "Two faces of active learning". In : *Theoretical Computer Science* (cf. p. 6).
- DELANGE, Matthias, Rahaf ALJUNDI, Marc MASANA, Sarah PARISOT, Xu JIA, Ales LEONARDIS, Greg SLABAUGH et Tinne TUYTELAARS (2021). "A continual learning survey : Defying forgetting in classification tasks". In : *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (cf. p. 61).
- DEPEWEG, Stefan, Jose-Miguel HERNANDEZ-LOBATO, Finale DOSHI-VELEZ et Steffen UDLUFT (2018). "Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning". In : *International Conference on Machine Learning (ICML)* (cf. p. 74).
- DESCHAMPS, Sebastien et Hichem SAHBI (2022a). "Reinforcement-based Display Selection for Frugal Learning". In : *International Conference on Pattern Recognition (ICPR)* (cf. p. ii, iii, 11, 122, 127, 146).
- DESCHAMPS, Sebastien et Hichem SAHBI (2022b). "Reinforcement-Based Frugal Learning for Interactive Satellite Image Change Detection". In : *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)* (cf. p. ii, iii, 11, 122, 127, 146).
- DOERSCH, Carl, Abhinav GUPTA et Alexei A EFROS (2015). "Unsupervised visual representation learning by context prediction". In : *IEEE International Conference on Computer Vision (ICCV)* (cf. p. 118).
- DOSOVITSKIY, Alexey, Lucas BEYER, Alexander KOLESNIKOV, Dirk WEISSENBORN, Xiaohua ZHAI, Thomas UNTERTHINER, Mostafa DEGHANI, Matthias MINDERER, Georg HEIGOLD, Sylvain GELLY et al. (2020). "An image is worth 16x16 words : Transformers for image recognition at scale". In : *arXiv preprint library* (cf. p. 43, 44, 118, 123).
- EBERT, Sandra, Mario FRITZ et Bernt SCHIELE (2012). "Ralf : A reinforced active learning formulation for object class recognition". In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cf. p. 7).
- FELDMAN, Dan (2020). "Core-sets : Updated survey". In : *Sampling techniques for supervised or unsupervised tasks* (cf. p. 59).
- FERECATU, Marin, Nozha BOUJEMAA et Michel CRUCIANU (2005). "Hybrid visual and conceptual image representation within active relevance feedback context". In : *Proceedings of the 7th ACM SIGMM international workshop on Multimedia information retrieval* (cf. p. 52, 147).

- FOLMSBEE, Jonathan, Xulei LIU, Margaret BRANDWEIN-WEBER et Scott DOYLE (2018). "Active deep learning : Improved training efficiency of convolutional neural networks for tissue classification in oral cavity cancer". In : *International Symposium on Biomedical Imaging (ISBI)* (cf. p. 9, 60).
- FREUND, Yoav et Robert E SCHAPIRE (1997). "A decision-theoretic generalization of on-line learning and an application to boosting". In : *Journal of Computer and System Sciences (JCSS)* (cf. p. 56).
- GAL, Yarin et al. (2016). "Uncertainty in deep learning". In : (cf. p. 53, 73).
- GAL, Yarin et Zoubin GHAHRAMANI (2015). "Bayesian convolutional neural networks with Bernoulli approximate variational inference". In : *arXiv preprint library* (cf. p. 62).
- GAL, Yarin et Zoubin GHAHRAMANI (2016). "Dropout as a bayesian approximation : Representing model uncertainty in deep learning". In : *International Conference on Machine Learning (ICML)* (cf. p. 72).
- GAL, Yarin, Riashat ISLAM et Zoubin GHAHRAMANI (2017). "Deep bayesian active learning with image data". In : *International Conference on Machine Learning (ICML)* (cf. p. 62, 72, 101).
- GAVVES, Efstratios, Thomas MENSINK, Tatiana TOMMASI, Cees GM SNOEK et Tinne TUYTELAARS (2015). "Active transfer learning with zero-shot priors : Reusing past datasets for future tasks". In : *IEEE International Conference on Computer Vision (ICCV)* (cf. p. 6).
- GEIFMAN, Yonatan et Ran EL-YANIV (2017). "Deep active learning over the long tail". In : *arXiv preprint library* (cf. p. 70).
- GIRSHICK, Ross (2015). "Fast r-cnn". In : *IEEE International Conference on Computer Vision (ICCV)* (cf. p. 1).
- GISSIN, Daniel et Shai SHALEV-SHWARTZ (2019). "Discriminative active learning". In : *arXiv preprint library* (cf. p. 63).
- GOKON, Hideomi, Joachim POST, Enrico STEIN, Sandro MARTINIS, Andre TWELE, Matthias MÜCK, Christian GEISS, Shunichi KOSHIMURA et Masashi MATSUOKA (2015). "A method for detecting buildings destroyed by the 2011 Tohoku earthquake and tsunami using multitemporal TerraSAR-X data". In : *IEEE Geoscience and Remote Sensing Letters* (cf. p. 132).
- GONZALEZ, Teofilo F (1985). "Clustering to minimize the maximum intercluster distance". In : *Theoretical computer science* (cf. p. 70).
- GOODFELLOW, Ian, Yoshua BENGIO et Aaron COURVILLE (2016). *Deep Learning*. <http://www.deeplearningbook.org>. MIT Press (cf. p. 20, 27, 31).
- HE, Kaiming, Xiangyu ZHANG, Shaoqing REN et Jian SUN (2016). "Deep residual learning for image recognition". In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cf. p. 1, 39, 41).

- HEMMER, Patrick, Niklas KÜHL et Jakob SCHÖFFER (2022). "Deal : deep evidential active learning for image classification". In : *Deep Learning Applications* (cf. p. 73).
- HINTON, Geoffrey E, Nitish SRIVASTAVA, Alex KRIZHEVSKY, Ilya SUTSKEVER et Ruslan R SALAKHUTDINOV (2012). "Improving neural networks by preventing co-adaptation of feature detectors". In : *arXiv preprint library* (cf. p. 37).
- HOI, Steven CH, Rong JIN et Michael R LYU (2006). "Large-scale text categorization by batch mode active learning". In : *Proceedings of the 15th international conference on World Wide Web (WWW'06)* (cf. p. 52).
- HOSSAIN, HM Sajjad et Nirmalya ROY (2019). "Active deep learning for activity recognition with context aware annotator selection". In : *SIGKDD International Conference on Knowledge Discovery in Data Mining (SIGKDD)* (cf. p. 63).
- HOULSBY, Neil, Ferenc HUSZÁR, Zoubin GHAHRAMANI et Máté LENGYEL (2011). "Bayesian active learning for classification and preference learning". In : *arXiv preprint library* (cf. p. 52, 74).
- HUIJSER, Miriam et Jan C van GEMERT (2017). "Active decision boundary annotation with deep generative models". In : *IEEE International Conference on Computer Vision (ICCV)* (cf. p. 84, 134).
- HWA, Rebecca (2004). "Sample selection for statistical parsing". In : *Computational Linguistics* (cf. p. 54).
- IOFFE, Sergey et Christian SZEGEDY (2015). "Batch normalization : Accelerating deep network training by reducing internal covariate shift". In : *International Conference on Machine Learning (ICML)* (cf. p. 25).
- JIA, Yangqing, Evan SHELHAMER, Jeff DONAHUE, Sergey KARAYEV, Jonathan LONG, Ross GIRSHICK, Sergio GUADARRAMA et Trevor DARRELL (2014). "Caffe : Convolutional architecture for fast feature embedding". In : *ACM International Conference on Multimedia* (cf. p. 17).
- JIN, Chi, Zeyuan ALLEN-ZHU, Sebastien BUBECK et Michael I JORDAN (2018). "Is Q-learning provably efficient?" In : *Advances in Neural Information Processing Systems (NIPS)* (cf. p. 115).
- JOSHI, Ajay J, Fatih PORIKLI et Nikolaos PAPANIKOLOPOULOS (2009). "Multi-class active learning for image classification". In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cf. p. 52).
- KARLIK, Bekir et A Vehbi OLGAC (2011). "Performance analysis of various activation functions in generalized MLP architectures of neural networks". In : *International Journal of Artificial Intelligence and Expert Systems* (cf. p. 18).
- KENDALL, Alex et Yarin GAL (2017). "What uncertainties do we need in bayesian deep learning for computer vision?" In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 65).



- KIM, Kwanyoung, Dongwon PARK, Kwang In KIM et Se Young CHUN (2021). "Task-aware variational adversarial active learning". In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cf. p. 68).
- KINGMA, Diederik P et Jimmy BA (2014). "Adam : A method for stochastic optimization". In : *arXiv preprint library* (cf. p. 30, 33).
- KINGMA, Diederik P et Max WELLING (2013). "Auto-encoding variational bayes". In : *arXiv preprint library* (cf. p. 67).
- KIRKPATRICK, James, Razvan PASCANU, Neil RABINOWITZ, Joel VENESS, Guillaume DESJARDINS, Andrei A RUSU, Kieran MILAN, John QUAN, Tiago RAMALHO, Agnieszka GRABSKA-BARWINSKA et al. (2017). "Overcoming catastrophic forgetting in neural networks". In : *Proceedings of the National Academy of Sciences (PNAS)* (cf. p. 2).
- KIRSCH, Andreas, Joost VAN AMERSFOORT et Yarín GAL (2019). "Batchbald : Efficient and diverse batch acquisition for deep bayesian active learning". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 62, 63, 74).
- KÖRNER, Christine et Stefan WROBEL (2006). "Multi-class ensemble-based active learning". In : *European Conference on Machine Learning (ECML)* (cf. p. 7).
- KRASIN, Ivan, Tom DUERIG, Neil ALLDRIN, Vittorio FERRARI, Sami ABU-EL-HAJJA, Alina KUZNETSOVA, Hassan ROM, Jasper UIJLINGS, Stefan POPOV, Andreas VEIT et al. (2017). "Openimages : A public dataset for large-scale multi-label and multi-class image classification". In : *Dataset available from <https://github.com/openimages>* (cf. p. 1).
- KRIZHEVSKY, Alex (2014). "One weird trick for parallelizing convolutional neural networks". In : *arXiv preprint library* (cf. p. 38).
- KRIZHEVSKY, Alex, Vinod NAIR et Geoffrey HINTON (2010). "Cifar-10 (canadian institute for advanced research)". In : *URL <http://www.cs.toronto.edu/kriz/cifar.html>* (cf. p. 110, 117).
- KRIZHEVSKY, Alex, Ilya SUTSKEVER et Geoffrey E HINTON (2012). "Imagenet classification with deep convolutional neural networks". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 35-38).
- LECUN, Yann (1998). "The MNIST database of handwritten digits". In : *<http://yann.lecun.com/exdb/mnist/>* (cf. p. 110).
- LECUN, Yann, Yoshua BENGIO et Geoffrey HINTON (2015). "Deep learning". In : *nature* (cf. p. 31).
- LECUN, Yann, Léon BOTTOU, Yoshua BENGIO et Patrick HAFFNER (1998). "Gradient-based learning applied to document recognition". In : *Proceedings of the IEEE* (cf. p. 35).
- LECUN, Yann, Lionel D JACKEL, Brian BOSER, John S DENKER, Henry P GRAF, Isabelle GUYON, Don HENDERSON, Richard E HOWARD et William HUBBARD (1989). "Handwritten digit recognition : Applications of neural network chips and automatic learning". In : *IEEE Communications Magazine* (cf. p. 35).

- LENAIL, Alexander (2019). "NN-SVG : Publication-Ready Neural Network Architecture Schematics." In : (cf. p. 19).
- LEWIS, David D et Jason CATLETT (1994). "Heterogeneous uncertainty sampling for supervised learning". In : *Machine Learning* (cf. p. 53).
- LEWIS, David D et William A GALE (1994). "A sequential algorithm for training text classifiers". In : *Special Interest Group on Information Retrieval (SIGIR)* (cf. p. 51-53).
- LI, Xin et Yuhong GUO (2013). "Adaptive active learning for image classification". In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cf. p. 8, 52).
- LIU, Peng, Hui ZHANG et Kie B EOM (2016). "Active deep learning for classification of hyperspectral images". In : *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing (J-STARS)* (cf. p. 9, 60, 66).
- LIU, Wei, Dragomir ANGUELOV, Dumitru ERHAN, Christian SZEGEDY, Scott REED, Cheng-Yang FU et Alexander C BERG (2016). "Ssd : Single shot multi-box detector". In : *European Conference on Computer Vision (ECCV)* (cf. p. 1).
- LIU, Weibo, Zidong WANG, Xiaohui LIU, Nianyin ZENG, Yurong LIU et Fuad E ALSAADI (2017). "A survey of deep neural network architectures and their applications". In : *Neurocomputing* (cf. p. 19, 21).
- MAAS, Andrew L, Awni Y HANNUN, Andrew Y NG et al. (2013). "Rectifier nonlinearities improve neural network acoustic models". In : *International Conference on Machine Learning (ICML)* (cf. p. 18).
- MAHAPATRA, Dwarikanath, Behzad BOZORGTABAR, Jean-Philippe THIRAN et Mauricio REYES (2018). "Efficient active learning for image classification and segmentation using a sample selection and conditional generative adversarial network". In : *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)* (cf. p. 65).
- MALDONADO, Ramon et Sanda M HARABAGIU (2019). "Active deep learning for the identification of concepts and relations in electroencephalography reports". In : *Journal of Biomedical Informatics* (cf. p. 60).
- MAYER, Christoph et Radu TIMOFTE (2018). "Adversarial Sampling for Active Learning". In : (cf. p. 101).
- MAYER, Christoph et Radu TIMOFTE (2020). "Adversarial sampling for active learning". In : *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)* (cf. p. 59, 66, 90, 100, 101).
- MAZARI, Ahmed et Hichem SAHBI (2020). "Coarse-to-fine aggregation for cross-granularity action recognition". In : *IEEE International Conference on Image Processing (ICIP)* (cf. p. 147).
- MCCALLUMZY, Andrew Kachites et Kamal NIGAMY (1998). "Employing EM and pool-based active learning for text classification". In : *International Conference on Machine Learning (ICML)* (cf. p. 59).



- MIRZA, Mehdi et Simon OSINDERO (2014). "Conditional generative adversarial nets". In : *arXiv preprint library* (cf. p. 65).
- MOTTAGHI, Ali et Serena YEUNG (2019). "Adversarial representation active learning". In : *arXiv preprint library* (cf. p. 66).
- MUNJAL, Prateek, Nasir HAYAT, Munawar HAYAT, Jamshid SOURATI et Shadab KHAN (2022). "Towards robust and reproducible active learning using neural networks". In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cf. p. 99).
- MUSLEA, Ion, Steven MINTON et Craig A KNOBLOCK (2006). "Active learning with multiple views". In : *Journal of Artificial Intelligence Research (JAIR)* (cf. p. 71).
- NAIR, Vinod et Geoffrey E HINTON (2010). "Rectified linear units improve restricted boltzmann machines". In : *International Conference on Machine Learning (ICML)* (cf. p. 18).
- NASEER, Muhammad Muzammal, Kanchana RANASINGHE, Salman H KHAN, Munawar HAYAT, Fahad SHAHBAZ KHAN et Ming-Hsuan YANG (2021). "Intriguing properties of vision transformers". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 41).
- O'SHEA, Keiron et Ryan NASH (2015). "An introduction to convolutional neural networks". In : *arXiv preprint library* (cf. p. 21).
- ODENA, Augustus, Christopher OLAH et Jonathon SHLENS (2017). "Conditional image synthesis with auxiliary classifier gans". In : *International Conference on Machine Learning (ICML)* (cf. p. 67).
- OLIVER, Avital, Augustus ODENA, Colin A RAFFEL, Ekin Dogus CUBUK et Ian GOODFELLOW (2018). "Realistic evaluation of deep semi-supervised learning algorithms". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 2).
- PASZKE, Adam, Sam GROSS, Francisco MASSA, Adam LERER, James BRADBURY, Gregory CHANAN, Trevor KILLEEN, Zeming LIN, Natalia GIMELSHEIN, Luca ANTIGA, Alban DESMAISON, Andreas KOPF, Edward YANG, Zachary DEVITO, Martin RAISON, Alykhan TEJANI, Sasank CHILAMKURTHY, Benoit STEINER, Lu FANG, Junjie BAI et Soumith CHINTALA (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 27).
- PERRET, Benjamin (2017). *Convolution*. <https://perso.esiee.fr/~perretb/I5FM/TAI/convolution/index.html> (cf. p. 24).
- PHILLIPS, Jeff M (2016). "Coresets and sketches". In : *arXiv preprint library* (cf. p. 68).
- PINSLER, Robert, Jonathan GORDON, Eric NALISNICK et José Miguel HERNÁNDEZ-LOBATO (2019). "Bayesian batch active learning as sparse subset approxi-

- mation". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 110).
- POP, Remus et Patric FULOP (2018). "Deep ensemble bayesian active learning : Addressing the mode collapse issue in monte carlo dropout via ensembles". In : *arXiv preprint library* (cf. p. 62).
- PRUDÊNÇIO, Ricardo BC et Teresa B LUDERMIR (2008). "Selective generation of training examples in active meta-learning". In : *International Journal of Hybrid Intelligent Systems (IJHIS)* (cf. p. 110).
- RADOSAVOVIC, Ilija, Piotr DOLLÁR, Ross GIRSHICK, Georgia GKIOXARI et Kaiming HE (2018). "Data distillation : Towards omni-supervised learning". In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cf. p. 2).
- RANGANATHAN, Hiranmayi, Hemanth VENKATESWARA, Shayok CHAKRABORTY et Sethuraman PANCHANATHAN (2017). "Deep active learning for image classification". In : *IEEE International Conference on Image Processing (ICIP)* (cf. p. 9, 110).
- REN, Pengzhen, Yun XIAO, Xiaojun CHANG, Po-Yao HUANG, Zihui LI, Brij B GUPTA, Xiaojiang CHEN et Xin WANG (2021). "A survey of deep active learning". In : *ACM computing surveys (CSUR)* (cf. p. 8, 60, 62).
- ROBINSON, Rob (2017). *CNN basics*. <https://mlnotebook.github.io/post/CNN1/> (cf. p. 21).
- ROSENBLATT, Frank (1958). "The perceptron : a probabilistic model for information storage and organization in the brain." In : *Psychological review* (cf. p. 14).
- ROY, Nicholas et Andrew McCALLUM (2001). "Toward optimal active learning through monte carlo estimation of error reduction". In : *International Conference on Machine Learning (ICML)* (cf. p. 52, 58).
- RUDER, Sebastian (2016). "An overview of gradient descent optimization algorithms". In : *arXiv preprint library* (cf. p. 32).
- RUMELHART, David E, Geoffrey E HINTON et Ronald J WILLIAMS (1986). "Learning representations by back-propagating errors". In : *nature* (cf. p. 30, 31).
- RUSSAKOVSKY, Olga, Jia DENG, Hao SU, Jonathan KRAUSE, Sanjeev SATHEESH, Sean MA, Zhiheng HUANG, Andrej KARPATHY, Aditya KHOSLA, Michael BERNSTEIN et al. (2015). "Imagenet large scale visual recognition challenge". In : *International Journal of Computer Vision (IJCV)* (cf. p. 1, 118).
- SABATO, Sivan, Anand D SARWATE et Nati SREBRO (2013). "Auditing : Active learning with outcome-dependent query costs". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 6).
- SAHBI, Hichem et Sebastien DESCHAMPS (2022). "Frugal Learning of Virtual Exemplars for Label-Efficient Satellite Image Change Detection". In : *arXiv preprint library* (cf. p. ii, iii, 11, 146).

- SAHBI, Hichem, Sebastien DESCHAMPS et Andrei STOIAN (2021). "Frugal Learning for Interactive Satellite Image Change Detection". In : *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)* (cf. p. ii, iii, 10, 117, 121, 134, 146).
- SANTORO, Adam, Sergey BARTUNOV, Matthew BOTVINICK, Daan WIERSTRA et Timothy LILLICRAP (2016). "Meta-learning with memory-augmented neural networks". In : *International Conference on Machine Learning (ICML)* (cf. p. 2).
- SCHEFFER, Tobias, Christian DECOMAIN et Stefan WROBEL (2001). "Active hidden markov models for information extraction". In : *International Symposium on Intelligent Data Analysis (IDA)* (cf. p. 55).
- SCHMIDHUBER, Jürgen (2015). "Deep learning in neural networks : An overview". In : *Neural networks* (cf. p. 31).
- SCHRÖDER, Christopher et Andreas NIEKLER (2020). "A survey of active learning for text classification using deep neural networks". In : *arXiv preprint library* (cf. p. 60).
- SENER, Ozan et Silvio SAVARESE (2017). "Active learning for convolutional neural networks : A core-set approach". In : *arXiv preprint library* (cf. p. 2, 6, 8, 59, 63, 69, 70, 100-102, 121).
- SETTLES, Burr (2009). "Active learning literature survey". In : (cf. p. 7, 49, 50, 52, 55, 56, 58, 63, 110).
- SETTLES, Burr (2011). "From theories to queries : Active learning in practice". In : *International Conference on Artificial Intelligence and Statistics (AISTATS)* (cf. p. 6).
- SETTLES, Burr et Mark CRAVEN (2008). "An analysis of active learning strategies for sequence labeling tasks". In : *Conference on Empirical Methods in Natural Language Processing (EMNLP)* (cf. p. 52, 55, 56, 58).
- SETTLES, Burr, Mark CRAVEN et Lewis FRIEDLAND (2008). "Active learning with real annotation costs". In : *Proceedings of the NIPS workshop on cost-sensitive learning* (cf. p. 6).
- SETTLES, Burr, Mark CRAVEN et Soumya RAY (2007). "Multiple-instance active learning". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 52, 58).
- SEUNG, H Sebastian, Manfred OPPER et Haim SOMPOLINSKY (1992). "Query by committee". In : *Proceedings of the fifth annual workshop on Computational learning theory (COLT'92)* (cf. p. 52, 56).
- SHUI, Changjian, Fan ZHOU, Christian GAGNÉ et Boyu WANG (2020). "Deep active learning : Unified and principled method for query and training". In : *International Conference on Artificial Intelligence and Statistics (AISTATS)* (cf. p. 101).

- SIMÉONI, Oriane, Mateusz BUDNIK, Yannis AVRITHIS et Guillaume GRAVIER (2021). "Rethinking deep active learning : Using unlabeled data at model training". In : *International Conference on Pattern Recognition (ICPR)* (cf. p. 63).
- SIMONYAN, Karen et Andrew ZISSERMAN (2014). "Very deep convolutional networks for large-scale image recognition". In : *arXiv preprint library* (cf. p. 38).
- SINHA, Samarth, Sayna EBRAHIMI et Trevor DARRELL (2019). "Variational adversarial active learning". In : *IEEE International Conference on Computer Vision (ICCV)* (cf. p. 66-68).
- SRIVASTAVA, Nitish, Geoffrey HINTON, Alex KRIZHEVSKY, Ilya SUTSKEVER et Ruslan SALAKHUTDINOV (2014). "Dropout : a simple way to prevent neural networks from overfitting". In : *Journal of Machine Learning Research (JMLR)* (cf. p. 26).
- STARK, Fabian, Caner HAZIRBAS, Rudolph TRIEBEL et Daniel CREMERS (2015). "Captcha recognition with active deep learning". In : *Workshop new challenges in neural computation* (cf. p. 9).
- STOIAN, Andrei, Marin FERECATU, Jenny BENOIS-PINEAU et Michel CRUCIANU (2015). "Scalable action localization with kernel-space hashing". In : *IEEE International Conference on Image Processing (ICIP)* (cf. p. 147).
- SUBRAMANIAN, Suraj, Seth JUAREZ, Cassie BREVIU, Dmitry SOSHNIKOV et Ari BORNSTEIN (2021). *PyTorch basics*. <https://pytorch.org/tutorials/beginner/basics/intro.html> (cf. p. 27).
- SUTTON, Richard S et Andrew G BARTO (2018). *Reinforcement learning : An introduction*. MIT press (cf. p. 114).
- SZE, Vivienne, Yu-Hsin CHEN, Tien-Ju YANG et Joel S EMER (2017). "Efficient processing of deep neural networks : A tutorial and survey". In : *Proceedings of the IEEE* (cf. p. 17, 19, 21, 23).
- SZEGEDY, Christian, Sergey IOFFE, Vincent VANHOUCKE et Alexander A ALEMI (2017). "Inception-v4, inception-resnet and the impact of residual connections on learning". In : *Conference on Artificial Intelligence (AAAI)* (cf. p. 40).
- THOMPSON, Cynthia A, Mary Elaine CALIFF et Raymond J MOONEY (1999). "Active learning for natural language parsing and information extraction". In : *International Conference on Machine Learning (ICML)* (cf. p. 52).
- TONG, Simon et Daphne KOLLER (2001). "Support vector machine active learning with applications to text classification". In : *Journal of Machine Learning Research (JMLR)* (cf. p. 52, 56).
- TORCHCONTRIBUTORS (2017). *Illustrations of PyTorch transforms*. [https://pytorch.org/vision/master/auto\\_examples/plot\\_transforms.html](https://pytorch.org/vision/master/auto_examples/plot_transforms.html) (cf. p. 27).
- TRAN, Toan, Thanh-Toan DO, Ian REID et Gustavo CARNEIRO (2019). "Bayesian generative active deep learning". In : *International Conference on Machine Learning (ICML)* (cf. p. 62, 63, 66, 67).

- TRAN, Toan, Trung PHAM, Gustavo CARNEIRO, Lyle PALMER et Ian REID (2017). "A bayesian data augmentation approach for learning deep models". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 67).
- VASWANI, Ashish, Noam SHAZEER, Niki PARMAR, Jakob USZKOREIT, Llion JONES, Aidan N GOMEZ, Łukasz KAISER et Illia POLOSUKHIN (2017). "Attention is all you need". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 43).
- VIJAYANARASIMHAN, Sudheendra et Kristen GRAUMAN (2011). "Cost-sensitive active visual category learning". In : *International Journal of Computer Vision (IJCV)* (cf. p. 6).
- WANG, Dan et Yi SHANG (2014). "A new active labeling method for deep learning". In : *International Joint Conference on Neural Network (IJCNN)* (cf. p. 62).
- WANG, Keze, Liang LIN, Xiaopeng YAN, Ziliang CHEN, Dongyu ZHANG et Lei ZHANG (2018). "Cost-effective object detection : Active sample mining with switchable selection criteria". In : *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)* (cf. p. 2, 48).
- WANG, Keze, Dongyu ZHANG, Ya LI, Ruimao ZHANG et Liang LIN (2016). "Cost-effective active learning for deep image classification". In : *IEEE Transactions on Circuits and Systems for Video Technology* (cf. p. 9, 62, 63, 73).
- WANG, Ling et Hichem SAHBI (2013). "Directed acyclic graph kernels for action recognition". In : *IEEE International Conference on Computer Vision (ICCV)*, p. 3168-3175 (cf. p. 147).
- WANG, Xuezhi, Tzu-Kuo HUANG et Jeff SCHNEIDER (2014). "Active transfer learning under model shift". In : *International Conference on Machine Learning (ICML)* (cf. p. 6).
- WANG, Yaqing et Quanming YAO (2019). "Few-shot learning : A survey". In : (cf. p. 48).
- WEISS, Karl, Taghi M KHOSHGOFTAAR et DingDing WANG (2016). "A survey of transfer learning". In : *Journal of big data* (cf. p. 6).
- WIKIPEDIA (2022a). *Convolutional neural network*. <https://commons.wikimedia.org/w/index.php?curid=45679374> (cf. p. 21).
- WIKIPEDIA (2022b). *Fonction softmax (note de bas de page)*. URL : [https://en.wikipedia.org/wiki/Softmax\\_function#cite\\_note-FOOTNOTEGoodfellowBengioCourville2016](https://en.wikipedia.org/wiki/Softmax_function#cite_note-FOOTNOTEGoodfellowBengioCourville2016) (cf. p. 20).
- WIKIPEDIA (2022c). *Gradient descent*. [https://en.wikipedia.org/wiki/Gradient\\_descent](https://en.wikipedia.org/wiki/Gradient_descent) (cf. p. 33).
- WIKIPEDIA (2022d). *Noyau (de convolution)*. [https://fr.wikipedia.org/wiki/Noyau\\_\(traitement\\_d%27image\)](https://fr.wikipedia.org/wiki/Noyau_(traitement_d%27image)) (cf. p. 22).
- WIKIPEDIA (2022e). *Receptive Field*. [https://en.wikipedia.org/wiki/Receptive\\_field](https://en.wikipedia.org/wiki/Receptive_field) (cf. p. 24).



- WU, Jianxin (2017). "Introduction to convolutional neural networks". In : *National Key Lab for Novel Software Technology. Nanjing University. China* (cf. p. 21).
- WU, Yong Cheng (2013). "Active learning based on diversity maximization". In : *Applied Mechanics and Materials* (cf. p. 71).
- XIA, Gui-Song, Xiang BAI, Jian DING, Zhen ZHU, Serge BELONGIE, Jiebo LUO, Mihai DATCU, Marcello PELILLO et Liangpei ZHANG (2018). "DOTA : A large-scale dataset for object detection in aerial images". In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cf. p. 9, 27, 77, 110, 117, 132).
- XU, Zuobing, Ram AKELLA et Yi ZHANG (2007). "Incorporating diversity and density in active learning for relevance feedback". In : *European Conference on Information Retrieval (ECIR)* (cf. p. 59).
- YAN, Songbai, Kamalika CHAUDHURI et Tara JAVIDI (2016). "Active learning from imperfect labelers". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 5).
- YANG, Yi, Zhigang MA, Feiping NIE, Xiaojun CHANG et Alexander G HAUPTMANN (2015). "Multi-class active learning by uncertainty sampling with diversity maximization". In : *International Journal of Computer Vision (IJCV)* (cf. p. 8).
- YI, Dong, Zhen LEI, Shengcai LIAO et Stan Z LI (2014). "Deep metric learning for person re-identification". In : *International Conference on Pattern Recognition (ICPR)* (cf. p. 1).
- YOO, Donggeun et In So KWEON (2019). "Learning loss for active learning". In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cf. p. 64, 68, 73).
- YOSINSKI, Jason, Jeff CLUNE, Yoshua BENGIO et Hod LIPSON (2014). "How transferable are features in deep neural networks?" In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 2).
- ZHANG, Cha et Tsuhan CHEN (2002). "An active learning framework for content-based information retrieval". In : *IEEE Transactions on Multimedia* (cf. p. 52).
- ZHANG, Chicheng et Kamalika CHAUDHURI (2015). "Active learning from weak and strong labelers". In : *Advances in Neural Information Processing Systems (NeurIPS)* (cf. p. 5).
- ZHANG, Min, Linpeng LI, Hai WANG, Yan LIU, Hongbo QIN et Wei ZHAO (2019). "Optimized compression for implementing convolutional neural networks on FPGA". In : *Electronics* (cf. p. 38).
- ZHANG, Tong et F OLES (2000). "The value of unlabeled data for classification problems". In : *International Conference on Machine Learning (ICML)* (cf. p. 58).
- ZHANG, Xin, Yongcheng WANG, Ning ZHANG, Dongdong XU et Bo CHEN (2019). "Research on scene classification method of high-resolution remote sensing images based on RFPNet". In : *Applied Sciences* (cf. p. 26).

- ZHDANOV, Fedor (2019). "Diverse mini-batch active learning". In : *arXiv preprint library* (cf. p. 63).
- ZHU, Jia-Jie et José BENTO (2017). "Generative adversarial active learning". In : *arXiv preprint library* (cf. p. 50, 59, 64-67).
- ZHU, Xiaojin, John LAFFERTY et Zoubin GHAHRAMANI (2003). "Combining active learning and semi-supervised learning using gaussian fields and harmonic functions". In : *International Conference on Machine Learning Workshop (ICML-W)* (cf. p. 52, 58).
- ŽLIOBAITĖ, Indrė, Albert BIFET, Bernhard PFAHRINGER et Geoffrey HOLMES (2013). "Active learning with drifting streaming data". In : *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)* (cf. p. 51).