THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*

Présentée et soutenue le *17/11/2023* **par :**

Mouna Sabrine MAYOUF

Intégration de connaissances de haut-niveau dans un système d'apprentissage par réseau de neurones pour la classification d'images

---

JURY

Adrian BASARAB	Professeur d'Université	Président du Jury
Béatrice DUVAL	Professeure émérite	Membre du Jury
Florence DUPIN DE SAINT-CYR	Maître de conférences	Membre du Jury
Nadjib LAZAAR	Maître de conférences	Membre du Jury
Didier Dubois	Professeur émérite	Membre invité
Insaf SETITERA	Docteur	Membre invité

École doctorale et spécialité :
EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse

Unité de Recherche :
IRIT : Institut de Recherche en Informatique de Toulouse

Directeur(s) de Thèse :
Florence DUPIN DE SAINT-CYR BANNAY

Rapporteurs :
Adrian BASARAB et *Béatrice DUVAL*

Contents

Remerciements **8**

Introduction (en français) **9**

Introduction (In English) **14**

I Preamble 18

1 Image classification in NNs: background and formalization **21**
1.1 Basics about Neural Networks (NNs) 24
1.1.1 The standard architecture of a NN 24
1.1.2 Data preparation 27
1.1.2.1 Data splitting 27
1.1.2.2 Data balancing and augmentation 28
1.1.2.3 Data normalization 29
1.1.3 The learning process 30
1.1.4 Evaluation metrics 31
1.1.5 Regularization techniques 32
1.1.6 Main standard architectures 33
1.2 Formalizing the probabilistic view of a NN 35

2 Background on High-level knowledge in NNs **39**
2.1 Expert knowledge 40
2.1.1 Integrating High-level knowledge on data preparation 41
2.1.2 Integrating high-level knowledge on the model architecture and loss function 41
2.2 Black-box "knowledge" 43
2.3 Layerwise Relevance Propagation (LRP) approach 44

3 BreakHis and the other datasets used in this study **47**
3.1 BreakHis dataset 48
3.1.1 About breast cancer 48
3.1.2 BreakHis composition 50
3.1.3 A traditional approach for BreakHis images classification 51
3.1.4 CNN-based approaches for BreakHis images classification 52
3.2 Other datasets 53

3.2.1 Kaggle Fashion Product Images dataset . . . 54
3.2.2 Fshion-MNIST . . . 54
3.2.3 Prime-MNIST . . . 55
3.2.4 HZoo . . . 55

II High-level knowledge used for feeding the network 59

4 How does data preparation impact the classification quality 62
4.1 Background about information metrics . . . 64
4.2 General principles for efficient data preparation . . . 65
4.3 Data preparation Formalization and Protocols for BreakHis data preparation . . . 66
4.3.1 BreakHis transformation operators . . . 66
4.3.2 Transformation signatures for BreakHis . . . 67
4.4 New metrics for measuring the dataset diversity . . . 68
4.5 Experimental Protocols . . . 70
4.6 Results and discussion . . . 71

5 How does data feeding impact the classification quality? 76
5.1 State of the art about curriculum learning . . . 77
5.2 BreakHis data preparation . . . 79
5.2.1 BreakHis data-subsets . . . 79
5.2.2 BreakHis data-preparation algorithm . . . 80
5.3 Experimental protocols on curriculum efficiency for classification . . . 82
5.3.1 CNN architecture and computational details . . . 82
5.3.2 Experimental protocols . . . 82
5.4 Results and discussion . . . 83
5.4.1 Accuracy results . . . 83
5.4.2 Duration of the training period . . . 85
5.4.3 Number of epochs for training stabilization: . . . 86

III Hierarchical knowledge used for designing the learning engine 89

6 Formalising Hierarchical classification 92
6.1 State of the art about hierarchical classification . . . 94
6.1.1 Hierarchical adjustment . . . 94
6.1.2 Branch-based CNN approaches . . . 95
6.1.3 An example of local approach . . . 96
6.1.4 Loss functions integrating hierarchical constraints . . . 96
6.2 Principles and notations about hierarchical dataset labeling . . . 97
6.3 The probabilistic view of HMC classification . . . 99
6.4 The two constraints of a hierarchical classifier . . . 100

7 An architecture compliant with ILE and ILH constraints 102
7.1 General architecture for GHC-CNN . . . 103
7.1.1 The hidden layers (HL) . . . 103
7.1.2 The penultimate layer of primary outputs (PNPO) . . . 104
7.1.3 The final adjusted finer output layer (FAFO) . . . 104
7.2 Hierarchical loss functions . . . 107
7.2.1 Level weighted loss function . . . 107
7.2.2 Hierarchy violation loss function . . . 108
7.3 Compliance with ILE and ILH constraints . . . 108
7.4 Experiments . . . 109
7.4.1 Data preparation and computational details . . . 109
7.4.2 Training strategy . . . 110
7.4.3 Results and discussion . . . 112

8 Abstention mechanism for a robuster hierarchical classification 116
8.1 State of the art about abstention in machine learning . . . 118
8.2 Introducing an abstention layer in the GHC-CNN . . . 121
8.2.1 Towards rational thresholds . . . 121
8.2.2 A non-drastic, level increasing and cascading threshold . . . 123
8.2.3 Hierarchically uncertainty alignment . . . 125
8.2.4 New metrics for compliance with ILA constraint . . . 126
8.3 Experimental protocols . . . 127

Conclusion (en français) 131

Conclusion (in English) 134

List of Figures

1 Summary diagram of the thesis context 12

1.1 Biological neuron in comparison to an artificial neural network: (a) human neuron; (b) artificial neuron; (c) biological synapse; and (d) ANN synapses [71]. 23
1.2 Example of the convolutional operation. 26
1.3 VGG19 architecture [132], `Conv i_j` are convolutional layers with kernels of size 3×3, `FC1` and `FC2` are fully connected layers. 35

2.1 Some mislabeling examples from Fashion-MNIST and CIFAR-100 datasets from [78]. 41
2.2 The multi-task learning workflow for both segmentation and fine-grained classification used in [130]. 42
2.3 The generic teacher-student framework for knowledge distillation [40]. 44
2.4 An example on the LRP process [9] 45

3.1 Images from BreakHis DataSet with different magnifications. 48
3.2 The evolution of breast cancer mortality and incidence rates among women in France between 1975 and 2020 [2] 49
3.3 Breast cancer diagnosis workflow 49
3.4 Examples of tumors subtypes from BreakHis. 51
3.5 Some images from Kaggle Fashion Product Images dataset [3]. 54
3.6 Some images from MNIST dataset [28]. 55
3.7 Some images from Fashion-MNIST dataset [126]. 55
3.8 The hierarchy structure of Fashion-MNIST dataset proposed by [98]. 56
3.9 The hierarchy structure of Prime-MNIST dataset. 56
3.10 examples of animal images chosen from Imagenet dataset for HZoo. 57

5.1 The Ctrl_INCR approach 83
5.2 The INCR approach 83
5.3 An Image from BreakHis (left), its associated saliency maps obtained with CTRL_INCR (center) and INCR (right) 85

6.1 The B-CNN architecture described in [136] 95
6.2 The hierarchical classifier of [79]. 96
6.3 Hierarchy structure of labels (0 being the fictive root, it is the first node of each path to any node in the hierarchy), for example, $ch(c_1^1) = \{1, 2, \ldots, k\}$ and $pa(c_l^2) = c_{\mathcal{N}_1}^1$. (A fictive node c_e^3 was added as a virtual child of the node c_k^2 to transform this latter into an internal node) 98

7.1 The GHC-CNN general architecture. The nodes ⨁ represent the Bayesian adjustment. 104
7.2 Training loss and training accuracy curves for GHC-CNN with a raw training strategy. 110
7.3 Training loss and training accuracy curves for GHC-CNN with a raw training strategy. 111
7.4 Training loss and training accuracy curves for GHC-CNN with a raw training strategy. 111
7.5 Training loss curves of GHC-CNN with/without Bayesian adjustment ((blue/green) for the BreakHis dataset 114

8.1 Illustration of a binary classification scenario where the two classes overlap in a region [49] 119
8.2 Abstention mechanism approaches and techniques inspired by [48] 120

Résumé du manuscrit :
Intégration de connaissances de haut-niveau dans un système d'apprentissage par réseau neuronal pour la classification d'images

Les réseaux neuronaux ont permis de réaliser des avancées dans des tâches réputées difficiles pour des ordinateurs (voire même pour des humains), comme la classification automatique à partir d'images ou le traitement du langage naturel. Toutefois, leur nature de boîte noire limite leur explicabilité et entrave leur capacité à exploiter des connaissances extérieures aux données. Cette thèse a pour but d'explorer et de proposer des techniques d'intégration des connaissances dans les réseaux neuronaux afin d'améliorer leurs performances, leurs interprétabilités et leur adaptabilités.

La première partie de la thèse est centrée sur l'intégration de connaissances aux données d'entrée d'un réseau. Son premier chapitre s'adresse à la préparation des données. On y propose une formalisation du prétraitement afin de permettre la transparence et la reproductibilité de cette étape. Cette formalisation nous permet d'étudier l'impact de la data augmentation : pour caractériser ce qu'est une bonne préparation des données, et l'état informatif d'un dataset, un ensemble de mesures et de principes est proposé, ensuite des protocoles expérimentaux sont conçus afin d'évaluer ces principes sur le dataset BreakHis. La technique "layer-wise relevance propagation" a permis de visualiser l'impact de la data augmentation.

Le deuxième chapitre de cette partie s'adresse à l'exploitation de connaissances haut-niveau pour l'établissement d'un ordre de présentation des données au réseau. Nous introduisons l'apprentissage par curriculum incrémental sur l'ordre de passage des données en entrée. Les résultats obtenus améliorent l'exactitude et la vitesse de convergence. Bien que cette étude soit menée sur le dataset BreakHis, nous pensons qu'elle est généralisable à des datasets contenant d'autres données que des images et d'autre connaissances que la résolution.

La deuxième partie est centrée sur l'intégration de connaissances au sein de l'archi-tecture d'un réseau et au niveau de sa sortie. Dans ce cadre, nous nous sommes intéressés à la classification multi-label hiérarchique, pour laquelle nous avons formalisé les connaissances représentant le lien hiérarchique. Pour cela nous avons introduit deux contraintes représentant pour l'une le fait qu'un objet ne peut être affecté qu'à une seule classe à un niveau donné de la hiérarchie, et l'autre imposant que l'affectation globale d'un objet respecte la hiérarchie de classe (par exemple, on interdit d'avoir un élément qui soit classé abeille et mammifère car il faudrait plutôt le classer comme abeille et insecte).

Nous avons conçu une architecture et une fonction de perte qui imposent ces contraintes hiérarchiques durant l'apprentissage. L'architecture se distingue des travaux de l'état de l'art par le fait qu'un seul réseau est utilisé pour prédire simultanément les labels des différents niveaux : toutes les couches sont responsables de la prédiction du n-uplet des classes. Plusieurs variantes du réseau ont été expérimentées sur cinq jeux de données (BreakHis, PrimeMNIST, Fashion-MNIST, DeepFashion, HZoo) et les résultats confirment l'efficacité de la prise en compte des contraintes hiérarchiques soutenant ainsi l'importance de la prise en compte de connaissances externes.

Afin de raffiner les résultats de cette classification hiérarchique, nous avons introduit un mécanisme d'abstention grâce auquel le réseau donne une prédiction au niveau de spécificité le plus précis possible sur lequel sa confiance est suffisante, et s'abstient sinon. Nous avons défini différents seuils de confiance et proposé différentes contraintes sur les seuils relatives à la hiérarchie des classes. Pour évaluer ce mécanisme, de nouvelles métriques de classification prenant en compte l'abstention (le taux de classifications correctes ignorées et le taux de classifications incorrectes évitées) ont

été définies. Nous avons mené des expérimentations sur les mêmes jeux de données et les résultats ont montré l'intérêt de l'abstention, et la nécessité de définir un seuil expérimental adapté à chaque dataset.

Pour conclure, les travaux de cette thèse soulignent l'intérêt d'exploiter des connaissances externes dans le domaine des réseaux de neurones ceci au niveau des trois composantes d'un système d'apprentissage profond : en entrée pendant la préparation des données, dans la structure du réseau (architecture et fonction de perte) et au niveau de la sortie lors de la prise de décision de classification.

Remerciements

Je tiens à exprimer ma profonde gratitude à tous ceux qui m'ont aidée, de près ou de loin, dans l'accomplissement de cette thèse.
Un merci tout particulier à mon encadrante, Madame Florence Dupin de Saint-Cyr, sans qui ce travail de recherche n'aurait pas été possible. Ses encouragements, sa patience, la disponibilité qu'elle a toujours montrée, ainsi que sa curiosité scientifique ont été pour moi une source d'inspiration inestimable. Ses orientations et suggestions ont grandement contribué à la qualité de ce travail.
Enfin, je souhaite remercier du fond du cœur ma famille, pour leur foi indéfectible en moi et leur soutien sans faille tout au long de ce parcours.

I would like to express my deep gratitude to everyone who has assisted me, both directly and indirectly, in the completion of this thesis.
A special thank you to my supervisor, Madame Florence Dupin de Saint-Cyr, without whom this research work would not have been possible. Her encouragement, patience, and the availability she always showed, as well as her scientific curiosity, have been an invaluable source of inspiration for me. Her insightful guidance and advice have greatly contributed to the quality of this work.
Finally, I wish to heartily thank my family for their unwavering faith in me and their relentless support throughout this journey.

Introduction (en français)

Pour introduire nos travaux qui traitent à la fois d'apprentissage automatique sur des données et de raisonnement sur des connaissances, nous nous basons sur la distinction classique établie par Kahneman et Tversky dans le domaine de la psychologie. La théorie de Kahneman et Tversky [52] a profondément influencé la compréhension des processus de raisonnement et de prise de décision humaine. Leur travail émet l'hypothèse que les processus décisionnels humains sont régis par deux système complémentaires : le système 1 (S1) et le système 2 (S2). Le système 1 est un système intuitif, automatique et rapide qui fonctionne par association, inconsciemment et sans effort. Il s'appuie sur les raccourcis mentaux pour évaluer rapidement les situations et porter des jugements. Tandis que le système 2 est un système délibéré, conscient et lent qui nécessite des efforts et des ressources mentales pour fonctionner. Il se caractérise par une pensée réfléchie et analytique, impliquant un raisonnement logique et des processus cognitifs plus profonds. S2 permet de prendre des décisions plus complexes, de résoudre des problèmes et de mettre en œuvre une pensée critique.

Dans le cadre des recherches en intelligence artificielle, l'objectif est entre autres de simuler le raisonnement et la prise de décision humaine au moyen d'outils informatiques. On peut diviser les sous-domaines de l'intelligence artificielle en deux courants en accord avec la théorie de Kahneman et Tversky, Le premier courant couvre la partie analytique et procédurale basée sur l'explicitation des raisonnements et des connaissances (ce courant est connu sous le nom de Représentation des Connaissances et du Raisonnement ou IA fondamentale (en anglais *Knowledge Representation and Reasoning KRR*). Le deuxième courant consiste à simuler les capacités d'évaluation rapide des situations grâce à la mise en place de mécanismes d'association : c'est le courant de l'apprentissage automatique (en anglais *Machine Learning ML*).

L'intelligence artificielle fondamentale s'appuie sur la logique et la représentation des connaissances pour mieux comprendre et formaliser comment les systèmes intelligents peuvent raisonner et prendre des décisions. Le domaine KRR est donc un domaine assez vaste. Parmi ses axes principaux, nous citons, dans le cadre de cette thèse, trois axes majeurs : le premier axe s'intéresse à l'encodage explicite des connaissances, plusieurs formalismes sont utilisés, entre autres, la logique classique et ses extensions et les représentations graphiques (comme les ontologies, les graphes d'argumentation, les réseaux bayésiens, etc.). Le deuxième axe s'articule autour de l'étude algorithmique des systèmes de résolution de problèmes, on y trouve les problèmes de satisfaction de contraintes (en anglais *Constraint Satisfaction Problem CSP*), les problèmes de satisfaisabilité booléenne SAT, qui, étant donné une formule en logique propositionnelle, déterminent s'il existe une valuation des variables qui la satisfasse. Aussi, les problèmes de planification qui s'intéressent à la génération de plans d'action permettant de réaliser un but depuis un état initial. Un troisième axe important du domaine KRR est l'aide à la décision, où l'on dispose de préférences (ou utilités) en plus des connaissances incertaines sur l'évolution du monde, on peut citer par exemple, l'aide à

la décision multi-critères. Notons que les connaissances au cœur du domaine KRR représentant la pièce maîtresse de ce premier courant, peuvent être imparfaites. Ces imperfections peuvent surgir de la source de connaissance comme de sa représentation. Parmi ces imperfections, on peut citer l'incohérence qui dénote des connaissances contradictoires. L'incertitude qui dénote un doute sur la validité des connaissances (dû par exemple à une source incomplète et/ou non fiable). L'imprécision qui correspond à une difficulté dans l'énoncé des connaissances (connaissances incomplètement ou vaguement spécifiées). Plusieurs approches ont été développées afin de pallier à ces imperfections, telles la logique floue, et le raisonnement sous incertitude.

L'apprentissage automatique, quant à lui, est le deuxième courant de l'IA dans lequel on s'intéresse à faire apprendre à la machine la réalisation d'une tâche donnée, soit à partir de bases d'exemples (apprentissage supervisé, non-supervisé et semi-supervisé), soit en fournissant des récompenses ou des pénalités selon les réponses (apprentissage par renforcement).

L'apprentissage par renforcement, est utilisé pour apprendre des actions séquentielles dans un environnement afin de maximiser une récompense cumulative. Le modèle prend des actions, observe les récompenses associées et ajuste ses décisions pour atteindre un objectif spécifique. Les tâches courantes incluent les jeux, où l'agent apprend à jouer pour maximiser le score, la robotique, où l'agent apprend à effectuer des tâches physiques, et la gestion de systèmes complexes, où l'agent prend des décisions pour optimiser les performances du système. Le Q-Learning est l'un des algorithmes d'apprentissage par renforcement les plus répandus, où l'agent maintient une fonction Q qui est utilisée pour estimer la récompense future attendue en prenant une certaine action dans un état donné.

Dans l'apprentissage non supervisé, le modèle traite des données non étiquetées et cherche à découvrir des structures ou des schémas intrinsèques dans les données. Les tâches courantes incluent le regroupement (ou en anglais *clustering*), où le modèle identifie des groupes similaires de données, la réduction de dimensionnalité, où le modèle réduit la complexité des données tout en préservant les caractéristiques importantes, ou encore la détection d'anomalies, où le modèle identifie des exemples inattendus ou aberrants. Parmi les algorithmes couramment utilisés en apprentissage non supervisé, on peut citer le k-moyennes (en anglais k-means), où les données sont divisées en k groupes distincts, l'objectif du modèle étant de minimiser la somme des distances entre chaque point représentant une donnée et le centre de son groupe attribué. Aussi, nous citons l'analyse en composantes principales (PCA) qui est utilisée pour la réduction de dimensionnalité où les données sont projetées dans un nouvel espace de dimensions réduites tout en préservant la variance maximale. la PCA est couramment utilisée pour la visualisation de données et pour l'amélioration les performances des modèles en éliminant les caractéristiques redondantes ou peu informatives ou encore les individus perturbateurs.

Dans cette thèse, nous centrons notre intérêt sur l'apprentissage supervisé, plus de détails sur l'apprentissage automatique non supervisé peuvent être consultés dans [135]. Dans l'apprentissage supervisé, le modèle est entraîné sur un ensemble de données étiquetées, où les exemples sont associés à des vérités-terrain (étiquettes ou sorties réelles). Les tâches courantes incluent la classification, où le modèle prédit des catégories discrètes (par exemple, spam/non-spam, chien/chat/lapin/loup), et la régression, où le modèle prédit des valeurs continues (par exemple, prédiction du prix des maisons en fonction de leurs caractéristiques). Le mécanisme de ce type d'apprentissage s'articule autours de l'optimisation de l'écart (souvent calculé grâce à une fonction de perte) entre la sortie prédite et la sortie réelle. Parmi les algorithmes répandus pour ce type d'apprentissage, nous citons les Machines à Vecteurs de Support (ou *Support Vector Machines* en anglais SVMs). Les SVMs

cherchent à trouver un hyperplan qui sépare les données dans un espace de grande dimension de manière optimale, maximisant la marge entre les classes. Les SVMs sont plus efficaces pour les problèmes de classification binaire, mais elles peuvent être étendues à des problèmes multi-classes. Les arbres de décision (ou *Decision trees* en anglais), sont un autre type d'algorithmes pour ce type d'apprentissage, utilisés pour les tâches de classification et de régression. Ils construisent une structure arborescente où chaque nœud représente une décision basée sur une caractéristique particulière des données. Bien qu'ils soient faciles à interpréter et à visualiser, ils peuvent être sensibles aux variations et biais. Une autre branche de l'apprentissage supervisé concerne les Forêts aléatoires (ou en anglais *Random Forests*), qui sont une extension des arbres de décision. Elles construisent un grand nombre d'arbres de décision indépendants et combinent leurs prédictions pour obtenir une meilleure performance globale. Les forêts aléatoires sont plus robustes que les arbres de décision individuels et ont tendance à produire de meilleurs résultats en termes de précision.

Dans cette thèse, nous mettons la lumière sur l'un des algorithmes les plus utilisés récemment, qui est l'apprentissage par réseaux de neurones artificiels (ou en anglais *Artificial Neural Networks ANNs, abrégés en NeuralNetworks NNs)*. Grâce aux progrès matériels et algorithmiques, ces modèles ont connu des avancées significatives. Les réseaux neuronaux artificiels sont composés d'unités reliées appelées "neurones artificiels". Ces neurones reçoivent des signaux d'entrée, les traitent et les transmettent à d'autres neurones connectés. Leur architecture est composée de couches interconnectées de neurones artificiels, où chaque neurone est une unité de traitement qui reçoit des entrées, effectue des calculs sur ces entrées, puis produit une sortie (le Chapitre 1 traite en détails du fonctionnement des NNs).

Les réseaux neuronaux convolutifs (en anglais *Convolutional Neural Networks CNNs*) sont un type de réseaux neuronaux artificiels spécialement conçus par bio-mimétisme à partir de la vision humaine pour analyser des données visuelles, telles que des images. Bien que les CNNs aient fait preuve d'avancées remarquables, ces réseaux restent considérés comme des “boîtes noires” en raison du manque de leur d'interprétabilité et de leur dépendance exclusive aux données pour prendre des décisions. L'explicabilité et l'interpretabilité des CNNs est un domaine émergent qui suscite de plus en plus l'intérêt de la communauté scientifique. Certaines approches sont basées sur la logique classique afin de générer des formules logiques d'explication d'une décision, où la participation d'une caractéristique est attestée si elle satisfait la formule. D'autres approches sont proposées pour tenter de visualiser et d'interpréter les critères sur lesquels s'est basé le réseau pour prendre sa décision. Certaines techniques visuelles comme la LRP [9], mettent la lumière sur les zones saillantes de l'image qui ont conduit à la prédiction finale. Ces cartes de caractéristiques peuvent être considérées comme mettant en avant une certaine génération de “connaissance” du fonctionnement du réseau. Cependant, le domaine est encore bourgeonnant, et aucune méthode ne permet une explicitation tangible et intelligible de façon complète. Les CNNs sont également sujets à plusieurs imperfections, à commencer par les données d'entrées qui constituent la matière première des CNNs, mais souffrent souvent de plusieurs problèmes tels que le manque de données étiquetées, les problèmes d'étiquetage, le bruit et les biais, etc. De plus, plusieurs paramètres peuvent affecter la qualité d'apprentissage, telles que la conception du réseau et les contraintes matérielles. Les biais des données et la qualité de l'apprentissage conditionnent donc considérablement les prédictions de classification. En outre, les CNNs standards ne prennent malheureusement pas compte des connaissances externes existantes sur les données labelisées, comme par exemple les liens hiérarchiques entre les étiquettes, ou les connaissances expertes sur des données.

Inspirée de ces motivations, cette thèse propose une approche qui vise à intégrer à un CNN

(considéré comme un genre de système S1) des mécanismes de raisonnements sur les connaissances (considéré comme un système S2). Nous explorons la possibilité d'intégrer ses connaissances à différents niveaux : en entrée en prenant en compte les connaissances externes, au niveau de la conception de l'archi-tecture et le paramétrage de la fonction de perte et au niveau de la sortie en raffinant les résultats des prédictions.

La Figure 1 récapitule le contexte de cette thèse et situe les chapitres la composant.

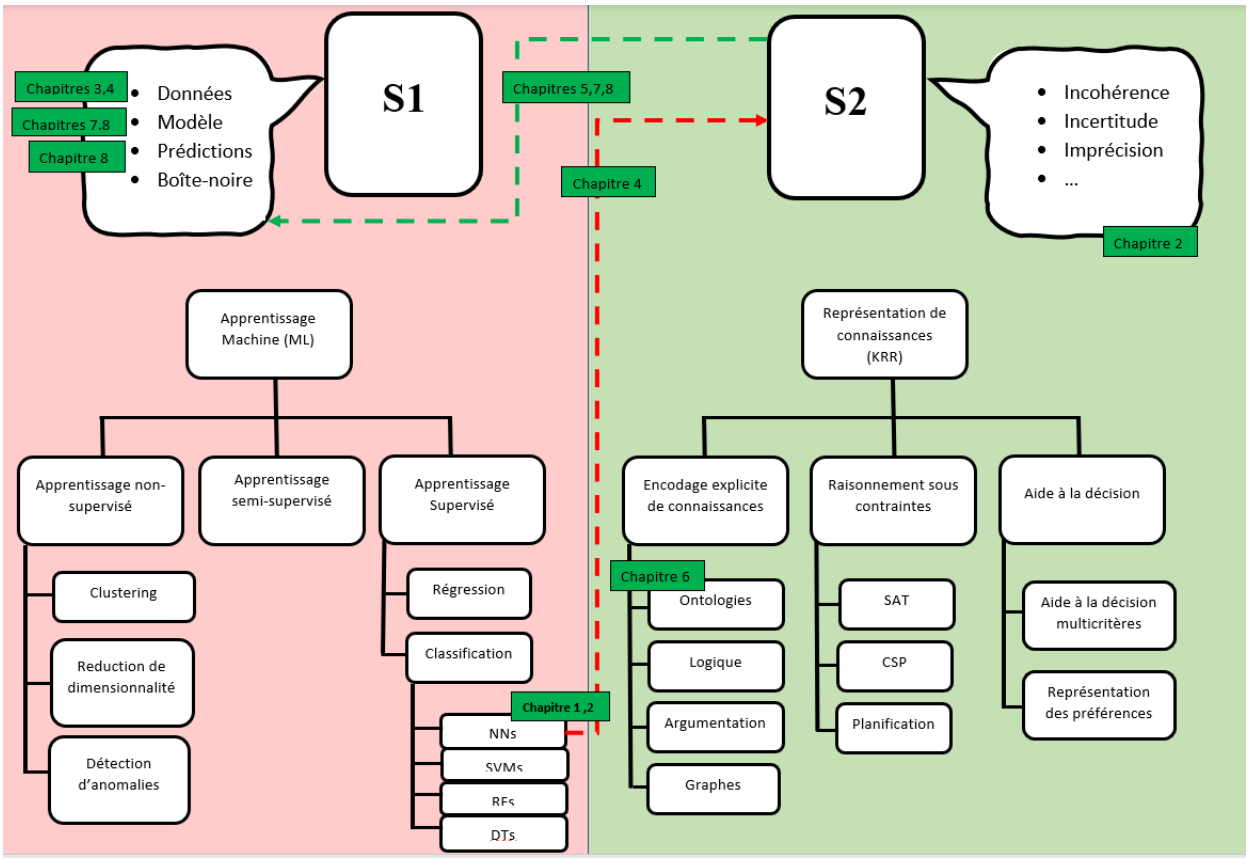


FIGURE 1 – Summary diagram of the thesis context

Après une partie préliminaire rappelant les principes fondamentaux à l'œuvre dans les réseaux de neurones, introduisant les travaux faisant intervenir des connaissances dans ce type de réseaux et expliquant les ensembles de données sur lesquels les expéritmentations ont étés menées, cette thèse est constituée de deux parties fondamentales. La première partie met l'accent sur l'intégration de connaissances dans les données d'entrée du réseau. Son premier chapitre se focalise sur la préparation des données, une étape clé, mais peu explicitée dans la majorité des travaux scientifiques actuels. Pour cela, nous mettons en place une formalisation du prétraitement des données afin d'assurer la transparence et fournir les détails nécessaires pour pouvoir reproduire les expérimentations. Nous nous sommes également intéressé à l'augmentation des données (ou en anglais *data augmentation*, une technique largement adoptée, mais en raison de l'opacité des CNNs, l'explication de son apport fructueux reste toujours méconnue. Pour cela, nous avons proposé une approche basée sur la méthode de la "Layerwise Relevance Propagation (LRP)", qui est une méthode de visualisation des pixels saillants, pour exhiber la différence entre les pixels saillants d'une image originale, et

de sa transformation par data augmentation, après passage dans le réseau. Nous présentons également un ensemble de mesures et de principes, suivis de protocoles expérimentaux élaborés pour évaluer l'informativité du dataset après préparation. Le deuxième chapitre de cette première partie se penche sur l'exploitation de connaissances de haut-niveau pour déterminer l'ordre de présentation des données au réseau. Pour cela, nous introduisons l'apprentissage par curriculum incrémental pour ordonnancer l'introduction des données d'entraînement. Les résultats obtenus sur le dataset BreakHis permettant d'augmenter l'exactitude et la vitesse de convergence, et encouragent donc la considération de cette approche, qui reste généralisable pour tout autre dataset.

La dernière partie de cette thèse se concentre sur l'intégration de connaissances externes dans la conception du modèle. Comme cas particulier, nous nous sommes situé dans la cadre de la classification multi-label hiérarchique, où la connaissance externe que nous avons pris en compte est représentée par le lien hiérarchique reliant les étiquettes. Dans ce contexte, nous formalisons les connaissances représentant les liens hiérarchiques en introduisant deux contraintes essentielles. La première contrainte impose qu'un objet doit être affecté à une classe unique par niveau hiérarchique et la deuxième contrainte impose que l'affectation globale respecte le lien hiérarchique reliant les classes, afin d'éviter l'occurrence de violations hiérarchiques. Pour cela, un réseau de neurones spécial a été conçu, avec une fonction de perte sur mesure pour la prise en compte des contraintes. Cinq jeux de données ont été sélectionnés pour les expérimentations et les résultats obtenus montrent l'impact positif de la considération des connaissances externes.

Pour améliorer la robustesse de ce réseau hiérarchique, nous introduisons ensuite un mécanisme d'abstention, permettant au réseau de s'abstenir si la confiance de prédiction est au dessous d'un seuil de confiance prédéfini. Afin d'évaluer ce mécanisme, nous introduisons de nouvelles métriques de classification prenant en compte l'abstention, telles que le taux de classifications correctes ignorées et le taux de classifications incorrectes évitées. Ce mécanisme est testé sur les mêmes cinq jeux de données.

Cette thèse est un premier travail exploratoire pour étudier l'intérêt de l'intégration des connaissances haut-niveau dans les CNNs, ce document décrit cette étude et les avancées produites et ouvre l'horizon à d'autres pistes citées en perspective.

Introduction (In english)

To introduce our work that deals with both machine learning on data and reasoning on knowledge, we base ourselves on the classical distinction established by Kahneman and Tversky in the field of psychology. The theory of Kahneman and Tversky [52] has deeply influenced the understanding of human reasoning and decision-making processes. Their work postulates that human decision-making processes are governed by two complementary systems: System 1 (S1) and System 2 (S2).

On one hand, S1 is an intuitive, automatic and fast system that works by association, unconsciously and effortlessly. It relies on mental shortcuts to quickly assess situations and make judgments. On the other hand, S2 is a deliberate, conscious and slow system that requires effort and mental resources to operate. It is characterized by reflective and analytical thinking, involving logical reasoning and deeper cognitive processes. S2 enables more complex decision-making, problem-solving and critical thinking.

In the context of research in artificial intelligence, one of the objectives is to simulate human reasoning and decision-making using computer tools. We can divide the fields of artificial intelligence into two streams in line with Kahneman and Tversky's theory. The first stream covers the analytical and procedural part based on the explicitation of the representation of reasoning and knowledge (known as Knowledge Representation and Reasoning (KRR) or Fundamental AI). The second stream is the Machine learning (ML) stream, which consists of simulating the ability to assess situations quickly, thanks to the implementation of association mechanisms. Fundamental artificial intelligence relies on logic and knowledge representation to better understand and formalize how intelligent systems can reason and make decisions. The KRR field is therefore quite vast. In the context of this thesis, we have identified three main lines of research: the first focuses on the explicit encoding of knowledge, using a variety of formalisms, including classical logic and its extensions, and graphical representations (such as ontologies, argumentation graphs, Bayesian networks, etc.). The second axis focuses on the algorithmic study of problem-solving systems, including constraint satisfaction problems (CSP) and Boolean satisfiability problems (SAT), which, given a propositional logic formula, determine whether there is a variable valuation that satisfies it. Also, planning problems, which are concerned with generating action plans to achieve a goal from an initial state. A third major focus of the KRR domain is decision support, where preferences (or utilities) are available in addition to uncertain knowledge about the evolution of the world, e.g. multi-criteria decision support. It's worth noting that the knowledge at the kernel of the KRR domain, representing the centerpiece of this first stream, may be imperfect. These imperfections can arise both from the knowledge source and from its representation. Among these imperfections are: incoherence, which denotes contradictory knowledge. Uncertainty, which denotes doubt about the validity of knowledge (due, for example, to an incomplete and/or unreliable source). Imprecision, which corresponds to a

difficulty in stating knowledge (incompletely or vaguely specified knowledge). Several approaches have been developed to overcome these imperfections, such as fuzzy logic and reasoning under uncertainty.

Machine learning, for its part, is the second stream of AI in which the focus is on teaching the machine to perform a given task, either from a base of examples (supervised, unsupervised and semi-supervised learning), or by providing rewards or penalties according to responses (reinforcement learning).

Reinforcement learning is used to learn sequential actions in an environment in order to maximize a cumulative reward. The model takes actions, observes the associated rewards and adjusts its decisions to achieve a specific goal. Common tasks include games, where the agent learns to play to maximize score, robotics, where the agent learns to perform physical tasks, and the management of complex systems, where the agent makes decisions to optimize system performance. Q-Learning is one of the most widespread reinforcement learning algorithms, where the agent maintains a Q-function which is used to estimate the expected future reward of taking a certain action in a given state.

In unsupervised learning, the model processes unlabeled data and seeks to discover intrinsic structures or patterns in the data. Common tasks include clustering, where the model identifies similar groups of data, dimensionality reduction, where the model reduces the complexity of the data while preserving important features, or anomaly detection, where the model identifies unexpected or outlier examples. Algorithms commonly used in unsupervised learning include k-means, where data are divided into k distinct groups, with the model's objective being to minimize the sum of the distances between each point representing a datum and the center of its assigned group. Principal Component Analysis (PCA) is also used for dimensionality reduction, where data are projected into a new space of reduced dimensions while preserving maximum variance. PCA is commonly used for data visualization and for improving model performance by eliminating redundant or uninformative features or interfering individuals.

In this thesis, we focus our interest on supervised learning, more details on unsupervised machine learning can be found in [135]. In supervised learning, the model is trained on a labeled dataset, where samples are associated with ground-truths (labels or actual outputs). Common tasks include classification, where the model predicts discrete categories (e.g. spam/non-spam, dog/cat/rabbit/wolf), and regression, where the model predicts continuous values (e.g. predicting house prices based on their characteristics). The mechanism of this type of learning revolves around optimizing the deviation (often calculated using a loss function) between predicted and real output. Among the most common algorithms for this type of learning are Support Vector Machines (SVMs). SVMs seek to find a hyperplane that optimally separates data in a high-dimensional space, maximizing the margin between classes. SVMs are most effective for binary classification problems but can be extended to multi-class problems. Decision Trees (DTs) are another type of learning algorithm, used for classification and regression tasks. They build a tree structure where each node represents a decision based on a particular feature of the data. Although easy to interpret and visualize, they can be sensitive to variation and bias. Another branch of supervised learning concerns Random Forests (RFs), which are an extension of decision trees. They construct a large number of independent decision trees and combine their predictions to achieve better overall performance. Random forests are more robust than individual decision trees and tend to produce better results in terms of sensitivity.

In this thesis, we shed light on one of the most widely used algorithms of recent times, which is artificial neural network learningks (ANNs). Thanks to hardware and algorithmic advances, these

models have made significant progress. Artificial Neural Networks consist of interconnected units or nodes called "artificial neurons." These neurons receive input signals, process them, and transmit them to other connected neurons. Their architecture is composed of interconnected layers of artificial neurons, where each neuron is a processing unit that receives inputs, performs calculations on these inputs, and then produces an output (Chapter 1 covers the functioning of NNs in detail).

Convolutional Neural Networks (CNNs) are a type of Artificial Neural Network specially designed, inspired by human vision, to analyze visual data such as images. Although CNNs have shown remarkable advancements, they are still considered "black boxes" due to their lack of interpretability and their exclusive reliance on data for decision-making. Explainability and interpretability of CNNs are an emerging field that is increasingly attracting the interest of the scientific community. Some approaches are based on formal logic to generate logical explanation formulas for a decision, where the involvement of a feature is confirmed if it satisfies the formula. Other approaches attempt to visualize and interpret the criteria on which the network makes its decision. Some visual techniques highlight salient regions of the images that led to the final prediction. These feature maps can be considered as generating knowledge from the network. However, the field is still budding, and no method allows a complete and tangible explanation.

Standard CNNs are also subject to several imperfections, starting with the input data that constitutes the raw material for CNNs but often suffers from various issues such as lack of labeled data, mislabeling problems, noise, and biases, etc. Furthermore, several parameters can affect the quality of learning, such as network design and hardware constraints. Data biases and learning quality significantly influence classification predictions. Additionally, standard CNNs unfortunately do not take external knowledge into account that may be present in labeled data, such as hierarchical links between labels or expert knowledge about the data.

Inspired by these motivations, in this thesis, we propose an approach that aims to integrate knowledge reasoning mechanisms (considered as a System S2) into a CNN (considered as a System S1). We explore the possibility of integrating this knowledge at different levels: at the input level by considering expert knowledge about data, at the design of the architecture and the customization of the loss function, and at the output level by refining prediction results.

The Figure 1 summarizes the context of this thesis and the positioning of its chapters.

This thesis consists of two fundamental parts. The first part focuses on integrating knowledge into the input data of the network. The first chapter deals with data preprocessing, a essential but often nonclarified step in the majority of current scientific works. To address this, we establish a formalization of preprocessing to ensure transparency and provide the necessary details for experiment reproducibility. We also investigate data augmentation, a widely adopted technique, but due to the opacity of CNNs, the explanation of its fruitful contribution remains unclear. To address this, we propose an approach based on the "Layerwise Relevance Propagation (LRP)" method, a visualization method for exhibiting the difference between the salient pixels of an original image and its transformation after passing through the network. We also present a set of measures and principles, along with elaborated experimental protocols, to assess dataset informativeness. The second chapter of this first part focuses on leveraging high-level knowledge to determine the order of presenting data to the network. To achieve this, we introduce incremental curriculum learning to order the training set. The results show increased accuracy and convergence speed, encouraging the consideration of this approach, which is generalizable to other datasets.

The second part of this thesis concentrates on integrating external knowledge into the model's design. As a particular case, we situate ourselves in the context of hierarchical multi-label clas-

sification, where external knowledge is represented by the hierarchical links between labels. We formalize two contextual constraints: the first constraint imposes that an object must be assigned to a unique class per hierarchical level, and the second constraint ensures that the global assignment respects the hierarchical link between classes, thereby avoiding hierarchical violations. To achieve this, we design a special neural network with a customized loss function to accommodate these constraints. We select five datasets for experimentation, and the results demonstrate the positive impact of considering external knowledge.

To improve the robustness of this hierarchical network, we then introduce an abstention mechanism, allowing the network to abstain if the prediction confidence is below a predefined confidence threshold. To evaluate this mechanism, we introduce new abstention-aware classification metrics, such as the rate of correct classifications ignored and the rate of incorrect classifications avoided. This mechanism is tested on the same five datasets.

This thesis is a first exploratory work to study the interest of high-level knowledge integration in CNNs, this document describes this study and the advances produced and opens the horizon to other tracks cited in perspective.

Part I

Preamble

Dans cette partie composée de trois chapitres, nous présentons un aperçu des concepts fondamentaux des réseaux neuronaux qui nous seront utiles tout au long de ce manuscrit. En effet, les réseaux neuronaux (et plus spécifiquement les réseaux de neurones convolutifs) occupent une place primordiale au sein de cette thèse, vu qu'ils simulent le système S1. Nous les avons choisis pour leur grande efficacité et leur utilisation répandue. Les CNNs sont spécialement conçus pour la classification d'images, et il est essentiel de détailler leurs mécanisme, afin de pouvoir ultérieurement intégrer des connaissances dans ces réseaux. Dans le premier chapitre, nous evoquons les concepts nécessaires à la compréhension du fonctionnement général d'un réseau de neurones. Entre autres l'architecture globale et les couches standard et les fonctions d'activation et de décision. À la fin de ce chapitre, nous présentons une formalisation probabiliste des réseaux neuronaux, qui servira de socle à la conception notre solution de classification hiérarchique.

Ensuite, il nous a paru important de répertorier les différentes connaissances de haut-niveau pouvant interagir avec les NNs, nous avons donc distingué, dans le deuxième chapitre, des types de connaisses : des connaissances humaines provenant d'un expert et des "connaissances" boîte-noire provenant de la machine. Nous présentons certains exemples de la littérature évoquant ces formes de connaissances et comment elles peuvent interagir avec les réseaux neuronaux. En outre, nous avons mentionné les connaissances que les réseaux neuronaux peuvent produire en tant que modèles "boîte noire" par le biais de méthodes d'explicabilité et d'interprétabilité. L'objectif de cette thèse n'étant pas d'étayer ces approches, nous n'avons fait que présenter une seule méthode qui sera également utilisée dans les chapitres 4 et 5. Enfin, le troisième chapitre présente les différents ensembles de données utilisés pour les expériences menées dans le cadre de cette thèse. Le caractère principal de ces ensembles de données réside dans leur nature hiérarchique, et ils ont été soigneusement choisis pour cette raison, car ils contiennent des connaissances supplémentaires sous la forme de relations hiérarchiques. Tous ces ensembles sont publiquement accessibles. Les deux premiers ensembles de données sont intrinsèquement hiérarchiques. Il s'agit notamment de l'ensemble de données "BreakHis", composé d'images histopathologiques de cancers du sein, étiquetées avec le type et le sous-type de la tumeur. Les deuxième et troisième ensembles de données sont des ensembles d'articles de garde robe (vêtements, chaussures, accessoires),organisés de manière hiérarchique, à savoir "Deep Fashion Kaggle" et "Fashion-MNIST". Les deux derniers ensembles ont specialement été modifiés en creant des niveaux d'hierarchie, il s'agit de "Prime-MINST" et "HZOO".

In this part, composed of three chapters, we present an overview of the fundamental concepts of neural networks, which will be useful throughout this manuscript. Neural networks (and more specifically convolutional neural networks) play a key role in this thesis, as they simulate the S1 system. We have chosen them for their high efficiency and widespread use. CNNs are specifically designed for image classification, and it is essential to detail their mechanism so that knowledge can be integrated into these networks at a later stage. In the first chapter, we describe the concepts required to understand the general operation of a neural network. These include the overall architecture and standard layers, as well as activation and decision functions. At the end of this chapter, we present a probabilistic formalization of neural networks, which will serve as the basis for the design of our hierarchical classification solution.

Next, we considered it important to list the different types of high-level knowledge that can interact with NNs, In the second chapter, we distinguish between types of knowledge: human knowledge coming from an expert and black-box "knowledge" coming from a machine. We present some examples from the literature evoking these forms of knowledge and how they can interact with neural networks. In addition, we have mentioned the knowledge that neural networks can produce "black-box" models through methods of explicability and interpretability. As the aim of this thesis is not to support these approaches, we have only presented one method, which will also be used in Chapters 4 and 5. Finally, the third chapter presents the different datasets used for the experiments carried out in this thesis. The main character of these datasets lies in their hierarchical nature, and they have been carefully chosen for this reason, as they contain additional knowledge in the form of hierarchical relationships. All these datasets are publicly accessible. The first three datasets are intrinsically hierarchical. These include the "BreakHis" dataset, composed of histopathological images of breast cancers, labeled with tumor type and subtype. The second and third datasets are hierarchically organized sets of wardrobe items (clothing, shoes, accessories), namely "Deep Fashion Kaggle" and "Fashion-MNIST". The last two sets have been specially modified to create hierarchy levels: "Prime-MINST" and "HZOO".

Chapter 1

Image classification in NNs: background and formalization

Résumé en français du Chapitre 1 : "Classification d'image dans les NNs : concepts de base et formalisation"

Les réseaux de neurones (NNs) sont réputés par leur performances remarquables dans la réalisation de nombreuses tâches dans divers domaines.
Dans ce chapitre, nous abordons les notions essentielles intervenant dans leour mécanisme. Nous commençons par présenter les couches standard d'un réseau de neurones, ainsi que la couche convolutionnelle spécifique aux réseaux convolutifs. Ensuite, nous présentons la phase de préparation des données, le mécanisme d'apprentissage, certaines techniques de régularisation et enfin les principales métriques d'évaluation de la classification. A la fin de ce chapitre, nous proposons une formalisation d'un classifieur en termes de probabilités. Cette formalisation nous servira de socle pour la conception de notre solution dans les chapitres suivants.

Contents

1.1 Basics about Neural Networks (NNs) 24
1.1.1 The standard architecture of a NN 24
1.1.2 Data preparation 27
1.1.2.1 Data splitting 27
1.1.2.2 Data balancing and augmentation 28
1.1.2.3 Data normalization 29
1.1.3 The learning process 30
1.1.4 Evaluation metrics 31
1.1.5 Regularization techniques 32
1.1.6 Main standard architectures 33
1.2 Formalizing the probabilistic view of a NN 35

In recent years, machine learning has made significant advances, notably with the development of neural networks, which have demonstrated impressive performance in numerous tasks and fields. They are designed to mimic the behavior of the human brain. They consist of interconnected nodes, called "neurons", organized in layers. Each neuron receives input data, processes it through mathematical operations, and produces an output, which is then transmitted to other neurons in the network. The strength of the connections between neurons, represented by numerical weights, enables the network to adapt and learn from the data. Learning in neural networks takes place through an iterative process known as "training". During training, the network adjusts its weights according to the labeled training data, comparing the predicted value with the actual value, in order to minimize the prediction error. This allows the network to generalize and produce accurate predictions on new data it has never encountered before. Figure 1 illustrates the standard architecture of a neural network.

Convolutional neural networks (CNNs) are particular artificial neural networks specifically designed to analyze visual data, such as images. CNNs are inspired by the biological visual cortex, the part of the brain responsible for processing visual information in humans and animals. The visual cortex is composed of a complex arrangement of interconnected neurons in a hierarchical organization. Similarly, CNNs are made up of several layers of artificial neurons that learn to extract and analyze visual features from input data. The key component of a CNN is the convolutional layer, which performs convolutions on input data using learnable filters. These filters mimic the receptive fields of the visual cortex, enabling the network to detect local patterns and features in the input data. By using clustering layers, CNNs reduce the spatial dimensions of feature maps while preserving important information. This clustering operation is reminiscent of the neural mechanisms that enable the visual cortex to summarize and reduce the sampling of visual information. In addition, in CNNs the same set of filters is applied to the entire layer, which helps to achieve translational invariance, enabling them to recognize patterns regardless of their position in the input. Similarly to the visual cortex, lower layers of a CNN learn basic visual features, such as edges and textures, while higher layers progressively learn more complex and abstract features. By combining these biological inspirations with powerful learning algorithms and hardware advances, CNNs have become the cornerstone of modern computer vision tasks, including classification, object detection, and image segmentation.

Among the most demanding tasks that neural networks can perform, we mention:

— Image classification: it involves classifying images into predefined categories. CNNs have been highly successful in this kind of task. We mention as an example, the seminal work on using neural networks for image classification [56]. The authors introduced the AlexNet architecture, which achieved a significant breakthrough in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012.

— Object detection: it involves identifying and localizing objects within an image. The task not only aims to recognize the presence of objects but also to provide the coordinates of their bounding boxes. Faster R-CNN, introduced in [35], has been a revolutionary approach in this field.

— Semantic segmentation: it involves assigning pixel-level labels to an image, enabling detailed understanding of the scene. The U-Net model proposed in [93] is one of the most famous semantic segmentation architectures.

— Synthetic data generation: it aims at producing new data similar to the real data. The Generative Adversarial Networks (GANs) introduced in [39] are considered as a groundbreaking

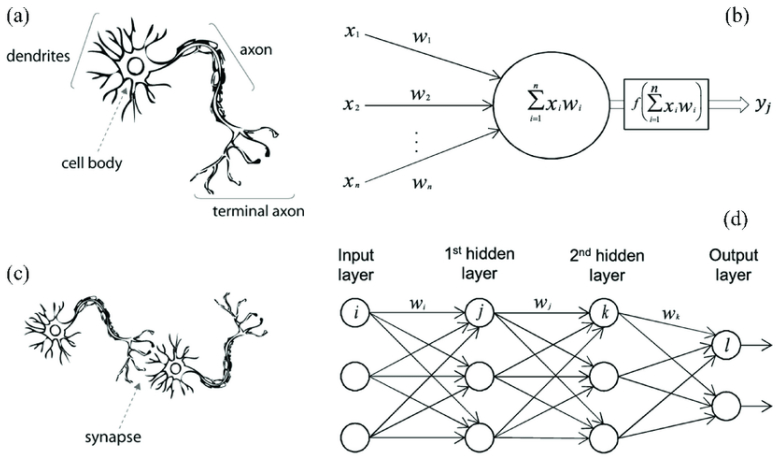


Figure 1.1 – Biological neuron in comparison to an artificial neural network: (a) human neuron; (b) artificial neuron; (c) biological synapse; and (d) ANN synapses [71].

model for this task.

— Natural Language Processing (NLP): it aims at understanding, interpreting, and generating human language in a way that is both meaningful and useful. With the rise of transformer-based models, starting by BERT (Bidirectional Encoder Representations from Transformers) model, introduced in [29], NLP has undergone a major evolution and a huge success with the achievement of generative bot like chatGPT.

— Speech Recognition: aims at enabling computers to accurately convert spoken language into written text allowing various applications such as transcription services, voice assistants, voice-controlled, etc. Recurrent Neural Networks (RNNs) and Transformer-based models have revolutionized speech recognition. DeepSpeech, proposed in [47] is considered as a seminal RNN for this purpose.

— Reinforcement Learning tasks: NNs have also been used for Reinforcement Learning tasks with the introduction of Deep Q-Networks (DQNs) in [74]. They are neural networks that serve as function approximators for a Q-table, generally used when the state-action space is excessively large to encode the reward function within a traditional Q-table representation.

In this thesis, our focus is on the task of image classification with convolutional neural networks. Specifically, we explore methods to integrate high-level knowledge into them to enhance the classification task.

In this chapter we presents the fundamental notions about the neural networks (NNs) mechanism. It covers the standard layers, the data preparation, the learning process, and the evaluation metrics. At the end of the chapter, we propose a probabilistic view of a neural network, which is useful for establishing Corollaries 6.3.1 and 6.3.2 in Chapter 6 and propositions and Equation 7.4 in Chapter 7.

Note that this chapter is based on the following documents: [37], [120], [17] and [43].

1.1 Basics about Neural Networks (NNs)

Let us consider a classification problem with k classes $c_1, c_2, \ldots, c_k$. Let $D = (X, Y)$ be the labeled dataset containing N samples s_1, s_2, ..., s_N. X is the set of images and Y is the set of their corresponding labels. Each sample s_i is a couple (x_i, y_i) of labeled images, where y_i represents the ground-truth label of the input x_i. y_i is a binarized vector of dimension k, which contains $k-1$ zeros and a unique one at the position of the corresponding class of the sample x_i. A CNN classifier aims at mapping each image to its corresponding label, by estimating a predicted label $\hat{y}_i$ that should be the closest possible to y_i. In other terms the predicted class $\hat{c}_i$ must correspond to the ground-truth class c_i. In this section, we present the essential elements of the NNs mechanism of classification.

1.1.1 The standard architecture of a NN

The main layers of a NN are:

The fully connected layer: (also known as a *dense layer*), is a type of layer where each neuron or node in the current layer is connected to every neuron in the previous layer. Suppose we have two consecutive layers in a neural network L^{i-1} with n_{i-1} neurons and L^i with n_i neurons. Each neuron in layer L^{i-1} is connected to each neuron of L^i by a weight. These weights are typically organized into a weight matrix w^i. This weight matrix of dimension $n_i \times n_{i-1}$ contains the weights for each connection between neurons in L^{i-1} and L^i. Each row of w^i corresponds to a neuron in L^i, and each column corresponds to a neuron in L^{i-1}. The output z^i of the fully connected layer is given by:

$$z^i = f(w^i \times z^{i-1}) + b^i$$

where: b^i is the bias vector of shape $(n_i, 1)$ and z^{i-1} is the input vector of the layer L^{i-1}. "$\times$" refers to the matrix product and "f" is the activation function (described below) of the layer L^i.

The fully connected layer performs a linear transformation of the input followed by a bias addition, making it capable of learning complex relationships between features in the data. Note that the weights and biases depend on each layer and will be updated during the learning phase, the activation function f is fixed for each layer.

We mention the most common activation functions:

— The sigmoid activation function: reduces the input to a value between 0 and 1. It is commonly used in binary classification and regression problems at the end of the neural network, or as a gate in recurrent neural networks. However, it is less popular in hidden layers of deep neural networks due to potential vanishing gradients. It is defined as:

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

— The ReLU (Rectified Linear Unit) activation function: outputs the input directly if it is positive and sets it to zero otherwise. ReLU is widely used in deep neural networks as it helps with gradient propagation and addresses the vanishing gradient problem. It is particularly effective in image classification and object detection tasks. It is defined as:

$$\text{ReLU}(x) = \max(0, x)$$

— The Leaky ReLU activation function is a modified version of ReLU that introduces a small

slope for negative inputs. This helps alleviate the "dying ReLU" problem, where neurons can get stuck in a state of zero activation. Leaky ReLU is often used as an alternative to ReLU in scenarios where a small amount of negative activation is desirable.

$$\text{LeakyReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{otherwise} \end{cases}$$

where α is a small positive constant.

— The hyperbolic tangent (Tanh) activation function reduces the input to a value between -1 and 1. Tanh is often used in recurrent neural networks (RNNs) or as an alternative to sigmoid. It is useful in cases where the input has negative values and it is effective for capturing complex dependencies.

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

— The softmax activation function is commonly used in the output layer for multi-class classification problems. It outputs a probability distribution over multiple classes, ensuring that the sum of the probabilities is equal to 1. Softmax is useful when you need to assign a probability to each class and make mutually exclusive predictions.

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}}$$

where x_i is the input for class i, and k is the total number of classes.

The convolutional layer: is the key piece of a CNN. It applies a set of learnable filters (also known as kernels or feature maps) to the input data. Each filter performs a convolution operation. The **output z^i of the ith convolutional layer** L^i is called a *feature map* and is defined as follows:

$$z^i = f(z^{i-1} \star w^i + b^i)$$

where $\star$ is the convolutional operator. Roughly speaking, the convolutional operator $\star$ is such that a $M \times N$ image I is downsampled thanks to a $m \times n$ kernel filter K, to obtain a new matrix C with the size $(M - m + 1) \times (N - n + 1)$ defined by:

$$C(i,j) = (I \star K) = \sum_{k=1}^{m} \sum_{l=1}^{n} I(i + k - 1, j + l - 1) \times K(k,l)$$

w^i and b^i are the convolutional weight tensor and the matrix biases respectively.

Figure 1.1.1 illustrates an example of the convolutional operator $\star$ between two matrixes.

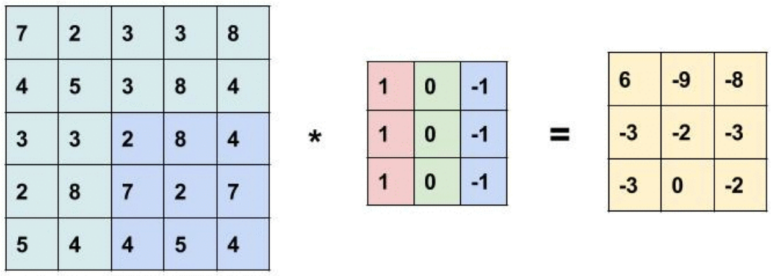


Figure 1.2 – Example of the convolutional operation.

Pooling layer: this layer generally follows a convolutional layer and aims at reducing the size of the feature map by downsampling the matrix (either taking the maximum, the minimum, or the average of the associated feature map area).

The output layer L^l: gives the predicted label $\hat{y}_i$. For example, in the case of multi-class classification, this layer is often a softmax output layer connected to the last fully connected layer (z^{l-1} in the following formula) with k neurons. The final output is:

$$z^l[i] = \frac{\exp(z^{l-1}[i])}{\sum_{j=1}^{k} \exp(z^{l-1}[j])} \text{ for all } i \in \{1, ..., k\}$$

Finally, the predicted vector $\hat{y}$ of dimension k is defined by $\hat{y} = (z^l[1], z^l[2], \ldots, z^l[k])$. Based on this output, the assignment is done using a decision function g to select the predicted class $\hat{c}$. We present below the standard decision policies usually used in neural networks:

— Argmax function: it returns the index of the maximal value in a vector. In the case of multiple maxima, it conventionally returns the first index. It is commonly used for multi-class classification tasks, to determine the class label with the highest confidence score. It is often used after a final softmax activation function. Given the vector $\hat{y}$ associated to a sample s, using the argmax decision function, the predicted class is:

$$\hat{c}(s) = c_j, \quad \text{with } j = \text{argmax}(\hat{y})$$

— threshold-based function: it is often used in binary classification tasks to convert confidence scores into binary decisions. It uses a threshold value, and any score above the threshold is classified in the positive class, while scores below are classified in the negative class. It is often used with a final sigmoid function. Given an output layer of k sigmoid neurons, $\hat{y}_j$, the confidence score of the jth class, the sample s is assigned to c_j only if its score is higher than the threshold τ:

$$\hat{c}_j(s) = \begin{cases} c_j & \text{if } \hat{y}_j \leq \tau \\ \emptyset & \text{otherwise.} \end{cases}$$

— Top-λ function: it is used in multi-class classification tasks when we want to consider the top $\lambda, (\lambda \leq k)$ predicted classes with the highest confidence scores.

$$\hat{c}(s) = \{c_{i1}, c_{i2}, \ldots, c_{i\lambda}\} \text{ where } \{i1, i2, \ldots, i\lambda\} = \text{nargmax}(\hat{y}, \lambda)$$

nargmax$(array, \lambda)$ being the function that returns the λ max elements of the array.

— Majority Voting function: In ensemble models, where a set of several classifiers are used to evaluate the prediction, each individual model makes its own prediction, and the final prediction is determined by the prediction that is given by the majority.

The NN classifier objective is to learn the optimal set of weights for its layers by minimizing the difference between the predicted labels and the ground-truth labels according to the chosen loss function. This is achieved through back-propagation and gradient-based optimization algorithms (described below).

1.1.2 Data preparation

Data preparation is a crucial step in building NNs, as it ensures the input data is in a suitable format for training the network and allows it to learn meaningful patterns and features. Effective data preparation involves three key aspects: data splitting, data balancing, and data augmentation. Before augmenting, the dataset is split into a training set, which will be augmented, a validation, and a test set described below. More samples are generated into the training dataset in order to balance the training dataset which may enhance the learning accuracy (as explored in Chapter 4). This generation is based on the original images, the transformed images are only used for training and not for testing, otherwise the same image (in an augmented version) could belong to both the test set and to the training set which would give overestimated accuracy results.

1.1.2.1 Data splitting

Let us first define the three notions that we are going to use, namely “Training”, “Validation” and “Test”, in the following we explain these notions and define the splitting rates that are used.

— Training and Validation: In the training phase, the network’s weights are updated at each iteration (an iteration corresponds to the learning of the weights by the network on one batch of samples [1] thanks to the back-propagation mechanism. The validation phase is done during the training on a different part of the dataset with no weight update mechanism. Its goal is to validate the new weights of the network on a set of samples that the CNN had not seen before, hence to appreciate the network ability to classify correctly novel data.

— Test: When the training is finished, the weights of the network are fixed and the network is ready to be tested on new data. The test phase is decisive because it determines how the final network deals with any data. It is essential to differentiate this phase from the validation one because the validation is internal to the learning process while the test is done with the final network whose weights cannot change anymore. A second difference is that the validation dataset is a small part of the data which is less representative and is used only to guide the learning process in terms of optimization.

Several data-splitting strategies can be used for CNNs, we mention the most common ones:

— Holdout Validation: The holdout validation strategy involves splitting the dataset into two parts: a training set and a validation set. Typically, a percentage of the data, such as 70

1. A "batch" refers to a subset of the training data used during one iteration of training a neural network. Instead of updating the model’s weights with each individual sample (which can be computationally expensive), training is often done in batches for efficiency.

to 80 percent, is used for training, while the remaining portion is used for validation. The holdout validation strategy does not involve a separate testing set. The model is trained on the training set and evaluated on the validation set in order to monitor its performance to tune the hyperparameters and adjust its architecture. However, this strategy may lead to overfitting [2] if the validation set is too small or not representative of the overall dataset.

— Train-Validation-Test Split: The train-validation-test split strategy involves dividing the dataset into three parts: a training set, a validation set, and a separate testing set. This strategy allows for a more reliable evaluation of the model performance on unseen data. Typically, the dataset is divided into approximately 60 to 80 percent for training, 10 to 20 percent for validation, and 10 to 20 percent for testing. The model is trained on the training set, hyperparameters are tuned using the validation set, and the final evaluation is performed on the testing set. This strategy is renowned for providing a more robust assessment of the model generalization ability.

— Cross-Validation: Cross-validation is a technique that involves splitting the dataset into multiple subsets or folds and performing training and validation iteratively. Common cross-validation methods include k-fold cross-validation and stratified k-fold cross-validation. In k-fold cross-validation, the dataset is divided into k equally sized subsets. The model is trained and evaluated k times, with each fold used as a validation set while the remaining folds are used for training. The results are averaged to obtain a more reliable estimate of the model's performance. Stratified k-fold cross-validation ensures that the class distribution remains consistent across folds, which is important for imbalanced datasets.

The choice of the data splitting strategy depends on various factors such as the size of the dataset, the availability of labeled data, and the desired level of accuracy for the evaluation.

1.1.2.2 Data balancing and augmentation

According to [38], CNN approaches require that the network be trained on a sufficient amount of data. One main issue is that the available data is often too small. Data augmentation has been introduced to address this problem and has become one of the best practices that improve CNN results. It is a technique used to artificially increase the size and diversity of a dataset by applying various transformations to the existing data samples. The goal of data augmentation is to enhance the generalization capability of the model and improve its performance by exposing it to a wider range of variations and patterns. Also, data augmentation helps in mitigating the risk of overfitting. By introducing variations through augmentation, the model is less likely to overfit the training set and can generalize well to unseen examples. Several techniques have been proposed for this data augmentation. The main standard strategies are based on applying direct transformations on the existing images, such as:

— Flipping: horizontally or vertically mirroring the image.

— Rotation: rotating the image by a given angle.

— Translation: shifting the image horizontally or vertically.

— Scaling: rescaling the image by a given factor.

— Shearing: applying shear transformations [3] to the image.

2. An overfitting occurs when the model does not learn to generalize from the data, it memorizes the training data set rather than learning the underlying patterns (see Section 1.1.5 for more details).

3. Shear transformation is an affine transformation that transforms the input image in the horizontal direction or vertical direction or both

— Noise Injection: Adding random noise to the image.
— Cropping: Extracting a portion of the image.

With the rise of generative models, data augmentation has undergone a major evolution, thanks to the ability to generate synthetic data. However, a crucial aspect of data augmentation is the labeling of these newly generated images. In some critical areas, expert labeling is essential to maintain the integrity of the dataset. To avoid this problem, people often limit themselves to label-conservative transformations that ensure that generated images get the same label as the original images.

Moreover, this lack of data can be associated with an unbalanced dataset, in which there is a considerable difference in the number of samples for one category compared to another. Depending on the classification task, this unbalanced rate may create a marginalized category during the training phase. There are several common ways to balance the dataset [101]:

— Over-sampling the minor class: the samples of the minor category are augmented by data augmentation techniques to get a total number of samples close to the one of the major class.
— Under-sampling the major class: some of the samples of the major category are removed in order to reach the same size as the minor category.
— Bagging the training set: the probability of selecting a sample in a marginalized category is raised to a higher level. Instead of augmenting the number of samples the weight of each sample is increased, which yields a balanced selection probability according to the category.

It is worth noting that the true impact of data augmentation and balancing is still unknown due to the black-box nature of neural networks. Additionally, the literature often lacks a well-detailed presentation of data preparation. One of the main objectives of this thesis is to highlight the impact of data augmentation and propose a formalization of this crucial step (see Chapter 4).

1.1.2.3 Data normalization

Data normalization, also known as data standardization or feature scaling, is an essential preprocessing step in machine learning, including CNNs. It involves transforming the input data to a standardized range to ensure that the samples in the dataset are on a similar scale. Without normalization, features with different ranges or units can lead to biased learning and suboptimal model performances. It helps to prevent some samples from dominating others due to their larger magnitude or scale. There are various normalization techniques, but two commonly used methods are:

— **Min-Max Scaling (Rescaling):** It scales the data to a specified range, often between 0 and 1. The formula for min-max scaling is:

$$x_{\text{norm}} = \frac{x - x_{\text{min}}}{X_{\text{max}} - x_{\text{min}}}$$

where x is the original sample, x_{min} and x_{max} are the minimum and maximum values in the dataset, respectively.

— **Z-Score Normalization (Standardization):** It transforms the data to have a mean of 0 and a standard deviation of 1. The formula for z-score normalization is:

$$x_{\text{norm}} = \frac{x - \mu}{\sigma}$$

where x is the original value, μ is the mean of the dataset, and σ is the standard deviation.

1.1.3 The learning process

The learning process consists of computing the weights of the network (the w^i and b^i of the previous Section). This process starts with a set of weights (either taken randomly or inherited from previously trained networks) and consists of optimizing the weights according to a training dataset. This dataset should be both reliable and big enough, which often requires some form of data augmentation (see Section 1.1.2), then the learning is done in two phases:

— The forward propagation: for a given state S of the CNN weights and a given input data associated with a ground truth label y (represented as a k binary vector with only one 1 for the ground-truth label), the network gives an output vector often denoted $\hat{y}$.

— At the end of the forward propagation a *loss* $d(y, \hat{y})$ is calculated to estimate the divergence between the ground-truth label vector y and the predicted vector label $\hat{y}$ provided by the CNN. There are many ways to estimate d. In our study, cross-entropy is chosen:

$$d(y, \hat{y}) = -\sum_{i=1}^{k} y[i] \log(\hat{y}[i])$$

— The backward propagation aims at minimizing this objective cost function by updating the weights. This update is called *gradient descent optimization*. Several algorithms exist for this purpose; we make use of the Adam optimizer [53], an iterative method that is recognized for its smooth convergence toward the objective. During backpropagation, first, the gradient of the loss function (d) with respect to the activations of the layer (f^i) is computed. This step measures how a change in the layer output affects the overall loss. The specific expression for $\frac{\partial d}{\partial f^i}$ depends on the choice of the loss function. Next, the gradient of the activation function with respect to the pre-activation values (z^i) is computed. This step captures how small changes in the pre-activation values translate to changes in the activations. The form of $\frac{\partial f^i}{\partial z^i}$ depends on the chosen activation function. To find how the loss d depends on the pre-activation values (z^i), the gradients obtained in the previous steps are combined using the chain rule:

$$\frac{\partial d}{\partial z^i} = \frac{\partial d}{\partial f^i} \times \frac{\partial f^i}{\partial z^i}$$

This gradient $\frac{\partial d}{\partial z^i}$ quantifies the impact of changes in z^i on the overall loss. Once we have the gradient $\frac{\partial d}{\partial z^i}$, we can use it to update the weights (w^i) of the layer. This update is performed to minimize the loss. The learning rate (η) controls the step size in the weight update, ensuring that the optimization process converges effectively:

$$w^i \leftarrow w^i - \eta \cdot \frac{\partial w}{\partial w^i}$$

Similarly, the biases (b^i) of the layer using the gradient $\frac{\partial d}{\partial z^i}$ and the learning rate η are updated:

$$b^i \leftarrow b^i - \eta \cdot \frac{\partial d}{\partial b^i}$$

We mention below the most common loss functions $d(y_i, \hat{y}_i)$ used to estimate the distance between the predicted label $\hat{y}_i$ and its ground truth label y_i in a training set of N samples:

— Mean Squared Error (MSE): calculates the average squared difference between the predicted and target values. It is commonly used in regression tasks and works well when

the output is continuous and the magnitude of the errors is critical.

$$\text{MSE}(y_i, \hat{y}_i) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

where N is the number of samples.

— Binary Cross-Entropy (BCE): is used for binary classification problems. It measures the dissimilarity between the predicted probabilities and the true binary labels. BCE is suitable when there are only two possible classes and the outputs are independent for each sample.

$$\text{BCE}(y_i, \hat{y}_i) = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)$$

— Categorical Cross-Entropy (CCE): is commonly used for multi-class classification problems. It also measures the dissimilarity between the predicted class probabilities and the true class labels. CCE is appropriate when there are more than two classes and the outputs are mutually exclusive.

$$\text{CCE}(y_i, \hat{y}_i) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{k} y_i[j] \log(\hat{y}_i[j])$$

where k is the total number of classes.

— Mean Absolute Error (MAE): calculates the average absolute difference between the predicted and target values. It is commonly used in regression tasks when outliers may heavily affect the model performance.

$$\text{MAE}(y_i, \hat{y}_i) = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

Several other loss functions can be designed derived from these standard functions or customized according to the task.

The learning phase requires a sufficient number of iterations to allow for a reliable update of the weights; this number depends on the number of batches and epochs. Indeed, the learning set is divided into batches of samples (of the same size). An epoch is the stage during which all the batches are processed successively by the network (for the forward and backward propagations).

1.1.4 Evaluation metrics

During the training and once it is over, the CNN ability to correctly classify is evaluated on the validation set and test set respectively. In binary classification with two classes a positive class and a negative one, these definitions are adopted:

— **True Positive (TP)**: The number of instances correctly classified as positive by the model.

— **True Negative (TN)**: The number of instances correctly classified as negative by the model.

— **False Positive (FP)**: The number of instances incorrectly classified as positive by the model when they actually belong to the negative class.

— **False Negative (FN)**: The number of instances incorrectly classified as negative by the model

when they actually belong to the positive class.

Now, we present the most common evaluation metrics based on these notions:

— Accuracy: is a common metric used to measure the overall performance of a classification model. It represents the ratio of correct predictions to the total number of predictions.

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

— Precision: measures the proportion of correctly predicted positive instances out of all instances predicted as positive. It is particularly useful when the cost of false positives is high.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

— Recall (Sensitivity/True Positive Rate): calculates the proportion of correctly predicted positive instances out of all actual positive instances. It is particularly relevant when the cost of false negatives is high.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

— F1 Score: combines precision and recall into a single metric, providing a balance between the two. It is the harmonic mean of precision and recall.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

— Specificity (True Negative Rate): measures the proportion of correctly predicted negative instances out of all actual negative instances. It is useful when the focus is on correctly identifying negative instances.

$$\text{Specificity} = \frac{\text{True Negatives}}{\text{True Negatives} + \text{False Positives}}$$

— AUC-ROC: The Area Under the Receiver Operating Characteristic (ROC) Curve (AUC-ROC) is a measure of the classifier's ability to distinguish between positive and negative instances across different probability thresholds. It provides a single scalar value that summarizes the overall performance of the model. It is computed by plotting the True Positive Rate (Recall) against the False Positive Rate (1 - Specificity) and computing the area under the resulting curve.

1.1.5 Regularization techniques

There are three main issues while training a network: *overfitting* [115]. Overfitting occurs when the network models too many details and noise from the training data in a way such that generalization to novel, unseen data is negatively impacted. The two other issues concern the convergence of the gradient descent towards the objective. The *vanishing gradient* problem (initially described in [50]) occurs when a vanishingly small gradient is obtained after the forward propagation, this prevents the weights from changing their value and may obstruct further training. The opposite problem is to avoid drastic changes of the weights, it is called *gradient descent divergence* [125]. Many regularization techniques are proposed in the literature in order to remediate these problems:

— *Loss regularization* aims at introducing a penalty term to the loss function in order to counter

overfitting and avoid vanishing and divergent gradient problems. A particular loss regularization is L2 (also called "Ridge regression") [119], it is relative to a given state S of the network weights $(w^i)_{i=1..l}$, namely:

$$Loss(y, \hat{y}) = D(y, \hat{y}) + \alpha \sum_{i=1}^{l} (w^i)^2 \tag{1.1}$$

where w^i is the weight vector of the ith layer (the layers being numbered from 1 to l) and $\alpha \in]0, 1[$ is a penalty factor (the most often used value for α in the literature is 0.01).

— *Early stopping* aims at identifying the best state of the network weights. It detects the overfitting state S_o, such that S_o is stable and its associated $Loss$ value computed from Equation 1.1 is minimal for the training batch. The overfitting aspect of S_o is characterized by the inability of the network to generalize to other samples than the ones of the training set, i.e., a low and decreasing accuracy for the validation dataset. The best state is the state S_{o-1} which is immediately before the overfitting state. As a result, this technique yields the combination of weights that minimizes the validation error and maximizes the validation accuracy [38].

— *Dropout* is a simple regularization technique in which a proportion of the network neurons is randomly set to zero during training. This prevents neurons from co-adapting to each other, which reduces overfitting [110].

1.1.6 Main standard architectures

There are several famous and influential Convolutional Neural Network (CNN) architectures that have made significant contributions to the field of computer vision. Here, we provide a brief description of some of the most notable CNN architectures:

— LeNet-5: introduced by Yann LeCun et al. in 1998, was one of the pioneering CNN architectures. It consists of several convolutional and subsampling layers, followed by fully connected layers. LeNet-5 was primarily designed for handwritten digit recognition tasks and played a crucial role in popularizing the concept of CNN.

— AlexNet: proposed by Alex Krizhevsky et al. in 2012, brought CNNs into the mainstream by winning the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012. AlexNet consists of multiple convolutional layers with large filter sizes, accompanied by max-pooling layers and fully connected layers. It demonstrated the effectiveness of deep CNNs in large-scale image classification tasks.

— VGGNet: introduced by the Visual Geometry Group (VGG) at the University of Oxford in 2014, is known for its simplicity and depth. VGGNet consists of multiple convolutional layers with small filter sizes (3x3), followed by max-pooling layers and fully connected layers. Its deep architecture with up to 19 weight layers helped establish the importance of depth in CNNs.

— GoogleNet (Inception v1): proposed by Szegedy et al. in 2014, introduced the concept of the Inception module, which consists of multiple parallel convolutional layers with different filter sizes, followed by pooling and concatenation. This architecture enabled efficient use of computational resources while maintaining strong performance. GoogLeNet also introduced the idea of network "inception" and paved the way for subsequent versions of the Inception architecture.

— ResNet: (Residual Network), presented by He et al. in 2015, introduced the concept of resid-

ual learning. ResNet addressed the challenge of training very deep networks by introducing skip connections that allowed the network to learn residual functions. This innovation enabled the training of networks with hundreds of layers, leading to improved performance and ease of optimization.

— DenseNet: proposed by Huang et al. in 2016, introduced the concept of dense connections, where each layer receives feature maps from all preceding layers. This architecture facilitates feature reuse, enhances information flow, and significantly reduces the number of parameters. DenseNet achieved leading results on several benchmark datasets.

— MobileNet: introduced by Howard et al. in 2017, aimed to optimize CNN architectures for mobile and embedded devices with limited computational resources. It employed depthwise separable convolutions, which separate the spatial and channel-wise convolutions, significantly reducing computational complexity while maintaining competitive accuracy.

— EfficientNet: proposed by Tan and Le in 2019, introduced a scalable and efficient CNN architecture that achieved state-of-the-art performance on ImageNet. EfficientNet uses a compound scaling method to balance model depth, width, and resolution (dimensions of the input data) to achieve optimal performance within resource constraints.

These are just a few examples of the most famous CNN architectures that had a significant impact on the field of computer vision. Each architecture brings unique innovations and insights, for more details, refer to [37] and [120].

In this thesis, after several attempts, we have chosen the VGG19 model [104] for our experimentations. VGG19 is a pre-trained convolutional network model, with 47 layers. There are 19 layers with learnable weights: 16 convolutional layers and 3 fully connected layers, with a total of 144M parameters. Here is a detailed description of its architecture:

— Input layer: VGG19 takes as input RGB images of size 224x224 pixels. The three color channels (Red, Green, and Blue) are processed separately.

— Convolutional Layers: The network consists of 16 convolutional layers, where each layer uses 3x3 filters with a stride of one (a stride refers to the step size used when sliding a filter kernel over an input image when computing the convolutional operation).

— Activation Function: After each convolutional layer, a Rectified Linear Unit (ReLU) activation function is applied element-wise to introduce non-linearity in the model.

— Max-Pooling Layers: After every two convolutional layers, the network employs max-pooling layers with a 2x2 window and a stride of two /flp[voir ma remarque plus haut sur "stride"]. Max-pooling helps reduce the spatial dimensions of the feature maps while retaining the most relevant information.

— Fully Connected Layers: Following the stack of convolutional and max-pooling layers, VGG19 includes three fully connected layers, each with 4096 neurons.

— Dropout: To prevent overfitting, dropout layers with a dropout rate of 0.5 are applied after each fully connected layer. Dropout randomly deactivates neurons during training, encouraging the network to learn more robust features.

— Softmax Layer: The final layer of VGG19 is a softmax layer, which is responsible for transforming the output of the network into a vector of the predictions to belong to each class. The number of neurons in the softmax layer corresponds to the number of classes for the classification task, in the initial version, VGG19 was designed for a multi-class classification of 1000 classes.

Figure 1.3 illustrates the VGG19 architecture.

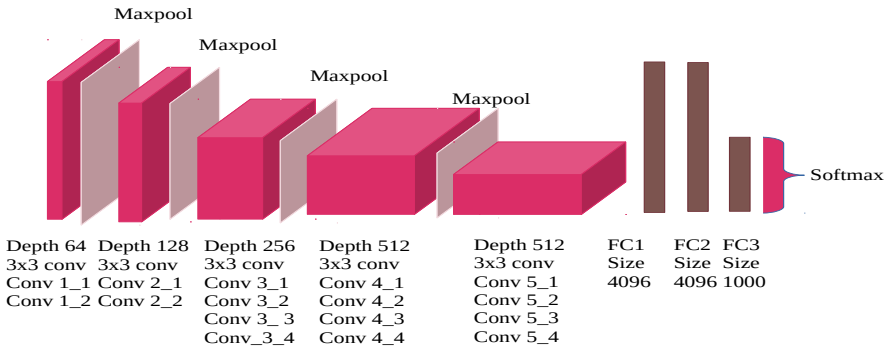


Figure 1.3 – VGG19 architecture [132], `Conv i_j` are convolutional layers with kernels of size 3×3, `FC1` and `FC2` are fully connected layers.

VGG19 was trained for many iterations on millions of images of the ImageNet dataset [51] (This dataset is described in details in Section 3.2.4). This pre-trained network can classify images into 1000 daily objects categories, such as keyboard, mouse, pencil, and many animals. VGG19 was placed second in classification and first in localization in ILSVRC'2014 competition [94]. The reason of our choice is also justified by the great success of this network on various other image classification tasks ([104] is cited more than 70000 times).

1.2 Formalizing the probabilistic view of a NN

In this section, we consider a multi-class classifier where the number of classes is k. We define a classifier as a function that associates a sample s with a set of predicted labels denoted $\hat{c}(s)$. The final output of the network is denoted $\hat{y}(s)$ and is obtained by applying the activation function f^l of the last layer l to the outputs $z^l(s)$ of the neurons z^l of this layer: $\hat{y}(s) = f^l(z^l(s))$. The last layer z^l contains k neurons. The classification decision is an "interpretation" of $\hat{y}(s)$ which selects a predicted label $\hat{c}(s)$ by applying the decision function g to $\hat{y}(s)$: $\hat{c}(s) = g(\hat{y}(s))$. In this context, $\hat{y}(s)$ is called *the confidence value of the assignment of s to $\hat{c}(s)$*. According to [84], the most recommended activation functions used at the last layer for the classification task, are the sigmoid and the softmax functions (detailed in Section 1.1.1). We explain below how these two functions are used in the classification task.

— The sigmoid of a real z is defined by: $sigmoid(z) = \frac{1}{1+\exp(-z)}$. When the activation function f^l is a sigmoid, it is applied to each output $z^l(s)$ of the penultimate layer. $\hat{y}(s))$ is then a vector of dimension k where each value $\hat{y}_j(s) = \text{sigmoid}(z^l(s)_j)$ represents the confidence with which the sample s can be affected to each class c_j from the k classes $\{c_1 \dots c_k\}$. Its complement, $\overline{\hat{y}(s)}$ is a vector of dimension k where each component $\overline{\hat{y}(s)}_j = 1 - \hat{y}_j(s)$ represents the confidence with which the sample s does not belong to the class $\hat{c}_j$. The sample is affected to the class c_j only if $\hat{y}_j(s) \geq \overline{\hat{y}_j(s)}$. More formally,

$$\hat{c}_j(s) = g(\hat{y}_j(s)) = \begin{cases} c_j(s) & \text{if } \hat{y}_j(s) \geq \overline{\hat{y}_j(s)} \\ \emptyset & \text{otherwise.} \end{cases} \tag{1.2}$$

Then $\hat{c}(s)$ is the set of possible classes among the k classes for which the confidence of

belonging is higher than its complement, hence the predicted class is not necessarily unique. More generally, it holds that:

$$0 \leq |\hat{c}(s)| \leq k$$

Example 1. *Consider a problem of multi-class classification of animal images with eight classes:* {*lion, cat, dog, bird, fish, snake, horse, panda*}.

— The softmax of a real vector $z = (z_1, \ldots, z_k)$ is defined by: $\mathrm{softmax}(z) = (s_1, \ldots s_k)$ where $s_j = \frac{\exp(z_j)}{\sum_{j=1}^{k} \exp(z_j)}$. When the activation function of the last layer is a softmax then it is applied to the vector $z(s)$ where $z_j(s)$ is the output of the jth neuron of the penultimate layer. Each value $\hat{y}_j(s)$ of the output vector $\hat{y}(s)$ is the confidence with which the sample s can be assigned to the corresponding class c_j. In the context of multi-class classification, the affectation decision is commonly done using a function g based on the argmax and designed by the user : the sample s is affected to a class with the highest confidence, $\hat{c}(s)$ is a unique predicted class such that :

$$\hat{c}(s) = g(\hat{y}(s)) = c_j \text{ such that } j \in \mathrm{argmax}\, \hat{y}(s) \tag{1.3}$$

where $\mathrm{argmax}\, \hat{y}(s)$ is (re)designed to return a unique value. It is up to the user to define the decision policy for argmax in case of equality. For example, in the Caffe library (which is one of the most common deep learning computational frameworks) in case of multiple max values, the sample is affected by the first one [4].

Example 1 (continued): *When we use a softmax output activation function instead of a sigmoid, the output is then a vector of dimension 8, whose sum is equal to 1, namely* $\hat{y}(s) = (0.7, 0.1, 0.05, 0.05,\ 0.08, 0.02, 0, 0)$. *The classification decision gives:* $\hat{c}(s) =$ { *lion* }. *The sample* s *is then affected to the unique predicted class "lion". Here, we see clearly that there is only one predicted class. If the forwarding had led to* $\hat{y}(s) = (0.35, 0.35, 0.05, 0.05, 0.08, 0.02, 0, 1, 0)$*, both "lion" and "cat" could be chosen, since they have the maximal value, but in the usual implementation of argmax, only the first one is selected.*

In the following, we establish some properties of these two activation functions.

Proposition 1.2.1

$$\text{For each } z \in \mathbb{R}, \quad \mathrm{sigmoid}(z) \in]0, 1[$$

$$\text{For each vector } z = (z_1, \ldots, z_k) \in \mathbb{R}^k, \forall j \in [1, k], \quad \mathrm{softmax}(z)_j \in]0, 1[$$

Proof. — Given a real number $z \in \mathbb{R}$, $\mathrm{sigmoid}(z) = \frac{1}{1+\exp(-z)}$. Due to the positivity of the exponential function: $\exp(-z) > 0, \forall z \in \mathbb{R}$, hence $\frac{1}{1+\exp(-z)} > 0$. More over $1 < 1 + \exp(-z(s))$, thus $0 < \mathrm{sigmoid}(z) < 1$

— Given a real vector $z = (z_1, \ldots, z_k) \in \mathbb{R}^k$, for all $j \in [1, k]$, $\mathrm{softmax}(z)_j = \frac{\exp(z_j)}{\sum_{i=1}^{k} \exp(z_i)}$. Due to the positivity of the exponential function, $\forall z \in \mathbb{R}^k, \forall j \in [1, k]$: $0 < \exp(z_j) < \sum_{i=1}^{k} \exp(z_i)$, hence $0 < \frac{\exp(z_j)}{\sum_{i=1}^{k} \exp(z_i)} < 1$.

□

4. https://github.com/BVLC/caffe/blob/master/docs/tutorial/layers/softmax.md

Proposition 1.2.2

Given a real vector $z = (z_1, \ldots, z_k) \in \mathbb{R}^k$,

— Let $\hat{y} = (s_1, \ldots, s_k)$ where $s_j = \text{sigmoid}(z_j)$, then it holds that

$$\forall j \in [1, k], \quad \hat{y}_j + \overline{\hat{y}}_j = 1$$

— Let $\hat{y} = (s_1, \ldots, s_k)$ where $s_j = \text{softmax}(z)_j$, then it holds that

$$\forall j \in [1, k], \quad \sum_{j=1}^{k} \hat{y}_j = 1$$

Proof. — By definition of the complement, $\overline{\hat{y}_j} = 1 - \hat{y}_j$, hence the result.

— For the softmax function, $\forall j \in [1, k] : \sum_{j=1}^{k} \hat{y}_j = \sum_{j=1}^{k} \frac{\exp(z_j)}{\sum_{j=1}^{k} \exp(z_j)} = 1$.

□

The following proposition shows that whatever the activation function (softmax or sigmoid) they are both probability functions, however, their domains are different. The sigmoid function corresponds to the event $\theta_j(s)$ that represents the fact that the network affects the class c_j to the sample s or not, i.e., the universe of θ_j is $\Omega_j = \{true, false\}$, $\theta_j(s) = true$ means that s is affected to c_j (false would mean s is not-affected to c_j). Concerning the softmax function, it is associated to the event $\theta(s)$ that represents the fact that the network affects a class in $[1, k]$ to the sample s, i.e., the universe of θ is $\Omega = [1, k]$, $\theta(s) = j$ means that s is affected to c_j (and no other class).

This interpretation of the activation functions as probabilities assumes an exclusivity of the issues in the universe. For the sigmoid, it is assumed that one cannot both affect a sample to a class j together with rejecting this affectation. For the softmax the interpretation is different, it is assumed that one cannot affect a sample to more than one class.

Proposition 1.2.3

Given an input vector $z(s) = (z_1(s), \ldots, z_k(s))$,

— if $\hat{y}_j(s) = \text{sigmoid}(z_j(s))$ for any j in $[1, k]$ then for any j in $[1, k[$, $\hat{y}_j$ is a probability function. This function associates to any event $\theta_j(s)$ a probability where $\theta_j(s)$ is the event of affecting the class c_j to the sample s.

— if $\hat{y}(s) = \text{softmax}(z(s))$ then $\hat{y}$ is a probability function. This function associates to any event $\theta(s)$ a probability where $\theta(s)$ is the event of affecting a class in $\{c_1, \ldots, c_k\}$ to the sample s.

Proof. — Concerning the sigmoid, for a given j in [1,k], let us consider the event $\theta_j(s)$ in the universe $\Omega_j = \{true, false\}$, we can set $\mathcal{P}(\theta_j(s) = true) = \hat{y}_j(s)$ and $\mathcal{P}(\theta_j(s) = false) = \overline{\hat{y}_j(s)}$. According to Propositions 1.2 and 1.2: for any $v \in \Omega_j$, $\mathcal{P}(\theta_j(s) = v)$ is in $[0, 1]$ and $\sum_{v \in \Omega_j} P(\theta_j(s) = v) = 1$. Hence, $\hat{y}_j(s)$ is a probability function.

— Concerning the softmax, let us consider the event $\theta(s)$ in the universe $\Omega = \{1, \ldots, k\}$, we can set $\mathcal{P}(\theta(s) = j) = \hat{y}_j(s)$ for any $j \in [1, k]$. According to Propositions 1.2 and 1.2:

for any $v \in \Omega$, $\mathcal{P}(\theta(s) = v)$ is in $[0, 1]$ and $\sum_{v \in \Omega} \mathcal{P}(\theta(s) = v)) = 1$. Hence, $\hat{y}(s)$ is a probability function.

□

Moreover, we can assume that the following events are independent:

— sigmoid: even if the two events of assigning a class j to a sample s and assigning a class j' to the same sample with $j \neq j'$ are done by the same neural network, it is nevertheless assumed that the assignments are independent of each other in the case of a sigmoid activation function.

— softmax and sigmoid: Given two samples s_1 and s_2, affecting a class to s_1 is independent of affecting a class to s_2, this is due to the fact that the network is not modified by the forward propagation of s_1. Note that the universe of the events concerning two distinct samples are disjoint they could have been named $\Omega(s_1)$ and $\Omega(s_2)$ (and for any $j \in [1, k]$, $\Omega_j(s_1)$ and $\Omega_j(s_2)$ respectively).

Proposition 1.2.4

For a network activated by a final softmax activation function, i.e., such that $\hat{y} = \mathrm{softmax}(z(s))$, for any sample s, and for any distinct j_1, j_2 in $[1, k]$, the events $\theta(s) = j1$ and $\theta(s) = j2$ are mutually exclusive.
Where $\theta(s) = j$ if and only if $g(\hat{y}(s)) = c_j$, is the event by which the network affects the sample s to the predicted class c_j.

Proof. Due to Equation 1.3, $g(\hat{y}(s)) = c_j$ for one and only one j hence the result. □

The above propositions will allow us to check the useful properties of the network proposed for hierarchical classification in Chapter 6.

In this chapter, we have presented the basic concepts and explained how a NN in general (and also a CNN in particular) works, as well as all the notions and metrics required to enable us to address the subject of this study. At the end of this chapter, we have introduced a probabilistic formalization of a CNN, which will guide the design of a specialized architecture for high-level knowledge integration in Chapter 7.

Chapter 2

Background on High-level knowledge in NNs

Résumé en français du Chapitre 2 : "Les connaissances de haut-niveau dans les réseaux de neurones"

Dans ce chapitre, nous nous intéressons à l'interaction existant entre des connaissances supplémentaires et les réseaux neuronaux. Nous commençons par distinguer deux formes de connaissances : des connaissances expertes provenant de l'humain et des connaissances boite-noire provenant du réseau de neurones. Pour les connaissances expertes, nous citons quelques travaux où elles sont utilisées de manière implicite ou explicite dans la littérature. Aussi, nous proposons une catégorisation de la prise en compte de ces connaissances-là au niveau des trois composantes principales du NN : à savoir en entrée du réseau, au niveau de la conception de l'architecture et de la fonction de perte, et enfin au niveau de la sortie du réseau. Nous décrivons quelques exemples de travaux pour chaque catégorie.
Pour les connaissances boite-noire, nous considérons les connaissances *implicites* que le réseau peut utiliser, telles que celles provenant d'un autre réseau pré-entraîné, sous forme de connaissances isuses de transfer-learning, de distilation ainsi que les connaissances produites lors du couplage de deux tâches simultanées. Nous attirons egalelment l'attention sur les "connaissances" produites via les méthodes d'explicabilité et d'interprétabilité, pour celà, nous exposons brièvement quelques méthodes phares d'intepretabilité, en détaillant seulement l'approche LRP (Layerwise Relevance Propagation) qui permet de souligner les pixels d'une image impliqués dans la décision de classification, car elle sera utilisée dans les chapitres 4 et 5.

Contents

2.1 Expert knowledge 40
2.1.1 Integrating High-level knowledge on data preparation 41
2.1.2 Integrating high-level knowledge on the model architecture and loss function 41
2.2 Black-box "knowledge" 43
2.3 Layerwise Relevance Propagation (LRP) approach 44

In general, image classification systems using a supervised learning approach rely only on labeled image datasets for training. The labels contain the information needed to guide the learning process. However, this exclusive focus on image labels can unfortunately lead to valuable external knowledge being overlooked, especially since integrating this knowledge into the learning process can significantly improve the classification task.

For example, during data preparation, external knowledge can be valuable for many tasks such as: cleaning up the dataset, validating label accuracy, selecting appropriate transformation operators for data augmentation, or labeling newly generated data. In addition, external knowledge can help design suitable network architectures and customize loss functions to better suit the task at hand. In addition, exploiting high-level knowledge can refine model results and improve the decision-making process. However, to the best of our knowledge, no work explicitly outlines all the different kinds of knowledge encountered in neural networks.

In this chapter, we illustrate some existing aspects of integrating external knowledge into neural networks. For that, we distinguish between two types of additional knowledge: expert knowledge coming from humans and intrinsic knowledge coming from the black box (which is not necessarily intelligible to humans, but useful to the machine). As the scope of this thesis does not cover the explicability of neural networks, we only present a few key methods, focusing on "layer-wise relevance propagation".

2.1 Expert knowledge

Expert knowledge refers to knowledge derived from human expertise in a specific domain. This knowledge can take different forms. We give below a list of several forms of explicit expert knowledge encountered during our study:

- Explicit rules: These rules describe relationships between data and/or labels: for instance, it may concern the label-conservative property of an operator. For example, knowing that flipping the image of a right-pointing arrow horizontally gives the image of a left-pointing arrow, can help to manage operations when increasing data.
- knowledge exceptions on input data labeling: this kind of knowledge concerning some exceptions about labels can be helpful for refining the annotation. For example, in the zoological context, the platypus is an animal that has characteristics derived from mammals, reptiles and ducks, but it lays eggs. Incorporating this knowledge refine data labeling and avoid mislabeling.
- knowledge constraints between labels: these constraints may represent relationships such as hierarchical links between labels. For example, in the classification of breast cancer tumor images, the "carcinoma" subtype is a subcategory of the "malignant" tumor type. This will be evocated in Chapter 6.
- Knowledge constraints on classification outputs: this kind of knowledge requires the network to have outputs that respect specific characteristics, such as uniqueness in a multi-class classification, where the network must assign the input sample to a single predicted class.
- knowledge on metadata associated to input data: this knowledge can be the data modality (image/text/sound/video, etc.) but also other information embedded in the data. For example in sequential data, the temporal link connecting data is a meta-data that could be very useful, for image resolution as we will see in Chapter 4. This involves designing specific network architectures, such as convolutional layers for images or recurrence mechanisms for text,

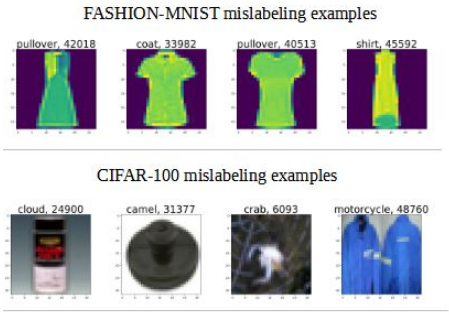


Figure 2.1 – Some mislabeling examples from Fashion-MNIST and CIFAR-100 datasets from [78].

adapted to different modalities of data.

— knowledge about dataset biases: this kind of knowledge helps to purify the dataset for a better classification. For example, in animal classification, where all the images of the "polar bear" class have a snow background, this could mislead the network about the discriminating features of this class. Such knowledge can induce a background preprocessing to avoid such influence. Other well-known examples concern racism or sexism biases caused by unbalanced or unfair datasets.

— Knowledge about the training process: such as the knowledge that adding penalizing terms to regularization techniques avoids overfitting. This means that regularization techniques are taken into account in the design of the architecture.

— Knowledge about the nature of the task: in multitasking collaboration for instance, the knowledge that two tasks are complementary may allow for enhancing the efficiency by combining them: e.g., combining a classification task with a segmentation task can have a positive impact on classification results.

Several techniques have been proposed to integrate this expert knowledge in the classification process. We have categorized these techniques into three categories according to the three key components on a neural network. For each category, we mention some relevant works.

2.1.1 Integrating High-level knowledge on data preparation

Exploiting high-level knowledge for data preparation can be observed in several forms. For instance, in [67],we deal with breast cancer images classification. Knowing that transforming an image tumor by a flip operator is label-conservative, has helped us to choose this operator for data augmentation. In addition, knowing that the microscopic magnification is an important metadata that can enhance the classification task, helped us to better organize the training phase. Also, in [78]using a labeling validation by experts, some wrongly labeled samples have been identified in several datasets, such as Fashion-MNIST dataset (which is a dataset of clothes) and CIFAR-10 dataset (which is a dataset of daily objects, as shown in Figure 2.1.1.

2.1.2 Integrating high-level knowledge on the model architecture and loss function

An example of use of external knowledge for guiding the conception of the architecture is Multi-task learning (MTL). It is a specialized area within machine learning that aims to solve multiple

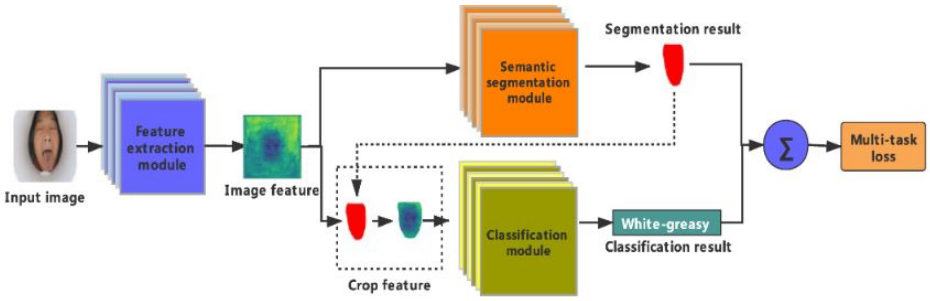


Figure 2.2 – The multi-task learning workflow for both segmentation and fine-grained classification used in [130].

learning tasks simultaneously while capitalizing on their shared features and distinctions. The core principle of MTL is to enhance learning efficiency and prediction accuracy for task-specific models by training them jointly rather than independently.

In a seminal paper from 1997, Rich Caruana defined MTL as "an approach to inductive transfer that improves generalization by using the domain information contained in the training signals of related tasks as an inductive bias. It does this by learning tasks in parallel while using a shared representation; what is learned for each task can help other tasks be learned better."

In the context of classification, MTL seeks to improve the performance of multiple classification tasks, or at least one classification task along with other related tasks, by training them jointly.

As a first example of MTL in neural networks, we can consider the Generative Adversarial Network (GAN) as a form of Multi-Task Learning. In a standard GAN (for more details about the GAN mechanism see [39]), two tasks are learned jointly: the generator task is to create realistic data samples that look like the true data distribution, while the discriminator task is to distinguish between real data samples from the training dataset and fake data samples generated by the generator. These newly generated images and the specific loss function can be considered as a way of knowledge sharing between the two parts.

Another example of MTL can be found in the work of Xu et al. [130], where they apply MTL in the context of traditional Chinese medicine (TCM). The authors tackle the challenging tasks of automatic tongue image segmentation and classification, both crucial for effective tongue characterization in TCM. They propose an approach that merges two neural networks: the UNET model for segmentation and the Discriminative Filter Learning (DFL) model for fine-grained classification. These two parts are related through a customized loss function that combines both segmentation and classification loss functions. The system starts with an initial feature extractor module that captures relevant features from the input image. This feature map is then shared as input for both the segmentor and classifier. Notably, the segmentation result is fed back into the classifier to crop the most relevant tongue regions, enhancing the shared feature-map for more accurate fine-grained classifications (see Figure 2.1.2 for illustration).

The shared feature-map, the injection of segmentation results as input for the classifier, and the constraints encoded in the combined loss function can be seen as supplementary knowledge, complementing the labeled dataset. These aspects significantly enhance both tasks, as evidenced by the obtained results.

Integrating high-level knowledge on output

High-level knowledge can also be very fruitful to refine the outputs of a NN. For instance, the abstention mechanism can be considered as a kind of using external knowledge to ensure the NN robustness, this aspect is discussed in Chapter 8. Another way to use external knowledge to rectify the outputs is by applying verification rules to the results.

2.2 Black-box "knowledge"

In this category, we consider the neural network as a black-box, that is available to produce a kind of "knowledge", not necessarily understandable by a human, but useful for the classification task.

One example of approach in this category is *transfer learning* [116]; It is a technique that involves leveraging knowledge gained from pre-training a model on one task or dataset and applying it to the new model. In the context of neural networks, transfer learning is achieved by using a pre-trained model, often trained on a large-scale dataset, as a feature extractor or a starting point for the new task. By reusing the learned representations from the pre-trained model, the target task can benefit from the “knowledge” acquired during the pre-training phase. Transfer learning is particularly useful when the target task has limited labeled data or belongs to the same semantic space as the original model. It allows the model to leverage the knowledge learned from a different, potentially more extensive dataset. The pre-trained model has already learned to capture general features and patterns, making it a valuable resource for related tasks with similar underlying structures. In 2014, Simonyan and Zisserman in [103] trained VGG16 and VGG19 models (introduced in Section 1.1.6) on the massive ImageNet dataset (described in Section 3.2.4), which contains millions of labeled images from various categories. The VGG models achieved remarkable accuracy on the ImageNet dataset. Subsequently, the researchers showed the power of transfer learning by using these pre-trained VGG models as feature extractors for other image recognition tasks. Instead of training the CNNs from scratch, they removed the last classification layers of VGG16 and VGG19 and used the earlier layers as fixed feature extractors. Then, they added new task-specific classification layers and fine-tuned the network on other smaller datasets. The results demonstrated that by leveraging the knowledge learned from the massive ImageNet dataset, the VGG models were able to achieve state-of-the-art performance on various image recognition tasks, such as object detection, image segmentation, and fine-grained image classification. Even when the target datasets for these tasks were relatively small and had different image distributions, the transfer learning approach significantly outperformed training the models from scratch.

Another aspect of this black-box "knowledge" is *knowledge distillation* [40]. It is a model compression technique in machine learning where knowledge from a larger, more complex model (known as the teacher model) is transferred to a smaller, more lightweight model (known as the student model). The main objective of knowledge distillation is to enable the student model to learn from the predictions or representations produced by the teacher model, allowing the student to mimic the teacher’s behavior and performance on a given task.

During knowledge distillation, the teacher model is typically trained on a task using the standard training procedure. Instead of using the labels for the ground truth, the student model is trained to match the final or intermediate predictions generated by the teacher model. This process involves minimizing the discrepancy between the teacher’s predictions and the student’s predictions on the

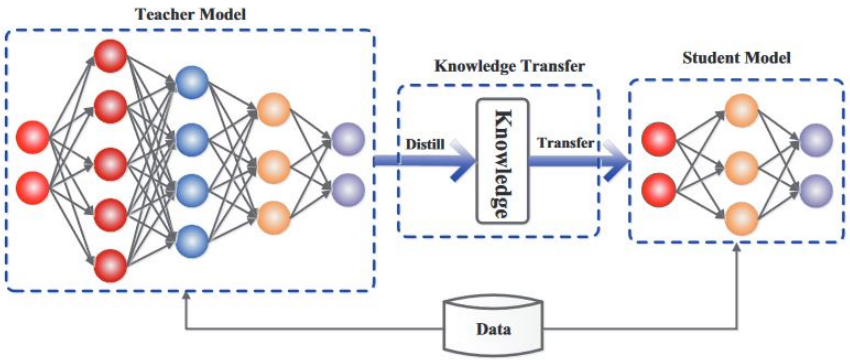


Figure 2.3 – The generic teacher-student framework for knowledge distillation [40].

same data points, as shown in Figure 2.2.

Another type of black-box "knowledge" that an NN can produce, concerns the results of explainability approaches that are used to discover how the network led to a given prediction. Indeed, Explainable Artificial Intelligence (XAI) has emerged as a crucial area of research, addressing the opacity and lack of interpretability in complex machine learning models, particularly in the context of deep neural networks. As neural networks have demonstrated remarkable performance in various tasks, their black-box nature has raised concerns regarding trust, accountability, and ethical implications. Explainable methods for neural networks seek to shed light on the decision-making process of these intricate models, providing some clues about how they arrive at their predictions.

In this context, the need for transparency becomes essential, especially in critical domains such as healthcare, finance, and autonomous systems, where model predictions influence human lives and decision-making. The lack of interpretability hinders the adoption of neural networks in these sensitive applications, as stakeholders demand explanations for the model's behavior.

As the focus of this thesis is not primarily on explainability methods, we will only present one specific method: Layerwise Relevance Propagation (LRP) which will serve as a visualization tool for some experiments in Chapters 4 and 5.

2.3 Layerwise Relevance Propagation (LRP) approach

Several works were proposed in the perspective to visualize and then understand the neural network functioning. In this chapter, we focus on one of the methods proposed by Bach et al. [9] to visualize the most relevant pixels of the image that explain the classification. This method is called *Layer-wise relevant propagation (LRP)*. Considering a CNN, the general equation translating the output of a neuron j of a given layer l connected to the neurons of the previous layer $l-1$ has the following form (as seen in Chapter 1):

$$z_j^l = g\left(\sum_{i\in\Gamma^-(j)} z_i^{(l-1)} \otimes w_{ij}^l + b_j^l\right) \tag{2.1}$$

where g is an activation function, $\Gamma^-(j)$ is the set of predecessors of j (i.e. the set of neurons entering into j) w_{ij} is the weight of the link (i,j) and b_j is an additional bias associated to the current layer and the operator $\otimes$ is either convolutional or multiplicator operator depending on the

nature of the layer.

The output of the CNN is a predicted probability vector obtained by the last layer. The aim of the LRP approach is to exploit the CNN structure and weights in order to discover the most relevant regions of the input image x called *"saliency map"* that drives the classifier decision [127]. More precisely, the relevance associated with a given neuron i of a layer l is denoted R_i^l. It is back-propagated from the network's last layer towards the first one. It starts by associating to the inputs of the neuron of the last level the highest probability classification value denoted $f(x)$, this value is then back-propagated through the networks up to the pixels, giving a pixel-wise relevance score R. This back-propagation computation is based on Kirchhoff's circuit law stipulating that at each node the sum of the inputs is equal to the sum of the outputs. In this case, the sum is a weighted sum using as weights the forward propagation values z_i^l of the CNN as shown in the following equation:

$$R_i^l = \sum_{j \in \Gamma^+(i)} \frac{z_i^l}{\sum_{j \in \Gamma^+(i)} z_j^l} R_j^{(l+1)} \tag{2.2}$$

Then, we get a final output R^1 which represents the most relevant regions of the image. Figure 2.4 illustrates the LRP mechanism. For any image i, R_i^1 is denoted $LRP(i)$.

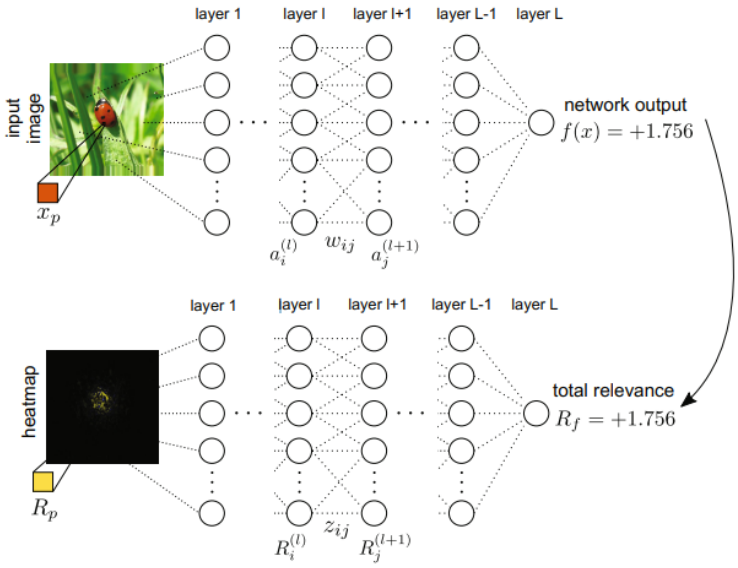


Figure 2.4 – An example on the LRP process [9]

The drawback of equation (2.2) is that it may produce unbounded values when $\sum z_j^l$ is small. In order to overcome this problem, [9] have introduced two variants of LRP:

— The $\epsilon - variant$ LRP:

$$R_i^l = \sum_{j \in \Gamma^+(i)} \frac{z_i^l}{\sum_{j \in \Gamma^+(i)}(z_j^l) + \epsilon \times sign(\sum_{j \in \Gamma^+(i)}(z_j^l))} R_j^{(l+1)} \tag{2.3}$$

with ϵ being a small positive constant in $[0, 1[$ and $sign(x) = \begin{cases} + & \text{if } x \geq 0 \\ - & \text{if } x < 0 \end{cases}$

However, in this variant, some relevant values may be absorbed by the stabilizer ϵ.

— The $\beta - variant$ LRP: is introduced to overcome the production of unbounded values with a possibility to control the importance of positive and negative terms by choosing different factors α and β.

$$R_i^l = \sum_{j \in \Gamma^+(i)} (\alpha \frac{z_i^{l^+}}{\sum_{j \in \Gamma^+(i)} z_j^{l+}} + \beta \frac{z_i^{l^-}}{\sum_{j \in \Gamma^+(i)} z_j^{l-}}) R_j^{(l+1)} \tag{2.4}$$

with $\alpha + \beta = 1$ and

the notations $^+$ and $^-$ being defined by $\begin{cases} \text{if } x \geq 0 & \text{then } x^+ = x \text{ and } x^- = 0 \\ \text{if } x < 0 & \text{then } x^+ = 0 \text{ and } x^- = x \end{cases}$

In our experiments in Chapters 4 and 5, we will use the ϵ-LRP variant because it is numerically stable and fined-grained. Indeed, according to [9] this variant is known for its ability to improve numerical stability during the relevance redistribution process. It also aims at maintaining local consistency in the attribution of feature importance. This means that the feature attributions will reflect how individual features contribute to specific parts of the model output, providing a detailed and fine-grained understanding of the model decision-making process.

In this chapter, we have discussed some forms of knowledge that interact with neural networks and some ways in which they can be taken into account. We have also evoked the black-box "knowledge" stored inside the neural network and exploited through inter-network transfer techniques or visualized through explicability approaches. Lastly, we have detailed the LRP approach which will be used in Chapters 4 and 5. Concerning expert knowledge integration, which is the subject of our study, it will be discussed in the two next parts of the thesis.

Chapter 3

BreakHis and the other datasets used in this study

Résumé en français du Chapitre 3 : "BreakHis et les autres ensembles de donnés utilisés dans cette étude"

Dans ce troisième chapitre, nous présentons les ensembles de données utilisés pour les expériences menées dans le cadre de cette thèse. Le caractère principal de ces ensembles de données réside dans leur nature hiérarchique, d'ailleurs, ils ont été soigneusement choisis pour cette raison, car ils contiennent des connaissances supplémentaires sous la forme de relations hiérarchiques. Tous ces ensembles sont publiquement accessibles. Les trois premiers ensembles de données sont intrinsèquement hiérarchiques. Il s'agit notamment de l'ensemble de données "BreakHis", composé d'images histopathologiques de cancers du sein, étiquetées avec le type et le sous-type de la tumeur. Les deuxième et troisième ensembles de données sont des ensembles d'articles de garde robe (vêtements, chaussures, accessoires), organisés de manière hiérarchique, à savoir "Deep Fashion Kaggle" et "Fashion-MNIST". Les quatrième et cinquième ensembles ont specialement été modifiés en créant des niveaux hiéarchiques, il s'agit de "Prime-MINST" et "HZOO".

Contents

3.1 BreakHis dataset 48
3.1.1 About breast cancer 48
3.1.2 BreakHis composition 50
3.1.3 A traditional approach for BreakHis images classification 51
3.1.4 CNN-based approaches for BreakHis images classification 52
3.2 Other datasets 53
3.2.1 Kaggle Fashion Product Images dataset 54
3.2.2 Fshion-MNIST 54
3.2.3 Prime-MNIST 55
3.2.4 HZoo 55

The aim of this chapter is to introduce the datasets that will be used in the experiments of this thesis.

We have selected five datasets that have the shared characteristic of being hierarchical. All these data sets are publicly accessible. The first three datasets are intrinsically hierarchical. These include the "BreakHis" dataset, composed of histopathological images of breast cancers, bilabeled with the tumor and subtype. The second and third datasets are hierarchically organized sets of wardrobe items (clothing, shoes, accessories), namely "Deep Fashion Kaggle" and "Fashion-MNIST". The two last datasets have been manually modified to create hierarchical levels, namely "Prime-MINST" and "HZOO".

3.1 BreakHis dataset

BreakHis [108] stands for “Breast Cancer Histopathological Images”, it is a public dataset [107] of histopathological images of breast cancer. The current version of BreakHis is composed of 7909 histopathological biopsy images collected from 82 patients by P&D Laboratory in Brazil. BreakHis images are observed by different microscopic magnifications: 40X, 100X, 200X and 400X with an effective pixel size of 0.49 μm, 0.20 μm, 0.10 μm, and 0.05 μm respectively. Figure 3.1 illustrates the four magnifications used in the BreakHis dataset [108].

It is worth mentioning that BreakHis is widely considered one of the most popular publicly available datasets for breast cancer histopathological images. With over a thousand citations, this large dataset is considered a benchmark for comparing various approaches in the field.

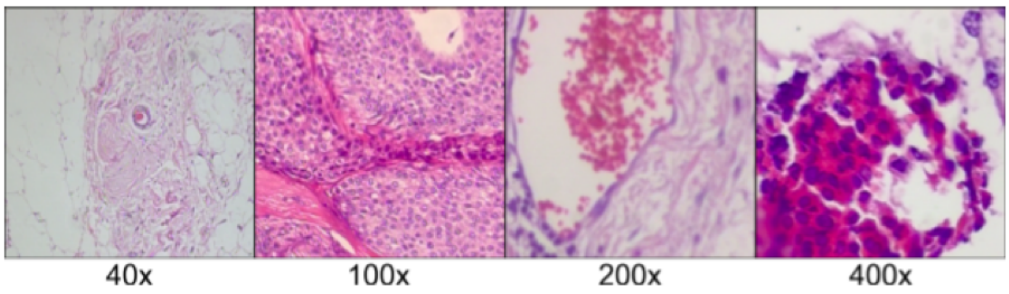


Figure 3.1 – Images from BreakHis DataSet with different magnifications.

3.1.1 About breast cancer

According to the latest World Health Organisation (WHO) report, breast cancer is the most common cancer type among women and also one of the most dangerous ones with the highest mortality rate, causing 627,000 deaths in 2018 [15]. Indeed, nowadays, 1 in 7 (14%) of women worldwide are affected by this pathology [10]. Figure 3.2 extracted from [2] illustrates the evolution of the incidence and the mortality rates of breast cancer in France between 1975 and 2020. The incidence and mortality rates of breast cancer have shown a dynamic evolution over time. The incidence rate has generally increased, which can be attributed to several factors such as improved screening methods, increased awareness leading to more women seeking medical attention, and changing lifestyle and reproductive patterns. Advancements in diagnostic techniques have also played a role in the

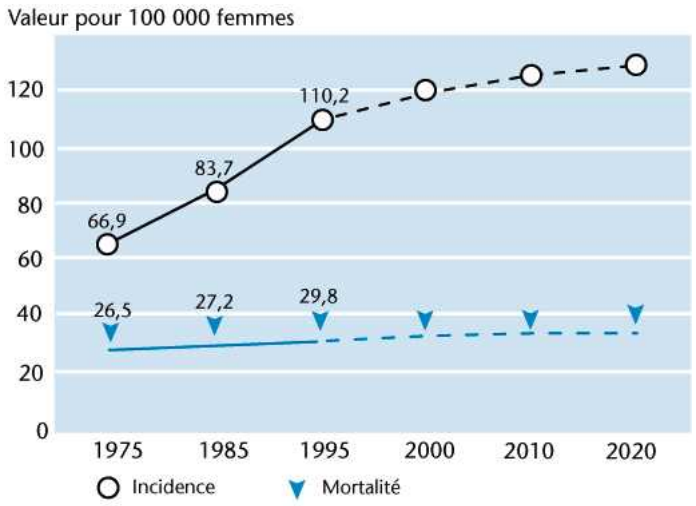
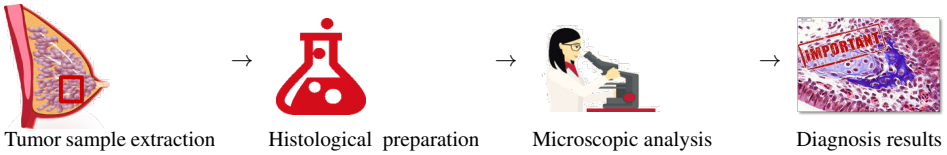


Figure 3.2 – The evolution of breast cancer mortality and incidence rates among women in France between 1975 and 2020 [2]

detection of breast cancer at earlier stages, resulting in higher reported incidence rates. On the other hand, the mortality rate of breast cancer has seen a decline in recent years. This positive trend can be attributed to advancements in treatment options, including targeted therapies, improved surgical techniques, and more personalized; approaches. Increased awareness about breast cancer symptoms and the importance of regular screenings has also contributed to earlier detection and timely interventions, ultimately reducing the mortality rate.

For a number of years now, breast cancer has been a world health priority and has triggered international scientific interest. Physicians and engineers have joined efforts in order to perform screening, diagnosis, and treatment. The earlier the disease is detected, the better it can be kept under control; hence the diagnosis phase is crucial. Figure 3.3 illustrates the diagnosis workflow established by the pathologist. First, a tumor tissue sample is extracted from the suspicious breast region. Second, the tissue is biologically prepared by adding chemical coloring agents and products for microscopic observation. The image generated from this microscopic observation is known as a "histopathological image." Third, the pathologist analyses the microscopic image to come up with the diagnosis. Finally the resulting diagnosis is delivered to the patient.



Tumor sample extraction → Histological preparation of the tissue → Microscopic analysis → Diagnosis results

Figure 3.3 – Breast cancer diagnosis workflow

The diagnosis step consists of differentiating between benign and malignant tumors (an instance of binary classification) and also identifying the sub-type of the tumor among the fifteen reference sub-types of breast cancer (an instance of multi-class classification). In this study, we will limit ourselves to eight sub-types: the ones used in the BreakHis dataset (described in Section 3.1.2]).

Currently, the diagnosis is predominantly carried out manually, which requires a lot of work and

time. Moreover, the task is subject to two main potential errors which are classical in traditional medical diagnosis and which were pointed out for breast cancer by [121], namely:

— Inter/intra observer variability: according to the U.S. National Library of Medicine, it is "The failure by the observer to measure or identify a phenomenon accurately, which results in an error. Sources for this may be due to the observer's missing an abnormality, or to faulty technique resulting in incorrect test measurement, or to misinterpretation of the data". Two varieties are inter-observer variation (variation due to two or more observers reporting on the same material) and intra-observer variation (variation caused by the same observer reporting on the same material more than once).

— The nature of histopathological image composition which is a complex image with different heterogeneous components, different global rendering of the initial colors and different background staining which may sometimes confuse the pathologist [122].

Erroneous medical diagnoses are considered to be the third cause of death in the U.S. [60]; it is even considered a stronger cause of death than the disease itself. An accurate diagnosis is evidently very important due to its life-threatening impact, and finding ways to obtain more accurate diagnoses has been the subject of many research projects.

Within this line of research, computer-aided diagnosis has been a significant research focus in recent years. To achieve this, doctors and engineers have collaborated to develop software solutions to assist practitioners in various stages. A large number of datasets have been collected, annotated, cleaned, and made publicly available. One of the most well-known datasets for breast cancer diagnosis, that we use in this thesis, is the BreakHis dataset.

3.1.2 BreakHis composition

Table 3.1 illustrates BreakHis composition. The images are divided into benign and malignant tumors that are themselves divided into different breast tumor subtypes since tumor subtypes are also very important for the treatment phase and the prognostic estimation in the clinical procedure. The BreakHis dataset contains eight breast tumor categories, four for the benign type and four for the malignant:

— Benign sub-types: Adenosis (A), Fibro Adenoma (F), Tubular Adenoma (TA), and Phyllodes Tumor (PT).

— Malignant sub-types: Ductal Carcinoma (DC), Lobular Carcinoma (LC), Mucinous Carcinoma (MC) and Papillary Carcinoma (PC)

Figure 3.4 illustrates some examples of benign and malignant subtypes.

Classes	Sub-classes	number of samples per Magnif. factors				Total
		40X	100X	200X	400X	
Benign	A	114	113	111	106	444
	F	253	260	264	237	1014
	TA	109	121	108	115	453
	PT	149	150	140	130	569
	total	625	644	623	588	2480
Malignant	DC	864	903	896	788	3451
	LC	156	170	163	137	626
	MC	205	222	196	169	792
	PC	145	142	135	138	560
	total	1370	1437	1390	1232	5429
Total		1995	2081	2013	1820	7909

Table 3.1 – BreakHis dataset composition

Due to this bi-labeling, two classification tasks are possible: the binary classification of the

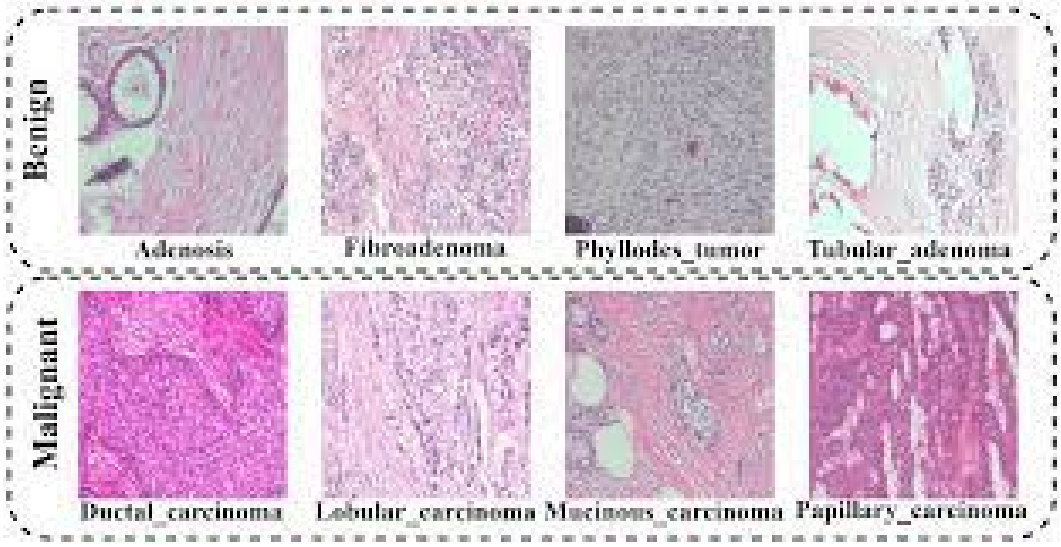


Figure 3.4 – Examples of tumors subtypes from BreakHis.

BreakHis dataset aims at classifying the images into benign and malignant, while multi-class classification aims at classifying the images into one of the eight tumor subtypes. Additionally, due to the bi-level composition of the labels, the BreakHis dataset can be viewed as a hierarchical dataset. This characteristic further strengthens our selection of BreakHis to conduct our research on the hierarchical multi-label classification problem, as detailed in Chapter 6.

Undoubtedly, the automatic classification of breast cancer images has been the goal of the scientific community for over 40 years. In the following, we present a concise state-of-the-art study, focusing on the most significant existing works on BreakHis images. In this review, we first discuss one relevant work that uses a traditional classification approach, and then we highlight several noteworthy works that employ Convolutional Neural Networks (CNNs).

3.1.3 A traditional approach for BreakHis images classification

The traditional approach to automatic analysis of histopathological images relied on handcrafted engineering of explicit features extraction and involved several user choices throughout the workflow. These choices encompassed the selection of regions of interest (ROIs) and the use of automatic methods for different steps. The workflow consisted of the following key steps:

1. ROI selection: Scientists had to in determinate which are the most relevant regions in the image to focus on.
2. ROI segmentation: Once the ROIs were identified, scientists had to choose which appropriate segmentation method to apply to extract them accurately.
3. Post-segmentation: In general, traditional segmentation approaches required a post-processing step to refine the segmentation results.
4. Descriptor selection: Scientists had to identify the most discriminant descriptors to consider for the classification task.
5. Descriptor extraction: After selecting the descriptors, scientists needed to choose the most suitable mathematical function to effectively represent these descriptors.

6. Classification: Once the discriminant descriptors were extracted, scientists selected the most suitable classifier and employed it to perform the final classification task.

As an example of traditional approach using BreakHis, Spanhol et al. in [108] have explored several state-of-the-art handcrafted features extraction methods, such as Local Binary Patterns (LBP), which is a texture analysis technique used in image processing to capture the local patterns or textures within an image by comparing the intensity of a central pixel with the intensity of its neighboring pixels. They also tried four different classifiers: 1-Nearest Neighbor (1-NN), Quadratic Linear Analysis (QDA), Support Vector Machines (SVM), and Random Forests of decision trees. In order to compute the global accuracy of these five steps of the traditional approach, the accuracy rates of the individual steps need to be multiplied, which highlights the interdependence of the steps. At each step, the choice for one particular method will necessarily result in a certain amount of errors. These errors propagate across the different steps in a snowball effect that impacts the global accuracy of the classification. At the last step of the workflow, Spanhol et al. achieved a binary classification with an accuracy rate of around 95% using a support vector machine fed by the descriptors extracted before, while the global classification rate is about 82% which is considerably lower (due to the bad accuracy rates of the previous steps) compared to the last step accuracy.

Due to the high importance of avoiding false negatives in the diagnosis phase, the results achieved by the different traditional automatic approaches – even if they were considered relatively acceptable as preliminary results – could not be considered sufficiently accurate. Moreover, due to the haphazardness of the choices made in the successive steps, the results were highly unstable. More precisely, the major drawback of the traditional approach is its workflow: first, by focusing on only ROIs, some information in the initial image may be lost even crucial for the discrimination step. For instance, the ROI on nuclei cells loses information for tumoral detection namely, the mitotic rate, the presence of tubule or mitotic spindle, the evolution of the cell shape and the tissue texture [87]. Then, for the segmentation step, the main critical issue of the cell segmentation algorithm is due to the fact that two or more cells may be clumped together. This overlapping of cells may mislead the cell-type classification, so it is important to split the touching cells [131]. Then comes the features extraction phase, which is the most critical step in the image classification because these selected features must be the most distinctive characteristics that lead to the right classification. This is a major issue in the feature engineering field. Once selected, these features must be represented by a descriptor, here comes also the problem of choosing the most suitable mathematical descriptor that represents well the feature. Another problem is to determine the number of descriptors to use and how to combine them. Finally, the choice of the classifier has a great impact on the accuracy rate and may lead to divergent outcomes.

This traditional approach relied heavily on human intervention and on domain expertise at various stages of the process. However, recent advancements in deep learning and machine learning have paved the way for automated and end-to-end approaches that alleviate the need for explicit feature engineering and user choices, the laborious and error-prone process of feature selection can be avoided.

3.1.4 CNN-based approaches for BreakHis images classification

The vast majority of the recent works using the BreakHis dataset use a CNN-based approach. Indeed, Throughout the different convolutional layers of the network, the features are implicitly chosen and extracted according to what suits best with the classification task. The most relevant

classification works are summarised in Table 3.2.

Ref.	Based-architecture	40X	100X	200X	400X	All
Binary classification						
[13]	AlexNet	81.87	83.39	82.56	80.69	NA
[106]	VGG-VD	87.00	86.2	85.20	82.90	NA
[109]	AlexNet	89.60	85.00	82.80	80.20	NA
[69]	3-Conv CNN	90.4	86.3	83.1	81.3	NA
[32]	AlexNet	90.96	90.46	90.37	89.75	NA
[133]	VGG16	91.28	91.45	88.57	84.58	NA
[11]	5-Conv CNN	96.82	96.96	96.36	95.97	NA
[128]	InceptionV3	96.84	96.76	96.49	94.71	NA
[124]	BiCNN	97.89	97.64	97.56	97.97	NA
[128]	InceptionResNetv2	97.90	96.88	96.98	96.98	NA
[5]	IRRCNN	97.95	97.57	97.32	97.36	NA
[61]	DenseNet21-AnoGAN	99.13	96.39	86.38	85.20	NA
[81]	VGG16	NA	NA	NA	NA	84.58
[70]	VGG16	NA	NA	NA	NA	92.60
[80]	VGG16	NA	NA	NA	NA	94.40
[77]	ResNetV1	NA	NA	NA	NA	98.70
Multi-class classification						
[128]	InceptionV3	90.28	85.35	83.99	82.08	NA
[128]	InceptionResNetv2	92.07	88.06	87.62	84.50	NA
[46]	CSDCNN	92.8	93.9	93.7	92.9	NA
[77]	ResNetV1	NA	NA	NA	NA	99.2
[77]	InceptionV1	NA	NA	NA	NA	99.2

Table 3.2 – CNN State of the art results for binary classification on BreakHis ("NA" meaning "Not available")

The accuracy rates obtained by these works (around 90%), demonstrate the efficiency of CNNs in breast cancer histopathology classification. Considering the performances on the BreakHis dataset of these approaches illustrated in Table 3.2), we observe that:

The lower the magnification,the higher the accuracy. (3.1)

This observation is of paramount importance and served as the central motivation for the study proposed in Chapter 5.

3.2 Other datasets

In this section, we shift our focus to three additional hierarchical datasets that will be employed in the experiments of this thesis. The first dataset is Deep Fshion-MNIST, while the other two are derived from existing datasets and have been specifically designed to incorporate the hierarchical dimension.

3.2.1 Kaggle Fashion Product Images dataset

The Kaggle Fashion Product Images dataset [3] is a widely recognized and extensively used dataset in the field of fashion classification research. It offers a vast collection of images related to fashion products. One of the notable aspects of this dataset is its predefined hierarchical structure. The dataset consists of 41027 images, and the classes are categorized into three levels:

- the coarse-grained level with 4 classes: "Apparel", "Accessories", "Footwear" and "Personal Care".
- the fine-grained level with 21 classes: "Topwear", "Bottomwear", "Dresses", "Bags", "Eyewear", "Headwear", "Footwear", "Socks", "Jewellery", "Watches", "Loungewear and Nightwear", "Innerwear and Sleepwear", "Swimwear", "Fragrance" ,"Beauty Accessories", "Belts", "Cufflinks", "Umbrellas", "Saree", "Apparel Set" and "Scarves".
- and the most fine-grained level with 45 classes: "T-shirts", "Shirts", "Jeans", "Dresses", "Handbags", "Sunglasses", "Hats", "Sneakers", "Kurtas", "Sandals", "Watches", "Sports Shoes", "Flip-Flops", "Heels", "Wallets", "Flats", "Formal Shoes", "Bracelets", "Tops", "Night Suits", "Earrings", "Backpacks", "Perfumes", "Trunk", "Luggage Accessories", "Briefs", "Rings", "Deodorant", "Jackets", "Track Pants", "Ties", "Belts", "Camisoles", "Lipstick", "Lip Gloss", "Foundation", "Eye Makeup", "Nail Polish", "Body Care", "Face Care", "Bangle", "Kurtis", "Necklace and Chains", "Cufflinks", "Rain Jacket", "Nightdress", "Dupatta", "Lip Liner", "Socks" and "Waistcoat".

Figure 3.5 illustrates some samples from this dataset.

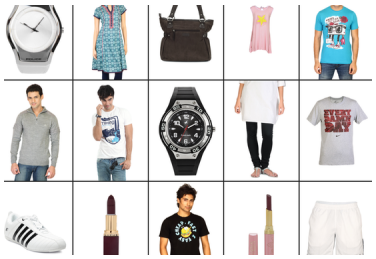


Figure 3.5 – Some images from Kaggle Fashion Product Images dataset [3].

The images in this dataset have different dimensions in terms of width and height. They range from smaller sizes, such as 32x32 pixels or 64x64 pixels, to larger sizes like 224x224 pixels or 512x512.

3.2.2 Fshion-MNIST

The MNIST dataset [28] is a widely used collection of handwritten digit images, containing 60,000 training samples and 10,000 test samples. Each image is a 28x28 grayscale picture of a handwritten digit from 0 to 9 with its ground-truth label. This dataset is a fundamental benchmark in the machine learning community, providing a standardized way of comparing and evaluating the performance of different models and techniques. Figure 3.6 illustrates some samples of this MNIST dataset. Zalando has developed Fashion-MNIST dataset [126] as a direct substitute for the original MNIST dataset. By maintaining the same image size and structure for training and testing sets, Fashion-MNIST serves as a suitable replacement for benchmarking machine learning algorithms, offering an alternative dataset for researchers to assess the performance of their algorithms. Figure

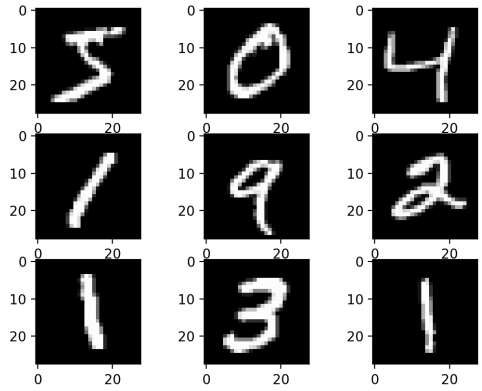


Figure 3.6 – Some images from MNIST dataset [28].



Figure 3.7 – Some images from Fashion-MNIST dataset [126].

3.7 illustrates some samples of this Fashion-MNIST dataset. The labeling of Fashion-MNIST contains 10 classes: "T-shirt/top", "Trouser", "Pullover", "Dress", "Coat", "Sandal", "Shirt", "Sneaker", "Bag", "Ankle boot".

Seo and Shin in [98] have created a 3-level hierarchy structure for this labeling, as shown in Figure 3.8.

3.2.3 Prime-MNIST

For this thesis, we designed a hierarchy strucure on the MNIST dataset with two levels, according to the primality of the numbers. The two-levels hierarchy created is as follows:

— First level (coarsest level): contains the classes "Prime" and "Non-prime".

— Second level (finest level): contains the ten classes corresponding to the ten digits from 0 to 9.

Figure 3.9 describes the designed hierarchy.

3.2.4 HZoo

For this thesis, we have also designed another hierarchical dataset, based on the Imagenet dataset [27], that we called "HZoo". ImageNet is a large-scale, publicly available dataset that has played a

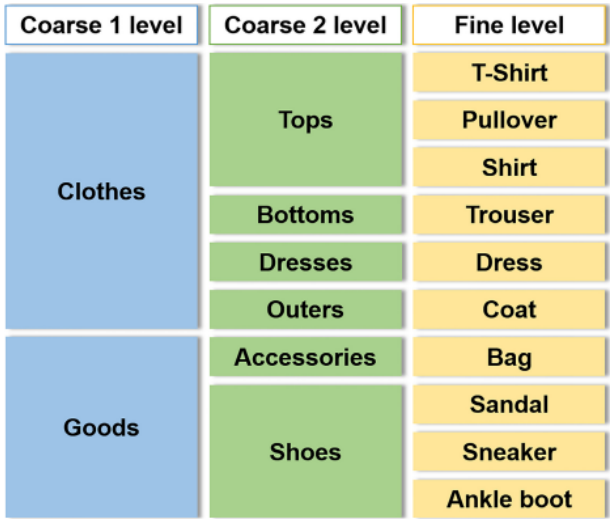


Figure 3.8 – The hierarchy structure of Fashion-MNIST dataset proposed by [98].

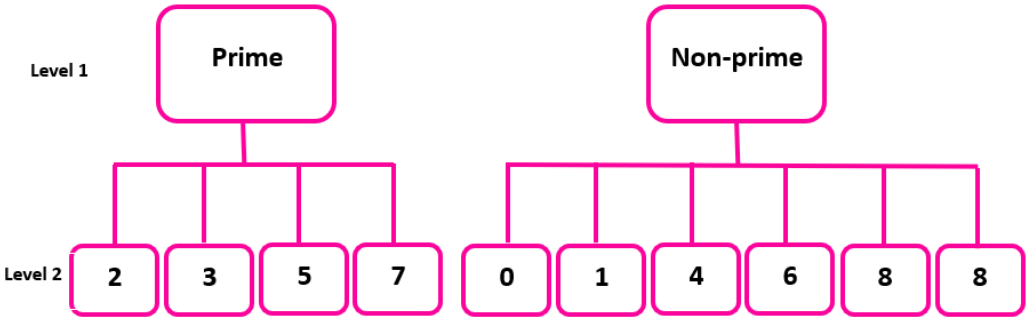


Figure 3.9 – The hierarchy structure of Prime-MNIST dataset.

significant role in advancing computer vision research and development. It has been used for years both as input for an initial training of machine learning models and for benchmarking. It consists of millions of labeled images covering thousands of different object categories. Originally created in 2009, the ImageNet dataset has undergone several iterations and updates over the years. The dataset was initially composed of 21,000 classes, but it has since evolved, notably with the creation of the ImageNet-1k version, specially designed for the ImageNet Large-Scale Visual Recognition Challenge (ICLR) in 2010. ImageNet-1k, contains 1,281,167 training images, 50,000 validation images and 100,000 test images. Based on this dataset, we extracted a subset of animal images by selecting 12 animal categories, and we chose 1000 image per category. We propose a three-levels hierarchy, as follows:

— First level (coarsest level): the animal class, we have chosen three types: "mammal", "bird" and "fish".

— Second level: for this level, we proposed non-visual criteria by considering the dangerousness of each animal: "domesticated mammal", "non-domesticated mammal", "aggressive-bird", "non-aggressive bird", "dangerous fish" and "non-dangerous fish".

— Third level: contains 12 types of animal: "cat", "dog", "bear", "lion", "flamingo", "indigo bird", "eagle", "vulture", "goldfish", "salmon", "shark" and "killer whale".

Figure 3.10 illustrates examples for the 12 selected animal categories, and Figure 3.9 illustrates the proposed hierarchy of the designed HZoo dataset.



Figure 3.10 – examples of animal images chosen from Imagenet dataset for HZoo.

In this chapter, we have presented the datasets that will be used for the experiments carried out in this thesis. In Chapters 4 and 5, we will use only the BreakHis dataset while in Chapters 6,7 and 8 we will use the five datasets for the experiments.

Through this chapter, we have selected datasets that have features in common, namely the hierarchy aspect as additional knowledge. We have also presented, through the fourth and fifth datasets, a way of conceiving extra levels of hierarchy in a dataset.

Part II

High-level knowledge used for feeding the network

Cette partie souligne l'importance du processus de préparation des données en tant qu'étape clé de la tâche de classification. Elle aborde deux questions principales : la première concerne la transparence de la préparation des données. En effet, nous avons remarqué que la plupart des articles analysés ne donnent pas de détails clairs sur le prétraitement des données, ce qui souligne la nécessité d'une formalisation explicite de cette étape. Pour combler cette lacune, le premier chapitre de cette partie présente une description complète de la préparation des données en fournissant des définitions formelles. Ce formalisme permet d'établir des propriétés précieuses pour l'ensemble de données d'entraînement utilisé dans les expériences, ce qui garantit l'équité des comparaisons et améliore la transparence de la recherche.

La deuxième question abordée dans cette partie concerne le mécanisme d'augmentation des données, une étape qui est devenue systématique dans tous les travaux sur l'apprentissage profond, mais la raison de son impact positif reste inexpliquée en raison de l'opacité des réseaux neuronaux. Ainsi, le premier chapitre propose des métriques pour évaluer l'informativité des jeux de données et pour quantifier la diversité des données à travers plusieurs expériences, afin de proposer à la fois une formalisation du processus de préparation des données et un ensemble de principes pour aider à choisir l'augmentation des données la plus appropriée. Il propose également une nouvelle technique pour mettre en évidence l'effet de l'augmentation des données.

En outre, sur la base des résultats de précision obtenus dans les études de pointe BreakHis, une hypothèse est formulée, suggérant que des connaissances de haut niveau sur l'agrandissement microscopique des images peuvent être exploitées pour améliorer les performances de classification des réseaux. Cette hypothèse conduit à une exploration de l'apprentissage basé sur le curriculum dans les réseaux neuronaux. Ensuite, des expériences sont menées sur l'ensemble de données BreakHis pour mettre en évidence l'importance de l'ordre dans lequel les données sont présentées au réseau neuronal pendant la formation. Les résultats confirment l'hypothèse.

This part highlights the importance of the data preparation process as a key step for the classification task. It addresses two main issues, the first one is about data preparation transparency. Indeed, we have noticed that most of the articles analyzed do not give clear details on data pre-processing, underlining the need for an explicit formalization of this step. To fill this gap, the first chapter of this part introduces a comprehensive description of data preparation by providing formal definitions. This formalism enables the establishment of valuable properties for the training dataset used in experiments, ensuring fairness in comparisons and enhancing research transparency.

The second issue addressed in this part concerns the data augmentation mechanism, a step that has become systematic in any work on deep learning, but the reason for its positive impact remains unexplained due to the opacity of neural networks. Thus, the first chapter proposes metrics to assess the informativeness of datasets and to quantify data diversity through several experiments, in order to propose both a formalization of the data preparation process and a set of principles to help choose the most appropriate data augmentation. It also proposes a new technique to highlight the data augmentation effect.

Furthermore, based on the accuracy results obtained in BreakHis state-of-the-art studies, a hypothesis is formulated, suggesting that high-level knowledge about the microscopic magnification of

images can be exploited to improve network classification performance. This hypothesis leads to an exploration of curriculum-based learning in neural networks. Subsequently, experiments are conducted on the BreakHis dataset to highlight the significance of the order in which data is presented to the neural network during training. The results confirm the hypothesis.

Chapitre 4

How does data preparation impact the classification quality

Résumé en français du Chapitre 4 : "Comment la préparation des données impacte la qaulité de la classification"

La préparation des données est une étape indispensable sur laquelle repose la qualité de l'apprentissage. Cependant, cette étape est souvent peu ou mal décrite dans les travaux de recherche. De plus, l'augmentation des données est adoptée comme pratique systématique suite à de son impact positif. Toutefois, les raisons de cet impact positif ne sont pas encore élucidées.
Dans ce chapitre, nous mettons la lumière sur l'augmentation des données, en proposant d'abord un formalisme pour présenter le protocole de préparation des données. Ensuite, nous énumérons des principes pour comprendre ce qu'est une bonne préparation de données et nous définissons des mesures pour évaluer l'informativité d'un dataset. Nous mettons en place des protocoles expérimentaux afin de valider les principes. Ils sont évalués par rapport à de nouvelles métriques d'informativité qui sont ajoutées aux métriques de performance de classification.
Enfin, nous proposons une approche basée sur la méthode d'explicabilité LRP pour visualiser l'effet de la data augmentation dans le dataset BreakHis.

Contents

4.1 Background about information metrics 64
4.2 General principles for efficient data preparation 65
4.3 Data preparation Formalization and Protocols for BreakHis data preparation 66
4.3.1 BreakHis transformation operators 66
4.3.2 Transformation signatures for BreakHis 67
4.4 New metrics for measuring the dataset diversity 68
4.5 Experimental Protocols 70
4.6 Results and discussion 71

Data preparation is a key step in every neural network training. However, most of the works in the state of the art do not clearly mention the way they have pre-processed the data, making the experiments difficult to reproduce. Besides, the positive impact of data augmentation, which has been adopted as a best practice of pre-processing, is still poorly understood. Based on these two issues, we design this chapter in order to provide a formalization of data preparation and to move towards an explanation for its efficiency.

For an efficient classification, CNNs require a huge amount of data. However, this is not always available: datasets are not always publicly available or may not contain sufficient data. Moreover, obtaining real data may be very expensive, also, annotating unlabeled data implies the intervention of an expert, which can be both costly and time-consuming. Consequently, data augmentation has been introduced to counter this issue and has become one of the best-practice steps of data preparation. However, due to the CNN black-box aspect, it is difficult to identify how the data structure is guiding the learning. Through this chapter, we aim at answering the following questions:

— Is data augmentation successful just because it gives a redundancy that helps the learning by a memory effect? [1]

— Is it necessary to provide fresh data or is it sufficient to generate data from the old ones?

— How can we quantify the information contained in a dataset for a given classification task?

— How does the augmentation technique impact the training process?

There has been an assumption that data augmentation is a mandatory standard step of data preparation. Indeed, it has been empirically demonstrated that data augmentation improves the classification quality [25]. We can divide the data augmentation strategies into two categories:

— Traditional data augmentation methods: are based on image transformations, such as simple transformations (flip, rotation, color transformations) that generate images extremely close to the initial data distribution space; or complex transformations, which may not seem to make sense to humans, have nevertheless had a positive impact on learning. Such as cut-out [30] (that randomly masks out square regions of the input images during training, creating holes or cutouts in the images), Mix-up [111] (combines pairs of images by linearly interpolating their pixel values), and RICAP [112] (Random Image Cropping And Patching (RICAP), creates new images by selecting random patches from different images and combining them to create a new image). However, in critical fields such as health, where the information label must be preserved, there are a lot of restrictions on the possible transformations.

— GAN-based augmentation methods: With the success of generative adversarial networks (GANs) [39], artificial fake images are generated. Even if these images are close to the initial data distribution, the labeling of these generated images represents an important issue, which restricts the use of these methods in critical fields, since it will require the validation of an expert.

Moreover, the lack of data which makes the learning process unsuccessful can be associated with an imbalanced dataset [92], in which there is a glaring difference in the number of samples for one category versus another. Depending on the classification task, this imbalanced rate may create a marginalized category during the training phase.

Through this chapter, we study how to quantify the amount of information in a dataset by first proposing several new measures, second enunciating a set of principles that should govern data preparation (and help to answer the questions introduced above), third designing several experimen-

1. The memory effect is defined in [82] as the ability of a NN to discover patterns in the training inputs that are unrelated to the label concepts. It can be equated to learning the background structure of the training data

tal protocols in order to check the validity of our set of principles, fourth experimenting them on the BreakHis dataset. And finally, propose a technique to visualize the data augmentation effect.

4.1 Background about information metrics

In this section, we first recall four classic metrics that quantify the informative state of a dataset. For this purpose, we consider a dataset D composed of n items, $D = \{s_1, \ldots, s_n\}$, each item s being associated with a unique class c which is called the label of s and denoted by $s.label = c$. The set of possible classes is denoted $\mathscr{C}$. Classically, in [76], the *abundance* $a_D(c)$ of a class c given a dataset D is the number of items of this class according to D, and the *proportional abundance* of a class c, $P_D(c)$, is the percentage of representation of a class among all the classes:

$$a_D(c) = |\{s \in D : s.label = c\}|\,, p_D(c) = \frac{a_D(c)}{n}$$

Also, according to [76], there are three indicators that have been defined in the literature for estimating the *diversity* of a dataset, namely, the *variety*, the *balance* and the *disparity*. Here we describe the metrics designed to provide values for these indicators:

— *Variety:* The richness $\mathcal{R}$ is a metric related to the *variety* and represents the number of classes effectively considered for the classification task:

$$\mathcal{R}(D) = |\{c \in C : p_D(c) > 0\}|$$

— *Balance:* The imbalance ratio $\mathcal{IR}$ is the ratio of the abundance of a majority class over the abundance of a minority class in a multi-class classification:

$$\mathcal{IR}(D) = \frac{a_D(\text{Majority class})}{a_D(\text{minority class})}$$

According to [85], the dataset is:
— little imbalanced when $1.5 < \mathcal{IR} < 3$.
— moderately imbalanced for $3 < \mathcal{IR} < 9$.
— very imbalanced when $\mathcal{IR} > 9$.
Note that in the case of a binary classification,

$$\mathcal{IR}(D) = \frac{1}{p_D(\text{minority class})} - 1$$

There are several other measures that capture the distribution of the data. However since they are all based on the proportional abundance p_D it means that they only take into account the number of items per class without considering the different inherent natures of these items [2].

— *Disparity:* The Disparity $\mathcal{D}$ quantifies the diversity of the data based on a pairwise distance

2. Three other measures are considered in [76], namely, Shannon entropy $\mathcal{H}$, Herfindahl-Hirschman $\mathcal{HHI}$ and Berger-Parker indexes $\mathcal{BPI}$, which capture respectively the uncertainty in predicting the type of an item taken at random, the probability of two random items to belong to the same class and the maximal proportional abundance:

$$\mathcal{H}(D) = -\sum_{c \in C} p_D(c) \times \log(p_D(c)) \quad \mathcal{HHI}(D) = \sum_{c \in C} p_D(c)^2$$
$$\mathcal{BPI}(D) = \max(p_D(c))$$

.

d between classes c ans c'.

$$\mathcal{D} = \sum_{c \in C} \sum_{c' \in C} d(c, c')$$

However, providing the distance d between two classes requires additional knowledge related to the context and the task.

Since Variety and Balance are only defined on the abundance of each class with respect to each other, the associated metrics will not help us very much in characterizing the quantity of information contained in the dataset. For this, we are going to propose several new metrics based on Disparity or on the diameter of the dataset, that incorporates a distance d more appropriate for images. But before this proposal, we need to define a general framework for data preparation in which we can express these metrics, which will be done after stating some general principles that should govern data preparation.

4.2 General principles for efficient data preparation

During data preparation, it is often the case that researchers use balancing techniques, or merely augment the data by doing some transformations on the samples. These so-called "best practices" are guided by the results obtained, some practices are known to work better than others. However, the hypothesis underlying the practices is not always made explicit. Moreover, it is not clear if some practices are good or not, for instance sometimes augmentation creates duplication of some samples, is it efficient to do so? Below, we enunciate a list of principles that are inspired by the "best practices" in order to give more awareness about what should be a "rational data preparation". Note that some of these principles are well known and some may seem obvious. However, the reason for their effectiveness remains unclear. By writing them we show that more experiments are required for attesting them. It also underlines the need for metrics that could characterize better the datasets, hence, justifying the work done in Section 4.4. We can enunciate six principles that may improve the selection of a good data preparation:

— *Balanced Dataset (BD)*: stipulates that a balanced dataset behaves better than an unbalanced one for a classification task

— *Sufficient Dataset Size (SDS)*: implies that too small a dataset may have a high negative impact (inefficiency and slow convergence) on the training process, even when the data is balanced.

— *No Duplication of Items (NDI)*: assumes that duplication does not compensate for the smallness of the dataset (it does not improve efficiency nor convergence)

— *Well Chosen Transformation Operators (WCTO)*: implies that choosing the right transformation operators (for instance label conservative ones) without adding "fresh" data has a positive impact on the training process. (WCTO) is also useful for balancing a dataset of sufficient size since by adding well-transformed items to the minority class we can obtain a positive impact on training.

— *Variety of Transformation Operators (VTO)*: stipulates that using diversified transformation operators has a positive impact.

— *Fresh External Data (FED)*: assumes that adding fresh external real data performs better than adding generated data (but may require more training time).

We have designed a set of experiments in Section 4.5 to evaluate these rational principles on the BreakHis dataset.

4.3 Data preparation Formalization and Protocols for BreakHis data preparation

In this section, we introduce the formalism and the signature of the transformation functions adopted in order to clearly present the data preparation of BreakHis.

4.3.1 BreakHis transformation operators

In order to augment a dataset, we need to perform transformations on its samples. A first transformation is *to do nothing*, this is done by using the identity operator denoted Id. In addition to it, our data augmentation process is based on two types of *elementary* operators that are label-conservative (see Proposition 4.3.1):

— Geometric operators: Due to the fact that BreakHis images are rectangles of 460x700, any non-mirror geometric operation would yield a different shape which would need to be reshaped or cropped in order to be fed to the CNN. To avoid this post-operation that may decrease the precision, we opted only for two operators that preserve the same shape: the horizontal and the vertical flip. These two operators are denoted respectively **H** and **V**.

— Color operators: In order to increase the number of images, we consider also the possibility of modifying the colors. We used two operators: a RGB color inversion and a transformation of the RGB encoding of the image into the HSV color encoding. These two operators are denoted respectively **c** and **C**.

Proposition 4.3.1: Label conservation[7, 114]

The operators H, V, c, and C are label conservative.

Proof. These geometric operators are label-conservative since the histopathological requirements for correct labeling impose that the tumorous characterization should be the same in every rotation and mirroring [7]. The color operators are also label-conservative according to [114]. □

In order to perform more than four distinct data augmentation, it is necessary to combine elementary operators by applying them successively. However, some combinations could create duplicate instances of the same images (e.g. HV=VH, Hc=cH, ...) due to the following symmetry property:

Proposition 4.3.2

Any application of two successive elementary operators, called combination, is symmetric except for cC (since Cc is not implemented and not guaranteed to be label conservative).

Proof. By definition of the elementary operators. □

In summary, due to symmetries, only 15 distinct combinations are possible, namely: H, V, c, C, HV, Hc, HC, Vc, VC, cC, HVc, HVC, HcC, VcC, HVcC.

4.3.2 Transformation signatures for BreakHis

We define the dataset transformations for BreakHis by using the following signatures first for splitting the dataset and then, for operating the transformations on the corresponding partition. In order to explain formally how the data is partitioned before doing the augmentation we need to introduce a function called $split$:

Definition 4.3.1: Split

Let D be a dataset, and $(r_1, r_2, \dots r_n)$ be a sequence of numbers whose total is 100, $split(D, (r_1, r_2, \dots r_n)) = (D_1, \dots D_n)$ s.t. $(D_i)_{1..n}$ is a partition of D and for all i, $r_i = |D_i|/|D| \times 100$.

Thanks to this function, we divide the BreakHis dataset into 3 subsets for training, validation, and test sets respectively. Then, we augment the training set using the following transformation function.

Definition 4.3.2: Signature of a data-transformation

A dataset D is a set of items or samples, each sample $s \in D$ has three attributes: a unique identifier $s.i$, an image $s.img$, a label $s.label$. The signature of a transformation of a dataset D is denoted

$$tr(D, ops, ratio)$$

where:

- ops: is the list of operators to apply to the different parts.
- $ratio$: is the division rate (in percentage) of the splitting of the dataset into parts (on which the operator(s) will apply).

The result of a transformation is such that:

$$tr(D, (op_1, \dots op_p), r) = (op_1(D_1), \dots, op_p(D_p))$$

where $(op_1(D_1), \dots, op_p(D_p))$ is the partition obtained by dividing D into $p = 100/r$ parts called $D_1, \dots D_p$ on which the operators $(op_1, \dots op_p)$ are applied respectively and $op(D_i)$ is an abbreviation for:

$$op(D_i) = \{(new_id, op(s.img), s.label) \quad | \quad s \in D_i\}$$

Here is a first example of an augmentation with a distinct operator applied to the distinct four parts of a dataset.

Example 2. *Let us consider the augmentation done by applying one of elementary operation among (H, V, c, C) to each 25% of the dataset D: this augmentation has the signature $(D, (H, V, c, C), 25)$. It consists in partitioning D into four parts (D_1, D_2, D_3, D_4) and apply H to D_1, V to D_2, c to D_3 and C to D_4 yielding a new dataset $D' = tr(D, (H, V, c, C), 25) = (H(D_1), V(D_2), c(D_3), C(D_4))$.*

Note that an augmentation that applies the same operator op to the whole dataset D has the following signature $tr(D, (op), 100)$.

Remark 1. *Note that* $split(X, (25, 25, 25, 25))$ *gives the same quadruplet of sets of images than* $tr(X, (Id), 25)$*, indeed, it amounts to splitting the dataset into four equal parts. However* tr *produces new samples while* $split$ *uses the existing ones.*

Thanks to this formalization, we can encode any transformation protocol for BreakHis dataset. It is worth noticing that this formalization can also be generalized to any other data set with any other transformation operators.

4.4 New metrics for measuring the dataset diversity

According to the definitions recalled in Section 4.1, in order to compute the *disparity* $\mathcal{D}$ of a dataset, we should be able to provide a way to compute the distance between the different classes. We propose to define the distance between two classes by introducing first the distance between two images. Then we base the distance between two classes on the distance between the means of each class. There are several ways to compute the distance between two images, for instance, the Euclidean distance is based on a point-to-point comparison of the pixels of each image (it is the norm of the matrix difference). Another idea is to take into account extra information in order to integrate into the distance the fact that horizontal and vertical symmetries should not increase the distance between images, because for a classification task these symmetries do not matter. This is why we choose to use a standard measure called SSIM (structural similarity index measure) [123] which estimates the similarity of two images based on a kind of contraction of the images according to their luminance, contrast and structure.

Definition 4.4.1: Structural Similarity Index measure (SSI) [123]

Let s_1, s_2 be two samples,

$$SSI(s_1, s_2) = \frac{(2\mu_1\mu_2 + \alpha_1)(2\sigma_{12} + \alpha_2)}{(\mu_1^2 + \mu_2^2 + \alpha_1)(\sigma_1^2 + \sigma_2^2 + \alpha_2)}$$

where μ_1, μ_2, σ_1^2, σ_2^2, σ_{12}, α_1, α_2 are the means and variance of $s_1.image$ and $s_2.image$, the co-variance of $s_1.image$ and $s_2.image$, and two small constants respectively. [a].

a. These constants were introduced by [134] to avoid instability when the denominator is close to 0 by setting $\alpha_1 = 0.01 \times L$ and $\alpha_2 = 0.03 \times L$ where L is the dynamic range of the pixel values. For BreakHis dataset, with 8 bits/pixel images, $L = 255$.

Note that this similarity measure is invariant through vertical and horizontal flip, since an image and its flipped version have the same average and variance. However this does not hold for color operation.

We propose a *SSI* variant, called *SSIC* that is label-conservative for all operations introduced in Section 4.3.1, i.e., which is invariant to color operations **c** and .**C**.

Definition 4.4.2: SSIC

Let s_1, s_2 be two samples,

$$SSIC(s_1, s_2) = \max_{op \in \{Id, c, C\}} SSI(s_1, op^{-1}(s_2))$$

Example 3. *for instance, if s is a sample to be compared to its transformed sample by RGB color inversion $c(s)$, the similarity by SSIC$(s, c(s))$ is 1 since $c^{-1}(c(s)) = s$*

Proposition 4.4.1

For any combination *cop* of the elementary operators $\{Id, H, V, C, c\}$, it holds that for all sample s, *SSIC*$(s, op(s)) = 0$

Proof. The proof concerning the geometric operators is due to the definition of *SSI* and is already explained in the Note below Definition 4.4. *SSIC* being built on *SSI* to ignore color transformations hence the result. □

We are now in position to define the best representative sample among a set X, called $\mu_I(X)$. It is the sample which is the most similar to the other samples of X:

$$\mu_I(X) = \text{argmax}_{x \in X} \sum_{y \in X \setminus \{x\}} \textit{SSIC}(x, y)$$

Another way to define $\mu(X)$ is to compute the most similar image according to a threshold (where the threshold th could be set for instance to the average of pairwise similarity, $th = \frac{\sum_{x \in X} \sum_{y \in X \setminus \{x\}} \textit{SSIC}(x,y)}{|X| \times (|X|-1)}$).

$$\mu(X) = \text{argmax}_{x \in X} \sum_{y \in X \setminus \{x\} | \textit{SSIC}(x,y) > th} \textit{SSIC}(x, y)$$

We propose to evaluate the diversity of a dataset D by its disparity and diameter. The disparity has already been recalled above and is related to the distinction between the different classes. The diameter is a general measure of the scope of the whole dataset independently of the classes, it is the maximum distance between any two images of the dataset. These two measures can be defined either on the Euclidean distance d or on the more informed similarity measure *SSIC* yielding four measures $diam$, $disp$, $diam_I$ and $disp_I$ where I stands for "informed measure". We normalize the Euclidean distance by the the greatest possible distance matrix denoted $|im255|$, i.e., the image composed of 255 on the three channels RGB (since the images are 460×700 then $|im_{255}| = \sqrt{460 \times 700 \times 3 \times 255^2} = 250627.5125$).

Definition 4.4.3: Informed metrics

Given a dataset D divided into two classes D_1 and D_2 ($D = D_1 \cup D_2$),

- diam(D)= $\frac{\max_{s_1, s_2 \in D} d(s_1.image, s_2.image)}{|im_{255}|}$
- disp(D)= $\frac{d(\mu(D_1), \mu(D_2))}{|im_{255}|}$
- $diam_I(D)$= $\max_{s_1, s_2 \in D}(1 - \textit{SSIC}(s_1.image, s_2.image))$.
- $disp_I(D) = (1 - \textit{SSIC}(\mu_I(D_1), \mu_I(D_2))$.

Note that $disp_I(D)$ and $diam_I(D)$ are based on the Structural Dissimilarity Index $DSSIM(x, y) = \frac{1-SSIM(x,y)}{2}$ [96]. Also concerning disparities, the definitions are given for a binary classification ($|\mathscr{C}| = 2$) where D_1 is the part of the set D containing the first class and D_2 is the part of the set containing the second class. [3]

3. If there were more than two classes, the disparity would be $\frac{2 \sum_{c \in C} \sum_{c' \in C \setminus \{c\}} d(\mu(D_c), \mu(D_{c'}))}{|C| \times (|C|-1) \times |im_{255}|}$

4.5 Experimental Protocols

In this section, we propose 13 different data preparation protocols for the BreakHis dataset, designed with the purpose of enabling us to validate the General Principles enunciated in Section 4.2. D denotes the part of BreakHis dataset items assigned for training (we took 2/3 of the initial dataset), and D_i denotes the new training dataset after preparation with the protocol P_i. In the following, all the samples s' of D_i are such that $s'.label = s.label$ where s is the original sample in D which results in s' in D_i (by a transformation in $\{Id, H, V, C, c, HC, Vc, CV, cH\}$). In other words, the protocols create new samples that are labeled according to their initial label in the original dataset.

BreakHis training dataset D is composed of two classes, the marginal class called m, is the benign category: $m = \{s \in D | s.label = benign\}$. The majority class denoted M is the malignant category: $M = \{s \in D | s.label = malignant\}$. Hence $D = M \cup m$. Note that m has a size equal to half the size of the majority class M, such that: $|D| = 5271, p_D(M) = 2/3, p_D(m) = 1/3$.

The protocols use these characteristics for balancing the data.

1. Protocols 1 (no data creation): P_{1a} is a control protocol where no balancing nor augmentation is processed to the dataset. $D_{1a} = D$; P_{1b} is a second control protocol where only one augmentation is done without bringing "new" information: mere identical duplication of the items of the already majority class $D_{1b} = D \cup tr(M, (Id), 100)$; P_{1c} is a third control protocol which does not bring any "new" information but increases the size by simple duplication of the items in order to balance and augment the data $D_{1c} = D \cup m \cup tr(D \cup m, (Id), 100)$.
2. Protocols 2 (balanced data): the size of the minority class with only one operator. P_{2a} uses a geometrical operator: $D_{2a} = D \cup tr(m, (H), 100)$; P_{2b} uses a color operator : $D_{2b} = D \cup tr(m, (C), 100)$; P_{2c}: balance by under-sampling. $D_{2c} = m \cup Sample(M, |m|)$ where $Sample(X, n)$ is a function that randomly selects n elements among the set X;
3. Protocols P_{3a} (augmented unbalanced data) uses a color operator to augment the size of the majority class: $D_3 = D \cup tr(M/2, (C), 100)$.
4. Protocols 4 (balanced and augmented data): with two single successive operators. P_{4a} uses the geometrical operators H and V: $m' = m \cup tr(m, (H), 100)$ (double the size of the minority), $D_{4a} = M \cup m' \cup tr(M \cup m', (V), 100)$ (augment the whole dataset); P_{4b} is similar to P_{4a} but uses the color operators C and c; P_{4c} uses the operators H and C; P_{4d} uses the operators V and c. P_{4e} uses the four operators applied on different parts of the dataset: $m' = m \cup tr(m, (H, V, C, c), 25)$ (double the size of minority), $D_{4e} = M \cup m' \cup tr(M \cup m', (C, c, V, H), 25)$ (augment [4] the whole dataset). P_{4f} supplies the lack of data by adding samples from another dataset [5]: $D_{4f} = M \cup m \cup m_extra \cup M_extra$ where m_extra (resp. M_extra) is a set of $3|m|$ (resp.$|M|$) minority (resp. majority) category images of the other dataset.

The following proposition shows that all the protocols that we provide except P_{4f} does not bring any "new" information to the dataset.

4. More precisely, $D_{4e} = M \cup m \cup tr(m, (H, V, C, c), 25) \cup tr(M, (C, c, V, H), 25) \cup tr(m, (C, c, V, H), 25) \cup tr(m, (HC, Vc, CV, cH), 25)$
5. https://iciar2018-challenge.grand-challenge.org/Dataset/

Proposition 4.5.1

for all the datasets D_{ij} obtained by the protocols except D_{4f}

$$diam_I(D_{ij}) = diam_I(D_{1a}) \text{ and } disp_I(D_{ij}) = disp_I(D_{1a})$$

Proof. The proof is based on the invariance of *SSIC* wrt color and geometric operators. □

4.6 Results and discussion

In this part, we aim to estimate the amount of information that is contained in the different datasets obtained by the previous protocols. As said in Section 4.1, the variety and balance can be respectively estimated through the richness measure $\mathcal{R}$ and the imbalance ratio $\mathcal{IR}$. This section evaluates the different protocols on two aspects, first the quantity of information present in the dataset produced by the protocols, second the classification efficiency given by a CNN trained on these datasets. Table 4.1 gives the different sizes D_{ij} of the datasets obtained by the different protocols P_{ij}. Note that the *richness* of the dataset obtained with any of the protocols remains the same since the number of classes remains constant: $\mathcal{R}(D_{ij}) = 2$ for all the datasets. Concerning the *balance*, the imbalance ratio $\mathcal{IR}$ of the datasets D_{ij} obtained by the different protocols is always 1 (due to the doubling of the size of the minority class that has a size equal to half the one of the majority class), for any protocol P_{ij} except P_{1a}, P_{1b} and P_3. Note that due to Proposition 4.5, the informed disparities and diameters are the same for all the datasets except D_{4f}.

For the binary classification task, we used the VGG19 neural network described in 1.1.6. We used the pre-trained "VGG19" convolutional neural network model as a classifier to compare the different data preparation protocols. In order to optimize the network training, we used several regularization techniques such as the L2 regularization with α set to 0.01, the early stopping, and the dropout. We trained our model for 3000 epochs with a batch size of 64. We opted for Adam-optimiser for a learning rate fixed initially at 0.0001.

P	$\lvert D_{ij}\rvert$	$\mathcal{R}$	$\mathcal{IR}$	$disp$	$diam$	$disp_I$	$diam_I$
1a	5271	2	2,19	0.0254	0.1299	0.0975	0.0157
1b	8785	2	4,38	0.0254	0.1299	0.0975	0.0157
1c	14056	2	1	0.0254	0.1299	0.0975	0.0157
2a	7028	2	1	0.0528	0.1299	0.0975	0.0157
2b	7028	2	1	0.0826	0.2453	0.0975	0.0157
2c	3514	2	1	0.0265	0.1045	0.0975	0.0157
3a	7028	2	3,28	0.0654	0.4213	0.0975	0.0157
4a	14056	2	1	0.038	0.1465	0.0975	0.0157
4b	14056	2	1	0.1168	0.5812	0.0975	0.0157
4c	14056	2	1	0.2051	0.3489	0.0975	0.0157
4d	14056	2	1	0.1030	0.4731	0.0975	0.0157
4e	14056	2	1	0.4361	**0.8312**	0.0975	0.0157
4f	14056	2	1	**0.4673**	0.4369	**0.1385**	**0.2413**

Table 4.1 – Metrics Results obtained on BreakHis

Table 4.2 describes the results obtained by the network trained on the datasets produced by the different protocols. *Acc*, *Prec*, *Rec* refer respectively to the accuracy, the precision, and the recall

rates. We also give an indication of the training behavior by mentioning the stabilization's epoch *StbE* which is computed thanks to the early-stopping regularization technique [90]. It is the epoch from which the training loss is nearly steady.

Table 4.3 shows the principles that seem to support the protocol performance, for WCTO and VTO we precise respectively the list of transformation operators and the number of distinct operators used.

P	StbE	Acc (%)	Prec (%)	Rec (%)
1a	∞	47.23	53.22	48.59
1b	∞	49.08	46.59	49.03
1c	∞	50.01	48.23	47.71
2a	1966	64.12	65.27	67.02
2b	2133	69.43	66.15	68.13
2c	∞	50.03	46.02	49.93
3a	∞	55.79	52.03	56.46
4a	2146	88.63	75.10	70.02
4b	2369	85.36	71.36	69.04
4c	1967	90.02	85.03	88.52
4d	2513	84.29	72.13	78.96
4e	2719	**95.63**	78.49	75.16
4f	∞	**96.03**	**89.46**	**91.75**

Table 4.2 – Accuracy Results obtained on BreakHis ("∞" means that the network remains unstable throughout the entire training phase and requires more time to reach stabilization).

P	BD	SDS	NDI	WCTO (ops)	VTO (nb ops)	FED
1a	no	no	yes	no	0	no
1b	no	no	no	no	0	no
1c	yes	yes	no	no	0	no
2a	yes	no	yes	H	1	no
2b	yes	no	yes	C	1	no
2c	yes	no	yes	no	0	no
3a	no	no	yes	C	1	no
4a	yes	yes	yes	V + VH	2	no
4b	yes	yes	yes	c + cC	2	no
4c	yes	yes	yes	C + CH	2	no
4d	yes	yes	yes	c + cV	2	no
4e	yes	yes	yes	C + c + V + H + CH + cV + VC + Hc	8	no
4f	yes	yes	yes	no	0	yes

Table 4.3 – Principles satisfied by the protocols

In Table 4.2, the bad results of P_{1a} and P_{2c} underlines that *a too small dataset has a high negative impact on the training process, even when the data is balanced* confirming the principle (SDS). Also, these two datasets have the smallest disparity and diameters (absolute and informed). Having a small disparity means that the images of the two classes are near to each other making more difficult the discrimination task.

Duplication does not compensate the smallness of the dataset. Also, compensating the lack of data by duplicate /ing identically the same images makes the training even more difficult and yields the CNN into over-fitting (P_{1b} and P_{1c}), because for these latter protocols, the CNN is unstable and blocked in a transitory regime with a bad accuracy under 50%, confirming the (NDI) principle.

Using data augmentation with no "fresh" data but with transformed items has a positive impact on the training process (since P_{4a} gives better results than P_{1c} and P_{2b} being better than P_{2c}) this confirms both (WCTO) and (BD) principles.

Moreover, the reader can check that *augmenting a balanced dataset increases the performances*, (see $P_{4abcdef}$ wrt P_{2abc}) which supports again the (SDS) and (BD) principles. Note that *the color transformations have better impact than the geometric ones* (P_{2b} being better than P_{2a} and P_{4c} than P_{4a}), consolidating the (WCTO) principle. In parallel, we observe that both the disparity and the diameter are augmented by adding transformed samples relating these measures to the (WCTO) and (VTO) principles.

In addition, we conclude *that varying the operators by using them on different parts of the dataset, increases the accuracy*: the best accuracy of 95.63% is obtained in that case (P_{4e}) with the use of 8 different operators demonstrating the importance of (VTO) principle which is again correlated with a high disparity and diameter.

Lastly, we see that P_{4f} has the best performances with the addition of fresh external data (FED) but this protocol needs more training time. Obviously having the possibility to add fresh external data is ideal, however, it is not always possible to find more real data, this is why we can consider that P_{4e} and P_{4c} are the best data preparations.

Contrary to what was expected, several datasets that have equal values with $disp_I$ and $diam_I$ may have very different efficiency. These measures capture a kind of brute richness similar to the one that a human expert could have given by understanding the equivalences between samples. It seems that the network is benefiting from the creation of equivalent samples which do not increase what we call "informed" disparity and diameter but increase the non-informed disparity and diameter.

When comparing the results of metrics used to evaluate informativeness with those used for classification, it becomes evident that the informativeness metrics do not sufficiently reflect the intrinsic state of the dataset and the effectiveness of transformations for the classification task.

To address this issue, we found it necessary to devise a method for evaluating what the network deems relevant in an image in comparison to its transformation by an operator. We propose a method based on the Layerwise Relevance Propagation (LRP) pixel visualization technique described in Section 2.3 and the similarity metric defined in Definition 4.4, leading to the new metric SSI_{tr} that evaluates the similarity between the feature map of a sample $s1$ and the saliency map of its transformed version by the operator op. Algorithm 0 details the SSI_{tr} function.

Algorithm 1: $SSI_{tr}(s, op)$ function

function $SSI_{tr}(s1, op)$
  $s2 \leftarrow op(s1)$
  $s1' \leftarrow LRP(s1)$
  $s2' \leftarrow LRP(s2)$
  if $(op \in \{H, V\})$: $\{s2' \leftarrow op^{-1}(s2')\}$
  return $(SSI(s1', s2'))$
end function

In Algorithm 0, we consider a sample $s1 \in D$ and its transformed version by the operator op: $s2 = op(s1)$. The aim of this approach is to compare what was relevant for the classification task for the sample $s1$ with what was relevant for its transformed version $s2$ using the operator op. We generate $s1'$ and $s2'$ the saliency maps of $s1$ and $s2$ respectively using the LRP approach: $s1' = LRP(s1)$ and $s2' = LRP(s2)$. Now, we have to compare these two saliency maps using the

SSI metric. Since the geometric operators used in our framework modify the location of pixels, we need to apply the inverse operator to $s2'$ in order to restore the same spatial distribution to guarantee a plausible comparison between $s1'$ and $s2'$. At the end the function SSI_{tr} returns the similarity measure between $s1'$ and $s2'$ which translates the difference between what the NN considers more discriminating in the original image and its transformed version by the given operator.

The closer the SSI_{tr} values are to 1, the more it indicates the similarity between the saliency maps s1 and s2, suggesting that they highlight the same areas. This implies that s1 and its transformation reveal the same salient pixels for this classification task. In this case, we can conjecture that this transformation operator seems to have a memoriry effect in the learning process.

In the opposite case, i.e. the further SSI_{tr} is from 1, this means that $s1$ and its transform rely on different zones for the classification task, leading us to conjecture that in this case, this transform operator seems to help to reveal new discriminant features, that we can call an exploratory effect.

To assess the impact of each operator, we introduce the function $SSI_{tr}(D, op, th)$, which calculates, for a given dataset D and a transformation operator op, the rate of samples in D that have their saliency maps quite similar to the saliency maps of their transformed versions by applying the operator op. We consider that the saliency map of a sample s is similar to the saliency map of its transformation by op if $SSI_{tr}(s, op)$ is higher than a defined threshold th. In this experiment, we choose $th = 0.65$. Algorithm 2 details the $SSI_{tr}(D, op, th)$ function.

Algorithm 2: $SSI_{tr}(D, op, th)$ function

```
function SSI_tr(D, op, th)
    cpt=0
    Foreach (s ∈ D) :
        if (SSI_tr(s, op) > th):
            cpt ← cpt + 1
    return (cpt / |D|)
end function
```

To evaluate the sensitivity of the dataset D for the four operators, namely H, V, C, and c, we applied the Algorithm 2 for each operator. The results are summarized in Table 4.4.

Function	Result(%)
$SSI_{tr}(D, H, th = 0.65)$	29.33
$SSI_{tr}(D, V, th = 0.65)$	35.91
$SSI_{tr}(D, c, th = 0.65)$	87.56
$SSI_{tr}(D, C, th = 0.65)$	61.40

Table 4.4 – Results of the sensitivity of the dataset D to the transformation operators.

From Table 4.4, we observe that for both the horizontal and vertical flip, the saliency map of an image and its transformed version by these geometrical operators are very different, leading to conjecture that these two operators help to discover new relevant distinctive features. And for the color transformations, it seems that they have more of a memory effect on the learning process.

From Table 4.2, we can observe that when geometric operators (that seems to have an exploratory effect) are used alone in Protocol P_{4a}, or color operators (that seems to have a memory effect) are used alone in Protocol P_{4b}, the resulting performance is lower compared to using the variety of the four operators as seen in Protocol P_{4e}e. In an attempt to understand the impact of data

augmentation, we hypothesize that the synergy between operators with a memory effect and those with an exploratory effect is a crucial factor contributing to the success of data augmentation for this classification task.

This chapter addressed the important question of data preparation, by focusing on the informativeness of a dataset, to find the best data augmentation process. For this purpose, new metrics and principles have been proposed and 13 data-preparation protocols were designed to conduct the experiments. We have identified the most suitable data preparation strategy for the BreakHis dataset. As a perspective of this work, we think that it would be interesting to generalize this kind of analysis to other datasets, by automating the steps for selecting the best data preparation in the form of a routine that can be used for any image classification experiment.

We also proposed an interesting formalization of the data preparation, that can be generalized to clarify this process to ensure transparency and reproducibility.

We also proposed a formalization of the data preparation process that can be extended to enhance transparency and reproducibility.

Regarding the metrics used to gauge dataset informativeness, we conclude that there is a need to explore alternative ways of defining such metrics, which should be more closely aligned with the intrinsic features of the dataset and more strongly correlated with the classification task.

In terms of understanding the impact of data augmentation, we presented an intriguing approach that can be generalized and further developed to provide a deeper insight into the influence of transformation operators on classifiers. While these conclusions remain hypothetical and strongly correlated with both the classification task and the dataset. The proposed SSI_{tr} functions represent innovative methods for a better understanding of the effects of data augmentation, thus paving the way for further exploration of this issue.

Chapter 5

How does data feeding impact the classification quality?

Résumé en français du Chapitre 5 : "Comment l'intégration des entrées impacte la qualité de la classification"

Après avoir déterminé à travers l'ensemble des expériences du chapitre 4 la meilleure façon de preparer les données, nous nous intéressons, dans ce chapitre, à l'exploitation de connaissances haut-niveau dans l'ordonnancement de la presentation des données d'entraînement. Ce chapitre est inspiré de l'observation des résultats de l'état de l'art sur la base de données BreakHis, conduisant à une hypothèse sur le lien entre le grossissement microscopique et l'exactitude de classification. Appuyés par une confirmation experte énonçant qu'il est plus simple d'identifier le type tumoral sur une image de faible grossissement, nous intégrons cette connaissance pour ordonner les données d'apprentissage par curriculum incremental. Plusieurs protocoles expérimentaux sont conduits pour déterminer le meilleur ordre et évaluer l'approche.

Contents

5.1 State of the art about curriculum learning 77
5.2 BreakHis data preparation 79
5.2.1 BreakHis data-subsets 79
5.2.2 BreakHis data-preparation algorithm 80
5.3 Experimental protocols on curriculum efficiency for classification 82
5.3.1 CNN architecture and computational details 82
5.3.2 Experimental protocols 82
5.4 Results and discussion 83
5.4.1 Accuracy results 83
5.4.2 Duration of the training period 85
5.4.3 Number of epochs for training stabilization: 86

After exploring the data augmentation process in the previous chapter, in this chapter, we study the impact of using high-level knowledge in the organization of the training set. The key idea of this chapter is inspired by the observation of some relevant state-of-the-art studies (described in Section 3.1.4 of Chapter 3). Drawing upon the accuracy results from these studies, we formulated a hypothesis (see Equation 3.1) suggesting that high-level knowledge on the microscopic magnification of images can be exploited to improve the network's classification performance. This led us to explore the concept of curriculum-based learning in neural networks. Consequently, we conducted experiments that highlight the significance of the order in which data is introduced to the neural network during the training phase.

Indeed, as already mentioned in Chapter 1, neural networks have shown remarkable success in various domains. However, training deep neural networks can be challenging, especially when dealing with complex and large-scale datasets. To address this challenge, researchers have exploited the concept of curriculum learning that mimics the way humans and animals learn by gradually introducing concepts or tasks in a structured and sequenced manner. In nature, humans and animals often learn through a curriculum-like process. They start with simpler tasks or concepts before gradually moving towards more complex ones. This sequential learning allows for the development of skills and the gradual integration of more advanced knowledge. Curriculum learning aims to replicate this natural learning progression in artificial neural networks, thereby enhancing their learning capabilities and generalization abilities.

In this chapter, we leverage this concept to enhance the classification of histopathological breast cancer images by gradually introducing the images according to the order of increasing microscopic magnification.

5.1 State of the art about curriculum learning

According to Bengio et al. [14], experiments showed that humans and animals learn better when the examples are presented in a suitable order. The idea of using this strategy in machine learning can be tracked back to [33] who trained a network to process complex sentences. The training succeeds only when the network begins with limited working memory and gradually increasing memory span. [57] has also used this kind of guided learning to train a recurrent neural network to predict the next word starting from simple sentences towards more and more complex ones. Similar ideas were also explored in robotics [95], by gradually raising the task difficulty.

Bengio et al. [14] have introduced the term of "Curriculum learning" to define this process of learning gradually from simple examples towards more and more difficult ones. Several experiments are presented in this article. The first one is about training a support vector machine (SVM) on a dataset made of randomly generated points in the presence of more and more noise. In another experiment, a Perceptron is trained to classify generated pairs (x, y) where y is a function of x with more or less additional pairs (x', y') where x' is irrelevant to y' (the more irrelevant pairs the more difficult the learning task). They analyzed three training strategies: curriculum (the examples are ordered by increasing difficulty), anti-curriculum (the examples are ordered by decreasing difficulty), and lastly the examples were ordered randomly. The results showed that the curriculum strategy generates fewer test errors. A last experiment concerning shape classification uses two datasets "BasicShapes" and "GeomShapes" that contain respectively regular shape images (circles, squares, and equilateral triangles) and not necessarily regular shapes images (ellipses, rectangles, triangles). The first dataset being less variable than the second one is considered "easier to learn". The curriculum

consists of a 2-step schedule: first perform the gradient descent with the BasicShapes then perform it with the GeomShapes training sets. The best generalization is obtained when the model spends half of its training epochs on the easiest examples and then moves to the hardest examples of "GeomShapes". The last experiment was about training a recurrent neural network to predict the most suitable word that will occur after a sequence of words. The results showed that learning gradually with sentences that are restricted to a limited vocabulary of 5000 words and progressively integrating sentences using more and more vocabulary (increasing the vocabulary by steps of 5000 words) gives better results than feeding the network with all the sentences of the training set directly from the beginning.

Inspired by these results, several works have adopted the curriculum strategy for their network training. In [45], curriculum learning is defined by increasing progressively the difficulty of the samples in each training batch. They propose two definitions of the *difficulty* associated to a sample: 1) the number of neurons activated at the penultimate layer.

2) the confidence degree of the classification (the lower the confidence the more difficult the sample). In [42], a neural network is also used to estimate the difficulty of a given task relative to the neural network "learning progress" and this indicator is then used to select the next samples to learn.

Note that the expression "Incremental" learning is also used in machine learning fields like decision trees [118, 97], decision rules [73], SVM [31], for characterizing the learning task done with data that arrives continuously over time. Note also that the term was used in the context of neural networks research, to allow the neural network to accommodate new arriving data and also new additional classes. For instance, [20] and [117] are using incremental learning in the context of fuzzy systems (systems where the frontiers of the classes are not clear-cut, since some items may be associated with different classes as time goes by). It has also been used by [16] which adapts the network with new heterogeneous remote sensing data and with the possibility of having evolving classes. [21] are addressing the problem of incremental learning that occurs when new data concerns new classes that were not present before, which could bring bad performances to the network. In order to handle this problem [21] proposes to feed the network with new data concerning a new class accompanied by data associated with old classes. For all these works, the "incremental" aspect aims at enabling the network to integrate new data coming from the changing environment and to discover new classes.

The aim of this chapter is to explore and check whether "curriculum incremental learning" enhances the recognition rate. For this purpose, we describe a set of experiments that take into account the ordering of the training phase. The training starts with the VGG19 neural network and then learns sets of images of different magnifications with their labels. These sets of images are issued from the BreakHis dataset, the CNN is fed with a sequence of patches of images ordered by magnification either in ascending, descending, or random order. We also made a control experiment with a set of images with magnifications that were randomly mixed.

The main originality of this approach is that, to the best of our knowledge, no studies have been done that exploit the inherent hierarchical ordering of image magnifications and the way these are fed to a CNN. The second originality of our work concerns the data preparation: we provide a clear and precise description of how the data are taken from the BreakHis dataset and we have re-implemented all the approaches of the literature in order to build a sound comparison. This allows us to provide a re-usable benchmark for future studies.

Based on the state-of-the-art works, presented in Chapter 3, about the approaches using BreakHis, this chapter is designed to confirm the hypothesis 3.1 (stated in Chapter 3) by using the curriculum learning strategy. The main idea is to incrementally train our model on the images, beginning with those of low magnifications, and gradually introducing the remaining images. Hence, in our approach, we define the difficulty of a sample by extra-information, namely the magnification (meaning that a more detailed image is considered as more difficult to learn), moreover the curriculum is done by training with more and more difficult sets of patches instead of increasing the difficulty inside each patch.

Also, by scrutinizing the most recent studies, we noticed that, as far as we know, none of these works gave a precise explanation about their data-preparation process (neither explains how they split the data into train, validation, and test, nor the data augmentation details), and none of them give access to the test set used. These shortcomings made the results difficult to reproduce, and therefore difficult to compare. A second drawback of most of these approaches (see e.g. [124, 70] [91]) is that the data-preparation may generate duplicate samples, for instance, images are augmented by applying two distinct operators: they are flipped vertically and they are rotated 180° (resulting in producing the same image twice!).

To address these limitations, in the next section, we explicitly describe our data preparation. The third section is dedicated to the implementation of the curriculum incremental learning and the experimental study.

5.2 BreakHis data preparation

Data preparation consists of splitting and augmenting the dataset in order to learn correctly as well as to validate the approach. As recalled in Chapter 1, the dataset is first split into a training set, which will be augmented, a validation, and a test set.

5.2.1 BreakHis data-subsets

In the BreakHis dataset, the benign and malignant categories are unbalanced: for instance, in Table 3.1 we can see that concerning 40X magnification, there are 625 samples for the Benign category and 1370 for the Malignant one, the Malignant category is the major class with an imbalance ratio IR over the minor class of $1370/625 = 2.19$, denoted IR). Moreover, among the benign types Fibro Adenoma (F) is the predominant subtype while Ductal Carcinoma (DC) is the predominant one among the malignant ones.

In order to deal with these unbalanced datasets, we opted to over-sample the marginal classes, because in the medical field, especially in oncology, the importance is not to detect one particular type rather than another one, but to identify correctly each category which should thus be well represented. Besides, we are dealing with small [1] datasets of around 2000 images per magnification (see Table 3.1), hence augmenting the number of samples is highly required.

We can distinguish the different parts of BreakHis datasets regarding the number of times they should be duplicated to obtain the same size as the majority category. This is done by quadrupling or octupling some sub-classes since according to the repartition of the samples presented in Table 3.1 the majoritarian sub-class (the Ductal Carcinoma DC) is roughly eight times bigger than the smallest sub-classes (A, TA, PT, LC, PC) and roughly four times bigger than (F). For the sake of generality,

1. BreakHis dataset is small compared to other works, for instance, VGG19 has been trained over millions of images.

the majoritarian sub-class is denoted M, the union of the samples that need to be octupled is denoted o and the ones to be quadrupled q.

Definition 5.2.1: BreakHis data-subsets

The different parts of BreakHis are named as follows:

— $X_i = L_i \cup V_i \cup T_i$ for $i \in \{$ 40X, 100X, 200X, 400X $\}$: X_i is the set of samples with the magnification i. X_i is split into three subsets: one for the training phase of the learning L_i, one for validation V_i, and the last one for testing T_i: $(L_i, V_i, T_i) = split(X_i, (70, 10, 20))$

— M_i: the set of samples of L_i that are in the major class. Here the majority category is Ductal Carcinoma (DC).

— q_i: the set of samples of L_i that belong to marginal categories that should be quadrupled (imbalance ratio ≈ 4). It covers the subtypes F and MC.

— o_i: the set of samples of L_i that should be multiplied by 8 (imbalance ratio ≈ 8). It contains A, TA, PT, LC, and PC subtypes.

5.2.2 BreakHis data-preparation algorithm

Thanks to Definition 4.3.2, 4.3.2 and 5.2.1, the balancing and augmentation done on the BreakHis dataset is summarized by Algorithm 3. This algorithm shows the transformations that were chosen, these choices are justified by the previous chapter (it was published in the proceedings of the "Conférence francophone sur l'apprentissage Cap'21" [66]).

The data preparation for binary and multi-class classification is done thanks to a call to Algorithm 3 with the parameters M, q and o as defined by Definition 5.2.1 (for sake of simplicity, the magnification i is omitted). Algorithm 3 aims at balancing the data in order to obtain an equal number of each sub-type (inducing an equal number of benign and malignant items). For the sake of shortness, the algorithm is described by using tr on the union of all the samples that should be quadrupled or multiplied by eight. However, in our implementation, the work has been done separately for each subcategory and each magnification, and for each of them, the augmentation is only done until reaching the threshold corresponding to the size of the majority category (e.g. Adenosis with the magnification 40X has been duplicated 8 times by doing Id, H, V, C, Hc, Vc, HVc, HVCc, and the last operation done on it (HVCc) is performed to reach a total of $864 \times 0.7 = 605$ images in total (since 605 is the size of the majoritarian images of DC 40X used for training)).

Algorithm 3: BreakHis preparation with $L = M \sqcup q \sqcup o$

```
/* Doubling q ⊔ o */
```

$o' \leftarrow tr(o, HVc, 100)$

```
/* Balancing Sub-classes */
```

$$L' \leftarrow \begin{cases} (M \sqcup q \sqcup o) \sqcup o' & /\star\ M + 1 \times q + 2 \times o\ \star/ \\ \sqcup\ \ tr(q \sqcup o \sqcup o', H, 100) & /\star\ +1 \times q + 2 \times o\ \star/ \\ \sqcup\ \ tr(q \sqcup o \sqcup o', V, 100) & /\star\ +1 \times q + 2 \times o\ \star/ \\ \sqcup\ \ tr(q \sqcup o \sqcup o', C, 100) & /\star\ +1 \times q + 2 \times o\ \star/ \end{cases}$$

return (L')

Proposition 5.2.1

After balancing the BreakHis dataset, no image is present twice.

Proof. Due to the fact that $H(HVc(x)) = Vc(x)$ and $V(HVc(x)) = Hc(x)$ and $C(HVc(x)) = HVCc(x)$, L' can be rewritten as follows:

$$L' = \begin{cases} M \sqcup q \sqcup o \sqcup HVc(o) \\ \sqcup H(q) \sqcup H(o) \sqcup Vc(o) \\ \sqcup V(q) \sqcup V(o) \sqcup Hc(o) \\ \sqcup C(q) \sqcup C(o) \sqcup HVCc(o) \end{cases}$$

As seen here, L' does not contain any duplicated instance. □

Remark 2. *The data preparation made in Algorithm 3 diversifies the transformations since it uses eight transformations with two concerning geometrical properties (H,V), one concerning colors (C) and four combinations of geometry and colors (Hc, Vc, HVc, HVCc).*

Proposition 5.2.2

After preparation, the dataset L' is such that

- for all magnification in 40X, 100X, 400X, the binary classification IR $= 1$. For 200X IR $= 1.011$.
- for all magnification in 40X, 100X, 400X, the multi-classification IR $= 1$. For 200X, the worst IR$=1.037$.

Proof. Since the duplication is done according to the threshold given by the majority subcategory, and due to the choices made for the categories to duplicate 8 and 4 times, we get an IR ratio exactly equal to 1 for all the subcategories except for A 200X (with IR=1.009 since twelves images are missing for reaching 896) and TA 200X (with IR=1.037, 864 images are obtained instead of 896). □

Note that some approaches generate samples without checking whether they are duplicates of already generated samples, our implementation pays attention to generating exactly the right number of new samples that are needed with no duplication. Note that any algorithm that produces more samples and then removes the duplicated ones is not as efficient as Algorithm 3 for two reasons: first, it cannot guarantee the built-in balanced property, second, it has a greater spatial complexity.

Proposition 5.2.3

Algorithm 3 has a smaller spatial complexity than any algorithm that generates a duplicate-free set containing the same amount of samples of each class for each magnification.

Proof. Due to proposition 4 and the use of the threshold relative to the majority class, the algorithm generates only the samples that are required without creating duplicated samples. Hence the spatial complexity [2] is only equal to the number of created samples. □

2. The spatial complexity of an algorithm is the space used by the algorithm without considering the size of the input data.

5.3 Experimental protocols on curriculum efficiency for classification

In this section, we first present the neural network architecture and computational details. Next, we design experimental protocols to confirm Hypothesis 3.1 of Chapter 3.

5.3.1 CNN architecture and computational details

In this work, after several attempts, we have selected the VGG19 model [103] as described in Section 1.1.6. Our choice is primarily motivated by the high performance of this network (recalled in Section 1.1.6), and by the fact that it has been widely employed in histopathological breast cancer classification studies.

The model is implemented with the Keras library, using Tensorflow GPU and deep learning libraries. The model is trained with 4000 epochs with a batch size of 128. Adam-optimiser is used with a learning rate fixed initially at 0.0001. The implementation is done on a compute node of the OSIRIM (Observatory of Systems Information Retrieval and Indexing of Multimedia contents) Platform [86] [3]. The source code is available in [62].

5.3.2 Experimental protocols

Let us recall that the training is done on the specific part of the data called L (separated in L_{40X}, L_{100X}, L_{200X} and L_{400X}) of the BreakHis Dataset, and the tests are done on the part called T of BreakHis (see Definition 5.2.1) which is also distributed into four sets according to the magnifications.

We first re-implemented the best approaches of the state of the art namely InceptionV3 and ResNetV2 with the data preparations and hyper-parameters defined in the previous Section. We call these methods *separated* since they learn on each magnification separately. We also implemented a separate version based on VGG19 transfer learning (called *VGG19* in Table 5.1).

We then introduced six approaches in order to verify the magnification impact on the learning process.

— The INCR (for incremental) approach aims at training the model on the grounds of the weights obtained successively for the 40X, 100X, 200X and 400X magnifications as illustrated on Figure 5.2. More precisely INCR trains the network on 40X images then uses the latest weights as starting point to train the 100X images and so on for other images) starting from VGG19 weights.

— The Ctrl_INCR (for control incremental) approach consists in training VGG19 independently with the same set of images as the incremental approaches but not incrementally (first train and test on 40X images then train on a mixed set composed of 40X and 100X images and test on 100X, then train on 40X + 100X + 200X and test on 200X then train on all images and test on the 400X), as shown in Figure 5.1.

— The DECR (for decremental) approach aims at training and testing successively on 400X, 200X, 100X and 40X magnifications.

3. The compute node has the following characteristics: DELL T630 server, 2 Xeon 2640 V4 processors (20 threads), 2 x 400 Go RAID1 SSD disk, 1 x 10 Gb/s network, 4 GPUs Nvidia GTX 1080 TI (3584 cuda cores each, 11 Go RAM).

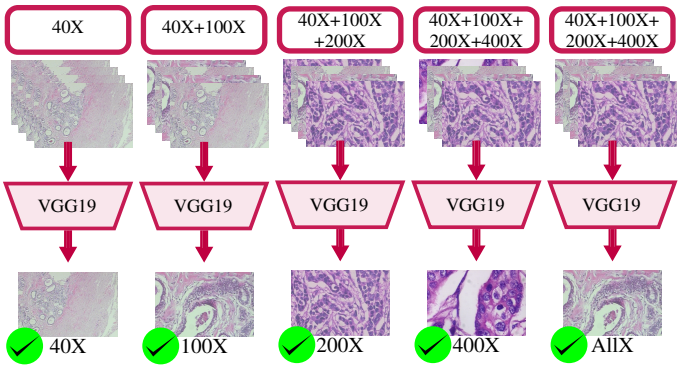


Figure 5.1 – The Ctrl_INCR approach

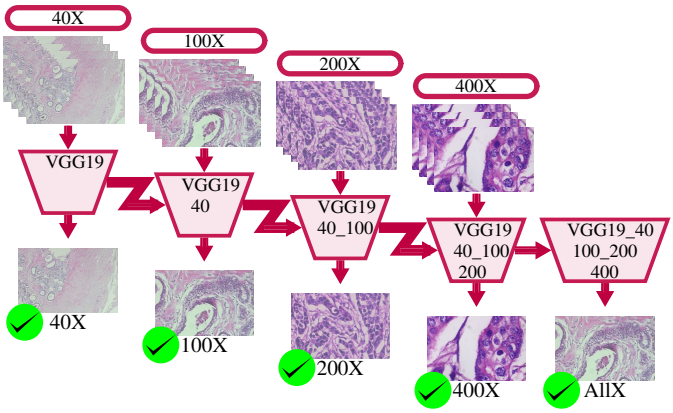


Figure 5.2 – The INCR approach

— The Ctrl_DECR approach consists in training VGG19 similarly to the Ctrl_INCR approach but with the respective appropriate sets of images (400X then 200X + 400X then 100X + 200X + 400X then all).
— The DIS (for disordered) approach aims at training successively on 200X, 40X, 400X, 100X.
— The Ctrl_DIS approach consists in training VGG19 similarly to the two others control approaches with the successive sets (200X, 40X + 200X, 40X + 200X + 400X, AllX).

5.4 Results and discussion

All the results of the experiments are shown in Table 5.1. The main conclusions that follow from these results are that 1) the hypothesis is verified and 2) the best approach is INCR.

5.4.1 Accuracy results

First, the experiment performed on the three separated approaches (VGG19, InceptionV3 and ResNetV1 [128]), where we trained then tested each magnification separately for both binary and multi-class classification are confirming Hypothesis 3.1: the lower the magnification the higher the accuracy.

Second, in most cases, VGG19 has a better accuracy than InceptionV3 and ResNetV1 on BreakHis

	40X	100X	200X	400X	All
Separated approaches: Binary classification					
VGG19	95.90	95.35	**95.30**	**94.98**	**97.05**
InceptionV3	95.68	95.52	94.78	94.22	96.95
ResNetV1	**96.20**	**95.68**	94.25	93.08	96.78
Incremental approaches: Binary classification					
CTRL_incr	95.90	95.96	95.52	96.89	97.05
INCR	95.90	96.23	**97.01**	**98.86**	**98.76**
CTRL_decr	96.82	96.97	95.41	94.98	97.05
DECR	**96.91**	96.06	95.43	94.98	97.38
CTRL_dis	96.17	**97.25**	95.30	96.98	97.05
DIS	96.33	97.06	95.30	97.03	97.11
Separated approaches: Multi classification					
VGG19	94.06	93.63	93.80	93.25	**95.49**
InceptionV3	**95.12**	**94.33**	**93.57**	93.15	95.37
ResNetV1	93.87	93.25	92.89	**93.52**	94.76
Incremental approaches: Multi classification					
CTRL_incr	94.06	94.17	94.71	95.62	95.49
INCR	94.06	95.35	**95.68**	**96.26**	**95.93**
CTRL_decr	95.68	94.91	94.22	93.25	95.49
DECR	**96.03**	94.54	94.40	93.25	95.67
CTRL_dis	95.18	96.07	93.80	95.32	95.49
DIS	95.23	**96.13**	93.80	95.78	95.71

Table 5.1 – Accuracy Results obtained on BreakHis, the approaches are in lines, the datasets are in columns

images. Moreover the 6 new approaches implemented are more accurate than VGG19 (thanks to the benefit of transfer learning from a more suitable field). Lastly, we observe that among the new approaches, the INCR approach, that sequentially feeds images with a larger zoom factor to the network, gives more accurate results with a general test accuracy of 98.76% (enhancing the VGG19 separated approach, and improving the state of the art), it shows also a better behavior than the Ctrl_INCR approach (that trains on the same amount of images but in a random way) and than the DECR and DIS ones. This is summarized by the following rank ordering that holds for Binary and Multi-classification:

$$\text{ResNetV1} <_{All} \text{InceptionV3} <_{All} \text{VGG19} =_{All} \text{Ctrl_}_{*} <_{All} \text{DECR} <_{All} \text{DIS} <_{All} \text{INCR}$$

where $<_{All}$ means more accurate in the test of images of any magnification (last column of Table 5.1).

The fact that INCR is more accurate than Ctrl_INCR may be explained by the fact that performing intra-transfer learning seems to better guide the gradient descent towards the local optima, as opposed to starting from VGG19 weights and exploring a huge amount of data from scratch.

In order to support these results, we used the Layer Wise Relevant Propagation (LRP) technique [75] which selects the pixels (forming a *saliency-map*) that are the most relevant for guiding the classification task. The LRP approach exploits the CNN structure and weights in order to discover this map. We use LRP on several wrongly classified images from the 400X test with CTRL_INCR but correctly classified with INCR, e.g. the image `SOB_M_DC-14-5694400-014` shown on Figure 5.3. This figure shows the initial image on the left, on the center its LRP saliency map obtained

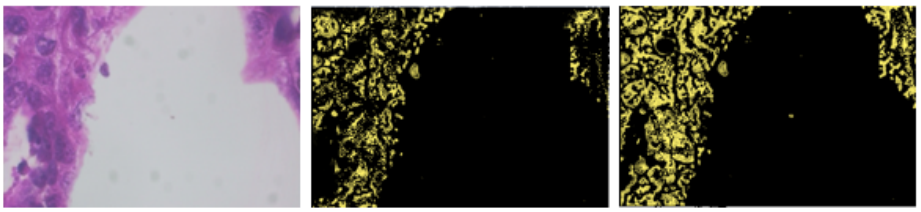


Figure 5.3 – An Image from BreakHis (left), its associated saliency maps obtained with CTRL_INCR (center) and INCR (right)

with the control protocol and on the right, the one obtained with the incremental protocol. We can see that the incremental protocol activates more regions of interests for the classification task than the image obtained by the control protocol. This experiment reinforces the idea that the network is learning better with the curriculum strategy.

Let us recall that a fair comparison with state of the art approaches was impossible, since as mentioned above, most of the state of the art proposals did not precise neither their data preparation nor the parameters of their network, nor the size and nature of the test set. Besides, none of the studies cited in Table 3.2 made experiments covering both the classification on the datasets of the four different magnifications separately and the classification on the whole dataset. This is why we re-implemented the two architectures that gave the best results in [77], namely InceptionV3 and ResNetV1. The accuracy rate obtained in table 5.1 demonstrates that our choice of VGG19 network and our incremental curriculum strategy gave better results than the two other architectures. Note that 3.2 shows that [77] obtained an accuracy rate of 99.2% in the multi-class classification of an image of any magnification. However, in this work, an additional external dataset has been added to supplement the lack of data.

One main goal of this work was to experiment the use of curriculum learning based on magnification knowledge in the cancer type and sub-type classification. The experiment has not only demonstrated the benefits of a guided feeding but also showed that our approach has outperformed the state of the art results.

5.4.2 Duration of the training period

Table 5.2 represents the computational time taken by the training process of the different protocols. Note that the training does not stop directly after stabilization since every protocol is systematically trained on a fixed number of epochs (4000 epochs for each magnification training set and 16000 epochs for the whole training set) for sake of fair comparison.

The average time taken for the training phases of the whole dataset (column All) is around 30 minutes and 59 seconds for the binary classification and around 31 minutes and 54 seconds for the multi-class classification. We note that for the separated approaches of the literature where each dataset is learned independently, VGG19 is the fastest network for both binary and multi-class classification for every magnification, namely the training time for the whole set is 30mn18s for the binary classification and 31mn02s for multi-class classification. This can be explained by the fact that InceptionV3 and ResNetV1 are deeper networks (with respectively 48 and 159 layers compared to 19 layers for VGG19) with complex intermingled blocks (contrarily to VGG19 which

	40X	100X	200X	400X	All
Separated approaches: Binary classification					
VGG19	07:23	08:05	07:56	07:02	30:18
InceptionV3	08:49	8:52	09:06	08:12	34:12
ResNetV1	09:11	09:40	09:33	08:57	38:03
Incremental approaches: Binary classification					
CTRL_incr	07:23	15:20	22:07	30:18	30:18
INCR	07:23	14:22	20:25	28:09	**28:09**
CTRL_decr	07:02	15:38	23:58	30:18	30:18
DECR	07:02	15:27	24:21	31:09	31:09
CTRL_dis	07:56	15:34	23:13	30:18	30:18
DIS	07:56	15:49	22:58	32:05	32:05
Separated approaches: Multi classification					
VGG19	7.29	08:11	08:02	07:05	31:02
InceptionV3	08:55	09:13	9:15	8:24	35:28
ResNetV1	9:19	09:52	09:47	09:20	40:05
Incremental approaches: Multi classification					
CTRL_incr	7.29	15:33	23:02	31:02	31:02
INCR	7.29	15:12	22:05	29:57	**29:57**
CTRL_decr	07:15	15:46	24:02	31:02	31:02
DECR	07:15	15:39	24:36	31:37	31:37
CTRL_dis	08:02	15:34	23:13	31:02	31:02
DIS	08:02	16:23	23:02	32:18	32:18

Table 5.2 – Computational training time in (minutes:seconds). The training is done with 4000 epochs for each magnification set and 16000 epochs for the whole training set, the approaches are in lines, the datasets are in columns

is sequentially organized).

Concerning the experiments done on curriculum learning, we can see that in the binary classification, the incremental approach gains *124 seconds* (which represents 6% of the training time) over the control approach and *65 seconds* in the multi-class classification. Also, the incremental approach reduces the training time compared with the decremental approach and the disordered approach.

5.4.3 Number of epochs for training stabilization:

To confirm the claim of this study that INCR is the best approach, we compare also the number of epochs for training stabilization (StbE described in Section 4.6). As said in the previous paragraph, the learning was not stopped at the stabilization epoch, but the training was done for a fixed number of epochs. However, the stabilization epoch is very useful for comparing the network's behaviors. We give here only two remarkable numbers: the stabilization epoch for INCR with the whole dataset is 12313 while the one of DIS approach is 14659 given that these curriculum approaches already require 12000 epochs to load the whole dataset (there are 4000 epochs to obtain network N0 then 4000 for obtaining the first network's weights for the first magnification package, and so on, hence 12000 epochs are required to obtain the final network that will be fine-tuned on the last magnification images package). We note that only 313 supplementary epochs are necessary to reach stabilization for the INCR approach, while the additional epochs required by DIS are 1659. So the stabilization time of the INCR approach is 5 times faster.

To sum up, there are three novelties in this chapter: the new way to make the data preparation explicit and reproducible (the implementation and the datasets are available in [62]), the use of magnification as a criterion for guiding the curriculum learning process, the proof of concept on the Breakhis dataset.

In order to check our hypothesis, we have implemented eighteen approaches including the algorithms found in the literature. This implementation was done for a fair comparison of the results with the same data preparation. The experiments confirm that *starting to train a network from the lowest magnification images towards the highest magnification* gives a more accurate network than the same one trained randomly. This technique of guided CNN feeding may be very useful for other domains such as multi-resolution satellite images or continuously-changing environment images.

The ideas developed in our study can be exported to other domains thanks to the formalization provided in Section 5.2. Concerning curriculum learning, it has already been applied in different fields (see Section 5.1) and our study emphasizes the benefits of using high-level knowledge as a curriculum criterion, especially in fields where images are available with different magnifications.

We also notice that there is a limitation concerning the BreakHis dataset itself since in this dataset, the same area is not available in the different magnifications. Indeed, contrary to what could be expected from the encoding of the names of the files in BreakHis, there is no link between two images with the same "slide-id" (also called "patient id") and the same sequential number. For instance, the images called `SOB_B_A-14-22549AB-40-007.png` and `SOB_B_A-14-22549AB-100-007.png` are named as if they were magnification X40 and X100 taken from the slide 22549 area 007, but in fact 007 is a random number, hence it does not imply that the two images cover the same area. This observation gave birth to doubts about the correctness of the existing labeling of BreakHis since all the images of the same slide-id have the same cancer-type label. This leads us to believe that the label of some images could have been inferred instead of obtained from an expert. This doubt about BreakHis dataset labeling is increased when one considers that some works use the label of an entire image for labeling its subarea (while a tissue image may contain normal, malignant, and benign regions at the same time).

These issues underline the need for benchmarks with clearly defined test data sets. Our proposal may be considered as a first step towards building the criteria for the certification of such benchmarks.

Coming back to the second scientific question of this work, namely, "is magnification a good criterion to guide curriculum learning ?". We have only shown that this is the case for the precise application of cancer classification with the BreakHis dataset, further experiments are required to show the validity of these criteria in other domains.

A perspective of this work is to explore another technique of guided learning, that we could call "focused feeding". The idea is to propose a quadruple (because we dispose of four magnification types) Siamese convolutional neural network where four parallel networks are joined by a common fully connected layer. Each Siamese network receives an image with a specified magnification of the same area. This architecture could bring more accuracy by benefiting from several magnified views of the same area. Moreover by using layer-wise relevant propagation [9] we could discover which parts of the images with which magnification were the most informative for the decision task.

Note that this perspective can only be done with an appropriate dataset. However, [58] could answer this limitation since they propose to use a GAN in order to generate ultrasound super-resolution images (SR) of good image quality wrt to LR images (in terms of image similarity metrics, infor-

mation fidelity criteria, and visual effects). The use of the GAN ensures that a re-degenerated image of the generated SR one is consistent with the original LR image, and vice versa. Hence, [58] opens an interesting perspective on the possibility of generating images of distinct magnifications of the same area starting from one image of this area.

Through this part, we explored the use of expert knowledge at the input level of the neural network. We presented significant contributions in the field of data preparation. First, we introduced new metrics to evaluate data diversity, and through carefully designed experimental protocols, we gained insights into the mechanisms of data augmentation through an approach that highlights the relevant supplementary features learned from the transformed images. Additionally, we address existing limitations in data preparation transparency and propose a framework that ensures fairness and efficiency in the process.

Furthermore, after an extensive review of the state-of-the-art literature, we identify a remarkable hypothesis that establishes a connection between image magnification and classification accuracy. This discovery leads us to explore the curriculum learning strategy, where we leverage high-level information about image magnification to order the training set accordingly.

The results of our experiments strongly confirmed the positive impact of ordering and highlighted the importance of using high-level knowledge during the data preparation phase. These findings underline the significance of our contributions to the progress in characterizing efficient data preparation for classification.

Part III

Hierarchical knowledge used for designing the learning engine

Dans cette partie, nous nous intéressons à un type spécifique de classification pour lequel des connaissances de haut-niveau supplémentaires sont fournies. Plus précisément, nous traitons des données multi-étiquetées pour lesquelles un chemin hiérarchique relie les étiquettes. Ce type particulier de classification est connu sous le nom de "classification hiérarchique multi-étiquettes" (HMC). Il s'agit d'une tâche complexe qui implique l'encodage de contraintes de haut-niveau dans le modèle de réseau neuronal. Plus précisément, le problème HMC est une extension du problème de la classification multi-labels (MLC) où les étiquettes prédites doivent respecter les contraintes hiérarchiques existant dans l'ensemble de données. Avant l'essor du domaine HMC, la MLC était réalisée sans prendre en compte les liens hiérarchiques entre les données. Pourtant, cette information supplémentaire reliant les classes et les sous-classes serait utile pour améliorer les performances de la classification. Récemment, certains travaux ont intégré cette information hiérarchique en proposant de nouvelles architectures de réseaux de neurones (appelées Branch-based CNN (B-CNN) ou Hierarchical-based CNN (H-CNN)), obtenant des résultats prometteurs. Cependant, avec ces architectures, le réseau est séparé en blocs où chaque bloc est responsable de la prédiction de la classe d'un niveau donné de la hiérarchie.

Pour améliorer ces solutions et tirer un meilleur parti de l'ensemble du réseau, nous proposons une nouvelle architecture dans laquelle toutes les couches du réseau participent simultanément à la prédiction de toutes les étiquettes d'un échantillon, c'est-à-dire de la classe située au niveau supérieur de la hiérarchie à la classe située au niveau inférieur. En plus de traiter le problème HMC, nous proposons un mécanisme d'abstention qui permet au réseau de s'abstenir de fournir sa décision de classification à partir d'un certain niveau si sa confiance est inférieure à un seuil défini. Cette partie contient trois chapitres : un premier chapitre formalisant le problème HMC, où les concepts de base sont introduits, un deuxième chapitre présentant notre proposition pour traiter le problème HMC, et enfin un troisième chapitre décrivant le mécanisme d'abstention.

Les principaux objectifs de cette partie peuvent être résumés comme suit :

- Établir un ensemble de définitions et de notations pour traiter les problèmes de HMC et d'abstention.
- Intégrer les connaissances hiérarchiques dans les réseaux neuronaux :
  - Concevoir une nouvelle architecture de réseau neuronal (en étendant la littérature existante).
  - Définir des fonctions de perte spécifiques adaptées à la nouvelle architecture.
- Concevoir un mécanisme d'abstention basé sur des informations hiérarchiques afin d'accroître la robustesse du système.
- Introduire de nouvelles mesures d'évaluation pour quantifier l'efficacité de l'abstention.

In this part, we are interested in a specific type of classification for which additional high-level knowledge is provided. Specifically, we are dealing with multi-labeled data for which a hierarchical path is linking the labels. This particular type of classification is known as "hierarchical multi-label classification" (HMC) which is a challenging task involving the encoding of high-level constraints into the neural network model. More precisely, the HMC problem is an extension of the multi-label

classification (MLC) problem where the predicted labels must respect the hierarchical constraints existing in the dataset. Prior to the rise of this HMC field, MLC was done without taking into account the hierarchical links between the data. Nevertheless, this extra information linking classes and subclasses is useful to improve classification performance. Recently, some works have integrated this hierarchical information by proposing new neural network architectures (called Branch-based CNN (B-CNN) or Hierarchical-based CNN (H-CNN)), obtaining promising results. However, with these architectures, the network is separated into blocks where each block is responsible for predicting the class of a given level of the hierarchy.

To improve these solutions and take better advantage of the whole network, we propose a new architecture in which all layers of the network are involved in the prediction of all the labels of a sample, i.e., from the class at the top level of the hierarchy to the class at the bottom one. In addition to dealing with the HMC problem, we propose an abstention mechanism that allows the network to abstain from providing its classification decision from a certain level if its confidence is lower than a defined threshold.

This part contains three chapters: a first chapter formalizing the HMC problem, where the basic concepts are introduced, a second chapter presenting our proposal to deal with the HMC problem, and finally a third chapter describing the abstention mechanism.

The main objectives of this part can be summarized as follows:

— Establish a set of definitions and notations for dealing with HMC and abstention problems.
— Integrate hierarchical knowledge in neural networks:
  — Design a new neural network architecture (extending the existing literature)
  — Define specific loss functions to suit the new architecture.
— Design an abstention mechanism based on hierarchical information to increase the robustness of the system.
— Introduce new evaluation measures to quantify the abstention efficiency.

Chapter 6

Formalising Hierarchical classification

Résumé en français du Chapitre 6 : "Formalisation de la classification hiérarchique"

La classification multi-labels hiérarchique est une tâche critique, imposant des contraintes hiérarchiques entre les étiquettes des classes, où il n'est pas juste question de bien assigner une image à son ensemble de classes, mais aussi que cet ensemble soit en accord avec la hiérarchisation des labels.
Dans ce chapitre, nous explicitons formellement le cadre de la classification multi-labels hiérarchique. Nous commençons par énoncer les travaux phares de la littérature ayant traité de la classification hiérarchique à l'aide des réseaux de neurones convolutifs. Ensuite, nous présentons une formalisation des principales notions et principes nécessaires à la mise en place de notre solution. Nous donnons ensuite une interprétation de ce type de classification dans un cadre probabiliste. Nous clôturons ce chapitre par l'identification et la formalisation de deux contraintes principales régissant cette problématique.

Contents

6.1 State of the art about hierarchical classification 94
6.1.1 Hierarchical adjustment 94
6.1.2 Branch-based CNN approaches 95
6.1.3 An example of local approach 96
6.1.4 Loss functions integrating hierarchical constraints 96
6.2 Principles and notations about hierarchical dataset labeling 97
6.3 The probabilistic view of HMC classification 99
6.4 The two constraints of a hierarchical classifier 100

As mentioned in Chapter 1, classification is a crucial task in everyday life, it is one of the primary and primordial faculties learned by an agent to discover its environment, and in certain animal species, it remains a vital task on which depend their survival and perennity [105]. A smarter task is hierarchical classification since it involves high-level structural knowledge. Indeed, hierarchy is present in many fields of applications (the reader can refer to the survey [102] for more details) such as:

— document classification: [22] designed an automatic document classifier by using hierarchies like e.g. USPatent dataset, the authors underlined the fact that hierarchical views of text databases can improve search and navigation for a user;

— speech recognition: the hierarchical organization of emotions can help to distinguish better some confusing emotions such as boredom and sadness, see, e.g., the Berlin Emotion Database (EMO-DB) [18];

— musicology: in genre recognition, several musical genres are hierarchically organized and this hierarchy plays a capital role in the classification, see e.g. the dataset constructed in [19];

— chemistry: in protein functions classification, the protein functions are naturally organized into hierarchies based on EC nomenclature [1], e.g. ExplorEnz is one of the largest hierarchical enzyme databases [68];

— genomics: in genetic disease, the genetic tree is a key piece of hierarchical information, see e.g. the catalog of Human Genes and Genetic Disorders[6]

— geometry: a semantic meaning can be assigned to geometry by using an existing class hierarchy, e.g. Princeton Shape Benchmark for 3D shape classification [100].

In this study, we adopt the point of view defended by Silla and Freitas in their survey [102] which defines hierarchical classification as the process of doing classification under the guidance of a pre-established taxonomy, in the context of supervised learning. For instance, hierarchical classification is a particular case of structured classification where extra information is available about the classes organization which can be a tree (like a taxonomy), a forest (like an ontology), or any other graph. The difficulty of the hierarchical classification is greater when the classes are organized into a DAG(Directed Acyclic Graph) structure than when they are in a tree structure (each node can have only one parent).

The approaches of hierarchical classification can be distinguished according to the depth at which it is performed. For instance, some works always classify at the level of leaf nodes, this kind of approach is called *mandatory leaf-node prediction* (MLNP), on the other hand, some approaches classify at any level of the hierarchy: *non-mandatory leaf-node prediction* (non-MLNP). In non-MLNP approaches, a sample can be assigned to a label at any level in the hierarchy, it is often done by using a confidence threshold under which no further digging inside the sub-levels of the classification is done. This categorization is refined according to the precise aim of the classifier and the way it exploits the hierarchy:

— *flat classifiers* are MLNPs that only aim to give the leaf class, then the hierarchy may be used a posteriori to deduce all the implicitly assigned ancestors. The drawback of this kind of approach is that it requires discriminating among a large number of classes (the leaves of the taxonomy), moreover, the hierarchy is not used to guide the learning.

— *local classifiers* are non-MLNPs that aim to give the class of the input at a specific level, they

1. EC stands for Enzyme Commission numbers which is a numerical classification of enzymes, based on the chemical reaction they catalyze.

are using the predicted upper class to narrow the choices of the current class (they are also called top-down approaches). A disadvantage of these approaches is that an error at a given level will propagate to the sub-levels.

— *global classifiers* are non-MLNPs that provide as an output, the entire path of classes in the hierarchy from the root class to the leaf instead of only the leaf (i.e. the finer class).

In order to overcome the limitations mentioned above, this chapter is dedicated to introducing the basic notations that allow us to express rigorously in an abstract way any hierarchical classification problem. The purpose of this chapter is to set up rational criteria to guide the construction of solutions to the hierarchical classification task. After summarizing some relevant state-of-the-art works (Section 6.1), we describe the formal framework that we have created for introducing the notations used in all this part (Section 6.2), then we formalize two constraints imposed to any hierarchical classifier (Section 6.4).

6.1 State of the art about hierarchical classification

As already mentioned in the head of this part, Hierarchical multi-label classification (HMC) aims at classifying objects with a set of labels that respects a set of given hierarchy constraints. In HMC, the classes of objects are organized as a tree where the edges correspond to superclass-subclass links. In this section, we review some works dealing with hierarchical classification, we exclude MLNP approaches because they do not exploit the hierarchical knowledge (they can be considered as mere MLC). In this state-of-the art, the considered non-MNLP approaches are divided into four families. The two first families are global approaches: the *hierarchical adjustment* approaches and the *branch based* CNN approaches. Hierarchical adjustment approaches are modifying the output of the superclass according to the one of the subclass. Branch-based CNN are using specific architectures to handle the hierarchical concept. The third family contains *local* approaches that use a cascade of classifiers (one per category) trained in a sequential workflow. The last family covers non-MNLP approaches that are neither global nor local since they translate the hierarchical constraints inside the *loss* function. Note that it is not a typical HMC approach since this mechanism may be applied to constraints that are not necessarily hierarchical.

6.1.1 Hierarchical adjustment

A natural idea was proposed by Giunchiglia and Lukasiewicz in [36], it consists in imposing a hierarchical constraint by adjusting the output according to the hierarchy. Their solution is based on enforcing *inclusion between the objects of a class to its superclass*: if an object is assigned to a category A, it should also be assigned to its supercategory B ($A \subseteq B$). For that, they adjust the output z_B of the superclass B wrt the output $\hat{y}_A$ of A, by defining $\hat{y}_B = \max(z_B, \hat{y}_A)$. The final loss function is a sum of the loss function concerning the output $\hat{y}_A$ and the loss function concerning the adjusted output $\hat{y}_B$.

A benefit of [36] is to impose the respect of the hierarchy (by enforcing each super-class prediction to be consistent with its subclass prediction). A drawback is that this approach is using a sigmoid function which may allow to have several predicted classes by level (no exclusiveness per level).

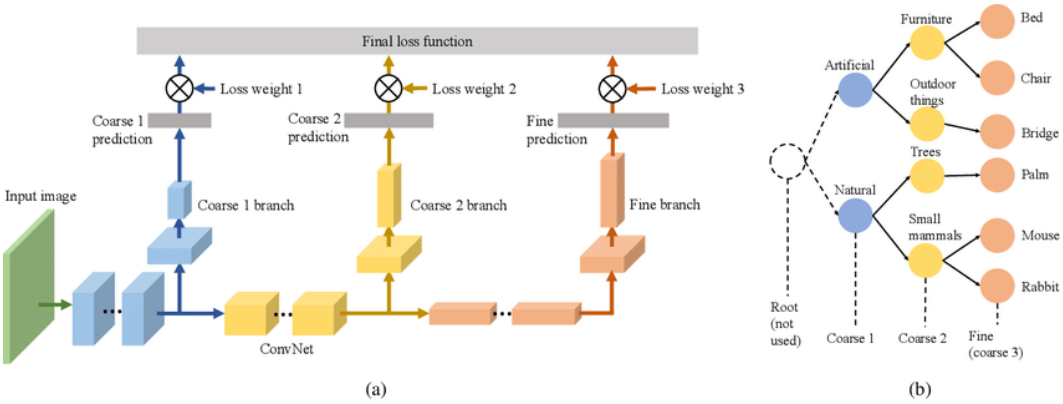


Figure 6.1 – The B-CNN architecture described in [136]

6.1.2 Branch-based CNN approaches

In the field of branch-based CNN approaches, Zhu and Bain [136] were the first to propose a neural network architecture, called B-CNN, to support the hierarchy constraint. The B-CNN for "Branch Convolutional Neural Network (B-CNN)" is a CNN with a particular architecture where the first layers are dedicated to the prediction of coarse classes and the last layers to the prediction o fine classes, according to a given hierarchical structure of the target classes. The predictions of the different hierarchical levels are then aggregated with a weighted sum of the loss functions associated to each of them. Moreover, the learning phase is done by following a curriculum incremental strategy (as the one we proposed in [63], see Chapter 5) consisting in successively learning coarse to fine concepts. The authors' experiments show that B-CNN improves over the corresponding baseline CNN on the benchmark datasets MNIST, CIFAR-10, and CIFAR-100 (See [1] for more details about these two last datests). Their architecture is shown in Figure 6.1.

Similarly, Seo and Shin [98] are using hierarchical classification for recognizing and classifying people's clothing in apparel images. Their proposal is a VGG19 architecture with additional intermediary outputs: the network is able to give three predicted values for a given sample: one at the top level of the hierarchy, which is the coarsest category, one at the middle level and the last one at the bottom, which is the finest level (these levels were respectively named "coarse 1", "coarse 2" and "fine" by Zhu and Bain in Figure 6.1).

In the approach of Kolisnik et al. [54] a new architecture, called H-CNN, is introduced based on B-CNN and designed on VGG16 model to classify the images of Kagfgle Fashion Image dataset under hierarchical constraints. The model is an extension of the solution proposed in Seo and Shin [98] which separates the neural network into connected blocks where each block is responsible for predicting a class at a given level. The novelty of this study is the computation of conditional probabilities by progressively updating a Conditional Probability Weight Matrix (CPWM) in order to obtain the adjusted probabilities of the subclasses knowing the probabilities of the superclasses. Note that, the conditional probability update was previously used by Phan et al. [89] to highlight the relationships among diseases in classification of chest X-rays, and also by Taoufiq et al. [113] for urban structure classification. The results of H-CNN are promising and enhance the accuracy of fine-classes prediction compared to a simple model and to a B-CNN model without conditional probability adjustment.

Note that the main goal of these approaches is to refine the prediction of fine classes, which

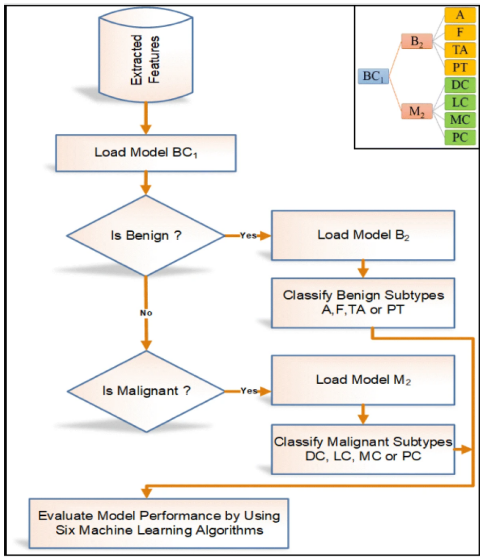


Figure 6.2 – The hierarchical classifier of [79].

guarantees neither respect for the hierarchy nor the improvement of the prediction accuracy of superclasses. One benefit of these approaches is that, as the neural network is divided into blocks, the learning parameters for the network as a whole can be reduced.

6.1.3 An example of local approach

We restrict our presentation for illustrating local classification to only one approach, namely [79], because it is the only hierarchical approach (as far as we know) experimented on BreakHis (which is one of the datasets we are interested in). Murtaza et al. [79] build three classifiers: a binary classifier for predicting whether the tissue is benign or malignant, a multi-class classifier for the benign subclasses (A, F, TA, PT) and a second multi-class classifier for the malignant ones (DL, LC, MC, PC). The architecture is a cascade network where the output of the binary classifier guides the choice of the second classifier to use.

The workflow of their approach is shown on Figure 6.2: first, the first part of the network (called BC_1 in this diagram) is trained for labeling the super-class (benign or malignant), then based on this prediction, a choice is made about loading either the network B_2 if the superclass is benign or the network M_2 otherwise. The training is thus done in two separate stages.

The approach is a local approach in the taxonomy of Silla and Freitas [102], and even if it allows for some predictions mistakes, there is no hierarchy violation by construction since they use the prediction of the main class in order to select the specialized network that will give the subclass in a following stage.

6.1.4 Loss functions integrating hierarchical constraints

As a third part of this state of the art, we expose works dealing with semantic loss functions. Lots of articles are dealing with the use of a semantic loss function for translating high-level knowledge. Among them are the work of Xu et al.

which proposes to integrate a Boolean logical constraint into a loss function, called semantic

loss function. The article focuses on constraints expressing the *exclusive membership to a unique class* while using a layer of sigmoid activation functions. More precisely, for each sample s_0, the network provides a vector of n probabilities $\hat{y} = (\hat{y}_1, \ldots, \hat{y}_n)$ where $\hat{y}_k$ represents the predicted probability that the variable X_k is true for the input s_0. The semantic loss function $Loss^s$ (where s stands for "semantic"), associated with the constraint, named α, saying that only one variable among $X_1, \ldots, X_n$ should be true (denoted $x \models \alpha$, moreover the fact that the precise variable is X_k is denoted by $x \models X_k$), is defined by:

$$Loss^s = -\log \sum_{x \models \alpha} \prod_{k:x \models X_k} \hat{y}_k \prod_{k:x \models \neg X_k} (1 - \hat{y}_k)$$

In other words, let x be a vector with a single 1 and 0's everywhere else (representing what the ideal output would be), let us say that the 1 is at position k in x, then the value $\hat{y}_k$ obtained by the sigmoid is multiplied by all the $1 - \hat{y}_i$ for all $i \neq k$, this represents the discrepancy between the obtained vector and an ideal one. The semantic loss $Loss^s$ is the sum of this computation for any possible ideal vector x (i.e., any x s.t. $x \models \alpha$).

Example 4. *Suppose that we have a vector x of dimension 2. The two ideal possible vectors for x are (1,0) and (0,1) (i.e., satisfying that only one variable is true)*

- *Suppose that $\hat{y} = (0.7, 0.8)$ the loss function defined above gives: $Loss^s(\hat{y}) = -\log(0.7 * 0.2 + 0.8 * 0.3) \sim 0.42$*
- *Suppose that $\hat{y} = (1, 1)$. It leads to $Loss^s(\hat{y}) = -\log(0) \sim +\infty$*
- *As expected if $\hat{y} = (1, 0)$ then $Loss^s(\hat{y}) = -\log(1 * 1 + 0 * 0) = 0$*

This example allows us to understand the behavior of this loss function which penalizes the violation of exclusive membership to a unique class. This paper presented a new way to translate the fact that there is only one output value into a Boolean constraint, and this encoding can be generalized to represent any logical constraint the predicted classes of a sample.

6.2 Principles and notations about hierarchical dataset labeling

In this section, we introduce the notations that we will adopt to handle the HMC problem. We consider a dataset $D = \{s_1, s_2, ..., s_n\}$ of n samples, with a hierarchy of labels $\mathscr{C} = \mathscr{C}^1 \cup \mathscr{C}^2 \cup ... \cup \mathscr{C}^C$, the labels are organized in a tree of depth $C \in \mathbb{N}$, where the more general labels are the nodes of the first stratum (or level) $\mathscr{C}^1$ and the most specific ones are the leaves (or terminal nodes) which are situated in the last stratum $\mathscr{C}^C$. The set of classes in a strata is called *a category*. Each stratum $\mathscr{C}^i$ is composed of a number $\mathscr{N}_i$ of classes: $|\mathscr{C}^i| = \mathscr{N}_i$ and $\sum_{i=1}^{C} \mathscr{N}_i = |\mathscr{C}|$. The classes are *uniquely identified* by two numbers: the index i of the level and the absolute index j of the class in this level ($j \in [1, \mathscr{N}_i]$): c_j^i denotes the jth class of the ith hierarchy level. The hierarchical relations between classes are described by two functions ch (for children) and pa (for parent) where $ch(c_j^i)$ gives the list of the indexes of the classes of level $i+1$ that are subclasses of the class c_j^i, and $pa(c_j^i)$ is the superclass of c_j^i at level $i-1$. Figure 6.2 illustrates the hierarchy structure of labels.

In this tree hierarchy, it is assumed that all *leaves* are situated at level C, and that each node of any other level has at least one child (in this case, it is called *an internal node*), and that, at each level, at least one node has more than one child (otherwise this level should be the last one [2]. If there

2. This assumption is done without loss of generality since every hierarchical tree can be transformed for verifying it: once the depth C of the hierarchy is established (it is the greatest level such that there is at least one node with a sibling).

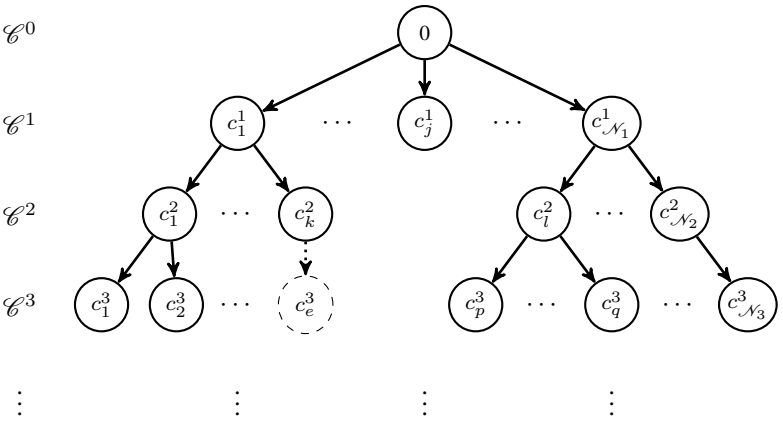


Figure 6.3 – Hierarchy structure of labels (0 being the fictive root, it is the first node of each path to any node in the hierarchy), for example, $ch(c_1^1) = \{1, 2, \ldots, k\}$ and $pa(c_l^2) = c_{\mathscr{N}_1}^1$. (A fictive node c_e^3 was added as a virtual child of the node c_k^2 to transform this latter into an internal node)

is a node c_j^i from a level $i \in [1, C-1]$ that has no child then this node can be replicated as many times as necessary in all the following levels to create fictive descendant nodes until the last level C.

Under this convention, it holds that:

$$\mathscr{N}_i < \mathscr{N}_j \text{ when } i < j \tag{6.1}$$

Also, in this study, we assume that each sample s of the dataset D, is associated to a ground-truth label which is a C-uplet, given by the attribute $glabel$, such that: $s.glabel = (c^1, \ldots, c^C)$ where each $c^i \subseteq \mathscr{C}^i$ is the ith label of the C-uplet, abbreviated $s.glabel(i)$, which corresponds to the ground-truth classes of the sample at the ith level. Here we consider that c^i is a set of classes in $\mathscr{C}^i$, since we want to cover the most general case where one sample can belong to several classes at the same level. However, this situation will be forbidden by the intra-category exclusivity (ILE) principle. Indeed, in this data labeling, two fu ndamental principles must hold:

— At each level i, any sample s from D has a unique ground-truth class given by $s.glabel(i)$: [3]

$$\forall s \in D, \forall i \in [1, C] : |s.glabel(i)| = 1 \qquad \text{(ILE principle)}$$

— For any sample s from D, the ground-truth label is hierarchically organized and its component labels represent a correct path from the root to a leaf in the hierarchy i.e. $s.glabel = (c_{j(1)}^1, c_{j(2)}^2, \ldots, c_{j(C)}^C)$ is s.t.:

$$c_{j(1)}^1 \in \mathscr{C}^1 \text{ and } \forall i \in [2, C], j(i) \in ch(c_{j(i-1)}^{i-1}) \text{ and } c_{j(i)}^i \in \mathscr{C}^i \qquad \text{(ILH principle)}$$

Example 5. *We recall that in the BreakHis dataset (See Section 3.1), each sample s is double-labeled, benign subtypes are Adenosis (A), Fibro Adenoma (F), Tubular Adenoma (TA) and Phyllodes Tumor (PT); and malignant sub-types are Ductal Carcinoma (DC), Lobular Carcinoma (LC), Mucinous Carcinoma (MC) and Papillary Carcinoma (PC). The hierarchy has two levels $C = 2$: the category $\mathscr{C}^1$ which represents the tumor type, and the category $\mathscr{C}^2$ which is the category of the*

3. Note that the assumption of exclusivity of label, ILE principle, is done without loss of generality. Any dataset where a sample is affected with several labels of the same category, could be redesigned in order to satisfy this principle (e.g. by creating a new label joining these labels).

tumor subtype. More precisely, $\mathscr{C}^1 = \{B, M\}$ *with* $\mathscr{N}_1 = 2$ *and* $\mathscr{C}^2 = \{A,\ F,\ TA,\ PT,\ DC,\ LC,\ MC,\ PC\}$ *with* $\mathscr{N}_2 = 8$.

Here, $c_1^1 = B$ *and* $ch(c_1^1) = \{1, 2, 3, 4\}$*, i.e., the subtypes of the benign class are the four first classes of category* $\mathscr{C}^2$ *namely: A, F, TA and PT respectively corresponding to* c_1^2*,* c_2^2*,* c_3^2 *and* c_4^2*. Similarly,* $ch(c_2^1) = \{5, 6, 7, 8\}$ *contains the indices of the classes of the malign subtypes. Moreover Fibroadenoma is a benign tumor, is translated into* $pa(c_2^2) = c_1^1 = B$*, while Lobular Carcinoma is malign is translated into* $pa(LC) = pa(c_5^2) = c_2^1 = M$.



In this example, the dataset respects the ILE and ILH principles. Indeed, concerning the ILE principle, each sample of BreakHis dataset has only one unique label per level [107]. Concerning the ILH principle, for any sample, the tumor type and subtype are consistent since there is no benign sample with a malign subtype and conversely.

Imagine that BreakHis would contain a sample labeled A and also labeled DC then it would violate ILE. Moreover the existence of a sample labeled (B, DC) in BreakHis dataset would violate ILH.

In the previous example, the violation of ILE or ILH would imply that the categorization is not exclusive or the dataset is noisy (some samples are incorrectly labeled).

Example 6. *Another example of violation of ILE would be a zoological dataset with an animal that would be both a mammal and a fish (such as a dolphin).*

In the following, we assume that all the considered datasets comply with ILE and ILH principles, this will imply that classifiers should obey the corresponding constraints, as explained in Section 6.4.

6.3 The probabilistic view of HMC classification

The aim of this section is to situate a HMC classifier in the context of probability theory. More precisely, we are interested in the prediction of all the classes at each level of the tree, i.e., we focus on non-MLNP global hierarchical classification where the dataset D respects the labeling principles defined in Section 6.2).

In this case, as presented in Section 1.2, we define a classifier as a function that associates a sample s with a predicted label, denoted $s.plabel$ or $\hat{c}$, which is a C-tuple of sets of the different classes predicted across the C levels. The classification decision to affect a class $\hat{c}$ to a sample s is based on the vector of vectors $\hat{y}(s) = (\hat{y}^1, \dots \hat{y}^C)$. In this section, without loss of generality, we consider a fixed level i, hence we focus on the determination of the predicted classes $\hat{c}^i$ at level i. At each level of the tree, we can adopt the vision described in Section 1.2, then, $\hat{y}^i(s)$ is the probability vector, such that each value $\hat{y}_j^i(s)$ represents the probability with which the sample s can be affected to the class c_j^i. Proposition 1.2 can then be projected at each level i:

Corollary 6.3.1: Of Proposition1.2

For any level i in [1,C], for any j in $[1, \mathscr{N}_i]$, $\hat{y}^i_j$ (resp. $\hat{y}^i$) based on a sigmoid (resp. softmax) activation function is a probability function.

— A sigmoid activation function associates to any event $\theta^i_j(s)$ a probability where $\theta^i_j(s)$ is the event of affecting the class c^i_j to a sample s at level i.

— A softmax activation function associates to any event $\theta^i(s)$ a probability where $\theta^i(s)$ is the event of affecting a class in $\{c_1, \ldots, c_{\mathscr{N}_i}\}$ to a sample s at level i.

Example 7. *Based on the the context of Example 5, we suppose that:*

— *The hierarchical classifier is activated with sigmoid functions. After forwarding the sample s, the event $\theta^1_1(s)$ is the event of affecting the sample s to the the class c^1_1 (which corresponds to the benign class). Suppose that we got at the first level: $\hat{y}^1 = (0.65, 0.85)$, that means that $\mathcal{P}(\theta^1_1(s)) = 0.65$. The sample s has a probability of 0.65 to be affected to the benign class and a probability of 0.35 to not be affected to it. In this example, with the sigmoid function, the sample s will be affected to both benign and malignant class (since there is no exclusivity in the same level and both values exceed 0.5.*

— *The hierarchical classifier is activated with a softmax function. the event $\theta^1(s) = 1$ is the event of affecting the sample s to the the class c^1_1 (which corresponds to the benign class). Suppose that we got at the first level: $\hat{y}^1 = (0.65, 0.35)$, that means that $\mathcal{P}(\theta^1(s) = 1) = 0.65$ and $\mathcal{P}(\theta^1(s) = 2) = 0.35$. The sample s has a probability of 0.65 to be affected to the benign class and a probability of 0.35 to be affected to the malignant class. In this example, with the softmax function, the sample s will be affected to the benign class (since the decision policy is based on argmax function).*

Corollary 6.3.2: of Proposition 1.2

Given a network activated by a final softmax activation function, for any level $i \in [1, C]$, and for any distinct j_1, j_2 in $[1, \mathscr{N}_i]$, for any sample s the events $\theta^i(s) = j_1$ and $\theta^i(s) = j_2$ are mutually exclusive.

These results allow us to use the term probability in the rest of this document and will help us to define the Bayesian adjustment in the next chapter.

6.4 The two constraints of a hierarchical classifier

In this study, we identify and formalize two constraints to deal with the HMC problem and to adapt to the labeling principles. We recall that $s.plabel$ returns the C-uplet of predicted labels $(\hat{c}^1, \hat{c}^2, \ldots \hat{c}^C)$ given by the classifier, from now on, $s.plabel(i)$ denotes the predicted class of the ith category (which is a set because some decision policies may allow to affect more than one class at each level): $s.plabel(i) = \hat{c}^i$. The two following constraints should be satisfied by a rational hierarchical classifier in order to comply with the two principles defined in Section 6.2.

Definition 6.4.1: ILE constraint

A classifier complies with *the exclusivity intra-level* (ILE) constraint if each sample s of the dataset D has a unique predicted label per level:

$$\forall s \in D, \forall i \in [1, C] : |s.plabel(i)| = 1 \qquad \text{(ILE constraint)}$$

In other words, it means that $\hat{c}^i$ is a singleton containing only one predicted class per level.

Definition 6.4.2: ILH constraint

A classifier complies with the *Hierarchy inter-levels* (ILH) constraint if for any sample s, its predicted labels represent a correct path from the root to a leaf in the hierarchy i.e. $s.plabel = (\hat{c}^1_{j(1)}, \hat{c}^2_{j(2)}, \ldots, \hat{c}^C_{j(C)})$ is s.t.:

$$\hat{c}^1_{j(1)} \in \mathscr{C}^1 \text{ and } \forall i \in [2, C], j(i) \in ch(\hat{c}^{i-1}_{j(i-1)}) \text{ and } \hat{c}^i_{j(i)} \in \mathscr{C}^i \qquad \text{(ILH constraint)}$$

Example 8. *In the case of BreakHis classification for example, we suppose that, forwarding a sample through a network associated with the hierarchy of Example 5 yields these outputs:* $\hat{y}^1 = (0.6, 0.4)$ *and* $\hat{y}^2 = (0.2, 0.2, 0.1, 0.1, 0.3, 0.025, 0.025, 0.05)$. *This means that the sample* s *has a probability of* 60% *to be benign and a probability of* 40% *to be malignant. The sample* s *is then assigned to the class "benign" and the subclass "Ductal Carcinoma" which violates ILH since it is a malignant subtype.*

In this chapter, we first reviewed the literature and summarized the most relevant related works, we analyzed the existing solutions and define aspects to improve. Starting by formalizing the HMC problem by abstracting the dataset principles, then we extracted the constraints that the classifier should satisfy in order to address the hierarchical classification. We note that the articles we have summarized, do not satisfy the ILH and ILE principles because they either use a sigmoid output activativation function (which does not ensure ILE constraint), or they do not pay attention to the hierarchical conformity (violating ILH constraint). Thhe third kind of approach, the B-CNN's take into account the ILH and ILE constraints, but the architecture is not exploited efficiently by over-specializing some parts of the network.

In the next chapter, and taking into account these limitations, we present a new global approach for handling the HMC problem inspired by the B-CNN approaches and show that this model behaves well, since the output provides a unique predicted class by level (contrarily to the approaches based on sigmoid functions) and respects the hierarchy constraints (contrarily to some local approaches of the literature) i.e., we show that the new model satisfies the ILE and ILH constraints.

Chapter 7

An architecture compliant with ILE and ILH constraints

Résumé en français du Chapitre 7 : "Une architecture conforme aux contraintes ILE et ILH"

Après avoir situé le contexte de la classification hiérarchique et mis en place le formalisme nécessaire pour aborder la problématique, nous proposons dans ce chapitre une solution répondant aux contraintes imposées. Pour cela, nous avons d'une part conçu une architecture particulière, renforcée par un ajustement bayésien au niveau de la dernière couche. D'autre part, nous avons défini une fonction de perte personnalisée, composée de deux parties : une partie responsable de capturer les erreurs de prédiction et une partie responsable de pénaliser le réseau en cas de violation de la hiérarchie.
Afin d'évaluer l'efficacité de notre solution, des expérimentations sont conduites sur les 5 jeux de données hiérarchiques présentés au chapitre 3.

Contents

7.1 General architecture for GHC-CNN 103
7.1.1 The hidden layers (HL) 103
7.1.2 The penultimate layer of primary outputs (PNPO) 104
7.1.3 The final adjusted finer output layer (FAFO) 104
7.2 Hierarchical loss functions 107
7.2.1 Level weighted loss function 107
7.2.2 Hierarchy violation loss function 108
7.3 Compliance with ILE and ILH constraints 108
7.4 Experiments 109
7.4.1 Data preparation and computational details 109
7.4.2 Training strategy 110
7.4.3 Results and discussion 112

In this chapter, in order to overcome some limitations of local classifiers and based on the idea that conditional probabilities should play a role to constrain the links between a superclass and its subclasses, we propose a new architecture called "Globally Hierarchically Coherent"-CNN (GHC-CNN) which exploits both Bayes' rule and branching CNN, yielding an efficient architecture with a suitable semantic loss function. This new classifier can be considered global since in the architecture that we propose the whole network is involved in the prediction of the entire label of a sample (i.e., from its class in the top level of the hierarchy to its class in the bottom level). GHC-CNN can also be considered as local since it is based on a Bayesian adjustment (which encodes the hierarchy in terms of conditional probabilities together with a customized semantic loss function that penalizes drastically the hierarchy violation. A teacher-forcing strategy learning (which uses the ground truth class of the superclass in order to predict the subclass) is used to enhance the learning quality.

We expose the proposed architecture and the customized loss function. Then, we describe the experiments done on two hierarchical datasets and the obtained results. Finally, we mention some perspectives that would be fruitful to explore in future works.

7.1 General architecture for GHC-CNN

The GHC-CNN architecture is designed to guarantee compliance with the hierarchical constraint between superclasses and subclasses (ILH) and the constraint of exclusivity within the same category (ILE). Given a sample s as input, an output is produced by the GHC-CNN network which is composed as follows:

— A set of hidden layers described in Section 1.1.1 with a final hidden layer whose output is denoted $z(s)$.

— A penultimate primary output layer (PNPO) with C outputs : $v^1, v^2, \ldots, v^C$, each output v^i is a vector of length $\mathscr{N}_i$ (containing the $\mathscr{N}_i$ membership probabilities for the sample to belong to each class of $\mathscr{C}^i$), see Section 7.1.2.

— The final adjusted finer output layer (FAFO) is a layer composed of $C-1$ outputs adjusted from the PNPO corresponding layers. The aim of this adjustment is to impose the respect of the ILH constraint. This adjustment is performed with a Bayesian update that encodes the hierarchical constraint, see Section7.1.3.

— A particular loss function that penalizes both hierarchy violation and classification errors (weighted wrt the depth in the hierarchy), see Section 7.2.

— The final output vector is the C-uplet GHC-CNN$(s) = (\hat{c}^1, \hat{c}^2, \ldots, \hat{c}^C)$, denoted by $\hat{c}(s)$, defined by the argmax decision policy where each $\hat{c}^i = c^i_k$ with $k = \operatorname{argmax} \hat{y}^i$.

Figure 7.1 illustrates the GHC-CNN general architecture.

To sum up, $PNPO(s) = (v^1, v^2, \ldots, v^C)$, with $v^i = \operatorname{softmax}(z^i)$ (see Section 1.1.3), then $FAFO(s) = (\hat{y}^1, \hat{y}^2, \ldots, \hat{y}^C)$ is the C-uplet of probability distributions associated to each level after the Bayesian adjustment. The final output, denoted GHC-CNN(s), corresponds to the the C-uplet of predicted classes GHC-CNN$(s) = (\hat{c}^1, \hat{c}^2, \ldots, \hat{c}^C)$.

7.1.1 The hidden layers (HL)

Any neural network backbone can be used for the first layers, the particularity of GHC-CNN architecture only appears at the penultimate and last layers of the network. In practice, we opted for the VGG19 model introduced in 1.1.6. We recall that this model is a pre-trained CNN, with 19

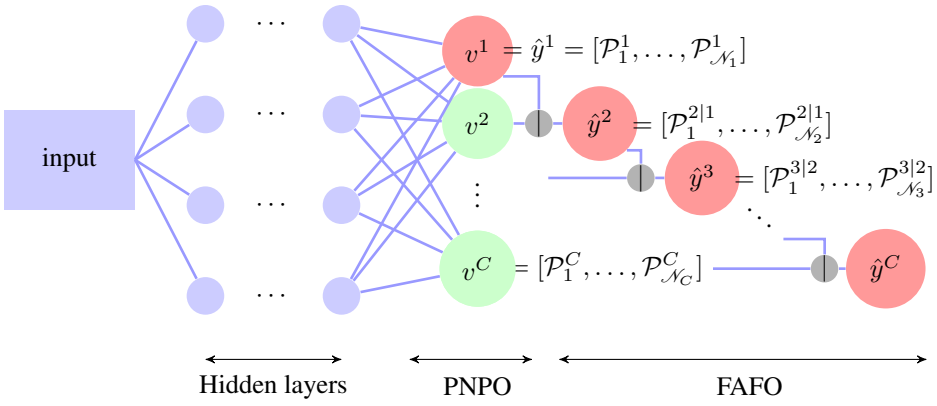


Figure 7.1 – The GHC-CNN general architecture. The nodes ⏀ represent the Bayesian adjustment.

learnable layers: 16 convolutional layers followed by 3 fully connected layers, with a total of 144M parameters. VGG19 was also used for fashion images hierarchical classification by Kolisnik et al [54] and by Seo and Shin [98] with a promising learning behavior. As a consequence, this model is supposed to have the faculty to identify the basic image features.

Let us denote by z(s), the output of the last fully connected layer. z(s) is then projected C times to give the C vectors $(z^1, z^2, \ldots, z^C)$. For each level i, z^i is a vector of dimension $\mathcal{N}_i$ obtained by projecting the initial vector $z(s)$ of the last fully-connected hidden layer into a vector of dimension $\mathcal{N}_i$. The output of this part is given by:

$$HL(s) = (z^1, z^2, \ldots, z^C)$$

7.1.2 The penultimate layer of primary outputs (PNPO)

In the GHC-CNN architecture, the PNPO layer produces C penultimate outputs denoted v^i with $i \in [1, C]$, one for each category $\mathscr{C}^i$ present in the dataset. The inputs of PNPO are activated with a softmax function (described in Section 1.1.1).

According to the probability vision of a hierarchical classifier, adopted in Section 6.3 and more precisely Corollary 6.3, at each level $i \in [1, C]$, v^i is a probability vector of dimension $\mathcal{N}_i$, i.e. $v^i = \text{softmax}(z^i) = (\mathcal{P}_1^i, \mathcal{P}_2^i, \ldots, \mathcal{P}_{\mathcal{N}_i}^i)$. Each $\mathcal{P}_j^i$ corresponds to the probability of the event $\theta^i(s) = j$ (defined in Section 6.3) for a sample s to be assigned to the class c_j^i: $\mathcal{P}_j^i = \mathcal{P}(\theta^i(s) = j) = v_j^i$.

Hence, the output of the PNPO layer is given by:

$$PNPO(s) = v(s)$$

7.1.3 The final adjusted finer output layer (FAFO)

The FAFO layer takes as input the vector of vectors $v(s) = (v^1, v^2, \ldots, v^C)$. and aims at adjusting each vector v^i through a Bayesian update in order to encode the hierarchical constraint. Then, the new final output vector is denoted $\hat{y}^i$:

$$FAFO(v(s)) = \hat{y}(s)$$

We propose to adjust each vector v^i such that it corresponds to the probability of assigning classes at level i under the ILH constraint. The idea of this adjustment comes from the probability domain. Indeed, the probability vector v^i (obtained in the PNPO layer) represent the probability distribution (as proven in Corollary 6.3) for the sample s to be classified by the network over the classes of the level i. Similarly v^{i+1} is the probability distribution over the following level $(i+1)$ in the hierarchy.

If we considered that the predicted class at level i should be c_k^i such that $k = \text{argmax}_{k\in[1,\mathscr{N}_i]}(v^i)$ and similarly for the level $i+1$ then these predicted classes would not have taken into account the fact that the class at level i should be the parent of the one at level $i+1$. However, in our consideration, for the top-down strategy, the knowledge of the superclass must condition the knowledge of its subclasses. In terms of probabilities, the aim of the adjustment is to propagate the probability of the super-class over the probabilities of its children. In this view $\hat{y}_j^i$ is considered as acting as a kind of conditional probability. The top-down Bayesian adjustment is defined as follows:

Definition 7.1.1: The Bayesian adjustment

Given a vector of vectors $v(s) = (v^1, v^2, \ldots, v^C)$, the top-down Bayesian adjustment of $v(s)$ is starting from the highest level of the hierarchy, as follows:

$$\begin{aligned} &a. \quad \hat{y}^1 = v^1 \\ &b. \quad \hat{y}_k^{i+1} = \frac{\hat{y}_j^i \times v_k^{i+1}}{\sum_{t\in ch(c_j^i)} v_t^{i+1}} \\ &\qquad \text{for all } i,j,k \in [2,C],[1,\mathscr{N}_i],[1,\mathscr{N}_{i+1}] \text{ s.t. } k \in ch(c_j^i) \end{aligned} \tag{7.1}$$

According to this update, we first show (in Lemma 7.1.3, Equation 7.2.a) that the adjustment has performed a transformation such that any parent class probability value is the sum of the probability of its children, the second bullet shows that the ratio between a class and its siblings is preserved, this means that the rescaling is rational and conserves the distribution of the values inside a set of siblings.

Lemma 7.1.1

$$\begin{aligned} &a. \quad \hat{y}_j^i = \textstyle\sum_{k\in ch(c_j^i)} \hat{y}_k^{i+1} \\ &b. \quad \frac{\hat{y}_k^{i+1}}{\sum_{k\in ch(c_j^i)} \hat{y}_k^{i+1}} = \frac{v_k^{i+1}}{\sum_{k\in ch(c_j^i)} v_k^{i+1}} \end{aligned} \tag{7.2}$$

Proof. First, let us prove 7.2.a. for all $i,j,k \in [2,C],[1,\mathscr{N}_i],[1,\mathscr{N}_{i+1}]$ s.t. $k \in ch(c_j^i)$. We have:

$$\sum_{k\in ch(c_j^i)} \hat{y}_k^{i+1} = \sum_{k\in ch(c_j^i)} \frac{\hat{y}_j^i \times v_k^{i+1}}{\sum_{t\in ch(c_j^i)} v_t^{i+1}} = \hat{y}_j^i \times \frac{\sum_{k\in ch(c_j^i)} v_k^{i+1}}{\sum_{t\in ch(c_j^i)} v_t^{i+1}} = \hat{y}_j^i$$

Second, let us prove 7.2.b for all $i,j,k \in [2,C],[1,\mathscr{N}_i],[1,\mathscr{N}_{i+1}]$ s.t. $k \in ch(c_j^i)$, we have:

$$\frac{\hat{y}_k^{i+1}}{\sum_{k\in ch(c_j^i)} \hat{y}_k^{i+1}} = \frac{\hat{y}_k^{i+1}}{\hat{y}_j^i} = \frac{\frac{\hat{y}_j^i \times v_k^{i+1}}{\sum_{t\in ch(c_j^i)} v_t^{i+1}}}{\hat{y}_j^i} = \frac{v_k^{i+1}}{\sum_{k\in ch(c_j^i)} v_k^{i+1}}$$

□

This lemma is crucial to show the achievement of the Bayesian adjustment goal. For a better explanation of the goal of this Bayesian adjustment, let us consider two successive levels i and $i+1$. Let us assume that at the PNPO layer, the sample s is affected to the class c^i_{j1} at the ith level and to the class c^{i+1}_{j2} for the $(i+1)^{th}$ level, in other words: $\mathcal{P}(\theta^i(s) = j1) = v^i_{j1}$ and $\mathcal{P}(\theta^{i+1}(s) = j2) = v^{i+1}_{j2}$. After the top-down Bayesian adjustment, we are now considering the NEW events $\Theta^i(s) = j1$ and $\Theta^{i+1}(s) = j2$ that the FAFO layer affects the class c^i_{j1} at the ith level (note that we use Θ instead of θ to underline that the events are not the same), these new probabilities are denoted by $\mathcal{P}^*(\Theta^i(s) = j1)$ and $\mathcal{P}^*(\Theta^{i+1}(s) = j2$ respectively.

From Lemma 7.1.3 Equation 7.2.a we have:

$$\mathcal{P}^*(\Theta^i(s) = j1) = \sum_{j2 \in ch(c^i_{j1})} \mathcal{P}^*(\Theta^{i+1}(s) = j2) \tag{7.3}$$

Due to Corollary 6.3, all the events at level i and all the events at level $i+1$ are mutually exclusive. Hence, due to the law of total probabilities, Equation 7.4 holds after the adjustment:

$$\mathcal{P}^*(\Theta^i(s) = j1) = \sum_{j2 \in ch(c^i_{j1})} \mathcal{P}^*(\Theta^i(s) = j1 | \Theta^{i+1}(s) = j2) \times \mathcal{P}^*(\Theta^{i+1}(s) = j2) \tag{7.4}$$

From 7.3 and 7.4, we conclude, that this top-down Bayesian adjustment imposes:

$$\mathcal{P}^*(\Theta^i(s) = j1 \,|\Theta^{i+1}(s) = j2) = 1 \tag{7.5}$$

Equation 7.5 justifies that after the update, knowing a subclass, i.e. $\Theta^{i+1}(s) = j2$ occurred, should make the event $(\Theta^i(s) = j1)$ certain to occur in order to respect the ILH constraint.

Let us illustrate the Bayesian adjustment mechanism:

Example 9. *Similarly, with the BreakHis context, suppose that forwarding a sample through the network yields a probability distribution $\mathcal{P}^1$ of the tumor type equal to 0.6 for the Benign class and 0.4 for the malignant one: $\mathcal{P}^1 = (0.6, 0.4)$ and $\mathcal{P}^2 = (0.3, 0.025, 0.025, 0.05, 0.2, 0.2, 0.1, 0.1)$.*

After the top-down Bayesian adjustment of $\mathcal{P}^2$, we obtain $\mathcal{P}^{2^*}$= (0.45 [1], 0.0375, 0.0375, 0.075, 0.13 , 0.14, 0.066, 0.068).The reader can check that: $\mathcal{P}^{2^*}(1) + \mathcal{P}^{2^*}(2) + \mathcal{P}^{2^*}(3) + \mathcal{P}^{2^*}(4) = 0.6 = \mathcal{P}^{1^*}(1)$ and $\mathcal{P}^{2^*}(5) + \mathcal{P}^{2^*}(6) + \mathcal{P}^{2^*}(7) + \mathcal{P}^{2^*}(8) = 0.4 = \mathcal{P}^{1^*}(2)$.

However, we note that imposing this Bayesian adjustment is necessary to impose the IHL constraint, since it allows the network to integrate the hierarchical link, but this is still not a sufficient condition to guarantee perfect compliance with hierarchical constraints as shown on the following example.

Example 10. *Continuing in the same vein as in the previous example, now suppose that, after forwarding another sample s, we got a probability distribution $\mathcal{P}^1$ of the tumor type equal to 0.4 for the Benign class and 0.6 for the malignant one: $\mathcal{P}^1 = (0.4, 0.6)$ and $\mathcal{P}^2$= $(0.3, 0.025, 0.025, 0.05, 0.2, 0.2, 0.1, 0.1)$*

After the Bayesian adjustment, we get: $\mathcal{P}^{1^*} = (0.4, 0.6)$ and $\mathcal{P}^{2^*}$= $(0.3, 0.025, 0.025, 0.05, 0.2, 0.2, 0.1, 0.1)$

The sample s will then be affected to the tumor type "Malignant" but to the tumor subtype Adenosis (A); which is a benign subtype, violating the hierarchy.

1. $0.45 = 0.3 \times 0.6/(0.3 + 0.025 + 0.025 + 0.05)$

This counter-example shows that this Bayesian adjustment is not sufficient to guarantee the absence of any violation. However, the experiments show that this adjustment improves hierarchical constraint compliance.

This top-down Bayesian adjustment is a way to encode the ILH constraint, such that the superclass is guiding the subclass.

This Bayesian adjustment at the FAFO layer is a way of presenting the ILH constraint to the network. However, the network must first learn to correctly classify the samples in the hierarchy by giving the right class at each level. To do this, we will need to customize a loss function to ensure successful learning. In the following section, we present the customized loss function.

7.2 Hierarchical loss functions

In order to support the ILH constraint and to take into account the different levels of robustness required at different levels of the hierarchy, the hierarchical loss function $Loss^{h.v}$ is composed of two parts $Loss^h$ that penalizes the errors with respect to the ground truth, this penalty is weighted according to the hierarchy level, and $Loss^v$ that translates the semantic constraint ILH.

$$Loss^{hv} \quad = \quad Loss^h + Loss^v \tag{7.6}$$

7.2.1 Level weighted loss function

$Loss^h$ is the part that guarantees the learning of the classification at each level of the hierarchy. It is a linear combination of the cross-entropy distance between the prediction and the ground truth at each level:

$$Loss^h \quad = \quad \sum_{i=1}^{C} \alpha_i \times d(y^i, \hat{y}^i), \text{ with } \alpha_i \in \mathbb{N} \tag{7.7}$$

where $d(q, \hat{q})$ is the cross-entropy between the two vectors q and $\hat{q}$: $d(q, \hat{q}) = -\sum_{j=1}^{k} q[j] \log(\hat{q}[j])$ (See Section 1.1.3). According to the nature of the hierarchy, several configurations of the weights α_i are worth noticing:

— **Egalitarian penalty**: all α_i are equal. The loss function considers errors made on superclasses or subclasses to be of equal importance (in the experiments, see Section 7.4, it is implemented with $\alpha_i = 1$ for all i).

— **Superclass/subclass enhanced penalty** These variants are proposed when superclasses (respectively subclasses) are considered as more important than the subclasses (respectively superclasses) for guiding the learning, the weights should be decreasing (respectively increasing) along the hierarchy. In Section 7.4, it is implemented with $\alpha_i = C - i + 1$ (or $\alpha_i = i$ respectively) for all i. Note that the setting of α_i depends on the needs (for example, if we want to avoid compensation between levels, we have to choose the right combination).

— **Finest/Coarsest basic model**: we denote by "coarsest" (resp. "finest") basic model, the model without any FAFO update; i.e. $\hat{y} = v$, by setting $(\alpha_1, \alpha_2, \ldots, \alpha_C) = (1, 0, \ldots, 0)$ (or resp. by setting $(\alpha_1, \ldots, \alpha_{C-1}, \alpha_C) = (0, \ldots, 0, 1)$), we obtain then, the basic neural

network that classifies the coarsest class (respectively the finest class) without taking into account its subclasses (respectively its superclasses), recovering a "local classifier" (as defined in the introduction of Chapter 6).

7.2.2 Hierarchy violation loss function

Since we are proposing a Bayesian adjustment layer (the FAFO layer which may lead to hierarchy violation, we introduce a loss term that penalizes the hierarchy violation: it is the highest error done on a prediction at a level where the predicted class and subclass are not consistent, (the subclass is not a child of the class). There is a lot of ways to define the $Loss^v$. At the beginning, we thought about a big fixed value to add to the $Loss^h$, by setting $Loss^v = \infty$ in case of a hierarchy violation. However, this definition of $Loss^v$ leads to a divergence of the gradient. To overcome this problem, we thought about a term $Loss^v$ which is of the same order of magnitude as the term $Loss^h$, in order to simplify the optimization phase, namely:

$$Loss^v = \max_{i \in [1, C-1] \text{ s.t. } \hat{c}^{i+1} \text{ not child of } \hat{c}^i} \max(d(y^i, \hat{y}^i), d(y^{i+1}, \hat{y}^{i+1})) \tag{7.8}$$

where $d(q, \hat{q})$ is the cross-entropy distance (see Section 1.1.3).

The meaning of this loss defined by Equation7.8 is that each time there is a violation between a level and its successor, the value of the largest prediction error between these two levels is taken as the penalty to be paid for this violation. Then, by traversing the tree vertically, and capturing all the levels where there is a violation, we assign the largest error among these violation errors to the $Loss^v$ term.

7.3 Compliance with ILE and ILH constraints

In this section, we study the theoretical compliance of our solution with ILE and ILH constraints.

Proposition 7.3.1: ILE Compliance

A GHC-CNN network satisfies the ILE constraint.

Proof. Due to the decision policy based on the argmax function associated to the softmax function (done by the PNPO layer), only one predicted class $\hat{c}^i$ (see Section 1.2) is predicted at each level. Hence, $|s.plabel| = 1$: ILE constraint is respected. □

Concerning the ILH constraint, we have seen that even with a Bayesian adjustment, in some cases (see Example 10), some violation errors can occur. In order to overcome this problem, the hierarchy violation loss function term $Loss^v$ is designed to penalize hierarchy violation occurrence. Hence, during the training, the network weights are adjusted in order to reduce these cases of hierarchy violations.

Note that, as a perspective of this chapter, we are proposing other variants

7.4 Experiments

In this section, we expose the experiments done on the five datasets described in Chapter 3 namely BreakHis, Kaggle Fashion Product, Fashion-MNIST, Prime-MNIST and Hzoo.

7.4.1 Data preparation and computational details

For the data preparation details, each of the five datasets was split into 70% for the training, 10% for the validation, and 20% for the test set. Then, the data preparation is as follows:

— For BreakHis data preparation: we applied the same data preparation detailed in Algorithm 3 in Section 5.2 since the dataset is balanced and sufficiently augmented through this algorithm.

— For Kaggle Fashion Product dataset: we took 500 images per category from the 45 categories of this dataset, leading to a dataset size of 22500 images. The images of the training set were flipped horizontally in order to proceed to a data augmentation that doubled the number of samples.

— For Hzoo dataset, the images of the training set were flipped horizontally for performing a data augmentation (i.e., doubling the number of samples of the training set).

— For both Fashion-MNIST and Prime-MNIST datasets, images were resized to 64x64 pixels in order to get larger images. The number of images being sufficient, we did not perform any data augmentation.

— For Kaggle Fashion datasets, and Hzoo, the images were resized to 250x250 pixels, and for BreakHis to 250x164 pixels. This resize is adopted to deal with the computational constraints (GPU memory limitation).

As proposed in Chapter 4, we formalized a method to ensure transparency in data preparation. In this section, we provide a detailed description of the data preparation algorithm for each dataset, except for the BreakHis data preparation, which is already detailed in Algorithm 3:

Algorithm 4: Fashion-MNIST data preparation algorithm

ForEach $s \in Fashion - MNIST : \{s.image \leftarrow resize(s.image, (64, 64))\}$
$train, valid, test \leftarrow Split(Fashion - MNIST, (70, 20, 10))$
$train \leftarrow train \cup tr(train, H, 100)$
return (train, valid, test)

Algorithm 5: Prime-MNIST data preparation algorithm

ForEach $s \in Prime - MNIST : \{s.image \leftarrow resize(s.image, (64, 64))\}$
$train, valid, test \leftarrow Split(Prime - MNIST, (70, 20, 10))$
$train \leftarrow train \cup tr(train, H, 100)$
return (train, valid, test)

Note that the function resize(image, (new_height, new_width)) takes the original image and the target size as arguments and resizes the image to the specified dimensions. Meanwhile, the function Sample(D, c, n) selects n samples from the dataset D where $s.label = c$, ensuring the samples match the specified class c."

Algorithm 6: Kaggle Fashion data preparation algorithm

ForEach $c \in \mathscr{C}^3_{KaggleFashion} : \{D \leftarrow Sample(KaggleFashion, c, 500)\}$
ForEach $s \in D : \{s.image \leftarrow resize(s.image, (250, 250))\}$
$train, valid, test \leftarrow Split(D, (70, 20, 10))$
$train \leftarrow train \cup tr(train, H, 100)$
return (train, valid, test)

Algorithm 7: HZoo data preparation algorithm

ForEach $s \in HZoo : \{s.image \leftarrow resize(s.image, (250, 250))\}$
$train, valid, test \leftarrow Split(HZoo, (70, 20, 10))$
$train \leftarrow train \cup tr(train, H, 100)$
return (train, valid, test)

All the algorithms of this part were implemented using the Keras library with Python 3 on the Osirim platform [86]. The training period of each experiment contains 1000 epochs, with Adam optimizer, and a train batch size of 128 samples.

7.4.2 Training strategy

Preliminary experiments launched on BreakHis and Fashion-MNIST with a raw training strategy (a standard training without any warm-up or particular strategy) produced highly disturbed training curves, as shown in Figures 7.2, 7.3 and 7.4.

Figure 7.2 illustrates the evolution of the training loss of GH-CNN on both BreakHis (in blue) and Fashion-MNIST (in orange) datasets. Figure 7.3 illustrates the evolution of the validation accuracies of GH-CNN for the two hierarchy levels of the BreakHis dataset, and Figure 7.4 illustrates the evolution of the validation accuracies of GH-CNN for the three hierarchy levels of the BreakHis dataset. We deduce from these disturbed curves, that this raw learning is not convenient for GHC-CNN, since the network does not stabilize even after 1000 epochs. In order to solve this issue, we propose a training strategy to improve learning quality. This strategy divides the training into three phases:

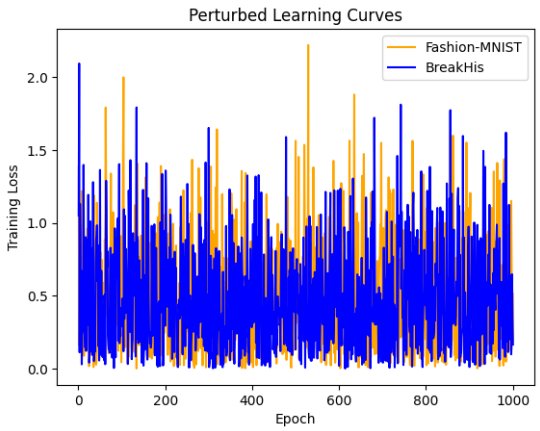


Figure 7.2 – Training loss and training accuracy curves for GHC-CNN with a raw training strategy.

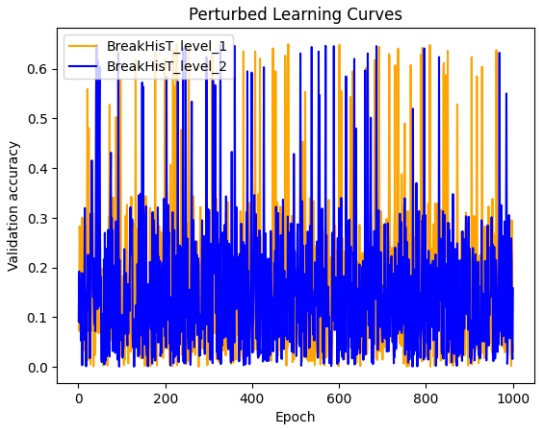


Figure 7.3 – Training loss and training accuracy curves for GHC-CNN with a raw training strategy.

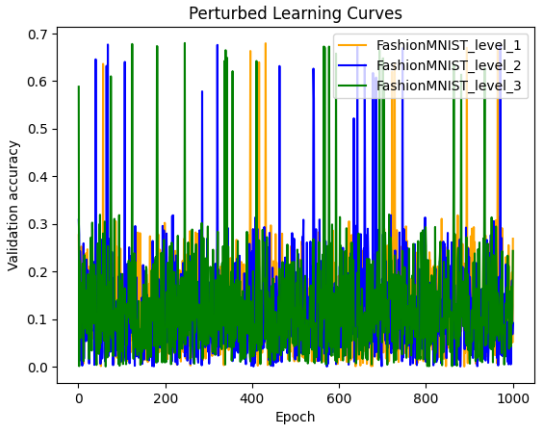


Figure 7.4 – Training loss and training accuracy curves for GHC-CNN with a raw training strategy.

1. A preliminary warm-up (during 15% of the training phase): Where the model is only trained on the coarsest category to ensure a more accurate classification at this level (which will guide the next levels).
2. A teacher-forcing strategy during 25% of the training phase): The superclass is responsible for guiding the learning of the subclasses. However, at an early stage of the training, the model is not yet able to produce accurate predictions of superclasses, which prevents proper adjustment of the subclasses. To overcome this problem, we proposed a teacher-forcing strategy (inspired by RNNs training [59]). The teacher-forcing strategy uses the ground truth about the superclass (instead of its prediction) in order to minimize errors and enable subclasses to rely on correct superclass predictions (which are their ground-truth), thus avoiding the propagation of superclass errors. This teacher-forcing strategy is applied during 25% of the training phase, in order to adapt the learning.
3. During the remaining time: The model is trained with one of the variants of $Loss^{hv}$.

We define **the hierarchy violation rate metric (HV)** in order to evaluate the variants of GHC-CNN. Note there are a lot of ways to define HV, (we can also consider the number of violations in the predicted path). For our experiments, HV is defined as the number of predicted samples disrespecting the hierarchy (at least between two successive levels) divided by the total number of wrongly classified samples in the test set.

Definition 7.4.1: Hierarchy violation rate (HV)

Given a dataset D, a hierarchy with C hierarchical levels, and a hierarchical classifier whose prediction at level i is denoted $\hat{c}^i$, the hierarchy violation rate is defined by:

$$HV = \frac{|s \in D, \text{ s.t. } \exists i \in [1, C] : \hat{c}^i(s) \neq pa(\hat{c}^{i+1}(s))|}{|s \in D|}$$

For each dataset, we experimented the GHC-CNN with several versions of the loss function, by choosing different combinations of $(\alpha^i)_{i\in[1,C])}$ for the $Loss^h$ term, as follows:

— The coarsest basic model which is the GHC-CNN which $loss^h$ is defined by setting $(\alpha_1, \alpha_2) = (1, 0)$ for the BreakHis and Prime-MNIST datasets (recall that they both have a 2 level hierarchy) and $(\alpha_1, \alpha_2, \alpha_3) = (1, 0, 0)$ for the other 3-level hierarchy datasets: the aim of this loss function variant is to train the model only on the coarsest classes, then test this new model on subsequent hierarchical levels.

— The finest basic model is a GHC-CNN with $loss^h$ defined by setting $(\alpha_1, \alpha_2) = (0, 1)$ for the BreakHis and Prime-MNIST datasets and $(\alpha_1, \alpha_2, \alpha_3) = (0, 0, 1)$ for the other datasets: the aim of this loss function variant is to train the model only on the finest classes, then to test this new model on all the levels.

— The egalitarian loss function is experimented with $\alpha_i = 1, \forall i \in [1, 3]$.

— The superclass enhanced penalty is experimented with $(\alpha_1, \alpha_2) = (2, 1)$ for the BreakHis and Prime-MNIST datasets and $(\alpha_1, \alpha_2\alpha_3) = (3, 2, 1)$ for the other datasets.

7.4.3 Results and discussion

Table 7.1 represents the accuracy per level and the hierarchy violation rate results obtained with a GHC-CNN for which the loss function is varied. The parameters of the loss function are inside

Dataset	Loss variant	Acc^1	Acc^2	Acc^3	HV
BreakHis	Loss^h (1,0)	97.03	76.45*	-	58.13
	Loss^h (0,1)	85.83*	95.49	-	65.14
	Loss$^h(1,1)$	97.65	94.91	-	11.06
	Loss$^h(2,1)$	98.43	96.78	-	9.15
	Loss^v	67.03	58.91	-	66.25
	Loss$^{hv}(2,1)$	**98.46**	**96.81**	-	**4.39**
Fashion Kaggle	Loss^h (1,0,0)	99.98	85.14*	69.03*	69.45
	Loss^h (0,0,1)	80.13*	78.02*	93.12	71.43
	Loss^h (1,1,1)	99.47.	86.63	91.79	18.67
	Loss^h (3,2,1)	99.81	88.95	94.61	10.19
	Loss^v	71.15	58.21	69.43	78.84
	Loss^{hv} (3,2,1)	**99.31**	**98.74**	**95.06**	**3.45**
Fashion MNIST	Loss^h (1,0,0)	98.13*	86.43*	85.01*	63.07
	Loss^h (0,0,1)	89.12*	67.31*	93.12	85.44
	Loss^h (1,1,1)	99.47.	86.63	91.79	18.67
	Loss^h (3,2,1)	99.81	88.95	94.61	10.19
	Loss^v	25.62	18.13	31.38	91.16
	Loss^{hv} (3,2,1)	98.37	**92.74**	**97.16**	**8.24**
Prime-MNIST	Loss^h (1,0)	95.32	70.32	-	65.92
	Loss^h (0,1)	80.32	99.73	-	58.42
	Loss$^h(1,1)$	92.65	98.96	-	27.12
	Loss$^h(2,1)$	92.34	96.24	-	69.185
	Loss^v	35.96	21.71	-	83.75
	Loss$^{hv}(2,1)$	**90.92**	**97.65**	-	**15.74**
Hzoo	Loss^h (1,0,0)	96.13	85.23*	76.62*	71.89*
	Loss^h (0,0,1)	70.13*	69.92*	98.12	64.03
	Loss^h (1,1,1)	93.45	88.67	97.51	43.22
	Loss^h (3,2,1)	92.23	90.19	94.35	36.76
	Loss^v	31.15	28.21	11.43	78.84
	Loss^{hv} (3,2,1)	**92.31**	**90.56**	**94.23**	**22.67**

Table 7.1 – Performances of a GHC-CNN

parenthesis, (α_1, α_2[, α_3]). **Acc**^{i} is the accuracy percentage for classes of level i. The ***** means that the model was only tested (but not trained) for this level.

Table 7.1 and Figures 7.2, 7.3, 7.4 and 7.5 shows that:

— For the three first datasets, the *coarsest basic model* approach is less accurate for classifying in the finest class than the *finest basic model* one, even if the first approach has a greater performance on the coarsest level. It seems that the network needs to be fine-tuned on finer classes in order to be more efficient.

— For the two last datasets, namely Prime-MNIST and HZoo, the *finest basic model* approach is less accurate for classifying in the coarsest class than the *coarsest basic model*, also From the basic models, it seems that the network performs better in the finest level than in the coarsest one.

— For all datasets, the basic models have the highest rate of hierarchy violation compared to the other loss function variants (with the exception of $loss^v$). This is because not only is the network not trained at all levels simultaneously, but also the learning of hierarchical links has not yet been established.

— Training the model with only the $loss^v$ term yields very poor performance, as the model is

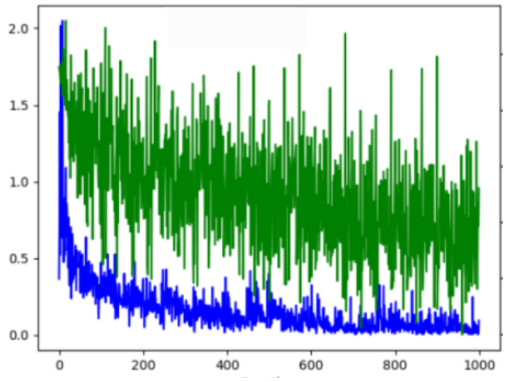


Figure 7.5 – Training loss curves of GHC-CNN with/without Bayesian adjustment ((blue/green) for the BreakHis dataset

not even trained to accurately determine the classes by level, let alone discover the hierarchical links that connect them.

— For BreakHis, Fashion-MNIST, and FashionKaggle, GH-CNN with $Loss^{hv}$ clearly outperforms the basic models. However, for Prime-MNIST and Hzoo, specifically at the coarsest level, this variant appears to perform slightly less well in terms of Acc^1 compared to the coarsest basic model. We can conjecture that perhaps in these hierarchy definitions, the coarsest level may not effectively guide the finest model. It might be easier to learn the finest classes than the coarsest ones. This could be due to factors such as primality not being a strong visual criterion for Prime-MNIST and the lack of a strong visual dangerosity criterion for Hzoo. One perspective to explore this hypothesis is to consider a bottom-up adjustment where the subclasses guide the superclasses.

— We also note that the HV rate drastically decreases when using $Loss^{hv}$ compared with the basic models, since the network is learning simultaneously superclasses and classes, the wrong classifications are decreasing and thanks to the Bayesian adjustment, the model is implicitly integrating hierarchical links.

— The *superclass enhanced penalty* loss function improves the performances on the superclasses, then improves the performances on subclasses due to the Bayesian adjustment.

— The *hierarchy violation* loss function can only be used as a complement it is not useful in itself in terms of classification because the model does not have any feedback about the accuracy of the class predictions. However, the model GHC-CNN trained with $Loss^{hv}$ has the greatest performances in terms of accuracy and F1-score for all the levels because the hierarchy violation loss forces the CNN to discover and respect the hierarchical link between labels.

— Experiments done *without the warm-up phase* showed unstable loss values which leads us to believe that warm-up helps the network to find the right starting weights for classifying the coarsest classes, then the teacher forcing strategy improves the learning of the subclasses.

— Experiments done *without the Bayesian adjustment* using the $Loss^{hv}$ showed a very disturbed training loss curve (the green curve in Figure 7.5 compared with the Bayesian adjustment training (the blue one in 7.5) attesting the crucial role of this Bayesian adjustment.

— Overall, GHC-CNN with $loss^{hv}$ variant remains fruitful for all datasets, especially as it can outperform the accuracy and significantly reduce the violation rate.

— Comparing with state-of-the-art works, we can see that, for BreakHis dataset, almost all of the approaches did not pay attention to the hierarchical link between classes, except for [79] where the hierarchy question was addressed (achieving an accuracy of 95.48% of the tumors type detection and 94.62% for the subtypes), while we obtained, with $Loss^{hv}$, 98.46% and 96.81% accuracy rates respectively. Concerning the Fshion-MNIST dataset, in [98] a B-CNN model was used (giving an accuracy of 93.33% for the finest level). In [54], a conditional probability update was used also with a B-CNN (achieving an accuracy of 99.75%, 98.06% and 91.04%) for the three levels respectively, while the GHC-CNN achieved an accuracy of 99.71%, 98.94% and 95.06% respectively.

This chapter presents the GHC-CNN, a new architecture, that encodes the labels hierarchy inside the network using both a Bayesian adjustment and a particular loss function penalizing hierarchy violations. GHC-CNN outperforms the state-of-the-art results for both BreakHis and Fshion-MNIST datasets. In conclusion, GHC-CNN is well designed such that all the layers of the network are involved in the determination of the classes at all levels. Also, the hierarchical consistency is imposed by the Bayesian adjustment before the back-propagation and guaranteed thanks to the well-designed loss functions. An additional novelty of this chapter is the flexibility of $Loss^{hv}$ which can be customized according to the nature of the task.

One primary perspective of this chapter is to compare GHC-CNN with a CNN in which the loss function encodes the hierarchical constraint using boolean logic, as proposed in [129]. A second perspective involves exploring different combinations of $Loss^{h}$. A third perspective of this work is to investigate alternative adjustment variants. For instance, we may consider a bottom-up adjustment where subclasses guide the superclasses, allowing us to assess the impact of the finest model on the coarser ones. Additionally, from a different standpoint, we can address the ILH constraint, inspired by [36]. Here, we can propose an adjustment variant in which no violation can occur and observe its impact on the quality of classification.

Chapter 8

Abstention mechanism for a robuster hierarchical classification

Résumé en français du Chapitre 8 : "Mécanisme d'abstention pour une classification hiérarchique plus robuste"

Le but de ce chapitre est d'exposer une façon de prendre en compte les connaissances haut-niveau afin de permettre d'avoir du recul sur la sortie du réseau de neurones. Pour cela, nous nous sommes placés dans le contexte de la classification hiérarchique avec le réseau GHC-CNN proposé dans le chapitre précédent. L'objectif est de post-traiter les prédictions de la classification afin d'accroître la robustesse du réseau en omettant toutes les classifications déclarées avec une probabilité inférieure à un seuil de confiance prédéfini. Ce traitement est effectué grâce à la mise en place d'un mécanisme d'abstention. Nous avons établi des propriétés en relation avec le contexte hiérarchique des classes que le seuil prédéfini devrait respecter.
Les expérimentations sont conduites sur les mêmes cinq ensembles de données hiérarchiques présentés au chapitre 3, et des métriques d'évaluation prenant en considération ce mécanisme sont définies.

Contents

8.1 State of the art about abstention in machine learning 118
8.2 Introducing an abstention layer in the GHC-CNN 121
8.2.1 Towards rational thresholds 121
8.2.2 A non-drastic, level increasing and cascading threshold 123
8.2.3 Hierarchically uncertainty alignment 125
8.2.4 New metrics for compliance with ILA constraint 126
8.3 Experimental protocols 127

As neural network applications expand into critical fields, the quality of predictions and the model's confidence in its assertions have become increasingly necessary. Particularly in sectors such as healthcare, where an erroneous decision can compromise a patient's vital prognosis, finance, where a false prediction can disrupt monetary transactions, or security, where recognition errors can compromise the integrity of a system. Errors can arise from the fact that large-scale real data is unbalanced or incorrect (noise, missing data, mislabeling). In addition, never-encountered inputs can deviate significantly from the distribution of training data, leading to erroneous outputs even in well-trained models. One solution is to prevent the model from making predictions in overly uncertain situations. By doing so, the model can achieve higher accuracy by abstaining on some samples. Thus, the optimal strategy is to minimize rejection while maintaining a high accuracy.

Abstention, also called "prediction with a reject option" or "selective prediction", or "IDK mechanism" for "I don't know"), in the context of classification, refers to the ability of a classifier to abstain from making a prediction in order to provide more cautious responses. According to [48], there are several approaches to implementing this mechanism, we highlight the most common ones (see Figure 8.2 for a diagram of the different types of approaches).

— Threshold-based approaches: involve setting a predefined confidence threshold. When the model's confidence in its prediction falls below this predefined threshold, it refrains from providing a prediction. Numerous studies have examined various threshold-based strategies (see next Section).

— Uncertainty estimation approaches: aim at estimating the global uncertainty associated with the model predictions. In these methods, the classifier assigns a numerical score that quantifies how uncertain or ambiguous the model is about its prediction. The classifier can then refrain from making predictions in the event of a high uncertainty score.

  The main difference between the threshold-based approaches and the uncertainty estimation approaches lies in how they determine when to abstain. Uncertainty approaches base their decisions on the model's estimation of uncertainty, which can provide a finer-grained understanding of prediction reliability. Threshold-based approaches, on the other hand, rely only on predefined cutoffs and do not consider the model's uncertainty explicitly.

— Ensemble-based approaches: take advantage of the collective decision-making of several models to determine whether to refrain from making predictions. The "ensemble" can be made up of several distinct models, or of models trained on different subsets of the data. By measuring disagreement or consensus between ensemble members, models can abstain when there is no agreement, indicating uncertainty or ambiguity in the classification of the instance.

It is worth noting that threshold-based approaches are widely adopted in state-of-the-art works. Consequently, our study will primarily focus on these approaches. In threshold-based methods, two prominent strategies are employed to implement the abstention mechanism [48]:

— The post-training abstention strategy: Within this category, the abstention function is applied after the completion of training. A specific metric is formulated to determine the acceptance or rejection of predictions based on this preliminary output, using a predefined threshold. The adoption of this approach can be observed in the majority of literature works in this field.

— The During-training abstention strategy: few recent studies have embraced this strategy where there is no post-training processing. Within this category, researchers have adopted two distinct approaches:

— the negative extra-class: In this approach, rather than introducing a separate metric, an additional class is added to the existing class categories within the prediction function. The entire model, including the extra class, is developed using the training data. If the extra class receives the highest score for a given test sample, the prediction for that input is considered rejected.
— the abstention loss function: In this approach, a new loss function is adopted, which incorporates both the prediction function and the rejection function. By optimizing this new loss function, the model is trained to simultaneously learn the prediction function and the rejection function.

The aim of this chapter is to refine the GHC-CNN by integrating the abstention mechanism through a threshold-based approach, allowing the model to abstain from making a decision at given levels of the hierarchy. The new model with this additional abstention mechanism is called GHA-CNN for "Global Hirerachical CNN with abstention mechanism".

We begin this chapter by discussing the most relevant state-of-the-art CNN approaches using abstention with a particular focus on threshold-based approaches. Then, we present our solution by highlighting a set of properties that our solution should satisfy in relation to the threshold. The experiments on GHA-CNN are conducted on the five datasets. experimented in the previous chapter, namely BreakHis, Fashion-MNIST, Kaggle Fashion, Prime-MNIST, and HZoo datasets in order to compare the results with the ones obtained with the GHC-CNN.

8.1 State of the art about abstention in machine learning

The concept of machine learning with a reject option was initially studied in 1970 by Chow [23], who first described the mechanism of abstention in a recognition system. It has recently gained considerable attention.

This cautious system has generated interest in various fields, including medical diagnosis (e.g., Kompa and his colleagues, in [55], proposed an uncertainty score to guide decision abstention), the financial sector (e.g., Habibpour and his colleagues, in [44] designed an uncertainty quantification metric for credit card fraud detection), and other domains such as climate change (e.g., Barnes and his colleagues, in [12], implemented a specific abstention loss function to improve predictions of anomalous global sea surface temperature).

According to [49], incorrect predictions may occur for two main reasons: ambiguity rejection (inability to distinguish two samples of distinct classes) and novelty rejection (inability to classify a never-encountered sample). Figure 8.1 taken from [49] provides an illustration of a classification scenario where two classes overlap in a given region. The figure shows different situations related to the abstention mechanism. The dotted lines represent the boundaries of the abstention mechanism, the dash-dotted line represents the boundary predicted by the classifier, and the solid line represents the ground truth relation. In the first image (a), ambiguity rejection occurs due to a non-deterministic relation between sample X and its ground truth label Y. The dotted lines enclose the rejection region, and examples within this region (marked with cross symbols) are rejected. This non-deterministic relation between X and Y can refer to noisy or unclean labeling. The second image (b), introduces ambiguity rejection caused by model bias. Similarly, the rejection region is bounded by the dotted lines, and examples within this region are rejected. In this case, the classifier has not learned enough to be able to achieve accurate discrimination. The last image (c) illustrates an example of novelty rejection. The rejected examples (marked with star symbols) are located outside the dotted line,

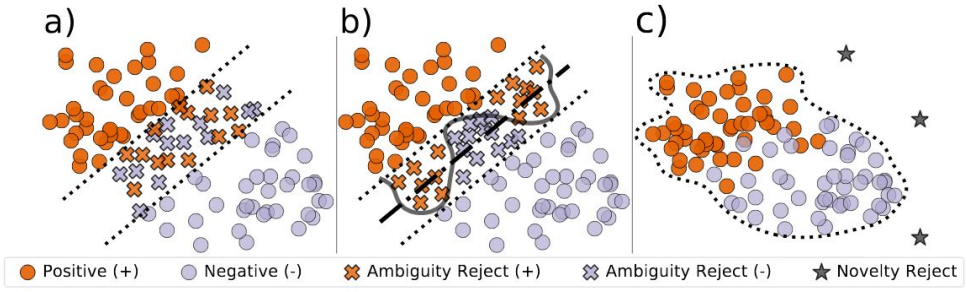


Figure 8.1 – Illustration of a binary classification scenario where the two classes overlap in a region [49]

representing novel instances that do not fit within the trained model abilities.

.

In this chapter, our primary focus will shift towards investigating the advancements and applications of abstention in neural networks, since our main objective is to enhance the GHC-CNN robustness.

First of all, let us recall Chow's rule [23]. As described in [41], to achieve a binary classification objective, in 1969, Chow established a decision rule $d : X \rightarrow A$, where A represents a set of actions typically involving the assignment of a label to a sample $x \in X$. In this context, the two classes are denoted as $+1$ and -1. There are two types of errors to consider. The first one is false positive (when an example labeled -1 is predicted $+1$), Chow associates it with a cost c^-. The second type is false negative (when an example labeled $+1$ is predicted -1), associated with a cost c^+. The author introduces a reject option represented by the value 0, which involves associated costs denoted as r^- and r^+ for examples labeled as -1 and $+1$, respectively.

The decision with abstention option d^*, known as Chow's rule is defined by:

$$d^*(x) = \begin{cases} +1 & \text{if } P(Y=1|X=x) > p^+ \\ -1 & \text{if } P(Y=1|X=x) < p^- \\ 0 & \text{Otherwise.} \end{cases} \tag{8.1}$$

where: $p^+ = \frac{c^- - r^-}{c^- - r^- + r^+}$ and $p^+ = \frac{c^- - r^-}{c^- - r^- + r^+}$. p^+ and p^- represent the thresholds that are derived from the costs and risks associated with false positive and false negative predictions. The study in In [41] focuses on the problem of binary classification by exploring Chow's rule, In order to design an effective SVM that is both consistent and sparse, the researchers derived a special loss function that focuses on estimating conditional probabilities in the vicinity of the threshold points defined by this decision rule.

Concerning the abstention in NNs, we will only focus on the threshold-based abstention mechanism, which consists of defining a threshold on the confidence score given by the classifier. If the confidence falls below the threshold, the classifier abstains from making a prediction. This approach allows the classifier to be more cautious when it lacks sufficient confidence in its predictions. A commonly used approach to determining the threshold is to use the posterior probability generated by neural networks (NNs). Several techniques have been proposed in the literature, we mention the most relevant ones:

— The fixed threshold: where a fixed value based on the top prediction value is used to refine the classification results, as adopted in [72, 4, 26].

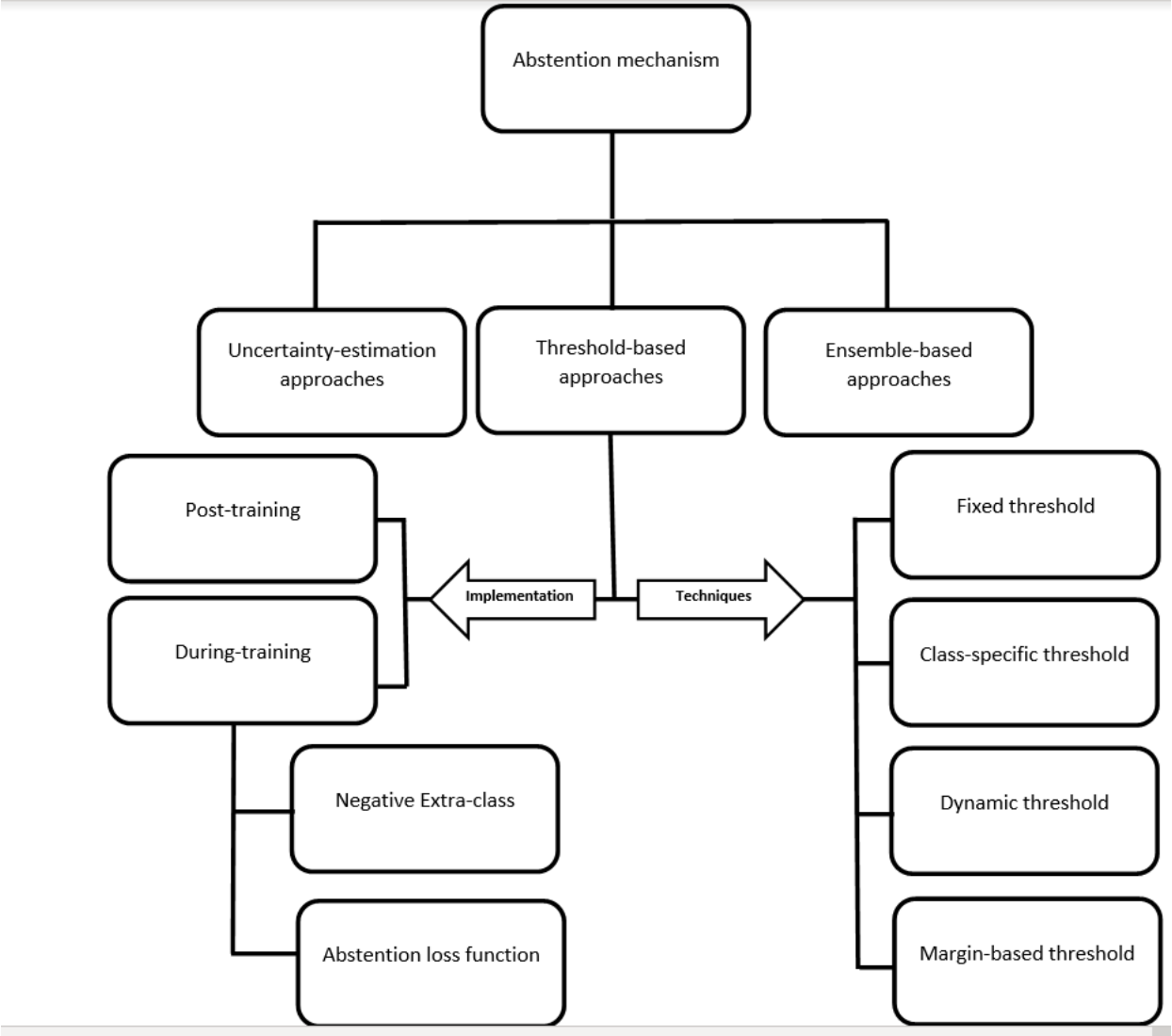


Figure 8.2 – Abstention mechanism approaches and techniques inspired by [48]

— The class-specific threshold: where there is a specific threshold for each class, as done in [34].
— The dynamic threshold: where the threshold is a function of the probability distribution of all the classes, several formulas are proposed, like in [99].
— The margin-based threshold: where the threshold integrates the difference between the two top values of the prediction as done in [88].

As mentioned in Figure 8.2, there are two main ways to implement the abstention mechanism inside the classifier. The first one is the post-training implementation, where the aim of the "rejection model", a neural network dedicated to deciding whether or not to abstain, is to refine the classification results, once the training is over. Recent studies have looked at implementing threshold-based abstention at the same time as training takes place. The aim of these approaches is to make the model learn when to abstain from making a prediction. The reader may refer, e.g., to [8], which is one of the first approaches where the abstention constraint was integrated into the loss function. Indeed, in this work, a meta loss function is introduced to simultaneously train two joint neural networks: one for the prediction model $f(x, \theta_f)$ and the other for the rejection model $g(x, \theta_g)$,

Based on these studies, we propose the use of a post-training abstention mechanism with class-specific thresholds. We justify this approach by considering the hierarchical nature of the context (HMC), which requires specific evaluations for each level.

8.2 Introducing an abstention layer in the GHC-CNN

Our objective is to enhance the solution described in 7.1 by making more confident outputs. We propose to abstain from making a classification decision $\hat{c}^i$ at level i if the maximal prediction probability at this level i, given by $\max(\hat{y}^i)$, is lower than a predefined confidence threshold τ^i. To achieve this, we introduce an additional layer called the *Decision with Optional Abstention Output (DOAO)* layer, which further refines the outputs of the FAFO layer according to this new constraint, that we call the "*Indside-Level Abstention (ILA)"* constraint, given by Definition 8.2. In practice, the DOAO layer takes as input the outputs of the FAFO layer $\hat{y}$ and generates the final output $\hat{\hat{c}}$ which is a C-uplet of decided classes $(\hat{\hat{c}}^i)_{i\in[1,C]}$, based on the predefined abstention threshold τ^i at each level, as follows:

Definition 8.2.1: ILA constraint

A classifier complies with *the indside-level abstention (ILA)* (ILA) constraint if:

$$\forall s \in D, \forall i \in [1, C] : \hat{\hat{c}}^i = \begin{cases} \hat{c}^i & \text{if } \max(\hat{y}^i) > \tau^i \\ \perp & \text{otherwise} \end{cases} \qquad \text{(ILA constraint)}$$

We recall that "$\forall i \in [1, C] : \hat{c}^i = c^i_j$ s.t. $j = \text{argmax}_{j\in[1,\mathcal{N}_i]}(\hat{y}^i)$
(Note that $\perp$ refers to an 'I DO NOT KNOW' answer)

8.2.1 Towards rational thresholds

In this section, we propose several formulas for τ^i and we look for the most suitable ones. For this last aim, we define rational properties that the threshold should satisfy.

The first threshold considered is the egalitarian threshold τ^i based on an assumption of uniform distribution of the classes at each level.

Definition 8.2.2: The egalitarian threshold

Given a tree hierarchy of C levels, the egalitarian threshold assumes a uniform distribution at each level i among the $\mathscr{N}_i$ classes of this level i.e.: $\tau^i = \frac{1}{\mathscr{N}_i}$ for all $i \in [1, C]$.

To augment the robustness of the classifier, the abstention threshold should increase with the level, because in this study, we assume that it is more difficult to predict a specific class rather than a general one.

Definition 8.2.3: level-increasing property

Given a tree hierarchy of C levels, for all $i \in [1, C]$, the threshold τ^i is *strictly increasing with the level* if

$$\forall i_1, i_2 \in [1, C], \text{ if } i_1 < i_2 \text{ then } \tau^{i_1} < \tau^{i_2}$$

Proposition 8.2.1

The egalitarian threshold is not strictly increasing with the level.

Proof. let us take $i1, i2$ from $[1, C]$, s.t. $i1 < i2$, due to the assumptions done in Chapter 6 leading to Equation 6.1, $i1 < i2 \implies \mathscr{N}_{i1} < \mathscr{N}_{i2} \implies \frac{1}{\mathscr{N}_{i1}} > \frac{1}{\mathscr{N}_{i2}} \implies \tau^{i_1} > \tau^{i_2}$ □

We now introduce another threshold called ratio threshold, which represents the ratio between the cardinality of the ith level and the one of the Cth level, as follows:

Definition 8.2.4: The ratio threshold

Given a sample s, at a given level i ($i \in [1, C]$), $\tau^i = \frac{\mathscr{N}_i}{\mathscr{N}_c}$.

Proposition 8.2.2

The ratio threshold is strictly increasing with the level.

Proof. let us take $i1, i2$ from $[1, C]$, s.t: $i1 < i2$, still due to Equation 6.1, it holds that $i1 < i2 \implies \mathscr{N}_{i1} < \mathscr{N}_{i2} \implies \frac{\mathscr{N}_{i1}}{\mathscr{N}_C} < \frac{\mathscr{N}_{i2}}{\mathscr{N}_C} \implies \tau^{i_1} < \tau^{i_2}$ □

Despite the fact that this threshold is strictly increasing with the level, it is a too strong constraint for imposing abstention in the DOAO layer because, at level C, the value of the threshold τ^C will be equal to 1, meaning that, only the predictions with a weight $\hat{y}_j^C$ higher than 1 (which is impossible) will be accepted, see the following proposition:

Proposition 8.2.3

Given a tree hierarchy of C levels, any threshold-based abstentionist neural network with a decision rule based on a sigmoid or on a softmax function that uses a ratio threshold τ^i is systematically abstaining at level C: $\hat{\hat{c}}^C = \perp$, where $\hat{\hat{c}}^C$ is the output of the network, i.e., the final predicted class, at level C.

Proof. The ratio threshold is such that $\tau^i = \frac{\mathscr{N}_i}{\mathscr{N}^C}, \forall i \in [1, C]$. Any network with a decision rule based on a sigmoid or a softmax produces an output $\hat{y} \in]0, 1[$ (due to Proposition 1.2). Hence, $\forall i \in [1, C], j \in [1, \mathscr{N}_i], 0 < \hat{y}_j^i < 1$. Now, since $\tau^C = 1, \forall j \in [1, \mathscr{N}_C]$, $\hat{y}_j^C < \tau^C$. □

Another property that we thought interesting to impose in a context where the finest classes are more difficult to predict, is that, for a given sample s and a level i, if the classification decision is to abstain at this level, then the decision classification is to abstain for all the following levels:

Definition 8.2.5: Cascading abstention on finer levels property

Given a tree hierarchy of depth C, for all $i_1 \in [1, C]$, a threshold-based abstentionist neural network satisfies the *cascading abstention on finer levels property* if

$$\hat{\hat{c}}^{i_1} = \perp \implies \hat{\hat{c}}^{i_2} = \perp \text{ for all } i_2 \in [1, C] \text{ s.t. } i_2 > i_1 \qquad \text{(cascading abstention)}$$

where $\hat{\hat{c}}^i$ denotes the output of the network at level i.

To sum up, we need a threshold-based abstention neural network with an increasing threshold by level, which does not drastically abstain at a given level and which respects the cascading abstention property.

8.2.2 A non-drastic, level increasing and cascading threshold

In order to avoid a systematic abstention at level C, we propose to relax the ratio threshold by a factor of $1/C$, as follows:

Definition 8.2.6: The cascading threshold

Given a tree hierarchy of depth C, the *cascading threshold* is $\tau^i = \frac{\mathscr{N}_i}{C.\mathscr{N}_C}$.

Proposition 8.2.4

A threshold τ such that $\tau^i = \frac{\mathscr{N}_i}{C\mathscr{N}^C}, \forall i \in [1, C]$ is increasing with the level.

Proof. From Equation 6.1, it holds that: $\forall i_1, i_2 \in [1, C]$: if $i_1 < i_2$ then $\mathscr{N}_{i_1} < \mathscr{N}_{i_2}$, hence $\frac{\mathscr{N}_{i_1}}{C\mathscr{N}_c} < \frac{\mathscr{N}_{i_2}}{C\mathscr{N}_c}$, thus $\tau^{i_1} < \tau^{i_2}$. □

Remark 3. *At level C, the cascading threshold is such that $\tau^C = \frac{1}{C} \neq 1$. Hence it is possible to design a threshold-based abstention network that does not abstain systematically at level C.*

Proposition 8.2.5

Given a tree hierarchy of depth C, a threshold-based abstentionist neural network with a decision rule based on a sigmoid or on a softmax function which uses a threshold τ such that $\tau^i = \frac{\mathscr{N}_i}{C\mathscr{N}_C}, \forall i \in [1, C]$ does not systematically abstain at level C.

Proof. At level C, the cascading threshold $\tau^C = \frac{1}{C} \neq 1$ □

We now show that a GHC-CNN with a DOAO layer based on a cascading threshold is rational with respect to abstention on finer levels.

Proposition 8.2.6

Given a tree hierarchy of depth C, the GHA-CNN with a DOAO layer based on a cascading threshold satisfies the cascading abstention on finer levels Property [8.2.5]

Proof. Let τ^i and τ^{i+1} be the thresholds of two successive levels respectively. Due to the fact that the threshold is strictly increasing with the level, we have $\tau^{i+1} > \tau^i$. For a given sample s, suppose that at level i, the classification decision is "abstain", i.e: $\hat{\hat{c}}^i = \perp$. Let j be the index of a maximum element of $\hat{y}^i$: $j \in [1, \mathscr{N}_i]$ is such that $\max_{k\in[1,\mathscr{N}_i]}(\hat{y}^i[k]) = \hat{y}^i_j$, the abstention means that $\hat{y}^i_j < \tau^i$. According to the Bayesian adjustment done in the FAFO layer, for all k in $ch(c^i_j)$: $\hat{y}^{i+1}_k \leq \hat{y}^i_j$, thus: $\hat{y}^{i+1}_k \leq \hat{y}^i_j < \tau^i < \tau^{i+1}$, i.e., $\hat{\hat{c}}^{i+1} = \perp$. Hence the result for any $i_1 < i_2$ by transitivity. □

By contraposition, we obtain the following corollary.

Corollary 8.2.1: Of Proposition 8.2.4

Given a tree hierarchy of depth C, the GHA-CNN with a DOAO layer based on a cascading threshold is such that: For any $i_1, i_2 \in [1, C]$, if $\hat{\hat{c}}^{i_1} = \perp$ and $\hat{\hat{c}}^{i_2} \neq \perp$ then $i_2 < i_1$
where $\hat{\hat{c}}^i$ is the output of the GHA-CNN at level i.

The following corollary considers a GHA-CNN with a cascading threshold and shows that if the network abstains at a given level for a sample but does not abstain for another sample at this same level, then it means that the confidence in the prediction for the first sample is greater than the one for the second sample at this precise level.

Corollary 8.2.2: Of Definition 8.2.1

Given a tree hierarchy of depth C, the GHA-CNN with a DOAO layer based on a cascading threshold is such that, for two samples $s1$ and $s2$ and any level i,
if $\hat{\hat{c}}(s1)^i \neq \perp$ and $\hat{\hat{c}}(s2)^i = \perp$ then $\max(\hat{y}(s1)^i) > \max(\hat{y}(s2)^i)$.
(where $\hat{\hat{c}}^i(s)$ is the class predicted by the GHA-CNN for sample s at level i and $\hat{y}^i(s)$ is the vector obtained by the FAFO layer for the sample s at level i).

Proof. Assume that $\hat{\hat{c}}(s1)^i \neq \perp$, $\hat{\hat{c}}(s2)^i = \perp$ and that $max(\hat{y}(s1)^i) \leq max(\hat{y}(s2)^i)$, this would mean that $max(\hat{y}(s1)^i) \leq max(\hat{y}(s2)^i) < \tau^i$ hence that $\hat{\hat{c}}(s1)^i = \perp$. □

8.2.3 Hierarchically uncertainty alignment

The following definition is inspired by the definition of N'Guyen et al. [83] stating that a well founded abstention should correspond to abstain when the sample has a prior probability (to belong to the targeted class) lower than one half, we generalize this definition to hierarchical classification. As N'Guyen et al. introduced the utility to abstain on a class. In our case it becomes the utility to abstain at a given level i for a given sample s; Since this utility concerns the classification decision at the level i, it will be function of the maximal probability value at this level, here denoted by $\hat{y}^i_m(s)$ such that: $\hat{y}^i_m(s) = \max(\hat{y}^i(s))$ The utility to abstain for a given sample s at a given level i is denoted by $u_\perp(s, \hat{y}^i_m(s))$ and should respect the following two constraints:

— $u_\perp(s, 1) = 0$

— $u_\perp(s, \frac{1}{\mathscr{N}_i}) = 1$

The idea is to maximize the gap between y^i_m and $\frac{1}{\mathscr{N}_i}$. The simplest way to represent the abstention utility of a prediction, is a linear function. By solving the constraints, we get: $u_\perp(s, \hat{y}^i_m(s)) = \frac{\mathscr{N}_i}{\mathscr{N}_i - 1}(1 - \hat{y}^i_m(s))$.

Definition 8.2.7: The utility to abstain

Given a tree hierarchy of depth C, given a hierarchical GHA-CNN classifier, given a samples s, given a level i, the utility to abstain for a sample s at level i is given by:

$$u_\perp(s, \hat{y}^i_m(s)) = \frac{\mathscr{N}_i}{\mathscr{N}_i - 1}(1 - \hat{y}^i_m(s))$$

Definition 8.2.8: Hierarchically uncertainty aligned classifier property

Given a tree hierarchy of depth C, given a hierarchical GHA-CNN classifier, given two samples $s1$ and $s2$. The classifier is *hierarchically uncertainty aligned* if and only if:

$$\forall i \in [1, C], \text{ if } \hat{\hat{c}}^i(s1) = \perp \text{ and } \hat{\hat{c}}^i(s2) \neq \perp \quad \text{then} \implies u_\perp(s1, \hat{y}^i_m(s1)) > u_\perp(s2, \hat{y}^i_m(s2))$$

Proposition 8.2.7

The system GHA-CNN is hierarchically uncertainty aligned.

Proof. Given a tree hierarchy of depth C, given a hierarchical GHA-CNN classifier, given two samples $s1$ and $s2$,

Suppose that at a given level i, we have: $\hat{\hat{c}}^i(s1) = \perp$ and $\hat{\hat{c}}^i(s2) \neq \perp$.

It means that (from Corollary 8.2.2): $\max(\hat{y}^i(s2)) > \max(\hat{y}^i(s1))$. Then we have: $\hat{y}^i_m(s2) > \hat{y}^i_m(s1) \implies 1 - \hat{y}^i_m(s2) < 1 - \hat{y}^i_m(s1) \implies \frac{\mathscr{N}_i}{\mathscr{N}_i - 1}(1 - \hat{y}^i_m(s2)) < \frac{\mathscr{N}_i}{\mathscr{N}_i - 1}(1 - \hat{y}^i_m(s1)) \implies$

$$u_\perp(s2, \hat{y}^i_m(s2)) < u_\perp(s1, \hat{y}^i_m(s1))$$

.

GHA-CNN is then uncertainty aligned.

□

8.2.4 New metrics for compliance with ILA constraint

Let us recall that the accuracy (Acc) refers to the ratio between the correctly predicted samples in the dataset over the number of samples in the test set. In chapter 7, we introduced the accuracy by level Acc^i compliant with the hierarchy levels. In the following definition, we extend the accuracy measure in order to integrate abstention, we also introduce three new measures for assessing the behavior of abstentionist models.

Definition 8.2.9: Abstention measures

For a neural network, using the abstention mechanism, we define:

— The ratio of correct declared predictions (by level) denoted by $Acc^i_\perp$:

$$Acc^i_\perp = \frac{|\{(s,i), s \in D \text{ s.t } \operatorname{argmax}(\hat{y}^i) = \operatorname{argmax}(y^i) \text{ and } \hat{\hat{c}}^i \neq \perp|}{|D|}$$

— The ratio of missed correct predictions by abstention (by level) denoted by Mcp^i:

$$Mcp^i = \frac{|\{(s,i), s \in D \text{ s.t. } \operatorname{argmax}(\hat{y}^i) = \operatorname{argmax}(y^i) \text{ and } \hat{\hat{c}}^i = \perp|}{|D|}$$

— The ratio of avoided wrong predictions by abstention (by level) denoted by Awp^i:

$$Awp^i = \frac{|\{(s,i), s \in D \text{ s.t. } \operatorname{argmax}(\hat{y}^i) \neq \operatorname{argmax}(y^i) \text{ and } \hat{\hat{c}}^i = \perp|}{|D|}$$

— The abstention gain by level denoted Ag^i is the ratio between the avoided wrong predictions and the missed correct predictions by abstention (by level). It reflects how useful the abstention has been :

$$Ag^i = \frac{Awp^i}{Mcp^i}$$

For a fruitful abstention at a level i, Ag^i must exceed 1.

Proposition 8.2.8

Given a tree hierarchy of C levels, We have that: $\forall i \in [1, C]$:

$$Acc^i_\perp = Acc^i - Mcp^i$$

Proof. Given, a tree hierarchy of C levels,

$\forall i \in [1, C]$: $Acc^i - Mcp^i =$
$\frac{|\{(s,i), s \in D, \text{ s.t } \operatorname{argmax}(\hat{y}^i) = \operatorname{argmax}(y^i)\}|}{|D|} - \frac{|\{(s,i), s \in D | \operatorname{argmax}(\hat{y}^i) = \operatorname{argmax}(y^i) \text{ and } \hat{\hat{c}}^i = \perp|}{|D|} =$
$\frac{|\{(s,i), s \in D, \text{ s.t } \operatorname{argmax}(\hat{y}^i) = \operatorname{argmax}(y^i) \text{ and } \hat{\hat{c}}^i \neq \perp|}{|D|} = Acc^i_\perp$ □

8.3 Experimental protocols

The experimental part of this chapter is an extension of the experimental part of the previous chapter, where we base our work on experiments carried out on the five datasets used in Chapter 7 with the GHC-CNN. We chose to evaluate the abstention mechanism on the GHC-CNN with $Loss^{hv}$ variant, where $Loss^{h}$ is the superclass enhanced penalty loss. The aim of these experiments is to evaluate the efficiency of the abstention mechanism added at the last layer, leading to GHA-CNN.

Table 8.1 represents the value of the cascading threshold $\tau^i = \frac{\mathscr{N}_i}{C\mathscr{N}_C}$ at each level for each dataset, this value is compared with the egalitarian threshold $\frac{1}{\mathscr{N}_i}$ to decide whether or not the abstention is **ignored** at each level, according to the following proposition.

Proposition 8.3.1

Given a tree hierarchy of depth C, given a GHA-CNN classifier,

$$\text{if } \exists i \in [1, C], \text{s.t.} \tau^i < \frac{1}{\mathscr{N}_i} \implies \hat{c}^i \neq \perp$$

The abstention at this i^{th} level is ignored.

Proof. Given a tree hierarchy of depth C, given a GHA-CNN classifier, Suppose that at a given level i, we have $\tau^i < \frac{1}{\mathscr{N}_i}$, but we know that $\max(\hat{y}^i) \leq \frac{1}{|\hat{y}^i|} = \frac{1}{\mathscr{N}_i}$

This leads to $\tau^i < \frac{1}{\mathscr{N}_i} \leq \max(\hat{y}^i) \implies \hat{c}^i \neq \perp$ □

This proposition means that if at a given level, the value of the threshold is lower than the value of a uniform distribution $\frac{1}{\mathscr{N}_i}$, for this level, the abstention process will be ignored.

That is why, we will compare the value of the cascading threshold with $\frac{1}{\mathscr{N}_i}$ to figure out if the abstention is ignored or not.

	BreakHis	Kaggle Fashion	Fashion-MNIST	Prime-MNIST	HZoo
τ^1	$\frac{1}{8}$	$\frac{4}{135}$	$\frac{1}{15}$	$\frac{1}{10}$	$\frac{1}{12}$
$\frac{1}{\mathscr{N}_1}$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{3}$
Abstain ignored(Y/N)	Y	Y	Y	Y	Y
τ^2	$\frac{1}{2}$	$\frac{7}{45}$	$\frac{1}{5}$	$\frac{1}{2}$	$\frac{1}{6}$
$\frac{1}{\mathscr{N}_2}$	$\frac{1}{8}$	$\frac{1}{21}$	$\frac{1}{6}$	$\frac{1}{10}$	$\frac{1}{6}$
Abstain ignored(Y/N)	N	N	N	N	Y
τ^3	-	$\frac{1}{3}$	$\frac{1}{3}$	-	$\frac{1}{3}$
$\frac{1}{\mathscr{N}_1}$	-	$\frac{1}{45}$	$\frac{1}{10}$	-	$\frac{1}{12}$
Abstain ignored(Y/N)	-	N	N	-	N

Table 8.1 – Cascading threshold values at each of the five datasets to figure out if the abstention is ignored (Y) or not (N).

From Table 8.1, we observe that, for the first level, the cascading threshold is ineffective for 4/5 of the datasets used. This leads us to the conclusion that the definition of such a threshold does not align well with the first level. However, the abstention process is taken into account for 80% of the datasets at the second level and for 100% of the datasets at the third level.

The abstention results in term of accuracy of the GHA-CNN are summarized in Table 8.2.

Metric	**BreakHis**	**Fashion Kaggle**	**Fashion-MNIST**	**Prime-MNIST**	**HZoo**
Acc^1	98,46	99,31	98.37	90,92	92,31
Acc^2	96,81	98,74	92.74	97,65	90,56
Acc^3	-	95,06	97.16	-	94,23
Acc^1_{bot}	98.46	99.31	98.37	90.92	92.31
Acc^2_{bot}	92,16	96,85	99,85	96.34	90.56
Acc^3_{bot}	-	79,34	83,34	-	89.44
Mcp^1	0	0	0	0	0
Mcp^2	4,65	1,89	2.89	1.31	0
Mcp^3	-	15.72	12.03	-	4.79
Awp^1	0	0	0	0	0
Awp^2	2,55	0,47	4,96	1.83	0
Awp^3	-	3,36	1.89	-	2.36
Ag^1	-	-	-	-	-
Ag^2	0.54	0.25	1.71	1.48	-
Ag^3	-	0.21	0.16	-	0,49

Table 8.2 – Abstention results for GHA-CNN

From Table 8.2, we remark that:

— The abstention mechanism was ineffective at the first level for all the datasets.

— For the BreakHis dataset, where the abstention mechanism was only effective at the last level, we observe that 2.55% of correctly classified classes were lost, while only 0.54% of misclassifications were avoided, resulting in an abstention gain lower than 1.

— For Fashion Kaggle, the abstention gain for the last two levels was also lower than one, indicating that we omitted correct classifications more often than we avoided incorrect ones. Additionally, for this dataset, we note that it has the highest rate of ignored correct classifications among all the datasets, with a rate of 15.72%. This high rate may be attributed to the fact that the cascading threshold τ^3 is very high compared to the uniform distribution, suggesting that at this level with 45 classes, a threshold equal to 1/3 is overly cautious.

— Fashion-MNIST and Prime-MNIST are the only datasets with an abstention gain higher than one, demonstrating that the abstention mechanism was more helpful in avoiding mistakes than in losing correct classifications. Moreover, the highest rate of avoided misclassifications, equal to 4.96%, was observed at the second level of Fashion-MNIST.

— For the Hzoo dataset, the abstention mechanism was ignored at both the first and second levels, and at the last level, the abstention gain was lower than one.

In this chapter, we have successfully incorporated an abstention layer into the GHC-CNN model to enhance the robustness of its outputs. We could discover the vast field of abstention in neural networks and the several techniques and approaches surrounding it. We have established a theoretical framework with specific properties for assessing the relevance of a threshold formula. Additionally, we have developed new metrics to evaluate the performance of an abstentionist network.

We conclude from these observations that, on a global scale, all the datasets were receptive to the abstention mechanism, successfully avoiding misclassifications and abstaining from uncertain

but correct classifications. However, it's important to note that the choice of the threshold is not universally suitable for all datasets. For instance, in the case of the first-level dataset, the abstention mechanism was ineffective due to an insignifiant threshold. Additionally, in three out of five datasets, employing such a threshold resulted in a higher likelihood of disregarding correct decisions rather than preventing incorrect ones.

These observations lead us to consider the need for a more suitable threshold formula that adheres to the properties defined in Definitions 8.2.3 and 8.2.5. Such a formula should be better aligned with the experimental context and the specific dataset in question. One potential avenue for exploration in this regard is to evaluate the utility of each sample and devise a formula based on both this utility and the empirical distribution of predictions.

Another perspective to ensure the robustness of the classifier involves the introduction of an additional loss function term, exploring the "during-training" approach, denoted as $Loss^a$. This term is designed to minimize the gap between the calculated probability and the uniform distribution at each level.

$$Loss^a = - \sum_{i \in C, \hat{c}^i \neq \perp} log(\frac{y_m^i}{\tau^i}) \tag{8.2}$$

However, after implementing this loss function, we observed that the training process was disrupted, as the loss curve exhibited significant fluctuations, indicating a challenging learning process. This disturbance can be attributed to the additional constraint of abstention introduced by the loss function. A thorough investigation of this issue and potential solutions will be explored in future work, drawing inspiration from the "during-training" state-of-the-art approaches.

In this part, we explored the integration of high-level knowledge at the neural network conception stage, through the design of a specific architecture and a customized loss function. High-level knowledge is also introduced at the output of the neural network, to enhance its robustness by adding the possibility for the network to abstain. We have chosen a particular kind of knowledge representing hierarchical links between labels. Multi-label hierarchical classification was therefore a very good choice through which we explored the integration of this external knowledge. In Chapter 6, we set up a formalization of the problem, which enabled us to encode formally two main constraints governing this type of classification. Inspired by existing solutions from the state of the art of multi-label hierarchical classification, we have proposed in Chapter 7 a new architecture, in which instead of sectioning the neural network into independent parts linked to a loss function, we have proposed a global architecture where all layers are involved in the prediction of all levels. To this end, we proposed a PNPO layer from which the primary prediction vector tree emerges. A Bayesian adjustment of these vectors is then performed at the FAFO layer level. The purpose of the adjustment is to impose that the probability of the superclass guides the prediction of the subclass. Next, we defined a two-part loss function, customized in order to impose hierarchy constraints, and to ensure correct learning. Several variants were tested, and experiments showed that taking additional knowledge into account further improved classification results.

Through the implementation of an abstention mechanism, in Chapter 8, we were able to encode a third constraint allowing the network to abstain from assigning a classification if the prediction confidence is below a pre-defined confidence threshold. We then developed properties that the threshold should respect. We also introduced new metrics among which: the rate of correct classifications ignored and the rate of incorrect classifications avoided. The results demonstrated the importance of implementing this abstention mechanism. Nevertheless, the theoretical definition of the threshold may not always align with the dataset and the prediction distribution. Therefore, we recommend an empirical determination of the threshold for improved consistency.

Conclusion (en français)

À travers les huit chapitres de cette thèse, nous avons exploré les différents aspects et domaines d'intégration des connaissances expertes au sein d'un réseau de neurones convolutif. Dans le premier chapitre, nous avons présenté les notions de base du fonctionnement d'un réseau de neurones convolutif en général, afin d'établir les fondements pour la suite de la thèse.

Le deuxième chapitre aborde certains aspects des connaissances de haut-niveau selon notre point de vue, en utilisant des exemples de travaux de la littérature. Nous avons catégorisé les connaissances supplémentaires en deux catégories : les connaissances humaines supplémentaires et les connaissances boîte-noire générées par le réseau de neurones. Nous avons présenté des exemples de ces connaissances ainsi que les approches utilisées pour les exploiter. Ce chapitre a souligné le manque d'études explicites sur le rôle des connaissances dans les réseaux de neurones, ce qui ouvre la voie à une étude plus approfondie pour recenser les types et aspects de connaissances utilisées dans les réseaux de neurones convolutifs et en général.

Le troisième chapitre a présenté les bases de données hiérarchiques utilisées dans cette thèse. Parmi celles-ci, deux jeux de données sont intrinsèquement hiérarchiques, tandis que les trois autres ont été modifiés pour incorporer une structure hiérarchique. Nous avons également présenté une méthode pour créer des liens hiérarchiques dans un jeu de données. D'autres types de liens pourraient être explorés, et des connaissances de haut-niveau pourraient être exploitées. Par exemple, le jeu de données IMAGENET contient diverses métadonnées telles que la couleur, le motif, la forme et la texture. Il serait possible d'établir des liens entre ces métadonnées et de générer des connaissances de haut-niveau sous forme de règles pour une classification multi-labels multicritères. En outre, il convient de noter qu'il existe des erreurs d'étiquetage signalées dans plusieurs études, en particulier dans la base de données ImageNet. Cela suggère la nécessité d'explorer des méthodes de nettoyage pour ces jeux de données.

Le quatrième chapitre traite de la préparation des données et aborde la question de ce qu'est une bonne préparation de données. Nous avons proposé un ensemble de principes à valider avec des protocoles correspondants. De plus, nous avons exploré des métriques pour quantifier l'informativité du jeu de données en utilisant des indices de similarité. Cependant, lors de la comparaison des métriques théoriques avec les métriques empiriques de la classification, les métriques théoriques se sont révélées peu informatives pour caractériser le jeu de données. Cette question d'informativité mérite une investigation approfondie. Nous proposons d'y donner suite en utilisant des métriques basées sur la comparaison des vecteurs latents à la dernière couche avant la sortie du réseau, car ces vecteurs résument les données après leur passage dans le réseau.

Nous avons également abordé la question de la data augmentation. Pour ce faire, nous avons proposé un ensemble de principes rationnels régissant la préparation des données. Pour mieux comprendre l'impact des augmentations nous avons expérimentés différents protocoles d'augmentation

et avons proposé une approche basée sur la technique LRP (Layer-wise Relevance Propagation), qui est une technique d'explication en IA. Cette nouvelle approche proposée se base sur la métrique de similarité définie précédemment afin de comparer les pixels discriminants dans l'image d'origine et dans sa version transformée. Les expérimentations ont permis de conjecture que les transformations géométriques pour la tâche de classification abordée ont tendance à avoir un effet exploiratoire puisqu'ils permettent de découvrir d'autres régions de l'image. Tandis que les opérateurs colorimétriques ont tendance à avoir un effet mémoire puisqu'ils pointent des regions similaires. Ces résultats méritent d'être explorés sur d'autres jeux de données d'images histopathologiques, et le protocol dans la visée de l'explication de l'effet de la data augmentation mérite d'être généralisé sur d'autres datasets. De plus, d'autres techniques de visualisation pourraient être testées en lieu de la LRP. Une autre perspective intéressante serait d'explorer le travail de Ekin D. Cubuk et al. [24], qui proposent un algorithme de recherche pour trouver la meilleure politique d'augmentation de données afin que le réseau de neurones obtienne la meilleure précision de validation sur un jeu de données cible.

Le cinquième chapitre a exploité la connaissance de haut-niveau liant le grossissement microscopique à la qualité de la classification. Cela a conduit à la mise en place d'un algorithme d'apprentissage par curriculum incremental pour les données en entrée. Les expérimentations sur le jeu de données BreakHis ont confirmé l'efficacité de la technique en améliorant la classification et la convergence. Cependant, il serait intéressant d'utiliser d'autres jeux de données d'images histopathologiques (ou d'images quelconques présentant différents grossissements possibles pour étudier la généralisabilité de cette méthode.

Dans la dernière partie de cette thèse, nous avons exploré l'intégration de la connaissance au niveau de la conception de l'architecture du réseau de neurones et de la personnalisation de la fonction de perte. Nous avons choisi le contexte de la classification hiérarchique, où le lien hiérarchique représente la connaissance de haut-niveau. Nous avons formalisé le contexte hiérarchique pour poser les concepts nécessaires à la présentation de notre solution. Dans le chapitre 7, nous avons conçu un réseau spécifique appelé GHC-CNN, inspiré des B-CNN présents dans la littérature, qui est global, c'est-à-dire que toutes les couches du réseau contribuent simultanément à la génération de toutes les étiquettes. Pour cela, nous avons ajouté deux couches spéciales : la couche PNPO qui génére les prédictions préliminaires données par la fonction softmax, et la couche FAFO qui ajuste ces predictions en prenant compte de la contrainte hiérarchique à travers un ajustement Bayesien. Nous avons proposé différentes variantes personnalisées de la fonction de perte permettant d'apprendre la prédiction appropriée à chaque niveau ainsi que de capturer les éventuelles violations hiérarchiques. Les expérimentations avec les variantes de GHC-CNN sur les cinq jeux de données ont montré que l'apprentissage avec ajustement bayésien obtient de meilleures performances que l'apprentissage sans prise en compte des connaissances externes. Plusieurs perspectives pour ce travail sont envisageables. Nous pourrions commencer par explorer d'autres types d'ajustements ainsi que différentes méthodes pour définir la fonction de perte. De plus, il serait pertinent de développer des métriques d'évaluation plus adaptées à la classification hiérarchique. L'expérimentation avec d'autres jeux de données hiérarchiques permettrait de confirmer le bon comportement du modèle GHC-CNN.

Le chapitre 8 constitue une extension du chapitre 7, où nous avons envisagé un mécanisme d'abstention pour ajuster la sortie. Ce mécanisme permet au réseau de s'abstenir de donner une prédiction lorsque la confiance associée à la prédiction est inférieure à un seuil prédéfini. Nous avons observé un grand nombre de travaux sur l'abstention, mais malheureusement, nous n'avons pas pu explorer toutes les approches disponibles. Nous nous sommes focalisés sur les approches basées sur le seuil de confiance pour les classifieurs par réseau de neurones. Le contexte hiérarchique nous a

guidé dans la définition des propriétés que devrait respecter le seuil de confiance. Nous avons identifié deux propriétés : tout d'abord, le seuil devrait augmenter en fonction du niveau hiérarchique, car les classes deviennent généralement plus complexes à mesure que l'on descend dans la hiérarchie. Deuxièmement, si l'abstention est effectuée à un niveau donné, elle devrait être appliquée à tous les niveaux suivants. Ces propriétés ont orienté notre choix de seuil pour nos expérimentations. Nous avons également défini des métriques appropriées liées à l'abstention, notamment le taux de classifications correctes ignorées et le taux de classifications erronées évitées grâce au mécanisme d'abstention. L'évaluation de ce mécanisme s'appuie sur les expérimentations réalisées dans le chapitre 7 sur les cinq jeux de données mentionnés précédemment. Les expériences ont démontré que le mécanisme d'abstention basé sur la définition théorique du seuil conduit davantage à l'ignorance de classes correctement prédites qu'à l'évitement de classes incorrectement prédites. Néanmoins, l'idée d'explorer le mécanisme d'abstention reste très prometteuse, en particulier dans un contexte hiérarchique. Nous suggérons donc comme perspective d'envisager une définition empirique du seuil de confiance, basée sur la distribution des probabilités. De plus, il serait intéressant d'explorer d'autres approches d'abstention basées sur le seuil, comme celles fondées sur les écarts entre les valeurs maximales du vecteur de probabilités. Une autre voie serait d'explorer les approches d'abstention basées sur l'incertitude et la logique floue, où les frontières entre les classes sont mal connues. Enfin, il il nous semblerait pertinent d'étudier les méthodes d'apprentissage de l'abstention en définissant une fonction de perte permettant au réseau d'apprendre quand il est judicieux de s'abstenir.

En résumé, cette thèse représente une première incursion dans le vaste domaine de l'intégration des connaissances de haut-niveau au sein des réseaux de neurones. Nous avons exploré différentes approches pour tirer parti de ces connaissances : à l'entrée du réseau, dans la conception de l'architecture et dans la personnalisation de la fonction de perte, ainsi qu'à la sortie du réseau pour affiner les prédictions. De nombreuses perspectives restent à explorer pour exploiter au mieux les connaissances externes afin d'améliorer l'apprentissage et ainsi d'être un jour capable de gouverner et de donner un éclairage sur le comportement de ces systèmes dits "boîtes-noires".

Cette thèse a abouti à la publication de trois articles scientifiques, dont deux au niveau international et un au niveau national, ainsi qu'à la création d'un référentiel GitHub où sont répertoriés tous les codes utilisés dans les expérimentations de cette thèse :

— MAYOUF, Mouna Sabrine et DE SAINT CYR-BANNAY, Florence Dupin. On Data-Preparation Efficiency Application on Breast Cancer Classification, 2021 [65].

— MAYOUF, Mouna Sabrine et DUPIN DE SAINT-CYR, Florence. Curriculum Incremental Deep Learning on BreakHis DataSet. In : Proceedings of the 2022 8th International Conference on Computer Technology Applications. 2022. p. 35-41 [67].

— MAYOUF, Mona-Sabrine et DUPIN DE SAINT-CYR, Florence. GH-CNN : A new CNN for coherent hierarchical classification. In : International Conference on Artificial Neural Networks. Cham : Springer Nature Switzerland, 2022. p. 669-681 [64].

— Repertoire Githib accessible à l'adresse [62].

Conclusion (in English)

Throughout the eight chapters of this thesis, we have explored the different aspects and domains of expert knowledge integration within a convolutional neural network. In the first chapter, we introduced the basic notions of how a neural network (and especially a convolutional neural network) works in general, in order to lay the foundations for the rest of the thesis.

The second chapter discusses some aspects of high-level knowledge from our point of view, using examples from work in the literature. We have categorized additional knowledge into two categories: additional human knowledge and black-box knowledge generated by the neural network. We presented examples of this knowledge and the approaches used to exploit it. This chapter highlighted the lack of explicit studies on the role of knowledge in neural networks, paving the way for further study to identify the types and aspects of knowledge used in neural networks.

The third chapter introduced the hierarchical databases used in this thesis. Two of these datasets are intrinsically hierarchical, while the three others have been modified to incorporate a hierarchical structure. We have also presented a method for creating hierarchical links in a dataset. Other types of links can be explored, and other high-level knowledge can be exploited. For example, the IMAGENET dataset contains a range of metadata such as color, pattern, shape, and texture. It would be possible to establish links between these metadata and generate high-level knowledge in the form of rules for multi-label, multi-criteria classification.

The fourth chapter deals with data preparation, addressing the question of what constitutes good data preparation. We have proposed a set of principles to be validated with corresponding protocols. In addition, we have explored metrics to quantify the informativeness of the dataset using similarity indices. However, when comparing theoretical metrics with empirical classification metrics, the theoretical metrics were found to be uninformative in characterizing the dataset. This question of informativeness merits further investigation. We propose to address it by using metrics based on the comparison of latent vectors at the last layer before network output since these vectors summarize the data after their forwarding through the network.

We also addressed the issue of data augmentation. To this end, we proposed a set of rational principles governing data preparation. To better understand the impact of augmentation, we experimented with different augmentation protocols and proposed an approach based on Layer-wise Relevance Propagation (LRP), an AI explanation technique. This new approach uses the previously defined similarity metric to compare discriminating pixels in the original image and its transformed version. Experiments have shown that geometric transformations for the classification task under consideration tend to have an exploitative effect since they allow the discovery of other regions of the image. Color operators, on the other hand, tend to have a memory effect, since they point to similar regions. These results deserve to be explored on other histopathological image datasets, and the protocol aimed at explaining the effect of data augmentation deserves to be generalized to

other datasets. In addition, other visualization techniques could be tested instead of PRL. Another interesting perspective would be to explore the work of Ekin D. Cubuk et al. [24], who propose a search algorithm to find the best data augmentation policy so that the neural network obtains the best validation accuracy on a target dataset.

The fifth chapter exploited the high-level knowledge linking microscopic magnification to classification quality. This led to the implementation of an incremental curriculum learning algorithm for the input data. Experiments on the BreakHis dataset confirmed the effectiveness of the technique in improving classification and convergence. However, it would be interesting to use other histopathological image datasets (or any images with different possible magnifications) to study the generalizability of this method.

In the final part of this thesis, we explored the integration of knowledge at the level of neural network architecture design and loss function customization. We chose the context of hierarchical classification, where the hierarchical link represents high-level knowledge. We formalized the hierarchical context to lay down the concepts needed to present our solution. In Chapter 7, we designed a specific network called GHC-CNN, inspired by the B-CNNs found in the literature, which is global, i.e. all layers of the network contribute simultaneously to the generation of all labels. To this end, we have added two special layers: the PNPO layer, which generates the preliminary predictions given by the softmax function, and the FAFO layer, which adjusts these predictions by taking into account the hierarchical constraint through a Bayesian adjustment. We have proposed various customized variants of the loss function to learn the appropriate prediction at each level and capture any hierarchical violations. Experiments with the GHC-CNN variants on the five datasets showed that learning with Bayesian adjustment outperforms learning without taking external knowledge into account. There are several possible perspectives for this work. We could begin by exploring other types of adjustment and different methods for defining the loss function. It would also be useful to develop evaluation metrics better suited to hierarchical classification. Experimentation with other hierarchical datasets would confirm the good behavior of the GHC-CNN model.

Chapter 8 is an extension of Chapter 7, where we considered an abstention mechanism for adjusting the network's output. This mechanism allows the network to abstain from giving a prediction when the confidence associated with the prediction is below a predefined threshold. We observed a large amount of research on forbearance, but unfortunately, we were not able to explore all the available approaches. We focused on threshold-based approaches for neural network classifiers. The hierarchical context guided us in defining the properties that the confidence threshold should respect. We identified two properties: first, the threshold should increase with the hierarchical level, as classes generally become more complex as one moves down the hierarchy. Second, if abstention is performed at a given level, it should be applied to all subsequent levels. These properties guided our choice of threshold for our experiments. We also defined appropriate metrics related to abstention, including the rate of correct classifications ignored and the rate of incorrect classifications avoided thanks to the abstention mechanism. The evaluation of this mechanism is based on the experiments carried out in Chapter 7 on the five datasets mentioned above. The experiments demonstrated that the abstention mechanism based on the theoretical definition of the threshold leads more to the ignoring of correctly predicted classes than to the avoidance of incorrectly predicted classes. Nevertheless, the idea of exploring the abstention mechanism remains very promising, especially in a hierarchical context. We therefore suggest the prospect of considering an empirical definition of the confidence threshold, based on the probability distribution. In addition, it would be interesting to explore other threshold-based abstention approaches, such as those based on deviations between maximum values

of the probability vector. Another avenue would be to explore abstention approaches based on uncertainty and fuzzy logic, where the boundaries between classes are poorly known. Finally, it would seem appropriate to study abstention learning methods by defining a loss function that enables the network to learn when it makes sense to abstain.

In summary, this thesis represents a first incursion into the vast field of high-level knowledge integration within neural networks. We have explored different approaches to taking advantage of this knowledge: at the input of the network, in the design of the architecture, and in the customization of the loss function, as well as at the output of the network to refine predictions. There are still many perspectives to be explored in order to make the best use of external knowledge to improve learning and thus one day be able to govern and shed light on the behavior of these so-called "black-box" systems.

This thesis has resulted in the publication of three scientific articles, two at the international level and one at the national level, as well as the creation of a GitHub repository where all the codes used in the experiments of this thesis are cataloged:

— MAYOUF, Mouna Sabrine et DE SAINT CYR-BANNAY, Florence Dupin. On Data-Preparation Efficiency Application on Breast Cancer Classification, 2021 [65].

— MAYOUF, Mouna Sabrine et DUPIN DE SAINT-CYR, Florence. Curriculum Incremental Deep Learning on BreakHis DataSet. In : Proceedings of the 2022 8th International Conference on Computer Technology Applications. 2022. p. 35-41 [67].

— MAYOUF, Mona-Sabrine et DUPIN DE SAINT-CYR, Florence. GH-CNN: A new CNN for coherent hierarchical classification. In: International Conference on Artificial Neural Networks. Cham: Springer Nature Switzerland, 2022. p. 669-681 [64].

— A Githib directory accessible at [62].

Bibliography

[1] Cifar datasets description. https://www.cs.toronto.edu/~kriz/cifar.html.

[2] The evolution of breast cancer mortality and incidence rates among women in France between 1975 and 2020 kernel description. http://www.cngof.asso.fr/d_cohen/coB_20.htm. Accessed: 2023-06-02.

[3] Param Aggarwal. Fashion product images dataset. *Retrieved from kaggle: https://www. kaggle. com/paramaggarwal/fashion-product-images-dataset*, 2019.

[4] Ali Ahmadi, Sigeru Omatu, and Toshihisa Kosaka. An effective reject rule for reliability improvement in bank note neuro-classifiers. In *2003 IEEE XIII Workshop on Neural Networks for Signal Processing (IEEE Cat. No. 03TH8718)*, pages 509–516. IEEE, 2003.

[5] Md Zahangir Alom, Chris Yakopcic, Mst Shamima Nasrin, Tarek M Taha, and Vijayan K Asari. Breast cancer classification from histopathological images with inception recurrent residual convolutional neural network. *Journal of digital imaging*, 32(4):605–617, 2019. http://dx.doi.org/10.1007/s10278-019-00182-7.

[6] Joanna S Amberger, Carol A Bocchini, François Schiettecatte, Alan F Scott, and Ada Hamosh. Omim. org: Online mendelian inheritance in man (omim®), an online catalog of human genes and genetic disorders. *Nucleic acids research*, 43(D1):D789–D798, 2015.

[7] Teresa Araújo, Guilherme Aresta, Eduardo Castro, José Rouco, Paulo Aguiar, Catarina Eloy, António Polónia, and Aurélio Campilho. Classification of breast cancer histology images using convolutional neural networks. *PloS one*, 12(6), 2017. https://doi.org/10.1371/journal.pone.0177544.

[8] Amina Asif et al. Generalized learning with rejection for classification and regression problems. *arXiv preprint arXiv:1911.00896*, 2019.

[9] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7), 2015. https://doi.org/10.1371/journal.pone.0130140.

[10] Rajesh Balasubramanian, Rachel Rolph, Catrin Morgan, and Hisham Hamed. Genetics of breast cancer: management strategies and risk-reducing surgery. *British Journal of Hospital Medicine*, 80(12):720–725, 2019. https://doi.org/10.12968/hmed.2019.80.12.720.

[11] Dalal Bardou, Kun Zhang, and Sayed Mohammad Ahmad. Classification of breast cancer based on histology images using convolutional neural networks. *IEEE Access*, 6:24680–24693, 2018. https://doi.org/10.1109/ACCESS.2018.2831280.

[12] Elizabeth A Barnes and Randal J Barnes. Controlled abstention neural networks for identifying skillful predictions for regression problems. *Journal of Advances in Modeling Earth Systems*, 13(12):e2021MS002575, 2021.

[13] Neslihan Bayramoglu, Juho Kannala, and Janne Heikkilä. Deep learning for magnification independent breast cancer histopathology image classification. In *2016 23rd International conference on pattern recognition (ICPR)*, pages 2440–2445. IEEE, 2016. https://doi.org/10.1109/ICPR.2016.7900002.

[14] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009. https://doi.org/10.1145/1553374.1553380.

[15] Freddie Bray, Jacques Ferlay, Isabelle Soerjomataram, Rebecca L Siegel, Lindsey A Torre, and Ahmedin Jemal. Global cancer statistics 2018: Globocan estimates of incidence and mortality worldwide for 36 cancers in 185 countries. *CA: a cancer journal for clinicians*, 68(6):394–424, 2018. https://doi.org/10.3322/caac.21492.

[16] Lorenzo Bruzzone and D Fernandez Prieto. An incremental-learning neural network for the classification of remote-sensing images. *Pattern Recognition Letters*, 20(11-13):1241–1248, 1999. https://doi.org/10.1016/S0167-8655(99)00091-4.

[17] Nithin Buduma, Nikhil Buduma, and Joe Papa. *Fundamentals of deep learning*. " O'Reilly Media, Inc.", 2022.

[18] Felix Burkhardt, Astrid Paeschke, Miriam Rolfes, Walter F Sendlmeier, Benjamin Weiss, et al. A database of german emotional speech. In *Interspeech*, volume 5, pages 1517–1520, 2005.

[19] Juan José Burred and Alexander Lerch. A hierarchical approach to automatic musical genre classification. In *Proceedings of the 6th international conference on digital audio effects*, pages 8–11. Citeseer, 2003.

[20] Gail A Carpenter, Stephen Grossberg, Natalya Markuzon, John H Reynolds, David B Rosen, et al. Fuzzy artmap: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on neural networks*, 3(5):698–713, 1992. https://doi.org/10.1109/72.159059.

[21] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248, 2018. https://arxiv.org/abs/1807.09536.

[22] Soumen Chakrabarti, Byron Dom, Rakesh Agrawal, and Prabhakar Raghavan. Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies. *The VLDB journal*, 7:163–178, 1998.

[23] C Chow. On optimum recognition error and reject tradeoff. *IEEE Transactions on information theory*, 16(1):41–46, 1970.

[24] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

[25] Anderson de Andrade. Best practices for convolutional neural networks applied to object recognition in images. *arXiv preprint arXiv:1910.13029*, 2019. https://arxiv.org/abs/1910.13029.

[26] Claudio De Stefano, Carlo Sansone, and Mario Vento. To reject or not to reject: that is the question-an answer in case of neural classifiers. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 30(1):84–94, 2000.

[27] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[28] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[29] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[30] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. https://arxiv.org/abs/1708.04552.

[31] Christopher P Diehl and Gert Cauwenberghs. SVM incremental learning, adaptation and optimization. In *Proceedings of the International Joint Conference on Neural Networks, 2003.*, volume 4, pages 2685–2690. IEEE, 2003. https://doi.org/10.1109/IJCNN.2003.1223991.

[32] Baolin Du, Qi Qi, Han Zheng, Yue Huang, and Xinghao Ding. Breast cancer histopathological image classification via deep active learning and confidence boosting. In *International Conference on Artificial Neural Networks*, pages 109–116. Springer, 2018. https://doi.org/10.1007/978-3-030-01421-6_11.

[33] Jeffrey L Elman. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99, 1993. https://doi.org/10.1016/0010-0277(93)90058-4.

[34] Giorgio Fumera, Fabio Roli, and Giorgio Giacinto. Reject option with multiple thresholds. *Pattern recognition*, 33(12):2099–2101, 2000.

[35] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[36] Eleonora Giunchiglia and Thomas Lukasiewicz. Coherent hierarchical multi-label classification networks. *Advances in Neural Information Processing Systems*, 33:9662–9673, 2020.

[37] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[38] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016. https://mitpress.mit.edu/books/deep-learning.

[39] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. ISBN=0262337371, 9780262337373.

[40] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129:1789–1819, 2021.

[41] Yves Grandvalet, Alain Rakotomamonjy, Joseph Keshet, and Stéphane Canu. Support vector machines with a reject option. *Advances in neural information processing systems*, 21, 2008.

[42] Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. In *international conference on machine learning*, pages 1311–1320. PMLR, 2017. http://proceedings.mlr.press/v70/graves17a.

[43] Antonio Gulli and Sujit Pal. *Deep learning with Keras*. Packt Publishing Ltd, 2017.

[44] Maryam Habibpour, Hassan Gharoun, Mohammadreza Mehdipour, AmirReza Tajally, Hamzeh Asgharnezhad, Afshar Shamsi, Abbas Khosravi, and Saeid Nahavandi. Uncertainty-aware credit card fraud detection using deep learning. *Engineering Applications of Artificial Intelligence*, 123:106248, 2023.

[45] Guy Hacohen and Daphna Weinshall. On the power of curriculum learning in training deep networks. In *International Conference on Machine Learning*, pages 2535–2544. PMLR, 2019. http://proceedings.mlr.press/v97/hacohen19a.html.

[46] Zhongyi Han, Benzheng Wei, Yuanjie Zheng, Yilong Yin, Kejian Li, and Shuo Li. Breast cancer multi-classification from histopathological images with structured deep learning model. *Scientific reports*, 7(1):1–10, 2017. https://www.nature.com/articles/s41598-017-04075-z.

[47] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

[48] Mehedi Hasan, Moloud Abdar, Abbas Khosravi, Uwe Aickelin, Pietro Lio, Ibrahim Hossain, Ashikur Rahman, and Saeid Nahavandi. Survey on leveraging uncertainty estimation towards trustworthy deep neural networks: The case of reject option and post-training processing. *arXiv preprint arXiv:2304.04906*, 2023.

[49] Kilian Hendrickx, Lorenzo Perini, Dries Van der Plas, Wannes Meert, and Jesse Davis. Machine learning with a reject option: A survey. *arXiv preprint arXiv:2107.11277*, 2021.

[50] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. https://doi.org/10.1162/neco.1997.9.8.1735.

[51] Imagenet. http://www.image-net.org.

[52] Daniel Kahneman and Amos Tversky. Choices, values, and frames. *American psychologist*, 39(4):341, 1984.

[53] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR'15) revised in 2017*, pages 1–15, 2017. https://arxiv.org/abs/1412.6980.

[54] Brendan Kolisnik, Isaac Hogan, and Farhana Zulkernine. Condition-CNN: A hierarchical multi-label fashion image classification model. *Expert Systems with Applications*, 182:115195, 2021.

[55] Benjamin Kompa, Jasper Snoek, and Andrew L Beam. Second opinion needed: communicating uncertainty in medical machine learning. *NPJ Digital Medicine*, 4(1):4, 2021.

[56] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[57] Kai A Krueger and Peter Dayan. Flexible shaping: How learning in small steps helps. *Cognition*, 110(3):380–394, 2009. https://doi.org/10.1016/j.cognition.2008.11.014.

[58] Heng Liu, Jianyong Liu, Shudong Hou, Tao Tao, and Jungong Han. Perception consistency ultrasound image super-resolution via self-supervised cyclegan. *Neural Computing and Applications*, pages 1–11, 2021. https://doi.org/10.1007/s00521-020-05687-9.

[59] He Lyu, Ningyu Sha, Shuyang Qin, Ming Yan, Yuying Xie, and Rongrong Wang. Advances in neural information processing systems. *Advances in neural information processing systems*, 32, 2019.

[60] Martin A Makary and Michael Daniel. Medical error—the third leading cause of death in the US. *BMJ*, 353, 2016. https://doi.org/10.1136/bmj.i2139.

[61] Rui Man, Ping Yang, and Bowen Xu. Classification of breast cancer histopathological images using discriminative patches screened by generative adversarial networks. *IEEE Access*, 8:155362–155377, 2020. https://doi.org/10.1109/ACCESS.2020.3019327.

[62] Mona Mayouf. Thesis experiments. https://github.com/monamayouf/incremental_learning, 2020.

[63] Mona S. Mayouf and Florence Dupin de Saint-Cyr. Formalizing data preparation in curriculum incremental deep learning on breakhis dataset. Technical report, IRIT, 2022.

[64] Mona-Sabrine Mayouf and Florence Dupin de Saint-Cyr. Gh-cnn: A new cnn for coherent hierarchical classification. In *International Conference on Artificial Neural Networks*, pages 669–681. Springer, 2022.

[65] Mouna Sabrine Mayouf and Florence Dupin de Saint Cyr-Bannay. On data-preparation efficiency application on breast cancer classification. 2021.

[66] Mouna Sabrine Mayouf and Florence Dupin de Saint Cyr-Bannay. Préparation efficace des données d'apprentissage. application à la classification d'images pour la détection du cancer du sein. In *Conférence sur l'Apprentissage Automatique (CAp 2021)*, 2021.

[67] Mouna Sabrine Mayouf and Florence Dupin de Saint-Cyr. Curriculum incremental deep learning on breakhis dataset. In *Proceedings of the 2022 8th International Conference on Computer Technology Applications*, pages 35–41, 2022.

[68] Andrew G McDonald and Keith F Tipton. Enzyme nomenclature and classification: The state of the art. *The FEBS Journal*, 2021.

[69] Rajesh Mehra et al. Automatic magnification independent classification of breast cancer tissue in histological images using deep convolutional neural network. In *International Conference on Advanced Informatics for Computing Research*, pages 772–781. Springer, 2018. https://doi.org/10.1007/978-981-13-3140-4_69.

[70] Rajesh Mehra et al. Breast cancer histology images classification: Training from scratch or transfer learning? *ICT Express*, 4(4):247–254, 2018. https://doi.org/10.1016/j.icte.2018.10.007.

[71] Zhenzhu Meng, Yating Hu, and Christophe Ancey. Using a data driven approach to predict waves generated by gravity driven mass flows. *Water*, 12(2):600, 2020.

[72] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.

[73] Ryszard S Michalski, Igor Mozetic, Jiarong Hong, and Nada Lavrac. The multi-purpose incremental learning system AQ15 and its testing application to three medical domains. *Proc.*

AAAI 1986, pages 1–041, 1986. https://dl.acm.org/doi/10.5555/2887770.2887940.

[74] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[75] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-wise relevance propagation: an overview. *Explainable AI: interpreting, explaining and visualizing deep learning*, pages 193–209, 2019. https://doi.org/10.1155/2013/832509.

[76] Pedro Ramaciotti Morales, Robin Lamarche-Perrin, Raphael Fournier-S'niehotta, Remy Poulain, Lionel Tabourier, and Fabien Tarissan. Measuring diversity in heterogeneous information networks. *arXiv preprint arXiv:2001.01296*, 2020.

[77] Nima Habibzadeh Motlagh, Mahboobeh Jannesary, HamidReza Aboulkheyr, Pegah Khosravi, Olivier Elemento, Mehdi Totonchi, and Iman Hajirasouliha. Breast cancer histopathological image classification: A deep learning approach. *bioRxiv*, page 242818, 2018. https://doi.org/10.1101/242818.

[78] Nicolas M Müller and Karla Markert. Identifying mislabeled instances in classification datasets. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[79] Ghulam Murtaza, Liyana Shuib, Ghulam Mujtaba, and Ghulam Raza. Breast cancer multi-classification through deep neural network and hierarchical classification approach. *Multimedia Tools and Applications*, 79(21):15481–15511, 2020.

[80] Abdullah Al Nahid and Yinan Kong. Histopathological breast-image classification using local and frequency domains by convolutional neural network. *Information*, 9(1):19, 2018. https://doi.org/10.3390/info9010019.

[81] Elaheh M. Nejad, Lilly S. Affendey, Rohaya B. Latip, and Iskandar Bin Ishak. Classification of histopathology images of breast into benign and malignant using a single-layer convolutional neural network. In *Proceedings of the International Conference on Imaging, Signal Processing and Communication*, pages 50–53, 2017. https://doi.org/10.1145/3132300.3132331.

[82] Elias Chaibub Neto. Detecting learning vs memorization in deep neural networks using shared structure validation sets. *arXiv preprint arXiv:1802.07714*, 2018.

[83] Vu-Linh Nguyen and Eyke Hüllermeier. Multilabel classification with partial abstention: Bayes-optimal prediction under label independence. *Journal of Artificial Intelligence Research*, 72:613–665, 2021.

[84] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. Activation functions: Comparison of trends in practice and research for deep learning. *arXiv preprint arXiv:1811.03378*, 2018.

[85] Albert Orriols-Puig and Ester Bernadó-Mansilla. Evolutionary rule-based systems for imbalanced data sets. *Soft Computing*, 13(3):213, 2009.

[86] Osirim (observatory of systems information retrieval and indexing of multimedia contents) platform description. https://osirim.irit.fr/site/. Accessed: 2020-06-18.

[87] Rebecca Oudsema, Esther Hwang, Sharon Steinberger, Rowena Yip, and Laurie R Margolies. Screening Mammography: Guidelines versus Clinical Practice. *Journal of Breast Imaging*, 2020.

[88] Ali Parchekani, Salar Nouri, Vahid Shah-Mansouri, and Seyed Pooya Shariatpanahi. Classification of traffic using neural networks by rejecting: a novel approach in classifying vpn traffic. *arXiv preprint arXiv:2001.03665*, 2020.

[89] Hieu H Pham, Tung T Le, Dat Q Tran, Dat T Ngo, and Ha Q Nguyen. Interpreting chest x-rays via cnns that exploit hierarchical disease dependencies and uncertainty labels. *Neurocomputing*, 437:186–194, 2021.

[90] Lutz Prechelt. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pages 55–69. Springer, 1998.

[91] Rishi Rai and Dilip Singh Sisodia. Real-time data augmentation based transfer learning model for breast cancer diagnosis using histopathological images. In *Advances in Biomedical Engineering and Technology*, pages 473–488. Springer, 2020. https://doi.org/10.1007/978-981-15-6329-4_39.

[92] D Ramyachitra and Parasuraman Manikandan. Imbalanced dataset classification and solutions: a review. *International Journal of Computing and Business Research (IJCBR)*, 5(4):1–29, 2014.

[93] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[94] Olga Russakovsky et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[95] Terence D Sanger. Neural network learning control of robot manipulators using gradually increasing task difficulty. *IEEE transactions on Robotics and Automation*, 10(3):323–333, 1994. https://doi.org/10.1109/70.294207.

[96] Umme Sara, Morium Akter, and Mohammad Shorif Uddin. Image quality assessment through fsim, ssim, mse and psnr—a comparative study. *Journal of Computer and Communications*, 7(3):8–18, 2019.

[97] Jeffrey C Schlimmer and Douglas Fisher. A case study of incremental concept induction. In *AAAI*, volume 86, pages 496–501, 1986. https://www.aaai.org/Library/AAAI/1986/aaai86-083.php.

[98] Yian Seo and Kyung-shik Shin. Hierarchical convolutional neural networks for fashion image classification. *Expert systems with applications*, 116:328–339, 2019.

[99] Kulin Shah and Naresh Manwani. Online active learning of reject option classifiers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5652–5659, 2020.

[100] Philip Shilane, Patrick Min, Michael Kazhdan, and Thomas Funkhouser. The princeton shape benchmark. In *Proceedings Shape Modeling Applications, 2004.*, pages 167–178. IEEE, 2004.

[101] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019. https://doi.org/10.1186/s40537-019-0197-0.

[102] Carlos N Silla and Alex A Freitas. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1):31–72, 2011.

[103] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[104] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[105] J David Smith, Alexandria C Zakrzewski, Jennifer M Johnson, and Jeanette C Valleau. Ecology, fitness, evolution: New perspectives on categorization. *Current Directions in Psychological Science*, 25(4):266–274, 2016.

[106] Yang Song, Ju Jia Zou, Hang Chang, and Weidong Cai. Adapting fisher vectors for histopathology image classification. In *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)*, pages 600–603. IEEE, 2017. https://doi.org/10.1109/ISBI.2017.7950592.

[107] F. A. Spanhol, L. S. Oliveira, C. Petitjean, and L. Heutte. Breakhis dataset. https://web.inf.ufpr.br/vri/databases/breast-cancer-histopathological-database-breakhis/.

[108] F. A. Spanhol, L. S. Oliveira, C. Petitjean, and L. Heutte. A dataset for breast cancer histopathological image classification. *IEEE Trans. on Biomed. Engineering*, 63(7):1455–1462, 2016.

[109] Fabio Alexandre Spanhol, Luiz S Oliveira, Caroline Petitjean, and Laurent Heutte. Breast cancer histopathological image classification using convolutional neural networks. In *2016 international joint conference on neural networks (IJCNN)*, pages 2560–2567. IEEE, 2016.

[110] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014. http://jmlr.org/papers/v15/srivastava14a.html.

[111] Lichao Sun, Congying Xia, Wenpeng Yin, Tingting Liang, Philip S Yu, and Lifang He. Mixup-transformer: dynamic data augmentation for nlp tasks. *arXiv preprint arXiv:2010.02394*, 2020.

[112] Ryo Takahashi, Takashi Matsubara, and Kuniaki Uehara. Ricap: Random image cropping and patching data augmentation for deep cnns. In *Asian conference on machine learning*, pages 786–798. PMLR, 2018.

[113] Salma Taoufiq, Balázs Nagy, and Csaba Benedek. Hierarchynet: Hierarchical CNN-based urban building classification. *Remote Sensing*, 12(22):3794, 2020.

[114] David Tellez, Geert Litjens, Péter Bándi, Wouter Bulten, John-Melle Bokhorst, Francesco Ciompi, and Jeroen van der Laak. Quantifying the effects of data augmentation and stain color normalization in convolutional neural networks for computational pathology. *Medical image analysis*, 58:101544, 2019. https://doi.org/10.1016/j.media.2019.101544.

[115] Igor V Tetko, David J Livingstone, and Alexander I Luik. Neural network studies. 1. comparison of overfitting and overtraining. *Journal of chemical information and computer sciences*, 35(5):826–833, 1995. https://doi.org/10.1021/ci00027a006.

[116] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global, 2010.

[117] Marko Tscherepanow, Marco Kortkamp, and Marc Kammer. A hierarchical art network for the stable incremental learning of topological structures and associations from noisy data. *Neural Networks*, 24(8):906–916, 2011. https://doi.org/10.1016/j.neunet.2011.05.009.

[118] Paul E Utgoff. Incremental induction of decision trees. *Machine learning*, 4(2):161–186, 1989. https://doi.org/10.1023/A:1022699900025.

[119] Twan Van Laarhoven. L2 regularization versus batch and weight normalization. *arXiv preprint arXiv:1706.05350*, 2017. https://arxiv.org/abs/1706.05350.

[120] Ragav Venkatesan and Baoxin Li. *Convolutional neural networks in visual computing: a concise guide*. CRC Press, 2017.

[121] Mitko Veta, Josien PW Pluim, Paul J Van Diest, and Max A Viergever. Breast cancer histopathology image analysis: A review. *IEEE transactions on biomedical engineering*, 61(5):1400–1411, 2014.

[122] Mitko Veta, Paul J van Diest, Robert Kornegoor, André Huisman, Max A Viergever, and Josien PW Pluim. Automatic nuclei segmentation in H&E stained breast cancer histopathology images. *PLoS*, 2013. https://doi.org/10.1371/journal.pone.0070221.

[123] Zhou et al. Wang. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee, 2003.

[124] Benzheng Wei, Zhongyi Han, Xueying He, and Yilong Yin. Deep learning model based breast cancer histopathological image classification. In *2017 IEEE 2nd international conference on cloud computing and big data analysis (ICCCBDA)*, pages 348–353. IEEE, 2017. 10.1109/ICCCBDA.2017.7951937.

[125] Philip Wolfe. Convergence conditions for ascent methods. *SIAM review*, 11(2):226–235, 1969. https://doi.org/10.1137/1011036.

[126] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[127] et al. Xie. Explainable deep learning: A field guide for the uninitiated. *arXiv preprint arXiv:2004.14545*, 2020.

[128] Juanying Xie, Ran Liu, Joseph Luttrell IV, and Chaoyang Zhang. Deep learning based analysis of histopathological images of breast cancer. *Frontiers in genetics*, 10:80, 2019. https://doi.org/10.3389/fgene.2019.00080.

[129] Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Broeck. A semantic loss function for deep learning with symbolic knowledge. In *Int. conf. on machine learning*, pages 5502–5511, 2018.

[130] Qiang Xu, Yu Zeng, Wenjun Tang, Wei Peng, Tingwei Xia, Zongrun Li, Fei Teng, Weihong Li, and Jinhong Guo. Multi-task joint learning model for segmenting and classifying tongue images using a deep neural network. *IEEE journal of biomedical and health informatics*, 24(9):2481–2489, 2020. 10.1109/JBHI.2020.2986376.

[131] Yungang Zhang, Bailing Zhang, Wenjin Lu, Tuan D Pham, Xiaobo Zhou, Hiroshi Tanaka, Mayumi Oyama-Higa, Xiaoyi Jiang, Changming Sun, and Jeanne Kowalski. Breast cancer classification from histological images with multiple features and random subspace classifier ensemble. In *AIP Conference Proceedings*, volume 1371, pages 19–28. AIP, 2011.

[132] Yufeng Zheng, Clifford Yang, and Alex Merkulov. Breast cancer screening using convolutional neural network and follow-up digital mammography. In *Computational Imaging III*, volume 10669, page 1066905. International Society for Optics and Photonics, 2018. https://doi.org/10.1117/12.2304564.

[133] Weiming Zhi, Henry Wing Fung Yueng, Zhenghao Chen, Seid Miad Zandavi, Zhicheng Lu, and Yuk Ying Chung. Using transfer learning with convolutional neural networks to diagnose breast cancer from histopathological images. In *Int. C. on Neural Information Proc.*, pages 669–676. Springer, 2017. https://doi.org/10.1007/978-3-319-70093-9_71.

[134] Wang Zhou. Image quality assessment: from error measurement to structural similarity. *IEEE Trans. image proc.*, pages 600–613, 2004.

[135] Zhi-Hua Zhou. *Machine learning*. Springer Nature, 2021.

[136] Xinqi Zhu and Michael Bain. B-CNN: branch convolutional neural network for hierarchical classification. *arXiv preprint arXiv:1709.09890*, 2017.