TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

TUPAC	Tumor Proliferation Assessment Challenge 2016
WSI	Whole-slide image
BACH	BreAst Cancer Histology
CIFAR	Canadian Institue for Advanced Research
AMIDA13	Assessment of Mitosis Detection Algorithms 2013
VAT	Virtual adversarial training
MT	Mean teacher
DCTR	Deep Cotraning
GANs	Generative adversarial networks
KLD	Divergence de Kullback-Leibler
CE	Entropie croisée
H	Entropie
LReLU	Leaky Rectifier Linear Units

LISTE DES SYMBOLES ET UNITÉS DE MESURE

C	Nombre de classes
L	Nombre de couches dans un modèle
$\mathcal{C}$	Fonction de coût
$\mathcal{A}$	Base de données annotées
$\mathcal{U}$	Base de données non annotées
$\mathcal{D}$	Base de données d'entrainement
f(.)	Vecteur de prédiction du modèle f
$\mathrm{v}_i(x)$	Perception i de la donnée x
h(.)	Fonction de classification
T(.)	Fonction de transformation linéaire ou non linéaire
N	Nombre de données d'entrainement
N_a	Nombre de données d'entrainement annotées
N_{unl}	Nombre de données d'entrainement non annotées
θ	Ensemble des paramètres d'un modèle
$z, \widetilde{z}, \hat{u}, u$	Variables latentes
x	Image de la base d'entrainement
$p(y_i)$	Probabilité de la classe i (ground truth)
$p(\widetilde{y}_i)$	Probabilité que le modèle prédise la classe i
κ	Statistique de Kappa
σ_c	Coefficient de corrélation
α,β,γ,ϵ,ξ	Paramètres spécifique à chaque approche

INTRODUCTION

Mise en situation

Le cancer du sein est la forme de cancer la plus répandue chez les femmes. Selon la Société canadienne du cancer, 1 cancer sur 4 chez la femme est un cancer du sein (Société canadienne du cancer (2018)). Afin de pouvoir traiter efficacement cette maladie, il est primordial de pouvoir la dépister le plus tôt possible. Les traitements sont alors les plus efficaces et les séquelles liées à ces traitements sont moins lourdes. Le pronostic dépend de plusieurs critères comme le degré de différenciation (grade) et le degré d'extension de la tumeur (stade). À l'heure actuelle, le grade d'une tumeur dans le cas du cancer du sein est obtenu par l'analyse d'images histologiques de ladite tumeur. Un pathologiste va effectuer une observation sur un morceau de tissu provenant de la zone concernée, prélevé par biopsie. Ce morceau de tissu va ensuite être traité suivant différentes étapes (Veta *et al.* (2018b)) afin qu'il puisse être observé au microscope. L'attribution du grade d'un cancer par un pathologiste se fait suivant les critères de Nottingham (Simpson *et al.* (2000)). Cette étude permet de déterminer de quel type de cancer il s'agit et quels sont les traitements à utiliser. Les pathologistes doivent observer en détail des images à larges champs, images de centaines de millions de pixels, pour en extraire des zones caractéristiques, représentatives de l'état du patient. Depuis quelques années, de nombreux concours de traitement automatique de telles images ont vu le jour. Le domaine du traitement d'images a fait une avancée très importante au cours de ces dernières années. Les techniques basées sur l'apprentissage de réseaux de neurones profond ont permis d'obtenir des performances inégalées jusqu'à ce jour, notamment dans la segmentation et la reconnaissance d'images. Elles forment l'état de l'art de différentes compétitions médicales récentes (Wang *et al.* (2016), Chennamsetty *et al.* (2018) ou encore Paeng *et al.* (2017)). L'utilisation de tels algorithmes pour assister les médecins dans leur diagnostic leur permettrait de gagner du temps et ainsi leur permettre de traiter plus de patients.

Problèmatique

Les performances des algorithmes profonds sont en partie liées au nombre d'exemples qui leur sont présentés. Un plus grand nombre de données annotées permet d'apprendre une meilleure représentation du problème. Ainsi, plus on utilise d'exemples différents durant l'apprentissage, meilleurs sont les performances. Néanmoins, les images médicales annotées par des professionnels de la santé restent peu nombreuses. C'est une tâche chronophage et fastidieuse qui rend la création de bases de données annotées particulièrement coûteuses. Ces images ont besoin d'être annotées manuellement par des experts pour permettre aux différents algorithmes d'apprendre à reconnaître automatiquement les zones d'intérêt. Malgré l'arrivée des scanneurs numériques d'imagerie pleins champs (*Whole Slide Image*), qui ont permis de faciliter la lecture des images histologiques par les pathologistes, beaucoup d'images restent peu ou pas annotées. Une branche de l'apprentissage automatique est spécialisée dans l'extraction d'information des données sans annotations. Les données, même sans labels, peuvent être utilisées pour améliorer les performances qu'un algorithme a du problème. De nombreuses méthodes semi-supervisées, basées sur une petite quantité de données annotées et une grande quantité de données non annotées, ont été développées afin de tirer parti de la grande majorité des nouvelles données récoltées.

Objectif du projet

L'objectif de ce projet est de comparer l'efficacité de plusieurs approches semi-supervisées d'apprentissage profond pour la reconnaissance d'images histologiques concernant le diagnostic du cancer du sein. Cette technique permet à la fois d'utiliser une petite quantité de données, apportant une information sur les classes à distinguer, et une grande quantité de données, permettant de bonifier les performances de l'algorithme et d'affiner l'extraction des caractéristiques.

Nous allons analyser l'effet de l'augmentation des données annotées et non annotées sur les performances des approches *Ladder Network* (Rasmus *et al.* (2015)), *Virtual Adversarial Training* (Miyato *et al.* (2017)), *Mean Teacher* (Tarvainen & Valpola (2017)) et *Deep Co-training* (Qiao *et al.* (2018)). Cette comparaison sera effectuée à l'aide de plusieurs expériences en utilisant deux bases de données issues de compétitions récentes. La première concerne la base de données BACH (*Breast Cancer Histology Images*), centrée sur la prédiction du stade du cancer parmi 4 classes : Normal, Bénin, InSitu et Invasif. La deuxième, provenant de la compétition TUPAC (*Tumor Proliferation Assessment Challenge 2016*), se focalise sur la prédiction du grade du cancer. Pour cette deuxième base, nous nous sommes intéressés à la reconnaissance des cellules en cours de mitoses, qui est l'un des critères de Notthingam.

Cette étude propose d'analyser les effets de l'utilisation croissante de données annotées et non annotées sur les performances de ces différents modèles. Puis, d'observer plus en détails l'approche *Deep Co-Training* en analysant l'impact du nombre de modèles utilisées et de l'augmentation de leur diversité sur les résultats. Puis, pour finir, d'utiliser l'approche *BadGan* (Dai *et al.* (2017)) afin de générer des données supplémentaires non annotées pour l'apprentissage des différents modèles. Ces données seront exploitées dans une fonction de coût additionnelle dans l'objectif d'améliorer les résultats obtenus. Ce projet va contribuer, dans un premier temps, à effectuer une synthèse des approches semi-supervisées compétitives. Dans un deuxième temps, à effectuer différentes expérimentations de certaines de ces approches sur des bases de données médicales d'images histologiques. Enfin, dans ce projet, nous allons combiner l'utilisation de données synthétiques produit durant l'apprentissage d'un Badgan avec les approches semi-supervisées compétitives. Ces données seront utilisées en tant que données non annotées supplémentaires, dans le but d'améliorer les performances des différentes approches.

A long terme, l'utilisation d'approches de reconnaissance d'images automatique permettrait de réduire la charge de travail nécessaire à l'analyse des images par les pathologistes et d'améliorer

la qualité de la reconnaissance du cancer du sein. De plus, cette application n'est pas restreinte à l'analyse d'images histologiques pour le cancer du sein : elle peut être adaptée à l'ensemble des problématiques touchant à l'analyse à l'échelle microscopique de tissus vivants.

Structure du document

Dans un premier temps, nous nous sommes intéressés au contexte de la reconnaissance d'images histologiques à l'aide de modèles d'apprentissage profond, en expliquant succinctement comment fonctionnent ces types de modèles. La deuxième partie présente l'état de l'art dans le domaine de la reconnaissance semi-supervisée et définit le cadre de l'étude. La troisième partie concerne la méthodologie utilisée pour effectuer les différentes expériences. La dernière partie porte sur les résultats obtenus lors des différentes expériences ainsi que sur leur analyse.

CHAPITRE 1

CONTEXTE

1.1 Diagnostic des images histologiques pour le cancer du sein

Afin de diagnostiquer de manière précise un cancer du sein, il est nécessaire d'en prélever un échantillon par biopsie. Plusieurs étapes sont nécessaires pour pouvoir observer les échantillons de tissus au microscope (Veta *et al.* (2018b)). L'échantillon doit être découpé, traité dans une solution de fixateur pour prévenir sa décomposition et incrusté dans un bloc de paraffine. Des tranches de l'échantillon sont alors obtenues à l'aide d'un microtome pour pouvoir être observées sur une lame de microscope. Puis, l'échantillon est teinté à l'aide d'une coloration hematoxyline-éosine afin de mettre en évidence les noyaux en violet et le cytoplasme en rose. Les tissus ainsi préparés sont destinés aux pathologistes pour qu'ils puissent diagnostiquer le type et le grade de cancer. Ce diagnostic est primordial pour définir le traitement le plus adéquat pour le patient. Il existe plusieurs types de cancer du sein. Dans cette étude nous nous concentrerons principalement sur la reconnaissance des images histologiques parmi les quatre classes présentées pour le concours BACH : normal, bénin, carcinome in situ et carcinome invasif (voir Fig. 1.1). Les carcinomes sont des tumeurs issues d'un épithélium (surface de la peau, muqueuse ou glande). L'appellation 'in situ' indique que le cancer est contenu dans un canal galactophore (tubulure qui conduit le lait) et qu'il ne s'est pas propagé au reste du corps, contrairement au carcinome invasif.

Comme les organisateurs du concours l'indiquent, la catégorisation des échantillons est particulièrement importante dans le cas de la présence d'un carcinome, in situ ou invasif, car les traitements appliqués sont radicalement différents (Aresta *et al.* (2018)). Cette analyse est actuellement effectuée par des pathologistes expérimentés. Le traitement de ces échantillons de grande taille est fastidieux et requiert un certain temps passé sur chaque cas. De plus, les diagnostics sont souvent différents d'un spécialiste à l'autre. La base de données utilisée pour la compétition BACH (Aresta *et al.* (2018)) a été soumise à plusieurs experts. Les annotations

ont été obtenues à partir du consensus de deux professionnels. Les images annotées ont été présentées à un panel de trois pathologistes externes à la compétition qui ont obtenu une exactitude de 81.66% ($\pm$ 10.96) pour la reconnaissance. Cette intra-variabilitée entre les pathologistes peut s'expliquer par la complexité des images. Les tissus peuvent représenter des structures très différentes d'un échantillon à l'autre.

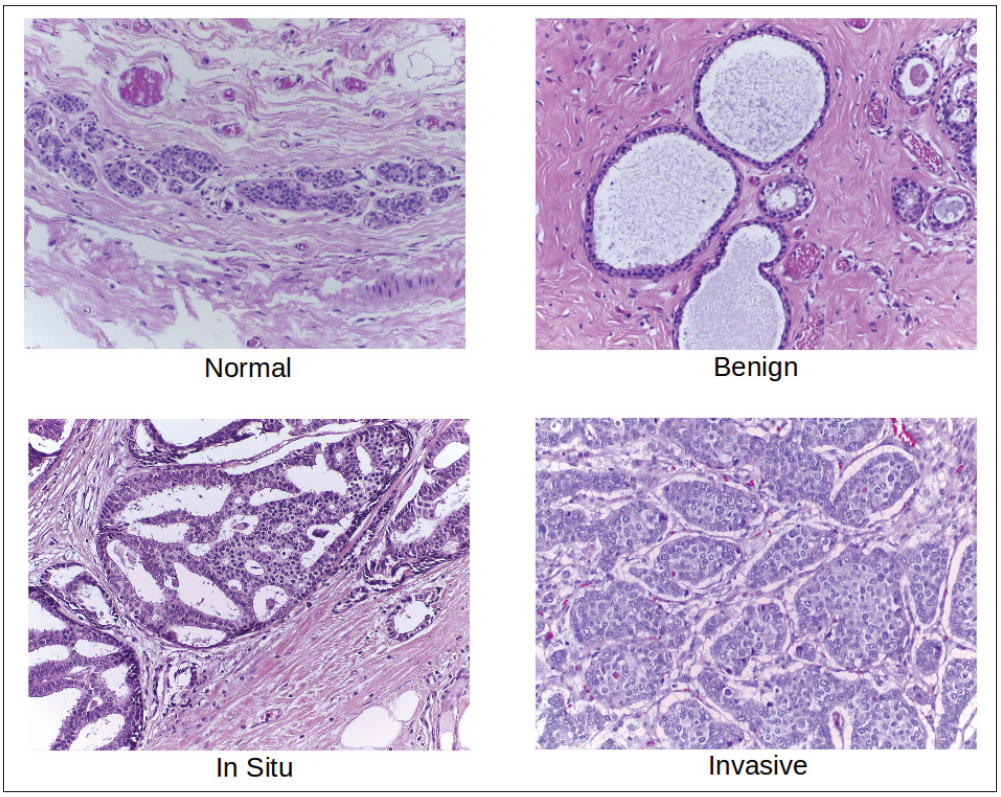


Figure 1.1 Exemple d'images de la base BACH

C'est également le cas pour l'analyse du grade. À l'heure actuelle, le grade d'une tumeur dans le cas du cancer du sein est obtenu à partir des critères de Nottingham (Elston & Ellis (1991)). Il existe trois critères qui permettent d'obtenir le grade d'un échantillon de tissu : la différenciation architecturale, le pléomorphisme nucléaire (forme des cellules) et l'activité mitotique. Chacun de ces critères est échelonné de 1 à 3 et le grade est obtenu en additionnant l'ensemble de ces scores (de 3 à 5, il s'agit du grade I, de 6 à 7 du grade II et de 8 à 9 du grade III). Dans ce projet, nous nous sommes intéressés au troisième critère, qui consiste à dénombrer les cellules

en cours de mitose dans les zones étudiées. La mitose est un phénomène naturel permettant la multiplication des cellules. Durant ce processus, une cellule mère va donner naissance à deux cellules filles. Le nombre de cellules en cours de division dans l'échantillon est particulièrement important pour connaître la progression de la tumeur. Si ce nombre est élevé (supérieur à 10 pour une zone de 2 mm^2), cela indique que la tumeur progresse très rapidement et on lui attribue la note maximale pour ce critère. Comme pour l'analyse du type de cancer, le dénombrement des cellules en cours de mitose est une tâche particulièrement complexe. En effet, le processus de mitose comporte plusieurs stades qui vont à chaque fois modifier l'apparence de la cellule et de son noyau.

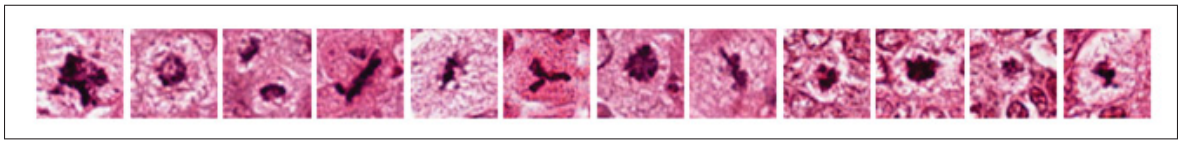


Figure 1.2 Exemple de cellules en cours de mitose à différents stades (Roux *et al.* (2014))

Ces différents stades doivent être pris en compte lors du dénombrement. Les pathologistes doivent ainsi faire attention à plusieurs types de cellules semblables à celles présentent sur la figure 1.2. De plus, il existe de nombreuses cellules dont l'apparence est très semblable aux cellules en mitose. Comme on peut le voir sur la figure 1.3, ces cellules dont le noyau est nécrosé, ou déformé par la compression des cellules voisines peuvent être assimilées à des cellules en cours de mitose. Enfin, l'aspect des échantillons n'est pas constant d'une coloration à l'autre. La quantité et la qualité des colorants utilisés peuvent varier d'un prélèvement à l'autre ce qui affecte la teinte des échantillons.

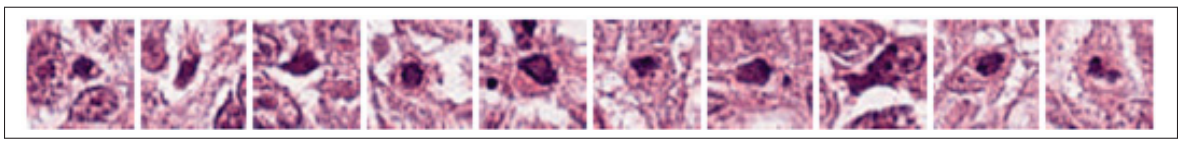


Figure 1.3 Exemple de cellules ressemblant aux cellules en cours de mitose (Chen *et al.* (2016))

Toutes ces aléas entraînent une recrudescence de faux positifs et de faux négatifs qui rendent le dénombrement de la part des spécialistes assez hétérogène. Pour étudier ce problème, nous avons utilisé la base de données issue du concours TUPAC (Veta *et al.* (2018a)). Une partie de la base à notre disposition est elle-même issue du concours AMIDA13 (*Assessment of Mitosis Detection Algorithms 2013*) (Veta *et al.* (2015)). Pour cette partie, un tableau récapitule le nombre de cellules en cours de mitose détectées par deux pathologistes. Pour la base de test par exemple, on peut constater un écart de 15% (101 détections) entre les deux experts parmi les cellules détectées. L'objectif de ce projet est d'utiliser des algorithmes basés sur l'apprentissage profond afin de voir dans quelles mesures ces techniques permettraient d'assister les pathologistes dans le traitement de ces images.

1.2 Approches conventionnelles

De manière générale, la reconnaissance automatique d'images (voir Fig. 1.4) est effectuée en trois grandes étapes : le prétraitement, l'extraction des caractéristiques et la classification.

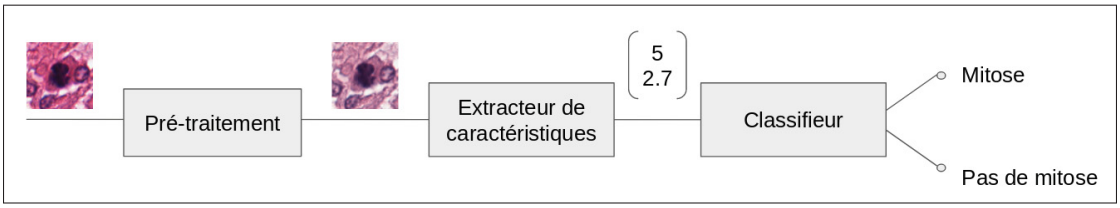


Figure 1.4 Exemple de reconnaissance automatique standard d'une image

Le prétraitement consiste à mettre en forme les images brutes d'entrée pour qu'elles soient traitées de la même manière dans les étapes suivantes. Ce prétraitement peut prendre plusieurs formes suivant le type d'images étudiées. Par exemple, dans le cas d'images histologiques, il est courant de normaliser la teinte entre les différentes images pour obtenir une teinte uniforme. La tâche de prétraitement peut également consister à extraire certaines parties des images d'origine pour focaliser l'apprentissage sur ces régions, comme par exemple en extrayant des patches centrés sur le noyau des cellules. L'extraction de caractéristiques consiste à transformer l'image d'entrée, généralement de très grande dimension, dans un espace de représentation plus petit. Du

point de vue de l'algorithme, une image en couleur de 64 x 64 pixels est une donnée représentée dans un espace de 12 288 dimensions (64 x 64 x 3). L'extraction de caractéristiques permet de calculer un certain nombre d'attributs propres à l'image qui en résumant l'information qu'elle contient suivant différents critères. Par exemple, une image de cellule peut être représentée par son périmètre et la moyenne des intensités des canaux de couleur. On aurait ainsi transformé un point représenté dans un espace de 12 288 dimensions dans un espace de 2 dimensions. Le choix des critères est primordial pour différencier les données selon le problème à résoudre.

La classification consiste à différencier ces images projetées dans un espace de plus petite dimension parmi les différentes classes du problème. Par exemple, si l'on tente de classer automatiquement les cellules qui sont en cours de mitose, le module de classification va associer les représentations réduites de chaque image à la classe correspondante. Le but sera ensuite de calculer la meilleure séparation entre les différentes classes pour prédire la classe de futures images. Durant le début des années 2000, la majorité des systèmes de reconnaissance s'appuyait sur une extraction des caractéristiques basée sur l'apparence. Comme le présente Doyle *et al.* (2007), les modèles étaient principalement basés sur l'extraction d'un grand nombre de caractéristiques en rapport avec la texture, la morphologie et la couleur des échantillons. Veta *et al.* (2015) présente les principales caractéristiques prises en compte dans le cas de la caractérisation de cellules issues d'images histologiques. Il s'agit du bilan de la compétition AMIDA13 dont le but était de détecter les cellules en cours de mitose à partir d'images histologiques pour le cancer du sein. Concernant la caractérisation de la texture, deux principales techniques ont été utilisées : les motifs binaires locaux, qui consistent à représenter les pixels en fonction de leurs voisins (Wang & He (1990)), et les caractéristiques d'Haralick (Haralick *et al.* (1973)) basées sur la répétition des motifs. En ce qui concerne la caractérisation de la morphologie, on retrouve de manière très fréquente l'utilisation des mesures de l'aire, du périmètre, de la longueur des axes majeurs et mineurs, de l'excentricité et de l'orientation. Pour ce qui est des couleurs, une majorité d'approches utilisent la moyenne des intensités et leur ratio avec les trois canaux de couleurs, critère également récupéré sur l'extraction des teintes d'éosine et d'hématoxyline. Parmi les modules de classification les plus utilisés, on retrouve la

machine à vecteurs de support (SVM) (Cortes & Vapnik (1995)) et la forêt d'arbres décisionnels (Random forests) (Breiman (2001)) encore populaire de nos jours.

1.3 Approches basées sur l'apprentissage profond

Avec l'augmentation des capacités de calcul, les algorithmes automatiques permettant d'extraire les caractéristiques des images sont devenus de plus en plus utilisés. Le principe consiste à apprendre automatiquement à un modèle à extraire les caractéristiques d'une image afin d'optimiser sa classification. Les algorithmes basés sur l'apprentissage profond sont particulièrement performants pour cette tâche. Ils ont permis d'obtenir les résultats faisant l'état de l'art dans de multiples domaines, en particulier dans la reconnaissance d'images. Nous allons détailler ici comment fonctionne ce type d'algorithme.

Tableau 1.1 Architecture du modèle *Conv-Large*

Layers	Input size
3x3 Conv padding 1 Conv 128 LReLU + BN	3 x 32 x 32
3x3 Conv padding 1 Conv 128 LReLU + BN	128 x 32 x 32
3x3 Conv padding 1 Conv 128 LReLU + BN	128 x 32 x 32
2x2 maxpool stride 2	128 x 32 x 32
DropOut (0,5)	128 x 16 x 16
3x3 Conv padding 1 Conv 256 LReLU + BN	128 x 16 x 16
3x3 Conv padding 1 Conv 256 LReLU + BN	256 x 16 x 16
3x3 Conv padding 1 Conv 256 LReLU + BN	256 x 16 x 16
2 x 2 maxpool stride 2	256 x 16 x 16
DropOut (0,5)	256 x 8 x 8
3x3 Conv padding 0 Conv 512 LReLU + BN	256 x 8 x 8
3x3 Conv padding 1 Conv 256 LReLU + BN	512 x 6 x 6
3x3 Conv padding 1 Conv 128 LReLU + BN	256 x 6 x 6
Average Pooling	128 x 6 x 6
Fully connected	128 -> 10

Pour l'entraînement de ce type de méthode, il est courant que les algorithmes combinent les rôles d'extracteur de caractéristiques et de classification. Le premier module consiste à transformer l'image d'entrée de très grande dimensions dans un espace plus petit, regroupant les données les plus représentatives de l'image. Pour se faire, les algorithmes d'apprentissage profond sont

composés de plusieurs couches qui vont être associées dans un ordre particulier. Le tableau 1.1 représente l'architecture de l'algorithme utilisé pour obtenir les performances de plusieurs approches d'apprentissage semi-supervisé faisant l'état de l'art sur la base CIFAR-10 (*Canadian Institue for Advanced Research*) (Krizhevsky & Hinton (2009)). La première colonne énumère les différentes couches qui composent une architecture d'apprentissage profond. Ces différentes couches sont décrites plus en détails dans l'énumération suivante :

- Les couches convolutives consistent à appliquer un certain nombre de filtres sur les images en entrée. Elles forment la base du modèle d'extraction. Les valeurs des différents filtres sont modifiées au cours de l'apprentissage afin d'améliorer la classification des images. Ces couches permettent de produire un certain nombre de cartes de caractéristiques à partir des images en entrée.

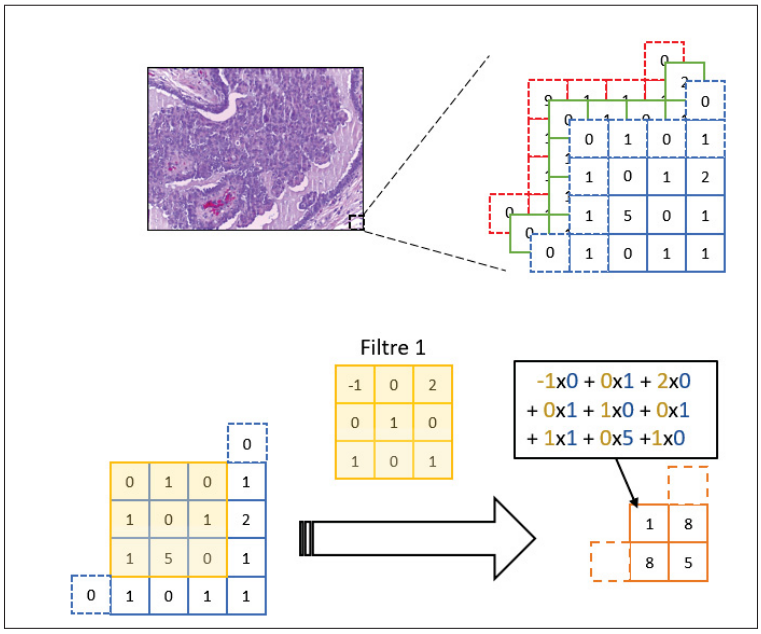


Figure 1.5 Exemple de l'application d'une convolution sur une image

Le schéma 1.5 représente un exemple d'application d'un filtre d'une couche convolutive sur une image histologique. Une image histologique se compose en trois matrices représentant les intensités des couleurs rouge, vert et bleu appelées canaux. La schématisation a été

simplifiée sur cette figure : seul un filtre est appliqué sur le canal bleu. En pratique, une couche convolutive va appliquer un filtre différent (cellule en jaune) sur chacun des canaux de couleurs. Le score calculé sur la carte de caractéristiques (ici en orange) est obtenu par la somme des scores obtenus par les filtres sur les différentes images en entrée. Ces couches possèdent plusieurs paramètres comme on peut le voir sur le tableau 1.1. Parmi les principaux, on retrouve le *kernel size*, le *padding*, le *stride* et le nombre de cartes de caractéristiques en sortie. Le *kernel* fait référence à la taille du filtre utilisé (généralement 3x3 pixels). Le *padding* correspond au nombre de pixels supplémentaire que l'on applique aux images d'entrée avant d'appliquer les différents filtres. L'ajout de ces pixels permet dans notre cas de conserver la taille des cartes de caractéristiques. Le *stride* correspond au pas du filtre. Habituellement égale à 1, il permettre de balayer l'image plus ou moins grossièrement par les filtres.

- Les couches de normalisation, aussi appelées *batchnorm* (BN), permettent dans un premier temps de régulariser le modèle en contraignant les cartes de caractéristiques à avoir une distribution normale (moyenne 0 écart type de 1). Cette normalisation est calculée à partir de la moyenne du batch μ_{batch} et sa variance v_{batch} avec l'équation :

$$x_{norm} = \frac{x_i - \mu_{batch}}{\sqrt{v_{batch}}} \tag{1.1}$$

  Le deuxième rôle de cette couche de normalisation vise à appliquer une transformation linéaire aux cartes de caractéristiques au travers de deux paramètres γ et β qui sont mis à jour lors de l'entraînement. Le batch normalisé et mise à l'échelle $\overline{x_i}$ est défini par :

$$\overline{x_i} = \gamma x_{norm} + \beta \tag{1.2}$$

  Cela ajoute de la flexibilité au modèle et de manière générale, ce type de couche a permis d'accélérer le temps d'entraînement et d'augmenter les performances des modèles (Ioffe & Szegedy (2015)).

- Les couches de non-linéarité sont indispensables à l'apprentissage d'un modèle profond complexe. Sans elles, la répétition des couches convolutives reviendrait à effectuer une

transformation linéaire sur l'image d'entrée. Il existe de nombreuses non-linéarités, parmi les plus fréquemment utilisées on retrouve le Rectifier Linear Units (ReLU) (Nair & Hinton (2010)) définie tel que :

$$f_{ReLU}(x) = \begin{cases} x & \text{si } x > 0 \\ 0 & \text{sinon.} \end{cases} \tag{1.3}$$

et le leaky ReLU (Maas *et al.* (2013)) défini par :

$$f_{LReLU}(x) = \begin{cases} x & \text{si } x > 0 \\ 0.01x & \text{sinon.} \end{cases} \tag{1.4}$$

L'assemblage couche convolutive et couche de non-linéarité permet d'obtenir des transformations non linéaires indispensables à l'apprentissage de caractéristiques complexes.

- La couche de *maxpooling* est l'une des méthodes utilisées pour réduire la taille des cartes de caractéristiques. Elle consiste à ne garder que les valeurs maximales des fenêtres étudiées (voir Figure 1.6).

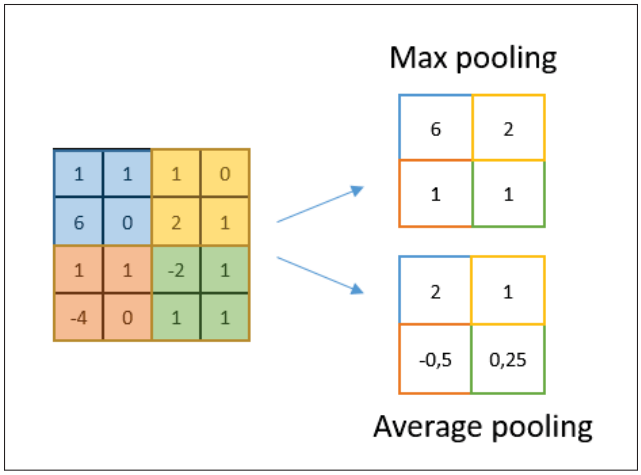


Figure 1.6 Exemple de couches de pooling

- Le *dropout* (Srivastava *et al.* (2014)) est une couche qui permet de réduire le surapprentissage d'un modèle. Elle consiste à désactiver (mettre à 0) les cartes de caractéristiques aléatoirement durant l'apprentissage. Cela a pour effet de rendre le modèle plus résilient.

- *L'average pooling* consiste à générer une valeur moyenne à partir des valeurs d'une zone en particulier (voir Fig. 1.6). Elle se situe généralement à la fin des couches convolutives pour fournir un vecteur de caractéristiques à classer. À ce stade, une image en entrée sera alors représentée par un vecteur regroupant les différentes caractéristiques extraites.
- Les couches entièrement connectées sont placées en fin de modèle afin de classer les caractéristiques extraites des images. Lors d'un apprentissage, on configure la dernière couche entièrement connectée pour qu'elle corresponde aux différentes classes que l'on veut reconnaître.

L'entraînement va ensuite consister à optimiser les différents paramètres du modèle (filtres des couches convolutives et les paramètres des couches de normalisation) à partir des différentes erreurs de classification issues de l'algorithme. La sortie de l'algorithme est généralement constituée d'une fonction *softmax* qui permet de convertir les valeurs de sortie en probabilité. Cette couche effectue le calcul suivant :

$$p(\tilde{y}_i) = \frac{e^{\tilde{y}_i}}{\sum_j^C e^{\tilde{y}_j}} \tag{1.5}$$

Dans cette équation, $\tilde{y}_i$ représente la réponse du modèle pour la classe i et $p(\tilde{y}_i)$ la probabilité de cette même classe. Par exemple, dans le cas de la reconnaissance d'images histologiques en trois classes, on peut imaginer que le modèle *Conv-Large* produit comme prédiction un vecteur [1.2 0.1 -0.5]. Le *softmax* va permettre de convertir ces valeurs d'activation en probabilité, soit [0.66 0.22 0.12] dans notre cas. On peut ainsi interpréter la réponse du réseau comme étant de 66% pour la première classe, de 22% pour la deuxième. et de 12% pour la troisième. L'entraînement supervisée se passe en deux phases : on présente une image en entrée dont on connaît la classe de l'image et on récupère les prédictions du modèle en sortie. Les prédictions sont comparées au label que l'on attendait et on calcule une erreur à partir d'une fonction de coût. Il en existe plusieurs, la plus répandue dans le cas d'entraînement supervisé est l'entropie

croisée, calculée entre deux distributions de probabilité y et $\tilde{y}$ par :

$$\mathcal{C}E(y,\tilde{y}) = -\sum_{i}^{C} p(y_i) log(p(\tilde{y}_i)) \tag{1.6}$$

Par exemple, si l'image présentée au modèle *Conv-Large* était de la classe 2, l'entropie croisée devrait être calculée entre le vecteur de prédiction $p(\tilde{y})$ de [0.66 0.22 0.12] et le label attendu $p(y)$ de [0 1 0]. La deuxième étape consiste à calculer le gradient de cette erreur par rapport aux différents paramètres du modèle. Les paramètres sont mis à jour par rétropropagation du gradient, qui est une méthode qui permet de calculer le gradient de l'erreur par rapport à chaque paramètre. Le but étant de modifier légèrement chaque paramètre de sorte à minimiser l'erreur entre la prédiction du modèle et la réponse espérée. Cette mise à jour des paramètres est encadrée par une fonction d'optimisation de descente de gradient comme l'algorithme du gradient stochastique (SGD). L'apprentissage consiste à rechercher les meilleurs paramètres du modèle qui permettent de réduire l'erreur d'apprentissage tout en ayant une meilleure généralisation.

L'apprentissage profond est un concept relativement ancien (Wang & Raj (2017)). Les modèles basés sur l'apprentissage profond sont devenus progressivement majoritaires dans les différents concours de reconnaissance, notamment dans le domaine médical. On peut par exemple constater qu'en 2013, pour le concours AMIDA13, moins de 10% des participants ont employé ce type de modèle. Lors du concours de reconnaissance d'images histologiques BACH, en 2018, la totalité des participants a utilisé au moins un module d'apprentissage profond, que ce soit pour l'extraction de caractéristiques, la classification ou les deux.

Ces types de modèles profonds sont particulièrement intéressants pour les performances qu'ils peuvent produire. Dans le domaine médical, et plus particulièrement pour la reconnaissance d'images histologiques pour le cancer du sein, les approches profondes ont obtenues la première place dans les concours depuis 2012 ((Cireşan *et al.* (2013)) (Paeng *et al.* (2017)) (Wang *et al.* (2016)) (Chen *et al.* (2016)) (Chennamsetty *et al.* (2018))).

De manière générale, les vainqueurs des concours ont utilisé une ou plusieurs architectures particulièrement performantes sur d'autres compétitions de reconnaissance populaire d'images comme CIFAR ou ImageNet (Deng *et al.* (2009)). Ces modèles profonds permettent d'apprendre de bout en bout à extraire et à classer. Les caractéristiques extraites sont alors déterminées automatiquement lors de l'apprentissage pour optimiser la classification.

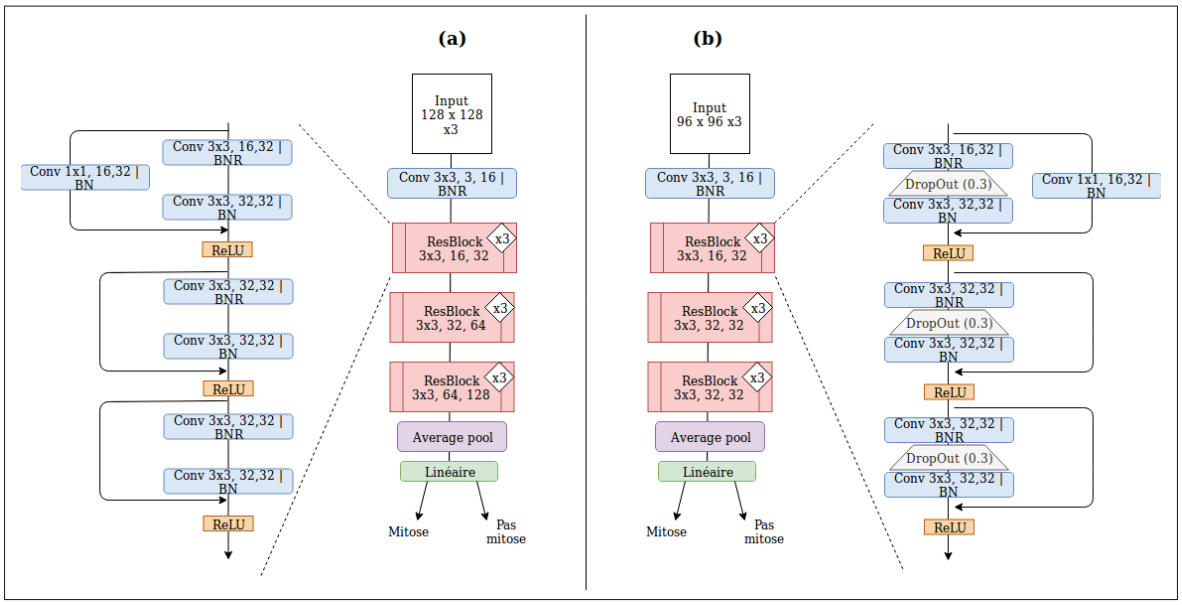


Figure 1.7 Architectures des réseaux victorieuses du concours TUPAC, (a) 1re place (Paeng *et al.* (2017)) et (b) 2de place (Zerhouni *et al.* (2017))

Lors de la compétition TUPAC16 (Veta *et al.* (2018a)) par exemple, toutes les équipes ont proposé des approches basées sur des réseaux de neurones convolutifs profonds. Les approches proposées par Paeng *et al.* (2017) et Zerhouni *et al.* (2017) ont obtenu les deux meilleures places du challenge avec des scores F1 (cf. section 3.4) respectif de 65.2 et 64.8 sur la base de test du concours. Les vainqueurs de la compétition ont utilisé une architecture à base de blocs résiduelles similaires à ceux employés dans le ResNet (He *et al.* (2016)) représentée par l'architecture (a) de la figure 1.7. Ils ont utilisé des patches de 128 x 128 pixels afin de produire une carte de probabilité pour chaque image. Les cellules pertinentes sont ensuite validées par un réseau similaire. L'équipe arrivée deuxième a utilisé une architecture inspirée du WideResnet (Zagoruyko & Komodakis (2016)) pour classer des patches de 96 x 96 pixels représentés par

la partie (b) sur la figure 1.7. Les deux meilleures approches de la compétition BACH (Aresta *et al.* (2018)) ont toutes les deux obtenues 87 % d'exactitude (cf. section 3.4) sur la base de test du concours. La figure 1.8 représente la méthode utilisée par l'une des approches victorieuse de la compétition.

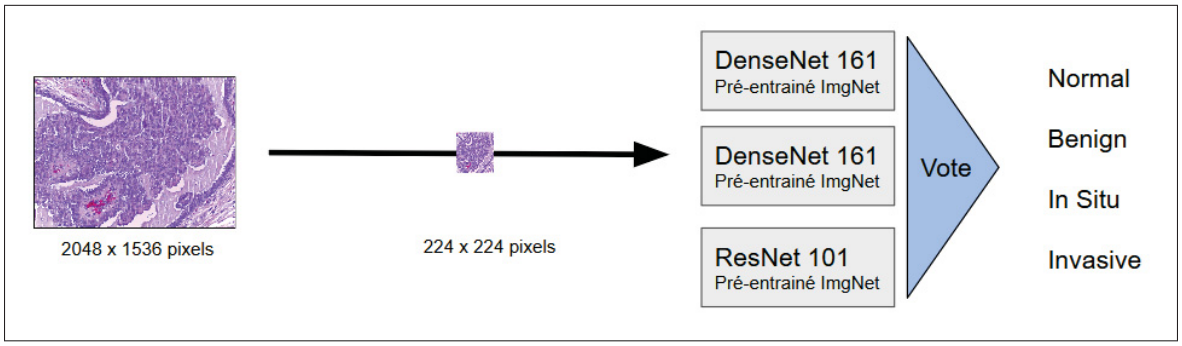


Figure 1.8 Schématisation d'une des approches vainqueur de la compétition BACH (Chennamsetty *et al.* (2018))

La première approche, proposée par Chennamsetty *et al.* (2018), consiste à redimensionner les images de 2048 x 1536 pixels en image de 224 x 224 pixels en utilisant deux techniques de normalisation (à partir de la moyenne et de l'écart type des images issues d'ImageNet et à partir de la moyenne et de l'écart type de la base BACH). La prédiction de la classe de l'image est obtenue par vote majoritaire de trois réseaux profonds pré-entraînés sur ImageNet (Deng *et al.* (2009)) : deux DensNet 161 (Huang *et al.* (2017)) et un ResNet 101 (He *et al.* (2016)). La deuxième approche victorieuse consiste à extraire des patches de 1495 x 1495 pixels qui sont ensuite redimensionnés en patch de 299 x 299 pixels. La reconnaissance a été effectuée à l'aide d'un réseau Inception-Resnet-v2 (Szegedy *et al.* (2017)) pré-entraîné sur Imagenet (Deng *et al.* (2009)).

1.4 Discussion

L'apprentissage profond permet d'obtenir des performances particulièrement élevées, notamment dans l'analyse d'images médicales. Néanmoins, ces performances sont liées à la quantité

d'images annotées disponibles. Un plus grand nombre d'images permet au modèle d'obtenir de meilleurs performances sur la tâche. L'arrivée des scanneurs numériques à larges champs a permis d'obtenir une grande quantité d'images histologiques. Toutefois, dans le domaine médical, l'annotation de ces images est une tâche complexe et onéreuse. Pour la reconnaissance d'images histologiques pour le cancer du sein par exemple, les bases de données constituées pour les concours TUPAC (Veta *et al.* (2018a)), AMIDA13 (Veta *et al.* (2015)), ICPR12 (Roux (2014)) ou encore BACH (Aresta *et al.* (2018)) ont dû être annotées par deux professionnels pour limiter les erreurs. Comme l'indique Mikto Veta (Veta *et al.* (2015)), membre organisateur de plusieurs de ces concours, la limite des performances des modèles actuels peut s'expliquer par le manque de grandes bases publiques annotées. Afin de pallier cette limite, nous nous sommes intéressés aux techniques d'apprentissages profonds basées sur l'utilisation conjointe d'images annotées et non annotées. Nous avons donc orienté notre étude sur les techniques utilisant les données faiblement annotées afin de tirer parti des grandes quantités de données qui sont sous-exploitées.

CHAPITRE 2

MÉTHODES POUR L'APPRENTISSAGE SEMI-SUPERVISÉ

2.1 Survol des approches d'apprentissage faiblement supervisé

L'expression 'faiblement annotée' peut paraître confuse, notamment dans le cas de l'apprentissage automatique. Comme le fait remarquer Zhia-Hua Zhou dans son article (Zhou (2017)), cette formule peut faire référence à trois cas de figure, souvent combinés. La figure 2.1 résume ces différents cas. L'apprentissage à partir de données faiblement annotées peut dans un premier temps faire référence au cas où seulement une partie de la base de données, généralement petite, est annotée. La supervision est alors qualifiée d'incomplète. L'auteur mentionne deux grandes familles dans ce cas : la prise en compte ou non d'un annotateur, faisant office d'oracle. Si on prend en compte l'intervention humaine dans la supervision, le but sera de présenter à cet annotateur les données sans annotations qu'il serait le plus pertinent d'annoter. Cette pertinence fait l'objet d'un domaine de recherche appelé apprentissage actif. De manière naïve, on pourrait par exemple solliciter l'expert sur les images que le modèle n'arrive pas à prédire de manière catégorique (Zhou *et al.* (2017)). D'un autre côté, l'apprentissage est effectué uniquement à partir des données.

La première famille d'approche, qui est la plus répandue, consiste à utiliser les données non annotées pour affiner l'apprentissage. Une fois l'algorithme entraîné, il est testé sur une base de test. Il s'agit de l'apprentissage semi-supervisé. La deuxième famille d'approches utilise les données de test comme données non annotées lors de l'entraînement, et le test est effectué sur ces mêmes images. Le *pseudo-labelling* (Lee (2013)) peut être utilisé à cette intention. Le but de cette approche vise à attribuer une annotation hypothétique aux données non annotées par l'algorithme pour qu'elles puissent être utilisées de la même manière que les données annotées. La distinction décrite dans l'article (Zhou (2017)) n'est pas ferme. Il est tout à fait envisageable d'effectuer un mélange de plusieurs types de supervision lors de l'apprentissage.

Pour l'utilisation d'images faiblement annotées, un deuxième cas de figure pourrait faire référence à la situation où les données sont annotées de manière globale. La supervision est alors qualifiée d'inexacte. La base de données pour le concours BACH entre dans cette catégorie. Seul le label de l'image complète est connu. Une des méthodes fréquemment utilisée pour ce type de supervision est l'apprentissage d'instances multiples (Foulds & Frank (2010)). Dans ce cas, les images sont considérées comme des sacs et les patches que l'on extrait comme des instances.

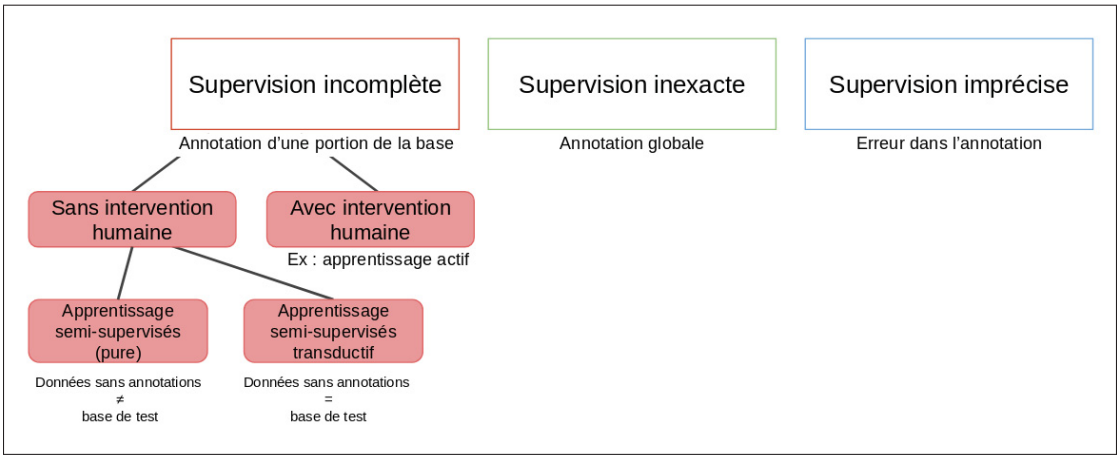


Figure 2.1 Schématisation de l'apprentissage à partir de données faiblement annotées (Zhou (2017))

Enfin, la formule 'faiblement annotée' peut évoquer l'imprécision dans l'attribution des labels. Dans ce cas, les annotations attribuées ne représentent pas toujours le label réel. Cette situation est courante dans le domaine du traitement d'images médicales. Ce cas de figure est traité dans la littérature en tant que 'label bruité'. Dans leur bilan (Frénay *et al.* (2014)), Frenay et Kaban répertorient trois grandes familles d'approches permettant de lutter contre ce phénomène.

Tout d'abord, la première famille concerne le surapprentissage. Les techniques qui permettent de limiter la mémorisation des données d'entraînement restreignent l'influence des exemples dont le label n'est pas correct. Parmi ces méthodes, on retrouve l'utilisation de modèles possédant un plus petit nombre de paramètres, l'utilisation de couche de *dropout*, les techniques d'arrêt

prématuré de l'apprentissage ou encore l'utilisation de pénalité (norme L1 ou L2) sur les paramètres appris.

La deuxième approche consiste à effectuer un tri dans les données d'apprentissage à partir de connaissances a priori. Ces techniques varient d'un problème à l'autre, le but étant de retirer les données ayant le plus de chance d'être bruitées. Cette famille suppose que certaines données ont plus ou moins de chance de correspondre au label qui leur a été attribué. Pour l'apprentissage à partir d'images histologiques, une pratique courante (Hou *et al.* (2016)) (Zanjani *et al.* (2018)) vise à éliminer les patches qui possèdent un certain pourcentage de pixels blancs (c.-à-d. les patches qui possèdent le moins de tissu). On retrouve aussi des méthodes plus globales comme la recherche de valeur aberrante dans les distributions des données, notamment à l'aide de méthode de partionnement de données (Muhlenbach *et al.* (2004)). Cette recherche peut aussi être effectuée durant l'apprentissage du modèle en retirant les patches que le modèle ou les modèles ne classent pas correctement (Hou *et al.* (2016)).

La dernière approche consiste à prendre en compte le bruit présent dans l'annotation. Dans cette catégorie on retrouve des approches visant à inférer sur le degré de confiance qu'une prédiction peut avoir (DeVries & Taylor (2018)). On retrouve aussi des approches comme celle proposée dans les travaux de Szegedy *et al.* (2016) dont le but est d'assouplir la fonction de coût. Au lieu de pousser le modèle à produire les prédictions catégoriques (0% ou 100%), on prend en compte l'imprécision dans l'annotation en pénalisant moins l'erreur de prédiction (par exemple, en déplaçant l'intervalle d'objectif à [0.1 0.9]).

Dans le cadre de cette étude, nous nous plaçons dans le cas d'une supervision incomplète et plus particulièrement dans le cas d'un apprentissage semi-supervisé pur. Néanmoins, nous avons aussi affaire à une supervision imprécise plus ou moins importante. Pour les bases de données TUPAC et BACH, l'annotation a été effectuée par le consensus de deux spécialistes. Cette supervision imprécise est particulièrement présente pour la base BACH (lorsque l'on divise les images en patches) et les stratégies mises en place pour lutter contre l'imprécision des annotations sont traitées plus en détail dans la section 3.3.2. Après avoir spécifié le cadre

de notre étude, nous allons présenter plus en détail comment utiliser les données annotées et non annotées pour l'apprentissage d'algorithmes automatiques.

2.2 Hypothèses de l'apprentissage semi-supervisé

L'objectif de l'apprentissage semi-supervisé est de pouvoir améliorer les décisions d'un modèle en faisant intervenir des images sans annotations. Dans leur livre (Chapelle *et al.* (2006)), les auteurs mettent en évidence plusieurs hypothèses qui permettent d'appliquer un tel type d'apprentissage. Ces hypothèses supposent un certain nombre de relations entre les espaces de départ que représentent les images avec l'espace d'arrivé, celle des annotations. L'hypothèse principale est celle de *smoothness*.

«Semi-supervised smoothness assumption : If two points x1 , x2 in a high-density region are close, then so should be the corresponding outputs y1 , y2.» (Chapelle *et al.* (2006))

Cette hypothèse suppose une certaine relation qu'il est possible d'imaginer entre les données et leurs annotations. S'il n'y a pas de corrélation entre les deux, l'utilisation des données non annotées devient alors inefficace. Comme l'indiquent Chapelle *et al.* (2006), cette hypothèse met en évidence que cette relation est plus probable dans les zones de forte densité, et a contrario, elle est moins probable dans les zones de faible densité.

Par ailleurs, il faut noter que cette relation n'est pas nécessairement directe. Une autre hypothèse importante émise par Chapelle *et al.* (2006) porte sur l'existence d'un espace, différent de l'espace de départ dans lequel il est possible d'utiliser certains algorithmes d'apprentissage semi-supervisés.

«Manifold assumption : The (high-dimensional) data lie (roughly) on a low-dimensional manifold.» (Chapelle *et al.* (2006))

Les points x1 et x2, ainsi que les points y1 et y2 peuvent être projetés dans des espaces de plus faible dimension où cette hypothèse s'applique.

Les deux prochaines hypothèses sont des cas particuliers qui découlent de la première supposition. Ils supposent que les points de départ forment des clusters dans un certain espace de représentation. Si l'on se fit au premier postulat, les points dans les zones de haute densité, dans un espace de représentation judicieusement choisi, devraient avoir la même classe. Dans ce cas, on peut imaginer la formation de zones de plus ou moins forte densité. Ce qui conduit aux deux prochaines hypothèses.

«Cluster assumption : If points are in the same cluster, they are likely to be of the same class. (Chapelle *et al.* (2006))»

«Low density separation : The decision boundary should lie in a low-density region. (Chapelle *et al.* (2006))»

Les données non annotées appartenant au même cluster devraient ainsi avoir la même classe. Et ces mêmes ensembles sont délimités par des régions de faible densité de données.

2.3 Modèles conventionnels

Cette partie présente un aperçu des familles d'approches fréquemment utilisées avant l'arrivée de l'apprentissage profond. Parmi les premières méthodes d'apprentissage semi-supervisées, on retrouve le *self-training*. Cette technique permet de prendre en compte les données sans annotations lors de l'entraînement d'algorithmes de reconnaissance. Elle consiste à annoter les données sans annotations par l'intermédiaire d'un algorithme automatique. On entraîne dans un premier temps le modèle à classer les données annotées. Une fois paramétré, on utilise le modèle pour prédire le label des données sans annotations. Ces dernières dont le modèle est confiant (supérieur à un seuil à définir) se voient attribuer le label prédit. L'algorithme est alors réentraîné à partir des données annotées ainsi qu'avec les nouvelles données annotées. On renouvelle alors l'opération pour un certain nombre d'itérations. Cette méthode très simple est particulièrement efficace lorsque l'on se trouve dans un espace de représentation où les classes sont nettement séparées (pas ou peu de chevauchement entre les différentes classes). Si ce n'est

pas le cas, certaines données vont alors se voir attribuer le mauvais label, ce qui va nuire à la classification.

Une des alternatives pour pallier cette condition a été d'utiliser l'avis de plusieurs algorithmes de classification pour attribuer des labels aux données non annotées.

L'approche originel de Blum & Mitchell (1998) consiste à classer des pages web représentant des cours. Cette classification est effectué à partir des mots clés présents sur la page elle-même d'une part et dans l'hyperlien d'autre part. Ces deux différentes perceptions des données sont utilisées pour tirer partie des données sans annotations. L'algorithme 2.1 résume la formulation du co-training décrite dans les travaux de Blum & Mitchell (1998).

Algorithme 2.1 Co-training proposé par Blum et Mitchell (Blum & Mitchell (1998))

Input : *Classifieurs* h_1 *et* h_2*, base de données annotées* $\mathcal{A}$*, base de données non annotées* $\mathcal{U}$*, paramètre* k *et* u
Output : *Classifieurs* h_1 *et* h_2 *entraînées*
for k **do**
    Entraîner h_1 sur la perception v_1 de $\mathcal{A}$
    Entraîner h_2 sur la perception v_2 de $\mathcal{A}$
    Ajouter aléatoirement des exemples de $\mathcal{U}$ dans $\mathcal{U}'$ tel que $\|\mathcal{U}'\| = u$
    for $h \in \{h_1, h_2\}$ **do**
        Prédire les α données positives de $\mathcal{U}'$ dont h est le plus confiant
        Prédire les β données négatives de $\mathcal{U}'$ dont h est le plus confiant
        Ajouter ces nouvelles instances annotées dans $\mathcal{A}$
    end
end

Dans ce cas, on considère que chaque page x peut être perçue de deux manières : $v_1(x)$ et $v_2(x)$. Ces deux perceptions sont ensuite utilisées pour entraîner deux modules de classification indépendante h_1 et h_2. On extrait ensuite un certain nombre α d'exemples prédits comme positifs et β exemples prédits comme négatifs dont les prédictions sont les plus élevées pour chaque module de classification. Ces pages sont alors ajoutées dans la base de données annotées. Ces étapes sont renouvelées un certain nombre k fois, déterminé en fonction des résultats obtenus sur la base de validation. L'intuition sous jacente à cette méthode vise à utiliser différents

points de vue sur les données annotées pour affiner les représentations des différentes classes à partir des données sans annotations. La figure 2.2 présente une schématisation très simplifiée de l'utilisation du co-training dans un problème de classification binaire. La partie de gauche représente la distribution des données dans l'espace de caractéristiques, ainsi que les classifieurs h_1 et h_2 entraînés sur deux perceptions différentes des données. Les triangles représentent les données sans annotations dans ce même espace de caractéristiques.

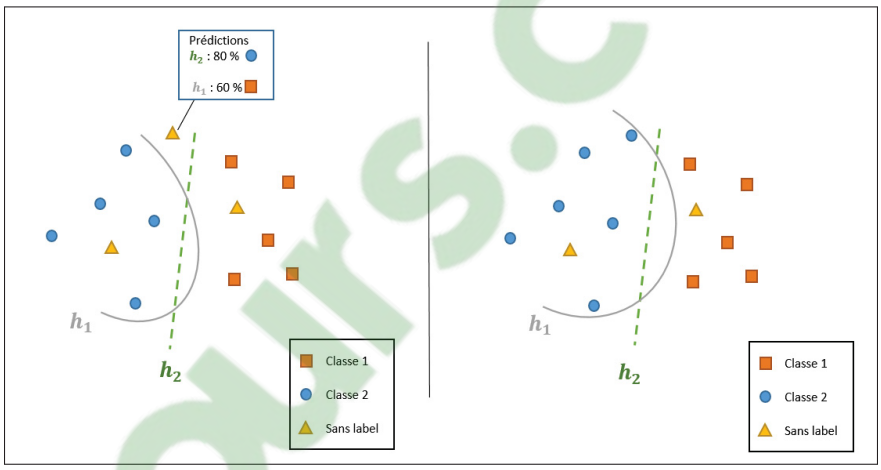


Figure 2.2 Schématisation d'une itération de co-training

Lors d'une itération de co-training, les deux classifieurs vont annoter l'exemple possédant la plus forte prédiction de sorte que l'autre puisse en bénéficier. Ainsi un exemple dont le classifieur h_1 est peu confiant pourra être ajouté en tant que donnée positive si le deuxième module h_2 est très confiant dans sa prédiction. Du point de vue du deuxième classifieur, cette donnée n'apporte pas beaucoup d'informations à la représentation des classes puisqu'elle possède une forte probabilité de prédiction, mais elle est très utile pour l'entraînement du premier.

Par suite, deux grandes hypothèses sur les données sont formulées dans l'article pour tirer profit au co-training :

- Chaque perception des données est suffisante (avec assez de données annotées) pour effectuer la classification.

- Les perceptions doivent être distinctes (conditionnellement indépendantes) pour permettre un apprentissage efficace (cette hypothèse a été relaxée dans les travaux de Abney (2002) : une faible dépendance des perceptions permet aussi au co-training d'être performant).

Le co-training tend donc à entraîner deux modèles h_1 et h_2 tel que : $\forall x \in \mathcal{A}, \exists\{\mathrm{h}_1, h_2\}$ tel que $\mathrm{h}_1(v_1(x)) = h_2(v_2(x)) = f(x)$ avec f une fonction de classification globale de la base de données que l'on cherche à atteindre. En théorie, si on arrive à trouver les bons modèles, on peut utiliser n'importe lequel des deux pour inférer sur la base de test. Mais en pratique utiliser une combinaison des deux permet d'avoir de meilleures performances, puisqu'il est difficile dans le cas réel d'assurer les deux hypothèses du co-training.

Pour ce type d'approches, plusieurs éléments sont à prendre en compte. Tout d'abord il faut s'assurer que les deux modèles soient bien différents. Dans le cas contraire (diversité = 0) aucune nouvelle information ne sera apportée. De plus, dans le cas où le label attribué est incorrect, l'ajout de telle donnée peut nuire aux performances des deux modules de classifications. Les courbes de performances obtenues par Blum et Mitchell sur leur exemple de classification montrent qu'à partir d'un certain nombre d'itérations k les performances commencent à chuter.

On retrouve différentes méthodes de classification intégrant les données non annotées dans leurs prédictions. Le *transductive support vector machine* (ou TSVM) (Vapnik (1998)) consiste à maximiser la marge de séparation entre deux classes. À l'image du self-training, certaines données non annotées se voient attribuer un label en fonction de la prédiction du SVM. Un poids est attribué aux données nouvellement annotées qui seront augmentées progressivement au fur et à mesure des itérations. On utilise ensuite l'ensemble des données annotées pour préciser la séparation entre les classes.

Parmi les méthodes de classification générative, on retrouve l'approche Gaussian Mixture Models (GMM) utilisée avec un algorithme d'Expectation Maximisation pour déterminer les paramètres les plus appropriés aux gaussiennes. L'objectif de cette approche consiste à déterminer les distributions gaussiennes qui représentent le mieux les classes, en utilisant les données annotées pour donner une indication sur la classe des clusters. Les données non annotées, en très

grande quantité, permettent à l'algorithme d'effectuer le clustering en ayant plus de données sur l'espace des caractéristiques.

Une autre catégorie de méthode de classification est basée sur la représentation structurelle des données. Ces techniques sont fondées sur des graphes dont les noeuds sont les données et les bords sont les distances entre les différents noeuds. Pour tirer profit des données non annotées, il faut supposer par exemple que des noeuds proches les uns des autres possèdent le même label.

2.4 Les approches basées sur l'apprentissage profond

Cette partie a pour but de faire la synthèse des modèles profonds les plus performants de nos jours en se basant sur la base de données CIFAR-10 (Krizhevsky & Hinton (2009)). Cette base est constituée de 60 000 images naturelles en couleurs de 32 x 32 pixels. Les données sont équitablement réparties entre 10 classes, 50 000 images sont utilisées pour l'entraînement et 10 000 pour le test. Pour les comparaisons, nous avons retenu le cas semi-supervisé dans lequel 4000 images avec label sont tirées aléatoirement et uniformément entre les classes et le reste de la base est utilisée comme données sans annotations.

Il existe plusieurs familles d'approches profondes pour l'apprentissage semi-supervisée. Le tableau 2.1 représente les approches les plus performantes en apprentissage semi-supervisé de ces dernières années. Ces techniques peuvent être regroupées en différentes catégories en fonction sur leur architecture et des stratégies utilisées pour tirer partie des données non annotées.

Tableau 2.1 Taux d'erreur (exactitude) de test des différentes approches profondes semi-supervisées sur la base CIFAR-10

Méthodes	**Architectures utilisées**	**Semi-supervisé** 4000 labels - 46 000 sans labels		**Entièrement supervisé** 4000 labels	
		Sans augmentation	Avec augmentation	Sans augmentation	Avec augmentation
Ladder Network (Rasmus *et al.* (2015))	Γ Conv-Large	20,40(± 0,47)	-	23,33(± 0,61)	-
VAT (Miyato *et al.* (2017))	Conv-Large	14,18 (± 0,38)	11,36(± 0,34)	-	-
VAT + Ent Min (Miyato *et al.* (2017))	Conv-Large	13,15 (± 0,21)	10,55(±0,05)	-	-
Π model (Laine & Aila (2016))	Conv-Large	16,55(±0,29)	12,36(±0,31)	35,56(±1,59)	34,85(±1,65)
Temporal ensembling (Laine & Aila (2016))	Conv-Large	-	12,16(±0,24)	-	-
Mean teacher (Tarvainen & Valpola (2017))	Conv-Large	-	12,31(±0,28)	-	20,66(±0,57)
Bad GAN (Dai *et al.* (2017))	Conv-Small	14,41(±0,3)	-	-	-
Stochatic transf. (n=5) (Sajjadi *et al.* (2016))	Sparse convnet	-	11,29(±0,24)	-	13,60(±0,24)
Deep Co-Training (Qiao *et al.* (2018))	Conv-Large	-	9,03(±0.18)	-	-

Le modèle *Conv-Large* fait référence à l'architecture décrite dans le tableau 1.1. Cette comparaison entre les différentes approches est basée sur les erreurs publiées dans les différents articles, obtenues sur la base de test de la base CIFAR-10. Cependant, on peut noter des différences entre les protocoles et les architectures utilisées. Par exemple, l'utilisation de la même architecture (*Conv-Large*) génère des taux d'erreurs différents dans le cas entièrement supervisé. Ces écarts peuvent s'expliquer par les différentes tailles de base de validation utilisées, ou encore le nombre d'itérations effectuées pour la recherche des hyperparamètres. Enfin, certaines approches, comme *Mean Teacher* et *Temporal ensembling*, utilisent différentes variantes de couche de normalisation *mean only* (Salimans & Kingma (2016)). Toutes ces remarques témoignent du manque de reproductibilité, qui n'est pas limitée au domaine de l'apprentissage semi-supervisés. Nous avons cherché ici à suivre l'exemple établit par Oliver *et al.* (2018) en comparant les approches sélectionnées sur une base commune. On retrouve plusieurs catégories de réseaux utilisant l'apprentissage semi-supervisé. La grande majorité des approches consistent à régulariser les prédictions d'un modèle en se basant sur l'hypothèse de *smoothness*.

Il faut noter que l'apprentissage semi-supervisée possède un certain nombre de limites. L'apport des données sans annotations va dépendre de la complexité du problème étudié, au nombre de données annotées qui le représente ainsi qu'au modèle utilisé . Dans un cas trivial, où un grand nombre de données annotées sont utilisées pour la classification d'exemples distincts, l'utilisation de données non annotées supplémentaire ne permettra pas d'améliorer les résultats du modèle. Les données sans annotations seront particulièrement intéressantes lorsqu'elles apportent un information supplémentaire à la représentation des classes par un modèle. Si les caractéristiques de la donnée sans annotation est fortement représenter dans la base d'entraînement annotées, sont impact sera probablement très limité car elle confirme les représentations apprises. Dans le cas contraire, si les caractéristiques de cette donnée ne sont pas courant dans la base d'entraînement, son impact sera plus important car il va concerner les bornes de décisions apprises par le modèle.

2.4.1 Cohérence des prédictions

Parmi les approches les plus récentes, celles basées sur la régularisation des prédictions d'un ou de plusieurs modèles sont fréquemment utilisées. Cette catégorie d'approche prend pour hypothèse qu'un modèle prédisant le label $\tilde{y}$ pour une donnée x devrait donner le même label si l'on ajoute une petite perturbation à cette donnée. Les réponses d'un modèle doivent être constantes aux petites variations appliquées aux données d'entrée puisqu'on considère une certaine corrélation entre les données et leurs annotations. En partant de cette hypothèse, les données sans annotations peuvent être utilisées pour entraîner le modèle à être cohérent dans ses prédictions. Parmi les premières approches intégrant des modèles profonds, on retrouve l'article de régularisation stochastique de Sajjadi *et al.* (2016). Le principe général consiste à présenter à un même modèle plusieurs fois la même image ayant subi à chaque fois une transformation différente. Les auteurs introduisent une fonction de coût non supervisé supplémentaire permettant d'optimiser la cohérence des prédictions d'un réseau à la même donnée d'entrée augmentée de différentes façons. Cette fonction est combinée à une fonction de coût supervisée de classification classique comme l'entropie croisée afin de permettre un apprentissage coordonné de la part des données annotées et des données non annotées. La fonction de coût introduite est composée de la somme pondérée de deux termes distincts. Le premier représente une fonction calculant la différence du carré de la moyenne d'une donnée d'entrée augmentée de différentes façons, passé plusieurs fois dans le même modèle. Si on note f la fonction qui représente les prédictions d'un modèle profond, $T^j(x_i)$ une augmentation linéaire ou non linéaire j de la donnée d'entrée x_i, alors la première partie de la fonction non supervisée introduite par Sajjadi *et al.* (2016) peut s'écrire sous la forme :

$$l_{tr} = \sum_{i=1}^{N} \sum_{j=1}^{n-1} \sum_{k=j+1}^{n} ||f(T^j(x_i)) - f(T^k(x_i))||_2^2 \tag{2.1}$$

Dans cette expression, n représente le nombre de passages d'une donnée dans le modèle profond et N représente le nombre de données d'entraînement. Cette première fonction de coût permet d'apprendre au réseau à fournir une distribution de prédiction similaire lorsque de petites variations sont appliquées aux données d'entrée. Le deuxième terme vise à pousser le réseau à fournir des prédictions mutuellement disjointes entre les différentes classes afin d'éviter les

prédictions triviales qui pourraient être apprises par la première partie de la fonction de coût. Si l'on note f_k la prédiction du modèle pour la classe k, en notant C le nombre de classes de l'étude, la deuxième partie de la fonction est formulée comme suit :

$$l_{me} = \sum_{i=1}^{N} \sum_{j=1}^{n} \left(- \sum_{k=1}^{C} f_k(T^j(x_i)) \prod_{l=1, l \neq k}^{C} (1 - f_l(T^j(x_i))) \right) \tag{2.2}$$

Cette fonction va pénaliser les vecteurs de prédictions qui ne produisent pas une probabilité élevée pour une seule classe. Les inconvénients de cette technique portent sur la limite des augmentations apportées en fonction des bases de données utilisées ainsi que l'augmentation importante du temps d'apprentissage.

Au lieu de se focaliser sur le cumule de transformations, les méthodes suivantes, basées sur la même idée, se sont orientés soit vers l'utilisation de plusieurs modèles identiques entraînés sur des images différentes soit vers la recherche de transformations particulièrement intéressantes. Dans ce premier cas de figure, on retrouve plusieurs méthodes dont celle proposée par Laine & Aila (2016). Ils mettent en avant le bénéfice d'utiliser un ensemble de réseaux pour effectuer des prédictions. La première approche, appelée Π-model, est un cas particulier de l'approche de Sajjadi *et al.* (2016). Comme on peut le voir sur la figure 2.3, le but est d'entraîner un réseau à partir de prédictions issues du passage de données augmentées différemment. L'utilisation de *dropout* permet de régulariser le réseau en produisant des prédictions différentes à chaque itération. L'utilisation des prédictions du réseau avec *dropout* peut être vue comme celui d'un ensemble de réseaux ayant à chaque itération des architectures légèrement différentes. L'entraînement est effectué comme précédemment à partir de données annotées, utilisées pour la fonction de coût supervisée, et de données non annotées utilisées pour le calcul de la fonction de coût non supervisée.

Une deuxième variante à cette méthode d'entraînement est appelée *Temporal ensembling*. Elle est basée sur la même idée d'entraînement à partir d'un ensemble fictif de modèles en ajoutant la dimension de temps à l'entraînement. La prédiction pour une donnée x_i à un temps t par un réseau f^t est gardée en mémoire. Elle sera comparée à la prédiction d'un réseau $f^{t+\alpha}$

ayant subi plusieurs mises à jour de paramètres. Dans cette expression, γ représente le temps nécessaire pour que le réseau f parcours l'ensemble de la base d'entraînement. Les prédictions enregistrées sont mises à jour par une moyenne exponentielle mobile (cf. Eq. 2.13) afin de garder une valeur historique des prédictions de chaque donnée, en prenant en compte les prédictions antérieures. Cette méthode reste limitée : les variations dans l'apprentissage à partir des données non annotées ne sont prises en compte qu'une fois par epoch et cette technique nécessite de sauvegarder les prédictions de chaque donnée d'entraînement, ce qui limite la taille des bases de données utilisées.

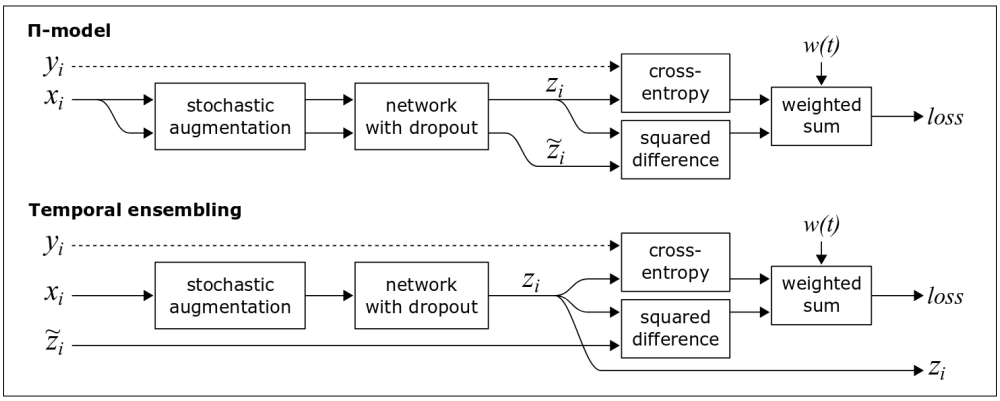


Figure 2.3 Schématisation du fonctionnement de Π-net et de *Temporal ensembling* (Laine & Aila (2016)).

Afin de pallier cette limite, l'article proposé par Tarvainen & Valpola (2017) présente une nouvelle approche inspirée par *Temporal ensembling*. L'approche *Mean teacher* met l'accent sur l'apprentissage des paramètres des réseaux plutôt que sur leurs prédictions. Cette méthode est composée de deux réseaux, un étudiant et un professeur. Lors de l'entraînement, une même donnée d'entrée est présentée avec différentes transformations aux deux réseaux. L'étudiant est utilisé pour le calcul de la fonction de coût supervisé et le professeur pour le calcul de fonction de coût non supervisé en comparant la distribution des scores obtenus avec l'étudiant. La mise à jour des paramètres du professeur est une moyenne mobile exponentielle des paramètres de l'étudiant, dont les paramètres sont mis à jour par rétropropagation. Lors des phases de tests, on utilise le modèle du professeur pour prédire le label des patches. Du point de vue expérimentale, c'est une approche qui nécessite un plus grand nombre d'epochs pour que le modèle

professeur converge puisque ses paramètres ne sont qu'une moyenne exponentielle de ceux du modèle étudiant. Un autre inconvénient de cette méthode est qu'elle utilise principalement les transformations appliquées aux images ainsi qu'aux potentielles couches *drop out* de pour tirer partie des données sans annotations.

L'approche proposée par Miyato *et al.* (2017), le *Virtual adversarial training* (VAT), utilise un autre point de vue pour répondre au défi de l'entraînement semi-supervisé. Cette approche se focalise sur l'optimisation du bruit apporté aux images afin de régulariser le réseau. Le but du *Virtual adversarial training* est de générer un bruit de faible intensité permettant de transformer une image d'entrée en une image particulièrement difficile à classer pour le modèle. Le bruit est optimisé pour forcer le réseau à apprendre des cas de figure les plus ambigus. Cette méthode de régularisation met l'accent sur les bornes de décision entre les différentes classes dans l'espace de représentation. Cette méthode permet d'ajouter un bruit optimisé supplémentaires aux différentes images étudiées. Néanmoins elle nécessite de calculer, à chaque itération, un bruit pour l'ensemble des patches étudiées, ce qui peut ralentir le temps d'entraînement.

2.4.2 Les réseaux antagonistes génératifs

Les *Generative adversarial networks* (GANs) sont des modèles génératifs non supervisés. Ils ont été introduits par Goodfellow *et al.* (2014a) et permettent d'entraîner un modèle à générer des images similaires à celles utilisées pour l'entraînement. Le GANs est composé de deux réseaux de neurones : un générateur qui a pour but de générer des images à partir d'un bruit aléatoire, et un discriminateur qui a pour but de classer les images qu'il reçoit en deux catégories : réelle ou fausse. L'apprentissage de ce modèle est effectué en opposant ces deux réseaux l'un contre l'autre.

Le générateur a pour but de générer des images qui ressemblent le plus possible aux images réelles issues de la base d'entraînement, et le discriminateur a pour objectif de distinguer le plus fidèlement possible les images issues de la base d'entraînement, classée comme réelle, ou les images créées par le générateur, classées comme fausses.

Ce modèle est devenu très rapidement populaire par la qualité des images qu'il pouvait générer, jusqu'alors obtenues à l'aide d'auto-encodeurs. Plusieurs méthodes semi-supervisées basées sur les GANs ont vu le jour ces dernières années. Contrairement à l'approche classique, l'entraînement de GANs pour l'apprentissage semi-supervisé vise à entraîner un discriminateur capable d'attribuer un label aux images qu'on lui présente. L'entraînement du discriminateur n'est plus binaire, en plus de devoir différencier les images réelles des fausses, il doit également différencier à quelle classe appartient l'image dans le cas où elle serait réelle.

Parmi les premières approches performantes, on retrouve le *Categorical GAN* (Springenberg (2015)). Le discriminateur est entraîné pour distinguer les K classes. L'intuition derrière cette approche est d'assigner une prédiction équitablement distribuée entre les différentes classes pour les images issues du générateur. De plus, un coût supplémentaire est ajouté à l'entraînement. Afin d'améliorer l'entraînement, ils font l'hypothèse que les données sont équitablement réparties entre les différentes classes. Un terme supplémentaire qui consiste à maximiser l'entropie des prédictions est appliqué à l'ensemble des images de la base d'entraînement, mais aussi aux images générées par le générateur.

Par la suite d'autres approches ont vu le jour comme *Improved techniques for training GANs* par Salimans *et al.* (2016). Cette approche semi-supervisée considère une classe supplémentaire pour la classification des images issues du générateur. Le discriminateur va être optimisé pour attribuer un des K classes aux images réelles et la classe K+1 pour les images générées. Pour les images sans annotations, ils vont chercher à minimiser la probabilité que le discriminateur attribue la classe K+1. Le générateur quant à lui ne va plus chercher à tromper la prédiction du discriminateur, mais plutôt chercher à reproduire la distribution des données réelles de la couche antérieures à la classification. L'objectif du générateur consiste à produire une correspondance des caractéristiques entre le générateur et le discriminateur.

L'approche la plus performante basée sur les GANs est proposée par Dai *et al.* (2017) dans l'article *Good semi-supervised learning that requires a bad GAN*. Elle s'inspire de la méthode proposée par Salimans *et al.* (2016) en modifiant l'objectif du générateur. Ils montrent que l'ap-

prentissage d'un bon discriminateur passe par l'utilisation d'un mauvais générateur. Mauvais dans le sens où il va produire des images aux limites des distributions et donc moins représentatives des données d'entraînement. Les GANs ont obtenu de très bonnes performances dans le domaine semi-supervisé.

Les GANs ont l'avantage de générer une grande quantité d'images synthétiques un fois entrainés. Néanmoins, ce type d'approches nécessitent d'entrainer au moins deux modèles profonds simultanément. De plus, l'utilisation de patches de grandes tailles va augmenter de manière importante la complexité du modèle générateur.

2.4.3 Les auto-encoders

Les auto-encodeurs sont formés de deux branches, un encodeur qui a pour but de projeter la donnée d'entrée dans un espace généralement de plus petite dimension, et le décodeur dont le but est de reconstruire la donnée à partir de sa projection. Durant l'entraînement, la différence entre la donnée d'entrée et la donnée décodée par le modèle est utilisée comme fonction de coût. Les auto-encodeurs peuvent être utilisés de plusieurs façons pour l'apprentissage semi-supervisé. La méthode la plus directe consiste à entraîner l'auto-encodeur à reproduire l'ensemble des données, annotées et non annotées, dans un espace de représentation pertinent. On utilise l'encodeur comme extracteur de caractéristiques pour ensuite entraîner un classifieur sur les données possédant des annotations.

Cette méthode de pré-entraînement est utilisée sur un auto-encodeur de débruitage pour le *pseudo-labelling* (Lee (2013)). C'est une technique très simple qui consiste à effectuer du *self-training* durant l'entraînement d'un réseau profond afin de tirer profit des données non annotées.

Le pré-entraînement de l'auto-encodeur est aussi une des approches utilisées sur un auto-encodeur variationnel proposé par Kingma *et al.* (2014) pour l'apprentissage semi-supervisé. Les auto-encodeurs variationnels sont des auto-encodeurs génératifs qui visent à encoder les données d'entrée dans une distribution normale. L'objectif de l'encodeur est alors d'apprendre

à transformer une donnée d'entrée en deux vecteurs latents, un pour la moyenne et l'autre représentant la déviation standard de la distribution dans laquelle se situe la donnée de départ. Ainsi, il est possible de générer de nouvelles images issues de la même distribution que les images d'entrées, en faisant des tirages aléatoires à partir des distributions obtenues par l'encodeur.

Une autre approche consiste à ajouter des couches supplémentaires à l'encodeur pour pouvoir effectuer la classification directement sur les images annotées et utiliser le décodeur pour calculer un coût non supervisé sur les images sans annotations. Cette technique est utilisée par l'approche *Ladder Network* (Rasmus *et al.* (2015)) et *Stacked What Where Auto-Encoder* (SWWAE) (Junbo Zhao & Lecun (2015)).

SWWAE est un auto-encodeur composé de plusieurs couches de *pooling* permettant d'extraire après chaque convolution l'information sur quoi (*What*) et où (*Where*) se trouve l'information pertinente sur une image. L'extraction de l'information *what* est effectuée par un *max-pooling* et transmise à la couche suivante, tandis que l'information *where* est transmise à la couche du décodeur pour aider la reconstruction de l'image dans les couches d'*un-pooling*. L'entraînement est effectué en une seule fois, les images annotées sont utilisées pour entraîner l'encodeur et la partie de classification au travers d'une fonction de coût supervisée. Les images non supervisées sont utilisées dans le décodeur et permettent de calculer un coût de reconstruction entre les paires de couches de l'encodeur et du décodeur.

Parmi les approches basées sur les auto-encodeurs les plus performants, on retrouve le *Ladder Network* (Rasmus *et al.* (2015)). Les données non annotées sont utilisées dans une fonction de coût basée sur la reconstruction des images. Cette approche utilise un auto-encodeur de débruitage par couche comme artifice pour l'entraînement du réseau. Une étude plus détaillée du fonctionnement est présente dans la section 2.5.1. Comme dans le réseau proposé par Junbo Zhao & Lecun (2015), l'information de l'encodeur est transmise au décodeur au travers de connexions annexes. Les auto-encodeurs ont la particularité de tirer parti de la reconstruction

des données. Néanmoins, l'entraînement du décodeur va doubler la quantité de paramètres a mettre à jour durant l'apprentissage.

2.4.4 Autre approche

L'approche *Deep Co-Training* (DCTR) (Qiao *et al.* (2018)) est une nouvelle approche de co-training appliquée à l'utilisation d'architectures profondes. La particularité de cette méthode porte sur le fait que l'apprentissage est effectué une seule fois. Contrairement au co-training introduit par Blum & Mitchell (1998), cette approche n'a pas pour but d'annoter les données sans annotations pour perfectionner les deux modèles de classification. Le perfectionnement des deux modèles est effectué durant l'entraînement au travers d'une combinaison de plusieurs fonctions de coût. Une même base de données est utilisée pour entraîner deux architectures profondes identiques. Cette approche est actuellement la plus performante sur la base de données CIFAR10. Elle tire partie à la fois de l'utilisation de plusieurs modèles durant l'apprentissage mais aussi de la génération de bruit adversaire. Le désavantage principal va se retrouver durant l'apprentissage, qui va être ralentit par le nombre de modèles envisagés ainsi que par la génération des exemples adversaires à chaque itération.

2.5 Étude détaillée des modèles utilisés

Nous avons choisi d'utiliser les modèles faisant l'état de l'art sur différentes bases de données utilisant des stratégies différentes pour traiter des images non annotées. Les modèles *VAT* (Miyato *et al.* (2017)) et *Mean teacher* (Tarvainen & Valpola (2017)) présentent tous deux des performances compétitives sur la base CIFAR-10 en utilisant des approches d'apprentissage semi-supervisées distinctes. Le premier se focalise sur l'optimisation du bruit à ajouter aux données sans annotations. Le deuxième porte son attention sur l'apprentissage d'un modèle adjacent dont le but consiste à lisser l'apprentissage du modèle étudiant. Notre attention s'est ensuite porté sur l'approche *Ladder Network* (Rasmus *et al.* (2015)) particulièrement performante sur la base MNIST et qui base son apprentissage sur la reconstruction des images sans annotations. Enfin, nous avons ajouté l'approche *Deep Co-Training* (Qiao *et al.* (2018)), qui

reporte les meilleurs résultats sur la base CIFAR-10, à nos expériences pour avoir un quatrième point de vue l'apprentissage semi-supervisé.

2.5.1 Ladder network

Le *Ladder network* (Rasmus *et al.* (2015)) est un réseau correspondant à un assemblage d'auto-encodeur profond dont le but est d'enlever le bruit associé à l'image d'entrée à chaque couche. Comme on peut le voir sur la figure 2.4, c'est un réseau composé de trois branches.

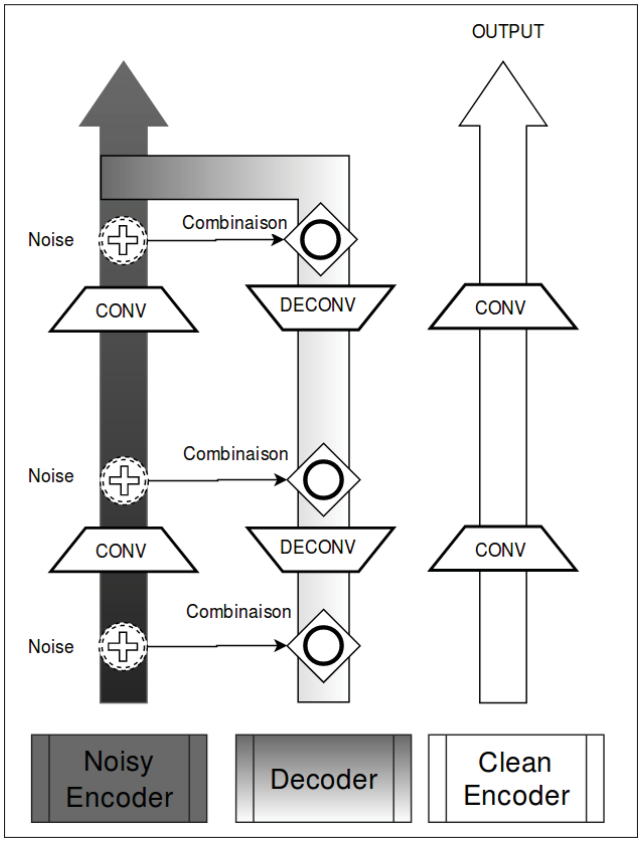


Figure 2.4 Schématisation du Ladder Network (Rasmus *et al.* (2015)).

La première est un encodeur auquel on ajoute du bruit (gaussien) à chaque niveau. La deuxième branche est un décodeur dont le but est de débruiter l'information du premier encodeur. La dernière branche est un encodeur sans bruit qui partage ses poids avec l'encodeur bruité et

dont la sortie est celle utilisée lors de l'inférence. L'ensemble encodeur bruité - décodeur, est un artifice utilisé uniquement lors de l'entraînement du réseau. Seul l'encodeur sans bruit est nécessaire lors de l'inférence. Ce réseau apprend à partir d'images annotées par le biais de l'encoder bruité et d'images non annotées via le décodeur. La fonction de coût supervisée consiste à effectuer un calcul d'entropie croisée (Eq. 2.3) entre la prédiction de l'encodeur bruitée $f_{enc}(\widetilde{x})$ et le label de l'image y.

$$\mathcal{C}_{supervisé} = \frac{1}{N_a} \sum_{x \in \mathcal{A}} CE(y, f_{enc}(\widetilde{x})) \tag{2.3}$$

Pour l'obtention du coût non supervisé, plusieurs paramètres sont à prendre en compte. L'objectif de cette fonction vise à minimiser la distance entre les variables latentes issues du décodeur $\hat{z}_{bn}^{(l)}$ avec leurs équivalents issues de l'encodeur non bruité $\hat{u}_{bn}^{(l)}$. Les variables latentes z représentent les cartes de caractéristiques obtenues après la couche (l), avant la normalisation du batch. Dans notre cas, $z^{(l)}$ correspond aux variables de l'encodeur bruité, $u^{(l)}$ aux variables du décodeur et $z^{(l)}$ à celles de l'encodeur non bruité. La grande particularité du modèle Ladder réside dans ses connexions entre les couches de l'encodeur bruité et du décodeur, comme on peut le voir sur le schéma 2.4. Ces connexions sont issues de l'encodeur bruité et sont combinées (Eq. 2.6) dans le décodeur. La combinaison des variables latentes s'effectue au travers d'une fonction de combinaison non linéaire composée de dix paramètres $a_{.,i}$ qui sont appris par le réseau lors de l'entraînement.

$$\mu_i(u_i^{(l)}) = a_{1,i}^{(l)} sigmoid(a_{2,i}^{(l)} u_i^{(l)} + a_{3,i}^{(l)}) + a_{4,i}^{(l)} u_i^{(l)} + a_{5,i}^{(l)} \tag{2.4}$$

$$\nu_i(u_i^{(l)}) = a_{6,i}^{(l)} sigmoid(a_{7,i}^{(l)} u_i^{(l)} + a_{8,i}^{(l)}) + a_{9,i}^{(l)} u_i^{(l)} + a_{10,i}^{(l)} \tag{2.5}$$

$$\hat{u}_i^{(l)} = (z_i(l) - \mu_i(u_i^{(l)}))\nu_i(u_i^{(l)}) + \mu_i(u_i^{(l)}) \tag{2.6}$$

La variable $u_i^{(l)}$ correspond à la combinaison des variables latentes entre l'encodeur bruité $z_i(l)$ et le décodeur $u_i(l)$. Ensuite, $\hat{u}_{bn}^{(l)}$ est obtenue en normalisant $u_i(l)$ à partir de la moyenne et de l'écart type du batch de chaque sortie de convolution de l'encodeur sans bruit. Une fois toutes ces variables obtenues, le coût non supervisé est calculé par la somme pondérée

des erreurs quadratiques moyennes des cartes de caractéristiques à chaque couche du réseau, comme indiqué dans l'équation 2.7.

$$C_{nonsupervisé} = \frac{1}{N_a + N_{unl}} \sum_{i=1}^{N} \sum_{l=0}^{L} \lambda_l ||z^{(l)}(x_i) - \hat{u}_{bn}^{(l)}(x_i)||^2 \tag{2.7}$$

Les coefficients de pondération λ_l sont des hyperparamètres associés à chaque couche. Les résultats présentés par Rasmus *et al.* (2015) indiquent que la couche la plus élevée a le poids le plus important. Enfin le coût total est obtenu en ajoutant les deux coûts obtenus précédemment.

$$C_{total} = C_{supervisé} + C_{nonsupervisé} \tag{2.8}$$

Il convient de noter que pour effectuer une itération, il est nécessaire d'obtenir les prédictions de l'encodeur bruité sur les images annotées, et d'obtenir les variables latentes de l'encodeur bruité, du décodeur et de l'encodeur non bruité sur les images annotées et non annotées.

Si l'on s'intéresse de plus près à l'ajout du bruit, on constate qu'il est effectué lors d'un calcul de normalisation du batch. En effet, la première partie de la normalisation (Eq. 1.1) est effectuée juste à la sortie des couches de convolutions. Les cartes de caractéristiques sont alors normalisées à partir de la moyenne et de l'écart type du batch. Un bruit gaussien est ensuite ajouté à l'ensemble du batch et la variable latente bruitée $z_i(l)$ est gardée en mémoire pour le calcul de la fonction de combinaison de la couche du décodeur correspondante. Une mise à l'échelle et un déplacement sont effectués à partir de deux paramètres appris lors de l'entraînement. Cela permet de corriger la normalisation avant le passage dans la non-linéarité (Eq. 1.2). Ces paramètres sont normalement directement calculés après la normalisation des données. Un schéma faisant le bilan des différentes connexions est présenté figure 2.5. C'est un exemple de modèle possédant 2 convolutions. Durant l'entraînement un batch de données annotées est utilisé avec un batch de données sans labels. Le but de ce modèle consiste à obtenir une structure hiérarchique permettant de décomposer les différentes connaissances pertinentes présentent dans l'image. Pour cela, les variables latentes présentes à chaque couche de l'encodeur bruité sont utilisées pour les couches du décodeur correspondantes. L'information

est ainsi transmise à l'aide d'une fonction combinatoire (Eq. 2.6) permettant de calculer les variables latentes de la couche inférieure à partir des informations obtenues d'une part de la couche de l'encodeur bruité correspondante et, d'autre part, des informations de la couche précédente du décodeur.

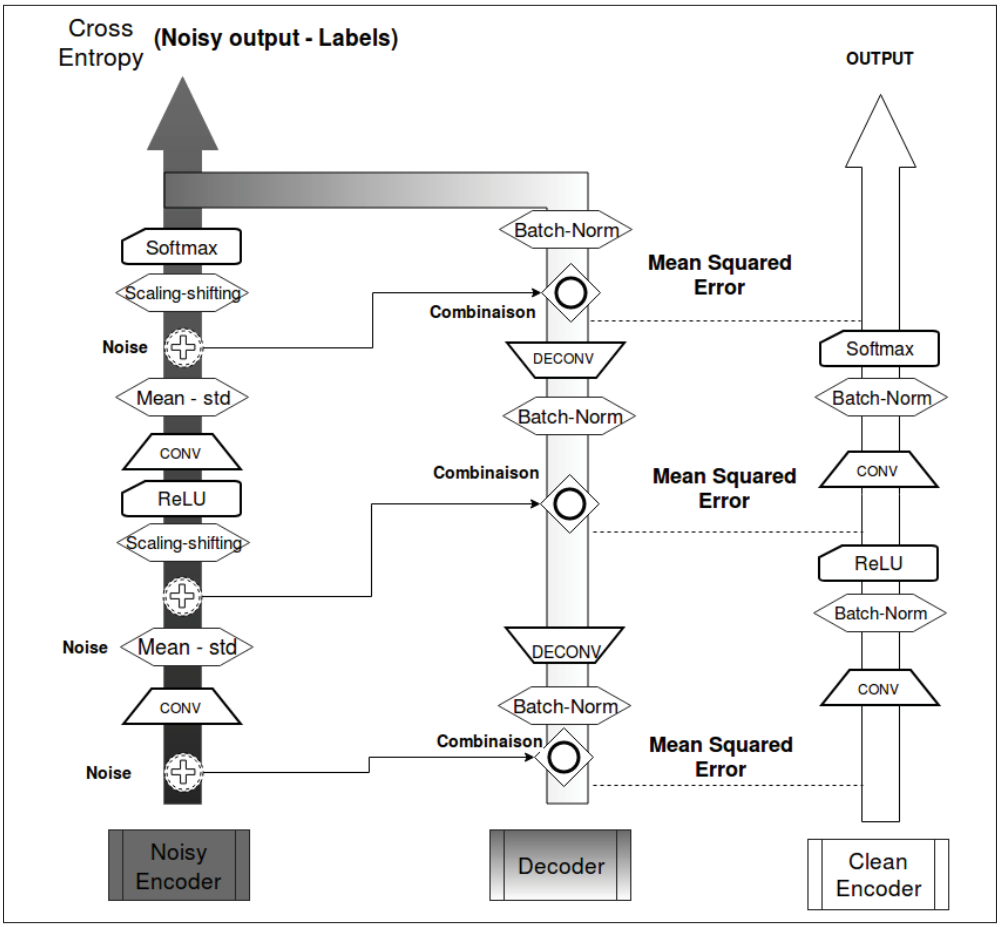


Figure 2.5 Schématisation plus détaillée des connexions du Ladder Network

Il a été nécessaire de faire des choix concernant l'utilisation de cette technique semi-supervisée sur une architecture profonde comme le *ResNet-18*. Les auteurs ne mentionnent pas ce cas de figure, les seuls résultats publiés concernent l'utilisation d'une architecture Γ basée sur le modèle *Conv-Large*. Cette architecture prend en compte la première couche du décodeur comme variable latente, ce qui réduit l'architecture du modèle *Ladder*. Le schéma 2.6 représente

l'architecture que nous avons utilisé pour limiter le nombre de variables latentes à garder en mémoire durant l'apprentissage. L'ajout de bruit est effectué après chaque bloc résiduel, ainsi qu'après les autres couches. De cette manière, l'apprentissage reste réalisable, même pour des patches importants comme pour le cas de la base BACH.

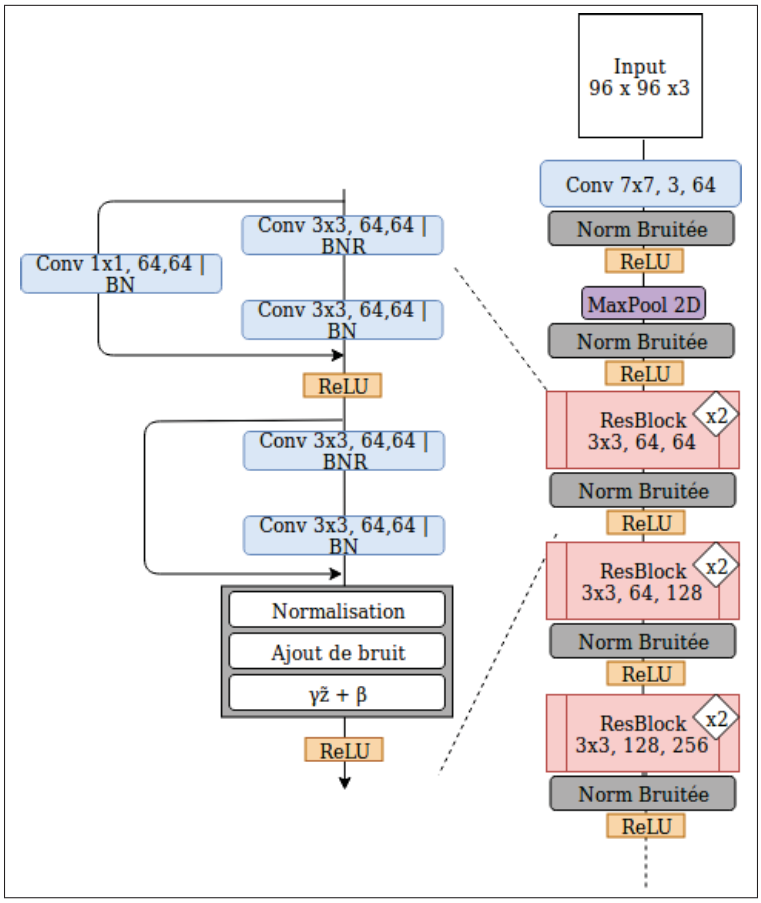


Figure 2.6 Schématisation d'une partie de l'encodeur bruité basé sur un ResNet-18

2.5.2 Virtual Adversarial Training

L'approche *Virtual adversarial training* (Miyato *et al.* (2017)) est inspiré de la fonction coût adversaire. Comme schématisé sur la figure 2.7, l'approche VAT est une technique d'entraînement permettant de régulariser un modèle profond en utilisant des données sans annotation. Le principe de cette technique consiste à optimiser un bruit adapté qui une fois ajouté aux images

non annotées les rendent plus difficiles à classer pour le modèle, car plus éloignées du centre de distribution de la classe. Le bruit virtuellement adversaire est calculé, pour chaque image, par rétropropagation de la fonction non supervisée des distributions des scores entre l'image bruitée et l'image sans bruit. Cette distance correspond à la prédiction du réseau (après softmax) entre l'image d'entrée x et $x + \xi_{VAT}\, d$, d étant un bruit issu d'une distribution gaussienne permettant le tirage de variables indépendantes et identiquement distribuées.

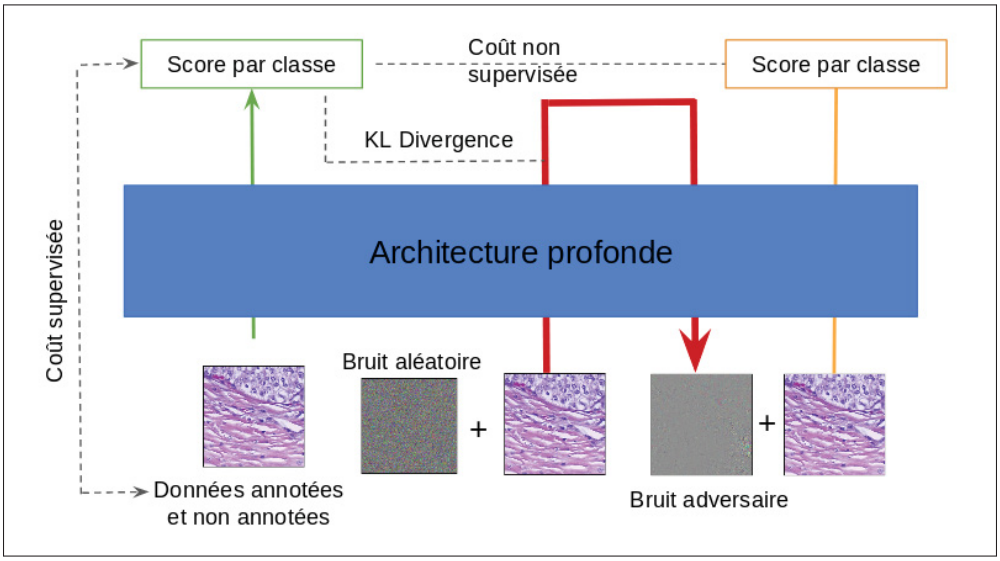


Figure 2.7 Schématisation du fonctionnement de VAT (Miyato *et al.* (2017))

La variable ξ_{VAT} est un hyperparamètre généralement très petit (ξ_{MT} = 1e-6 dans notre cas). La divergence de Kullback-Leibler permet de mesurer la dissimilarité entre deux lois de probabilité. Dans notre cas, la fonction de coût non supervisée calcule la distance entre les distributions issues du réseau à partir des images avec et sans bruit adversaire.

$$KLD(P,Q) = \sum_i P(i)\, log \frac{P(i)}{Q(i)} \tag{2.9}$$

La distance ainsi obtenue est normalisée (en la divisant par sa norme L2) pour ensuite être retropropagé dans le réseau. Ce calcul nous permet de savoir comment modifier le bruit en entrée pour maximiser la fonction de coût et ainsi produire un exemple particulièrement difficile

pour le réseau. Ce bruit adversaire, noté r_{adv}, est pondéré par un deuxième hyperparamètre ϵ_{VAT}. Les auteurs indiquent qu'empiriquement une seule itération est suffisante pour obtenir un exemple adversaire efficace. Les courbes présentées dans la section 4.3 de l'article de Miyato *et al.* (2017) montrent que le bruit généré reste relativement le même après la première itération. À chaque itération, la partie du batch possédant des labels est utilisée de manière classique pour calculer la fonction de coût supervisée (entropie croisée dans notre cas). Toutes les images du batch sont utilisées pour générer des exemples adversaires afin de calculer la fonction de coût non supervisée (une deuxième divergence de Kullback-Leibler dans notre étude). Pour cette méthode, la fonction de coût total s'exprime de la manière suivante :

$$\mathcal{C}_{VAT} = \frac{1}{N} \sum_{x \epsilon D} KLD(p(y|x), p(\widetilde{y}|x + r_{adv})) \tag{2.10}$$

$$\mathcal{C}_{supervisé} = \frac{1}{N_a} \sum_{x \in \mathcal{A}} CE(y, f(x)) \tag{2.11}$$

L'objectif de la fonction de coût non supervisée vise à comparer les prédictions obtenues par un même réseau entre une image sans bruit et une image bruitée. La comparaison étant effectuée au niveau des distributions des scores de prédictions, il n'est pas nécessaire d'avoir les annotations pour générer le bruit.

$$\mathcal{C}_{total} = \mathcal{C}_{supervisé} + \lambda_{VAT} \mathcal{C}_{VAT} \tag{2.12}$$

L'un des avantages de cette méthode d'entraînement est qu'elle peut s'appliquer directement aux architectures profondes et possède peu d'hyperparamètres (la recherche est focalisée sur ϵ_{VAT}). Une variante est aussi proposée par les auteurs en ajoutant un terme d'entropie à la fonction de coût total. Cet ajout contraint le modèle à fournir des prédictions plus catégoriques (l'entropie est maximale lorsque la probabilité est égale entre les différentes classes). Cette régularisation par l'entropie a montré ses preuves dans d'autres approches (Sajjadi *et al.* (2016))(Lee (2013)) et permet généralement d'obtenir de meilleures performances. Dans cette étude, nous avons choisi de ne pas la prendre en compte, puisqu'elle peut être appliquée de manière similaire à toutes les autres approches étudiées.

2.5.3 Mean teacher

L'approche *Mean teacher* (Tarvainen & Valpola (2017)) est une technique d'entraînement composé de deux architectures identiques : un étudiant et un professeur. La figure 2.8 schématise l'entraînement de cette approche. Le réseau symbolisé par l'étudiant est un réseau qui va s'entraîner directement à partir des images en entrée.

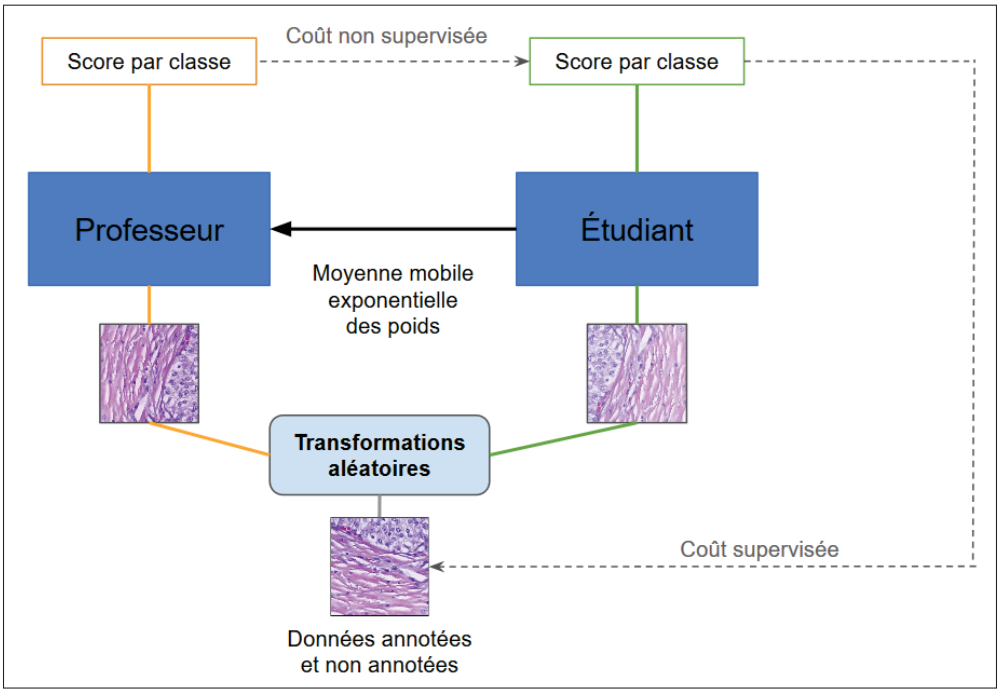


Figure 2.8 Schématisation du fonctionnement de Mean teacher (Tarvainen & Valpola (2017))

À chaque itération, un coût est calculé à partir des données annotées et non annotées et rétropropagé pour mettre à jour ses paramètres. Le professeur quant à lui observe l'étudiant. À chaque itération, les paramètres du professeur sont mis à jour à partir d'une partie des paramètres de l'étudiant. Cette mise à jour étant effectuée à l'aide d'une moyenne mobile exponentielle, l'apprentissage du professeur est plus lissé dans le temps. Soit θ_t^P et θ_t^E les paramètres respectifs des réseaux professeur et étudiant à l'itération t. La mise à jour par

moyenne exponentielle mobile peut se formuler de la manière suivante :

$$\theta_t^P = \alpha\theta_{t-1}^P + (1-\alpha)\theta_t^E \tag{2.13}$$

L'hyperparamètre α_{MT} permet de réguler la proportion des paramètres du réseau étudiant qui sera transmis au professeur. Pour les premières epochs, ce critère est plus important pour permettre aux paramètres du professeur de suivre l'apprentissage rapide du début d'entraînement. Le but étant d'entraîner le professeur à généraliser l'apprentissage directement acquis de l'étudiant. Lors de l'entraînement, le même batch d'image augmenté de manière aléatoire est présenté aux deux réseaux. À partir des prédictions de l'étudiant avec les labels des images annotées présent dans le batch, un coût supervisé est calculé. L'ensemble de ses prédictions sont comparées avec celles obtenues par le réseau professeur afin de calculer un coût de cohérence non supervisé entre les deux réseaux. L'addition de ces deux coûts permet d'obtenir le coût retropropagé pour mettre à jour les paramètres de l'étudiant. Dans ce modèle, la fonction d'entropie croisée est utilisée comme fonction de coût supervisée et l'erreur quadratique moyenne est utilisée comme fonction de coût non supervisée. Les tests sont effectués sur le réseau du professeur.

2.5.4 Deep Co-training

L'approche *Deep Co-training* (Qiao *et al.* (2018)) est inspirée du co-training introduite par Blum & Mitchell (1998). Pour Qiao *et al.* (2018) la différence de perception est générée artificiellement. Le but étant de créer une base de données D' d'exemples tel que $\forall x \in \mathcal{D}', h_1(v_1(x)) \neq h_2(v_2(x))$. Pour cela, les exemples adversaires du modèle h_1 seront utilisés pour entraîner le modèle h_2 et inversement. Les exemples adversaires sont des exemples générés qui ont pour but de tromper (fausser la prédiction) un modèle précis. Ainsi, des données seront produites, semblables aux données réelles sur lesquelles les deux modèles n'auront pas les mêmes prédictions. L'approche proposée consiste à entraîner deux modèles profonds identiques sur une fonction de coût commune composée de trois termes : $\mathcal{C}_{sup}$, $\mathcal{C}_{cot}$ et $\mathcal{C}_{dif}$.

Le premier terme, $\mathcal{C}_{sup}$, représente la partie supervisée, calculée sur la base annotée. Cette fonction a pour but de quantifier la différence entre les prédictions des deux modèles obtenues sur la base annotée avec leur label. En considérant les données annotées $(x, y) \in \mathcal{A}$ avec x une donnée d'entrée et y son label, on a :

$$\mathcal{C}_{sup} = \frac{1}{N_a} \sum_{x \in \mathcal{A}} (CE(y, h_1(v_1(x))) + CE(y, h_2(v_2(x)))) \tag{2.14}$$

Dans ce cas, v_1 représente la partie extraction de caractéristique de l'architecture profonde et h_1 représente la partie classification, souvent composée de couches entièrement connectées. Le deuxième terme de la fonction de coût est $\mathcal{C}_{cot}$. Le but de cette fonction est de pousser les prédictions $h_1(v_1(x))$ et $h_2(v_2(x))$ à être égale pour $x \in \mathcal{U}$. Pour cela, les auteurs ont utilisé la divergence de Jensen-Shannon entre les prédictions obtenues par les deux modèles h_1 et h_2 sur la base $\mathcal{U}$ tel que :

$$\mathcal{C}_{cot} = \frac{1}{N_{unl}} \sum_{x \in \mathcal{U}} (H(\frac{1}{2}(h_1(v_1(x)) + h_2(v_2(x))) - \frac{1}{2}(H(h_1(v_1(x))) + H(h_2(v_2(x))))) \tag{2.15}$$

avec H représentant l'entropie telle que :

$$H(p) = -\sum_{x} p(x) log(p(x)) \tag{2.16}$$

Cette fonction pousse les deux modèles à prédire les mêmes probabilités des différentes classe pour chaque image non annotée. Elle tend à ce que les deux modèles prédisent une distribution moyenne commune aux données non annotées. Cette fonction aura tendance à diminuer les prédictions du modèle le plus confiant et à augmenter celle du second.

La dernière partie de la fonction de coût consiste à forcer une différence de perception entre les deux modèles. Pour cela, il est nécessaire de générer des exemples adversaires $\tilde{x}_1$ et $\tilde{x}_2$ à partir respectivement de h_1 et h_2. L'expression $\tilde{x}_1$ représente les exemples x qui ont été rendus adversaires pour le classifieur h_1.

Ces exemples sont générés avec la méthode *Fast Gradient Sign Method* (Goodfellow *et al.* (2014b)) qui consiste à ajouter un faible bruit aux images afin de les rendre plus difficiles à prédire par le modèle. Concrètement, ce bruit est égal à un hyperparamètre ϵ_{DCTR} multiplié par la rétropropagation du gradient appliqué a l'image d'entrée. Ainsi, on applique une transformation sur l'image dans le sens opposé du gradient, c'est-à-dire qu'on calcule le bruit nécessaire pour maximiser l'erreur de prédiction, au lieu de la minimiser. En entraînant les modèles à correctement prédire les exemples adversaires du modèle opposé, on induit la différence de point de vue des deux modèles. Dans le cas des données annotées, il suffit de comparer les prédictions à la classe de l'image. Pour les données sans annotations, la rétropropagation est effectuée par rapport au label le plus probable.

La fonction $\mathcal{C}_{dif}$ consiste à minimiser l'entropie croisée entre la prédiction d'un classifieur $h_1(v_1(x))$ avec la prédiction du deuxième classifieur sur les mêmes données, mais adversaires $h_2(v_2(\tilde{x}_1))$. On a donc :

$$\mathcal{C}_{dif} = \frac{1}{N_a + N_{unl}} \sum_{x \in \mathcal{D}} (CE(h_1(v_1(x)), h_2(v_2(\tilde{x}_1))) + CE(h_2(v_2(x)), h_1(v_1(\tilde{x}_2)))) \quad (2.17)$$

La fonction de coût finale utilisée consiste en la somme de ces trois fonctions, pondérée par des hyperparamètres λ_{cot} et λ_{dif} tel que :

$$\mathcal{C}_{global} = \mathcal{C}_{sup} + \lambda_{cot}\mathcal{C}_{cot} + \lambda_{dif}\mathcal{C}_{dif} \quad (2.18)$$

Dans leur article, seul un des deux modèles entrainés est utilisé en tant que comparaison équitable avec les autres modèles de CIFAR10. Il est intéressant de noter qu'utiliser deux ou plus de modèles afin de prédire les classes permet d'obtenir encore de meilleures performances. Nous allons donc dans un premier temps comparer les résultats obtenus avec un seul des deux modèles. Dans une deuxième partie des expériences, nous allons étudier, pour les différentes bases de données, l'écart de prédiction généré par la fonction de coût $\mathcal{C}_{dif}$, afin de comparer l'impact de cette méthode de diversité avec d'autres méthodes plus classiques. Nous allons aussi nous

intéresser aux différentes techniques de vote afin de tirer parti des différentes représentations des deux modèles.

Récapitulatif

Le tableau 2.2 représente succinctement les stratégies utilisées par les différentes approches sélectionnées. Seule la fonction de coût non supervisée est indiquée car l'ensemble de ces approches utilise l'entropie croisée comme fonction de coût supervisée. Dans ce tableau, la 'Prédiction de l'image' correspond au vecteur de probabilité obtenu en utilisant un modèle de reconnaissance d'image.

Tableau 2.2 Tableau récapitulatif des approches sélectionnées

Approches	**Coût non supervisé**	**Caractéristiques prises en compte**	**Nombre d'architectures entrainées**
Ladder Network (Rasmus *et al.* (2015))	Erreur quadratique moyenne	Variables latentes du décodeur et de l'encoder non bruitée	2
Virtual Adversarial Training (Miyato *et al.* (2017))	Divergence de Kullback-Leibler	Prédiction de l'image avec et sans bruit adversaire du même modèle	1
Mean teacher (Tarvainen & Valpola (2017))	Erreur quadratique moyenne	Prédiction entre le professeur et l'étudiant	1
Deep Co-training (Qiao *et al.* (2018))	Entropie croisée	Prédiction de l'image du modèle 1 avec la prédiction de l'image adversaire du modèle 2 et inversement	2
	Jensen-Shannon	Prédiction de l'image entre le modèle 1 et le modèle 2	

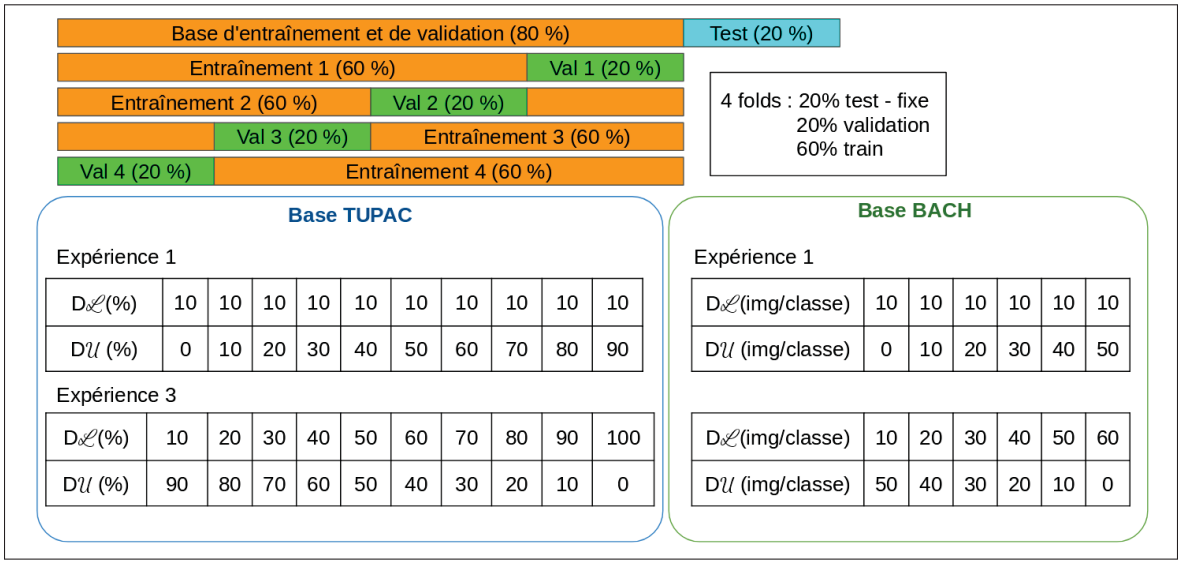
Il convient de noter que pour l'ensemble de ces approches, l'inférence est effectué avec une architecture de même taille. Le 'Nombre d'architectures entrainées' correspond au nombre de modèles profond qu'il est nécessaire d'entraîner lors de l'apprentissage de ces différentes approches. L'architecture profonde utilisée dans le cas entièrement supervisé étant fixée pour référence. Par exemple, le *Ladder Network* nécessite l'apprentissage d'un encodeur et d'un décodeur, tout les deux équivalents à l'architecture profonde supervisée. Pour le cas de *Mean Teacher*, seul le modèle étudiant nécessite d'être entraîné. Ces différences vont se répercuter sur les temps de calcul et la mémoire utilisée (voir Tab. 4.9)

CHAPITRE 3

MÉTHODOLOGIE EXPÉRIMENTALE

3.1 Expériences

Les expériences menées dans le cadre de nos travaux ont pour but d'étudier l'influence de la variation de données annotées et non annotées sur les différentes approches basées sur l'apprentissage semi-supervisé que nous avons sélectionnées. La section 4.1 des résultats est une partie à part entière de notre étude. Elle consiste à utiliser 35 patches par images pour traiter le cas de la base BACH comme le présente l'article de référence (Araújo *et al.* (2017)) pour cette compétition. Afin de traiter ce cas de figure, nous avons comparé l'utilisation de différentes techniques (section 3.3.2) pour limiter l'imprécision de l'annotation des patches. Pour cette configuration, nous avons effectué les expériences en utilisant 70% de la base pour l'entraînement et la validation, et 30% pour le test. Nous avons effectué chaque cas de figure 7 fois en utilisant des bases de validations indépendantes (voir Fig. I-3). Pour cette configuration, nous avons effectué les expériences 1 et 2.



$D_{\mathscr{L}}$(%)	10	10	10	10	10	10	10	10	10	10
$D_{\mathscr{U}}$ (%)	0	10	20	30	40	50	60	70	80	90

$D_{\mathscr{L}}$(%)	10	20	30	40	50	60	70	80	90	100
$D_{\mathscr{U}}$ (%)	90	80	70	60	50	40	30	20	10	0

$D_{\mathscr{L}}$(img/classe)	10	10	10	10	10	10
$D_{\mathscr{U}}$ (img/classe)	0	10	20	30	40	50

$D_{\mathscr{L}}$(img/classe)	10	20	30	40	50	60
$D_{\mathscr{U}}$ (img/classe)	50	40	30	20	10	0

Figure 3.1 Répartition des données pour la base BACH et TUPAC

Pour le reste de l'étude, nous avons changé le contexte expérimental afin que la base de validation soit plus représentative du problème. Nous avons donc utilisé 80% de la base totale pour l'entraînement et la validation et 20% pour le test. Dans ce cas nous avons effectué chaque expérience sur 4 bases de validation indépendantes (voir Fig. 3.1). Dans cette partie de l'étude, nous avons pu comparer l'influence apportée par le pré-entrainement sur *ImageNet* des paramètres des modèles sur les résultats. Pour cette configuration, nous avons effectué les expériences 1 et 3.

Pour l'ensemble des expériences, les bases d'entraînements avec et sans labels ont été tirées aléatoirement parmi les images restantes de la base de données une fois la base de test et de validation retirées. Ainsi, l'augmentation des données annotées et non annotées prend en compte les tirages précédents. Pour la base de données BACH, nous avons raisonné sur des tranches de 10% issues de la base de données totale. Chaque tranche représente donc 10 images/classe, soit 40 images. Pour cette base de données nous avons donc étudié les performances de 6 paliers. Ayant accès à plus de données, pour la base issue du concours TUPAC, nous avons effectué les tranches en fonction de la base totale d'entraînement. Nous avons ainsi 10 paliers correspondant à 10 % de la base d'entraînement. Les différentes expériences sont définies comme suit.

Expérience 1

La première expérience a pour but d'évaluer l'influence de la quantité de données non annotées sur les performances des différents modèles semi-supervisés. Nous avons augmenté de manière progressive le nombre de données non annotées tout en fixant une petite quantité de données annotées. Cette expérience permet d'étudier l'influence de l'ajout de données non annotées sur les différentes techniques semi-supervisées. Nous allons ainsi pouvoir observer dans quelle mesure l'augmentation de données non annotées est bénéfique aux différentes méthodes.

Expérience 2

La deuxième expérience a pour but d'évaluer l'influence de l'augmentation simultanée de la quantité de données annotées et non annotées sur les performances des différents modèles. L'expérience consiste à sélectionner à chaque fois autant de données annotées que non annotées.

Expérience 3

La troisième expérience consiste à augmenter de manière progressive la quantité de données annotées tout en conservant le nombre total de données. On se place ici dans le cas où un expert annote à chaque fois une quantité fixe d'images. Cette expérience nous permet d'observer dans quelle mesure l'ajout d'annotations nous permet d'obtenir de meilleures performances. Pour cette expérience, toutes les données sont utilisées à chaque palier.

3.2 Base de données : Tumor Proliferation Assessment Challenge 2016

3.2.1 Description

Lors de ce concours, trois tâches indépendantes pouvaient être réalisées par les participants : la prédiction du score de prolifération du cancer basée sur le dénombrement des mitoses, la prédiction du score de prolifération du cancer basée sur les données moléculaires et la détection des mitoses. Nous nous sommes intéressés à la troisième tâche de ce concours. Les bases de données mises à la disposition des participants pour cette tâche sont les suivantes :

- Une base d'entraînement composée de la base AMIDA13 plus 50 nouveaux cas (soit 73 patients). Ces différentes images sont les portions de $2mm^2$ dont on connaît la position des mitoses données par des spécialistes. Chaque cas comporte au minimum 10 HPFs (High power fields) qui correspondent à une zone d'étude sélectionnée par un pathologiste expérimenté (voir Fig. 3.2).
- Une base de test composée de 34 cas du même format que la base d'entraînement.

- Une base de données de 500 WSI annotées par niveau de grade (I, II ou III) ainsi qu'une base de 148 WSI annotées des ROI par des spécialistes sont disponibles.

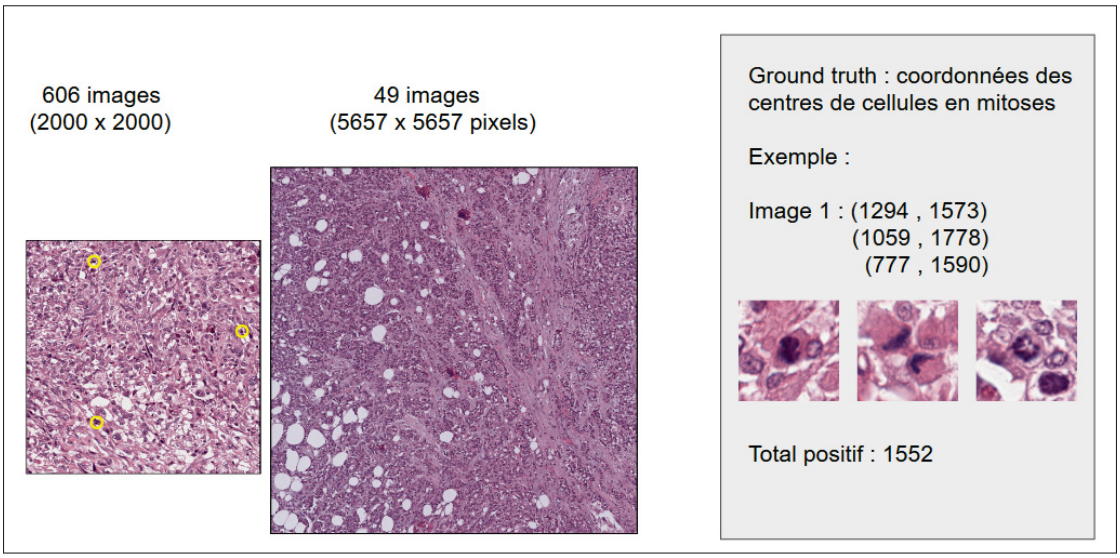


Figure 3.2 Exemple d'images de la base TUPAC (Veta *et al.* (2018a))

La base utilisée comporte 656 images décomposées en 73 patients. Dans cette base on retrouve 606 images de 2000 x 2000 pixels et 50 images de 5657 x 5657 pixels. Le but étant de déterminer les coordonnées des mitoses à partir d'images histologiques. Nous avons choisi d'étudier cette base de données en extrayant des patches centrés sur le noyau des cellules. Les coordonnées à notre disposition permettent ainsi d'annoter les patches extraits.

3.2.2 Mise en forme de la base de données

Pour étudier cette base de données, nous avons décidé d'effectuer l'apprentissage et le test de ces images à partir des patches centrés sur les cellules qui les composent. Pour cela nous nous sommes inspirés d'un logiciel (Cellprofiler (Carpenter *et al.* (2006))) utilisé par le vainqueur de la compétition pour détecter le nombre de cellules sur les images. Ce logiciel permet de détecter différents blobs à partir de seuillages successifs de l'image à étudier. Pour cela, nous avons paramétré la fonction *SimpleBlobDetector* de la librairie *OpenCV* (Bradski (2000)) qui effectue les mêmes étapes de seuillage. La détermination des différents paramètres s'est faite

empiriquement à partir des images de la base de design. L'algorithme consiste à effectuer plusieurs seuillages de manière incrémentale. Dans notre cas 8 seuillages sont effectués, de 50 à 120 par pas de 10. Les groupes de pixels communs aux différents seuillages sont considérés comme des blobs et les coordonnées de leurs centres sont extraites. Les blobs ainsi détectés sont pris en compte s'ils remplissent un certain nombre de critères. Nous avons pris en compte les blobs qui étaient répétés au moins 2 fois lors des différents seuillages et qui possèdent une aire comprise entre 20 et 2500 pixels pour éliminer les trop petits blobs qui ne sont pas des cellules, et les trop grosses masses.

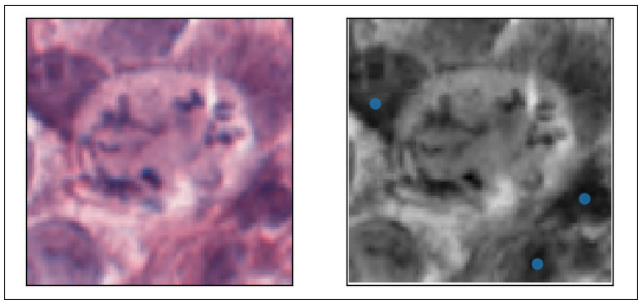


Figure 3.3 Cellule en cours de mitose non détecté. Les noyaux des cellules détectés sont marqués par un point bleu (à gauche)

Au final, parmi 1552 mitoses de la base de données seules 10 n'ont pas été détectées : 7 dans la base de design et 3 dans la base de test. La figure 3.3 représente une des cellules en cours de mitose qui n'a pas été détectée. Les points bleus sur l'image de droite représentent les centres des cellules détectés par notre méthode. Il faut noter que la séparation des images s'est faite de manière uniforme : chaque fold correspond au même nombre d'images. Cette configuration implique une limite : il se peut que les images d'un même patient servent dans différentes partition. Cette limite doit être prise en compte dans l'interprétation des résultats. L'une des principales difficultés de ce concours est le débalancement des classes. Si l'on dénombre les cellules présentes sur l'ensemble des images, les cellules en cours de mitose représentent 0.20% de la base de données. Pour palier à ce problème, les vainqueurs du concours ont sous échantillonné les patches négatifs et sur échantillonné les patches positifs. Afin de focaliser

l'apprentissage sur les cas difficiles, il est nécessaire d'échantillonner en priorité les patches négatifs particulièrement ressemblants aux mitoses. Pour cela, les candidats entraînent un premier modèle, qualifié d' "oracle", dont les prédictions seront utilisées pour séparer les pacths négatifs en patches 'faciles' et 'difficiles'. Les patches 'difficiles' correspondent aux patches négatifs de la base d'entraînement qui sont classés par le modèle en tant que mitoses.

3.2.2.1 Choix des patches

La stratégie utilisée pour sélectionner les patches consiste à augmenter le nombre de patches positifs afin d'obtenir 100 patches positifs et 100 patches négatifs pour chaque image. Pour les patches négatifs, 50 patches sont des faux positifs sélectionnés par un modèle 'oracle' et les 50 autres ont été sélectionnés aléatoirement parmi le reste des patches négatifs. Cette stratégie a pour objectif de focaliser l'apprentissage sur les exemples particulièrement difficiles à reconnaître. L'augmentation des patches est effectuée par rotation, réflexion et variation de la luminosité, du contraste et de la saturation. Pour le test, tous les patches sont classés par le modèle, sans balancement entre les patches positifs et négatifs. Le choix des patches est nécessaire pour rendre l'apprentissage possible. La base de données comptabilise plus d'un million de patches au total, la réduction du nombre de patches étudié permet de limiter le temps d'apprentissage des différents modèles.

Le modèle 'oracle' est entraîné à partir de toutes les images d'une itération, en choisissant 100 patches négatifs aléatoirement pour 100 patches positifs augmentés par image. Ce modèle est testé sur une base de validation distribuée de la même manière. Une fois entraîné, ce modèle est utilisé pour classer les patches d'entraînement afin de sélectionner les patches faux positifs (voir Fig. 3.4).

L'arrêt de l'entraînement est obtenu à l'epoch permettant d'obtenir le meilleur score de la fonction de coût de la base de validation (entropie croisée) afin d'éviter le surapprentissage. Nous avons utilisé le score F1 (voir section 3.4) pour le calcul des performances pour ne pas prendre en compte le débalancement. Pour l'ensemble des expériences, nous ne pouvons pas

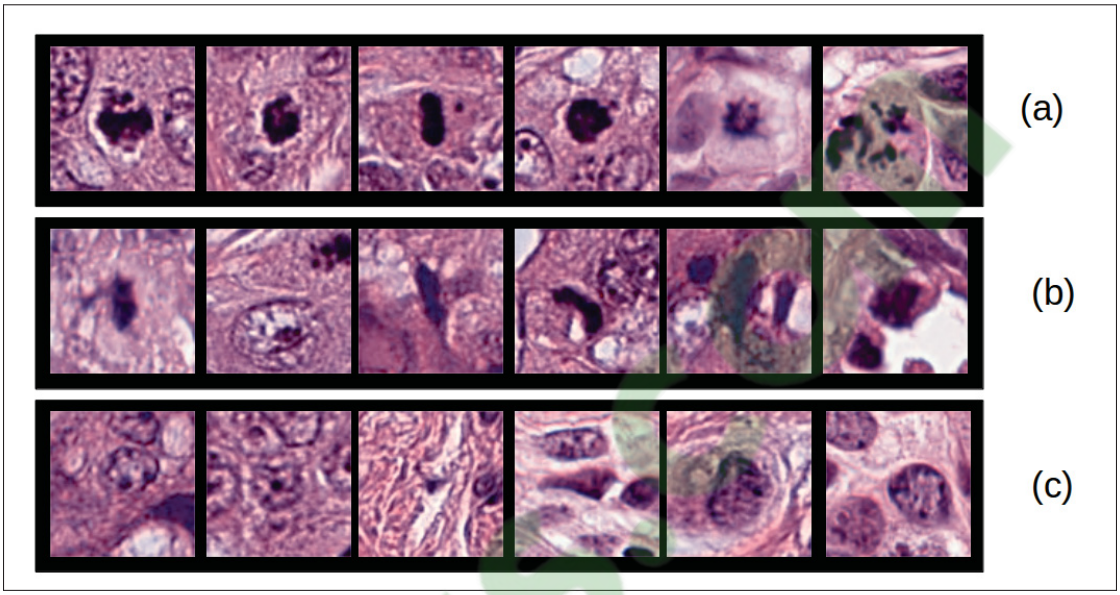


Figure 3.4 Exemple de patches : en cours de mitose (a), faux positifs du modèle oracle (b), négatifs sélectionnés aléatoirement (c)

utiliser la base de validation entière car elle est trop grande. Les différentes bases de validations sont constituées de 800 patches positifs, 10 000 patches négatifs et 10 000 patches négatifs 'difficiles' sélectionnés par l'oracle. Nous avons choisi de déséquilibrer la base de validation pour qu'elle corresponde davantage à la base de test. Nous avons augmenté les patches positifs pour en avoir un nombre fixe pour différentes itérations et nous avons fixé un nombre plus important de patches négatifs et négatifs 'difficiles'. La base de test est composée de tout les patches détectés sur les images sélectionnées, soit de 347 patches positifs et 203 169 patches négatifs.

3.3 Base de données : Grand Challenge on BreAst Cancer Histology images

3.3.1 Description

Le concours *Grand Challenge on Breast Cancer Histology images* (BACH) fait partie de la *5th International Conference on Image Analysis and Recognition* (ICIAR). C'est une suite à la compétition effectuée lors de la conférence *Bioimaging* 2015 qui été focalisé sur la re-

connaissance de grands patches histologiques en 4 classes. Cette compétition comprenait 140 images histologiques de 2048 x 1536 pixels (120 pour l'entraînement et 20 pour le test) issus de différents échantillons de biopsie.

Le concours BACH a pour but d'encourager la recherche autour de la détection et du diagnostic automatique d'images histologiques de cancer du sein. Le challenge se compose de deux parties A et B : la reconnaissance automatique de grands patches (2048x1536 pixels) en quatre classes (normal, benign, insitu carcinoma, invasive carcinoma) et la segmentation de ces mêmes quatre classes à partir de *whole slide images*.

La partie A est composée de 400 images de 2048 x 1536 pixels répartis équitablement entre 4 classes : normal, benign, insitu carcinoma, invasive carcinoma. Un exemple d'image est représenté sur la figure 3.5.

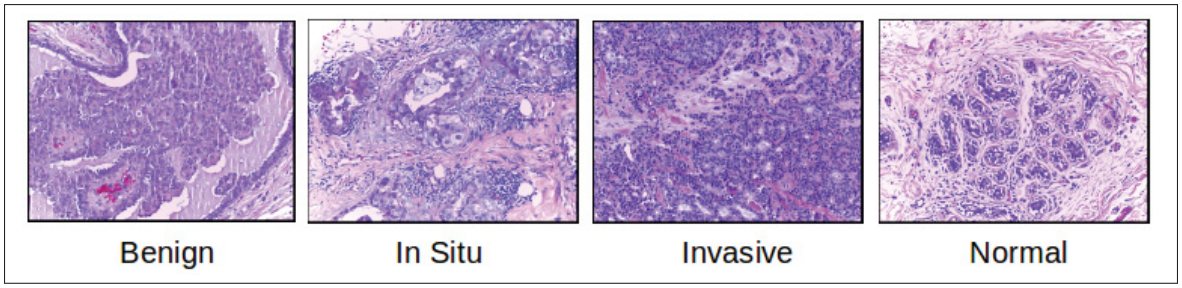


Figure 3.5 Exemple d'images histologiques pour la partie A (Aresta *et al.* (2018))

Cette base de données est une extension de la base utilisée lors du challenge Bioimaging2015. Les images ont toutes la même résolution ($0.42\,\mu\mathrm{m}$ x $0.42\,\mu\mathrm{m}$ par pixel). Il n'est néanmoins pas mentionné à partir de combien de patients proviennent ces échantillons ni le type de microscope utilisé. L'annotation des images a été faite par deux pathologistes, seules les images annotées identiquement par les spécialistes ont été gardées. Une base de test est fournie aux participants pour évaluer et classer les différents modèles.

La partie B est composée de 30 *whole slide images* dont 10 d'entre elles sont annotées à chaque pixel dans les quatre mêmes classes : normal, benign, insitu carcinoma, invasive carcinoma.

Dans ce projet, nous nous sommes focalisés sur la partie A de la compétition pour évaluer les différentes approches semi-supervisées sélectionnées.

3.3.2 Techniques pour traiter la supervision imprécise : le cas de la base BACH

La première approche pour étudier cette base consite à apprendre à partir de l'image complète. Le pré-entraînement sur ImageNet joue un rôle primordial : il permet d'obtenir des résultats significativement supérieurs et permet de faciliter l'apprentissage. De plus, cette méthode est très dépendante du nombre d'images : dans le cas semi-supervisé, la petite taille de la base d'entraînement rend l'utilisation d'un grand nombre de petits patches plus performant. Le tableau 3.1 représente les scores d'exactitude obtenus à l'aide d'un ResNet-18 pré entraîné sur ImageNet. L'approche la plus directe pour étudier cette base est d'utiliser les images complètes comme données pour l'apprentissage. C'est cette approche, couplée avec l'utilisation d'un pré-entrainement sur *ImageNet*, qui a obtenu les meilleurs résultats lors de la compétition. Nous nous sommes rendu compte parmi les gagnants de la compétition, qu'il était courant de réduire la dimension de ces patches (Kwok (2018),Chennamsetty *et al.* (2018)). Pour étudier l'impact de la taille des patches sur les scores d'exactitude en classification, nous avons fait varier la taille des patches extraits que nous avons redimensionnés en patch de 256 x 256. Les patches ont été extraits avec du chevauchement, à raison de 165 patches de taille 256x256, 35 patches de taille 512 x 512 pixels, 6 patches de taille 1024 x 1024 pixels et 2 patches de taille 1536 x 1536 pixels. Dans cette première expérience, nous avons attribué à chaque patch extrait le label de l'image dont il est issu. Le tableau 3.1 représente les scores d'exactitude obtenus sur la base de test à l'aide d'un ResNet-18 pré entraîné sur *ImageNet* en faisant varier la taille des patches utilisées. Les scores sur les images de la base de test ont été obtenus à partir de la moyenne calculé sur les scores de probabilités, obtenues sur l'ensemble des patches constituant cette même image.

L'entraînement a été effectué avec 240 images (60 de chaque classe) et validé sur 40 images (10 de chaque classe). Les tests ont été effectués 7 fois, en prenant en compte 7 bases de validation indépendantes pour l'apprentissage du modèle. On constate que les résultats sont

Tableau 3.1 Scores d'exactitude (fusion des patches par somme des probabilités) obtenus sur la base de test en fonction de la taille des patches

256 (165 patches)	**512 (35 patches)**	**1024 (6 patches)**	**1536 (2 patches)**
87.85 ± 4.87	90.35 ± 6.36	89.64 ± 4.19	87.14 ± 4.87

meilleurs si l'on utilise 35 patches représentant de plus petites régions. Notre étude va se baser, dans un premier temps, sur l'utilisation de 35 patches par image pour entraîner les différentes techniques d'apprentissage semi-supervisées. Dans un deuxième temps, nous allons utiliser les images complètes en tant que patch afin de se placer uniquement dans le cas d'un apprentissage semi-supervisé (pure).

Pour l'étude des patches nous allons extraire 35 patches de 512 x 512 pixels que nous avons redimensionnés. Dans ce cas de figure, deux difficultés sont à prendre en compte :
(1) Comment annoter les patches que l'on va extraire ?
(2) Comment effectuer la prédiction de l'image complète à partir des patches ?

3.3.2.1 Annotation des patches

Les différentes classes témoignent d'un certain avancement de la tumeur. De manière grossière, Normal = pas de tumeur, Bénin = début de tumeur ou tumeur localisée, In Situ = tumeur présente uniquement dans des canaux galactophores (tumeur "emprisonnée" dans une membrane présentent Fig. 3.7) et Invasif = tumeur qui prolifère. Il faut noter que la classe invasive peut témoigner de plusieurs cas de figure. La rupture de la membrane dans le cas In Situ est qualifiée d'invasif (voir image de gauche Fig. 3.6). Cette même classe peut aussi représenter un morceau de tissu plus homogène (image de droite Fig. 3.6). Il faut noter que la tumeur fait référence à des cellules qui se multiplient de manière non contrôlée.

La question est alors de savoir si la tumeur à une influence sur son environnement, ou si elle se limite à certaines parties de l'image.

Si l'on fait l'hypothèse qu'il y a une indépendance entre les classes, dans ce cas, attribuer directement le label de l'image complète à celui des patches qui la compose est une solution cohérente. Les performances représentées dans le tableau 3.1 sont alors une bonne représentation des performances auxquels il faut s'attendre pour ce type de problème. Dans ce cas on suppose alors que chaque patch est représentatif du label attribué globalement.

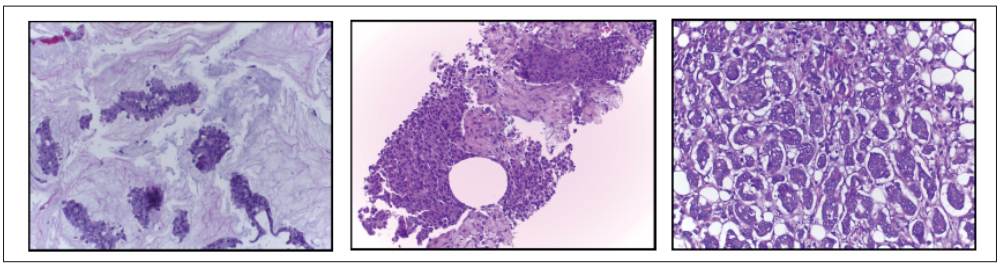


Figure 3.6 Exemple d'images de la classe Invasive

Dans le deuxième cas de figure, on fait l'hypothèse que les classes ne sont pas indépendantes entre elles. On suppose alors que les différents stades de tumeurs sont localisés sur l'image. C'est-à-dire qu'il peut exister, par exemple, des patches de la classe Normal dans des images de la classe In Situ. Dans ce cas, on s'attend à obtenir de meilleures performances en sélectionnant les patches les plus représentatifs pour chaque classe. Cette question est plus complexe qu'il n'y paraît : l'algorithme automatique va apprendre à reconnaître des caractéristiques spécifiques à chaque classe qui ne sont pas nécessairement reliées à celles utilisées par les pathologistes.

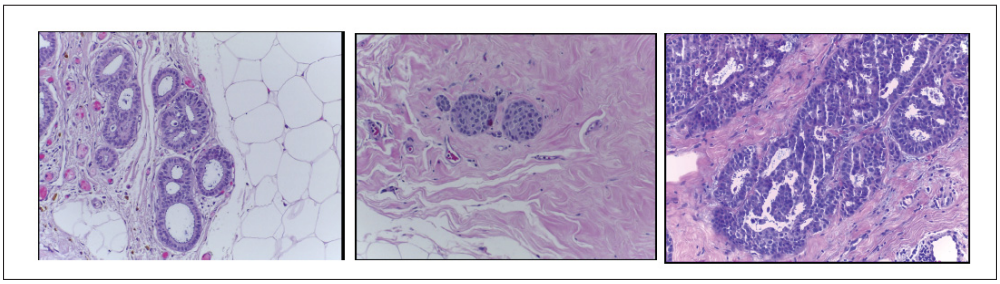


Figure 3.7 Exemple d'images de la classe In Situ

Nous allons comparer les résultats obtenus avec ces deux approches pour savoir quelle est la méthode d'apprentissage la plus adaptée pour cette base de données dans le cas d'une

supervision incomplète. Étudions alors plus en détail le deuxième cas de figure : plusieurs classes peuvent être présentes dans une image. On se place alors dans un contexte de label bruité comme défini précédemment. Nous avons donc cherché des méthodes pour limiter l'influence de patches bruités. La question qui se pose alors est : comment sélectionner les patches qui représentent le plus le label de l'image complète ? La première approche consiste à enlever les patches possédant une majorité de pixels blancs. Ces zones représentent l'arrière-plan : il s'agit soit des limites des l'échantillons observé soit de déchirure liée à leur découpe. Ce critère est souvent utilisé pour effectuer une présélection grossière des patches d'images histologiques. Ces patches, qui représentent en grande partie l'arrière-plan, peuvent ajouter du bruit lors de l'apprentissage puisqu'ils peuvent être présents sur des images de différentes classes.

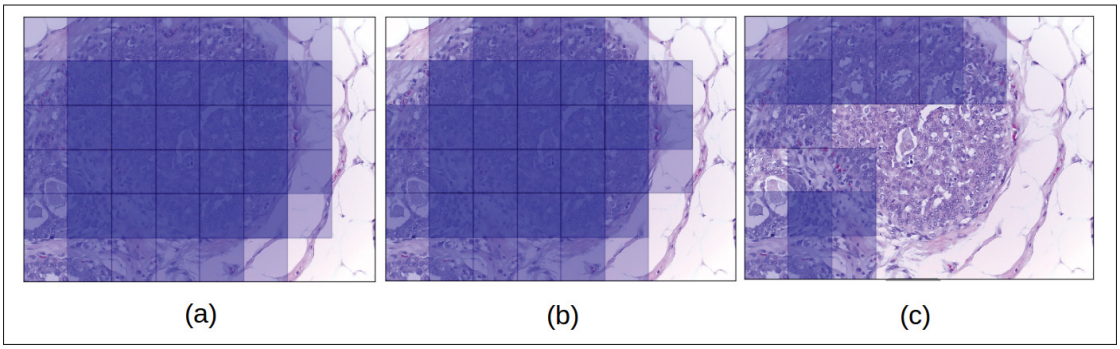


Figure 3.8 Sélection des différents patches suivant le critère de seuillage (a), du nombre de cellules (b) et de choix par un modèle (c)

La figure 3.8 (a) représente une image dont les patches ont été sélectionnés par le critère de seuillage. Pour ce critère, nous avons dans un premier temps normalisé les images pour limiter l'influence des conditions des différents contrastes dans les images. Une fois les images normalisées, les patches sont extraits de chaque image et convertis en teintes de gris. On dénombre ensuite les pixels supérieurs à une certaine valeur déterminée empiriquement (190 dans notre cas). L'ensemble des patches des images pour l'entraînement est ensuite classé en fonction de leur nombre de pixels supérieur au seuil défini empiriquement (75 % dans notre cas), correspondant aux pixels de l'arrière-plan.

L'objectif de la classification étant de différencier les différents stades de tumeurs, pour la deuxième approche nous nous sommes focalisés sur les patches possédant le plus de cellules sur chaque image. Pour ce critère l'algorithme de détection des blobs proposés par la librairie OpenCV a été reparamétré empiriquement pour ce cas. Il a été nécessaire dans un premier temps de normaliser les couleurs des différentes images pour être capable d'avoir une extraction robuste entre deux images n'étant pas teintées de la même façon.

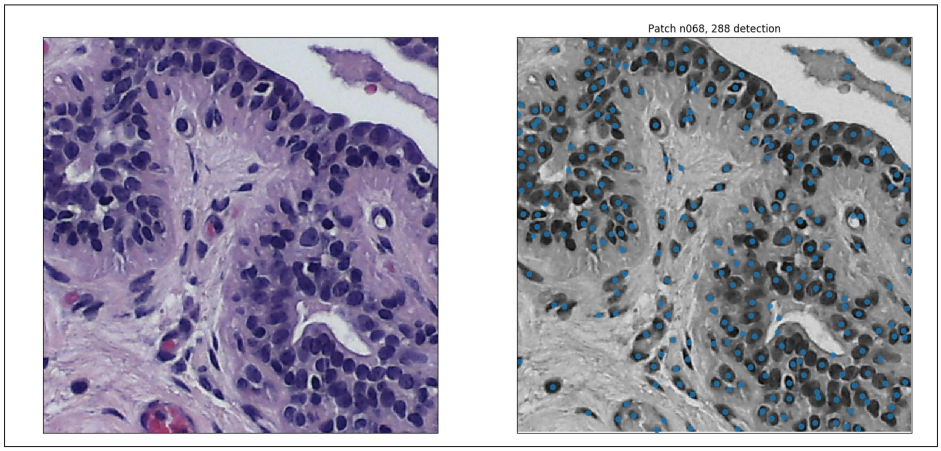


Figure 3.9 Exemple du critère de dénombrement des cellules

L'algorithme consiste à effectuer plusieurs seuillages de manière incrémentale. Dans notre cas 8 seuillages ont été effectués, de 50 à 120 par pas de 10. Les groupes de pixels communs aux différents seuillages sont considérés comme des blobs et les coordonnées de leurs centres sont extraites. Les blobs ainsi détectés sont pris en compte s'ils remplissent un certain nombre de critères. Nous avons pris en compte les blobs qui étaient répétés au moins 2 fois lors des différents seuillages et qui possèdent une aire comprise entre 20 et 2500 pixels pour éliminer les trop petits blobs qui ne sont pas des cellules, et les trop grosses masses. La figure 3.9 représente un exemple de détection pour un patch. Les points bleus sur l'image de droite représentent les centres des blobs détectés. Cet algorithme est utilisé pour extraire les coordonnées et le nombre de cellules détectées pour chaque patch. Les patches sont alors classés en fonction du nombre de blobs détectés. Pour les différentes expériences, nous avons sélectionné 75 % des patches possédant le plus de cellules pour chaque image.

Nous avons voulu tester une méthode basée sur l'apprentissage d'un modèle. Nous entraînons dans un premier temps le modèle sur l'ensemble des patches, en leur attribuant le label de leur image d'origine. Une fois entraîné, nous sélectionnons dans la base d'entraînement les patches correctement prédits par le modèle. Cette nouvelle base épurée sera ensuite utilisée pour entraîner un nouveau modèle. Plusieurs articles sont basés sur une méthode de sélection des patches similaires (Frénay & Verleysen (2014))(Hou *et al.* (2016)). Le principe est de laisser au modèle le choix des patches les plus représentatifs de chaque classe. Il faut néanmoins faire attention au surapprentissage, pour éviter que le modèle apprenne par coeur les images de la base d'entraînement. La figure 3.8 (c) représente un exemple de sélection de patches avec ce critère. Comme on peut le constater, ce choix est difficilement interprétable, il ne dépend que de la modélisation du problème par le modèle.

Enfin, nous avons voulu tester une méthode d'apprentissage prenant en compte le bruit présent dans les échantillons. Nous nous sommes orientés vers le lissage des labels (*label smoothing*)(Szegedy *et al.* (2016)), car c'est une méthode simple et efficace qui a été largement pratiqué ces deux dernières années. En déplaçant les bornes d'objectif de la fonction de coût, on donne un certain degré de validité à chaque label. Avec cette méthode, on peut prendre en compte les erreurs potentielles au niveau du label des patches.

3.3.2.2 Prédiction de l'image complète

La deuxième question qui se pose alors est : comment obtenir une classification de l'image globale à partir d'un modèle de reconnaissance de patch ? Les techniques classique de fusion supposent une certaine distribution des instances. En faisant la moyenne des scores obtenus sur une image, on suppose qu'il y a une majorité de patches de la classe présentes sur l'image. Il en est de même pour le vote majoritaire. L'attribution du label à partir des top k patches possédant les plus fortes prédictions suppose que les patches qui activent le plus le modèle sont représentatifs de la classe en question. Toutefois si l'on suppose que les zones représentant les tumeurs sont localisées, le modèle peut très bien, dans le cas In Situ par exemple, obtenir des prédictions élevées pour des patches de la classe Normal et In Situ.

Ainsi, en fonction des exemples, la classe de l'image peut être représentée en minorité sur l'image 3.7. Dans cette partie, outre l'utilisation des méthodes classiques, nous allons expérimenter plusieurs méthodes de fusion des patches dans le but d'améliorer le score d'exactitude obtenu.

WILDCAT

L'approche *Wildcat* (Durand *et al.*) consiste à produire des cartes de probabilité pour chaque classe et à sélectionner un certain nombre d'instances k faisant partie des extremums de ces cartes de probabilité. De plus, cette approche permet d'entraîner le modèle sur des patches et d'inférer sur des images, puisque ce dernier est basé sur une architecture entièrement convolutionnelle. Cette méthode est composée de trois grandes parties, schématisées sur la figure 3.10. Les dimensions des cartes de caractéristiques sont indiquées en dessous des différents modules, en prenant l'exemple d'une image et d'un patch.

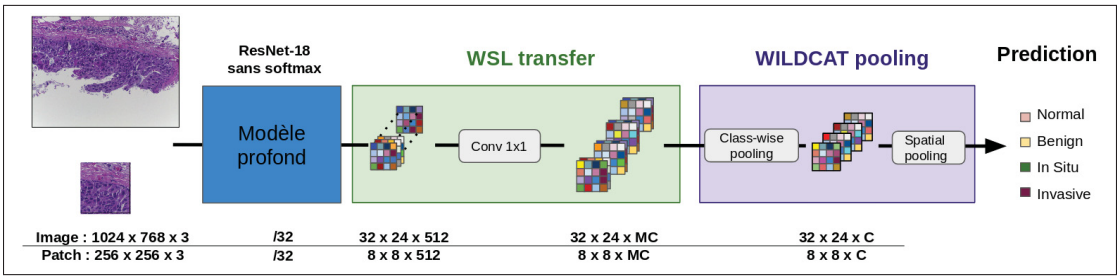


Figure 3.10 Schématisation de la méthode WILDCAT (Durand *et al.*)

La première partie consiste à extraire les caractéristiques de l'image d'entrée, indépendamment de sa taille. Cette particularité permet de rendre l'entraînement et l'inférence indépendants de la taille des données d'entrées. On peut par exemple entraîner le modèle d'extraction sur des patches et utiliser celui-ce pour produire des prédictions à la fois sur des patches et des images complètes. L'approche WILDCAT consiste en l'application de deux modules successifs : un module de transfert multi-map appelé *Weakly Supervised Learning transfer* (WSL tranfer) et une étape d'agrégation appelée *WILDCAT pooling*. Le premier module consiste à changer l'apparence et le nombre de cartes de caractéristiques à la sortie de l'extracteur de caractéristiques.

Le but vise à répartir les cartes équitablement entre les différentes classes du problème. Cette étape est effectuée à l'aide d'une couche de convolution possédant un *kernel* de 1x1. Le nombre de cartes par classe est déterminé par un hyperparamètre M commun à toutes les classes. Le deuxième module comporte deux étapes. La première consiste à fusionner les différentes cartes pour obtenir une carte de probabilité par classe. La deuxième étape consiste à obtenir un score de probabilité à partir de ces cartes en faisant la somme pondérée des k régions les plus probables et les moins probables pour chaque classe. Nous avons utilisé l'architecture ResNet-18 pré-entraînée et nous avons affiné l'apprentissage à l'aide des images complètes uniquement sur la partie WILDCAT. De cette manière, on limite le surapprentissage dû au faible nombre d'images, car seule une convolution doit être apprise.

Histogrammes

Cette approche est basée sur les travaux de Hou *et al.* (2016). Un histogramme est calculé pour chaque image d'entraînement (voir Fig. 3.11) pour permettre de créer une base d'entraînement spécifique à la classification des images à partir de la prédiction obtenue sur les différents patches. Le but est d'apprendre un module de classification basé sur la distribution des labels des patches sur chaque image.

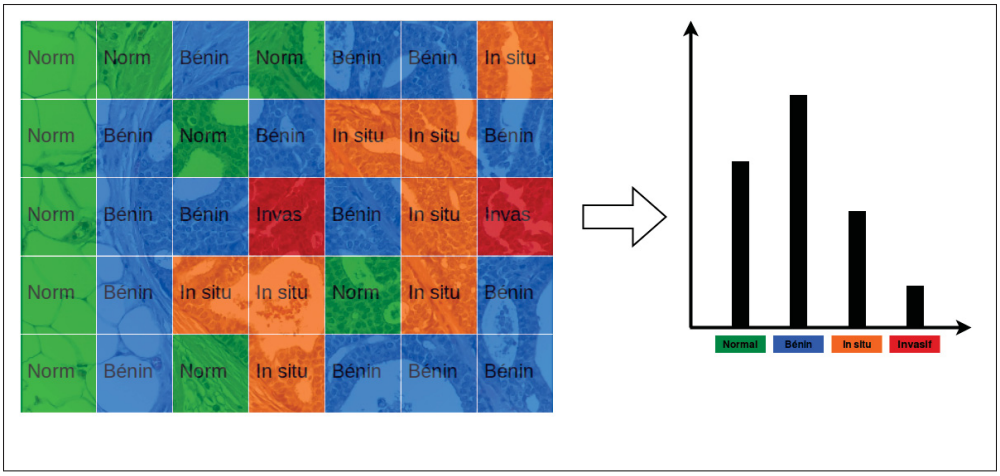


Figure 3.11 Exemple d'histogramme permettant d'entraîner différents modules de classifications

Hou *et al.* (2016) ont comparé l'utilisation d'un SVM et d'un modèle de régression logistique pour cette tâche. Cette méthode permet de prendre en compte, dans le cas d'un module d'extraction parfait, les patches dont le label ne correspond pas à celui de l'image globale. En pratique cette méthode souffre du surapprentissage de la base d'entraînement, ce qui biaise le module de classification entraîné sur ces mêmes images. Une attention toute particulière doit être apportée à ce type d'apprentissage, car les mêmes images sont utilisées pour l'apprentissage de deux modèles successifs. Nous avons utilisé cette méthode en utilisant la base d'entraînement et de validation comme base de design du SVM.

3.4 Mesures de performances

Pour observer le comportement des différentes approches, nous allons aussi étudier les matrices de confusions obtenues sur certains essais. Pour étudier les performances, nous allons calculer le score F1 dans le cas de la base TUPAC et le score d'exactitude pour la base BACH, ce qui correspond aux mesures respectivement utilisées dans les compétitions. Pour étudier plus en détail le *Deep Co-training*, nous allons utiliser différentes mesures de diversité, afin de quantifier la différence obtenue entre les modèles concurrents lors de son apprentissage, sur différents essais.

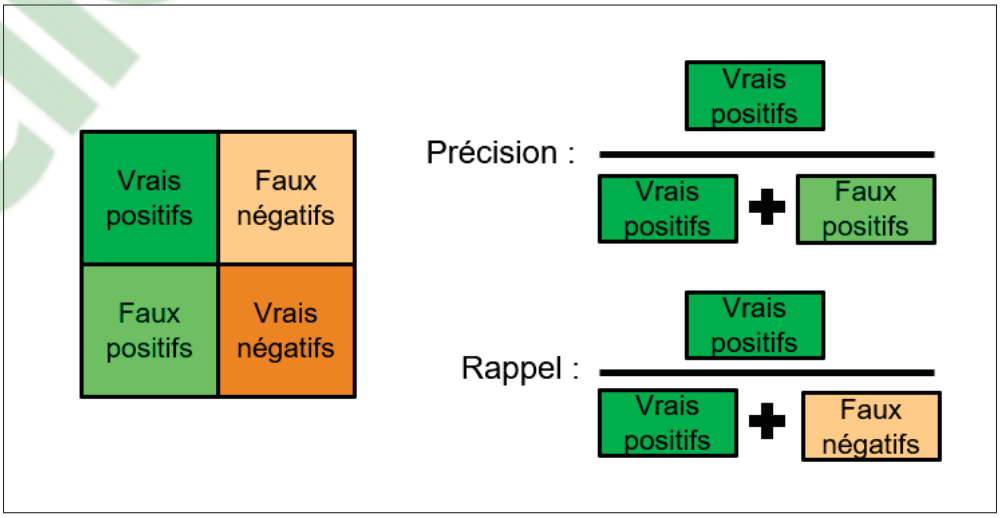


Figure 3.12 Schématisation d'une matrice de confusion (cas binaire)

La matrice de confusion (Fig. 3.12) est une matrice qui regroupe les différentes prédictions effectuées par un modèle en quatre catégories : les vrais positifs et négatifs sont les patches correctement prédits, les faux positifs correspondent aux patches négatifs prédits en tant que positifs et les faux négatifs correspondent aux patches positifs prédits comme négatifs. Le score $F1$ est une mesure qui prend en compte le rappel et la précision.

$$F_1 = 2\,\frac{précision \times rappel}{précision + rappel} \tag{3.1}$$

Il est particulièrement intéressant dans le cas de données non balancées puisqu'il ne prend pas en compte le nombre de vrais négatifs. Le score d'exactitude correspond aux rapports entre l'ensemble des exemples correctement prédits par le modèle avec la totalité des exemples.

$$S_{exactitude} = \frac{VraisPositifs + VraisNégatifs}{VraisPositifs + VraisNégatifs + FauxNégatifs + FauxPositifs} \tag{3.2}$$

Concernant l'approche *Deep Co-training*, nous allons aussi nous intéresser à mesurer la différence entre les deux modèles obtenus après l'entraînement. Cette mesure a pour but de mettre en évidence la différence de point de vue obtenue par la méthode proposée et de la comparer à d'autres méthodes de création de diversité. Pour cela, nous allons nous focaliser sur les mesures de diversité par paires, permettant de comparer les prédictions de deux modèles. Il existe de nombreuses recherches regroupant ces mesures (Kuncheva & Whitaker (2003),Zhou (2012)). Ces mesures sont basées sur le nombre d'exemples similairement et différemment prédit par l'un ou par l'autre des modèles. Le tableau 3.2, aussi appelé tableau de contingence (Zhou (2012)), représente les relations entre deux modèles de prédictions. Dans ce tableau C_1 et C_2 représentent les deux classifieurs à comparer.

Tableau 3.2 Tableau de relations entre $C1$ et $C2$

	$C1$ prédit 1	$C1$ prédit 0
$C2$ prédit 1	a	b
$C2$ prédit 0	c	d

Les différentes valeurs a, b, c, d sont obtenues en faisasnt le bilan des prédictions obtenues pour deux classifieurs sur les images de la base de test. Par exemple, a correspond au nombre d'exemples de la base de test similairement prédits comme positif par les deux modèles, b correspond à l'ensemble des exemples prédit comme positif par C_2 et comme négatif par C_1 et ainsi de suite.

Nous nous intéresserons ici à deux mesures : le coefficient de corrélation σ (Sneath *et al.* (1973)) et la statistique kappa κ (Cohen (1960)). Le coefficient de corrélation σ est calculé de la manière suivant :

$$\sigma = \frac{ad - bc}{\sqrt{(a+b)(a+c)(c+d)(b+d)}} \tag{3.3}$$

Si le coefficient est maximale (=1), cela signifie que les deux modèles effectuent les mêmes prédictions, en revanche plus ce coefficient est petit, moins ces modèles sont similaires. Un coefficient négatif indique qu'une majorité d'exemples ne sont pas prédits de la même manière par les deux modèles. La statistique kappa κ prend en compte la probabilité que les deux classifieurs fournissent les même prédictions par hasard. La probabilité P_c que les deux modèles prédisent le même label est calculée par :

$$P_s = \frac{a+d}{a+b+c+d} \tag{3.4}$$

La probabilité P_h que les deux modèles prédisent le même label par hasard peut être calculée par :

$$P_h = \frac{(a+b)(a+c)+(c+d)(b+d)}{(a+b+c+d)^2} \tag{3.5}$$

La statistique kappa est alors obtenue suivant l'équation :

$$\kappa = \frac{P_s - P_h}{1 - P_h} \tag{3.6}$$

Pour cette mesure, un score de 1 indique que les deux modèles produisent les même prédictions, un score de 0 indique que la probabilité P_h et P_s sont égales, ce qui signifie que les deux classifieurs prédisent le même label par hasard.

3.5 Protocole expérimental

Pour les différentes approches, nous avons choisi l'algorithme d'optimisation ADAM pour mettre à jour ces paramètres. Afin de lutter contre le surapprentissage, plusieurs stratégies ont été mises en place. Nous avons paramétré, sur une petite quantité de données, les augmentations des patches que nous avons utilisées dans l'ensemble de nos expériences. Ces transformations consistent en 4 rotations suivies de réflexion de manière aléatoire pour chaque patch extrait. Ces transformations permettent d'augmenter le nombre de données puisque le problème étudié ne dépend pas de l'orientation des tissus. La figure 3.13 donne un exemple des types de transformation effectués sur les données.

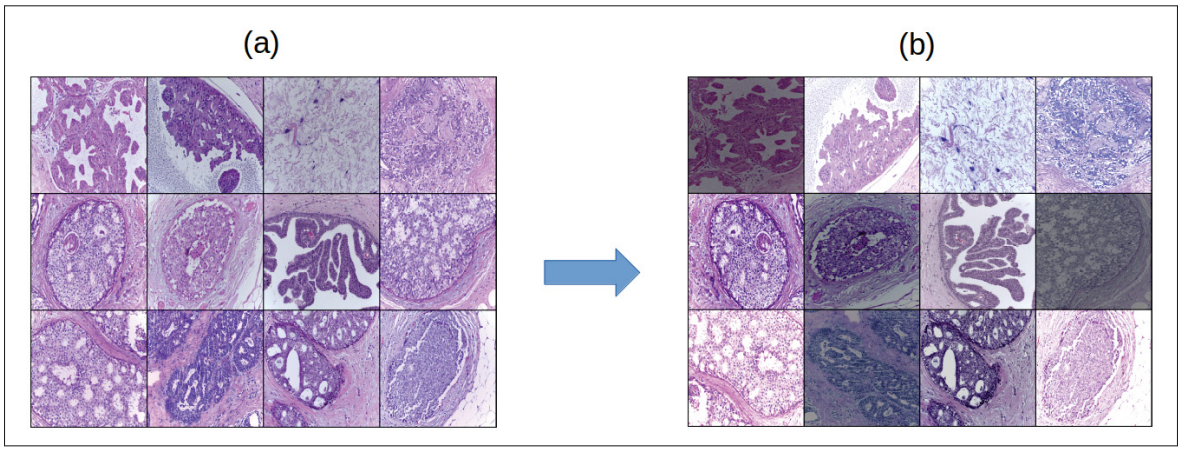


Figure 3.13 Exemple d'augmentation des données. (a) Portion de batch sans augmentation (b) Portion du même batch avec transformation

Nous avons également appliqué une variation aléatoire de la luminosité, du contraste, de la saturation et de la teinte. De telles augmentations ont pour objectif d'améliorer l'apprentissage du modèle profond en limitant la mémorisation des données d'entraînement. De plus, cette variation du contraste rend le modèle robuste aux différentes variations de teintes liées au colorant hematoxyline-éosine. Nous avons ensuite mis en place un coût supplémentaire de régularisation basé sur la norme L1 et L2 des paramètres du réseau. Ces coûts pénalisent le modèle en fonction du poids de ses paramètres. La norme L1 appliquée sur les paramètres du réseau vise à faire tendre vers 0 les paramètres ayant un faible impact sur la prédiction. La

norme L2 va pénaliser le poids total des hyperparamètres, en visant en priorité les plus élevés. En notant θ_i les paramètres du modèle pour le batch i, la fonction de coût total s'écrit :

$$\mathcal{C}_{totale} = \mathcal{C}_{modèle} + \lambda_{L1} \sum_{i=1}^{n} |\theta_i| + \lambda_{L2} \sum_{i=1}^{n} \theta_i^2 \tag{3.7}$$

Les hyperparamètres λ_{L1} et λ_{L2} permettent de donner plus ou moins d'importance à ces coûts de régularisation et sont déterminés empiriquement. Nous avons fixé les différentes variables aléatoires jouant un rôle dans l'apprentissage (ordre de présentation, augmentation des données et initialisation des paramètres des modèles) afin que les différentes méthodes soient comparées dans les mêmes configurations. La recherche d'hyperparamètre s'est faite en deux temps : la recherche des hyperparamètres communs et la recherche au cas par cas.

Tableau 3.3 Intervalle de recherche des hyperparamètres

Approches	**Hyperparamètres**	**Intervalle de recherche**
Architecture supervisée	Taux d'apprentissage	[1e-3,5e-4,1e-4,5e-5,1e-5]
	Norme L1	[0,1e-3,1e-4,1e-5]
	Norme L2	[0,1e-3,1e-4,1e-5]
VAT	ϵ_{VAT}	[0.1,0.2,0.5,0.8,1,2]
	λ_{VAT}	fixé à 1
Mean Teacher	λ_{MT}	[1,10,20,30]
	α	[0.9,0.99,0.999]
Deep Co-training	ϵ_{DCTR}	[0.01,0.02,0.1,1]
	λ_{cot}	[1,10,20,30]
	λ_{dif}	fixé à 0.5
Ladder Network	σ_{bruit}	[0.1,0.5,1]
	$\lambda_{LAD}{}^{1}$	[500,1000,2000,5000]
	$\lambda_{LAD}{}^{2}$	[0.1,1,10,100]
	$\lambda_{LAD}{}^{>2}$	[0.01,0.1,1]

Pour l'ensemble des expériences, nous avons dans un premier temps cherché le taux d'apprentissage et les poids des normes L1 et L2 adéquats à partir de l'architecture de base. Une fois le taux d'apprentissage fixé, nous effectuons la recherche pour les hyperparamètres spécifiques à chaque approche dans un intervalle proche des valeurs utilisé dans leur article d'origine. Cette recherche est effectuée par grille : nous avons essayé toutes les combinaisons possibles dans

l'intervalle définies. Nous avons cherché à optimiser les performances sur un tirage des données, les hyperparamètres étant conservés pour les autres tirages de la même configuration. Le critère d'arrêt porte sur la fonction de coût la plus basse obtenue sur la base de validation afin de limiter le surapprentissage. Le tableau 3.3 répertorie les valeurs utilisées pour la recherche d'hyperparamètres pour les différents modèles. Il faut aussi noter que lors de l'utilisation de critères, nous appliquons ces mêmes critères sur la base de validation pour étudier les performances sur une base homogène avec celle utilisée durant l'entraînement. Les résultats présentés ont tous été obtenus sur la même base de test sur laquelle aucun critère n'a été appliqué.

Pour le choix de l'architecture, pour la base BACH, nous avons choisi d'utiliser comme base de nos modèles un ResNet-18 (He *et al.* (2016)) pré-entraîné sur *ImageNet* pour sa popularité et ses performances. Le cas du *Ladder Network* est particulier puisqu'il nécessite de modifier l'architecture profonde sous-jacente. Utiliser des poids pré-entraînés n'aurait pas vraiment de sens puisqu'on ajoute des couches entre chaque bloc résiduel. L'apporche *Ladder Network* sera donc comparer aux autres approches dans le cas sans pré apprentissage. Pour la base TUPAC, nous avons décidé, pour l'apprentissage sans pré-entrainement, d'utiliser le modèle utilisé par l'équipe arrivée 2ème lors de la compétition (Fig. 1.7 partie (b)). C'est une architecture qui est plus robuste au surapprentissage (moins grand nombre de paramètres et utilisation de *dropout*) tout en permettant d'obtenir des résultats compétitifs.

CHAPITRE 4

RÉSULTATS EXPÉRIMENTAUX ET ANALYSES

Le but des expériences effectuées vie à déterminer quelle est l'incidence de l'ajout de données non annotées aux performances de différentes approches faisant l'état de l'art pour l'apprentissage semi-supervisé. Dans un premier temps, nous avons utilisé les différentes techniques d'apprentissage semi-supervisées sur la base CIFAR10, en appliquant le modèle Conv-Large (Tab. 1.1). Nous nous sommes appuyé sur l'approche de l'article *Deep Co-training*, c'est-à-dire que nous avons utilisé les couches de normalisation du batch classique et que nous avons appliqué comme augmentation une réflexion horizontale et une translation de plus ou moins deux pixels (Qiao *et al.* (2018)).

Tableau 4.1 Erreurs de test (%) sur CIFAR10

DCTR	**VAT**	**Mean Teacher**
11.96 ± 0.20	23.37 ± 0.38	13.43 ± 0.20

Les modèles ont été évalués à partir des différents hyperparamètres mentionnés dans leurs articles d'origine. Le tableau 4.1 présente les scores d'erreurs d'exactitude obtenue sur la base de test. De manière générale, on constate que ces scores sont inférieurs à ceux reportés sur les différents articles, mais que l'ordre de performances est identique.

4.1 Première partie des expériences : la base BACH

Le tableau 4.2 représente les résultats des meilleurs participants de cette compétition.

Tableau 4.2 Taux de classification obtenus sur la base de test de la base BACH (Partie A) (Aresta *et al.* (2018))

Participants	**Taux de classification (Partie A)**
Chennamsetty *et al.* (2018)	87 %
Kwok (2018)	87 %
Brancati *et al.* (2018)	86 %

Il faut noter que ce score est obtenu avec la base de test du concours. De ce fait, il ne pourra pas être directement comparé aux résultats obtenus lors des différentes expériences puisqu'il ne s'agit pas de la même base de test.

4.1.1 Étude des patches

4.1.1.1 Choix du critère

La première étape a été de tester les différentes combinaisons de critères et de méthodes de fusion des patches pour avoir une méthode performante commune aux différentes expériences. Le tableau 4.3 répertorie l'ensemble des résultats obtenus sur la base de test à partir des différentes stratégies avec 30 images par classe annotées. Pour les critères réduisant le nombre de patches, les bases de validations utilisées ont été affectées de la même réduction pour limiter l'influence de patches mal annotées.

Tableau 4.3 Scores d'exactitude (%) obtenus sur la base de test

Critères	Vote majoritaire	Top 1	Somme des scores	WILDCAT	SVM
Sans critère	83.80 $\pm$ 1.88	82.61 $\pm$ 1.88	83.92 $\pm$ 1.70	79.64 $\pm$ 2.91	83.80 $\pm$ 1.88
75% cells	84.76 $\pm$ 2.73	84.52 $\pm$ 1.53	84.52 $\pm$ 1.59	79.28 $\pm$ 2.96	84.04 $\pm$ 1.96
75% seuil	82.73 $\pm$ 1.45	82.73 $\pm$ 2.46	83.57 $\pm$ 1.87	79.52 $\pm$ 1.93	83.45 $\pm$ 1.03
Entropie	84.04 $\pm$ 1.43	83.33 $\pm$ 1.54	85.35 $\pm$ 1.46	82.97 $\pm$ 2.51	84.28 $\pm$ 0.82
Loss 0,85	85.47 $\pm$ 1.39	82.85 $\pm$ 2.94	85.0 $\pm$ 0.89	81.90 $\pm$ 3.92	85.11 $\pm$ 1.43
Loss 0,7	83.80 $\pm$ 3.38	80.23 $\pm$ 4.31	82.87 $\pm$ 4.58	78.21 $\pm$ 3.34	83.57 $\pm$ 3.82

La colonne de gauche répertorie l'ensemble des critères utilisés :

- 'Sans critères' : utilisation des 35 patches par image.
- '75% cells' : utilisation de 75% des patches ayant le plus de cellules.
- '75% seuil' : utilisation de 75% des patches ayant le moins de pixels d'arrière-plan.
- 'Entropie' : utilisation des patches correctement prédits par les modèles 'Sans critères'.
- 'Loss 0,85' : utilisation des 35 patches par image en modifiant la fonction de coût supervisée (entropie croisée) pour prédire à 85% la classe souhaitée et 5% les autres classes.
- 'Loss 0,7' : utilisation des 35 patches par image en modifiant la fonction de coût supervisée (entropie croisée) pour prédire à 70% la classe souhaitée et 10% les autre classes.

On remarque que l'utilisation de certains critères n'est pas toujours bénéfique. La faible variation de performances entre 'Sans critère' et '75% seuil' laisse à penser que le bruit des patches d'arrière-plan, qui n'est pas représentatif de la classe, est pris en compte lors de l'entraînement. On constate que retirer ces patches ne permet pas d'obtenir de meilleures performances. On peut supposer que les patches très similaires appartenant à des classes différentes vont se retrouver sur les bordures de prédiction entre les différentes classes tandis que les patches plus propres à une certaine classe seront mieux reconnus. De plus, l'augmentation des performances en utilisant le critère 'Entropie' nous conduit à penser que le modèle arrive, dans une certaine mesure, à prendre en compte le bruit dans l'annotation des patches. Mais on peut aussi en conclure que tous les patches ne sont pas représentatifs de la classe, puisqu'en enlever a tendance à accroître les performances. Pour la suite des expériences, nous avons retenu l'approche ayant obtenu le meilleur score en moyenne, soit l'utilisation du critère 'Loss 0.85', additionné au vote majoritaire comme méthode de fusion des patches. Pour l'entraînement des différents modèles, nous avons donc modifié légèrement les fonctions de coût supervisées.

4.1.1.2 Résultats des expériences

Dans un premier temps, nous avons effectué les expériences 1 et 2 en prenant comme architecture de base un ResNet-18, pré-entraîné sur ImageNet. Il convient de souligner que l'approche *Ladder Network* n'est pas présente sur les graphiques, car nous ne pouvons pas réutiliser les paramètres de pré-entraînement, puisque cette approche modifie l'architecture du modèle. Les résultats de la première expérience sont représentés sur la figure 4.1. Il faut noter que le point sans annotations correspond aux performances obtenues avec un ResNet-18 pré-entraîné en utilisant 10 images par classe annotée.

On constate dans l'ensemble que l'augmentation de données non annotées permet, dans le cas d'une faible proportion de données annotées, d'obtenir, en moyenne, de meilleures performances. Contrairement à nos attentes, cette augmentation n'est pas toujours proportionnelle au nombre de données non annotées et ne se manifeste pas de la même façon en fonction du modèle utilisé. De plus, l'importance des écarts types nous empêche de conclure sur l'efficacité

des modèles dans ce cas de figure. Les résultats de la deuxième expérience sont présents sur la figure 4.2.

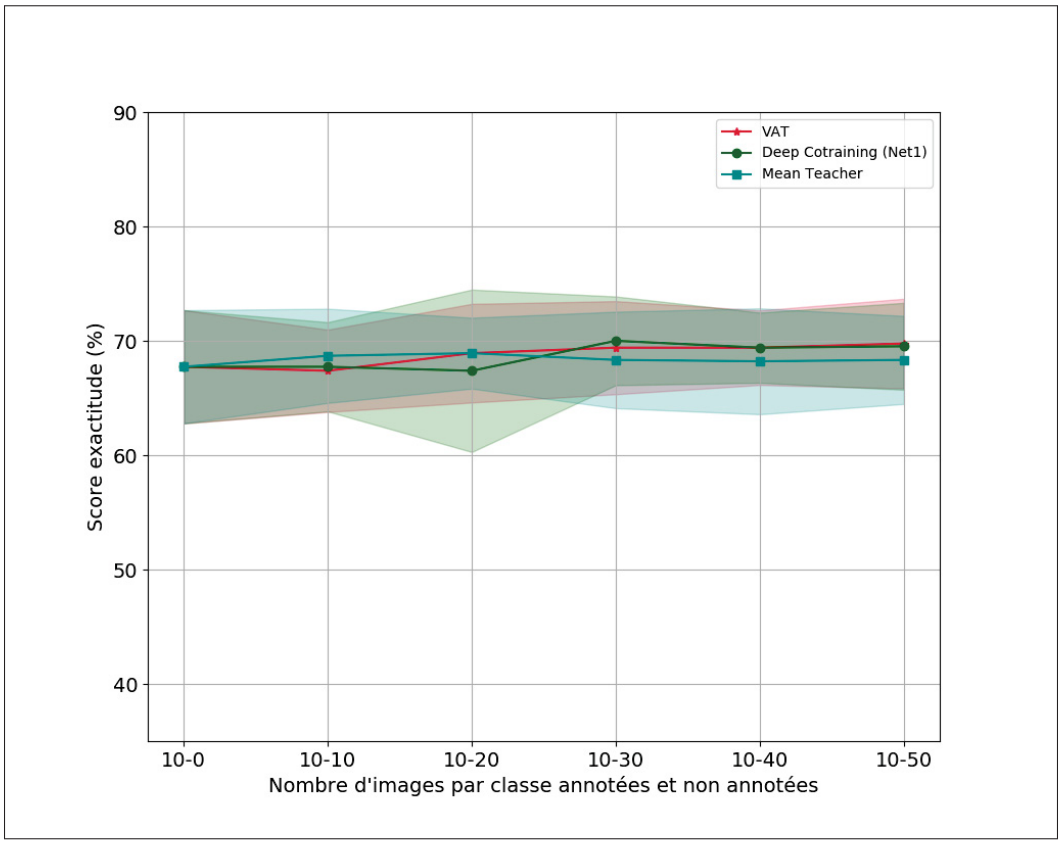


Figure 4.1 Expérience 1 avec pré-entraînement sur la base BACH (avec 35 patches par image)

Cette expérience vise à mettre en évidence l'apport simultané des données annotées et non annotées. On constate que dans le cas de l'utilisation de modèles pré-entraînés l'écart entre les différentes approches est faible. L'observation de ces différents résultats montre qu'il est difficile de conclure sur l'influence des données non annotées sur l'entraînement des réseaux en raison du chevauchement des écarts types.

En moyenne, l'approche *Deep Co-training* semble tirer parti des données non annotées. De plus, on constate que l'écart entre les différentes approches semble constant avec l'ajout de

données. À l'exception de l'approche VAT, l'écart type obtenu à partir des 7 itérations diminue avec l'augmentation du nombre de données.

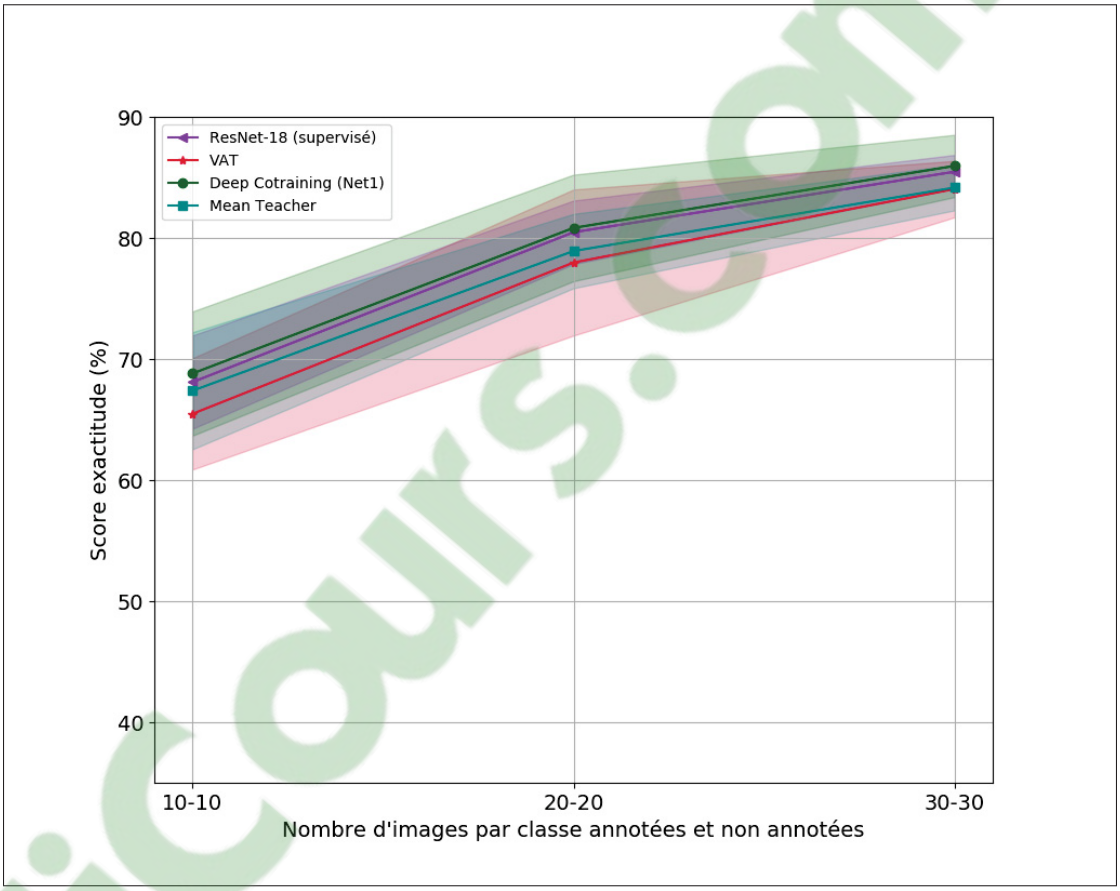


Figure 4.2 Expérience 2 avec pré-entrainement sur la base BACH (avec 35 patches par image)

Si l'on se penche sur l'entraînement des différents modèles, on constate que le choix des images a une influence importante sur les résultats, ce qui n'est pas le cas pour des bases plus importantes comme CIFAR-10. Les écarts types importants peuvent s'expliquer par le manque de données permettant de représenter correctement chaque classe. D'une itération à l'autre, ce n'est pas la même classe qui se retrouve moins bien représentée.

La figure 4.3 représente les matrices de confusion obtenues avec le ResNET-18 entraînées avec 30% de données annotées. On constate que les classes les mieux reconnues ne sont pas

les mêmes en fonction des images d'entraînement et de validation sélectionnées. De la même manière, les zones d'intérêt mises en avant diffèrent d'une approche à l'autre.

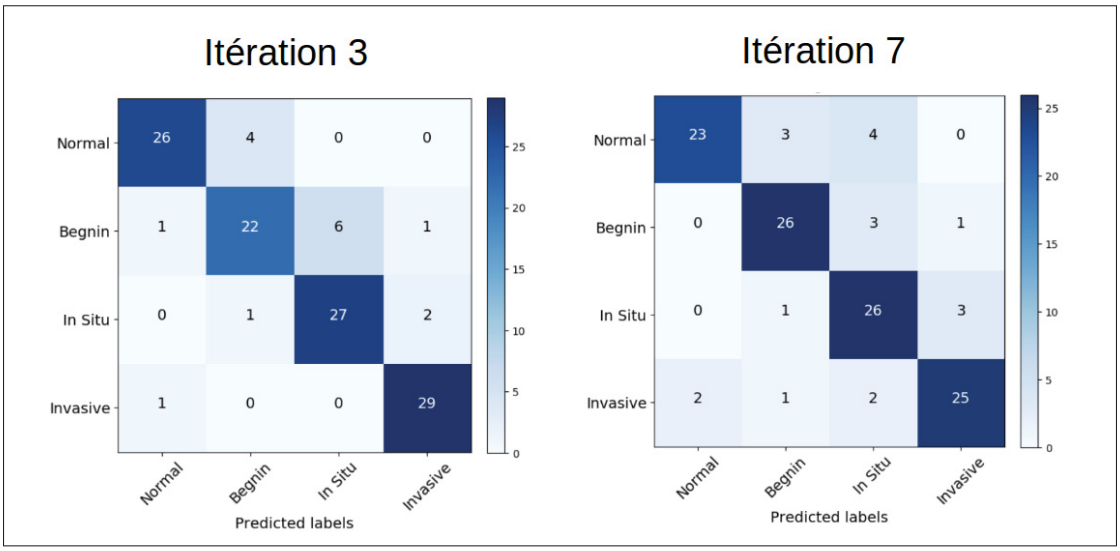


Figure 4.3 Matrices de confusion du ResNET18 pour 30% de données annotées

La figure 4.4 représente les prédictions obtenues sur les différents patches sur une image de la base de test. Pour obtenir cette représentation, nous avons séquencé l'image d'origine en 713 patches de 512 x 512 pixels redimensionnés en 256 x 256 pixels pour que ces derniers puissent être utilisés par les différents modèles.

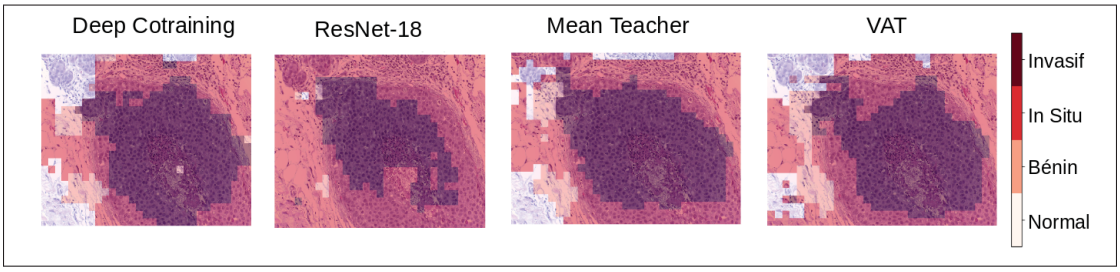


Figure 4.4 Prédictions des différentes classes (GT : In Situ)

Même si le label est correctement prédit, les zones mises en avant par les différentes approches ne sont pas exactement identiques. Les patches étant de taille importante, le décapage des régions n'est pas très précis. Pour ce cas, les différentes zones mises en évidence par le modèle

peuvent être interprétables. On s'attend, pour le label In Situ, à trouver une partie de tumeur emprisonnée par une paroi. Mais dans certains cas, les prédictions obtenues ne sont pas aussi claires (voir Fig. I-4).

Notre hypothèse est que l'utilisation d'un pré-entraînement limite l'apport de données sans annotations dans les résultats, le même phénomène ayant été observé sur d'autres bases de données (Oliver *et al.* (2018)).

Nous avons effectué les mêmes expériences sans pré-entraînement (initialisation aléatoire des paramètres) sur quelques une des approches. Le gros désavantage de cette technique est le temps d'entraînement beaucoup plus important et la baisse des performances.

Tableau 4.4 Architecture Conv-Large modifié

Couches du modèle	**Taille d'entrée**
3x3 Conv stride 2 Conv 96 LreLu + BN	256 x 256 x 3
3x3 Conv stride 2 Conv 96 LreLu + BN	128 x 128 x 96
3x3 Conv stride 1 Conv 96 LreLu + BN	64 x 64 x 96
2 x 2 maxpool stride 2	64 x 64 x 96
DropOut (0,5)	32 x 32 x 96
3x3 Conv stride 2 Conv 192 LreLu + BN	32 x 32 x 96
3x3 Conv stride 2 Conv 192 LreLu + BN	16 x 16 x 192
3x3 Conv stride 1 Conv 192 LreLu + BN	8 x 8 x 192
2 x 2 maxpool stride 2	8 x 8 x 192
DropOut (0,5)	4 x 4 x 192
3x3 Conv stride 1 Conv 192 LreLu + BN	4 x 4 x 192
3x3 Conv stride 1 Conv 192 LreLu + BN	4 x 4 x 192
3x3 Conv stride 1 Conv 192 LreLu + BN	4 x 4 x 192
Average Pooling	1 x 1 x 192
Fully connected	192 -> 4

Les modèles très profonds sont très performants si pré-entraîné, mais leur très grand nombre de paramètres ne les rend pas optimaux si on démarre de 'zéro'. Le ResNet-18 à été comparé à une architecture inspirée de Conv-Large dont l'architecture est décrite sur le tableau 4.4.

Pour la sélection du modèle sans pré-entraînement, nous avons retenu les mêmes données d'entraînement et de validation. Nous avons comparé l'aspect des courbes d'apprentissage et

les performances obtenues sur la base de validation pour ces deux architectures. Les courbes des différentes performances obtenues sont représentées sur les figures I-1 et I-2.

On constate que l'apprentissage avec le ResNet 18 est très instable : d'une epoch à l'autre, les performances varient de plusieurs pourcents. Pour le modèle Conv-Large modifié, plus petit, ces variations sont moins importantes et se manifestent surtout pendant la phase de sur apprentissage. Nous avons donc préféré choisir ce modèle (Tab. 4.4) pour la comparaison.

Tableau 4.5 Résultats sans pré-entraînement

Conv-Large	**VAT**	**Mean Teacher**
72.73 ± 4.60	75.11 ± 6.11	74.25 ± 4.61

Les résultats du tableau 4.5 ont été obtenus sans pré-entraînement dans le dernier cas de figure de l'expérience 2, c'est-à-dire avec 30 images par classe annotées et non annotées. Ces résultats représentent les scores d'exactitudes (%) obtenues sur la base de test en utilisant le modèle *Conv-Large modifié*. On remarque alors que l'ajout de données non annotées permet, dans ce cas de figure, d'améliorer sensiblement les performances obtenues. Une plus forte instabilité lors de l'entraînement provoque de plus forts écarts-types pour l'ensemble des modèles.

4.1.1.3 Limites de l'étude

Les caractéristiques de cette étude doivent être prises en compte dans les résultats obtenus. En effet, le choix des éléments retenus ont un effet sur les performances obtenues et limite la portée de la comparaison entre ces différentes approches.

Tout d'abord, le choix de la méthode de la fusion des patches a été déterminé à partir des performances du modèle ResNet-18 sur la base de test. C'est une des raisons qui pourrait expliquer le faible écart observé entre ce modèle et les modèles d'apprentissages semi-supervisés.

Ensuite, le choix des hyperparamètres a été effectué sur une itération, et conservé pour effectuer les expérimentations. Néanmoins, les écarts types importants observés nous indiquent que les images sélectionnées ne sont pas assez diversifiées pour représenter la complexité des différentes

classes. Une méthode plus longue, mais plus rigoureuse consisterait à paramétrer les différentes approches sur chaque nouveau jeu de données d'apprentissage et de validation.

Tableau 4.6 Scores d'exactitude obtenus avec 30 images par classe annotées et non annotées (avec pré-entrainement)

Base utilisée	**ResNet-18**	**VAT**	**Mean Teacher**	**Deep Cotraining**
Base de validation (score par patch)	76.40 ± 1.74	76.05 ± 1.40	78.25 ± 1.29	77.98 ± 1.00
Base de test (score par image)	85.47 ± 1.39	84.04 ± 2.33	84.16 ± 1.88	84.16 ± 1.88

Enfin, la manière dont est utilisée la base de données n'est pas adaptée à la comparaison d'approches. Il est normal de comparer les modèles sur les performances obtenues sur les images complètes, puisque seul leur label est à notre disposition. Néanmoins, afin d'obtenir de meilleures performances, l'entraînement des différents modèles est effectué sur la reconnaissance de patches, dont le label est bruité. Mais l'obtention d'un meilleur score de reconnaissance de patches n'implique pas d'avoir de meilleures performances sur la reconnaissance des images (dans une certaine mesure), comme nous le montre le tableau 4.6.

Ce tableau représente les scores d'exactitudes obtenus sur la reconnaissance des patches de la base de validation (en haut) et reconnaissance d'images complètes sur la base de test (en bas), en attribuant aux 35 patches de chaque image le label de l'image complète. Lors de l'optimisation des approches, nous avons cherché à obtenir le meilleur score d'exactitude par patch sur la base de validation. Toutefois, le tableau 4.6 nous montre que les performances entre le score par patch et le score de l'image complète ne sont pas directement reliées. Dans une certaine mesure, l'augmentation du score de reconnaissance des patches ne va pas se retrouver sur le score obtenu sur les images. En termes d'entraînement, les modèles *Mean teacher* et *Deep Co-training* obtiennent des performances nettement supérieures à celles obtenues avec le ResNet-18 sur la reconnaissance de patches dont le label est incertain, mais cet avantage ne se vérifie pas lorsque l'on compare les résultats sur la base de test.

Dans un premier temps, nous nous étions limités à l'étude de cette base de données par l'apprentissage des patches, permettant d'obtenir de meilleures performances. Mais, cette démarche biaise les résultats, car le score obtenu en test n'a de véracité que si l'on se restreint aux an-

notations globales. Afin de pallier l'ajout de bruit dans l'annotation des patches, nous avons décidé de focaliser la suite de notre étude sur l'apprentissage des images complètes, ce qui aura pour effet de diminuer légèrement nos performances et de diviser par 35 la taille de la base de données. Ensuite, nous avons augmenté la proportion de la base de validation afin qu'elle représente mieux la base de test. Pour cela, nous sommes passés à un découpage de 60% des données pour l'entraînement 20 % pour la validation et 20 % pour le test avec 4 bases indépendantes de validation. Avec cette nouvelle répartition, nous ne serons plus en mesure de comparer les performances obtenues sur la base de test entre l'approche par patch et l'approche par image complète.

Pour la recherche d'hyperparamètres, nous nous sommes inspirés de différentes recherches. Une étude similaire à l'expérience 1 (Oliver *et al.* (2018)) a été menée afin de comparer les différents modèles profonds sur les bases de références comme CIFAR-10. Pour effectuer la comparaison, les auteurs ont fixé les taux de normalisations pour l'ensemble des modèles. À partir d'un essai, ils ont fixé le taux d'apprentissage et les hyperparamètres spécifiques à chaque modèle sur l'ensemble des expériences. Si on se réfère aux différents articles présentant les architectures, les hyperparamètres donnés sont communs à l'ensemble des essais.

Dans le cas de l'expérience 3, sur le ResNet-18, nous avons effectué une recherche plus approfondie des hyperparamètres, en traitant indépendamment chaque essai (voir Annexe II). Il s'avère qu'une telle optimisation permet d'obtenir des performances plus élevées sur la base de validation, néanmoins ces hausses ne sont pas généralisables à la base de test. On peut supposer que dans notre configuration, la base de validation, bien qu'étant de taille similaire à la base de test, ne soit pas représentative.

Nous nous sommes donc limité à utiliser des hyperparamètres communs aux différents essais, une nouvelle recherche étant effectuée en fonction du nombre de données annotées étudiées. Nous avons remarqué qu'en général un des tirages de données est particulièrement difficile à étudier et fournit des performances les plus basses sur sa base de validation. L'optimisation de la moyenne des performances étant majoritaires due aux performances obtenues dans ce

cas particulier, nous avons décidé de choisir, pour chaque base de données, le tirage le moins performant pour effectuer notre recherche d'hyperparamètres. La prochaine partie des travaux va consister à effectuer les différentes expériences sur la base BACH en se basant sur les images complètes afin de ne plus prendre en compte l'influence du choix du critère et de la méthode de fusion des patches.

4.1.2 Étude des images complètes

Dans un premier temps nous avons effectué les expériences sur un ResNet-18 pré-entraîné sur ImageNet afin d'imiter les modèles du concours. La figure 4.5 représente les résultats obtenus sur l'expérience 1. Le point 10-0 correspond à l'utilisation du ResNet-18 uniquement.

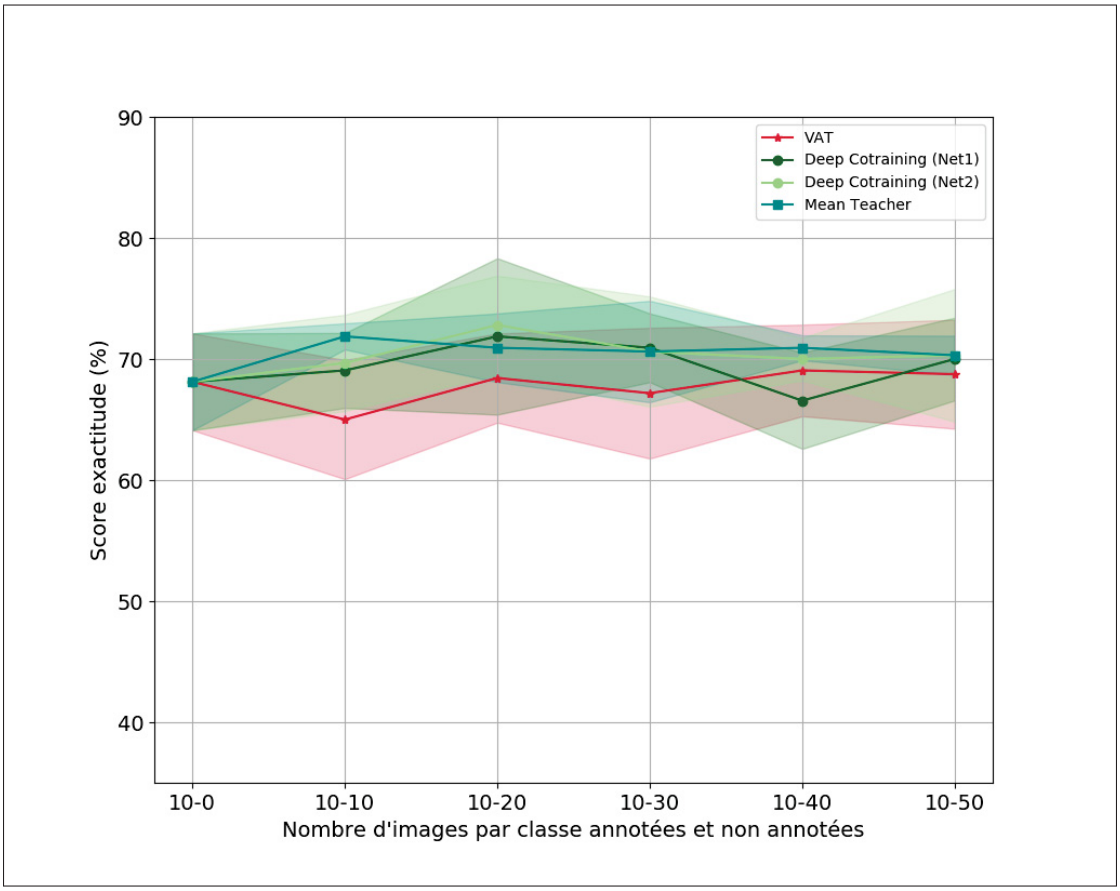


Figure 4.5 Expérience 1 avec pré-entraînement, base BACH

De manière générale, on constate que l'apport des données non annotées est particulièrement remarquable pour l'ajout respectivement de 10 et 20 images par classe pour les approches *Mean teacher* et *Deep Co-training*.

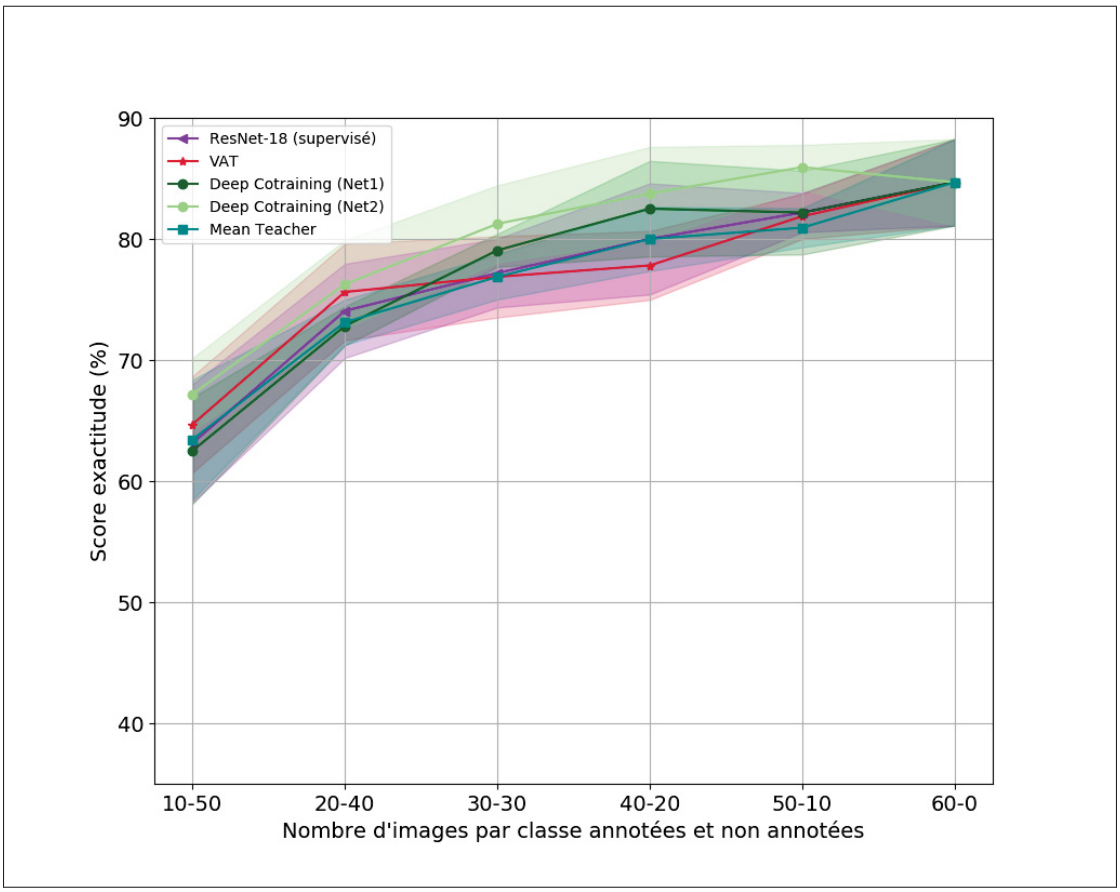


Figure 4.6 Expérience 3 avec pré-entraînement, base BACH

En revanche, ajouter plus d'exemples diminue les performances de ces approches. On peut supposer qu'à partir d'un certain seuil, la régularisation apportée par les données non annotées supplémentaires ne permet pas d'augmenter la représentation des différentes classes par le modèle. Pour le cas du VAT, on observe une légère augmentation de performances à partir de l'ajout de 40 images par classe annotées. La baisse de performances que l'on observe pour le cas 10-10 est due à la différence de représentation entre les bases de validation et de test mentionnées précedemment. Cette chute n'est pas présente dans le tableau II-6 qui répertorie les performances obtenues par les différents modèles sur la base de validation.

En ce qui concerne l'expérience 3, qui consiste à augmenter graduellement le nombre de données annotées, on constate sur la figure 4.6 que globalement, comme on pouvait s'y attendre, augmenter le nombre de données annotées augmente les performances des différents modèles. Il convient de noter que le cas 60-0 correspond à l'utilisation d'un ResNet-18 entièrement supervisé. Le deuxième modèle de l'approche *Deep Co-training* semble particulièrement performant dans cette configuration. L'approche *Mean teacher* dépasse les performances obtenues par l'approche entièrement supervisée à partir de 30 images par classe annotées et 30 images non annotées.

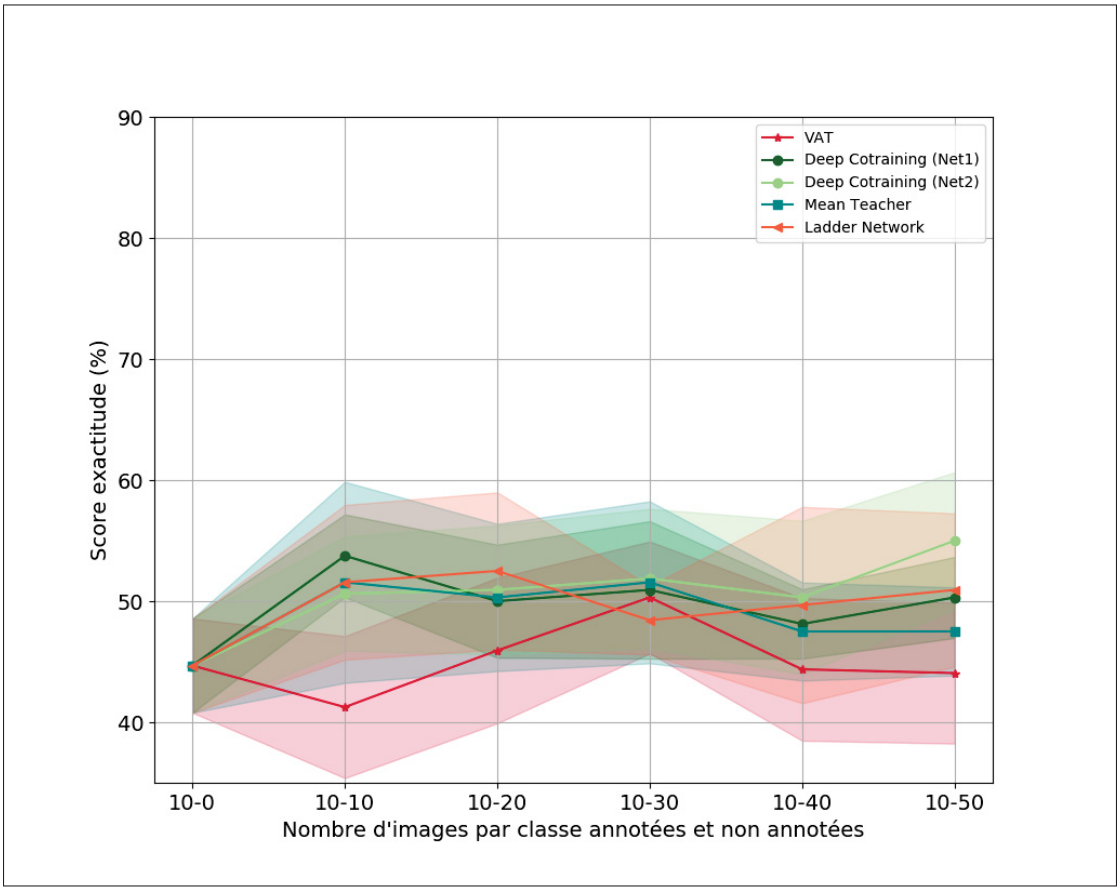


Figure 4.7 Expérience 1 sans pré-entraînement, base BACH

Comme on a pu le constater sur l'expérience 1, il semblerait que cette approche soit particulièrement performante sur cette base de données lorsque l'on utilise autant de données annotées que non annotées. L'approche VAT dépasse le score d'exactitude de l'approche supervisée

lorsque le nombre de données non annotées est important, dans le cas 10-10 et 20-20. Au-delà, on constate que les performances sont en dessous de celles obtenues avec le ResNet-18.

Dans un deuxième temps, nous avons analysé les comportements de ces différents modèles en nous basant sur un modèle non supervisé. Les expériences avec l'utilisation de patches nous ont montré des écarts plus importants entre les différents modèles lorsque l'on ne pré-entraîne pas les modèles d'origine. La courbe 4.7 représente les scores d'exactitudes des différents modèles obtenues en augmentant graduellement le nombre de données non annotées.

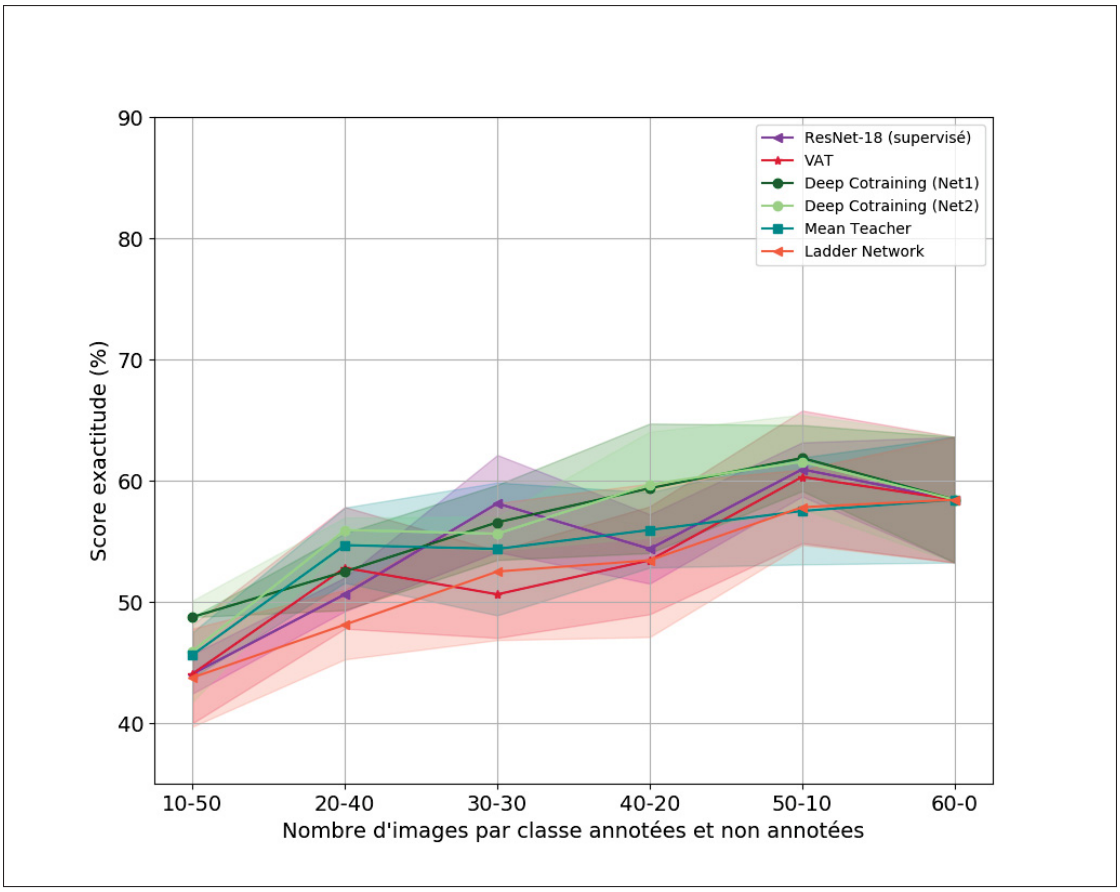


Figure 4.8 Expérience 3 sans pré entraînement, base BACH

On constate que l'écart de performances maximal entre le cas entièrement supervisé et non supervisé a doublé passant d'environ 3% avec pré-entraînement à plus de 6%. Cependant, on observe, dans le même temps, une diminution des performances moyennes aux alentours

de 20 images par classe non annotées. Contrairement au cas précédent, les deux modèles de l'approche *Deep Co-training* ne se comportent pas de la même façon. On peut remarquer que le modèle 2 tire un avantage à utiliser un grand nombre de données non annotées contrairement au modèle 1. Nous allons nous intéresser plus en détail aux comportements de ces deux modèles dans la partie 4.4. Hormis l'augmentation de la différence de performances, le modèle *Mean Teacher* semble toujours tirer profit d'un faible nombre de données non annotées, ce qui est aussi le cas du *Ladder Network*. L'approche VAT reste moins performante dans ce cas que les autres approches, et ne semble pas tirer parti de l'ajout de données à partir un certain seuil. Pour l'expérience 3, la courbe 4.8 nous montre une baisse de performances moyennes de la part de ResNet-18 avec 40 et 60 images par classe annotées. Dans ce cas de figure, l'approche *Deep Co-training* est en moyenne plus performante que l'approche entièrement supervisée à l'exception du cas utilisant 30 images par classe annotées.

Si on compare les résultats obtenus entre les deux expériences pour le cas 10-50, on constate que les écarts types sont bien moins importants pour l'expérience 3. Or, seul le choix des données d'entraînement change. Comme nous l'avons constaté précédemment, le choix des données est ici particulièrement important, contrairement aux bases de références. Le faible nombre de données retenu pour représenter le problème entraîne des bases de validations et de test de petite taille. Cela a deux conséquences dans notre cas. La première : les bases de validations ne sont pas représentatives de la base de test. La deuxième : on obtient de plus grands écarts types ; en effet, si un modèle se trompe d'une image, il aura plus de 1% d'erreur supplémentaire puisque la base de test n'est représentée que par 80 images. En ce qui concerne l'optimisation, comme mentionné dans la partie 4.1.1.3, un choix des paramètres au cas par cas ne représente pas toujours une bonne solution. Pour la suite du projet, nous nous sommes intéressés à une base de données plus importante, issue de la compétition TUPAC, dont le but est de dénombrer les cellules en cours de mitose sur des images histologiques.

4.2 Deuxième partie des expériences : la base TUPAC

Le tableau 4.7 représente les scores F1 obtenu par les meilleurs participants lors de la cette compétition.

Tableau 4.7 Scores F1 obtenus sur la base de test de TUPAC
(Partie détection automatique de cellules en mitose)

Participants	**Score F1 (détection de mitose)**
Paeng *et al.* (2017)	65.2 %
Zerhouni *et al.* (2017)	64.8 %
Rousson *et al.* (2018)	61.6 %

Ce tableau est donné à titre d'exemple, bien que le base de test utilisée lors du concours ne corresponde pas à celle utilisée pour les différentes expériences.

4.2.1 Résultats des expériences 1 et 3

Dans un premier temps, nous avons entraîné un modèle *WResDrop* (architecture (b) Fig. 1.7) pour effectuer l'expérience 1 avec les différentes méthodes semi-supervisées. La figure 4.9 représente les scores F1 obtenus en augmentant progressivement le nombre de données non annotées. Il convient de noter qu'une partie des patches négatifs des données annotées et non annotées a été sélectionnée par le modèle 'oracle' afin de focaliser l'apprentissage sur les cas particulièrement difficiles.

On constate que l'approche VAT est particulièrement performante dans ce cas de figure. Une faible quantité de données non annotées permet d'obtenir des performances significativement supérieures au cas entièrement supervisé. Cette approche ne semble pas tirer profit d'un plus grand nombre d'exemples supplémentaires, on constate une stagnation moyenne du score F1. Pour les approches *Deep Co-training* et *Mean teacher*, on constate une augmentation progressive moyenne des performances avec l'augmentation des images.

On peut supposer qu'ajouter des données supplémentaires pourrait permettre d'augmenter un peu plus leurs performances. Le modèle 2 du *Deep Co-training* est en moyenne toujours plus

performante que le modèle 1. Le *Ladder Network* ne semble pas tirer parti de l'augmentation du nombre de données non annotées. Cela peut être dû à plusieurs facteurs comme le grand nombre de couches ajoutant du bruit lors de l'apprentissage ou encore l'optimisation de cette approche (quantité importante d'hyperparamètres).

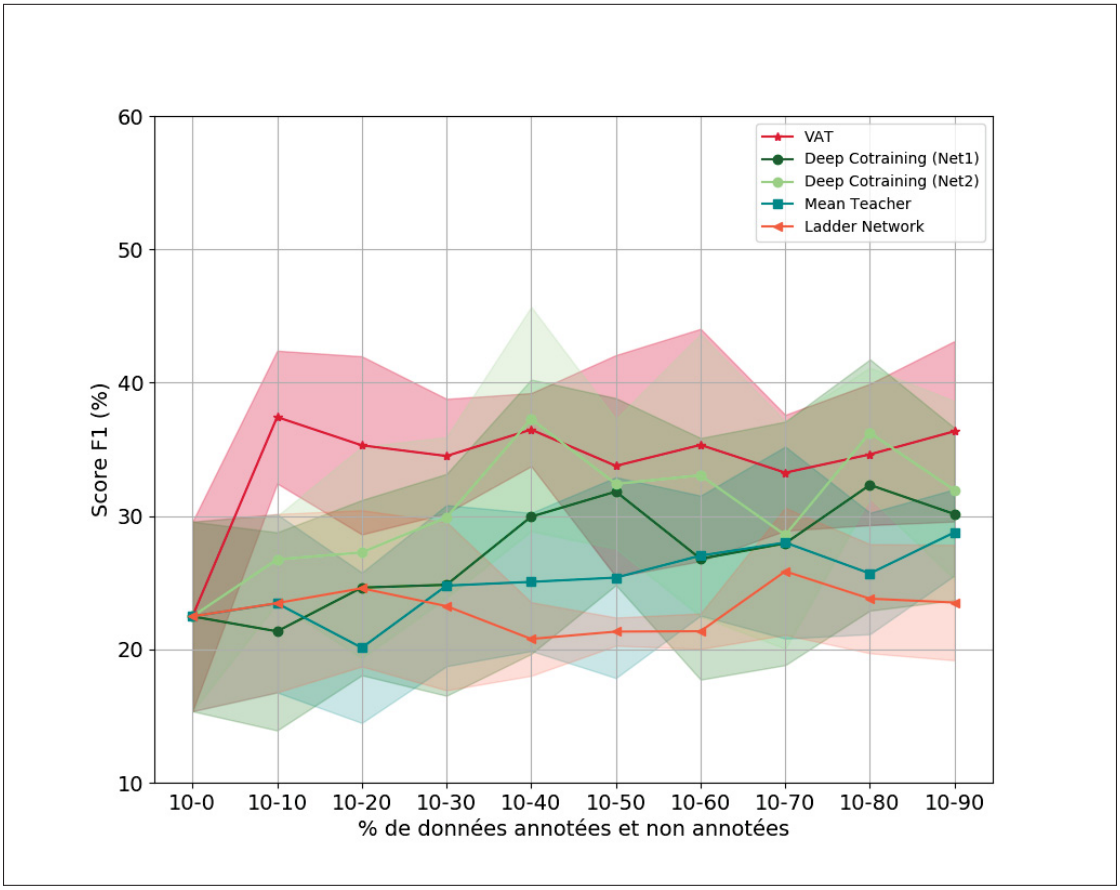


Figure 4.9 Expérience 1 sans pré-entraînement, base TUPAC

La figure 4.10 représente les scores F1 obtenues par les différents modèles pour l'expérience 3. On constate que l'augmentation des performances est particulièrement importante lorsque l'on a peu de données annotées et un grand nombre de données non annotées. En effet, on constate une augmentation moyenne de 15% dans le cas 10-90 pour l'approche VAT, et de 14% pour le Net2 de l'approche *Deep Co-training*. Pour l'approche *Mean teacher* l'apport est plutôt constant lorsque l'on augmente le nombre de données annotées au profit des exemples sans

annotations. De manière générale, on constate que l'apport des données sans annotations se réduit avec l'augmentation des données annotées.

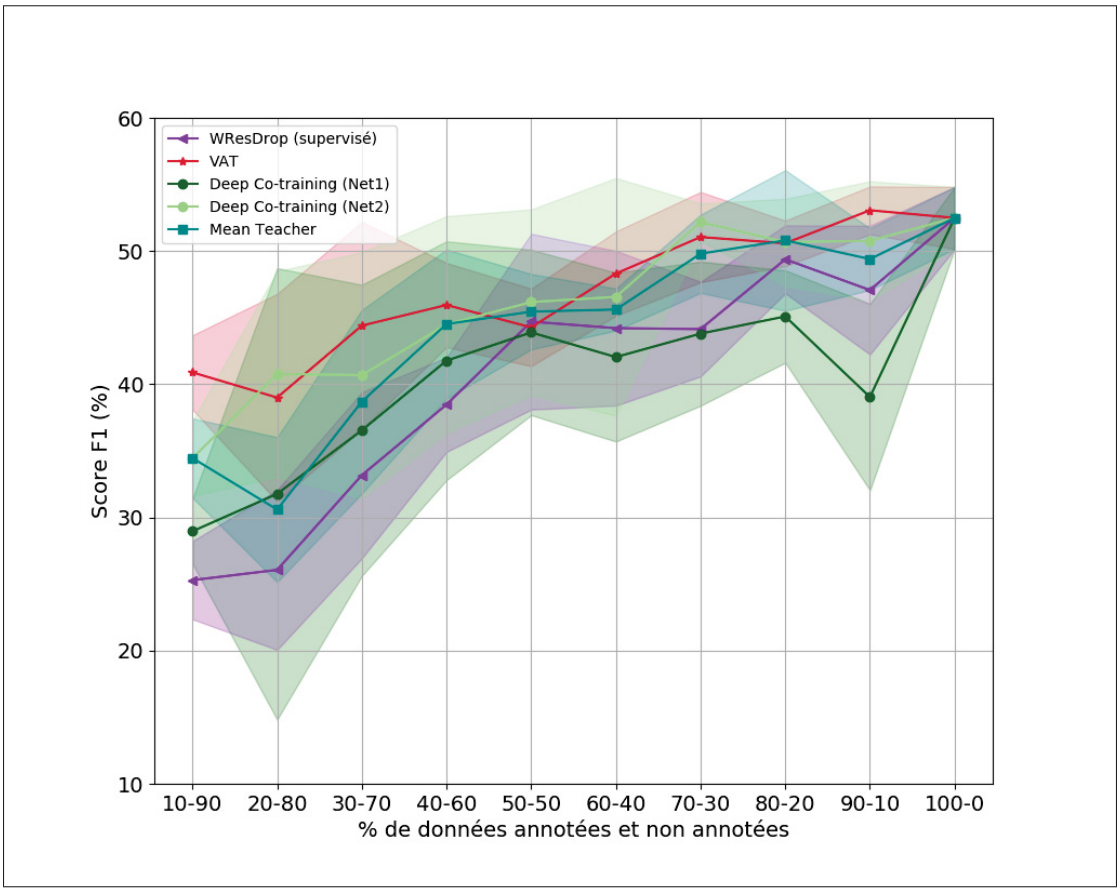


Figure 4.10 Expérience 3 sans pré-entraînement, base TUPAC

Pour conclure sur l'apport du pré-entrainement, l'expérience 3 a été effectué en utilisant un ResNet-18 pré-entrainé. La figure 4.11 répertorie les performances obtenues par les différents modèles dans ce cas de figure. On constate de manière générale que les performances ne sont pas particulièrement supérieures dans ce cas. On retrouve aussi l'approche VAT comme étant en moyenne la plus performante dans ce cas de figure. Contrairement à la base BACH où le pré-entrainement avait un impact très important sur les performances, pour cette base de données, l'utilisation d'un plus petit modèle, qui n'est pas pré-entrainé, semble être plus avantageux. Cela est probablement lié à lacompléxité des problèmes et au nombre d'exemple à notre disposition. Dans le cas de la base BACH, il s'agit de classer des images de grande taille très hétérogènes

avec peu d'images annotées. Dans le cas de la base TUPAC, les patches sont de petites tailles et représentent toujours des cellules. De plus, la quantité d'exemple est plus importante.

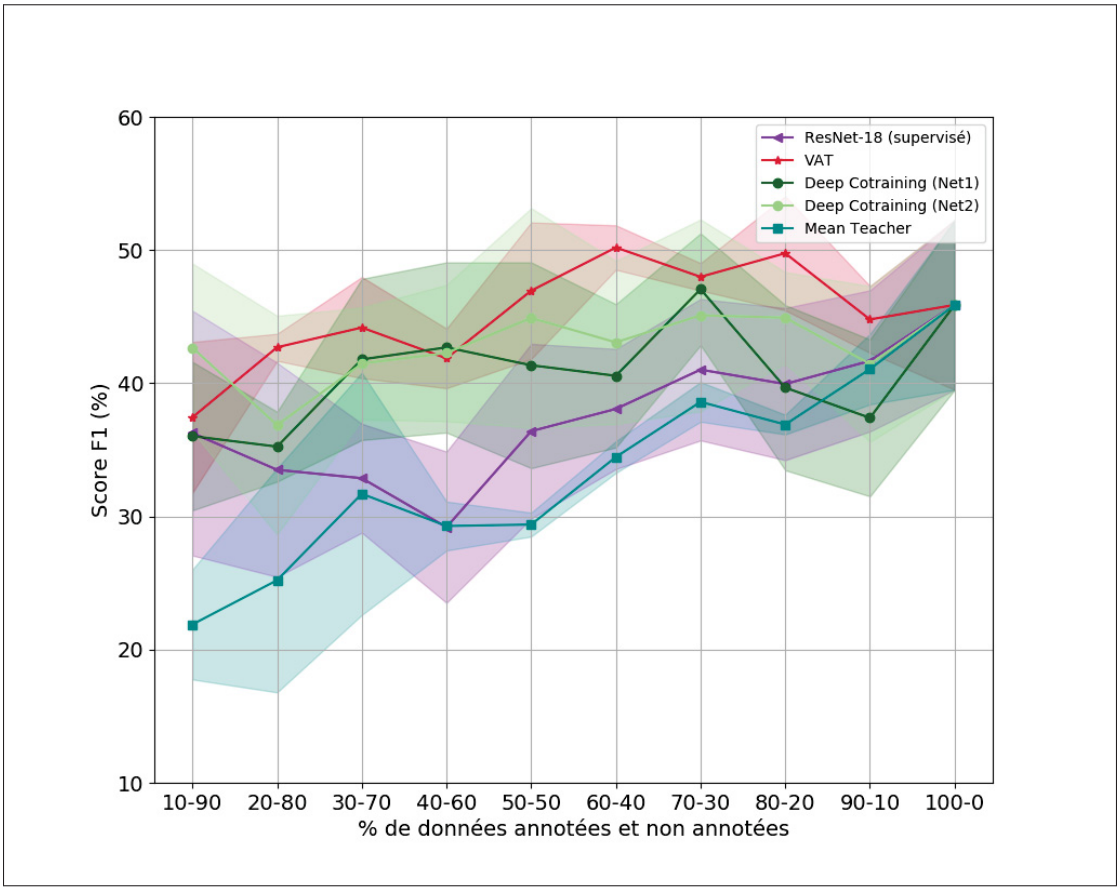


Figure 4.11 Expérience 3 avec pré-entraînement, base TUPAC

4.2.2 Limites de l'étude

Parmi les limites de cette comparaison, on peut noter l'effet du modèle 'oracle' qui a permis d'effectuer la sélection des patches négatifs particulièrement difficile. Ce modèle a été entraîné avec l'ensemble des données de mitoses et utilisé pour la sélection des patches annotées et non annotées. Dans un cas réel, si l'on se place dans le cas de 10% de données annotées et 90% de données non annotées, le modèle 'oracle' ne pourra être entraîné qu'avec une toute petite portion de données annotées, ce qui pourra limiter sa capacité à prédire les patches négatifs particulièrement difficile à reconnaître. En ce sens, les performances obtenues lors des

différentes expériences ne représentent pas exactement le cas ’réel’ où seule une toute petite portion de la base est accessible. Cette remarque est aussi valable dans le découpage de la base globale en base de validation et en base de test. L’objectif de cette étude à été de produire un socle commun aux différentes approches pour les comparer dans la limite du temps imparti.

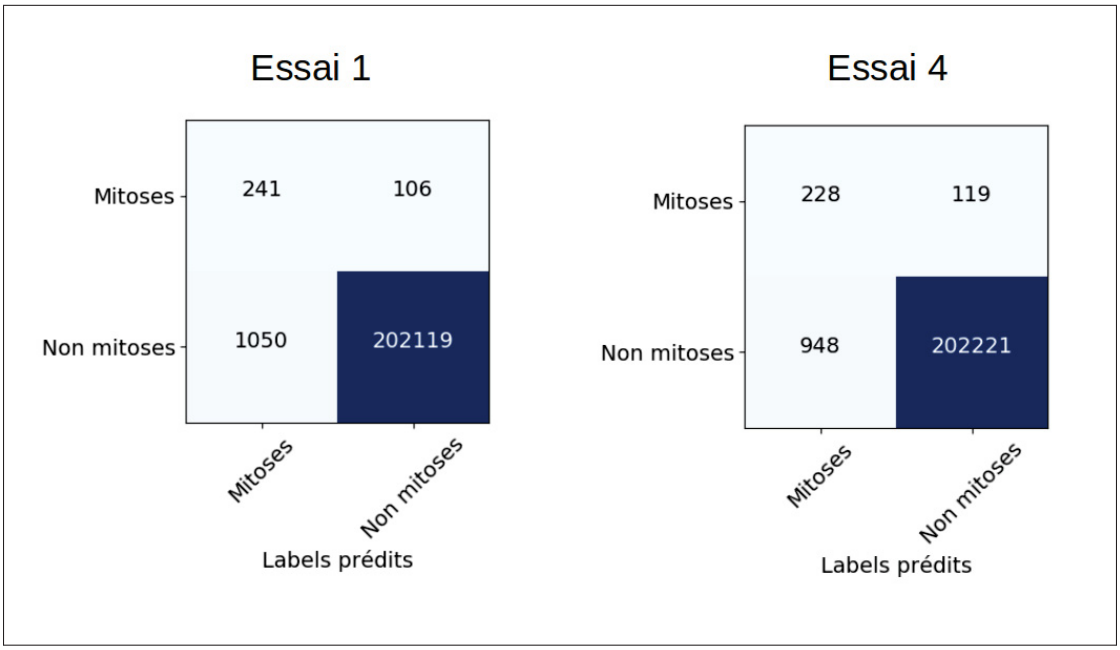


Figure 4.12 Exemple de matrice de confusion obtenue avec un *WResDrop* avec 20-0

Une autre limite provient de l’utilisation du score F1 pour comparer les différentes approches. Dans la compétition, seul ce score est pris en compte. Or, dans le cas déséquilibré, le poids du score de précision et de rappel ne sont pas les mêmes. Par exemple, si l’on compare deux modèles ayant obtenu un score F1 relativement proche (voir Tab. 4.8), on constate une différence importante dans les proportions de faux positifs. La figure 4.12 représente deux matrices de confusion obtenues avec un *WResDrop* en utilisant 20% de la base annotée. Une augmentation de 10% du nombre de faux positifs peut être équilibrée par une baisse de 10% du nombre de faux négatifs pour que le score F1 reste le même. Néanmoins, compte tenu du caractère déséquilibré de cette base, cela ne représente pas le même nombre de patches mal classés par le modèle. Pour le cas de la figure 4.12, un patch faux négatif représente environ 10 patches faux

positifs vis-à-vis du score F1. On retrouve ces écarts dans les scores de rappel et de précision respectifs. Le tableau 4.8 recense les différentes performances obtenues dans ce cas de figure.

Tableau 4.8 Mesures de test avec *WResDrop* pour 20-0

Mesures	**Essai 1**	**Essai 4**
Score F1 (%)	29.42	29.22
Précision (%)	18.66	18.88
Rappel (%)	69.45	64.55

Enfin, comme dans le cas précédent, la petite taille de la base de données (exemples positifs) rend l'interprétation des résultats particulièrement difficile. Les figures II-1 et II-2 représentent respectivement les performances obtenues sans et avec pré-entraînement sur la base de validation. La différence est particulièrement notable dans le cas avec pré-entrainement : l'approche *Mean teacher* se trouve être la plus performante lorsque l'on augmente le nombre de données annotées, mais sur la base test, elle est l'approche la moins adéquate.

4.3 Bilan sur les approches

Au termes de ces travaux sur ces différentes approches, on constate que le *Ladder Network* est l'approche la moins versatile. Elle nécessite de construire un auto-encodeur bruité, ce qui peut devenir problématique pour des réseaux très profonds. De ce fait, l'utilisation d'un pré-entraînement sur de grandes bases de données nécessite de le faire 'à la main'. De plus, son nombre d'hyperparamètre augmente avec le nombre de couches, et la sauvegarde des couches intermédiaires limite l'utilisation de grands patches et de grands batchs. D'un autre côté, ce modèle a fait ses preuves sur la base de données MNIST (Rasmus *et al.* (2015)) en utilisant des perceptions à couches multiples. C'est une approche qui peut être performante dans des problèmes qui nécessitent des architectures avec peu de paramètres et dont le pré-entraînement n'est pas nécessaire. Le *Mean teacher*, le VAT et le *Deep Co-training* sont des approches versatiles, qui peuvent facilement s'adapter à différentes tailles d'architectures. La différence entre ces méthodes provient de la manière dont les coûts non supervisés entre les prédictions

d'un ou de plusieurs modèles sont calculés. L'approche *Mean teacher* consiste à obtenir un modèle paramétré par la moyenne exponentielle d'un modèle étudiant. La fonction de coût de cohérence des prédictions est calculée à l'aide de l'erreur quadratique moyenne entre le professeur et l'étudiant, de sorte que les prédictions du professeur convergent vers celle de l'étudiant. Pour le VAT, ce coût non supervisé est calculé à l'aide de la divergence de Kullback Leibler, sur le même modèle, entre une image avec et sans bruit adversaire ; de sorte que les prédictions des images adversaires convergent vers celles sans bruit. La fonction de coût proposée par l'approche *Deep Co-training* est plus ambiguë. Dans un premier temps on va chercher à entraîner deux modèles résistant aux exemples adversaires de l'autre. La deuxième partie consiste à effectuer la moyenne des prédictions de ces deux modèles sur les données sans annotations en cherchant à minimiser la divergence de Jensen-Shannon.

Tableau 4.9 Mémoire et temps d'apprentissage pour la base de données TUPAC (sans pré-entraînement)

Approche	**Mémoire GPU**	**Temps (1000 itérations)**
WresDrop	1.7 GB	20 minutes
Mean teacher	6.2 GB	40 minutes
Virtual Adversarial Training	6.6 GB	80 minutes
Ladder Network	10.2 GB	175 minutes
Deep Co-training	8.6 GB	130 minutes

Le tableau 4.9 répertorie la quantité de mémoire utilisée et le temps nécessaire pour effectuer 1000 itérations avec les différentes approches. Cette comparaison a été effectuée sur la même carte graphique (Nvidia GEFORCE GTX 1080 Ti). Les approches VAT et *Mean teacher* sont les moins consommatrices de ressources. La différence du temps d'apprentissage entre ces deux modèles vient de la génération du bruit adversaire. L'approche *Ladder Network* nécessite le plus de mémoire, cela est dû aux variables latentes qu'il est nécessaire de stocker en mémoire à chaque itération. Dans la prochaine partie, nous allons étudier l'impact du nombre de modèles sur l'approche *Deep Co-Training*.

4.4 Le cas du Deep Co-Training

4.4.1 Mesure de diversité

Cette approche est particulièrement intéressante pour son utilisation des données non annotées. Le but est de garder un équilibre entre les deux modèles pour qu'ils aient des prédictions différentes sur les exemples adversaires du modèle opposé sur les données annotées, mais des prédictions similaires sur les données non annotées. On cherche donc à créer de la diversité entre ces deux modèles tout en conservant un socle de prédiction commun.

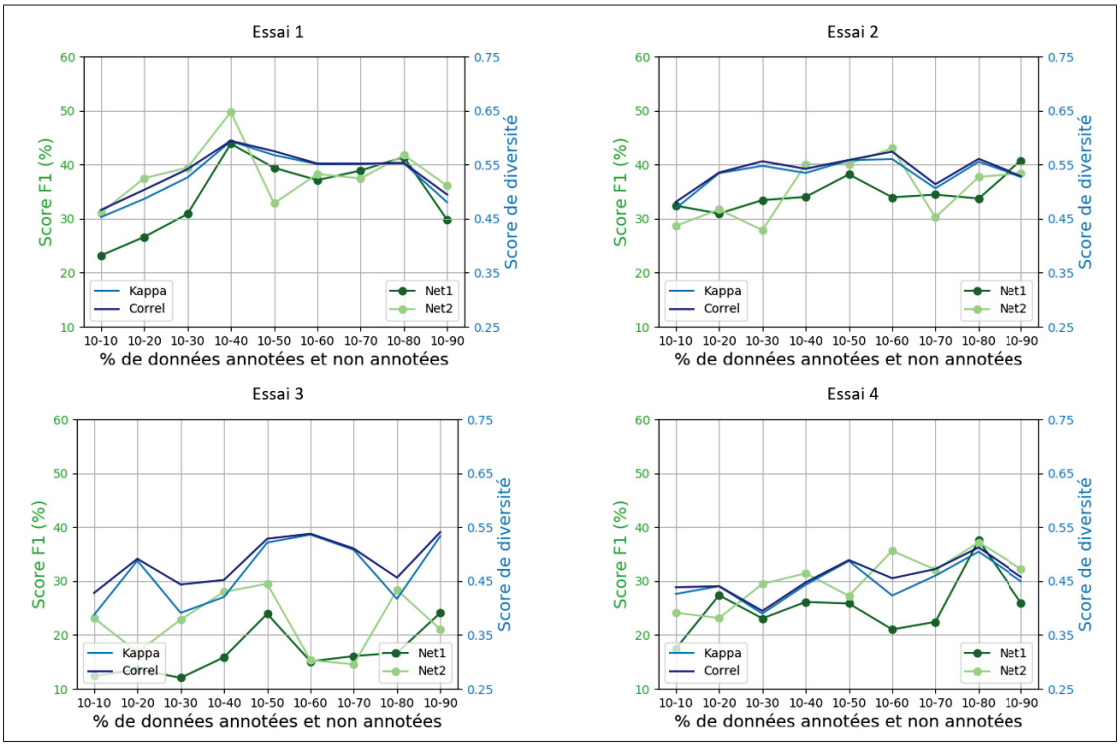


Figure 4.13 Mesures de diversité obtenues sur la base de test des deux modèles de l'approche Deep Cotraining, expérience 1

Les figures 4.13 et 4.14 représentent les scores de diversité obtenus pour différentes configurations sur l'expérience 1 et 3 de la base TUPAC. Ces résultats sont à prendre en compte avec les scores F1 obtenus sur l'ensemble des essais. On constate que les deux scores de diversité témoignent du même phénomène et qu'ils ont tendance à suivre les fluctuations des scores

F1. Bien que les résultats obtenus entre les essais soient très différents on peut remarquer que l'écart entre les deux modèles a tendance à diminuer avec l'augmentation du nombre de données annotées, ainsi qu'avec l'augmentation du score F1. On peut aussi remarquer, si l'on compare les différents essais, que ce ne sont pas nécessairement lorsque l'on obtient les meilleures performances que le score de diversité est le plus élevé.

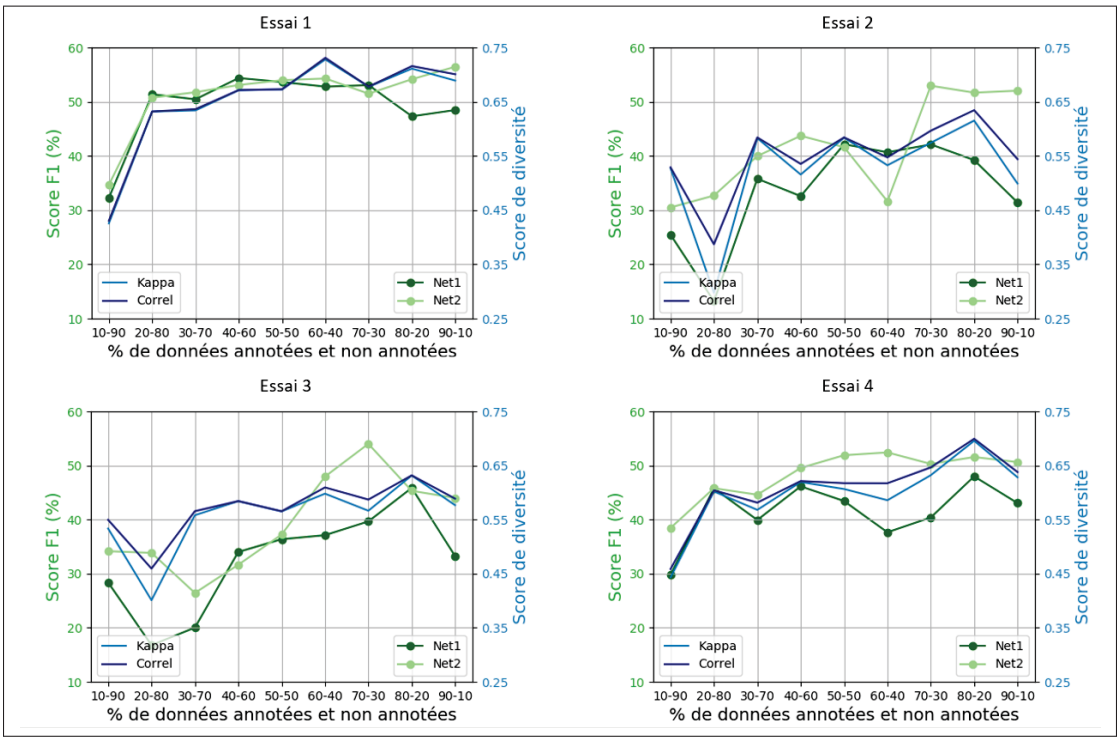


Figure 4.14 Mesure de diversité obtenue sur la base de test des deux modèles de l'approche Deep Cotraining, expérience 3

Cela nous indique que la fonction adversaire apporte bien de la diversité entre les deux modèles, mais que d'autres moyens sont possibles pour créer cette différence. Ainsi, d'autres aspects intéressants de cette méthode restent à explorer. Jusqu'à présent, pour effectuer une comparaison équitable entre les différents modèles, nous nous étions limités à présenter les prédictions des deux modèles séparément afin qu'un seul modèle soit utilisé pour chaque approche. Cependant, cela limite l'apport du second, qui, s'il est suffisamment différent du premier, permettrait par diverses méthodes de vote d'obtenir des performances supérieures à l'utilisation d'un unique modèle.

Il convient de noter que cet apport n'est bénéfique que si les prédictions des deux modèles sont différentes sur les mêmes patches. Dans ce sens, plus les performances seront importantes, moins les modèles seront différents. On doit s'attendre à ce que ce genre de méthode soit moins efficace sur des bases de données où le score de classification est élevé.

Pour allez plus loin dans l'étude de cette approche, nous avons testé comment l'ajout de modèles et la multiplication de sources de diversité joue un rôle dans les performances obtenues.

4.4.2 Influence du nombre de modèles et de l'ajout de diversité

Dans un premier temps, nous nous sommes intéressés à l'ajout de modèles sur les performances. Pour y parvenir, nous avons comparé les scores F1 obtenus en faisant la moyenne des prédictions des différents modèles. Pour l'entraînement, les approches seront utilisées par paire, comme indiqué dans l'article d'origine (Qiao *et al.* (2018)). Le tableau 4.10 répertorie ces performances dans le cas où les bases de données utilisées sont les mêmes. Dans ce cas de figure, seuls vont jouer dans la diversité l'ordre de présentation des patches, le choix des modèles pour l'entraînement par couple, et l'influence des couches de *dropout*.

Tableau 4.10 Influence du nombre de réseaux sur les performances. Les scores F1 (%) ont été obtenus sur la base de test en utilisant la même base d'entraînement (équilibrée)

Essai	**2 réseaux (NN)**	**3 réseaux (NNN)**	**4 réseaux (NNNN)**
Essai 1	42.39	48.96	49.72
Essai 2	31.87	30.88	43.80
Essai 3	36.70	40.12	42.37
Essai 4	39.77	45.00	42.22
Moyenne	37.68 ± 3.91	41.24 ± 6.75	44.52 ± 3.06

On constate qu'en moyenne, plus on utilise de modèles, meilleures sont les performances. Cet apport n'est pas toujours linéaire, comme on peut le voir pour l'essai 2. En raison du temps d'entraînement disponible, les différentes expériences ont été effectuées pour le cas où 10% des patches sont annotés et 90% sont utilisés comme sans annotations.

Nous avons testé pour ce même cas de figure comment l'ajout de diversité influence les performances. Pour cela, nous avons choisi d'augmenter le nombre de données négatives, présentes en abondance pour cette base de données. Nous avons pris en compte 4 cas de figure, le cas standard (N) correspondant à une distribution équilibrée entre patch positif et négatifs (1 : 1), le cas légèrement déséquilibré (A) avec un ratio de 1 : 10, le cas moyennement déséquilibré (B) avec un ratio de 1 : 50 et enfin le cas grandement déséquilibré, avec un ratio de 1 : 100.

Tableau 4.11 Influence du nombre de réseaux sur les performances. Les scores F1 (%) ont été obtenus sur la base de test en utilisant différentes bases d'entraînement (N = 1 :1, A = 1 :10, B = 1 :50, C = 1 :100)

Essai	**2 réseaux (NN)**	**3 réseaux (NAB)**	**4 réseaux (NABC)**
Essai 1	42.39	51.60	32.37
Essai 2	31.87	50.96	37.39
Essai 3	36.70	45.05	35.47
Essai 4	39.77	42.40	30.70
Moyenne	37.68 ± 3.91	46.25 ± 5.10	33.98 ± 2.60

Le tableau 4.11 répertorie les scores F1 obtenus sur la base de test avec ces différentes configurations. On constate que la combinaison de 3 modèles avec les proportions N-A-B permet d'obtenir les meilleurs scores. On constate aussi une chute de performance dans le cas N-A-B-C. Cela peut s'expliquer par le fait que la base d'entrainement la plus déséquilibrée interfère avec le modèle entrainé. En effet, les modèles étant entrainés par paire, la borne de décision particulièrement biaisée par l'approche du modèle 4 a influencé les autres modèles.

La Figure 4.15 représente les scores de Kappa obtenus pour l'essai 1 en utilisant 4 modèles. On constate que l'approche 3 est celle la plus affectée par l'apprentissage biaisé, même si la base de données utilisée (B) n'est pas la plus déséquilibrée. La méthode *Deep Co-training* est basée sur l'apprentissage de modèles par paire, ce qui à tendance à diminuer les prédictions du modèle le plus confiant si le second modèle est inadapté. Il convient de remarquer dans ce cas que les fonctions de coût supervisées n'ont pas été pondérées en conséquence afin d'augmenter davantage la diversité entre les modèles.

Nous pouvons également constater dans le cas de l'utilisation de 3 modèles, que l'augmentation de la diversité ne se reflète pas sur le coefficient de kappa obtenu pour les différents essais, bien qu'on puisse observer une hausse significative du score F1 obtenu (voir Fig. I-5).

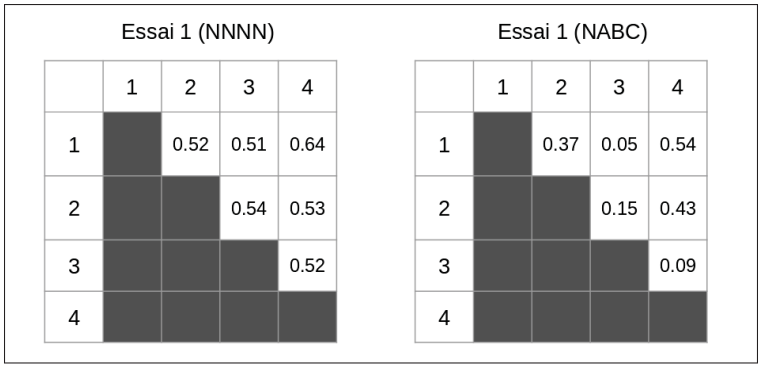


Figure 4.15 Score de kappa obtenue avec 4 modèles

4.4.3 Influence de l'ajout de données générées

Dans cette dernière partie, nous avons voulu tester l'influence de l'ajout de données générées par le générateur de l'approche *BadGan* (Dai *et al.* (2017)) sur les performances du modèle *Deep Co-training*. Le but étant d'apprendre aux modèles à fournir les mêmes prédictions sur les données générées au bord des différentes distributions en ajoutant le coût $\mathcal{C}_{cotgen}$ (Eq. 4.1) à la fonction de coût totale.

$$\mathcal{C}_{cotgen} = H(\frac{1}{2}(h_1(g(x)) + h_2(g(x))) - \frac{1}{2}(H(h_1(g(x))) + H(h_2(g(x)))) \tag{4.1}$$

Il s'agit de donner au modèle une meilleure représentation du problème. Pour cela nous avons entraîné l'approche *BadGan* dans le cas de figure 10-90 et nous avons utilisé le générateur afin de créer un grand nombre d'images similaires à celles présentes sur la figure 4.16. Comme on peut le voir sur la figure, l'approche a été entrainée à reproduire des patches augmentés provenant de la base originale.

Afin de ne pas biaiser les représentations, nous avons entrainé l'approche *BadGan* sur chaque sélection de données afin d'obtenir un générateur d'images par essai.

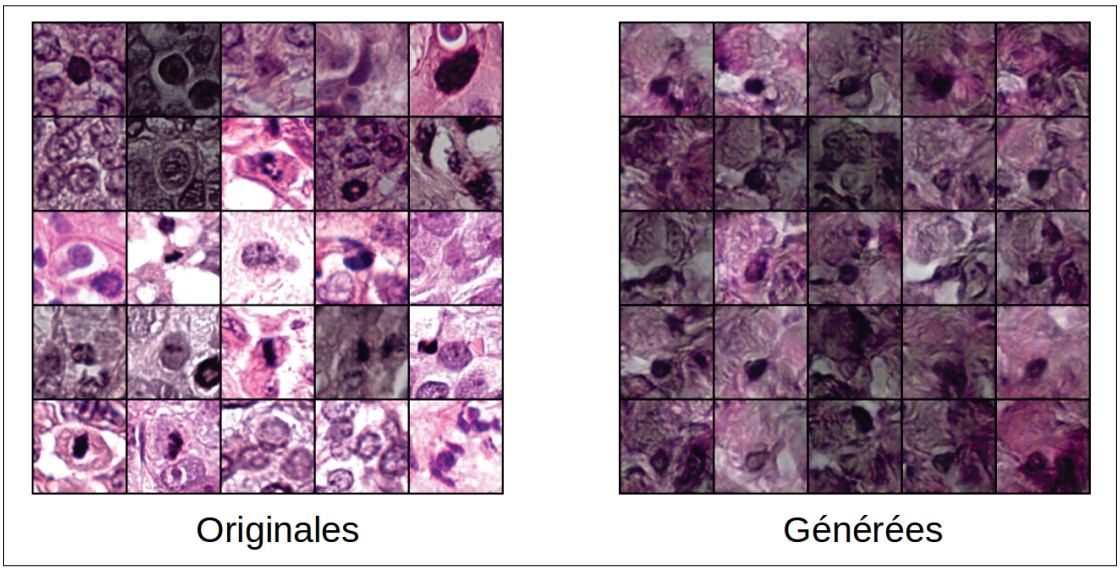


Figure 4.16 Comparaison des patches augmentés de la base originale avec les patches générés par le BadGan

Le tableau 4.12 représente les performances obtenues en combinant les performances de plusieurs modèles entraînés avec des données générées. Dans ce cas, les données générées ont été obtenue à l'aide d'un *BadGan* entrainé sur des bases de données équilibrées.

Tableau 4.12 Influence de l'ajout de données générées sur les performances. Les scores F1 (%) ont été obtenus sur la base de test en utilisant différentes bases d'entraînement (N = 1 : 1, A = 1 : 10, B = 1 : 50).

Essai	**2 réseaux (NN)**	**3 réseaux (NNN)**	**3 réseaux (NAB)**	**4 réseaux (NNNN)**
Essai 1	48.76	52.52	48.61	49.13
Essai 2	46.98	48.07	53.10	48.19
Essai 3	45.46	49.61	46.01	48.60
Essai 4	45.02	46.47	33.25	47.31
Moyenne	46.55 ± 1.46	$\mathbf{49.16 \pm 2.23}$	45.24 ± 7.37	48.3 ± 0.66

On constate des résultats significativement supérieurs comparés au cas classique qui n'utilise pas de données générées. On observe une augmentation moyenne de presque 9% du score F1 en

ajoutant des données générées dans le cas de deux modèles. En moyenne, les meilleurs résultats sont obtenus avec l'utilisation de 3 modèles pour effectuer la prédiction d'un patch. La baisse de performance observé pour le cas N-A-B peut s'expliquer par la distribution des données générées utilisées.

Il convient de noter que cette hausse des performances est obtenue au profit de la complexité. En effet, l'utilisation de plusieurs modèles pour effectuer la prédiction et l'ajout de la fonction de coût (voir Eq. 4.1), basée sur les données générées, conduit à entraîner un plus grand nombre de paramètres et à augmenter le temps d'inférence.

Tableau 4.13 Nombre de paramètres d'entraînement en fonction de l'approche utilisée

Approche utilisée	2 modèles	3 modèles	4 modèles	BadGan
Nombre de paramètres (en millers)	331k	497k	662k	30 637k

Le tableau 4.13 regroupe la quantité de paramètres d'entraînement nécessaire pour l'apprentissage des différents cas de figure étudiés. Nous avons pris en compte le modèle *WResDrop* pour les calculs des paramètres dans le cas du *Deep Co-Taining*. La quantité de paramètres pour le *BadGan* est particulièrement importante à cause de la taille des patches générées (96 x 96 pixels).

CONCLUSION ET RECOMMANDATIONS

Dans ce mémoire, nous avons effectué dans un premier temps une synthèse des approches semi-supervisées. Les approches les plus compétitives, dans le cas de la classification d'images, sont basée sur l'utilisation de modèles profonds.

Ensuite, nous avons comparé les différentes approches semi-supervisées compétitives sur des bases de données médicales d'images histologiques. L'utilisation de telles bases a représenté un certain nombre de défis. Le choix des images est particulièrement influant sur les performances obtenues. La faible quantité d'exemples des différentes classes rend l'entraînement de ces modèles profonds instable, en particulier lorsque l'on n'utilise pas de pré entraînement. Dans les cas ou trop peu de données sont utilisé pour représenter le problème, ce qui est souvent le cas dans les bases de données médical, il arrive qu'on spécialise le modèle à correspondre à la base de validation durant l'apprentissage. Or, dans ce genre de cas, la base de validation et de test peuvent représenter très différemment le problème de classification, en fonction des exemples qui les composent.

En ce qui concerne l'utilisation du pré apprentissage, il va dépendre de la base utilisée. Sur la base BACH, qui est composée d'un faible nombre d'images, l'utilisation du pré entrainement permet d'augmenter significativement les taux de classification obtenus avec l'ensemble des approches. Néanmoins, dans ce cas, l'apport des données sans annotations s'en retrouve amoindri. Dans le cas de la base TUPAC, composée d'un grand nombre d'exemples, il est préférable d'utiliser une architecture, sans pré entrainement, qui a été adaptée pour cette tâche.

De manière générale, l'utilisation de bruit adversaire est particulièrement intéressant pour ce type de données. Ces approches ont l'avantage de générer des transformations supplémentaires, spécifique à l'apprentissage des modèles profonds. Sur la base BACH, l'approche *Deep Co-training* est la plus performante, et sur la base TUPAC, l'approche *VAT* obtient des résultats significativement supérieurs aux autres méthodes, notamment lorsque l'on utilise une petite

quantité de données annotées. Sur les deux bases de données, l'approche *Ladder Network* ne permet pas d'obtenir des performances compétitives dans ce cas de figure. C'est une approche qui est particulièrement performante sur des bases de données comme MNIST, qui nécessite des réseaux de petite taille.

Enfin, nous avons effectuer différents tests sur l'approche *Deep Co-training*. Nous avons pu remarquer une augmentation importante des performances lorsque l'on augmente le nombre de modèle utilisée. L'ajout de diversité permet, jusqu'à une certaine limite, d'améliorer les performances obtenues. L'apprentissage étant effectué par paire, l'utilisation de base d'entrainement trop débalancé pour un des modèles peut corrompre l'apprentissage des autres.

Le premier essai de l'ajout de données générées par l'approche *BadGan* est encourageant. Pour la suite de cette étude, il serait intéressant d'étudier différentes configurations avec l'ajout de ce type de données. Plusieurs axes pourraient être explorer. Par exemple, est-ce l'ajout de données aux bordures des distributions ou l'augmentation d'exemples étudiées qui est à l'origine de cette augmentation ? Est-il plus pertinent d'ajouter des données générées exclusivement sur la classe la moins représenté ? Dans quelle mesure la quantité de données générée impacte les performances des différents modèles ? ou encore Quelle fonction de coût est la plus adaptée pour prendre en compte ce type de données sur d'autres approches semi-supervisées ?

ANNEXE I

COURBES D'APPRENTISSAGE

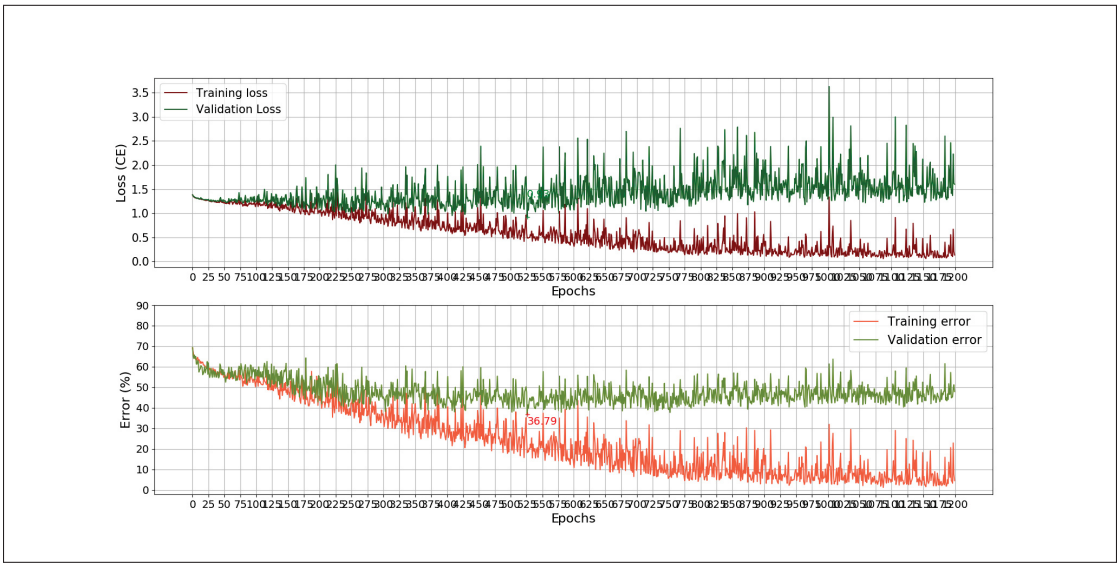


Figure-A I-1 BACH (Patch) ResNet 18 - Sans pré-entrainement LR = 5e-6 essai 1

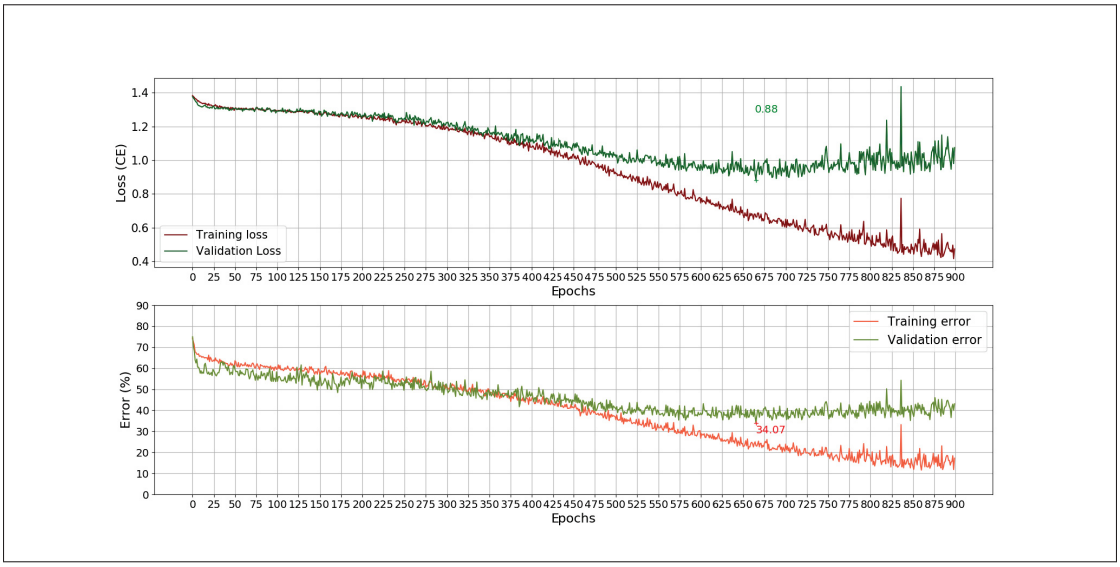


Figure-A I-2 BACH (Patch) ConvL modifié - LR = 1e-5 essai 1

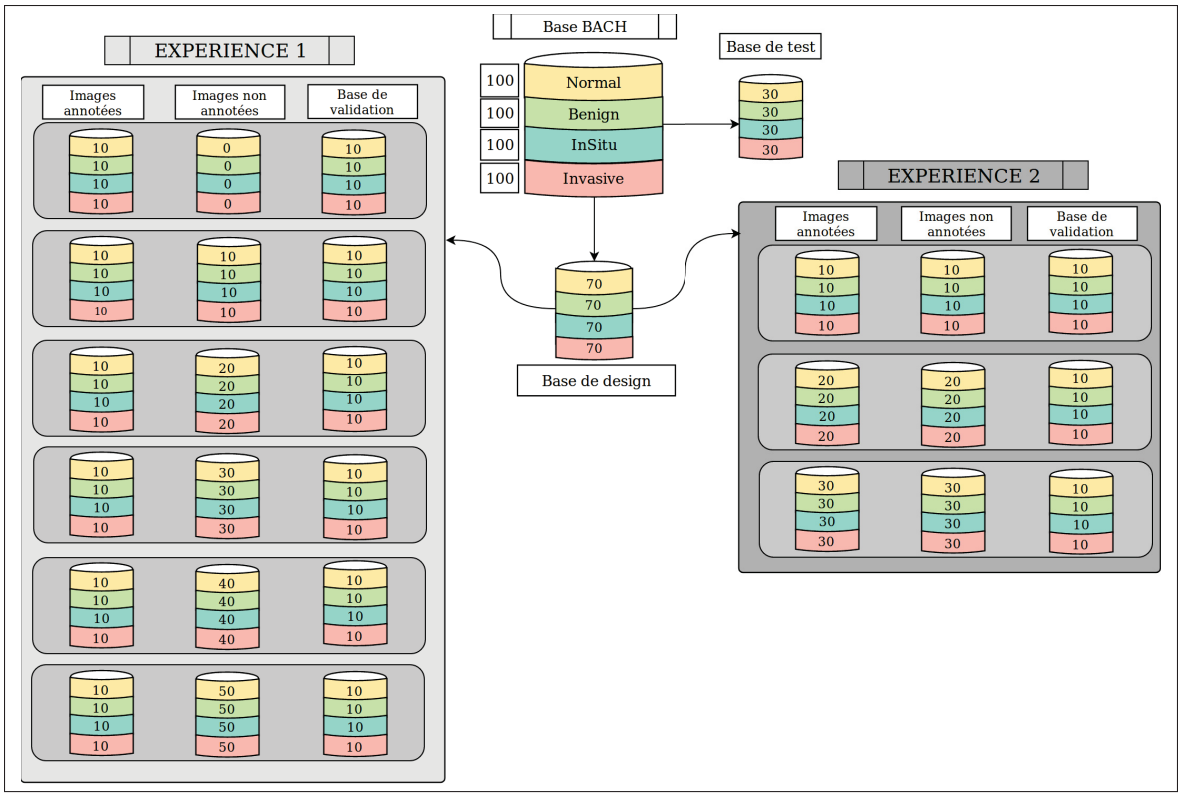


Figure-A I-3 FRépartition des données de la base BACH (utilisation de patches)

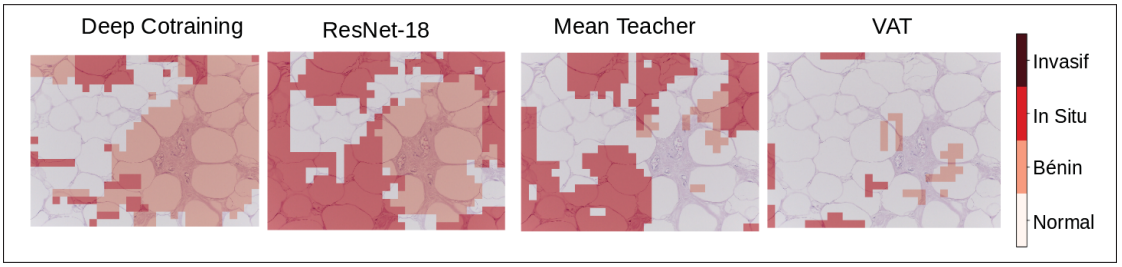


Figure-A I-4 Prédictions des différentes classes (GT : Normal)

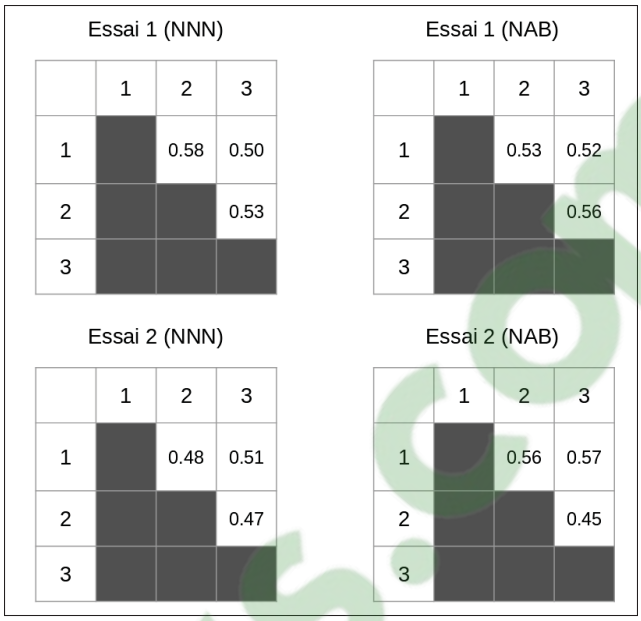


Figure-A I-5 Score de kappa obtenu avec 3 modèles

ANNEXE II

CHOIX DES HYPERPARAMRETRES

Le but de cette partie est d'étudier l'effet des divers hyperparamètres communs (taux d'apprentissage, normalisation) et spécifiques, sur les performances en fonction des différentes techniques semi-supervisées et de différentes proportions d'images annotées et non annotées. Dans un premier temps, nous avons cherché à optimiser le modèle ResNet pour les différentes configurations de l'expérience 3. Nous avons fait varier les hyperparamètres de la manière suivante : taux d'apprentissage [1e-5,5e-5,1e-4,5e-4,1e-3] ; Norme L1 [0,1e-3,1e-4,1e-5], Norme L2 [0,1-3,1e-4,1e-5]. Le tableau II-1 résume les meilleures performances obtenues pour les différents essais. On constate que d'un essai à l'autre, les performances sont assez inégales. Le choix de la base d'entraînement et de validation joue un rôle particulièrement important dans notre cas. De plus, on peut remarquer que les meilleurs hyperparamètres pour une donnée ne sont pas toujours généralisables. Le tableau II-2 comporte les différents résultats obtenus sur la base de test en comparant différentes techniques d'optimisation. L'optimisation globale (GLOB) est basée sur la recherche de la meilleur moyenne, on choisit ici les hyperparamètres ayant permis d'obtenir les meilleurs performances en moyenne sur les 4 essais pour chaque configuration. L'approche au cas par cas (CPC) prend en compte les essais de manière indépendante : ont été sélectionnés ici uniquement les essais ayant obtenus les meilleurs performances sur la base de validation. Enfin, l'approche générale (GEN) consiste à fixer un ensemble d'hyperparamètres communs aux différentes configuration. Dans le cas de l'expérience 3, les taux de normalisation ont été fixé à 0 pour l'ensemble des approches.

Tableau-A II-1 Recherche d'hyperparamètres ResNET-18 pré-entraînée (Expérience 3)

ResNET 10 img/classe (EXP 3)	**Essai 1**	**Essai 2**	**Essai 3**	**Essai 4**	**Moyenne (Val)**
LR = 1e-4 || L1 = 1e-4 || L2 = 1e-5	**73.75**	68.75	76.25	**73.75**	**73.12 ± 2.72**
LR = 1e-4 || L1 = 1e-4 || L2 = 0	72.5	**70.0**	**77.5**	72.5	**73.12 ± 2.72**
ResNET 20 img/classe (EXP 3)	**Essai 1**	**Essai 2**	**Essai 3**	**Essai 4**	**Moyenne (Val)**
LR = 1e-4 || L1 = 1e-4 || L2 = 1e-3	**83.75**	73.75	75.0	77.5	77.5 ± 3.85
LR = 1e-4 || L1 = 1e-5 || L2 = 1e-5	81.25	**77.5**	**80.0**	76.25	**78.75 ± 1.97**
LR = 1e-4 || L1 = 0 || L2 = 1e-5	76.25	76.25	73.75	**78.75**	76.25 ± 1.76
ResNET 30 img/classe (EXP 3)	**Essai 1**	**Essai 2**	**Essai 3**	**Essai 4**	**Moyenne (Val)**
LR = 1e-4 || L1 = 1e-3 || L2 = 0	**81.25**	78.75	83.75	80.0	80.93 ± 1.84
LR = 1e-4 || L1 = 1e-5 || L2 = 1e-3	78.75	**80.0**	**86.25**	**81.25**	**81.56 ± 2.84**
ResNET 40 img/classe (EXP 3)	**Essai 1**	**Essai 2**	**Essai 3**	**Essai 4**	**Moyenne (Val)**
LR = 1e-4 || L1 = 0 || L2 = 0	**85.0**	**82.5**	83.75	81.25	**83.12 ± 1.39**
LR = 1e-4 || L1 = 1e-4 || L2 = 0	83.75	81.25	**85.0**	81.25	82.81 ± 1.62
LR = 1e-4 || L1 = 1e-3 || L2 = 1e-3	78.75	80.0	82.5	**83.75**	81.25 ± 1.97
ResNET 50 img/classe (EXP 3)	**Essai 1**	**Essai 2**	**Essai 3**	**Essai 4**	**Moyenne (Val)**
LR = 1e-4 || L1 = 0 || L2 = 1e-3	**88.75**	83.75	81.25	**85.0**	**84.68 ± 2.70**
LR = 1e-4 || L1 = 1e-3 || L2 = 1e-5	85.0	**87.5**	**83.75**	81.25	84.37 ± 2.25
ResNET 60 img/classe (EXP 3)	**Essai 1**	**Essai 2**	**Essai 3**	**Essai 4**	**Moyenne (Val)**
LR = 1e-4 || L1 = 0 || L2 = 0	91.25	86.25	83.75	82.5	85.93 ± 3.35
LR = 1e-4 || L1 = 1e-4 || L2 = 0	90.0	88.75	82.5	81.25	85.62 ± 3.80
LR = 1e-4 || L1 = 1e-5 || L2 = 1e-4	86.25	82.5	85.0	80.0	83.43 ± 2.40

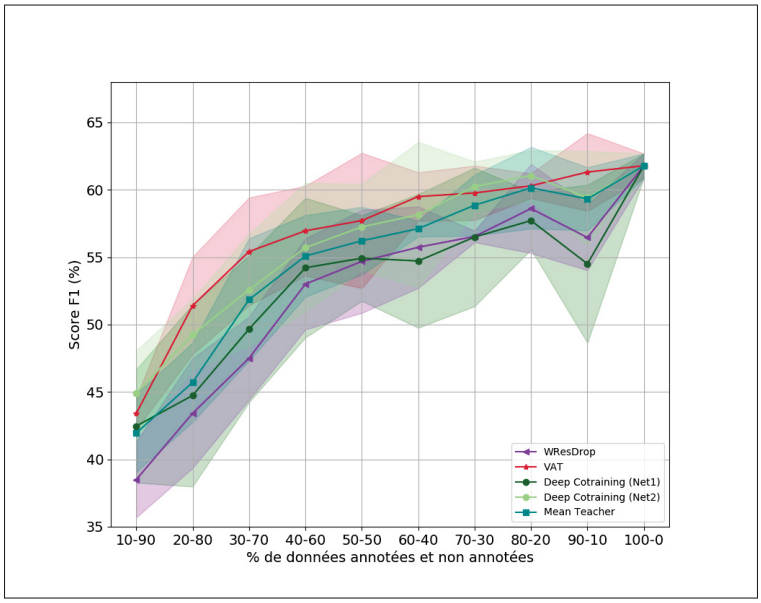


Figure-A II-1 Expérience 3, base TUPAC, base de validation, sans pré-entrainement

Tableau-A II-2 Comparaison performances obtenues sur la base de test avec le ResNet-18 pré-entraîné (Expérience 3)

ResNET 10 img/classe (EXP 3)	**Essai 1**	**Essai 2**	**Essai 3**	**Essai 4**	**Moyenne (Test)**
GLOB	71.25	55.00	61.25	63.74	62.81 ± 5.76
CPC	71.25	55.00	62.5	63.74	63.12 ± 5.76
GEN	70.0	56.25	62.5	63.74	63.12 ± 4.88
ResNET 20 img/classe (EXP 3)	**Essai 1**	**Essai 2**	**Essai 3**	**Essai 4**	**Moyenne (Test)**
GLOB	75.0	73.75	70.0	71.25	72.5 ± 1.97
CPC	76.25	73.75	70.0	71.25	72.81 ± 2.40
GEN	80.0	70.0	75.0	71.25	74.06 ± 3.89
ResNET 30 img/classe (EXP 3)	**Essai 1**	**Essai 2**	**Essai 3**	**Essai 4**	**Moyenne (Test)**
GLOB	76.25	77.5	81.25	73.75	77.18 ± 2.70
CPC	78.75	77.5	81.25	73.75	77.81 ± 2.70
GEN	77.5	78.75	80.0	72.5	77.18 ± 2.84
ResNET 40 img/classe (EXP 3)	**Essai 1**	**Essai 2**	**Essai 3**	**Essai 4**	**Moyenne (Test)**
GLOB	72.5	83.75	83.75	80.0	80.0 ± 4.59
CPC	72.5	83.75	73.75	73.75	75.93 ± 4.53
GEN	72.5	83.75	83.75	80.0	80.0 ± 4.59
ResNET 50 img/classe (EXP 3)	**Essai 1**	**Essai 2**	**Essai 3**	**Essai 4**	**Moyenne (Test)**
GLOB	78.75	83.75	81.25	85.0	82.18 ± 2.40
CPC	78.75	81.25	77.5	85.0	80.62 ± 2.86
GEN	81.25	80.0	83.75	83.75	82.18 ± 1.62
ResNET 60 img/classe (EXP 3)	**Essai 1**	**Essai 2**	**Essai 3**	**Essai 4**	**Moyenne (Test)**
GLOB	87.5	88.75	82.5	80.0	84.68 ± 3.57
CPC	87.5	86.25	88.75	80.0	85.62 ± 3.36
GEN	87.5	88.75	82.5	80.0	84.68 ± 3.57

Tableau-A II-3 Recensement des hyperparamètres de la base BACH (avec pré-entrainement)

Approches	**Hyperparamètres**	**EXP 1**	**EXP 3**
Architecture supervisée	Taux d'apprentissage	1e-4	1e-4
	Norme L1	0	0
	Norme L2	0	0
	Itérations	250	250
	Batch	20	20
VAT	ϵ_{VAT}	0.8	0.8
	λ_{VAT}	fixé à 1	fixé à 1
Mean Teacher	λ_{MT}	10	$D_L \in$ [10,20,30]$\Rightarrow$ 30 $D_L \in$ [40,50] $\Rightarrow$ 10
	α	0.999	0.99
Deep Cotraining	ϵ_{DCTR}	0.01	$D_L \in$ [10,20] $\Rightarrow$ 0.02 $D_L \in$ [30,40,50] $\Rightarrow$ 0.01
	λ_{cot}	20	1
	λ_{dif}	fixé à 0.5	fixé à 0.5

Tableau-A II-4 Recensement des hyperparamètres de la base BACH (sans pré-entrainement)

Approches	**Hyperparamètres**	**EXP1 & 3**
Architecture supervisée	Taux d'apprentissage	5e-5
	Norme L1	1e-5
	Norme L2	1e-4
	Itérations	500
	Batch	20
VAT	ϵ_{VAT}	0.8
	λ_{VAT}	fixé à 1
Mean Teacher	λ_{MT}	30
	α	0.99
Deep Cotraining	ϵ_{DCTR}	0.02
	λ_{cot}	10
	λ_{dif}	fixé à 0.5
Ladder Network	σ_{bruit}	0.1
	${\lambda_{LAD}}^{1}$	1000
	${\lambda_{LAD}}^{2}$	10
	${\lambda_{LAD}}^{>2}$	0.01

Tableau-A II-5 Recensement des hyperparamètres de la base TUPAC (sans pré-entrainement)

Approches	**Hyperparamètres**	**EXP 1**	EXP 3
Architecture supervisée	Taux d'apprentissage	1e-3	1e-3
	Norme L1	0	0
	Norme L2	1e-4	1e-4
	Itérations	10 000	10 000
	Batch	128	128
VAT	ϵ_{VAT}	0.1	$D_L \in$ [10,20,30,40,50] $\Rightarrow$ 0.5 $D_L \in$ [60,70,80,90] $\Rightarrow$ 0.1
	λ_{VAT}	fixé à 1	fixé à 1
Mean Teacher	λ_{MT}	5	5
	α	0.9	0.9
Deep Cotraining	ϵ_{DCTR}	0.02	0.02
	λ_{cot}	1	1
	λ_{dif}	fixé à 0.5	fixé à 0.5
Ladder Network	σ_{bruit}	0.1	0.1
	${\lambda_{LAD}}^1$	1000	1000
	${\lambda_{LAD}}^2$	10	10
	${\lambda_{LAD}}^{>2}$	0.01	0.01

Tableau-A II-6 Comparaison des performances obtenue entre les bases de validation et de test (Expérience 1 sans pré-entrainement, base BACH)

Approches	**10-0 (RES)**	**10-10 (VAT)**	**10-20 (VAT)**	**10-30 (VAT)**	**10-40 (VAT)**	**10-50 (VAT)**
Score exactitude (VAL)	43.12 ± 6.76	47.18 ± 8.54	45.0 ± 5.51	48.12 ± 7.93	45.31 ± 8.26	43.43 ± 8.11
Score exactitude (TEST)	44.68 ± 3.89	41.25 ± 5.86	45.93 ± 6.01	50.31 ± 4.62	44.37 ± 5.89	44.06 ± 5.82

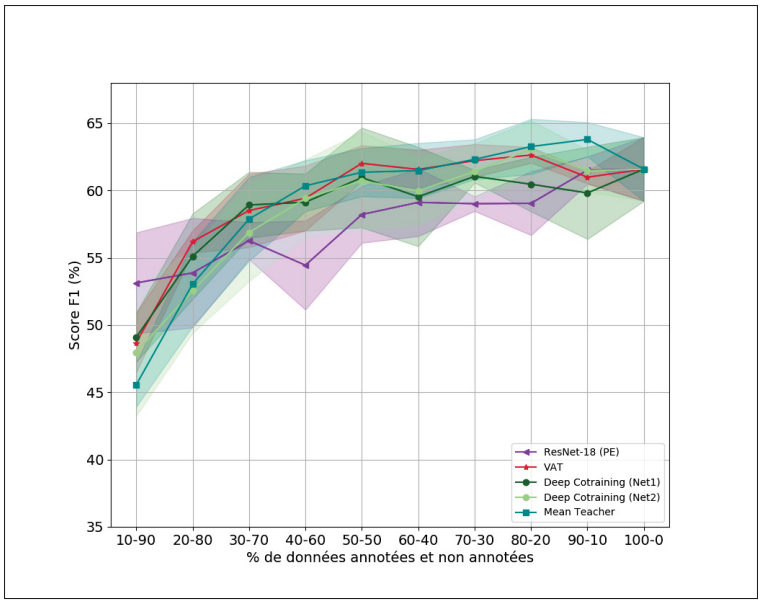


Figure-A II-2 Expérience 3, base TUPAC, base de validation, avec pré-entrainement

ANNEXE III

MATRICES DE CONFUSION DU DEEP COTRAINING

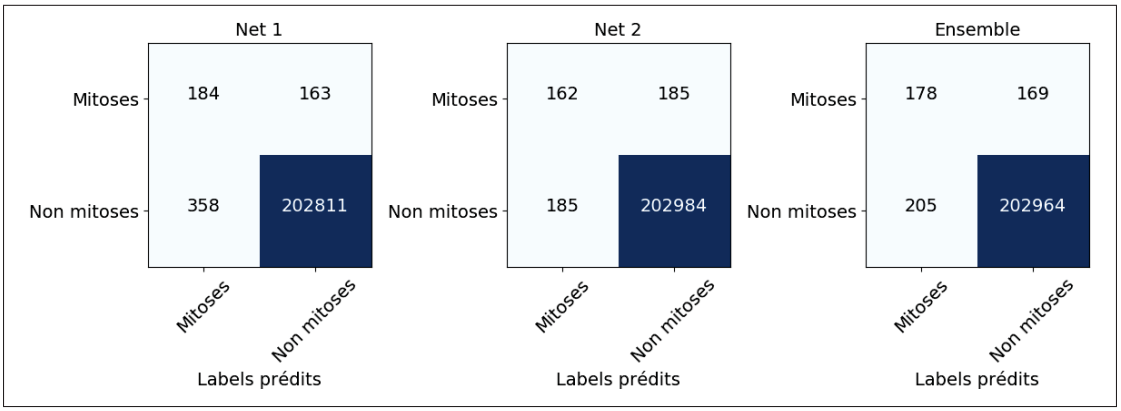


Figure-A III-1 Matrice de confusion avec images générées, essai 1

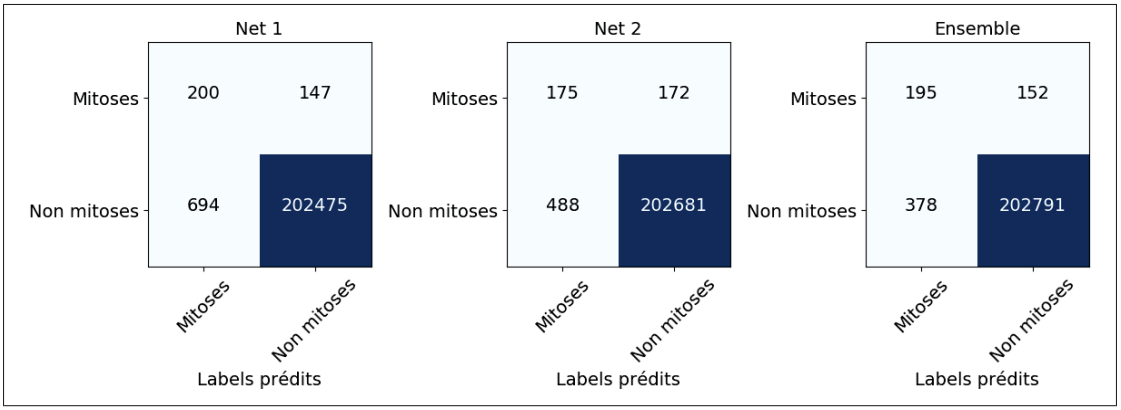


Figure-A III-2 Matrice de confusion NN, essai 1

BIBLIOGRAPHIE

Abney, S. (2002). Bootstrapping. *Proceedings of the 40th annual meeting of the association for computational linguistics*.

Araújo, T., Aresta, G., Castro, E., Rouco, J., Aguiar, P., Eloy, C., Polónia, A. & Campilho, A. (2017). Classification of breast cancer histology images using convolutional neural networks. *Plos one*, 12(6), e0177544.

Aresta, G., Araújo, T., Kwok, S., Chennamsetty, S. S., Safwan, M., Alex, V., Marami, B., Prastawa, M., Chan, M., Donovan, M. et al. (2018). Bach : Grand challenge on breast cancer histology images. *arxiv preprint arxiv :1808.04277*.

Bejnordi, B. E., Veta, M., Van Diest, P. J., Van Ginneken, B., Karssemeijer, N., Litjens, G., Van Der Laak, J. A., Hermsen, M., Manson, Q. F., Balkenhol, M. et al. (2017). Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer. *Jama*, 318(22), 2199–2210.

Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. Dans *Neural networks : Tricks of the trade* (pp. 437–478). Springer.

Bengio, Y., Louradour, J., Collobert, R. & Weston, J. (2009). Curriculum learning. *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48.

Blum, A. & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. *Proceedings of the eleventh annual conference on computational learning theory*, pp. 92–100.

Bradski, G. (2000). The OpenCV Library. *Dr. dobb's journal of software tools*.

Brancati, N., Frucci, M. & Riccio, D. (2018). Multi-classification of breast cancer histology images by using a fine-tuning strategy. *International conference image analysis and recognition*, pp. 771–778.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1), 5–32.

Carpenter, A. E., Jones, T. R., Lamprecht, M. R., Clarke, C., Kang, I. H., Friman, O., Guertin, D. A., Chang, J. H., Lindquist, R. A., Moffat, J. et al. (2006). Cellprofiler : image analysis software for identifying and quantifying cell phenotypes. *Genome biology*, 7(10), R100.

Chapelle, O., Schlkopf, B. & Zien, A. (2006). *Semi-supervised learning* [The MIT Press]. doi : 978-0-262-03358-9.

Chen, H., Dou, Q., Wang, X., Qin, J., Heng, P.-A. et al. (2016). Mitosis detection in breast cancer histology images via deep cascaded networks.

Chennamsetty, S. S., Safwan, M. & Alex, V. (2018). Classification of breast cancer histology image using ensemble of pre-trained neural networks. *International conference image analysis and recognition*, pp. 804–811.

Cireşan, D. C., Giusti, A., Gambardella, L. M. & Schmidhuber, J. (2013). Mitosis detection in breast cancer histology images with deep neural networks. *International conference on medical image computing and computer-assisted intervention*, pp. 411–418.

Coelho, L. P., Ahmed, A., Arnold, A., Kangas, J., Sheikh, A.-S., Xing, E. P., Cohen, W. W. & Murphy, R. F. (2010). Structured literature image finder : extracting information from text and images in biomedical literature. Dans *Linking Literature, Information, and Knowledge for Biology* (pp. 23–32). Springer.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1), 37–46.

Cortes, C. & Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3), 273–297.

Dai, Z., Yang, Z., Yang, F., Cohen, W. W. & Salakhutdinov, R. R. (2017). Good semi-supervised learning that requires a bad gan. *Advances in neural information processing systems*, pp. 6510–6520.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. & Fei-Fei, L. (2009). ImageNet : A Large-Scale Hierarchical Image Database. *Cvpr09*.

DeVries, T. & Taylor, G. W. (2018). Learning confidence for out-of-distribution detection in neural networks. *arxiv preprint arxiv :1802.04865*.

Doyle, S., Hwang, M., Shah, K., Madabhushi, A., Feldman, M. & Tomaszeweski, J. (2007). Automated grading of prostate cancer using architectural and textural image features. *Biomedical imaging : from nano to macro, 2007. isbi 2007. 4th ieee international symposium on*, pp. 1284–1287.

Durand, T., Mordan, T., Thome, N. & Cord, M. Wildcat : Weakly supervised learning of deep convnets for image classification, pointwise localization and segmentation.

Elston, C. W. & Ellis, I. O. (1991). Pathological prognostic factors in breast cancer. i. the value of histological grade in breast cancer : experience from a large study with long-term follow-up. *Histopathology*, 19(5), 403–410.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M. & Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639), 115.

Foulds, J. & Frank, E. (2010). A review of multi-instance learning assumptions. *The knowledge engineering review*, 25(1), 1–25.

Frénay, B. & Verleysen, M. (2014). Classification in the presence of label noise : a survey. *Ieee transactions on neural networks and learning systems*, 25(5), 845–869.

Frénay, B., Kabán, A. et al. (2014). A comprehensive introduction to label noise. *Esann*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y. (2014a). Generative adversarial nets. *Advances in neural information processing systems*, pp. 2672–2680.

Goodfellow, I. J., Shlens, J. & Szegedy, C. (2014b). Explaining and harnessing adversarial examples. *arxiv preprint arxiv :1412.6572*.

Haralick, R. M., Shanmugam, K. et al. (1973). Textural features for image classification. *Ieee transactions on systems, man, and cybernetics*, (6), 610–621.

He, K., Zhang, X., Ren, S. & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the ieee conference on computer vision and pattern recognition*, pp. 770–778.

Hou, L., Samaras, D., Kurc, T. M., Gao, Y., Davis, J. E. & Saltz, J. H. (2016). Patch-based convolutional neural network for whole slide tissue image classification. *Proceedings of the ieee conference on computer vision and pattern recognition*, pp. 2424–2433.

Huang, G., Liu, Z., Van Der Maaten, L. & Weinberger, K. Q. (2017). Densely connected convolutional networks. *Cvpr*, 1(2), 3.

Ilse, M., Tomczak, J. M. & Welling, M. (2018). Attention-based deep multiple instance learning. *arxiv preprint arxiv :1802.04712*.

Ioffe, S. & Szegedy, C. (2015). Batch normalization : Accelerating deep network training by reducing internal covariate shift. *arxiv preprint arxiv :1502.03167*.

Junbo Zhao,Michael Mathieu, R. G. & Lecun, Y. (2015). Stacked what-where auto-encoders. *arxiv preprint arxiv :1506.02351*.

Kingma, D. P., Mohamed, S., Rezende, D. J. & Welling, M. (2014). Semi-supervised learning with deep generative models. *Advances in neural information processing systems*, pp. 3581–3589.

Krizhevsky, A. & Hinton, G. (2009). *Learning multiple layers of features from tiny images*.

Kuncheva, L. I. & Whitaker, C. J. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2), 181–207.

Kwok, S. (2018). Multiclass classification of breast cancer in whole-slide images. *International conference image analysis and recognition*, pp. 931–940.

Laine, S. & Aila, T. (2016). Temporal ensembling for semi-supervised learning. *arxiv preprint arxiv :1610.02242*.

Lee, D.-H. (2013). Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. *Workshop on challenges in representation learning, icml*, 3, 2.

Liu, B., Wang, M., Foroosh, H., Tappen, M. & Pensky, M. (2015). Sparse convolutional neural networks. *Proceedings of the ieee conference on computer vision and pattern recognition*, pp. 806–814.

Maas, A. L., Hannun, A. Y. & Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. *Proc. icml*, 30(1), 3.

Macenko, M., Niethammer, M., Marron, J., Borland, D., Woosley, J. T., Guan, X., Schmitt, C. & Thomas, N. E. (2009). A method for normalizing histology slides for quantitative analysis. *Biomedical imaging : From nano to macro, 2009. isbi'09. ieee international symposium on*, pp. 1107–1110.

Miyato, T., Maeda, S.-i., Koyama, M. & Ishii, S. (2017). Virtual adversarial training : a regularization method for supervised and semi-supervised learning. *arxiv preprint arxiv :1704.03976*.

Muhlenbach, F., Lallich, S. & Zighed, D. A. (2004). Identifying and handling mislabelled instances. *Journal of intelligent information systems*, 22(1), 89–109.

Nair, V. & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. *Proceedings of the 27th international conference on machine learning (icml-10)*, pp. 807–814.

Oliver, A., Odena, A., Raffel, C., Cubuk, E. D. & Goodfellow, I. J. (2018). Realistic evaluation of deep semi-supervised learning algorithms. *arxiv preprint arxiv :1804.09170*.

Otsu, N. (1979). A threshold selection method from gray-level histograms. *Ieee transactions on systems, man, and cybernetics*, 9(1), 62–66.

Paeng, K., Hwang, S., Park, S. & Kim, M. (2017). A unified framework for tumor proliferation score prediction in breast histopathology. Dans *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support* (pp. 231–239). Springer.

Pezeshki, M., Fan, L., Brakel, P., Courville, A. & Bengio, Y. (2016). Deconstructing the ladder network architecture. *International conference on machine learning*, 2368-2376.

Qiao, S., Shen, W., Zhang, Z., Wang, B. & Yuille, A. (2018). Deep co-training for semi-supervised image recognition. *Proceedings of the european conference on computer vision (eccv)*, pp. 135–152.

Rasmus, A., Berglund, M., Honkala, M., Valpola, H. & Raiko, T. (2015). Semi-supervised learning with ladder networks. *Advances in neural information processing systems*, 3546-3554.

Rousson, M., Hedlund, M., Andersson, M., Jacobsson, L., Lathen, G., Norell, B., Jimenez-del Toro, O., Mueller, H. & Atzori, M. (2018). Tumor proliferation assessment of whole slide images. *Medical imaging 2018 : Digital pathology*, 10581, 105810Y.

Roux, L., Racoceanu, D., Capron, F., Calvo, J., Attieh, E., Le Naour, G. & Gloaguen, A. (2014). Mitos & atypia.

Roux, M. (2014). Éditorial. *Vie & sciences de l'entreprise*, (1), 10–13.

Sajjadi, M., Javanmardi, M. & Tasdizen, T. (2016). Regularization with stochastic transformations and perturbations for deep semi-supervised learning. *Advances in neural information processing systems*, pp. 1163–1171.

Salimans, T. & Kingma, D. P. (2016). Weight normalization : A simple reparameterization to accelerate training of deep neural networks. *Advances in neural information processing systems*, pp. 901–909.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A. & Chen, X. (2016). Improved techniques for training gans. *Advances in neural information processing systems*, pp. 2234–2242.

Simpson, J. F., Gray, R., Dressler, L. G., Cobau, C. D., Falkson, C. I., Gilchrist, K. W., Pandya, K. J., Page, D. L. & Robert, N. J. (2000). Prognostic value of histologic grade and proliferative activity in axillary node–positive breast cancer : results from the eastern cooperative oncology group companion study, est 4189. *Journal of clinical oncology*, 18(10), 2059–2069.

Sneath, P. H., Sokal, R. R. et al. (1973). *Numerical taxonomy. the principles and practice of numerical classification.*

Société canadienne du cancer. (2018). Statistiques canadiennes sur le cancer 2018. Repéré à cancer.ca/Canadian-Cancer-Statistics-2018-FR.

Spanhol, F. A., Oliveira, L. S., Petitjean, C. & Heutte, L. (2016). A dataset for breast cancer histopathological image classification. *Ieee transactions on biomedical engineering*, 63(7), 1455–1462.

Springenberg, J. T. (2015). Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arxiv preprint arxiv :1511.06390*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. (2014). Dropout : a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1), 1929–1958.

Stanitsas, P., Cherian, A., Li, X., Truskinovsky, A., Morellas, V. & Papanikolopoulos, N. (2016). Evaluation of feature descriptors for cancerous tissue recognition. *Pattern recognition (icpr), 2016 23rd international conference on*, pp. 1490–1495.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. *Proceedings of the ieee conference on computer vision and pattern recognition*, pp. 2818–2826.

Szegedy, C., Ioffe, S., Vanhoucke, V. & Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. *Aaai*, 4, 12.

Tarvainen, A. & Valpola, H. (2017). Mean teachers are better role models : Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, pp. 1195–1204.

Vapnik, V. (1998). *Statistical learning theory. 1998*. Wiley, New York.

Veta, M., Pluim, J. P., Van Diest, P. J. & Viergever, M. A. (2014). Breast cancer histopathology image analysis : A review. *Ieee transactions on biomedical engineering*, 61(5), 1400–1411.

Veta, M., Van Diest, P. J., Willems, S. M., Wang, H., Madabhushi, A., Cruz-Roa, A., Gonzalez, F., Larsen, A. B., Vestergaard, J. S., Dahl, A. B. et al. (2015). Assessment of algorithms for mitosis detection in breast cancer histopathology images. *Medical image analysis*, 20(1), 237–248.

Veta, M., Heng, Y. J., Stathonikos, N., Bejnordi, B. E., Beca, F., Wollmann, T., Rohr, K., Shah, M. A., Wang, D., Rousson, M. et al. (2018a). Predicting breast tumor proliferation from whole-slide images : the tupac16 challenge. *arxiv preprint arxiv :1807.08284*.

Veta, M., Josien P.W. Pluim,Nikolaos Stathonikos, P. J. v. D. F. B. & Beck, A. (2018b). Tumor proliferation assessment challenge 2016. Repéré à http://tupac.tue-image.nl/node/2.

Wang, D., Khosla, A., Gargeya, R., Irshad, H. & Beck, A. H. (2016). Deep learning for identifying metastatic breast cancer. *arxiv preprint arxiv :1606.05718*.

Wang, H. & Raj, B. (2017). On the origin of deep learning. *arxiv preprint arxiv :1702.07800*.

Wang, L. & He, D.-C. (1990). Texture classification using texture spectrum. *Pattern recognition*, 23(8), 905–910.

Zagoruyko, S. & Komodakis, N. (2016). Wide residual networks. *arxiv preprint arxiv :1605.07146*.

Zanjani, F. G., Zinger, S. et al. (2018). Cancer detection in histopathology whole-slide images using conditional random fields on deep embedded spaces. *Medical imaging 2018 : Digital pathology*, 10581, 105810I.

Zerhouni, E., Lányi, D., Viana, M. & Gabrani, M. (2017). Wide residual networks for mitosis detection. *Biomedical imaging (isbi 2017), 2017 ieee 14th international symposium on*, pp. 924–928.

Zhou, Z.-H. (2012). *Ensemble methods : foundations and algorithms*. Chapman and Hall/CRC.

Zhou, Z.-H. (2017). A brief introduction to weakly supervised learning. *National science review*, 5(1), 44–53.

Zhou, Z., Shin, J. Y., Zhang, L., Gurudu, S. R., Gotway, M. B. & Liang, J. (2017). Fine-tuning convolutional neural networks for biomedical image analysis : Actively and incrementally. *Cvpr*, pp. 4761–4772.