# Résolution variable et information privilégiée pour la reconnaissance d'images

Marion Chevalier

**THÈSE DE DOCTORAT DE**
**l'UNIVERSITÉ PIERRE ET MARIE CURIE**

Spécialité

**Informatique**

École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

**Marion CHEVALIER**

Pour obtenir le grade de

**DOCTEUR de l'UNIVERSITÉ PIERRE ET MARIE CURIE**

Sujet de la thèse :

# Résolution variable et information privilégiée pour la reconnaissance d'images

Jury composé de :

| | | |
|---|---|---|
| Mme Catherine Achard | Université Pierre et Marie Curie | Examinatrice |
| M. François Brémond | INRIA Sophia Antipolis | Rapporteur |
| M. Stéphane Canu | INSA Rouen | Examinateur |
| M. Matthieu Cord | Université Pierre et Marie Curie | Directeur de thèse |
| M. Gilles Hénaff | Thales Optronique S.A.S. | Encadrant |
| M. Patrick Pérez | Technicolor | Rapporteur |
| M. Nicolas Thome | Université Pierre et Marie Curie | Co-directeur |

# Résumé

La classification des images revêt un intérêt majeur dans de nombreuses tâches de reconnaissance visuelle, et plus particulièrement dans le cadre de la reconnaissance de véhicules au sol via les systèmes aéroportés. Dans ce contexte, les images traitées sont de faible résolution du fait de la large distance entre le porteur et la scène observée. Durant la phase d'apprentissage, des données complémentaires à ces images faiblement résolues peuvent être disponibles, qu'il s'agisse de connaissances relatives aux conditions de prise de vue ou de la version haute-résolution des images observées.

Dans nos travaux, on s'intéresse au problème de la reconnaissance d'images faiblement résolues. Pour répondre à ce problème, on examine d'abord l'adéquation des méthodes de l'état de l'art pour la représentation et la classification de telles images. Plus précisément, on évalue l'impact de la résolution des images sur les performances de plusieurs types de représentations. On montre en particulier l'intérêt des réseaux de neurones convolutionnels, des modèles profonds pour la représentation et la classification d'images ayant récemment prouvé leur efficacité sur une multitude de tâches en reconnaissance visuelle. On propose également une architecture apprise sur les données d'intérêt permettant d'améliorer significativement les performances des représentations pré-entraînées sur une base de données externe.

D'autre part, dans ce contexte, on dispose des versions haute-résolution des images d'apprentissage. Afin de bénéficier de ces données complémentaires durant la phase d'entraînement, on s'appuie sur le cadre de l'apprentissage avec information privilégiée. Cette approche récente permet de prendre en compte des données complémentaires aux exemples d'apprentissage, indisponibles pour les exemples de test, dite *privilégiée*. On se place dans le cadre où l'information privilégiée associée à un exemple d'entraînement permet d'en définir un niveau de difficulté. Dans ce manuscrit, nous proposons deux méthodes permettant d'intégrer l'information privilégiée dans l'apprentissage des réseaux de neurones convolutionnels profonds. Dans notre premier modèle deep+, l'information privilégiée permet de calculer des coefficients qui pondèrent les exemples d'apprentissage, attribuant un poids plus important aux images les plus facilement reconnaissables, *i.e.* les plus représentatives de leur classe. Notre deuxième modèle deep++ introduit une contrainte de similitude entre le modèle appris sur l'information privilégiée et le modèle appris sur les exemples d'apprentissage, exploitant un terme additionnel de difficulté relative entre les deux espaces de représentation.

Expérimentalement, on montre l'intérêt de nos deux modèles par rapport aux méthodes concurrentes de l'état de l'art intégrant l'information privilégiée. On montre notamment l'intérêt d'utiliser des réseaux de neurones profonds plutôt que des méthodes plus superficielles, comme c'est le cas dans de nombreuses approches historiques de l'apprentissage avec information privilégiée. De plus, on montre l'intérêt de nos modèles par

rapport aux méthodes standard sans information privilégiée. On considère pour cela plusieurs contextes de classification d'images. Ainsi, on montre l'efficacité de nos modèles sur la base historique MNIST ainsi que sur plusieurs bases orientées grain-fin. En particulier, l'étude menée sur UPMC-Food-101, une large base d'images récoltées sur le net, met en évidence l'intérêt de nos approches sur une base de données contenant du bruit d'annotation. On fournit également une large évaluation quantitative et qualitative des différents aspects de nos modèles. Enfin, on montre l'intérêt de nos méthodes dans un contexte plus opérationnel, en évaluant nos approches sur une base d'images propre à Thales Optronique S.A.S..

# Table des matières

# Liste des figures

# Liste des tableaux

# Liste des acronymes

| | |
|---|---|
| AP | Average Precision |
| AREOS | Airborne Reconnaissance Observation System |
| ATD | Automatic Target Detection |
| ATI | Automatic Target Identification |
| ATR | Automatic Target Recognition |
| BOSSA | Bag of Statistical Sampling Analysis |
| BoW | Bag of Words (*Sac de mots*) |
| CCA | Canonical Correlation Analysis (*Analyse de corrélations canoniques*) |
| CNN | Convolutional Neural Network (*Réseau de neurones convolutionnel*) |
| DBN | Deep Belief Network |
| DPM | Deformable Part Model |
| ELU | Exponential Linear Unit |
| FGVC-Aircraft | Fine-Grained Visual Classification of Aircraft |
| FT | Fine-tuned (*adapté* ou *ré-appris*) |
| FV | Fisher Vector |
| GMM | Gaussian Mixture Model (*Mélange de gaussiennes*) |
| HOG | Histogram of Oriented Gradients (*Histogramme de gradients orientés*) |
| ILSVRC | ImageNet Large Scale Visual Recognition Challenge |
| $k$-NN | $k$ Nearest Neighbours ($k$ *plus proches voisins*) |
| LBP | Local Binary Pattern |
| LIR | Loss Inequality Regularization |
| LUPI | Learning Using Privileged Information (*Apprentissage avec information privilégiée*) |
| LR-CNN | Low-Resolution CNN |
| LSTM | Long-Short Term Memory Network |
| mAP | Mean Average Precision |
| MKL | Multiple Kernel Learning |
| MLP | Multi-Layer Perceptron |
| MS-COCO | Microsoft Common Objects in Context |
| MT | Margin Transfer |
| NN | Nearest Neighbour (*Plus proche voisin*) |
| OHEM | Online Hard Example Mining |
| PPMI | People Playing Musical Instruments |
| PReLU | Parametric Rectified Linear Unit |
| RBM | Restricted Boltzmann Machine |
| ReLU | Rectified Linear Unit |

| | |
|---|---|
| R-FCN | Region-based Fully Convolutional Network |
| RNN | Recurrent Neural Network (*Réseau de neurones récurrent*) |
| RoI | Region of Interest (*Région d'intérêt*) |
| SGD | Stochastic Gradient Descent (*Descente de gradient stochastique*) |
| SIFT | Scale-Invariant Feature Transform |
| SPM | Spatial Pyramid Matching (*Pyramide spatiale*) |
| STANAG | Standardization Agreement (*Accord de normalisation*) |
| SURF | Speeded-Up Robust Features |
| SVM | Support Vector Machine (*Séparateur à Vaste Marge* ou *Machine à vecteurs de support*) |
| VLAD | Vector of Locally Aggregated Descriptors |
| VLAT | Vector of Locally Aggregated Tensors |
| YOLO | You Only Look Once |

# Chapitre 1

# Introduction

**Sommaire**



Depuis plusieurs années, la démocratisation des moyens de prise de vue comme les smartphones ou les appareils photo numériques couplée à l'importance toujours croissante des réseaux sociaux a conduit à l'explosion de la quantité de données visuelles disponibles sur Internet. Par exemple, le nombre de photos mises en ligne chaque jour s'élève à 2 millions sur Flickr [1], et à 350 millions sur Facebook [2]. En 2016, ce sont plus de 95 millions de photos et de vidéos qui sont mises en ligne sur Instagram chaque jour [3], et 300 heures de nouvelles vidéos chaque minute sur Youtube [4]. Cette quantité toujours plus importante de données visuelles sur le net confère chaque jour un intérêt grandissant au problème de l'annotation d'images.

La tâche de l'annotation d'images est un domaine particulièrement exploré car particulièrement difficile. En effet, il s'agit de parvenir à extraire des concepts très sémantiques sur les images à partir de leur contenu. Cependant, le contenu brut de ces images (*i.e.* les pixels) ne fournit pas directement l'information permettant de conclure à un ou plusieurs concepts sémantiques, comme des connaissances sur le type de scène observée ou les objets qui s'y trouvent. La figure 1.1 illustre ce problème, que Smeulders et al. (2000) qualifient de "fossé sémantique", ou *semantic gap*.

Pour aborder ce problème, deux communautés se sont particulièrement intéressées à ce sujet. D'une part, le domaine de la vision par ordinateur (ou *Computer Vision*) s'est attaché à proposer différentes manières de représenter les images, afin d'aborder la difficile tâche de la compréhension des images. D'autre part, la communauté de l'apprentissage automatique (ou *Machine Learning*) a introduit de nombreux modèles statistiques pour lier des représentations d'images aux concepts sémantiques. Les travaux présentés dans ce manuscrit sont à la croisée de ces deux domaines, puisqu'on explore aussi bien des

1. https://www.flickr.com/photos/franckmichel/6855169886
2. http://www.businessinsider.com/facebook-350-million-photos-each-day-2013-9?IR=T
3. http://www.instagram.com/press/
4. http://www.statisticbrain.com/youtube-statistics/

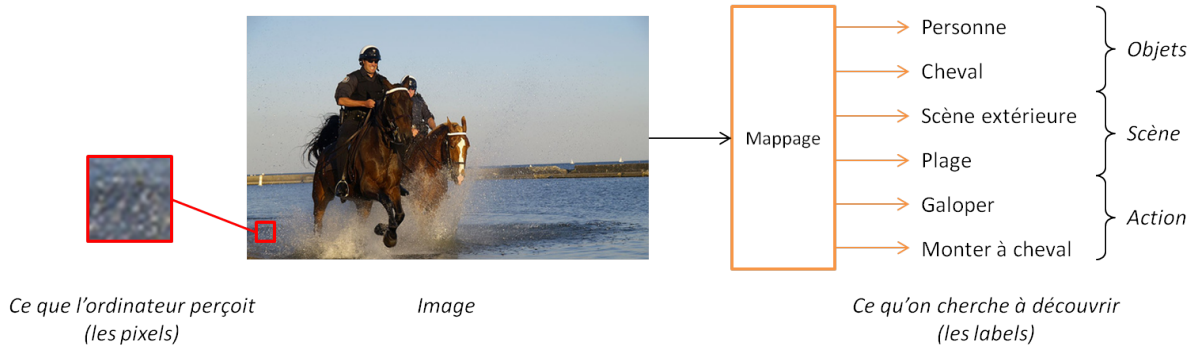



FIGURE 1.1 – Illustration du problème de l'annotation d'images. Cette discipline vise à combler le fossé sémantique qui sépare l'information brute perçue par l'ordinateur (les pixels) des concepts sémantiques qu'on cherche à reconnaître (concepts liés aux objets, à la scène, aux actions...).

problématiques de représentation des images que des méthodes permettant de mieux généraliser l'apprentissage des concepts à partir des données.

Dans le cadre de cette thèse, on aborde le problème de la reconnaissance d'images pour des applications Thales. Dans ce contexte, on s'intéresse plus particulièrement à la classification d'images de faible résolution. On explore pour cela les méthodes d'apprentissage supervisé classiques, mais on considère également un cadre d'apprentissage plus riche. En particulier, on cherche à améliorer un classifieur opérant sur des images faiblement résolues en bénéficiant d'informations complémentaires à ces images.

## 1.1 Contexte académique

La classification d'images est un des grands défis relevés par les domaines de la vision par ordinateur et de l'apprentissage automatique. Pour traiter ce problème, l'un des enjeux majeurs est de construire un vecteur représentant les images : cette étape est plutôt abordée par les travaux de vision par ordinateur. Ces représentations sont ensuite traitées par un classifieur, dont le but est de déterminer une fonction de décision, *i.e.* une frontière séparant les représentations des éléments des différentes classes recherchées. Durant les dernières décennies, plusieurs approches majeures ont été proposées par ces deux communautés.

**Apprentissage automatique : les réseaux de neurones dans les années 1980.** Adaptés des travaux de McCulloch and Pitts (1943) sur la modélisation mathématique des neurones ainsi que du perceptron de Rosenblatt (1958), les réseaux de neurones ont fait l'objet de nombreux travaux en apprentissage automatique dans les années 1980. Un des principaux intérêts de ces méthodes est que le perceptron multi-couche est un approximateur de toute fonction continue (Hornik, 1991). De plus, comme les paramètres des réseaux de neurones sont appris sur les données, le modèle qui en résulte est adapté au problème abordé.

Ces modèles présentent cependant deux particularités majeures, qui ont atténué leur succès au profit d'autres méthodes dans les années 1990. D'une part, l'apprentissage des paramètres de ces réseaux nécessite beaucoup d'images, ainsi que des ressources de calcul importantes, qui n'étaient alors pas facilement accessibles. D'autre part, l'optimisation de

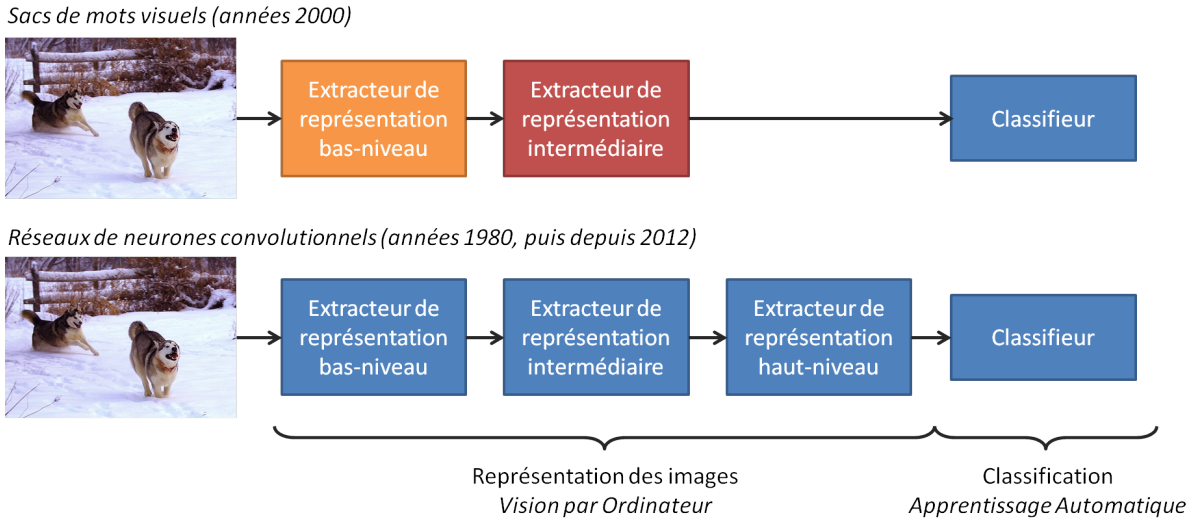



FIGURE 1.2 – Illustration des différentes familles majeures pour la reconnaissance des images. On représente en orange les étapes définies manuellement (ou *hand-crafted*, sans apprentissage), en rouge les étapes apprises de manière non-supervisée, et en bleu les étapes apprises de manière supervisée. Les méthodes de représentation des images sont devenues plus hiérarchisées, permettant de représenter les images à des niveaux plus sémantiques. Les étapes par apprentissage permettent d'extraire des représentations plus adaptées aux données traitées.

ces méthodes est non-convexe, c'est pourquoi ces approches souffraient d'une image de "boîte noire". La communauté d'apprentissage automatique s'est plus particulièrement intéressée à des modèles moins profonds (*i.e.* des modèles dits *shallow*) comme le SVM (Cortes and Vapnik, 1995), plus intuitif et dont l'optimisation convexe a permis de mettre en évidence les propriétés de convergence vers le minimum global de la fonction.

**Vision par ordinateur : les modèles par sacs de mots visuels dans les années 2000.** Dans le domaine de la vision par ordinateur, une famille de méthodes de représentation des images s'impose comme une approche majeure dans les années 2000 : il s'agit de la représentation par sacs de mots visuels. Cette méthode introduite par Ma and Manjunath (1999) dans le domaine de la recherche d'images à la fin des années 1990, puis par Sivic and Zisserman (2003) en vision par ordinateur en 2003, à partir des travaux en recherche d'information textuelle (Baeza-Yates and Ribeiro-Neto, 1999), est un processus de quatre étapes successives : extraction de descripteurs locaux, codage par rapport à un dictionnaire, agrégation, puis classification.

Un ensemble de descripteurs locaux, basés sur des critères assez intuitifs tels que les gradients orientés (Lowe, 1999; Dalal and Triggs, 2005) ou des variations locales d'intensité (Pietikäinen et al., 2011), sont d'abord extraits des images. Ces représentations dites de *bas-niveau* permettent d'introduire de l'invariance à plusieurs transformations géométriques, mais ne permettent pas un niveau très sémantique de représentation des images, c'est-à-dire qu'ils ne permettent pas de représenter des concepts abstraits présents dans l'image, comme par exemple le type de scène observée (*cf.* figure 1.1). Pour répondre à ce problème, ces descripteurs sont encodés par rapport à un dictionnaire de mots visuels

appris de manière non-supervisée sur les données. L'introduction de cette étape permet de représenter les images à un niveau sémantique supérieur (niveau *intermédiaire*).

Ces modèles et leurs extensions, comme les Fisher Vectors, se sont rapidement imposés comme les principales méthodes de l'état de l'art pour la représentation des images. Cependant, le processus de calcul de ces représentations est relativement plat, puisqu'il ne comprend en général qu'une seule couche, ce qui limite le pouvoir de représentation sémantique du descripteur. De plus, le processus est prédéfini, ce qui limite l'adaptabilité des représentations à la tâche abordée.

**À la croisée des deux communautés : les réseaux de neurones convolutionnels.** Les réseaux de neurones, plutôt développés par la communauté d'apprentissage automatique dans les années 1980, ont cependant été adaptés plus spécifiquement à des tâches de vision par ordinateur dans les travaux de Fukushima (1980) dans les années 1980, puis ceux de LeCun et al. (1989) dans les années 1990. Afin de prendre en compte la structure des images dans le calcul des représentations internes du réseau, ces travaux proposent les réseaux de neurones convolutionnels. Bien qu'ils soient appliqués avec succès sur des tâches de reconnaissance visuelle (LeCun et al., 1998), ces modèles ne remportent que peu de popularité du fait de leur image de boîte noire et du manque de moyens matériels nécessaires à leur apprentissage, évoqués précédemment.

Avec la démocratisation de ressources efficaces comme les GPU et la mise à disposition de larges bases d'images annotées comme ImageNet (Deng et al., 2009) ou MS-COCO (Lin et al., 2014), les réseaux de neurones convolutionnels ont cependant su s'imposer en 2012 (Krizhevsky et al., 2012) comme la nouvelle méthode de l'état de l'art pour la reconnaissance visuelle. Dans ces modèles, l'information extraite des images est très hiérarchisée, et la profondeur de ces structures permet de calculer des représentations à un niveau sémantique bien plus élevé que dans les méthodes de type sacs de mots visuels (représentation *haut-niveau*). L'originalité de ces modèles vient également du fait que toutes les étapes intermédiaires sont apprises sur les données, ce qui en fait un modèle très adaptable aux différentes tâches abordées. On met en regard cette méthode par rapport aux représentations par sacs de mots visuels dans la figure 1.2. Aujourd'hui, toutes les tâches de reconnaissance visuelle ont été traitées avec succès par les réseaux de neurones convolutionnels, ce qui fait d'eux une méthode majeure de l'état de l'art pour les tâches de vision par ordinateur.

**Apprentissage avec information privilégiée.** Introduit par Vapnik and Vashist (2009), l'apprentissage avec information privilégiée est un domaine de l'apprentissage automatique permettant de prendre en compte, durant l'entraînement d'un classifieur, des données complémentaires aux exemples d'apprentissage. Ces données complémentaires, dites *privilégiées*, sont indisponibles pour les exemples de test, et peuvent être de différentes natures, comme par exemple une description textuelle des images (Sharmanska et al., 2013) ou la structure géométrique d'un élément dans l'espace (Vapnik and Vashist, 2009). Bien qu'émanant du domaine de l'apprentissage automatique, cette approche a été appliquée sur plusieurs tâches de vision par ordinateur, comme la reconnaissance d'images (Sharmanska et al., 2014).

À mi-chemin entre les techniques de fusion d'informations hétérogènes telles que la fusion précoce de Snoek et al. (2005), et de sélection des exemples comme le curriculum

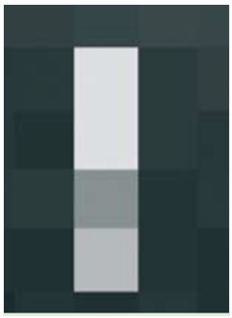

FIGURE 1.3 – Illustration de la nacelle AREOS, produite par Thales Optronique. Cette nacelle est placée sous l'avion, et est dédiée à la reconnaissance air-sol à longue portée, basée sur l'imagerie proche infra-rouge et infra-rouge.

learning de Bengio et al. (2009), cette approche permet d'utiliser l'information privilégiée comme un indicateur de la difficulté des exemples, qui est ensuite intégré au classifieur appris sur les images ciblées (*i.e.* le type d'exemples disponibles durant l'apprentissage et le test). Différentes méthodes ont été proposées pour intégrer cet indice de difficulté au sein du classifieur, comme par exemple en pondérant les exemples dans l'espace ciblé (Sharmanska et al., 2014) ou en imposant une ressemblance entre les scores des deux représentations (Lopez-Paz et al., 2016). Ces méthodes définissent un cadre d'apprentissage permettant de bénéficier de cette information complémentaire durant l'apprentissage du classifieur.

## 1.2 Contexte opérationnel

Les travaux présentés dans ce manuscrit sont le fruit d'une collaboration entre le Laboratoire d'Informatique de Paris 6 (LIP6, Sorbonne Universités) et Thales Optronique S.A.S.. Dans ce cadre, les travaux de recherche ont été guidés par certaines des motivations industrielles de Thales Optronique, entreprise spécialisée dans les systèmes optroniques pour la défense. On s'intéresse ici en particulier aux systèmes optroniques aéroportés. Thales Optronique propose plusieurs produits pour la reconnaissance aéroportée, notamment la nacelle AREOS (*Airborne Reconnaissance Observation System*), présentée dans la figure 1.3. AREOS s'appuie sur l'imagerie proche infra-rouge et infra-rouge pour la reconnaissance air-sol sur une longue portée, de jour et de nuit.

La reconnaissance automatique d'objets sur AREOS a deux apports opérationnels. D'une part, durant le vol, la nacelle acquiert des images couvrant des surfaces particulièrement importantes, qui sont ensuite analysées par des photo-interprètes au sol. La transmission des données vers le sol doit donc être efficace afin de réduire au maximum le temps qui s'écoule entre l'acquisition de l'image et la prise de décision. En mettant en lumière automatiquement les zones contenant certains types d'objets (par exemple des véhicules), les algorithmes de reconnaissance doivent permettre de transmettre un maximum d'information sur les zones d'intérêt opérationnel (*i.e.* avec une compression faible, voire sans compression)l, tout en ne transmettant qu'une image plus fortement



**Détection**
(ici 3,6 × 5 pixels)
*Quelque chose est là*

**Reconnaissance**
(ici 14,4 × 4 pixels)
*Il s'agit d'une personne*

**Identification**
(ici 28,8 × 8 pixels)
*Cette personne tient un fusil*

FIGURE 1.4 – Illustration de la taille minimale pour différentes tâches de reconnaissance visuelle par l'humain (STANAG 3769). La reconnaissance par un humain est précise à 100% sur une cible d'au moins 24 pixels d'envergure. Ici, une zone de 14,4 × 4 pixels suffisent pour reconnaître qu'il s'agit d'une personne.

compressée des autres zones. Cela permettrait de réduire le temps global de transmission des données vers le sol, et donc le délai nécessaire à la prise de décision. D'autre part, ce système de reconnaissance doit permettre d'aider le pilote à analyser la scène observée en temps réel à bord de l'appareil, par exemple en signalant une zone d'intérêt au pilote afin que la décision soit prise de s'intéresser ou non à cette zone. La sollicitation de l'attention du pilote ne doit cependant pas devenir trop importante. Aussi, suivant le contexte dans lequel évolue l'avion, le niveau d'alerte doit s'adapter. Par exemple, dans un contexte urbain, une voiture civile ne doit pas être signalée au pilote, tandis qu'un véhicule militaire peut être considéré comme digne d'intérêt. Dans un contexte désertique, tout type de véhicule peut en revanche être signalé au pilote. Il faut donc être capable de différencier les véhicules pour prendre la décision d'avertir l'utilisateur.

Pour des raisons de furtivité et de sécurité, l'appareil ne peut pas s'approcher des zones d'intérêt, et acquiert donc les images de la scène observée à une certaine distance. La résolution au sol des images acquises est donc assez faible sur des petits objets comme les véhicules. Cependant, le traitement global de ces images est semi-automatique, puisque c'est toujours un opérateur humain qui doit prendre une décision sur la marche à suivre dans une situation donnée. Pour ce faire, la résolution des images doit permettre de conserver le minimum d'information nécessaire à la prise de décision. Le STANAG 3769 (*Standardization Agreement*) de l'OTAN (STANAG 3769) définit la résolution minimale pour qu'un humain parvienne à accomplir certaines tâches visuelles avec une certaine probabilité. Notamment, pour une probabilité de 100%, la détection d'un objet (*i.e.* déterminer qu'une zone ressort du bruit) nécessite 6 pixels d'envergure. La reconnaissance (*i.e.* déterminer de quel type d'objet il s'agit) requiert 24 pixels d'envergure. Enfin, l'identification (*i.e.* parvenir à comprendre des détails fins) à 100% nécessite à 48 pixels d'envergure. Ces différentes étapes sont illustrées dans la figure 1.4. Suivant ces informations, on s'intéressera principalement à la reconnaissance sur des images de taille $32 \times 32$ pixels.

Suivant les produits concernés et les contraintes techniques, ces différents niveaux de discrimination se reflètent plus ou moins fortement dans les étapes de la chaîne de traitement. Dans notre cas, on peut résumer schématiquement la chaîne de traitement

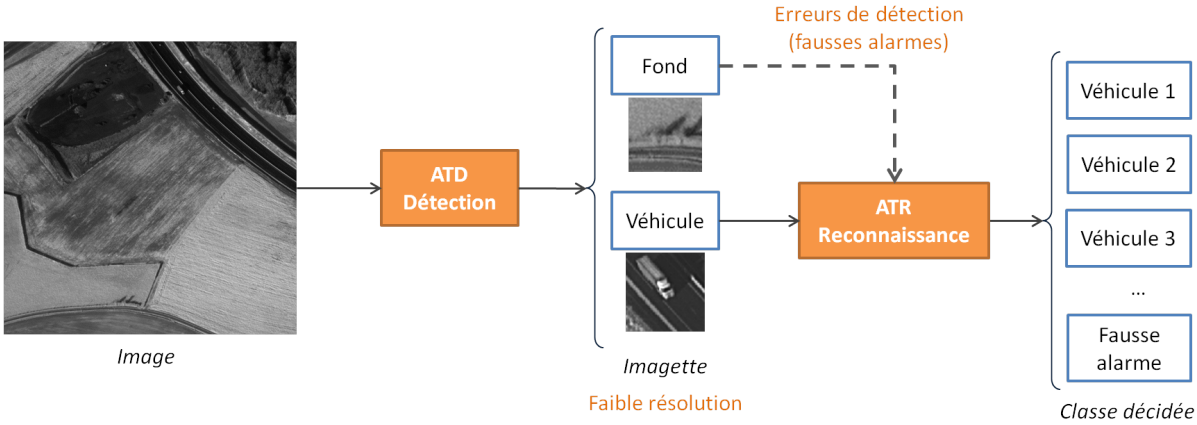



FIGURE 1.5 – Schéma du fonctionnement d'un système aéroporté pour la reconnaissance de véhicules au sol.

de la manière représentée dans la figure 1.5. Une image globale de la scène observée est d'abord acquise par les capteurs. Cette image est acquise à une distance assez grande, de fait elle couvre une large surface au sol, sur laquelle le nombre de véhicules présents peut être assez important. Un premier module, l'ATD (*Automatic Target Detection*), permet d'extraire des imagettes contenant potentiellement des véhicules sur cette image globale. Les imagettes ainsi extraites sont ensuite traitées par le module ATR, ou *Automatic Target Recognition*, dont le but est de reconnaître plus précisément le véhicule présent dans l'image. Ainsi, les images traitées par le module ATR sont supposées être centrées sur un véhicule. Afin d'être sûr de ne pas éliminer d'emblée une zone d'intérêt, le module ATD privilégie le nombre de détections au détriment de la précision. Il arrive donc que de fausses détections soient transmises au module ATR : ainsi, des imagettes de fond peuvent arriver en entrée du module ATR. Dans ces cas-là, il faut que le modèle de classification appris pour l'ATR sache discriminer également les images sans véhicules, *i.e.* les fausses alarmes.

Dans nos travaux, on s'intéresse uniquement à modéliser l'ATR. De plus, on se place dans le cas où l'ATD est parfait, *i.e.* on ne s'intéresse qu'à des images contenant un objet. On traite cependant le cas des fausses alarmes dans plusieurs expériences, pour étudier leur impact sur les performances des modèles proposés.

Ce contexte opérationnel de reconnaissance de véhicules apporte donc son lot de contraintes techniques aux travaux de recherche proposés. D'abord, les avions ne peuvent pas se rapprocher indéfiniment d'un véhicule repéré, si bien que le système embarqué doit pouvoir reconnaître des véhicules assez petits dans l'image : il faut donc parvenir à traiter des images faiblement résolues. De plus, comme il s'agit d'un système embarqué, la question des ressources de calcul disponibles influence le type de représentations d'images considérées. Dans ce contexte, on privilégiera naturellement des approches peu coûteuses. Ensuite, l'obtention d'images réelles annotées est très coûteuse, pour de multiples raisons. D'une part, il faut pour cela prévoir plusieurs vols, ce qui représente une dépense non négligeable. Durant chaque vol, la variabilité des véhicules observés peut être limitée, de plus il est difficile d'observer toutes les cibles d'intérêt opérationnel. D'autre part, une fois acquises, ces images doivent être annotées manuellement, ce qui représente un pro-

cessus assez coûteux. De fait, le nombre d'images réelles annotées pour l'apprentissage est limité, ce qui peut poser problème lors de l'apprentissage de modèles complexes comme les réseaux de neurones convolutionnels.

Dans cette thèse, afin de disposer d'un nombre suffisant de données pour alimenter un réseau de neurones profond, Thales Optronique a mis en place une base d'images de véhicules simulés, afin de traiter le problème du faible nombre d'images d'apprentissage réelles annotées. Le principe de ces images repose sur la superposition d'une image de synthèse d'un véhicule calculé à partir d'un modèle 3D texturé, avec des images de fond réelles. Une description plus détaillée de cette base de données est disponible en annexe A. Grâce à la simulation, des informations complémentaires aux images peuvent être disponibles, comme par exemple les conditions de la prise de vue. On peut donc s'appuyer sur des informations de vol complémentaires aux images, ou encore sur des versions haute-résolution des images d'apprentissage. Dans le cadre de nos travaux, la base de données simulées nous permet de disposer, pour chaque image d'apprentissage de faible résolution, de sa version hautement résolue. Les travaux présentés dans ce manuscrit s'attacheront donc à aborder le problème de la classification d'images faiblement résolues dans un contexte de système embarqué, et avec les solutions telles que la mise à disposition de données complémentaires durant la phase d'apprentissage. Dans ce contexte, on cherche d'une part à étudier l'adéquation des méthodes majeures de l'état de l'art pour ce problème, et d'autre part à proposer des méthodes permettant de prendre en compte ces données complémentaires.

## 1.3 Objectifs et contributions

Les travaux présentés dans ce manuscrit sont le fruit de l'harmonisation des problématiques de recherche en classification d'images avec les contraintes industrielles posées par le contexte de la reconnaissance d'objets lointains au sein d'un système aéroporté. On s'intéresse dans ce manuscrit à la reconnaissance d'images faiblement résolues. Par ailleurs, on se place dans un cadre où on dispose, en plus des images d'apprentissage de faible résolution, d'informations complémentaires à ces images.

D'un point de vue académique, on se place donc à la convergence entre les travaux menés par les deux communautés de vision par ordinateur et d'apprentissage automatique, d'une part en explorant plusieurs méthodes de reconnaissance d'images, et d'autre part en intégrant dans nos recherches le cadre de l'information privilégiée. D'un point de vue industriel, on cherche à répondre aux différentes contraintes posées, c'est-à-dire la faible résolution des images traitées, le contexte d'un système embarqué (qui mène à privilégier des solutions facilement implémentées), et l'exploitation de données complémentaires aux images disponibles durant l'apprentissage. Dans ce contexte, les contributions des travaux présentés dans ce manuscrit peuvent être catégorisées comme suit :

**Étude et adaptation de représentations profondes pour le problème posé.** D'abord, on cherche à connaître l'impact de la perte de résolution des images sur les performances des représentations de l'état de l'art en classification d'images, afin de déterminer leur adéquation pour la tâche abordée. Dans ces travaux, on propose donc une comparaison des méthodes majeures de représentation des images sur des ensembles d'images à différentes résolutions. On met ainsi en évidence l'intérêt des représentations profondes pour le

problème de la classification d'images faiblement résolues. On propose également une architecture profonde adaptée au contexte, qui peut être apprise directement sur les données ciblées (Chapitre 3).

**Intégration de l'information privilégiée dans les réseaux de neurones.** On propose ensuite deux modèles permettant d'intégrer l'information privilégiée dans les réseaux de neurones profonds. Notre premier modèle deep+, inspiré de l'approche Margin Transfer de Sharmanska et al. (2014), permet d'intégrer l'information privilégiée sous la forme d'une pondération des exemples ciblés, en focalisant l'apprentissage sur les exemples les plus représentatifs de leur classe (Chapitre 5). On propose ensuite un deuxième modèle, deep++, intégrant une condition de ressemblance entre les modèles appris respectivement sur les données ciblées et sur les données privilégiées (Chapitre 6). Nos deux approches offrent un cadre d'intégration de l'information privilégiée dans les réseaux de neurones profonds, permettant ainsi à toutes les représentations internes de bénéficier de cette information complémentaire.

**Evaluation empirique des méthodes proposées.** Enfin, on fournit une évaluation empirique de nos modèles, montrant qu'ils permettent d'améliorer les résultats de méthodes standard ainsi que des méthodes de l'état de l'art concurrentes sur plusieurs tâches de reconnaissance d'images variées, telles que la reconnaissance des chiffres manuscrits de MNIST (LeCun et al., 1998) ou l'apprentissage sur un ensemble contenant du bruit d'annotation sur UPMC-Food-101 (Wang et al., 2015b). On fournit également des résultats améliorant les performances des méthodes de l'état de l'art testées sur FGVC-Aircraft (Maji et al., 2013), une base d'images orientée grain-fin. Sur ces bases de données, on fournit des résultats quantitatifs ainsi qu'une étude approfondie des différents aspects de nos modèles. Pour finir, on reporte les résultats sur une base de données simulées propre à Thales, montrant ainsi l'intérêt de nos méthodes dans un contexte plus opérationnel.

Nos travaux ont fait l'objet des publications suivantes :

**Marion Chevalier**, Nicolas Thome, Matthieu Cord, Jérôme Fournier, Gilles Hénaff, and Elodie Dusch. Low-resolution convolutional neural network for Automatic Target Recognition. *7th International Symposium on Optronics in Defence and Security*, Paris, France, 2016. (Chevalier et al., 2016)

**Marion Chevalier**, Nicolas Thome, Matthieu Cord, Jérôme Fournier, Gilles Hénaff, and Elodie Dusch. LR-CNN for fine-grained classification with varying resolution. *IEEE International Conference on Image Processing*, Québec City, Canada, 2015. (Chevalier et al., 2015)

## 1.4 Plan du manuscrit

Pour répondre aux questions soulevées précédemment, on présente les travaux menés suivant l'enchaînement décrit ci-après.

**Chapitre 2 - Modèles et apprentissage pour la reconnaissance visuelle.** Dans ce chapitre, on présente les motivations et les principes de différents modèles majeurs de l'état de l'art pour la représentation des images. On introduit notamment les Fisher Vectors, qui étendent les méthodes par sacs de mots visuels, une approche prédéfinie manuellement basée sur l'encodage de descripteurs locaux par rapport à un dictionnaire appris de manière non-supervisée sur les images d'apprentissage. On met en perspective ces méthodes avec les réseaux de neurones convolutionnels profonds, une structure dont les poids sont tous appris de manière supervisée et qui permettent une représentation des images de niveau sémantique très élevé.

**Chapitre 3 - Etude des représentations pour la classification à grain-fin d'images faiblement résolues.** Afin d'examiner l'adéquation des méthodes décrites dans le chapitre 2 par rapport à notre contexte de faible résolution, on évalue dans ce chapitre l'impact de la résolution des images sur ces méthodes de représentation dans le contexte de la classification axée grain-fin. On évalue notamment plusieurs aspects complémentaires, notamment l'influence de l'intégration naïve d'images fortement résolues durant l'apprentissage. On propose par ailleurs LR-CNN, un réseau de neurones profond adapté à la reconnaissance des petites images, dont on montre expérimentalement l'intérêt pour le contexte abordé.

**Chapitre 4 - Méthodes d'apprentissage avec information privilégiée pour la classification d'images.** On présente dans ce chapitre le contexte de l'apprentissage avec information privilégiée, qui permet de prendre en compte des données supplémentaires durant l'apprentissage. On décrit notamment plusieurs méthodes phares de ce domaine basées sur différentes approches et par rapport auxquelles on positionne nos travaux.

**Chapitre 5 - Deep+.** Afin de prendre en compte les images fortement résolues disponibles durant l'apprentissage, on propose deep+, un modèle permettant d'intégrer l'information privilégiée dans l'apprentissage d'un réseau de neurones profond. Notre approche permet d'utiliser l'information privilégiée comme une pondération des exemples, attribuant un poids plus important aux exemples les plus faciles à reconnaître et atténuant l'influence des exemples les plus difficiles.

**Chapitre 6 - Deep++.** Dans ce chapitre, on présente le modèle deep++, une extension du modèle précédent basée sur une pénalisation relative entre les exemples et leur information privilégiée. Outre la pondération des exemples, on introduit une deuxième façon d'intégrer l'information privilégiée permettant de prendre en compte une information relative entre la classification dans l'espace privilégiée et dans l'espace ciblé.

**Chapitre 7 - Conclusions et perspectives.** Pour finir, on présente un résumé des contributions scientifiques de ces travaux, ainsi qu'une discussion sur les perspectives offertes par ce manuscrit.

# Chapitre 2

# Modèles et apprentissage pour la reconnaissance visuelle

**Sommaire**



Dans ce manuscrit, on aborde la tâche de la reconnaissance d'images dans un contexte particulier, influencé par les contraintes industrielles de Thales Optronique : les images d'intérêt sont faiblement résolues. Pour traiter ce problème, on présente ici différents modèles majeurs de l'état de l'art pour la représentation et la classification des images, en mettant en lumière les motivations de ces méthodes.

Diverses approches ont été proposées pour la représentation des images, s'étalonnant depuis des représentations locales, qui représentent l'image d'entrée à un faible niveau d'abstraction, jusqu'à des représentations plus complexes, qui parviennent à refléter un niveau élevé d'abstraction. Issues du domaine de la recherche d'information, les représentations de type sacs de mots visuels (ou BoW pour *Bag of visual Words*) et plus récemment les Fisher Vectors (FV) sont des méthodes importantes de l'état de l'art pour la représentation des images. Ces modèles se basent sur l'encodage de descripteurs locaux par rapport à un dictionnaire appris de manière non-supervisée sur les données d'apprentissage, et permettent d'encoder une certaine invariance à des transformations locales, ce qui en fait des représentations intermédiaires assez robustes. L'utilisation de ces représentations en conjonction avec le Séparateur à Vaste Marge (ou SVM) a permis d'atteindre de très bons résultats en classification à la fin des années 2000.

On met en perspective ces descripteurs, représentant les images à un niveau d'abstraction relativement peu profond, avec les réseaux de neurones, constitués de nombreuses couches d'extraction de caractéristiques consécutives. Cette structure profonde permet de fortement hiérarchiser l'information extraite de l'image d'entrée, et donc de la représenter à un niveau d'abstraction bien supérieur. Contrairement aux représentations par sacs de mots, la fonction calculée par le réseau de neurones sur une image d'entrée est apprise de bout en bout sur les données d'apprentissage. Cette caractéristique en fait une méthode très adaptable aux différents contextes ciblés. Lorsque l'architecture contient trop de paramètres à entraîner par rapport au nombre d'exemples d'apprentissage disponibles, afin d'éviter le sur-apprentissage, une technique consiste à transférer les paramètres appris sur une autre base plus large, *i.e.* contenant plus d'exemples. Autrement dit, un réseau dont les paramètres ont été appris sur une base externe est utilisé comme un extracteur de représentation sur les images d'intérêt, puis un classifieur est appris sur ces représentations. Ces deux approches ont montré d'excellentes performances en classification, notamment depuis la compétition ILSVRC 2012 (*ImageNet Large Scale Visual Recognition Challenge*), remportée par un réseau de neurones convolutionnel améliorant de près de 10% les meilleures performances des années passées. Ces méthodes détiennent aujourd'hui l'état de l'art sur la très large majorité des tâches de vision par ordinateur.

## 2.1 Représentations intermédiaires

### 2.1.1 Représentation par sacs de mots visuels

Les représentations en sacs de mots ont d'abord été proposées par Baeza-Yates and Ribeiro-Neto (1999) pour les travaux de recherche d'information textuelle. Suivant cette méthode, un document est représenté comme la fréquence des mots du dictionnaire y apparaissant. Dans le cas de l'image, l'utilisation d'une telle représentation pose un certain nombre de questions. D'une part, lorsqu'il est aisé d'extraire des mots dans un texte, l'équivalent pour l'image n'est pas immédiat : que peut-on extraire d'une image ? D'autre part, il n'existe pas de dictionnaire *a priori* pour encoder ces mots, contrairement au cas du texte : que peut-on alors quantifier sur chaque image ?

Proposé notamment par la *NeTra toolbox* de Ma and Manjunath (1999) puis popularisé par les travaux de Sivic and Zisserman (2003), le modèle de représentation des images par sacs de mots visuels (ou BoW) est une représentation majeure de l'état de l'art pour la reconnaissance visuelle. Basé sur l'approche des sacs de mots pour la recherche d'information textuelle, le calcul de la représentation BoW d'une image est un processus en quatre étapes majeures, représentées sur la figure 2.1 : extraction des descripteurs locaux, encodage par rapport à un dictionnaire visuel, agrégation, puis classification. Dans la littérature, de nombreux travaux ont questionné et apporté des réponses sur les manières de traiter chacune de ces étapes. Dans cette partie, on structure les problèmes de recherche en vision par ordinateur pour ce type de représentations.

#### Extraction de descripteurs locaux

D'abord, des représentations locales sont extraites de l'image d'entrée. L'enjeu de cette première étape du processus de représentation des images consiste à extraire des descripteurs locaux de manière à instaurer de l'invariance à plusieurs types de transformations,

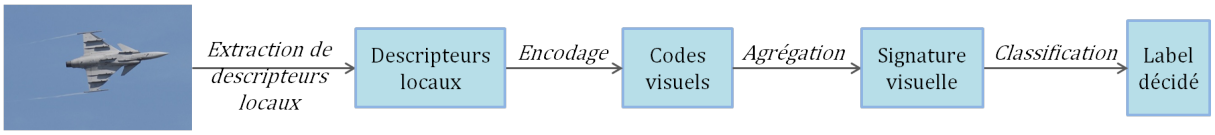



FIGURE 2.1 – Schéma des différentes étapes de la représentation BoW.

telles que des rotations ou des translations. Dans ce but, de nombreux types de représentations locales ont été proposés, se basant sur différentes approches. En particulier, les descripteurs de type SIFT (Scale-Invariant Feature Transform) (Lowe, 1999), SURF (Speeded Up Robust Features) (Bay et al., 2006), HOG (Histogram of Oriented Gradients) (Dalal and Triggs, 2005) et GIST (Oliva and Torralba, 2001) s'appuient sur les gradients orientés, ce qui permet d'introduire une forme d'invariance en rotation. D'autres descripteurs comme LBP (Local Binary Pattern) (Pietikäinen et al., 2011) et ses extensions (Zhang et al., 2005; Liao et al., 2007) encodent plutôt des variations locales d'intensité, ce qui permet d'introduire une forme d'invariance à des changements de luminosité uniformes sur l'image. Pour nos travaux, on considère le descripteur SIFT (Lowe, 1999), qui est le plus couramment utilisé dans les représentations de type BoW.

Ces descripteurs locaux peuvent être extraits de différentes manières au sein d'une image. Une méthode très simple consiste à échantillonner aléatoirement les localisations et tailles des filtres appliqués sur l'image. Une autre approche basée sur la détection de points d'intérêt permet de n'extraire des descripteurs locaux qu'autour de ces points spécifiques. Le plus souvent, ces descripteurs locaux sont néanmoins extraits de manière dense sur toute l'image et à différentes échelles. Autrement dit, des filtres de différentes tailles sont appliqués sur la totalité de l'image selon une grille préétablie. Cette méthode d'échantillonnage permet de décrire l'image d'entrée en totalité et à différents niveaux de détails. Cette méthode est notamment très utilisée pour les modèles de type BoW (Chatfield et al., 2011; Law et al., 2012), puisqu'elle permet d'assurer une description exhaustive de l'image d'entrée, et se révèle expérimentalement plus efficace que les échantillonnages décrits auparavant (Jurie and Triggs, 2005).

#### Apprentissage du dictionnaire

Afin de représenter la distribution des descripteurs locaux, un dictionnaire est appris sur les descripteurs locaux extraits de l'ensemble des images d'apprentissage. Un algorithme très couramment utilisé est l'algorithme des k-moyennes (ou *k-means*) (Lloyd, 1982). Dans cette méthode, $N_{clusters}$ cellules sont apprises de manière non-supervisée pour partitionner les descripteurs locaux extraits de toutes les images d'apprentissage, $N_{clusters}$ étant un hyper-paramètre. D'autres méthodes ont été proposées pour apprendre ce dictionnaire, notamment de manière supervisée (Goh et al., 2012), mais le k-means reste la méthode la plus utilisée dans les BoW. Les centres des clusters $w_j$ ainsi formés sont appelés les mots visuels, et constituent le dictionnaire.

#### Codage

Les descripteurs locaux extraits d'une image d'entrée sont ensuite encodés par rapport au dictionnaire appris. Ce processus est illustré par la figure 2.2. L'une des méthodes de codage les plus répandues est l'assignement dur (ou *hard coding*), dans lequel pour chaque

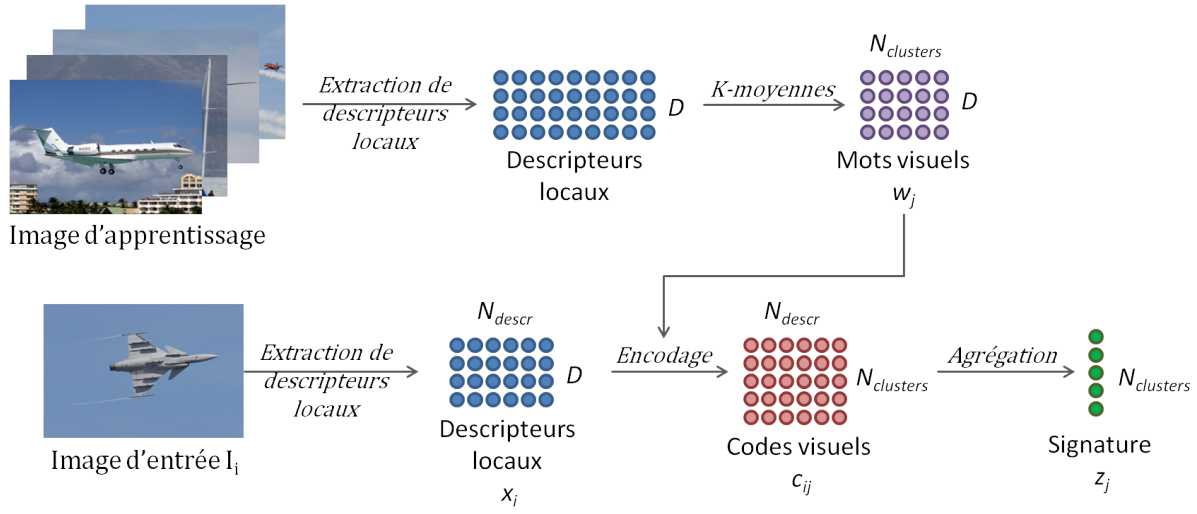



FIGURE 2.2 – Représentations matricielles des étapes de calcul d'un BoW.

descripteur local, on associe la valeur 1 pour le mot visuel le plus proche (au sens d'une distance prédéfinie, généralement la distance euclidienne), et 0 pour tous les autres mots visuels. Autrement dit, pour chaque descripteur local $x_i$, le code visuel $c_{ij}$ associé au mot visuel $w_j$ est 1 si $j = \arg\min_{j'=1..N_{clusters}} \|x_i - w_{j'}\|^2$, et 0 sinon.

D'autres méthodes de codage ont été proposées pour réduire les erreurs de quantification et introduire une meilleure mesure de l'ambiguïté entre les mots visuels. L'assignement souple, ou *soft coding* (Liu et al., 2011), consiste à associer à chaque code visuel $c_{ij}$ une mesure de la distance entre le descripteur $x_i$ et le mot visuel $w_j$. Les codes visuels $c_{ij}$ prennent ainsi des valeurs réelles comprises entre 0 et 1.

Enfin, plusieurs travaux ont proposé des méthodes d'encodage vectoriel, *i.e.* pour chaque mot de dictionnaire $w_j$, on n'associe plus une valeur pour chaque descripteur, mais un vecteur résumant la *distribution* des descripteurs autour de ce mot. Par exemple, les modèles VLAD (pour *Vector of Locally Aggregated Descriptors*) (Jégou et al., 2010) et leur extension VLAT (*Vectors of Locally Aggregated Tensors*) (Negrel et al., 2012a,b), ainsi que les Super-Vector (Zhou et al., 2010) encodent, pour chacun des mots visuels $w_j$, la somme des différences de vecteurs $x_{V_j(i)} - w_j$ où les $x_{V_j(i)}$ sont les descripteurs pour lesquels le mot visuel le plus proche est $w_j$. La signature $z$ de l'image est alors de dimension $N_{clusters} \times D$, où $D$ est la dimension des descripteurs locaux. De même, (Perronnin and Dance, 2007; Perronnin et al., 2010) ont également introduit les Fisher Vectors, qui permettent de prendre en compte les premier et second ordres de statistiques de la distribution des descripteurs, et sont développés dans la partie 2.1.2.

### Agrégation

Afin de résumer l'information contenue dans la matrice des $c_{ij}$, les codes visuels sont agrégés pour former la signature de l'image, un vecteur $z$ de taille fixée. Ce processus est décrit par la dernière étape de la figure 2.2. Un des enjeux majeurs de cette dernière étape consiste à trouver un bon compromis entre une représentation invariante et une représentation spécifique. En effet, si la représentation est trop spécifique, elle variera trop

en fonction de transformations appliquées sur l'objet (translations dans l'image, changement de point de vue, *etc*). En revanche, une représentation trop invariante perdra trop d'information sur l'image d'intérêt et donc son pouvoir de discrimination des différents objets.

Plusieurs méthodes courantes privilégient plutôt l'invariance. C'est le cas notamment du *sum pooling*, une méthode très largement répandue qui consiste à sommer les valeurs associées à un même mot du dictionnaire : $z_j = \sum_{i=1}^{N_{descr}} c_{ij}$. D'autres approches similaires ont été explorées, comme par exemple la méthode *average pooling*, qui moyenne les valeurs au lieu de simplement les sommer, ou le *max pooling*, où on ne conserve que la composante la plus élevée. Ces approches se sont révélées très populaires pour les BoW du fait de leur simplicité, cependant elles ne permettent de prendre en compte que le premier ordre de statistiques de la distribution des descripteurs locaux.

Pour répondre à ce problème, d'autres méthodes plus récentes ont cherché à étendre l'encodage à d'autres ordres de statistiques. Par exemple, le modèle BossaNova (Avila et al., 2011, 2013) propose de calculer, pour chaque mot visuel, l'histogramme des distances aux descripteurs qui lui sont associés. Ainsi, la matrice des codes visuels $c_{ij}$ de taille $N_{clusters} \times N_{descr}$ contient la distance entre chaque descripteur local et chaque mot visuel (*soft coding*), ce qui permet de calculer un histogramme des distances de $N_{bins}$ autour de chaque mot visuel. Ainsi, la signature $z$ de l'image a une taille de $N_{clusters} \times N_{bins}$.

Un problème majeur de toutes ces méthodes réside dans le fait qu'elles perdent toute information spatiale sur l'image. La volonté de construire une représentation invariante aux translations dans l'image fait perdre ici une information de localisation dans l'image trop importante. Une solution à ce problème a été proposée par Lazebnik et al. (2006). Dans leur méthode Pyramide Spatiale (ou *Spatial Pyramid Matching*, soit SPM), l'image est découpée en $M$ zones spatiales, se recouvrant ou non, et ayant différentes échelles. Dans chacune de ces zones, on n'agrège que les codes visuels $c_{ij}$ émanant des descripteurs locaux $x_i$ présents dans la zone en question. La signature $z$ représentant l'image est ainsi la concaténation des $M$ vecteurs $z_m$, représentant chacun une région $m$. $z$ a donc une taille $M \times N_{clusters}$. Sur la figure 2.3, on représente le cas d'un SPM d'échelles $1 \times 1 + 2 \times 2 + 3 \times 3$. Comme illustré sur cette figure, cette méthode permet d'encoder des informations à différents niveau de granularité de l'image, allant d'un point de vue global ($1 \times 1$) à des détails plus précis (par exemple la tête ou les oreilles du chat à l'échelle $3 \times 3$).

Une étape importante du calcul des représentations BoW sur les images est la normalisation des vecteurs. En effet, il a été prouvé expérimentalement que les classifieurs pouvaient être sensibles à cette étape. Par exemple, la normalisation $L_2$ des signatures de type BoW est souvent utilisée car elle permet d'obtenir de bonnes performances avec un classifieur linéaire (Chatfield et al., 2011; Perronnin et al., 2010).

### 2.1.2 Représentation Fisher Vectors

Introduits par Perronnin et al. (Perronnin and Dance, 2007), les Fisher Vectors (FV) sont une représentation importante en vision par ordinateur. Cette méthode de représentation a notamment remporté la compétition ILSVRC de classification d'images en 2011 (Russakovsky et al., 2015), et a prouvé son efficacité dans les contextes à grain-fin en remportant la première place de la compétition Fine-Grained classification challenge 2013 (FGComp13). On peut même souligner que les FV avaient alors battu les méthodes utilisant les réseaux de neurones ayant participé à cette compétition, bien qu'aujourd'hui

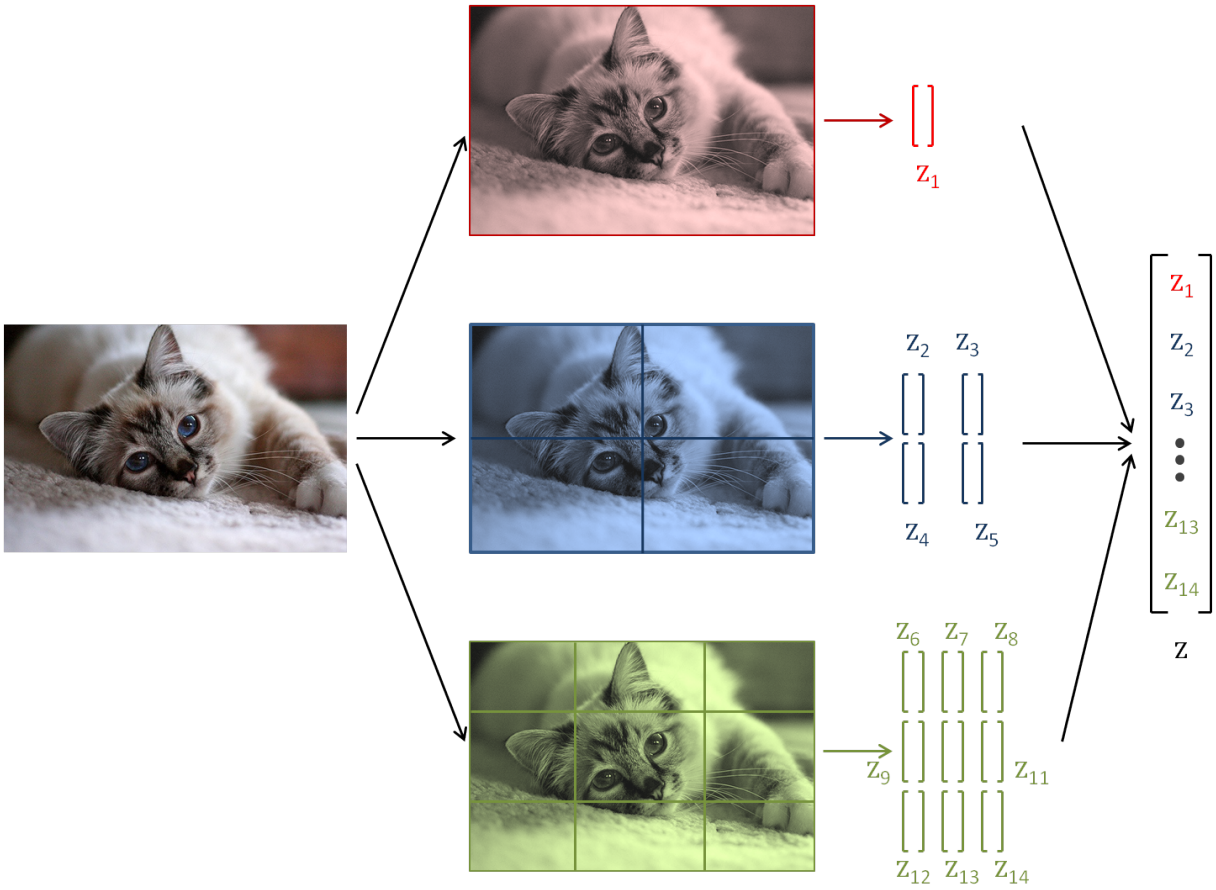



FIGURE 2.3 – Schéma illustrant un SPM (Lazebnik et al., 2006) de taille $1 \times 1 + 2 \times 2 + 3 \times 3$. Cette procédure permet de calculer des représentations localisées et à des échelles variées dans l'image, et ainsi de conserver de l'information spatiale lors de l'agrégation des codes visuels.

cette tendance soit plutôt inversée. Cette représentation est basée sur le processus BoW, et peut être vue comme une extension des BoW à l'encodage de plusieurs ordres de statistiques de la distribution des descripteurs. Dans cette méthode, les descripteurs locaux sont extraits de la même manière que pour les BoW, mais l'apprentissage du dictionnaire ainsi que les étapes d'encodage et d'agrégation diffèrent.

#### Apprentissage du dictionnaire

Afin d'encoder les descripteurs locaux en prenant en compte les premier et second ordres de statistiques de la distribution des descripteurs locaux, celle-ci est maintenant estimée grâce à un mélange de gaussiennes (ou *Gaussian Mixture Model*, soit GMM). Plus précisément, ce mélange de $K$ gaussiennes est appris de manière non supervisée sur l'ensemble des descripteurs locaux extraits des images d'apprentissage. Le nombre de ces gaussiennes $K$ est un hyper-paramètre.

#### Codage

Pour chaque image d'entrée, les descripteurs locaux sont encodés par rapport aux premier et second ordres de statistiques de ce GMM. Ainsi, sur chaque image d'entrée, pour chaque gaussienne $k$ et pour chaque dimension $j$ des descripteurs locaux, on calcule les deux valeurs suivantes :

$$u_{jk} = \frac{1}{N_{descr}\sqrt{\pi_k}} \sum_{i=1}^{N_{descr}} q_{ik} \frac{x_{ij} - \mu_{jk}}{\sigma_{jk}} \tag{2.1}$$

$$v_{jk} = \frac{1}{N_{descr}\sqrt{2\pi_k}} \sum_{i=1}^{N_{descr}} q_{ik} \left( \left( \frac{x_{ij} - \mu_{jk}}{\sigma_{jk}} \right)^2 - 1 \right) \tag{2.2}$$

où $\pi_k$, $\mu_k$ et $\sigma_k$ sont respectivement le poids, le vecteur moyen et la matrice de covariance (diagonale) associés à la gaussienne $k$, et $q_{ik}$ est la probabilité *a posteriori* que le descripteur local $x_i$ ait été généré par la gaussienne $k$ (Perronnin and Dance, 2007).

Ainsi, l'image d'entrée est maintenant représentée par $2 \times K \times D$ valeurs, où $K$ est le nombre de gaussiennes du GMM et $D$ est la dimension des descripteurs locaux. Alors que la première de ces deux valeurs (eq. 2.1) prend en compte la dérivée par rapport à la moyenne, la seconde (eq. 2.2) prend en compte la dérivée par rapport à l'écart-type. De ce fait, cette méthode d'encodage permet de calculer une représentation plus riche que les BoW standards de l'image d'entrée par rapport au dictionnaire appris.

#### Agrégation

Une méthode d'agrégation classique consiste à concaténer les valeurs ci-dessus : on a alors $z = \{u_{jk}, v_{jk}\}_{j=1..D,k=1..K}$. Pour chaque image, le FV a donc une taille de $2KD$, où $K$ est le nombre de gaussiennes et $D$ la dimension des descripteurs locaux. Cependant, comme souligné dans la partie 2.1.1, cette méthode empêche néanmoins la conservation de l'information spatiale.

Pour répondre à ce problème, la solution SPM de Lazebnik et al. (2006) est également beaucoup utilisée avec les FV. Dans chacune des $M$ zones spatiales, on calcule les valeurs $u_{jk}$ et $v_{jk}$ en ne prenant en compte que les descripteurs locaux $x_i$ présents dans la zone

en question. On a ainsi un vecteur $z_m = \{u_{jk}, v_{jk}\}_{j=1..D,k=1..K,x_i \in m}$ pour chaque région $m$. Le FV $z$ représentant l'image est ainsi la concaténation de ces $M$ vecteurs $z_m$, et a donc une taille $2KDM$.

#### *Improved Fisher Vectors*

Perronnin et al. (2010) proposent deux améliorations majeures pour augmenter les performances des FV. D'abord, l'augmentation du nombre de gaussiennes impacte la distribution des descripteurs locaux affectés aux différentes gaussiennes. En effet, plus le nombre de gaussiennes est élevé, plus les descripteurs locaux affectés à chaque gaussienne se raréfient. Pour contrer ce phénomène, les auteurs proposent une normalisation individuelle permettant de rendre la représentation moins parcimonieuse. Selon leurs travaux, chaque élément $z_i$ est donc normalisé par $sign(z_i) \times |z_i|^\alpha$, où $\alpha$ est un hyper-paramètre.

Par ailleurs, dans l'utilisation du SPM, chaque FV $z_m$ représentant une région $m$ est normalisé $L_2$ avant la concaténation de ces $m$ vecteurs. Cette normalisation s'avère expérimentalement efficace et largement utilisée par la suite (Perronnin et al., 2010; Chatfield et al., 2011), notamment lors de l'utilisation de SVM linéaires pour classifier ces représentations.

## 2.2 Classification

La reconnaissance visuelle se traduit comme un problème de classification dans un espace vectoriel, qui est celui des représentations des images. Pour traiter ce problème, de nombreux types de classifieurs ont été proposés dans la littérature. Un classifieur très utilisé pour la reconnaissance visuelle est l'algorithme des $k$ plus proches voisins (ou $k$-NN pour *k-Nearest Neighbour*) (Cover and Hart, 1967; Duda et al., 2001) par exemple ne demandent pas d'apprentissage. Ces méthodes ont l'avantage d'être particulièrement simples à implémenter, cependant elles requièrent beaucoup de ressources de calcul et de mémoire. De plus, elles ne permettent pas de bien généraliser un concept sur une tâche donnée, car l'idée fondamentale revient schématiquement à apprendre l'ensemble des données d'entraînement par cœur. Les méthodes par apprentissage permettent à l'inverse d'apprendre un modèle généralisant la connaissance apportée par les données d'apprentissage.

D'autres méthodes s'appuient sur des ensembles de classifieurs. Par exemple, les forêts d'arbres décisionnels (ou *random forests*) (Breiman, 2001; Moosmann et al., 2006) reposent sur l'apprentissage de plusieurs arbres de décision sur des sous-ensembles tirés aléatoirement parmi les exemples d'apprentissage. De même, la stratégie de boosting (Schapire, 1990; Viola and Jones, 2001; Opelt et al., 2004) consiste à apprendre un ensemble de classifieurs en pondérant plus fortement les images d'apprentissage mal reconnues par le classifieur le plus fort à chaque itération.

Les méthodes de classification par réseaux de neurones peuvent également être utilisées pour la reconnaissance d'images (Bishop, 1995). Ces modèles consistent en un ensemble de poids organisés en couches successives après lesquelles s'appliquent notamment des étapes d'agrégation et des fonctions de non-linéarité. L'intérêt de ces approches réside dans la profondeur qu'il est possible d'exploiter, et donc dans la capacité de hiérarchisation de l'information au sein des réseaux de neurones. Ces méthodes ne sont toutefois que peu utilisées pour classifier des représentations de type BoW, bien que certains travaux se

soient intéressés à concilier les deux (Gong et al., 2014). Les réseaux de neurones sont bien plus souvent exploités pour leur capacité à *représenter* puis classer les images : cette partie sera abordée en détails dans la section 2.3.

L'une des familles de méthodes les plus largement utilisées sur les tâches de reconnaissance d'images est la famille des méthodes à vecteurs de support ou Séparateurs à Vaste Marge (SVM pour *Support Vector Machine*) (Boser et al., 1992; Cristianini and Shawe-Taylor, 2000). En particulier, les représentations BoW ont très souvent été utilisées en conjugaison avec des SVM non-linéaires, comme le noyau gaussien (Avila et al., 2013). Avec les représentations FV, la dimension de l'espace d'entrée est devenue trop importante, et les SVM à noyaux trop coûteux à manipuler : on s'est donc ramené à des classifieurs linéaires.

Dans cette partie, on présente les formulations binaire et multi-classe du SVM, ainsi que les modèles à noyaux. Au-delà d'être une méthode phare pour la classification de représentations visuelles (notamment de type BoW ou FV), plusieurs éléments de cette méthode ont structuré notre réflexion sur les modèles intégrant l'information privilégiée, détaillés dans les chapitres 4, 5 et 6.

### 2.2.1 SVM binaire

Introduit par Boser et al. (1992), le Séparateur à Vaste Marge (ou *Support Vector Machine*, soit SVM) est une fonction apprise de manière supervisée dont le but est de partitionner l'espace des représentations. Cet algorithme a beaucoup été utilisé dans le cadre de la classification d'images (Perronnin et al., 2010; Liu et al., 2011; Wang et al., 2010; Law et al., 2012). Dans ce contexte, le SVM est une frontière séparant d'une part les représentations des images qui ne contiennent pas la classe recherchée, et d'autre part les représentations des images qui la contiennent. Cette frontière doit également maximiser la distance entre les exemples et l'hyperplan de séparation. Plus précisément, on cherche à maximiser la marge entre l'hyperplan et les exemples les plus proches (qui sont les vecteurs de support). D'une part, la frontière de décision doit être le plus éloignée possible de tous les exemples, de sorte à déterminer un séparateur qui soit correct dans la plupart des cas.

Formellement, il s'agit donc d'apprendre le vecteur $w$, normal à la frontière de décision, ainsi que le biais $b$ pour séparer l'espace des représentations en deux sous-espaces pour optimiser le problème à marge souple (ou *soft margin*) suivant (Cortes and Vapnik, 1995) :

$$\begin{aligned} \min_{w,b,\xi_i} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i \\ \text{s.c. } \forall i = 1..N,\ \ y_i(\langle w, x_i\rangle + b) \geq 1 - \xi_i,\ \ \xi_i \geq 0 \end{aligned} \tag{2.3}$$

où $C$ est un hyper-paramètre, et où chaque image $i$ de représentation $x_i$ a le label $y_i = 1$ si la classe cherchée est présente, et -1 sinon. La quantité $\langle w, x_i\rangle + b$ représente la distance de l'exemple $i$ à l'hyperplan $(w, b)$. Autrement dit, cette quantité représente le score de la représentation $x_i$ par rapport à l'hyperplan séparateur. Les variables $\xi_i$ sont appelées variables ressorts (ou *slack variables*), et représentent une tolérance par rapport à la distance à la frontière imposée pour chaque exemple.

Les contraintes traduisent bien la volonté que chaque exemple soit bien classé avec une marge de 1 par le SVM : en effet, on force chaque exemple $x_i$ à avoir un score $\langle w, x_i\rangle + b$

du même signe que son label $y_i$, *i.e.* à être du bon côté du séparateur. De plus, on force ce score à être supérieur à 1 (c'est la marge du classifieur). Les variables $\xi_i$ traduisent quant à elles la possibilité laissée au classifieur de classer avec une moindre marge certains exemples. Cette tolérance a cependant un coût : dès qu'une variable ressort est non nulle, elle intervient dans la quantité à minimiser. Ainsi, chaque exemple classé avec une marge de $1 - \xi_i$ pèse de $C\xi_i$ dans la quantité à minimiser. Une valeur de $C$ élevée traduit donc une volonté d'avoir des valeurs de $\xi_i$ le plus faibles possible, et donc d'avoir un modèle qui classifie tous les exemples correctement avec une marge de 1, quitte à être trop proche des données d'apprentissage.

### 2.2.2 SVM multi-classe

Le modèle développé précédemment permet de traiter des problèmes de classification binaires. Chaque exemple ne peut avoir que deux classes : 1 s'il contient la classe cherchée, -1 sinon. Une vaste majorité des problèmes de classification sont cependant multi-classes, c'est-à-dire que les exemples peuvent être répartis en $N_{classes}$ classes différentes. Pour traiter ces problèmes, plusieurs solutions *ad hoc* peuvent être utilisées pour se ramener à la formulation binaire ci-dessus. La formulation *1 contre 1* découpe par exemple le problème en $\frac{N_{classes}(N_{classes}+1)}{2}$ sous-problèmes binaires de type "classe $k$ vs. classe $j$". La méthode *1 contre tous* en revanche s'appuie sur l'apprentissage de $N_{classes}$ frontières, chacune de type "classe $k$ vs. autre classe". Dans ces deux méthodes, la décision finale pour un exemple est donnée par une agrégation des scores de chaque classifieur. Dans la méthode "1 contre 1", c'est souvent un système de vote qui est utilisé, alors que pour la méthode "1 contre tous", on considère souvent la classe qui a obtenu le score le plus fort. Bien qu'elles permettent de s'attaquer aux problèmes multi-classes, ces méthodes ne sont cependant pas vraiment adaptées à ce type de tâches.

Dans la littérature, d'autres types de classifieurs multi-classes ont été proposés, comme par exemple les réseaux de neurones avec le perceptron multi-classe (Rosenblatt, 1958), ou l'algorithme des k-plus proches voisins (ou *k-Nearest Neighbours*) (Cover and Hart, 1967), qui assigne à toute image d'entrée le résultat d'un vote entre les labels des images les plus proches dans l'espace des représentations.

Dans le domaine des SVM, plusieurs modèles de classifieurs multi-classes ont vu le jour, comme le modèle *Discriminative One-Class* SVM de Labbé (2011). Dans leurs travaux, Crammer and Singer (2001) ont proposé un algorithme de classification permettant d'attaquer directement l'aspect multi-classe d'un problème de classification :

$$\min_{\mathbf{w},\xi_i} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i \qquad (2.4)$$
$$\text{s.c. } \forall i = 1..N,\ \langle w_c, x_i\rangle - \max_{k\neq c}\langle w_k, x_i\rangle \geq 1 - \xi_i,\ \xi_i \geq 0$$

où $\mathbf{w}$ est la matrice des vecteurs normaux $w_k$, chaque $w_k$ représentant le classifieur pour la classe $k$, et $w_c$ est le classifieur pour la classe vérité-terrain $c$ de l'image d'entrée. Ainsi, pour chaque exemple d'apprentissage $i$, on force son score pour sa vraie classe $\langle w_c, x_i\rangle$ à être supérieur au meilleur score parmi les autres classes $\max_{k\neq c}\langle w_k, x_i\rangle$ avec une marge de 1. Grâce aux variables ressorts $\xi_i$, cette marge peut être réduite, au prix d'un coût supplémentaire dans la quantité à minimiser.

Les méthodes présentées dans cette section sont toutes des modèles linéaires. On peut néanmoins étendre ces méthodes aux modèles non-linéaires grâce aux machines à noyaux, en projetant les représentations dans un espace de plus grande dimension puis en cherchant un séparateur linéaire dans ce nouvel espace. De nombreux types de noyaux ont été utilisés dans les diverses applications du SVM non-linéaire (Picard et al., 2013). En particulier, pour le domaine de la reconnaissance d'images, les représentations BoW sont souvent utilisées en conjonction avec un SPM de Lazebnik et al. (2006), inspiré du noyau intersection de Grauman and Darrell (2005), puis avec un SVM à noyau gaussien (Avila et al., 2013; Durand et al., 2013). En revanche, certaines représentations sont déjà de grande dimension : les plonger dans un espace plus riche présente donc peu d'intérêt, et peut au contraire rendre le système plus sensible au sur-apprentissage. De plus, les classifieurs deviennent alors très coûteux à apprendre ainsi qu'à évaluer. Ces cas ont donné lieu à de plus amples réflexions sur des projections dans des espaces non-linéaires appropriés pour utiliser des classifieurs linéaires. En particulier, les FV, qui sont de grande dimension et intègrent de nombreuses transformations non-linéaires, sont le plus souvent discriminés par un SVM linéaire (Perronnin et al., 2010; Chatfield et al., 2011; Avila et al., 2013).

## 2.3 Architectures profondes pour la classification d'images

Dans cette partie, on présente d'abord les réseaux de neurones profonds, puis on s'intéresse plus finement aux blocs élémentaires qui constituent ces structures. On s'attache ensuite à décrire plus précisément les évolutions majeures récentes qui ont permis d'aboutir à des modèles toujours plus performants, puis on montre un panorama de l'utilisation de ces modèles dans la plupart des domaines de vision par ordinateur. Enfin, on discute des problématiques que pose l'apprentissage des réseaux de neurones et notamment des techniques de transfert largement utilisées dans la littérature.

### 2.3.1 Réseaux de neurones

Les architectures profondes en apprentissage automatique sont le fruit d'une volonté de modéliser certains aspects du comportement du cerveau humain dans les réseaux *feed-forward*. Dans le domaine de l'apprentissage automatique, le perceptron de Rosenblatt (1958) s'appuie sur les travaux de modélisation mathématique du neurone (McCulloch and Pitts, 1943). Dans cette définition, les entrées $x_i$ sont pondérées par les poids $w_i$, puis sommées. Une fonction d'activation non-linéaire $\sigma$ est ensuite appliquée sur cette somme. Cette fonction agit comme le principe du potentiel d'action dans les neurones du cerveau humain, qui active le neurone suivant si la charge électrique reçue est suffisante. Ainsi, on peut dire que la sortie $y$ d'un neurone s'écrit $y = \sigma\left(\sum_{i=1}^{m} w_i x_i\right)$. La particularité des poids $w_i$ est qu'ils sont appris sur les données d'apprentissage.

Plusieurs modèles de Perceptron multi-classes peuvent être empilés en un réseau de neurones multi-couches, dénommé perceptron multi-couche (ou MLP pour *Multi-Layer Perceptron*). Dans ce modèle, les neurones sont organisés en couches : chaque neurone d'une couche est connecté à chaque neurone de la couche suivante. Une illustration d'un MLP est proposée sur la figure 2.4. Sur ce schéma, on prend l'exemple d'un MLP à deux couches : une première couche permet de calculer les sorties cachées $h_i$ à partir des entrées $x_i$, puis une deuxième couche permet de calculer les sorties du réseau $y_i$. Les flèches bleues

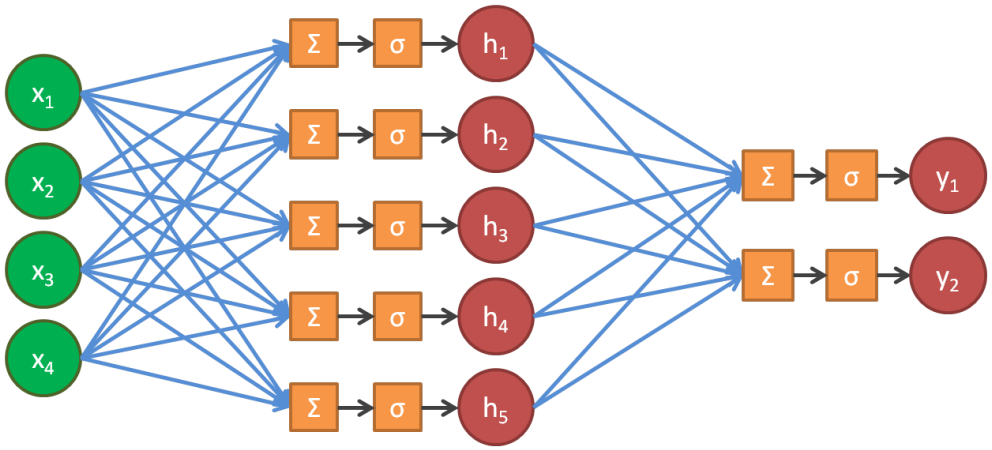



FIGURE 2.4 – Schéma d'un perceptron multi-couche à 1 couche cachée. Les flèches bleues représentent chacune un poids $w_i$ appris sur les données d'apprentissage.

désignent les connexions contenant des poids $w_i$. Ce modèle, pour peu qu'il contienne assez de paramètres, est un approximateur universel de toute fonction continue (Cybenko, 1989; Hornik, 1991). Dans notre cas, on cherche à approximer une sortie précise, définie par la vérité-terrain disponible sur les exemples d'apprentissage. Pour ce faire, différentes fonctions de coût, discutées plus précisément dans la partie 3.3.1, ont été proposées pour différentes tâches. De surcroît, si la fonction de coût est différentiable, alors le réseau de neurones peut être appris grâce à l'algorithme de rétro-propagation du gradient de Werbos (1974) (ou *back-propagation*), appliquée avec succès au cas du perceptron multi-couche à la fin des années 1980 (Rumelhart et al., 1986; LeCun et al., 1989).

Le traitement des images et l'exploitation de la corrélation spatiale entre les pixels a ensuite donné naissance à de nombreux travaux spécialisés dans les réseaux de neurones spécifiques à l'image, avec en particulier le Neocognitron de Fukushima (1980). Enfin, LeCun et al. (1989, 1998) développent les réseaux de neurones convolutionnels, notamment avec le réseau LeNet présenté dans la figure 2.5, maintenant appris via l'algorithme de rétro-propagation du gradient, et obtiennent d'excellentes performances sur la reconnaissance de chiffres manuscrits (LeCun et al., 1989, 1998).

### 2.3.2 Architecture des CNN profonds

Un réseau de neurones profond est constitué d'un certain nombre de neurones, organisés en couches. Ces couches peuvent être connectées entre elles de différentes façons suivant la tâche abordée. Par exemple, les réseaux récurrents (Boden, 2002; Cho et al., 2014; Greff et al., 2015) sont beaucoup utilisés pour le traitement du langage, tandis que les réseaux de type auto-encodeurs (Ranzato et al., 2006; Bengio et al., 2007; Goh et al., 2013) s'adressent plutôt aux tâches non supervisées telles que l'apprentissage de représentations.

Dans ce manuscrit, on s'intéresse aux réseaux dits *feedforward* : chaque couche de neurones prend en entrée la sortie de la couche précédente. Plus précisément, on s'intéresse à un type de réseaux *feedforward* particulier : les réseaux de neurones convolutionnels, ou CNN (*Convolutional Neural Network*). Un CNN est un réseau de neurones dans lequel

une ou plusieurs couches sont des filtres de convolution. Ces réseaux sont constitués de différents types de blocs fonctionnels, qui peuvent être vus comme des briques élémentaires : les couches de neurones, qui peuvent être des filtres de convolution ou des couches totalement connectées, sont souvent suivies de fonctions non-linéaires. Les couches convolutionnelles sont également souvent suivies d'étapes d'agrégation ou de normalisation des sorties. La fonction de coût du réseau, généralement précédée d'une étape de normalisation des sorties, permet de spécifier le but que le réseau doit chercher à atteindre. On donne ici le rôle de chacune de ces fonctions, et le choix détaillé des blocs est discuté dans la partie 3.3.1.

Les couches contenant des neurones permettent de calculer une combinaison linéaire des entrées. Ainsi, les fonctions non-linéaires permettent d'obtenir une représentation plus complexe à chaque niveau de la structure du réseau. Les étapes d'agrégation quant à elles ont un rôle similaire à celles utilisées dans les BoW : elles permettent de résumer l'information extraite grâce à une couche de convolution, et ainsi d'introduire un peu d'invariance à certaines transformations, comme les translations ou rotations locales.

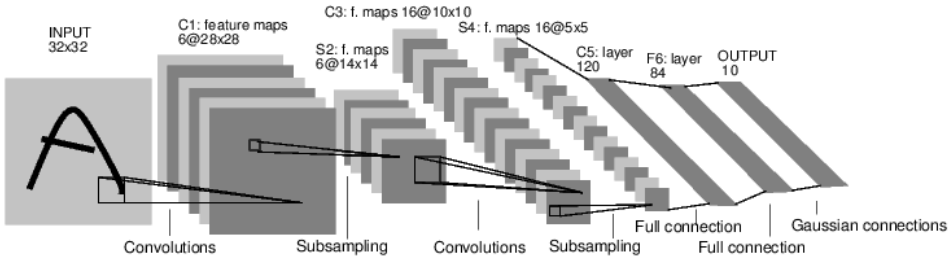
Les filtres de convolution, qui sont généralement les premières couches de neurones, sont appliqués de manière glissante sur toute l'image et à différentes échelles. En ce sens, les couches de convolution peuvent donc schématiquement être vues comme un extracteur de représentations. Les couches les plus profondes sont quant à elles souvent des couches totalement connectées, *i.e.* la sortie de chaque neurone de la couche précédente est connectée à chaque neurone de la couche. De telles couches contiennent donc un nombre considérable de poids, qui doivent être appris sur les images de la base ciblée. Lorsque le nombre de poids est trop important par rapport à la taille de la base de données, le réseau risque plus fortement de sur-apprendre sur ces données. L'intérêt des couches de convolution est justement de diminuer le nombre de poids à apprendre par rapport à un réseau totalement connecté. De plus, ces couches convolutionnelles permettent d'assurer une invariance locale à plusieurs types de petites transformations dans l'image, comme des translations ou des rotations locales.

### 2.3.3 Les CNN profonds pour la reconnaissance visuelle

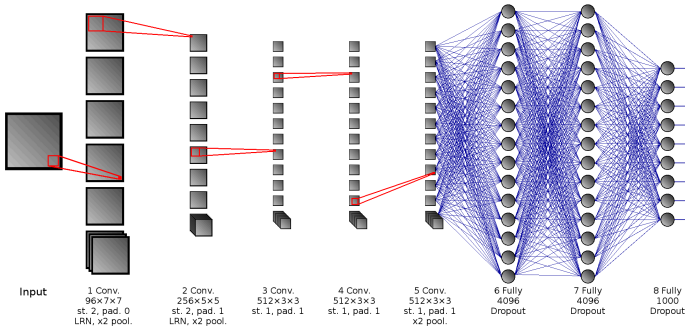
Depuis les premières architectures convolutionnelles, de nombreux modèles ont successivement vu le jour pour améliorer les performances, revisitant à chaque fois l'utilisation des blocs élémentaires décrits auparavant. Dans les années 1990, LeNet est l'un des premiers CNN à avoir été appliqués avec succès sur une tâche de reconnaissance d'images, en l'occurrence la reconnaissance des chiffres manuscrits (LeCun et al., 1998). Ce réseau est très simple car peu profond : il est composé de deux couches de convolution et deux couches totalement connectées, et il n'est appliqué qu'à de petites images ($28 \times 28$ pixels).

Malgré ces très bons résultats, les réseaux de neurones pâtissent de leur image de « boîte noire », difficile à interpréter. De plus, on ne sait pas encore bien les optimiser, et les années 2000 sont plutôt l'âge d'or des modèles de la famille des BoW, discriminés par un SVM. En 2006, l'apprentissage non supervisé permet néanmoins d'apprendre des réseaux de neurones dont les couches sont totalement connectées entre elles : il s'agit des *Deep Belief Nets* (ou DBN) de Hinton et al. (2006) qui s'appuient sur les machines de Boltzmann restreintes (ou RBM pour *Restricted Boltzmann Machines*) de Smolensky (1986).

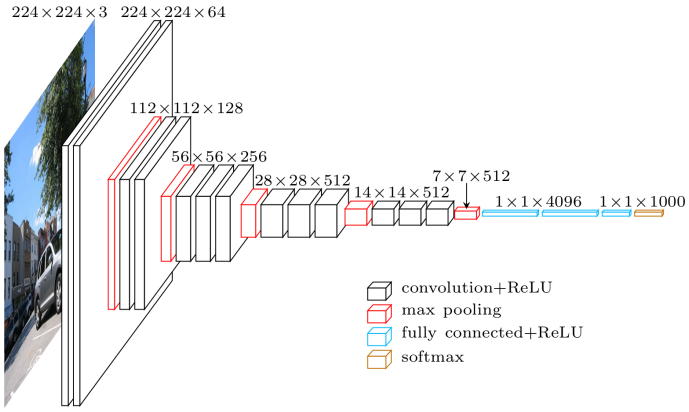
L'année 2012 marque l'avènement des réseaux de neurones dans le paysage de la recon-



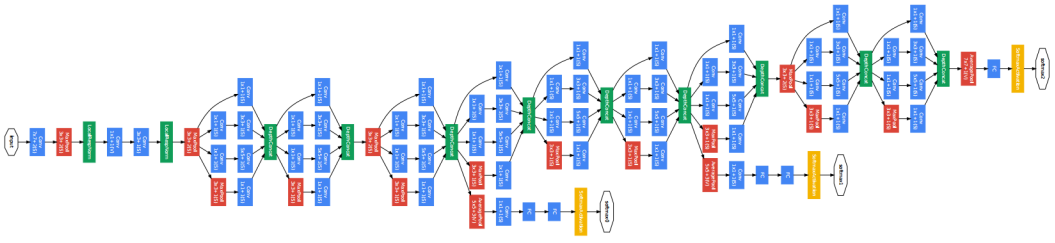

LeNet (LeCun et al., 1998)




VGG-M (Chatfield et al., 2014) (similaire à AlexNet (Krizhevsky et al., 2012))




Very-Deep-VGG16 (Simonyan and Zisserman, 2015)



GoogLeNet (Szegedy et al., 2015)

ResNet-152 (He et al., 2016)

FIGURE 2.5 – Schémas des principaux CNN de l'état de l'art pour la reconnaissance d'images depuis 1998.

naissance d'images, avec les résultats exceptionnels du réseau AlexNet de Krizhevsky et al. (2012) lors de la compétition ILSVRC 2012, améliorant de 9,75% les résultats des années précédentes (Russakovsky et al., 2015). Ce CNN d'environ 60 millions de paramètres est beaucoup plus profond que le LeNet, puisqu'il contient cinq couches de convolution et trois couches totalement connectées. Cela lui permet d'extraire des représentations beaucoup plus sémantiques que les méthodes concurrentes, plus superficielles. De plus, les travaux de Krizhevsky et al. (2012) introduisent de nombreuses astuces d'apprentissage comme le dropout ou l'augmentation de données (discutées dans la section 3.3.1), qui permettent maintenant d'aborder de larges bases de données comme ImageNet, ce qui était impossible auparavant. Cette réussite s'explique notamment par la mise à disposition de larges bases d'images annotées très variées, mais également par l'évolution des ressources de calculs et la démocratisation des systèmes GPU, qui ont permis d'apprendre plus rapidement et plus efficacement l'ensemble des poids sur la très large base qu'est ImageNet (Deng et al., 2009).

D'autres structures du même type comme Overfeat (Sermanet et al., 2014) ou ZFNet (Zeiler and Fergus, 2014) ont vu le jour, et c'est en 2014 que Simonyan and Zisserman (2015) montrent l'intérêt des réseaux très profonds avec les very-deep-VGGNet VGG16 et VGG19, contenant respectivement 13 et 16 couches de convolutions suivies de 3 couches totalement connectées pour un total d'environ 140 millions de paramètres.

L'un des problèmes principaux de ces réseaux est le nombre de poids à apprendre, qui devient beaucoup trop important pour la plupart des tâches. Pour répondre à ce problème, Szegedy et al. (2015) proposent GoogLeNet, un CNN de seulement 6,8 millions de paramètres constitué uniquement de couches de convolutions. La dernière couche est suivie d'une étape d'agrégation de type *average-pooling*, permettant de prendre une décision sur la classe de l'image. Ces travaux s'appuient notamment sur l'introduction du module Inception, qui permet de réduire drastiquement le nombre de paramètres du réseau.

Enfin, ResNet (He et al., 2016) introduit une structure contenant des connexions "court-circuits" qui permettraient de pallier le problème du gradient évanescent et ainsi d'apprendre des architectures beaucoup plus profondes, allant jusqu'à 200 couches de paramètres sur les larges bases de données comme ImageNet (Deng et al., 2009), et 1 202 couches sur de plus petites bases comme CIFAR-10 (Krizhevsky, 2009). La figure 2.5 offre un aperçu des différents CNN ayant marqué l'état de l'art.

Grâce à toutes ces évolutions, les réseaux de neurones ont atteint aujourd'hui des performances exceptionnelles sur la plupart des tâches de vision par ordinateur, et ont ainsi éveillé l'intérêt de nombreux travaux de recherche (Lenc and Vedaldi, 2015b; Szegedy et al., 2014; Yosinski et al., 2014; Nguyen et al., 2015; Mahendran and Vedaldi, 2015; Zeiler and Fergus, 2014; Carvalho et al., 2016). Par exemple, la reconnaissance d'objets (Razavian et al., 2014; Wan et al., 2015; Oquab et al., 2015; Liang and Hu, 2015), d'actions (Gkioxari et al., 2015; Diba et al., 2016) ou d'images axées grain-fin (Reed et al., 2016; Zhang et al., 2014; Huang et al., 2016) ont également suscité beaucoup d'intérêt de la part de la communauté.

Dans le domaine de la détection d'objets, de nombreuses méthodes (Szegedy et al., 2013; He et al., 2014; Bell et al., 2016; Dai et al., 2016) parmi lesquelles R-CNN (Girshick et al., 2014), Fast R-CNN (Girshick, 2015), Faster R-CNN (Ren et al., 2015), R-CNN minus R (Lenc and Vedaldi, 2015a), et YOLO (Redmon et al., 2016) ont successivement

établi les performances état de l'art sur les challenges de détection. Plusieurs travaux ont par ailleurs exploré l'intégration des modèles à parties déformables (*Deformable Part Models* ou DPM) (Felzenszwalb et al., 2010) dans le cadre des CNN (Savalle et al., 2014; Girshick et al., 2015; Ouyang et al., 2015; Wan et al., 2015). De plus, certains travaux s'appuient sur la détection d'objets dans les images pour améliorer la classification d'images. En effet, malgré l'intérêt de ces architectures profondes, les représentations calculées au sein du réseau sont très faiblement invariantes aux transformations, comme par exemple les translations ou les rotations. Plusieurs travaux se sont intéressés à ce problème, notamment en intégrant de l'apprentissage faiblement supervisé pour détecter les objets dans l'image pour améliorer la classification (Durand et al., 2016; Oquab et al., 2015). Les réseaux de neurones ont également été appliqués avec succès aux domaines de l'estimation de pose (Gkioxari et al., 2014; Sinha et al., 2016; Toshev and Szegedy, 2014) et la segmentation sémantique (Chen et al., 2015; Liang et al., 2016; Zheng et al., 2015; Long et al., 2015). Dans le cadre du traitement de la vidéo, les CNN ont également fait l'objet de nombreux travaux en classification (Ballas et al., 2016), prédiction de vidéo (Mathieu et al., 2016), flot optique (Fischer et al., 2015) ou en poursuite d'objets (Li et al., 2014; Wang et al., 2015a).

L'un des sujets de recherche les plus étudiés aujourd'hui consiste néanmoins à allier les domaines du traitement du langage et de la vision par ordinateur. Plusieurs bases de données telles que MS-COCO (Lin et al., 2014) ou Visual Genome (Krishna et al., 2016) ont récemment permis l'émergence de nouvelles applications telles que le sous-titrage (*image captioning*) et la description d'images (Johnson et al., 2016; Donahue et al., 2015; Vinyals et al., 2015; Fang et al., 2015; Xu et al., 2015) ou encore la réponse à des questions (*question answering*) posées sur des images (Gao et al., 2015; Malinowski et al., 2015; Yang et al., 2016; Wu et al., 2016; Noh et al., 2016).

Le traitement de toutes ces données d'entrée nécessite également une implémentation efficace des réseaux de neurones. En ce sens, l'utilisation à grande échelle des GPU a permis de paralléliser des traitements rapides des données et ainsi d'apprendre les CNN dans un temps raisonnable. Aujourd'hui, de nombreuses bibliothèques publiques ont été optimisées pour paralléliser les calculs en CPU ou en GPU dans différents langages, comme Torch (Collobert et al., 2011) (Lua), Caffe (Jia et al., 2014) (C++), MatConvNet (Vedaldi and Lenc, 2015) (MatLab), Theano (Theano Development Team, 2016) (Python) ou encore TensorFlow (TensorFlow Development Team, 2015) (Python).

#### Problématique de transfert

Les réseaux de neurones présentent deux originalités majeures par rapport aux méthodes de type BoW qui détenaient l'état de l'art dans les années 2000 sur la plupart des tâches de reconnaissance d'images. D'abord, les poids des différentes couches sont appris sur la tâche abordée. Ainsi, l'ensemble des représentations internes des images est directement adapté à la tâche d'intérêt. D'autre part, la profondeur du réseau permet d'extraire de l'information des images à différents niveaux. Il semblerait que la hiérarchie profonde de ces architectures permette d'interpréter les images à un niveau sémantique bien supérieur à celui des FV, comme le soulignent les visualisations des représentations profondes de la figure 2.6. Cette course à la profondeur se fait néanmoins au prix d'un nombre élevé de neurones. De fait, jusqu'aux travaux de Krizhevsky et al. (2012), l'apprentissage des réseaux de neurones restait un frein majeur à l'utilisation de ces méthodes.

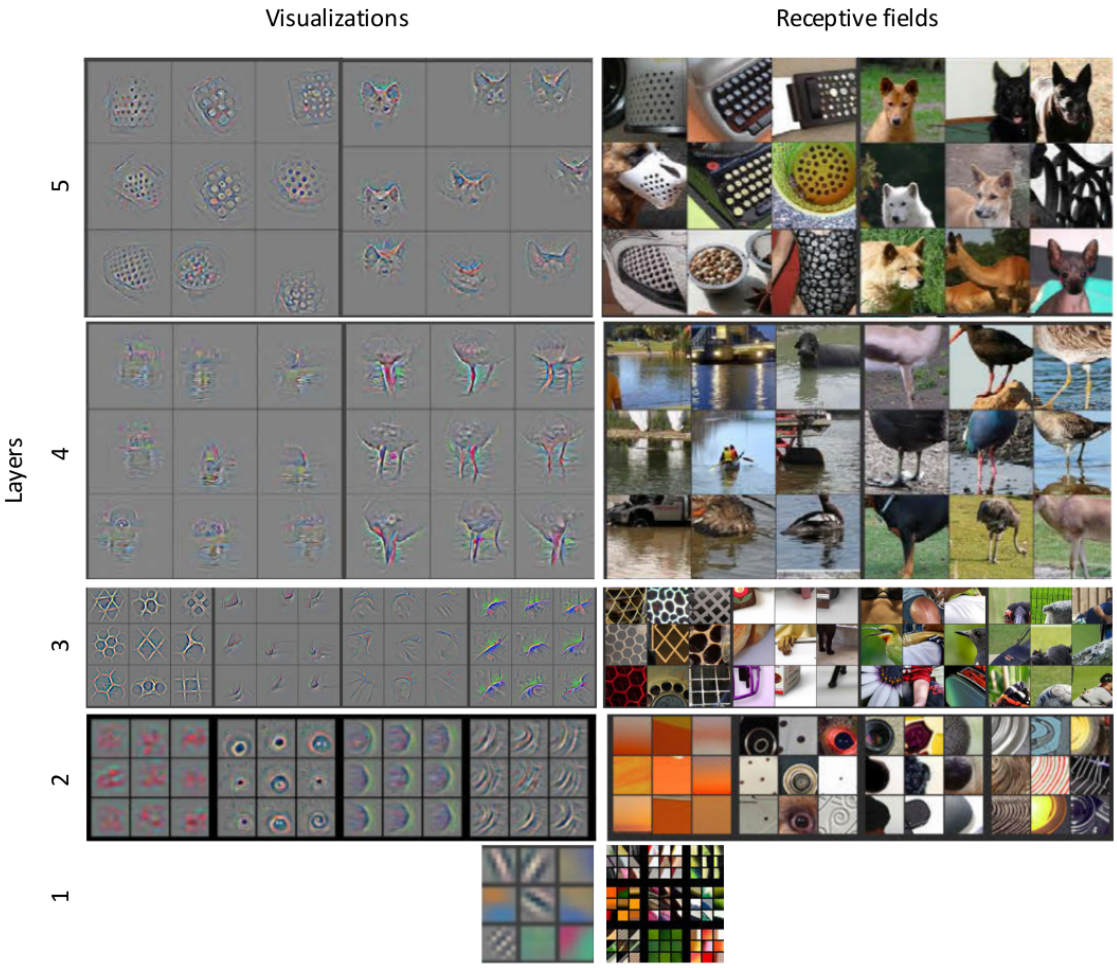



FIGURE 2.6 – Visualisation des représentations extraites à différents niveaux de profondeur d'un CNN similaire à AlexNet (illustration extraite de Zeiler and Fergus (2014)). Les représentations extraites des couches de plus en plus profondes sont de plus en plus sémantiques.

L'apprentissage de tous ces poids nécessite un nombre important de données, ce qui pose problème lorsqu'on cherche à utiliser les réseaux de neurones sur des bases de données de faible ou moyenne envergure (*i.e.* jusqu'à quelques milliers d'exemples d'apprentissage). Afin d'éviter le sur-apprentissage, une technique consiste à transférer les poids appris sur une large base de données externe vers la base de données d'intérêt. Ainsi, les poids du réseau sont appris sur une large base de données externe telle que ImageNet (Deng et al., 2009) (1,2 millions d'exemples), Places (Zhou et al., 2014) (8 millions d'exemples) ou MS-COCO (Lin et al., 2014) (100 000 exemples). Le réseau ainsi appris est ensuite utilisé tel quel comme un extracteur de représentations sur les données d'intérêt ; ces représentations sont discriminées par un classifieur appris de manière supervisée sur ces descripteurs, par exemple un SVM. Cette technique dite de transfert a donné de très bonnes performances notamment en classification d'images, et ce sur des bases variées (Razavian et al., 2014; Oquab et al., 2014; Chatfield et al., 2014; Carvalho et al., 2016).

## 2.4 Bilan

Dans ce chapitre, on s'est attaché à présenter différentes méthodes de l'état de l'art pour la représentation et la classification des images. Deux méthodes principales se dégagent de cette analyse. D'abord, la méthode des Fisher Vectors, basée sur l'encodage de descripteurs locaux extraits de manière dense dans l'image par rapport à un dictionnaire appris de manière non-supervisée sur les images d'apprentissage, donne de bons résultats de reconnaissance d'images en conjonction avec un SVM linéaire multi-classe. Cette représentation est dite de niveau intermédiaire, et le processus général est déterminé manuellement.

Ensuite, les réseaux de neurones convolutionnels se sont récemment affirmés comme une méthode état de l'art sur la plupart des tâches de reconnaissance d'images depuis les excellents résultats de Krizhevsky et al. (2012). Ces modèles permettent d'extraire de l'information à différents niveaux des images, mais aussi de s'adapter assez facilement à une tâche d'intérêt, puisque les poids sont appris sur les données de manière supervisée. Deux méthodes d'utilisation de ces architectures ressortent principalement dans l'état de l'art. D'une part, les paramètres peuvent être appris (voire ré-appris dans le cadre du *fine-tuning*) sur les données d'apprentissage de la tâche ciblée, et le réseau permet alors directement de classifier les images de test. Cette approche est adaptée au cas où on dispose d'assez d'exemples d'apprentissage par rapport au nombre de paramètres à entraîner. D'autre part, le réseau peut être utilisé en transfert : les poids sont appris sur une large base de données externe, puis le réseau est utilisé comme un extracteur de descripteur. Il permet de calculer une représentation profonde sur les données ciblées, qui est alors discriminée par un classifieur appris sur ces vecteurs. Cette méthode permet de bénéficier de l'expressivité des CNN profonds sur des bases de données contenant trop peu d'images pour y entraîner les paramètres sans risquer de sur-apprentissage.

Dans la suite de ce manuscrit, on s'intéresse à comparer les performances de ces deux familles de méthodes dans notre contexte de reconnaissance de petites images orientées grain-fin. Le contexte de notre étude nous amène à traiter des images de faible résolution. Ainsi, on se pose ici la question suivante : jusqu'à quelle résolution peut-on utiliser les méthodes de l'état de l'art sans souffrir d'une perte de performances trop importante ? Quel est l'impact de la résolution sur ces méthodes ? Pour répondre à ces questions, on

évalue donc d'une part les représentations Fisher Vectors, et d'autre part des représentations profondes pré-entraînées sur une base de données externe sur une même chaîne de classification d'images de résolution variable dans un contexte grain-fin.

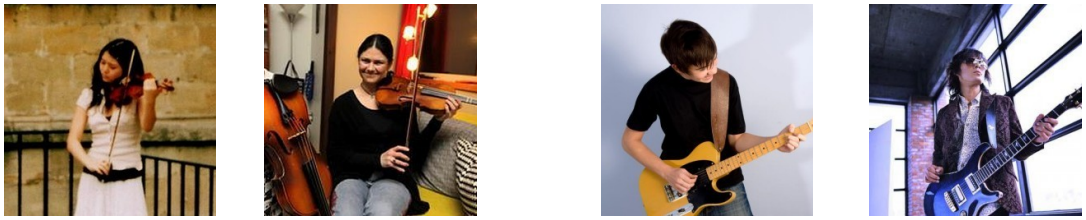
# Chapitre 3

# Etude des représentations pour la classification à grain-fin d'images faiblement résolues

**Sommaire**



Dans notre contexte industriel, on s'intéresse à la reconnaissance visuelle de véhicules assez lointains par rapport au porteur, *i.e.* dans des images de résolution assez faible. On étudie dans un premier temps l'adéquation des méthodes de l'état de l'art pour la représentation des images par rapport à ce contexte de reconnaissance d'images de faible résolution : les Fisher Vectors (FV) et les réseaux de neurones convolutionnels profonds (CNN), deux familles de méthodes majeures identifiées dans le chapitre 2. Dans ce but, on veut étudier l'impact de la résolution des images dans le choix des méthodes de traitement. On s'intéresse pour cela à la reconnaissance d'images dans un contexte grain-fin. Comme l'illustre la figure 3.1, ce type de bases de données contient des classes assez difficiles à discriminer : dans cette figure, on représente des images appartenant à différentes classes



Joue du violon | Tient le violon | Joue de la guitare | Tient la guitare

FIGURE 3.1 – Exemples d'images de PPMI (Yao and Fei-Fei, 2010), une base d'images orientée grain-fin. Différencier les personnes qui jouent ou qui tiennent simplement l'instrument est une tâche difficile, même pour l'œil humain.

de la base PPMI, consistant à différentier visuellement les personnes qui jouent d'un instrument, de celles qui n'en jouent pas.

Dans ce chapitre, on évalue d'abord l'impact de la résolution sur les performances des méthodes de l'état de l'art pour la reconnaissance d'images dans un contexte grain-fin. On s'intéresse ensuite à améliorer les performances de représentations de l'état de l'art, notamment grâce à l'utilisation naïve de données complémentaires durant l'apprentissage. On propose également LR-CNN, un réseau de neurones profond adapté à notre contexte.

## 3.1 Comparaison des représentations intermédiaires

### 3.1.1 Description des bases de données

La reconnaissance à grain-fin est une application particulièrement difficile de la vision par ordinateur : elle consiste à distinguer des classes très similaires. De nombreuses bases de données orientées grain-fin ont été proposées pour traiter cette tâche, comme par exemple FGVC-Aircraft (Maji et al., 2013), Caltech-UCSD Birds-200-2011 (Welinder et al., 2010), 102 Category Flower Dataset (Nilsback and Zisserman, 2008), PPMI (Yao and Fei-Fei, 2010). Dans nos travaux, on s'intéresse à deux bases d'images orientées grain-fin, qu'on décrit dans cette partie : FGVC-Aircraft et PPMI.

#### FGVC-Aircraft

FGVC-Aircraft (pour *Fine-Grained Visual Classification of Aircraft*) (Maji et al., 2013) est une base d'images très résolues (de l'ordre de $800 \times 1000$ pixels environ). Cette base de données comprend 10 000 images d'avions. Plusieurs niveaux de granularité sont disponibles au sein des labels : en effet, les images sont réparties en 100 variantes, usinées par 30 fabricants différents. Les images sont équitablement réparties entre les variantes (100 images par variante), ce qui n'est pas le cas de la répartition des variantes au sein des fabricants. Pour nos expériences, on s'intéresse uniquement au niveau le plus fin de reconnaissance, *i.e.* les 100 classes de variantes d'avions. On représente sur la figure 3.2 des images de 6 classes différentes, qui illustrent d'une part la variabilité intra- et inter-classe, et d'autre part la ressemblance entre les classes. En effet, la base contient des classes d'avions très différentes (*e.g.* Spitfire vs. 737-200), cependant certaines classes peuvent être très semblables, notamment les variantes d'un même constructeur (*e.g.* A310 vs. A340-500 pour Airbus) ou des modèles similaires chez deux concurrents (*e.g.* 737-200

chez Boeing vs. A310 chez Airbus). De plus, bien qu'une grande partie des images soient des avions vus de profil et assez centrés dans l'image (colonne *Vue standard* de la figure), un certain nombre d'images contiennent des vues atypiques ou partielles des avions.

Les données sont équiréparties entre un ensemble d'apprentissage (3 334 images), un ensemble de validation (3 333 images) et un ensemble de test (3 333 images). Pour toutes nos expériences, on reporte les résultats des méthodes apprises sur les ensembles d'apprentissage et de validation, et testées sur l'ensemble de test.

Par sa nature, cette base de données présente des similitudes avec le contexte applicatif de la reconnaissance de véhicules, ce qui en fait un outil idéal pour notre étude.

#### PPMI

PPMI (pour *People Playing Musical Instruments*) (Yao and Fei-Fei, 2010) est une base de 4 800 images contenant des personnes avec des instruments de musique. Sur chaque image sont présents une personne et un instrument de musique tiré de la liste de 12 instruments suivante : basson, violoncelle, clarinette, erhu, flûte traversière, cor anglais, guitare, harpe, flûte à bec, saxophone, trompette, violon. Le but est de déterminer si la personne joue de l'instrument ou le tient seulement sans y jouer. Ainsi, pour chaque classe d'instrument, la tâche abordée est un problème de classification binaire. On présente dans la figure 3.1 quelques exemples de cette base de données, qui montrent la difficulté de cette tâche.

Pour nos expériences, on utilise la base d'images $258 \times 258$ disponible en ligne. Ces images sont des zones de dimensions fixes cadrées autour de la tête de l'instrumentiste. De plus, les données sont réparties en Cette deuxième base de données nous permet de corroborer nos observations dans un contexte différent de la première, ce qui nous permet de généraliser quelque peu notre analyse.

### 3.1.2 Pré-traitement des images

Pour montrer l'influence de la résolution sur les performances des méthodes de l'état de l'art pour la représentation d'images, on s'intéresse à sept résolutions spécifiques, comprises entre $200 \times 200$ et $10 \times 10$ pixels. Pour ce faire, on génère, pour chaque base de données (Maji et al., 2013; Yao and Fei-Fei, 2010), sept bases dérivées, chacune contenant la totalité des images retaillées à $s \times s$ pixels. Les images sont retaillées via une interpolation bicubique. Ces bases sont considérées comme des problèmes de classification indépendants : pour chaque résolution $s$, les classifieurs sont entraînés sur les images d'apprentissage $s \times s$ et testés sur les images de test de taille $s \times s$.

Les deux représentations étudiées sont cependant très sensibles à ces forts changements de taille d'image. D'une part, on utilise VGG-M (Chatfield et al., 2014), un réseau de neurones de cinq couches de convolution suivies de trois couches totalement connectées (comme le montre le schéma de la figure 2.5). Du fait de sa structure fixe, VGG-M requiert en entrée une image de taille $224 \times 224$. D'autre part, de trop petites images ne permettent pas de calculer assez de SIFT, ce qui affecte fortement le calcul des FV. Pour remédier à ces deux problèmes, les petites images de taille $s \times s$ sont agrandies via une interpolation au plus proche voisin. Ce processus permet d'obtenir des images ayant assez de pixels sans pour autant altérer la qualité obtenue en faible résolution. Un schéma de l'ensemble du processus est représenté sur la figure 3.3. Expérimentalement, on vérifie que le choix


FIGURE 3.2 – Exemples d'images de 6 classes différentes de FGVC-Aircraft (Maji et al., 2013). Il y a beaucoup de variabilité entre les classes (F-16A/B vs. Spitfire), malgré cela certaines classes peuvent se ressembler, notamment les variantes d'avions d'un même fabricant (A310 vs. A340-500), et la variabilité intra-classe peut être élevée (vues partielles ou atypiques).

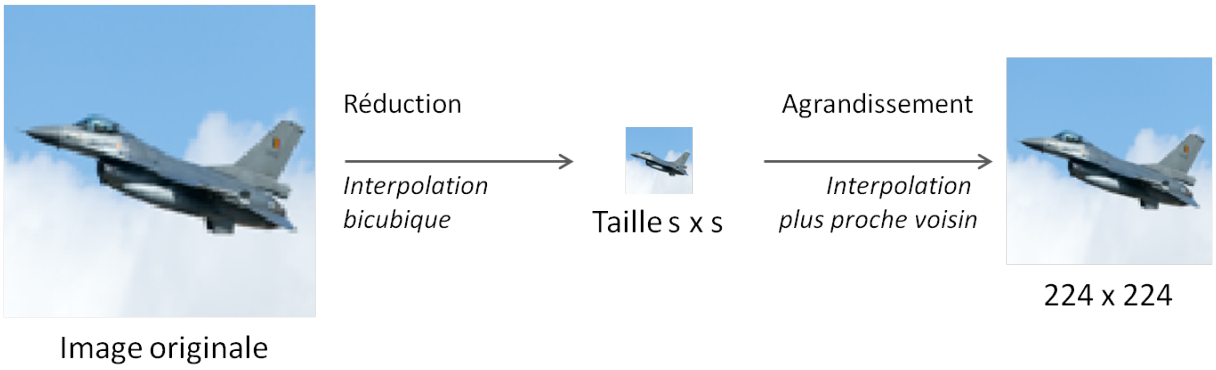



FIGURE 3.3 – Schéma du processus de pré-traitement des images.

des deux méthodes de retaillage n'impactent pas les conclusions de la suite : en testant d'autres méthodes de retaillage, on observe que les résultats ne sont que très peu modifiés, et la comparaison des performances reste inchangée.

### 3.1.3 Impact de la résolution

**Paramétrages**

Pour les représentations FV, sur les deux bases, on extrait des SIFT multi-échelle denses en niveaux de gris (Lowe, 1999) On utilise un GMM de 64 gaussiennes et une pyramide spatiale $1 \times 1 + 2 \times 2 + 4 \times 4$. Le FV résultant a une taille de 344 064. Suivant le protocole de (Perronnin et al., 2010), les FV sont normalisés L2 et normalisés en puissance avec $\alpha = 0,5$. Les FV sont implémenté grâce à la bibliothèque VlFeat (Vedaldi and Fulkerson, 2008).

Pour cette étude, on utilise le réseau de neurones profond VGG-M de (Chatfield et al., 2014), qui contient 5 couches convolutionnelles et 3 couches totalement connectées. Ce réseau est pré-entraîné sur ILSVRC12, contenant 1,2 millions d'images appartenant à 1 000 classes distinctes (Deng et al., 2009). Dans notre étude, on utilise une partie de ce réseau pré-entraîné comme un extracteur de représentations, dans un protocole similaire à ceux de (Razavian et al., 2014) sur Caltech-UCSD Birds ou (Chatfield et al., 2014) sur PASCALVOC. On l'implémente en utilisant la bibliothèque MatConvNet (Vedaldi and Lenc, 2015). Pour montrer l'importance du choix de la représentation profonde, on extrait les représentations en utilisant les sorties de différentes couches du réseau. Ainsi, on utilise la sortie de la deuxième couche totalement connectée (VGG-M_fc2) puisqu'il s'agit d'une configuration standard dans l'analyse des représentations profondes, ainsi que la sortie de la première couche totalement connectée (VGG-M_fc1). Dans le tableau 3.1, on rappelle la taille des vecteurs de représentation en sortie des couches à différents niveaux de profondeur du réseau. Comme le montre ce tableau, les vecteurs de représentation considérés ont tous deux une taille 4 096, et sont normalisés L2.

Pour discriminer ces représentations, on choisit d'utiliser un classifieur linéaire car d'une part on s'intéresse à la comparaison des performances de ces deux types de représentations, et d'autre part, comme souligné dans le chapitre 2, dans les deux cas, ce classifieur est souvent utilisé dans la littérature. On utilise une optimisation multi-classe Crammer-Singer sur FGVC-Aircraft, et un SVM linéaire de coût L1 (*hinge loss*)

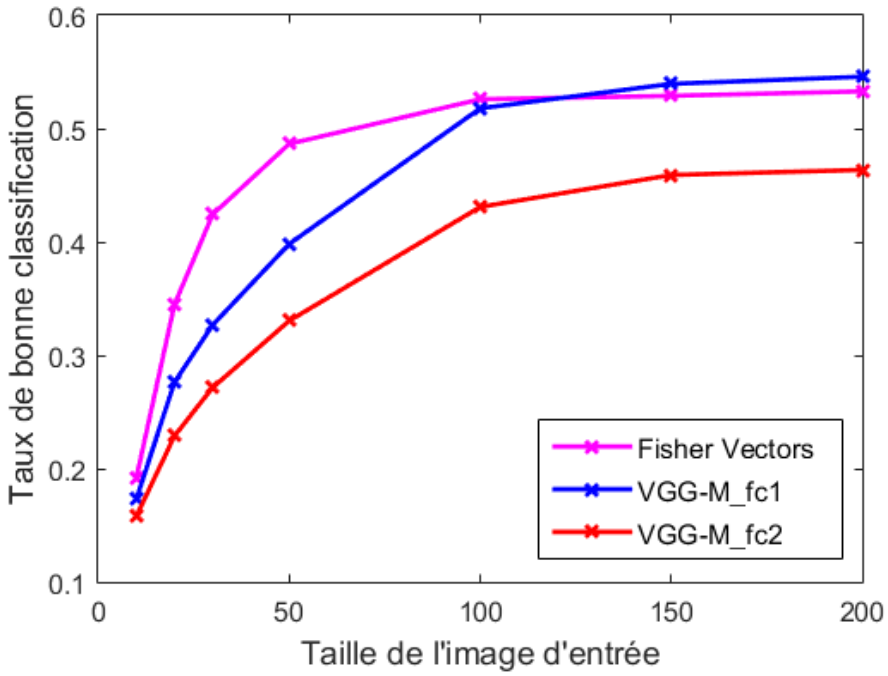



FIGURE 3.4 – Performances des représentations FV et VGG-M en fonction de la résolution des images FGVC-Aircraft.

et régularisé L2 sur la tâche binaire PPMI. Ces deux classifieurs sont implémentés via la bibliothèque LibLinear (Fan et al., 2008).

**Résultats**

On représente sur les figures 3.4 et 3.5 les performances de classification en fonction de la taille des images d'entrée pour plusieurs résolutions échantillonnées entre $200 \times 200$ et $10 \times 10$ pixels respectivement sur FGVC-Aircraft et PPMI. Il convient de remarquer que les performances état de l'art sur les images fortement résolues pourraient être améliorées en utilisant par exemple des données additionnelles ou des bases de données externes. Dans ces travaux, on cherche cependant à comparer les méthodes entre elles, et non à obtenir l'état de l'art avec les méthodes testées.

On s'intéresse à l'impact de la résolution sur les performances de classification. Il convient de remarquer que les trois méthodes présentent le même comportement bimodal sur les deux bases de données par rapport à la diminution de résolution. Tant que la résolution reste assez élevée, les performances restent stables : pour FV, la perte de performances entre $200 \times 200$ et $100 \times 100$ est de 0,7% sur FGVC-Aircraft et 0,4% sur PPMI. Ensuite, pour toutes les résolutions en-deçà de ces valeurs critiques, les méthodes semblent devenir beaucoup plus sensibles à la résolution des images : l'écart de performances de FV entre $50 \times 50$ et $20 \times 20$ est de 14,2% sur FGVC-Aircraft et 4,2% sur PPMI. Les représentations profondes démontrent également un comportement similaire : sur FGVC-Aircraft, les performances de VGG-M_fc2 ne perdent que 3,3% entre $200 \times 200$ et $100 \times 100$, puis s'écroulent de 20,2% entre $100 \times 100$ et $20 \times 20$.

Pour illustrer ce comportement, on représente sur la figure 3.6 des images de deux

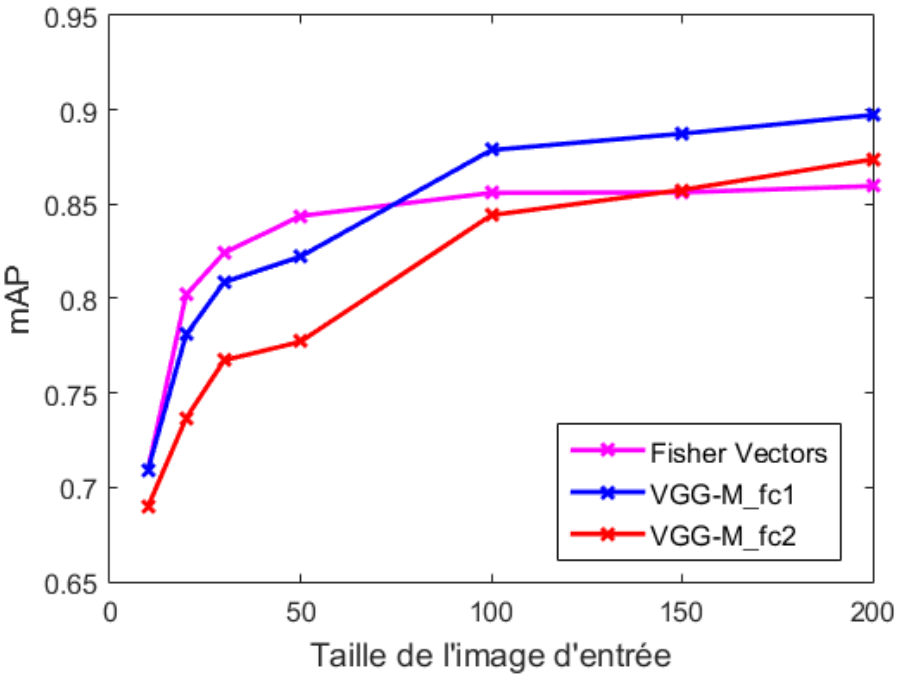



FIGURE 3.5 – Performances des représentations FV et VGG-M en fonction de la résolution des images PPMI.

avions Airbus à différentes résolutions. Avec $200 \times 200$ pixels, la variante A380 est facilement lisible sur la queue et le fuselage de l'appareil du haut. On peut également voir que l'A380 a deux étages et deux réacteurs par aile, alors que l'A330-200 n'a qu'un étage et un seul réacteur de chaque côté. Lorsqu'on diminue la résolution, à $50 \times 50$ pixels, on ne lit plus la variante, en revanche on peut toujours deviner le nombre d'étages et de réacteurs de chacun des avions. Finalement, à résolution critique ($20 \times 20$ pixels), il devient impossible de distinguer l'un ou l'autre de ces détails discriminants.

Ces courbes montrent également que le FV reste compétitif par rapport aux représentations profondes en termes de performances. Il convient cependant de remarquer que les FV ont une taille de près de 350 000 dimensions, ce qui est assez lourd à manipuler d'un point de vue calculatoire. Les représentations profondes quant à elles ne comportent que quelques milliers de dimensions.

### 3.1.4 Influence de la profondeur d'extraction des représentations CNN

Sur les deux figures 3.4 et 3.5, on montre que le niveau d'extraction des représentations profondes peut influencer les performances sur la base de données ciblée. En effet, les résultats cd VGG-M_fc1 sont systématiquement meilleurs que les performances de VGG-M_fc2, et ce sur les deux bases de données FGVC-Aircraft et PPMI. Afin de pousser l'étude plus avant, on reporte sur la figure 3.7 les performances des représentations extraites à d'autres niveaux de VGG-M. On extrait systématiquement après chaque couche la représentation profonde des images. Lorsqu'il s'agit de couches de convolution, les sorties ne sont pas sous forme de vecteur, mais d'une matrice, qui représente un ensemble de cartes de réponses pour les différents filtres : dans ces cas-là, on concatène tous les

FIGURE 3.6 – Exemples d'images de deux avions Airbus (A380 en haut, A330-200 en bas) à résolutions $200 \times 200$ (gauche), $50 \times 50$ (milieu) et $20 \times 20$ (droite). La dégradation de résolution occulte les détails discriminants qui permettent de distinguer ces classes très proches.

| conv1 | conv2 | conv3 | conv4 | conv5 | fc1 | fc2 | fc3 |
|---|---|---|---|---|---|---|---|
| $96 \times 109 \times 109$ | $256 \times 26 \times 26$ | $512 \times 13 \times 13$ | $512 \times 13 \times 13$ | $512 \times 13 \times 13$ | | | |
| 1 140 576 | 173 056 | 86 528 | 86 528 | 86 528 | 4 096 | 4 096 | 1 000 |

TABLE 3.1 – Taille des représentations extraites du réseau VGG-M pré-entraîné sur ImageNet à différents niveaux de profondeur.

éléments pour en faire un vecteur de représentation. On garde ensuite le même protocole expérimental qu'auparavant : pour une profondeur donnée $p$, on apprend et on teste un SVM linéaire structuré sur les représentations extraites à cette profondeur. On reporte également dans le tableau 3.1 la taille des représentations extraites aux différents niveaux de profondeur.

Pour un niveau assez superficiel du réseau (conv2), on constate que les performances ne sont pas particulièrement élevées. En effet, comme le réseau VGG-M est une structure assez profonde et hiérarchisée, les représentations extraites des premières couches de convolution ne sont pas très complexes, et ne permettent peut-être pas de représenter les images d'entrée assez précisément. Les performances augmentent lorsqu'on extrait des représentations dans des niveaux intermédiaires du réseau (conv3, conv4, conv5). Les cartes de réponses ainsi calculées sont de plus en plus complexes et complètes, ce qui permet une meilleure représentation des images d'entrée. En revanche, lorsqu'on extrait à des niveaux profonds, au niveau des couches totalement connectées (fc1, fc2, fc3), les performances de classification s'écroulent au fur et à mesure que la profondeur augmente. Ce comportement tend à supporter la conclusion que les dernières couches d'un réseau sont plus spécifiques aux données sur lesquelles elles ont été apprises, alors que les couches plus superficielles offrent une description plus générique de l'image d'entrée, ce qui rejoint les travaux de (Yosinski et al., 2014).

On a donc montré que les représentations à des niveaux intermédiaires du réseau

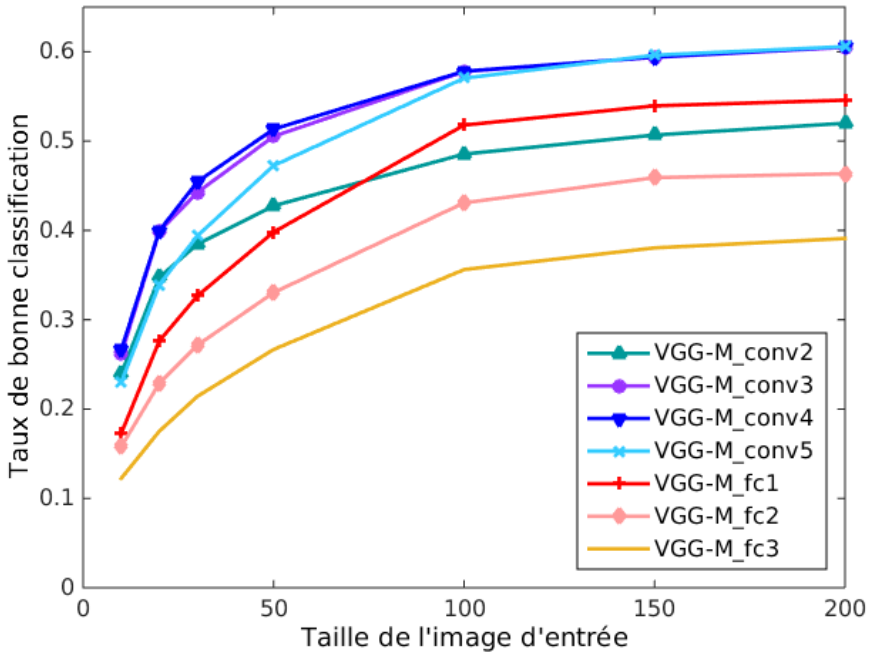



FIGURE 3.7 – Performances de représentations extraites à différents niveaux de profondeur de VGG-M sur FGVC-Aircraft.

peuvent améliorer les performances. Il convient cependant de noter les tailles des représentations extraites, rappelées dans le tableau 3.1. En effet, les représentations intermédiaires sont de plus de 86 000 dimensions. Cette taille est considérable, et ne permet pas de traiter rapidement et simplement les images.

### 3.1.5 Conclusion

Ici, on a comparé deux représentations compétitives : des représentations architecturées manuellement (les FV), et des représentations profondes pré-apprises sur une large base de données externe. Ces représentations sont évaluées sur une tâche de classification à grain-fin, en fonction de la résolution des images d'entrée. On a mis en lumière un comportement bimodal : pour les résolutions moyennes et élevées ($\geq 100 \times 100$ pixels), les performances restent plutôt stables, puis elles s'écroulent lorsqu'on atteint une résolution critique ($\leq 50 \times 50$ pixels). On montre aussi que les représentations profondes sont intéressantes puisqu'elles sont assez compactes et leurs performances restent plutôt élevées sur les résolutions assez hautes. Cependant, lorsque la résolution diminue, leurs performances s'écroulent, et les représentations profondes pré-entraînées deviennent alors peu attractives.

On se pose alors la question de l'amélioration de ces performances : l'utilisation des images haute résolution en plus des images faiblement résolues peut-elle permettre d'améliorer les performances ? Peut-on construire un réseau plus adapté aux faibles résolutions ? On traite ces deux questions successivement dans la suite de ce chapitre. Par ailleurs, on ne considère plus les résultats sur la base PPMI, qui n'est pas assez concluante par rapport à d'autres bases de données testées par la suite.

## 3.2 Transfert d'une résolution à l'autre

### 3.2.1 Intuition

On a montré précédemment que les performances de classification s'écroulaient drastiquement pour les représentations profondes pré-entraînées lorsque la résolution diminuait trop. Pour améliorer les performances sur les images faiblement résolues, on pourrait imaginer se servir des images hautement résolues. Comme les versions d'une même image à différentes résolution peuvent être assez proches visuellement, il convient de se demander si, dans l'espace des représentations, leurs vecteurs descripteurs le sont aussi. Si c'est le cas, la proximité des exemples dans l'espace des pixels devrait être conservée dans l'espace des représentations, et il est alors possible de transférer un modèle appris sur les descripteurs des images hautement résolues vers ceux des images faiblement résolues. L'étude d'un tel transfert permettrait de caractériser quelque peu l'espace dans lequel vivent les représentations profondes. De plus, ce transfert pourrait s'avérer bénéfique pour la reconnaissance des images faiblement résolues.

### 3.2.2 Différents scenarii testés

Pour cette étude, on considère différents scenarii d'apprentissage de classifieur pour la reconnaissance d'images à résolution $s \times s$ pixels. D'une part, on reporte le scenario sans transfert étudié plus haut, qui est l'expérience témoin. On s'intéresse ensuite à deux scenarii de transfert entre les représentations profondes extraites à différentes résolutions. On retient ici les représentations extraites après la première couche totalement connectée de VGG-M, *i.e.* les représentations VGG-M_fc1. En effet, ce sont les représentations profondes qui permettent d'obtenir les meilleures performances tout en étant relativement compactes. On utilise le même pré-traitement des images que précédemment. Voici la description détaillée des trois scenarii testés :

**ScenA : standard.** Le classifieur est appris sur les représentations VGG-M_fc1 extraites sur les images d'apprentissage à résolution $s \times s$.

**ScenB : apprentissage multi-résolution.** Le classifieur est appris sur un ensemble composé des représentations VGG-M_fc1 extraites sur les images d'apprentissage à résolution $s \times s$ ainsi que de celles extraites sur les mêmes images à résolution $224 \times 224$.

**ScenC : transfert pur.** Le classifieur est appris sur les représentations VGG-M_fc1 extraites sur les images d'apprentissage à résolution $224 \times 224$.

### 3.2.3 Résultats expérimentaux

On reporte sur la figure 3.8 le taux de bonne classification de chacun des trois scenarii sur les images de test à résolution $s \times s$ pixels. On montre d'une part que le scenario ScenC donne rapidement de mauvais résultats. Autrement dit, un classifieur appris sur des images haute résolution ne peut pas être directement transféré à une résolution plus faible sans une perte dramatique de performances. Pour des résolutions assez proches, le gap de performances est assez important : pour des images $150 \times 150$, ScenC obtient un taux de bonne classification de 45,9%, soit une perte de 14,9% par rapport à la méthode standard ScenA. Lorsque les images de test deviennent trop petites (moins de $50 \times 50$

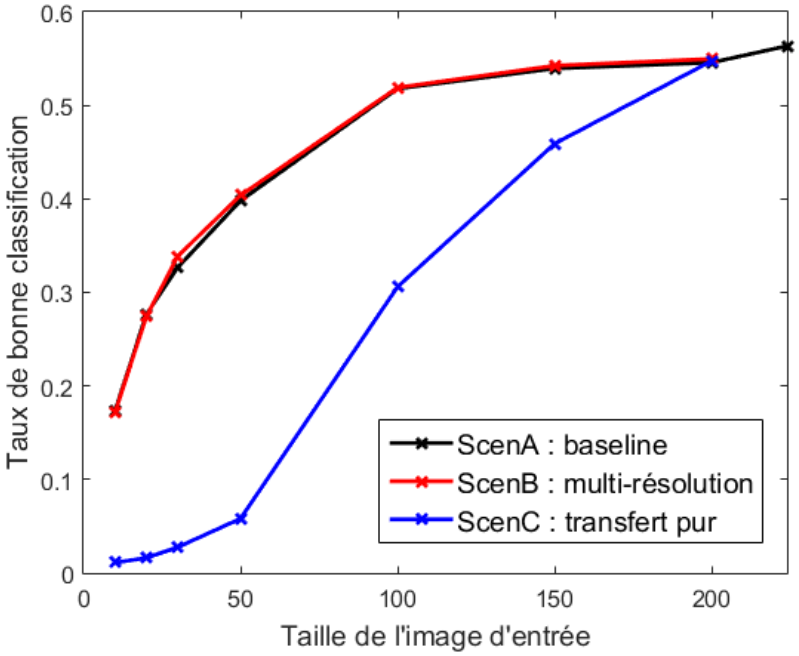



FIGURE 3.8 – Performances des représentations VGG-M_fc1 en transfert entre les résolutions sur FGVC-Aircraft.

pixels), les résultats sont proches de l'aléatoire (5,8% pour les images $50 \times 50$). Ce résultat montre que les représentations sont trop différentes d'une résolution à l'autre pour pouvoir brutalement transférer un modèle appris sur les grandes images.

D'autre part, on montre aussi que le fait d'ajouter des images hautement résolues dans l'ensemble d'apprentissage ne détériore pas les performances, mais ne permet pas de les améliorer significativement. Par exemple, le gain de performances sur des images de taille $100 \times 100$ n'est que de 0,1%.

### 3.2.4 Conclusion

Il ressort de nos expérimentations que l'utilisation naïve des données à haute résolution pour améliorer la reconnaissance des petites images n'est pas efficace. En effet, lorsqu'un classifieur est appris sur les représentations d'une image à deux résolutions différentes, les performances sont quasiment inchangées. Le transfert brutal d'un classifieur appris sur des données hautement résolues vers des données faiblement résolues quant à lui est catastrophique, car les représentations sont trop différentes.

Il nous reste maintenant à examiner la possibilité d'adapter un modèle spécifiquement pour la reconnaissance des petites images, puisqu'un processus naïf n'est d'aucun secours.

## 3.3 LR-CNN : Low-Resolution CNN

Dans les deux parties précédentes, on a mis en évidence l'intérêt des architectures profondes pour la classification d'images dans un contexte grain-fin. On a cependant montré que les représentations pré-entraînées souffraient du changement de résolution, et que la

prise en compte naïve de représentations pré-entraînées extraites sur les images haute-résolution ne permettait pas d'améliorer les performances. D'autre part, les réseaux de neurones profonds état de l'art classiquement utilisés pour les larges bases de données (Krizhevsky et al., 2012; Szegedy et al., 2015; He et al., 2016) se constituent généralement de plusieurs dizaines de millions de poids, et ne peuvent donc pas être convenablement appris sur des bases de moyenne envergure (*i.e.* quelques milliers d'images). Pour répondre à ce problème, nous proposons LR-CNN, une structure profonde adaptée à la reconnaissance d'images faiblement résolues dans un contexte grain-fin.

### 3.3.1 Stratégie d'adaptation de l'architecture

Dans cette partie, on précise la stratégie adoptée pour définir l'architecture de notre réseau LR-CNN. On présente également les différents modules utilisés dans l'état de l'art des réseaux de neurones profonds pour la reconnaissance d'images, abordés dans la section 2.3.2, en justifiant les choix des modules dans l'architecture proposée ainsi que les choix de paramétrage pour l'apprentissage.

#### Architecture

Dans cette partie, notre objectif n'est pas de proposer une nouvelle stratégie d'apprentissage profond, mais bien de tirer profit des divers réglages et réflexions sur les modèles profonds proposés récemment pour des images haute-résolution dans le cadre de notre chaîne basse-résolution. Basée sur la structure VGG-M, cette architecture est une version des structures adaptées à la classification d'images hautement résolues mise à l'échelle pour convenir au contexte de la faible résolution. Pour ce faire, on construit une structure profonde permettant l'extraction de représentations riches via trois couches de convolution et deux couches totalement connectées. Tous les blocs convolutionnels sont composés de 64 filtres, ce qui offre un large nombre de descripteurs d'image.

Les filtres de la première couche de convolution ont une taille $5 \times 5$, ce qui permet de calculer une convolution à une échelle globale de l'image d'entrée. Les réponses à ces filtres sont ensuite normalisés, puis une étape de max-pooling résume l'information locale dans chaque carte de réponse. On introduit du recouvrement dans ce processus, afin d'enrichir cette étape d'agrégation. La deuxième couche de convolution contient 64 filtres de taille $5 \times 5$. Un max-pooling est effectué de la même manière que dans la couche précédente. La dernière couche de convolution comprend des filtres de taille $3 \times 3$ : cette taille de filtres apparaît comme significative, puisqu'elle détermine quelle proportion de l'image d'entrée est associée à une réponse. Le nombre et la taille des filtres sont déterminés pour que le réseau couvre une proportion de l'image d'entrée qui permette de distinguer de petits détails discriminants. Ces couches convolutionnelles sont suivies d'une couche totalement connectée, qui calcule une combinaison des sorties de la couche précédente. Une dernière couche totalement connectée calcule les scores de l'image d'entrée pour chacune des classes. La structure complète est résumée dans le tableau 3.2, et on justifie ci-après les choix de normalisation et de fonctions d'activation utilisées dans cette architecture.

| Bloc | Conv1 | Conv2 | Conv3 | Fc4 | Fc5 |
|---|---|---|---|---|---|
| Détails | 64 filtres $5 \times 5$, pas 1<br>ReLU, norm. contraste<br>max. pool [3 3], pas 2 | 64 filtres $5 \times 5$, pas 1<br>ReLU<br>max. pool [3 3], pas 2 | 64 filtres $3 \times 3$ pas 1<br>ReLU<br>max. pool [3 3], pas 2 | sortie de taille 128<br>ReLU | sortie de taille $N_{classes}$<br>softmax |

TABLE 3.2 – Structure du réseau LR-CNN.

**Risque et fonctions de coût**

Les poids du réseau sont appris de sorte à ce que le réseau parvienne au mieux à remplir un objectif donné. D'un point de vue mathématique, la fonction calculée par le réseau est apprise pour minimiser un risque empirique, qui dépend d'une fonction de coût donnée. Cette fonction reflète le but de la tâche à accomplir : un réseau appris pour une tâche de classification d'image n'optimisera donc pas nécessairement la même fonction de coût qu'un réseau appris pour localiser les objets dans une image.

Pour la classification, la dernière couche totalement connectée produit autant de sorties que ce qu'il y a de classes à discriminer. Ainsi, pour une image donnée, la sortie $k$ de la dernière couche totalement connectée représente un score de l'image d'entrée pour la classe $k$. Ce score n'est cependant pas borné, et n'est donc pas directement interprétable tel quel : plusieurs fonctions se donnent ce rôle. Dans le cadre de la classification multi-classe mono-label, une fonction très couramment utilisée est le softmax (Krizhevsky et al., 2012; Simonyan and Zisserman, 2015; Szegedy et al., 2015; He et al., 2016), qui s'exprime comme ceci :

$$\text{softmax} \ : \tilde{z}_k = \frac{e^{z_k}}{\sum_{k'=1}^{N_{classes}} e^{z_{k'}}}. \tag{3.1}$$

Cette fonction présente deux caractéristiques majeures : d'une part, elle permet de ramener tous les scores entre 0 et 1. Les sorties du softmax peuvent dès lors se lire comme une probabilité, ce qui facilite l'interprétation : c'est la raison pour laquelle ce type de fonction est souvent appelée "fonction de prédiction". D'autre part, elle lie les scores de toutes les classes entre eux, et la somme des scores par classe est de 1. De fait, lorsque le score d'une classe $k$ est élevé, les autres scores sont mathématiquement diminués, ce qui convient parfaitement au cadre de la reconnaissance multi-classe mono-label. Le score d'une classe $k$ donné peut donc se lire comme la probabilité que la classe globale de l'image soit $k$. Une fonction de coût classiquement utilisée après le softmax est le coût logarithmique (ou *log-loss*), qui s'écrit pour chaque image

$$\text{log-loss} \ : loss = -\ln(\tilde{z}_c) \tag{3.2}$$

où $c$ est la classe vérité-terrain de l'image d'entrée. Le coût global du réseau est alors la somme des coûts de toutes les images d'apprentissage. Ce coût est bien toujours positif car $\tilde{z}_c \leq 1$, et il est d'autant plus élevé que $\tilde{z}_c$ est proche de 0. Autrement dit, le prix à payer est d'autant plus élevé que le score de la vraie classe est faible, ce qui est bien le comportement voulu.

Certaines tâches requièrent néanmoins de pouvoir distinguer plusieurs labels au sein d'une même image. Pour ce faire, certains travaux proposent d'utiliser la sigmoïde (Oquab et al., 2015; Durand et al., 2016) :

$$\text{sigmoïde} \ : \tilde{z}_k = \frac{1}{1 + e^{-z_k}}. \tag{3.3}$$

Comme le softmax, cette fonction de prédiction remappe chaque sortie en un score compris entre 0 et 1. Les scores de toutes les classes ne sont cependant plus liés, si bien que plusieurs classes peuvent avoir simultanément un score proche de 1. Le score de la classe $k$ peut donc être interprété comme la probabilité que la classe $k$ soit présente dans l'image, indépendamment des scores des autres classes. Ce comportement est bien adapté au cas des images multi-labels. La fonction de coût associée à la prédiction par sigmoïde est généralement l'entropie croisée (ou *cross-entropy*) :

$$\text{entropie croisée} \ : loss = -\sum_{k=1}^{N_{classes}} y_k \ln(\tilde{z}_k) \tag{3.4}$$

où $y_k = 1$ si la classe $k$ est présente dans l'image, et 0 sinon. De la même manière que pour le coût logarithmique, l'entropie croisée est d'autant plus forte que les scores des classes présentes sont faibles. On remarque par ailleurs que dans le cas où une seule classe est présente dans l'image, on retrouve la formulation du coût logarithmique.

Comme évoqué plus haut, la vision par ordinateur comporte des multitudes de tâches spécifiques, et pour chaque tâche, de nombreux travaux se sont attachés à développer des fonctions de coût adaptées au problème posé. On peut par exemple citer des coûts de régression sur la position des boîtes englobantes pour les tâches de localisation (Erhan et al., 2014; Girshick, 2015; Zhang et al., 2014), les fonctions de coût prenant en compte des réseaux de traitement du langage pour les tâches de sous-titrage (Johnson et al., 2016; Mao et al., 2015; Vinyals et al., 2015) ou encore les coûts spécifiques aux tâches de segmentation sémantique (Liang et al., 2016; Long et al., 2015). Pour nos travaux, on s'inscrit dans la continuité de la très large majorité des travaux de la littérature en retenant la fonction softmax accompagnée du coût logarithmique.

#### Optimisation par rétro-propagation du gradient de l'erreur

Les poids d'un réseau à $L$ couches, notés $\mathbf{w} = \{\mathbf{w}_l\}_{l=1..L}$ où $\mathbf{w}_l$ est la matrice de poids associée à la couche $l$ du CNN, sont appris pour que le réseau calcule une fonction $f$, dépendant directement de la tâche traitée. Pour que cette fonction atteigne son but, elle est optimisée pour minimiser le risque empirique $R_{emp}$, dépendant d'une fonction de coût $\ell$, qui exprime l'écart entre la prédiction faite par le réseau $\hat{y}_i = f(\mathbf{w}, I_i)$, et la véritable réponse visée $y_i$ pour chaque exemple (ici, le label de l'image d'entrée). Dans les réseaux de neurones convolutionnels, l'erreur globale du réseau sur la totalité des images d'apprentissage s'écrit souvent comme la moyenne des coûts individuels pour chaque image :

$$R_{emp}(\mathbf{w}, I_i, y_i) = \frac{1}{N} \sum_{i=1}^{N} \ell(f(\mathbf{w}, I_i), y_i) \tag{3.5}$$

où $I_i$ est l'image à laquelle est associé le vecteur de labels $y_i$ (par exemple, si $I_i$ est de classe $c$, $y_i$ est un vecteur de zéros avec $y_i(c) = 1$). Lorsque l'erreur empirique est dérivable

presque partout, pour optimiser ce risque, on peut apprendre les poids $\mathbf{w}$ par descente de gradient. Pour ce faire, on a besoin du gradient de la fonction $f$ par rapport à chaque poids $w$. En prenant en compte que le CNN est constitué de couches successives, on peut écrire :

$$\begin{aligned} f(\mathbf{w}, I_i) &= f_L(\mathbf{w}_L, x_L) \\ &= f_L(\mathbf{w}_L, f_{L-1}(\mathbf{w}_{L-1}, x_{L-1})) \\ &= f_L(\mathbf{w}_L, f_{L-1}(\mathbf{w}_{L-1}, ...f_2(\mathbf{w}_2, f_1(\mathbf{w}_1, I_i)))) \\ &= f_L \circ f_{L-1} \circ ... \circ f_2 \circ f_1(\mathbf{w}, I_i) \end{aligned} \tag{3.6}$$

où chaque couche $l$ de poids $\mathbf{w}_l$ calcule une fonction $f_l$ sur l'entrée $x_l$, qui est la sortie de la couche précédente : $f_l(\mathbf{w_l}, x_l)$ est donc la sortie du réseau à la couche $l$. Le gradient de la fonction totale du CNN par rapport à la matrice de poids de la couche $l$ s'écrit alors :

$$\begin{aligned} \frac{\partial f}{\partial \mathbf{w}_l}(\mathbf{w}, I_i) &= \frac{\partial}{\partial \mathbf{w}_l}\Big(f_L \circ f_{L-1} \circ ... \circ f_2 \circ f_1(\mathbf{w}, I_i)\Big) \\ &= \frac{\partial\Big(f_L \circ f_{L-1} \circ ... \circ f_{l+1}\Big)}{\partial \mathbf{x}_{l+1}} \frac{\partial f_l(\mathbf{w}_l, x_l)}{\partial \mathbf{w}_l} \\ &= \frac{\partial\Big(f_L \circ f_{L-1} \circ ... \circ f_{l+2}\Big)}{\partial \mathbf{x}_{l+2}} \frac{\partial f_{l+1}(\mathbf{w}_{1+1}, x_{l+1})}{\partial \mathbf{x}_{l+1}} \frac{\partial f_l(\mathbf{w}_1, x_l)}{\partial \mathbf{w}_l} \end{aligned} \tag{3.7}$$

donc en poursuivant la décomposition,

$$\frac{\partial f}{\partial \mathbf{w}_l}(\mathbf{w}, I_i) = \frac{\partial f_L(\mathbf{w}_L, x_L)}{\partial \mathbf{x}_L} \frac{\partial f_{L-1}(\mathbf{w}_{L-1}, x_{L-1})}{\partial \mathbf{x}_{L-1}} ... \frac{\partial f_{l+1}(\mathbf{w}_{l+1}, x_{l+1})}{\partial \mathbf{x}_{l+1}} \frac{\partial f_l(\mathbf{w}_l, x_l)}{\partial \mathbf{w}_l}. \tag{3.8}$$

Ainsi, pour mettre à jour les poids de la couche $l$, il suffit d'une part de disposer des dérivées des couches plus profondes par rapport à leurs entrées respectives, et d'autre part de calculer $\frac{\partial f_l(\mathbf{w}_l, x_l)}{\partial \mathbf{w}_l}$. De fait, il suffit de commencer par la couche la plus profonde $L$, puis de remonter à la couche précédente $L-1$ en conservant l'information calculée à l'étape précédente, *i.e.* les dérivées par rapport à l'entrée et par rapport aux poids. On fait donc une première passe *forward* pour calculer les sorties de chaque couche, puis la passe *backward* permet de faire « remonter » le gradient de l'erreur à des couches de plus en plus superficielles. Ce mécanisme lui vaut le nom d'apprentissage par rétro-propagation du gradient (LeCun et al., 1989). Une illustration du processus est proposée en figure 3.9.

Afin de traiter de larges bases d'images, la descente de gradient stochastique (ou SGD pour *Stochastic Gradient Descent*) par batches consiste à optimiser les poids du réseau itérativement sur des sous-ensembles des données d'apprentissage (les *batches*) tirés aléatoirement. Cette technique a été utilisée avec succès pour l'apprentissage de réseaux de neurones sur de larges bases de données (Krizhevsky et al., 2012; Szegedy et al., 2015; He et al., 2016). La manière de rétro-propager le gradient a elle aussi fait l'objet de nombreux travaux de recherche. La descente de gradient classique est par exemple assez lente et très facilement piégée au niveau de point-selles (*saddle points*). La descente de gradient avec momentum de (Rumelhart et al., 1986) a ensuite permis d'accélérer la convergence de l'apprentissage en introduisant une sorte d'inertie, accélérant la descente du gradient et ralentissant sa remontée. Le Nesterov Accelerated Gradient de (Sutskever

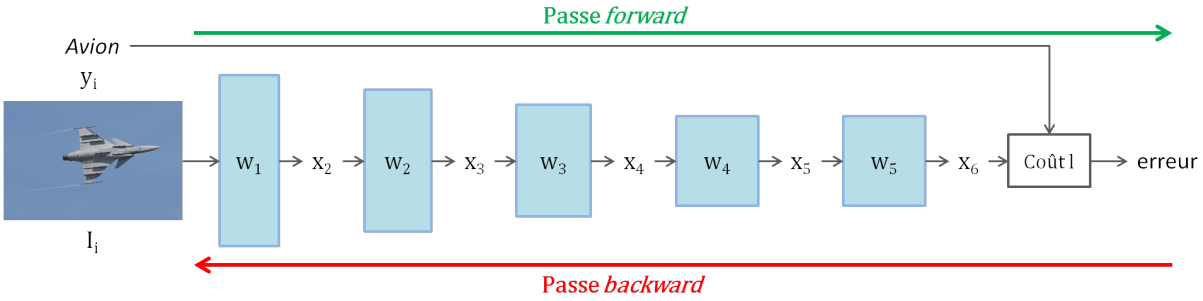



FIGURE 3.9 – Illustration de l'algorithme de rétro-propagation pour l'apprentissage d'un réseau de neurones.

et al., 2015) adapte la formulation en calculant le gradient au point mis à jour après chaque époque, ce qui accélère la convergence. Afin d'éviter les problèmes liés aux point-selles et aux vallées, la méthode AdaGrad (Duchi et al., 2011) introduit une mémoire sur les gradients précédents, accélérant très fortement la descente vers une direction long-temps restée « plate ». D'autres méthodes comme RMSProp (Tieleman and Hinton, 2012), AdaDelta (Zeiler, 2012) ou encore ADAM (Kingma and Ba, 2015) ont également permis d'améliorer la convergence de l'apprentissage. Dans nos travaux, nous nous sommes intéressés à la descente de gradient stochastique avec momentum, qui est une des méthodes les plus populaires pour l'apprentissage des réseaux de neurones du fait de sa simplicité et de ses bons résultats (Krizhevsky et al., 2012; Szegedy et al., 2015; He et al., 2016). Selon ce modèle, un poids $w^t$ à l'instant $t$ est mis à jour de la manière suivante :

$$w^{t+1} = w^t - \eta \Delta w^t \tag{3.9}$$

où

$$\Delta w^t = -\gamma \Delta w^{t-1} + \lambda w^t + \frac{\partial R_{emp}}{\partial w} \tag{3.10}$$

avec $\gamma$ le momentum, $\lambda$ le *weight decay*, $\eta$ le pas d'apprentissage et $\frac{\partial R_{emp}}{\partial w}$ le gradient de l'erreur empirique du réseau sur le batch d'images considéré par rapport au paramètre $w^t$. L'équation 3.10 traduit bien le fait que le momentum et le weight decay régulent tous deux une forme d'inertie du poids à l'état précédent, et le pas d'apprentissage régule de combien le poids est mis à jour par rapport au gradient de l'erreur.

Comme précisé auparavant, les poids de toutes les couches sont appris sur une base de données. Puisque le nombre de poids peut être largement supérieur au nombre d'images d'apprentissage, plusieurs méthodes ont été proposées pour éviter le sur-apprentissage. Notamment, le dropout (Srivastava et al., 2014) consiste à « éteindre » chaque poids avec une probabilité fixée durant l'apprentissage : à chaque époque, chaque poids a une probabilité donnée d'être mis à jour. Une plus faible quantité de paramètres est donc apprise sur les données d'entrée, ce qui limite de fait le sur-apprentissage. Cette technique est le plus souvent utilisée sur les couches contenant le plus de paramètres, *i.e.* les couches totalement connectées. D'autres techniques ont également émergé, comme le dropconnect (Wan et al., 2013) où ce sont les connexions (et non plus les poids) qui sont gelés. Dans

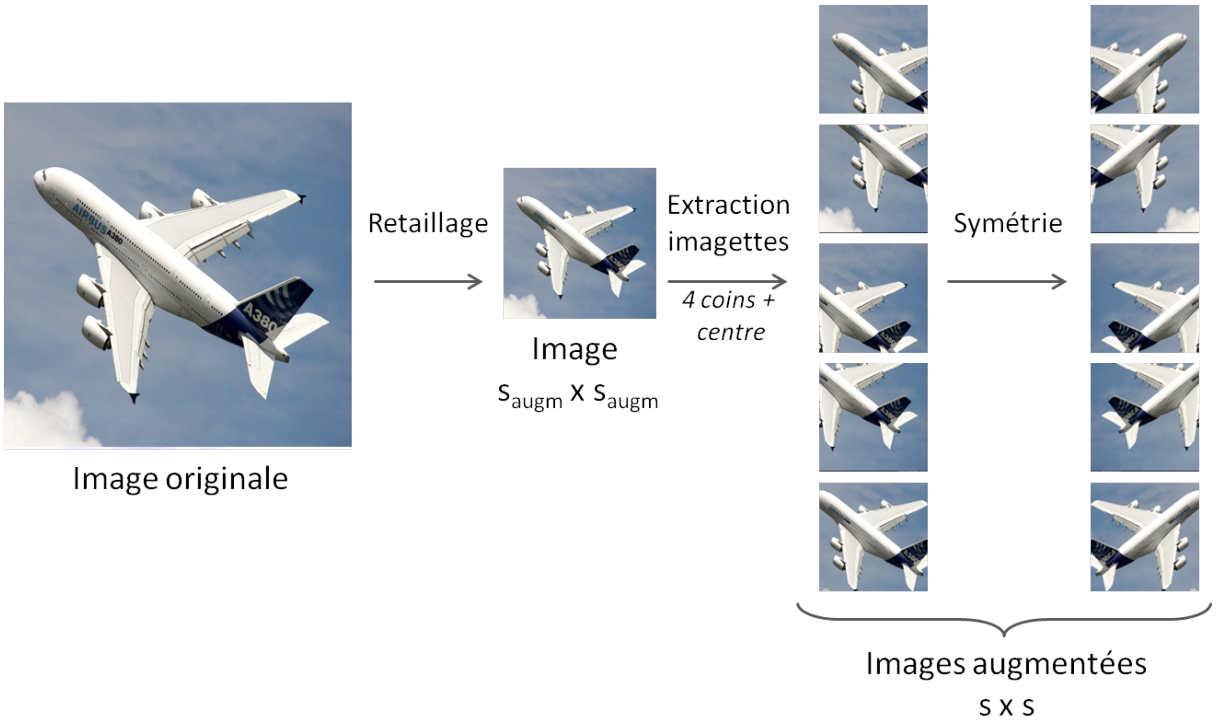



FIGURE 3.10 – Schéma illustrant l'augmentation de données.

nos travaux, on utilise le dropout, qui est la technique la plus répandue (Szegedy et al., 2015) et donne de très bons résultats sur les bases de données testées.

Une autre manière de maîtriser le sur-apprentissage est l'augmentation de données (ou *data augmentation*) (Krizhevsky et al., 2012; He et al., 2016). Proposée par Hinton et al. (2012), cette technique consiste à démultiplier les images d'apprentissage en leur faisant subir des transformations qui laissent inchangé le label, comme par exemples de petites translations, des rotations ou encore l'ajout d'un bruit additif sur l'image. Cette technique consiste à retailler les images originales à un format $s_{augm} \times s_{augm}$, puis de ce carré sont extraites $N_{augm}$ imagettes de taille $s \times s$ (la taille d'entrée du réseau). On peut également retourner ces images autour des axes verticaux ou horizontaux. Une illustration de cette technique est proposée en figure 3.10. Cela permet d'agrandir artificiellement l'ensemble d'apprentissage mais également de rendre le réseau invariant à certaines transformations géométriques, et donc de réduire les risques de sur-apprentissage. Dans nos travaux, on utilise cette technique en extrayant des images de chaque coin et du centre de chaque image d'apprentissage.

**Fonctions d'activation**

Afin d'augmenter la complexité du modèle appris, les différentes couches peuvent être suivies de non-linéarités, aussi appelées fonctions d'activation. Inspirées d'une approche biologique, ces fonctions rappellent le comportement des neurones dans le cerveau humain, qui ne laissent passer une information que si le potentiel d'activation en entrée est assez élevé. De nombreuses fonctions de non-linéarité ont été utilisées dans les réseaux

de neurones. Historiquement, la fonction sigmoïde $\sigma(x) = \frac{1}{1+e^{-x}}$ a longtemps été utilisée puisqu'elle permet de projeter toute entrée réelle dans l'intervalle $[0; 1]$. Cette fonction est assez proche du comportement biologique de notre cerveau, puisque les valeurs d'entrée trop faibles sont réduites à 0, alors que les valeurs les plus fortes saturent le neurone. La fonction tangente hyperbolique $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ a ensuite supplanté la sigmoïde du fait d'être centrée en 0.

L'inconvénient majeur de ces deux fonctions est le fait que le gradient s'estompe dès que les valeurs d'entrée sont trop faibles ou trop élevées. De fait, lors de l'étape d'apprentissage, le gradient rétro-propagé est écrasé par cette non-linéarité, et plus rien n'est propagé aux couches plus superficielles : c'est le problème dit du gradient évanescent (ou *vanishing gradient*). Pour pallier cette difficulté, Nair and Hinton (2010) démontrent l'intérêt de la ReLU au sein des CNN, qui s'exprime comme suit :

$$ReLU(x) = \begin{cases} x & \text{si } x > 0, \\ 0 & \text{si } x \leq 0. \end{cases}$$

Le gradient de cette fonction étant l'identité lorsqu'il y a lieu de rétro-propager, on s'affranchit du problème du gradient évanescent. C'est aujourd'hui la fonction de non-linéarité la plus classiquement utilisée (Krizhevsky et al., 2012; He et al., 2016; Simonyan and Zisserman, 2015). Dans nos travaux, on utilise cette fonction d'activation pour sa simplicité et les bons résultats qu'elle permet d'obtenir.

Cette fonction a fait l'objet de nombreux travaux complémentaires, et plusieurs améliorations ont vu le jour. Pour contrer l'annulation de toutes les valeurs négatives par la ReLU, Blot et al. (2016) proposent un modèle dupliquant les valeurs d'entrée de cette non-linéarité afin que la couche d'agrégation permette ensuite de retenir la plus petite valeur négative en plus de la plus grande valeur positive. Maas et al. (2013) soulèvent le fait que la ReLU ne permet pas de rétro-propager lorsque le neurone n'est pas actif (*i.e.* lorsque l'entrée du neurone est négative). De fait, dans certains cas, ce neurone pourrait ne jamais être activé. Pour contrer ce problème, les auteurs proposent la Leaky ReLU, qui s'exprime comme suit :

$$LeakyReLU(x) = \begin{cases} x & \text{si } x > 0, \\ \alpha x & \text{si } x \leq 0, \end{cases}$$

avec $\alpha$ une valeur assez faible (typiquement 0,01). La Parametric ReLU (ou PReLU) proposée par (He et al., 2015) généralise cette approche en considérant une valeur de $\alpha$ propre à chaque neurone, apprise lors de l'apprentissage. (Clevert et al., 2016) propose aussi l'Exponential Linear Unit ou ELU, qui s'exprime ainsi :

$$ELU(x) = \begin{cases} x & \text{si } x > 0, \\ \alpha(e^x - 1) & \text{si } x \leq 0. \end{cases}$$

Tout comme la Leaky ReLU, cette fonction d'activation ne s'annule pas en les valeurs négatives, ce qui évite le problème des neurones morts. En revanche, elle sature pour des valeurs assez faibles, *i.e.* le gradient rétro-propagé s'annule pour des valeurs trop faibles, ce qui rend le réseau plus robuste au bruit. Pour une meilleure vue d'ensemble, on représente sur la figure 3.11 l'effet des différentes fonctions d'activation sur la sortie du neurone (à gauche) lors de la passe d'exécution, ainsi que sur le gradient rétro-propagé (à droite) durant la phase d'apprentissage.

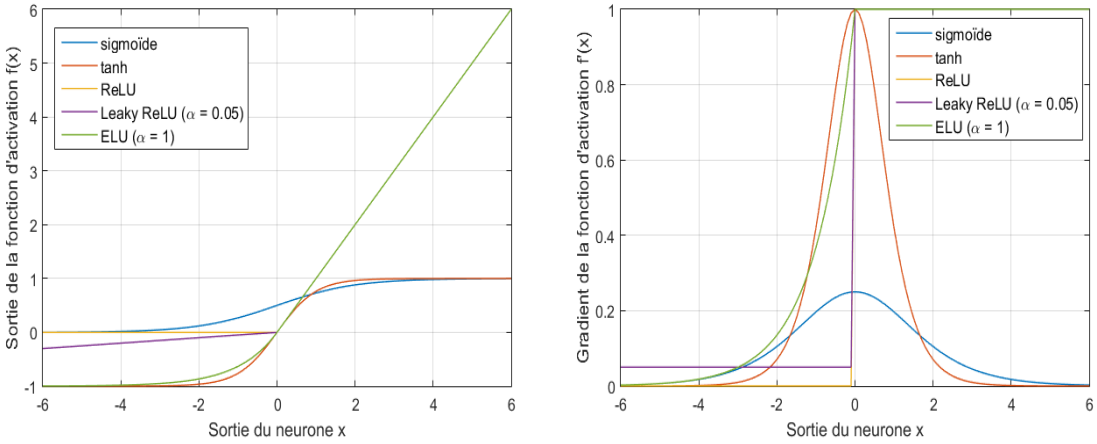



FIGURE 3.11 – Représentation des différentes fonctions de non-linéarité (à gauche) ainsi que leur dérivée (à droite) en fonction de la valeur de sortie du neurone.

### Agrégation

Une étape d'agrégation (ou *pooling*) suit généralement une couche de convolution suivie de sa fonction d'activation. Cette fonction résume l'information contenue dans une partie de la sortie de la couche précédente. Ainsi, après chaque étape de pooling, une valeur de sortie correspond à une plus grande région de l'image d'entrée : on peut donc considérer ces étapes de pooling comme un changement d'échelle au sein de l'image d'entrée. La méthode la plus usitée est le max-pooling (Krizhevsky et al., 2012; Simonyan and Zisserman, 2015; He et al., 2016), qui ne conserve que la valeur maximale présente dans la zone considérée. Cela correspond à l'intuition qu'on cherche à retenir la plus forte activation de chacun des filtres. La variante maxout, introduite par Goodfellow et al. (2013), propose un type d'agrégation similaire, en retenant, pour une zone spatiale donnée, la valeur maximales à travers toutes les cartes de réponses. D'autres méthodes d'agrégation du même type ont été utilisées, comme par exemple l'*average pooling* (Szegedy et al., 2015), une technique selon laquelle la valeur agrégée est la moyenne des valeurs d'entrée considérées. Cette méthode induit généralement une légère perte de performances lorsqu'elle est utilisée dans les couches intermédiaires, et est le plus souvent utilisée en fin de réseau pour remplacer les couches totalement connectées (Szegedy et al., 2015). Ce processus introduit une forme d'invariance aux translations locales, puisque la sortie reste inchangée quelle que soit la répartition des valeurs dans la zone de la carte d'activations en entrée du pooling.

D'autres méthodes d'agrégation ont été proposées, notamment pour pallier la perte d'information spatiale induite par le max-pooling. Dans SPP-Net (He et al., 2014), les auteurs proposent d'intégrer l'intuition de Lazebnik et al. (2006) (décrite dans la figure 2.3) au sein des CNN. Les convolutions et max-pooling sont successivement appliqués à l'image d'entrée du réseau, puis la carte de réponses du dernier filtre est découpée en un nombre fixé de zones spatiales de différentes échelles. Dans chacune de ces zones, on agrège les réponses ; le vecteur concaténé est alors l'entrée des couches totalement connectées. Comme pour le SPM utilisé dans les méthodes type BoW, ce modèle permet d'une part de conserver l'information spatiale, et d'autre part de calculer des représentations sur des

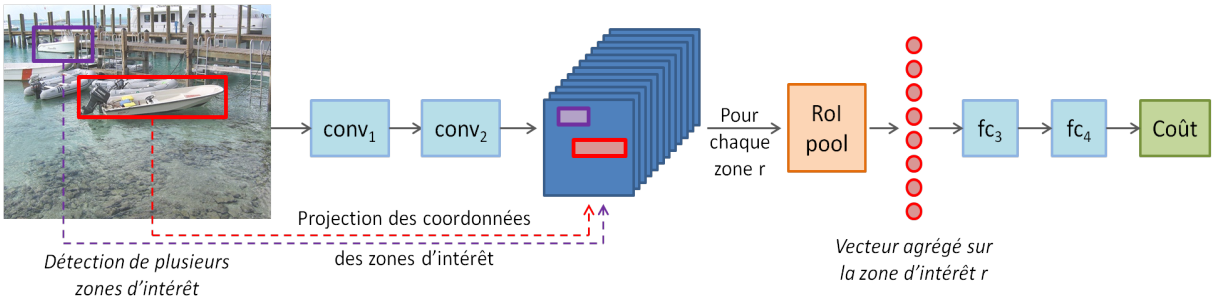



FIGURE 3.12 – Schéma du RoI pooling de Girshick (2015). Exemple avec un réseau à 2 couches convolutives et 2 couches totalement connectées.

images de taille variable.

L'architecture Fast R-CNN (Girshick, 2015) introduit le RoI-pooling (*Region of Interest pooling*, ou agrégation sur une région d'intérêt) dans le cadre de la localisation d'objets. Cette méthode consiste à projeter les coordonnées d'une région d'intérêt au niveau des cartes d'activation de la dernière couche de convolution. Pour chaque région d'intérêt, seules les réponses correspondant aux coordonnées projetées sont agrégées. Cette méthode permet d'accélérer considérablement l'exécution et l'apprentissage du réseau par rapport à la méthode naïve (Girshick et al., 2014) tout en conservant l'information spatiale. On représente sur la figure 3.12 un schéma décrivant ce processus. La méthode d'agrégation proposée dans Inside-Outside net (Bell et al., 2016) applique le RoI-pooling à plusieurs niveaux de profondeur du réseau. Pour chaque zone d'intérêt et chaque couche de convolution, un vecteur est extrait par RoI-pooling, puis ces vecteurs sont ensuite eux-mêmes agrégés pour former le vecteur d'entrée des couches totalement connectées. Ce processus permet d'extraire de l'information à différentes échelles dans l'image d'entrée. Ces méthodes d'agrégation sont cependant spécifiques à la localisation, c'est pourquoi on s'intéresse dans nos travaux aux techniques de type max-pooling.

#### Bilan des choix effectués pour l'apprentissage de LR-CNN

L'architecture de notre réseau LR-CNN, inspirée des CNN de l'état de l'art pour la reconnaissance d'images fortement résolues tels que VGG-M (Chatfield et al., 2014), vise à contenir assez peu de paramètres, qui puissent être entraînés sur les données d'intérêt. Dans cette optique, on cherche par ailleurs à conserver assez de profondeur afin d'extraire des représentations de niveau sémantique assez élevé, c'est pourquoi on choisit notamment d'utiliser deux couches totalement connectées plutôt qu'une seule. De plus, on conserve certains modules présents dans l'architecture de VGG-M, comme la présence de max-pooling. Ces choix architecturaux seront par ailleurs explorés et validés expérimentalement dans la section 3.3.3.

Pour optimiser les poids de ce réseau, on considère une prédiction softmax avec fonction de coût logarithmique, fonction de coût majeure pour la classification multi-classe mono-label. Les poids sont appris via un algorithme d'apprentissage par descente de gradient stochastique avec momentum, qui a été utilisée avec succès dans de nombreux domaines. Afin d'éviter le sur-apprentissage, on utilise une étape de dropout sur l'avant-dernière couche totalement connectée. De plus, lorsque le nombre d'images d'apprentissage est faible, on réalise de l'augmentation de données en amont du réseau pour éviter le sur-

apprentissage. Dans notre cas, les images sont réduites à une taille de $37 \times 37$ pixels, dans lesquelles cinq fenêtres de taille $32 \times 32$ sont extraites (une dans chaque coin et une au centre de l'image). Chacune de ces fenêtres est retournée autour de l'axe vertical. Ainsi, pour chaque image originelle, on fournit dix images légèrement translatées et éventuellement miroir de l'image d'origine.

### 3.3.2 Description des bases de données

On s'intéresse à la classification de petites images dans un contexte grain-fin. Pour ce faire, on évalue notre LR-CNN sur FGVC-Aircraft (Maji et al., 2013), déjà explorée dans la partie 3.1 et détaillée dans la section 3.1.1. En effet, cette base de données consiste à discriminer différents types d'avions, ce qui en fait une tâche assez proche des intérêts opérationnels de Thales Optronique. On considère également une deuxième base de données orientée grain-fin : UPMC-Food-101 (Wang et al., 2015b). Ces deux bases de données contiennent un nombre similaire de classes, cependant le nombre d'images d'apprentissage de UPMC-Food-101 est très supérieur au nombre d'images de FGVC-Aircraft.

UPMC-Food-101 (Wang et al., 2015b) est une large base de données contenant près de 100 000 images de plats cuisinés, et réparties en 101 classes. Le protocole proposé dans (Wang et al., 2015b) prévoit de choisir aléatoirement 600 images par classe pour l'apprentissage, et le reste pour le test. Ainsi, on dispose de 60 600 images d'apprentissage, et de 25 935 images de test.

Cette base de données a la particularité de contenir du bruit d'annotation, du fait qu'elle est constituée d'images récoltées sur le web. Cette base de données permet donc de mesurer la robustesse de notre réseau face à ce phénomène. On présente sur la figure 3.13 des images tirées de 6 classes de cette base de données, qui montrent l'aspect grain-fin de cette base (*e.g.* les classes *tiramisù* et *strawberry shortcake* sont assez difficiles à discriminer) ainsi que le type de bruit d'annotation présent dans la base (*e.g.* images des ingrédients de la recette pour le *beef tartare* ou adaptation du plat à un autre parfum comme pour le *tiramisù*, qui devient alors proche d'une présentation atypique de *strawberry shortcake*). D'un point de vue opérationnel, la difficulté d'annoter des images acquises sur le terrain peut engendrer quelques erreurs d'annotation dans les bases de données réelles, ce qui justifie l'intérêt de cette base par rapport au contexte Thales.

### 3.3.3 Résultats expérimentaux

#### FGVC-Aircraft

Sur cette base de données, on fixe le weight decay à 0,01 et le momentum à 0,9. Pour éviter le sur-apprentissage, le taux de dropout est fixé à 50%, et on utilise de l'augmentation de données. Le pas d'apprentissage est fixé à $10^{-2}$, puis graduellement diminué jusqu'à $10^{-5}$ lorsque l'erreur d'apprentissage stagne.

On reporte les résultats dans le tableau 3.3. Sur cette base de données, notre modèle obtient un taux de bonne classification de 39,8%, dépassant largement ceux des représentations profondes pré-entraînées VGG-M_fc1 (32,7%) et VGG-M_fc2 (27,2%) étudiées plus haut. On montre ainsi que notre LR-CNN parvient à capturer de petits détails, apportant ainsi un gain significatif sur les réseaux pré-entraînés dans le contexte de la reconnaissance orientée grain-fin de petites images.

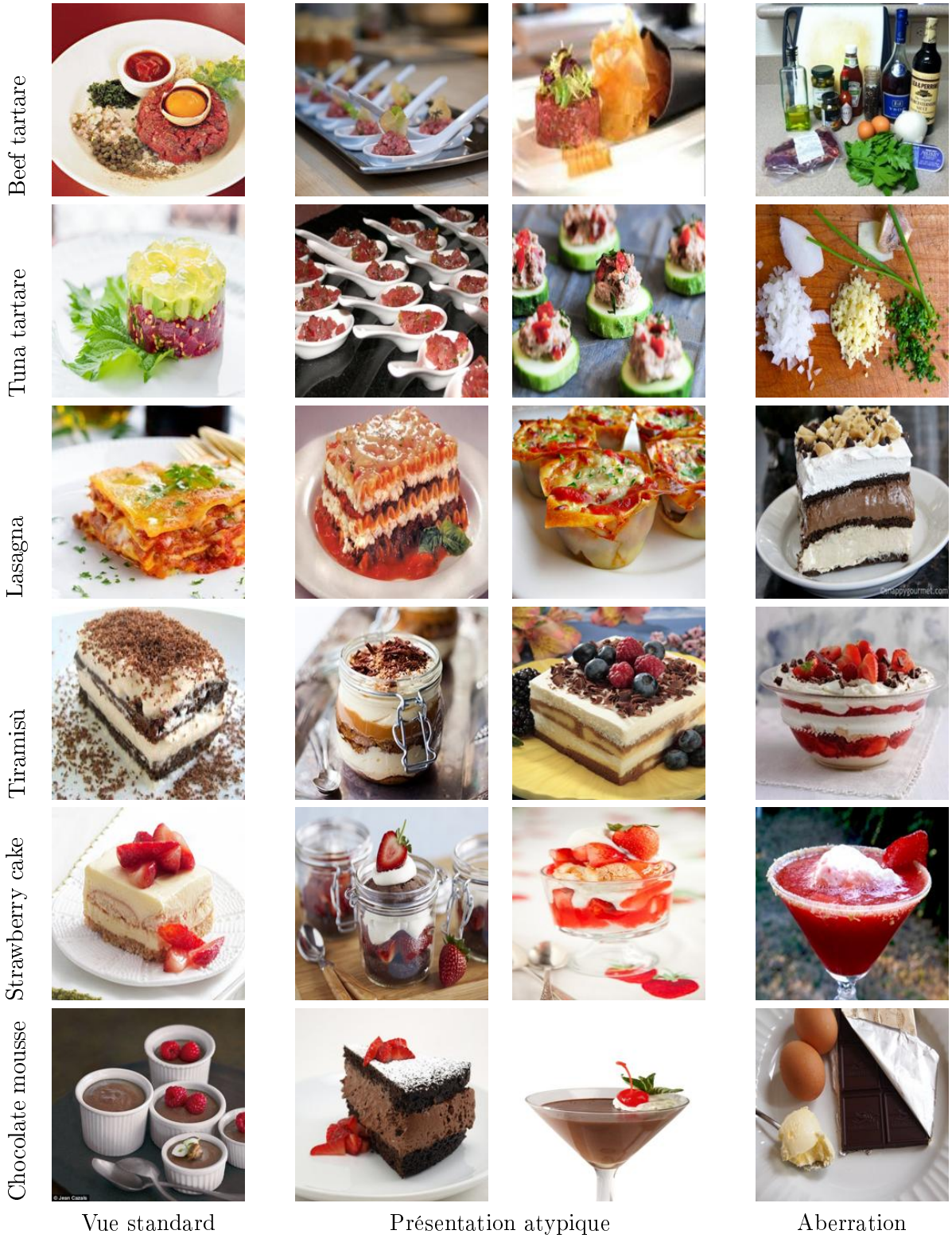



FIGURE 3.13 – Exemples d'images de 6 classes de UPMC-Food-101 (Wang et al., 2015b). Malgré la diversité des classes (*beef tartare* vs. *tiramisù*), certaines se ressemblent (*e.g. tiramisù* vs. *strawberry shortcake*), et la variabilité intra-classe peut être élevée du fait des présentations originales proposées. On reporte plusieurs exemples mal annotés (*i.e.* des aberrations), ce qui montre la présence d'un certain bruit d'annotation dans la base.

| Méthode de classification | Taux de bonne classification |
|---|---|
| VGG-M_fc1 | 32,7% |
| VGG-M_fc2 | 27,2% |
| LR-CNN, 1 fc | 33,6% |
| LR-CNN, average pool | 33,8% |
| LR-CNN, pool sans recouvrement | 35,1% |
| LR-CNN, conv3 $5 \times 5$ | 38,2% |
| **LR-CNN** | **39,8%** |

TABLE 3.3 – Performances de notre réseau LR-CNN sur FGVC-Aircraft (Maji et al., 2013).

On s'est intéressé de plus près au détail de notre architecture. Pour ce faire, on reporte dans le tableau 3.3 les résultats en utilisant la même structure (décrite dans le tableau 3.2), et en changeant à chaque expérience un seul des éléments de cette architecture. On montre ainsi l'intérêt d'utiliser deux couches totalement connectées (ligne *LR-CNN, 1 fc*), puisqu'en supprimant la couche Fc4, les performances chutent de 6,2%. Le paramétrage des blocs d'agrégation revêt également une certaine importance. Par exemple, en utilisant des zones d'agrégation de taille [2 2] avec un pas de 2 dans les étapes de *pooling*, il n'y a plus de recouvrement entre les zones agrégées, ce qui diminue les performances de 4,7% (ligne *LR-CNN, pool sans recouvrement*). Enfin, la taille des filtres apparaît comme un élément important de l'architecture, puisque les performances diminuent de 1,6% lorsque les filtres de conv3 sont agrandis à $5 \times 5$.

On peut également optimiser ces performances en utilisant notre réseau LR-CNN comme un extracteur de représentations profondes. Par exemple, en considérant l'ensemble des couches de convolution comme un extracteur de représentations profondes, on peut extraire des représentations de taille 2 304 sur lesquelles est appris un SVM. Dans ce cas, on obtient sur cette base un taux de bonne classification de 44,8%. Bien que ce processus soit très peu employé pour des réseaux pré-appris sur une base externe et contenant plusieurs dizaines voire centaines de millions de paramètres, on peut s'intéresser aux représentations pré-entraînées extraites au niveau des couches de convolution de VGG-M, dont on a montré qu'elles atteignaient les meilleures performances dans la section 3.1.4. On montre que notre réseau atteint des performances compétitives avec les meilleures représentations pré-entraînées (*e.g.* les représentations VGG-M_conv3, contenant plus de 86 000 dimensions, atteignent 44,3%). De même, les FV, de plus de 344 000 dimensions, atteignent un score de 42,4%. En optimisant encore notre méthode, on parvient donc à obtenir des résultats compétitifs voire meilleurs que les autres représentations testées, en proposant pour autant des descripteurs beaucoup plus compacts (notamment près de 150 fois plus petits que les FV).

On a donc montré l'intérêt de notre architecture par rapport à des représentations profondes pré-entraînées pour la classification des petites images sur FGVC-Aircraft. En effet, notre réseau parvient à obtenir de meilleures performances que les représentations profondes extraites d'un réseau pré-entraîné sur une base externe, et en optimisant davantage notre méthode, on parvient à des résultats encore meilleurs que les autres représentations testées tout en proposant des représentations compactes. Cette optimisation complémentaire permet ici de mettre en lumière l'aspect compact de notre méthode, ce-

| VGG-M_fc1 | LR-CNN |
|---|---|
| 28,9% | 30,7% |

TABLE 3.4 – Performances de notre réseau LR-CNN sur UPMC-Food-101 (Wang et al., 2015b).

pendant l'intérêt d'un réseau de neurones réside dans l'apprentissage de bout en bout des poids de toutes les couches. Ainsi, dans la suite, on ne s'intéressera qu'aux résultats obtenus par le réseau complet.

#### UPMC-Food-101

Sur cette base de données, du fait du large nombre d'images d'apprentissage (suffisant par rapport au nombre de poids à apprendre dans notre réseau LR-CNN), le taux de dropout est fixé à 40%, et on n'utilise pas d'augmentation de données. Les hyper-paramètres d'apprentissage sont les mêmes que sur FGVC-Aircraft, en prenant toutefois un momentum de 0,95. On s'intéresse ici aux représentations profondes VGG-M_fc1.

On reporte les résultats dans le tableau 3.4. Sur cette base de données, notre réseau LR-CNN est meilleur que VGG-M_fc1 de 1,8%. On montre ainsi que notre architecture parvient à atteindre de meilleurs résultats que les représentations profondes de VGG-M sur une nouvelle base de données. De plus, cette base contient dix fois plus d'images que FGVC-Aircraft. Ainsi, on montre qu'en plus de pouvoir être appris sur une base de moyenne envergure telle que FGVC-Aircraft, notre réseau parvient à traiter des bases de grande envergure, ce qui est une tâche difficile avec des représentations trop lourdes telles que FV. Cette étude montre donc la robustesse de notre modèle face au bruit d'annotation ainsi que sa capacité à aborder une base de données de taille beaucoup plus conséquente que FGVC-Aircraft.

### 3.3.4 Bilan

Dans cette partie, on propose LR-CNN, un réseau de neurones inspiré des architectures construites pour la reconnaissance des images hautement résolues. Notre structure est directement adaptée aux images faiblement résolues puisqu'elle prend en entrée des images de taille $32 \times 32$, ce qui n'est pas le cas des méthodes concurrentes testées FV et les représentations profondes extraites de VGG-M. Dans notre modèle, il n'y a donc pas de pré-traitement des images nécessaire. De plus, cette architecture ne comporte que peu de poids à apprendre, contrairement aux réseaux classiques de l'état de l'art, si bien qu'il est aisé de les apprendre sur des bases de moyenne envergure telles que FGVC-Aircraft.

On a montré expérimentalement notre réseau obtient de meilleurs résultats que FV et les représentations profondes pré-entraînées testées dans les contextes de classification à grain-fin de FGVC-Aircraft et UPMC-Food-101. Les bons résultats obtenus sur UPMC-Food-101 nous ont également permis de montrer la capacité de notre LR-CNN à être appliqué à de grandes bases de données. Enfin, on a prouvé la robustesse de notre méthode face au bruit d'annotation sur UPMC-Food-101.

## 3.4 Conclusion

Dans ce chapitre, on s'est intéressé à évaluer différentes représentations sur la tâche de la classification orientée grain-fin au regard de la résolution des images d'entrée, un paramètre essentiel bien que peu exploré dans la littérature. Plus précisément, on s'est intéressés à des représentations extraites d'un réseau de neurones pré-entraîné sur une large base de données externe, ainsi qu'à des représentations Fisher Vector, basés sur les systèmes de type Bag-of-Visual-Words. Cette étude nous a permis de montrer que les représentations profondes pré-apprises atteignent de bonnes performances sur les images fortement résolues, mais qu'en revanche, lorsque la résolution de l'image d'entrée diminue, les performances en transfert brut chutent.

Cela peut être interprété comme le reflet du phénomène de transfert entre la base externe (sur laquelle est appris le réseau de neurones profond) et la base ciblée. En effet, en nous intéressant à l'impact de la profondeur des représentations extraites sur les performances, nous avons mis en lumière le fait que les couches les plus profondes sont plus adaptées à la base de données sur laquelle ont été appris les poids. En revanche, les couches les plus superficielles, bien qu'offrant une représentation plus générale, ne permettent pas d'extraire une description assez complète face à la complexité de la tâche abordée. Avec des représentations profondes extraites d'un réseau pré-entraîné, les meilleures performances peuvent être atteintes par des représentations intermédiaires, assez complexes mais pas encore trop spécialisées. Ces représentations sont néanmoins assez lourdes à manipuler, si bien qu'il peut être rédhibitoire de chercher à représenter les images de test ainsi.

Pour surmonter ces difficultés, nous avons alors proposé LR-CNN, un réseau de neurones inspiré des réseaux état de l'art sur les tâches de classification d'images fortement résolues. Nous avons prouvé l'efficacité de notre architecture sur deux bases orientées grain-fin, ce qui montre l'intérêt de notre réseau. Par ailleurs, les performances reportées sur une base contenant du bruit d'annotation montre la robustesse de notre méthode face à ce problème, et les bons résultats reportés sur une large base de données montrent la capacité de cette structure à traiter un grand nombre de données.

Pour améliorer les performances, on a proposé d'utiliser plusieurs résolutions pendant l'apprentissage. On a alors montré que l'intégration naïve des images fortement résolues dans l'apprentissage d'un classifieur visant ŕeconnaître des images faiblement résolues ne permettait pas d'améliorer les performances, voire les détériorait. Ces images haute résolution sont néanmoins disponibles durant l'apprentissage, et les ressources peuvent permettre de les traiter pendant cette phase. Une utilisation plus réfléchie de cette information complémentaire pourrait permettre d'améliorer les performances de classification en test. Pour ce faire, on s'intéresse dans la suite à des méthodes d'apprentissage avec information privilégiée (ou LUPI), un cadre permettant d'intégrer de l'information complémentaire durant la phase d'apprentissage d'un système. On développe plusieurs modèles permettant d'incorporer les images haute résolution dans l'apprentissage des réseaux de neurones.

# Chapitre 4

# Méthodes d'apprentissage avec information privilégiée pour la classification d'images

**Sommaire**



Dans le cadre de notre étude, on cherche à reconnaître des véhicules contenus dans de petites images. Pour ce faire, on dispose de ces petites images durant les phases d'apprentissage et de test. Durant la phase d'apprentissage, on possède également des données visuelles complémentaires : pour chaque petite image d'apprentissage, on dispose aussi de sa version hautement résolue. Dans un contexte opérationnel, l'utilisation d'images simulées permet notamment de se placer dans ce cadre d'étude (*cf.* annexe A pour de plus amples détails sur ces données). On cherche donc à utiliser ces données additionnelles pour améliorer le système de reconnaissance visuelle.

Dans le chapitre 3, nous avons montré l'intérêt des réseaux de neurones profonds pour la classification d'images dans le contexte de la reconnaissance à grain-fin. Dans la partie 3.2, nous avons cependant souligné que l'intégration naïve de ces images complémentaires durant l'apprentissage d'un classifieur ne permet pas d'améliorer les performances en test : il faut donc définir un cadre d'intégration plus poussé pour pouvoir réellement bénéficier de ces informations supplémentaires. Dans cette optique, dans la suite de ce manuscrit, on s'intéresse au cadre de l'apprentissage avec information privilégiée (ou LUPI pour *Learning*

*Using Privileged Information*), un domaine de l'apprentissage automatique permettant d'intégrer de l'information complémentaire durant la phase d'apprentissage.

À cette fin, dans ce chapitre, on introduit le cadre de l'apprentissage avec information privilégiée, en l'inscrivant à la convergence entre les techniques de fusion d'informations hétérogènes et de pondération des exemples. On présente ensuite plusieurs méthodes de l'état de l'art issues de cette approche, exploitant l'information privilégiée comme une pondération des exemples d'apprentissage ou comme une contrainte sur la forme de la sortie du classifieur.

## 4.1 L'information privilégiée à la convergence de plusieurs domaines

Dans de nombreuses tâches de vision par ordinateur et d'apprentissage automatique, il est fréquent d'avoir accès à des données issues de différentes sources d'information. Ainsi, un même élément peut être décrit par plusieurs modalités. Ces modalités peuvent être visuelles, comme par exemple le cas où on a accès à un masque de segmentation ou à une boîte englobante sur les images. Dans d'autres cas, les modalités peuvent aussi être de natures différentes. On peut citer par exemple l'information des points de fixation du regard sur les images (Wang et al., 2015b), les données de navigation associées aux images captées depuis les systèmes aéroportés, le texte de sous-titrage associé à une image, ou encore le code html de la page web contenant l'image. Plusieurs de ces exemples sont illustrés par la figure 4.1.

### 4.1.1 Fusion d'informations hétérogènes

Lorsque ces différentes modalités sont disponibles sur toutes les images d'apprentissage et de test, les cadres du traitement multi-modal ou multi-vues sont particulièrement adaptés pour aborder le problème. Dans ce domaine, le but est de trouver une manière de fusionner au mieux les informations disponibles. Il peut s'agir d'une fusion précoce lorsqu'on cherche à combiner les représentations, ou tardive lorsqu'on combine les décisions de plusieurs classifieurs (Snoek et al., 2005). Dans un contexte plus proche de l'apprentissage automatique, des travaux plus récents tels que SimpleMKL (Rakotomamonjy et al., 2008) et LP-$\beta$ (Gehler and Nowozin, 2009) ont également proposé un cadre de fusion de noyaux (Picard et al., 2012), appliqués notamment à l'image (Picard et al., 2010). Ce principe peut être vu comme une sorte d'intermédiaire entre les fusions précoce et tardive. Ces méthodes sont néanmoins utilisées dans le cas où toutes les modalités sont disponibles pour les images d'apprentissage mais aussi celles de test.

Enfin, l'apprentissage de représentations vise à apprendre une projection vers un espace (ou *embedding*) dans lequel les représentations des différentes modalités sont alignées. C'est le rôle notamment de l'analyse canonique des corrélations (ou CCA pour *Canonical Correlation Analysis*) (Hotelling, 1936; Hardoon et al., 2004) avec les extensions aux méthodes à noyaux comme Kernel CCA (Akaho, 2001), puis aux réseaux de neurones comme Deep CCA (Andrew et al., 2013) et DeepCorrNet (Chandar et al., 2016). Les représentations ainsi apprises peuvent notamment servir à d'autres tâches comme l'indexation multi-modale (Ranjan et al., 2015).

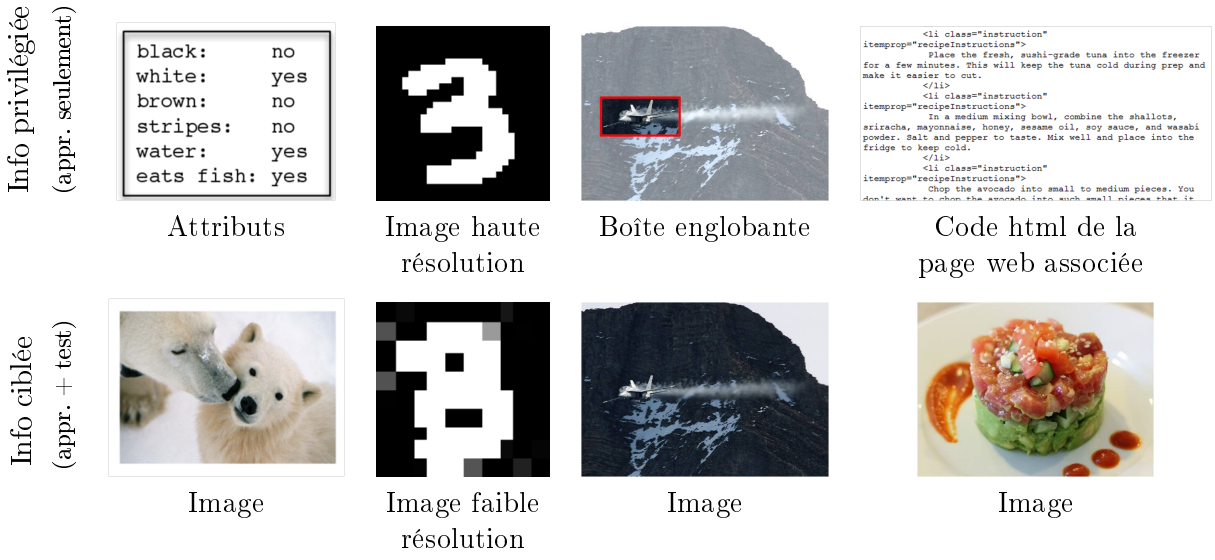



FIGURE 4.1 – Exemples de différents types d'information privilégiée.

### 4.1.2 Techniques de pondération des exemples

Dans leurs travaux, Wu and Srihari (2004) proposent une méthode d'incorporation de connaissances *a priori* sur les données d'apprentissage dans l'entraînement d'un SVM. Ces connaissances sont introduites sous la forme de coefficients qui pondèrent le coût induit par chaque exemple. Dans le cas du SVM à marge souple (eq. 2.3), chaque exemple $x_i$ n'a plus un poids $C\xi_i$ mais $c_i\xi_i$, où $c_i$ est un coefficient propre à l'exemple $x_i$. Dans leurs travaux, le modèle est appliqué à un contexte de classification de textes, où les poids $c_i$ sont déterminés comme la proportion de mot-clés représentant chacune des classes de documents cherchées. Ce modèle permet d'obtenir de bons résultats avec moins d'exemples annotés que pour le SVM standard.

Dans le même ordre d'idée, plusieurs techniques de sélection des exemples ont été appliquées avec succès pour l'apprentissage automatique. Notamment, le boosting (Schapire, 1990; Viola and Jones, 2001; Opelt et al., 2004) repose sur l'idée que les exemples les plus difficiles sont ceux qui permettront le mieux de faire progresser le classifieur. Selon cette intuition, cette démarche consiste à focaliser l'apprentissage sur les exemples les plus difficiles à chaque étape, *i.e.* ceux qui n'étaient pas correctement reconnus à l'étape précédente.

Basé sur la même inspiration, le modèle dit *Online Hard Example Mining* (Shrivastava et al., 2016; Dai et al., 2016) peut être vu comme une technique de boosting dans le cadre de la détection dans les images : pour chaque image, le réseau de neurones appris se focalise uniquement sur les régions qui ont mal été reconnues á l'étape précédente.

À l'inverse, le *curriculum learning* (Bengio et al., 2009; Pentina et al., 2015) s'appuie sur l'intuition qu'il est plus facile pour un élève de commencer par assimiler les concepts les plus faciles avant d'essayer d'intégrer les concepts les plus difficiles. Dans ce sens, le curriculum learning restreint d'abord l'ensemble d'apprentissage aux exemples les plus simples, puis ajoute à chaque itération des exemples un peu plus complexes.

### 4.1.3 Asymétrie entre les annotations et la décision voulue

L'apprentissage faiblement supervisé est un cadre d'apprentissage où l'information disponible est moins précise que la décision qu'on cherche à produire. Par exemple, Durand et al. (2015, 2016) cherchent à localiser les objets dans les images en ne connaissant que le label global de l'image.

Dans le cas de l'apprentissage semi-supervisé, on ne dispose durant la phase d'entraînement que d'un petit ensemble de données annotées, complété par une grande quantité de données sans label (Delalleau et al., 2005). Une méthode courante d'aborder ce problème consiste à annoter les données sans label grâce à un classifieur appris auparavant sur les données annotées (Guillaumin et al., 2010; Rasmus et al., 2015). Ces cadres d'étude sont néanmoins assez éloignés de nos travaux, car on s'attache ici à des cas d'apprentissage supervisé où tous les labels sont disponibles durant la phase d'apprentissage.

Dans un contexte similaire, l'approche du co-training, introduit par Blum and Mitchell (1998), permet de traiter des problèmes où on dispose d'un grand nombre de données d'apprentissage, dont un petit ensemble $L$ sont annotées, et les autres sont sans label. Chaque exemple est décrit par deux modalités, et chacune des deux modalités est suffisante pour entraîner un classifieur. Dans ce contexte, deux modèles sont appris indépendamment sur chacune des modalités des exemples annotés $L$. Chaque classifieur prédit ensuite un label sur un ensemble de données non annotées choisies aléatoirement $U$. On sélectionne alors un nombre fixé d'images sur lesquelles les classifieurs sont les plus confiants. Ces images sont ajoutées à l'ensemble d'exemples annotés $L$ pour ré-apprendre chacun des deux modèles. Cette approche, appliquée principalement à la classification de pages web, est à la croisée des approches de fusion multi-modale, de sélection des exemples et d'apprentissage semi-supervisé. Ces approches s'adressent néanmoins à des contextes dans lesquels un large ensemble de données sans label sont disponibles, ce qui n'est pas notre cadre d'études. Le co-training s'appuie aussi sur le fait que chaque modalité est suffisante pour classifier les exemples, ce qui peut ne pas être vérifié, comme dans le cas où on dispose d'informations de navigation du porteur associées aux images acquises. De plus, cette approche s'appuie sur l'hypothèse que les distributions des modalités sont conditionnellement indépendantes. Cette contrainte n'est pas nécessairement vérifiée dans le contexte qui nous intéresse, notamment lors de l'utilisation d'images haute-résolution pour améliorer la reconnaissance d'images faiblement résolues.

### 4.1.4 L'apprentissage avec information privilégiée

Comme décrite par Vapnik and Vashist (2009), l'information privilégiée est un ensemble de données qui n'est disponible que durant la phase d'apprentissage. Ces données additionnelles viennent compléter l'information dite "information ciblée", qui est disponible durant les phases d'apprentissage et de test. Le but du système appris par le cadre de l'apprentissage avec information privilégiée (ou *LUPI* pour *Learning Using Privileged Information*) a pour but de traiter l'information ciblée en test, et utilise pour cela les données ciblées d'apprentissage assorties de leurs représentations privilégiées.

Cette information privilégiée, plus riche que l'information ciblée, peut être de natures très diverses : attributs (Sharmanska et al., 2013, 2014; Feyereisl et al., 2014; Wang and Ji, 2015; Hernández-Lobato et al., 2014), description textuelle des images (Vapnik and Vashist, 2009; Li et al., 2016; Sharmanska et al., 2014; Hernández-Lobato et al., 2014),

information de profondeur (Chen et al., 2014; Hoffman et al., 2016), masques de segmentation (Feyereisl et al., 2014), structure géométrique d'un élément dans l'espace (Vapnik and Vashist, 2009), représentation spectrale des objets lointains (Fouad et al., 2013; Fouad and Tino, 2013), versions haute résolution des images d'apprentissage (Lapin et al., 2014; Vapnik and Vashist, 2009; Vapnik and Izmailov, 2015; Li et al., 2016; Lopez-Paz et al., 2016). On présente sur la figure 4.1 plusieurs exemples de données ciblées et de leurs représentations privilégiées, utilisées dans l'état de l'art LUPI.

Historiquement, Vapnik and Vashist (2009) interprètent ces données supplémentaires comme un moyen d'améliorer la vitesse de convergence du SVM appris pour reconnaître les données ciblées. En effet, les auteurs partent du constat que, pour $N$ exemples d'apprentissage plongés dans un même espace de représentations, le SVM binaire à marge souple a un taux de convergence de $O(\sqrt{h/N})$ (où $h$ est la capacité de l'ensemble des fonctions admissibles), alors que celui du SVM binaire à marge dure est de $O(h/N)$. Autrement dit, le SVM à marge dure converge plus vite (avec moins d'exemples). Le modèle SVM+ repose sur l'intuition que, si les variables ressorts $\xi_i$ intervenant dans l'optimisation du SVM à marge souple (2.3) étaient connues, on se ramènerait alors à un problème à marge dure avec une marge propre à chaque exemple. En s'appuyant sur cette intuition, Vapnik and Vashist (2009) proposent donc d'utiliser l'information privilégiée comme un oracle fournissant les valeurs des variables ressorts, *i.e.* prédisant la difficulté des exemples, ce qui permet au système de converger plus rapidement.

De manière générale, le cadre LUPI peut être vu comme un cadre exploitant une information complémentaire pour déterminer la difficulté des exemples d'apprentissage. L'information privilégiée peut ainsi être interprétée comme une modalité complémentaire aux images, au même sens que les méthodes évoquées dans la section 4.1.1, à ceci près qu'elle n'est disponible que pour les exemples d'entraînement. Le but des algorithmes LUPI consiste donc à définir la meilleure relation possible entre ces deux espaces pour bénéficier au mieux des données additionnelles. La connaissance de la difficulté des exemples peut ainsi être interprétée comme une pondération lors de l'apprentissage, dans le même esprit que les méthodes présentées dans la section 4.1.2, cependant il s'agit ici d'une pondération issue d'un espace complémentaire (*i.e.* l'espace privilégié). Par ailleurs, d'autres approches de l'état de l'art LUPI s'appuient sur une condition de ressemblance entre les modèles appris sur chacun des deux types d'information. Ces méthodes font d'ailleurs écho aux modèles d'apprentissage d'*embedding* cités dans la section 4.1.1. Ainsi, à plusieurs points de vue, les méthodes LUPI sont bien à la jonction des approches de fusion multi-modales et de pondération des exemples citées plus haut. Dans ce chapitre, on présente le cadre de l'apprentissage avec information privilégiée, ainsi que différentes méthodes majeures de l'état de l'art dans ce domaine.

## 4.2 Présentation des méthodes de l'état de l'art LUPI

L'intérêt de l'information privilégié est d'orienter l'apprentissage du classifieur dans l'espace ciblé, en lui apportant une information complémentaire issue de ces données. On peut notamment traduire le modèle SVM+ comme un cadre où l'information privilégiée permet de connaître la difficulté des exemples d'apprentissage : l'information privilégiée permet d'indiquer au classifieur dans l'espace ciblé les exemples qui sont les plus dif-

ficiles et leur niveau de difficulté. De manière plus générale, les approches de l'état de l'art LUPI peuvent être vues comme différentes méthodes pour utiliser l'information privilégiée comme un indicateur de la difficulté de reconnaître les exemples, et la manière d'intégrer cette indication dans l'espace ciblé est également traitée différemment suivant les approches.

Plusieurs méthodes ont proposé leurs approches sur cette question. Dans le SVM+ de Vapnik and Vashist (2009), la difficulté est estimée sans que l'information privilégiée ne soit classée, et le classifieur dans l'espace ciblé se focalise plutôt sur les exemples proches de la marge (*i.e.* les exemples difficiles). En revanche, dans le Margin Transfer de Sharmanska et al. (2014), appliqué aussi à l'apprentissage d'ordonnancement (Sharmanska et al., 2013), l'information privilégiée doit être correctement classifiée, et l'accent est mis sur les exemples les plus faciles dans l'espace ciblé.

Une autre manière d'aborder le cadre LUPI consiste à introduire un coût relatif entre l'espace ciblé et l'espace privilégié. Cette approche est notamment celle adoptée par des techniques basées sur la compression de Bucila et al. (2006) et la distillation de Hinton et al. (2014), qui forcent la sortie du CNN dans l'espace ciblé à ressembler à celle du CNN appris dans l'espace privilégié. Le modèle d'*hallucination* de Hoffman et al. (2016) *fine-tune* les poids d'une CNN appris sur les données privilégiées en forçant une couche prédéfinie à prédire la même sortie que la couche correspondante d'un CNN appris sur les données ciblées. Durant la phase de test, ce CNN parvient alors à "halluciner" l'information privilégiée indisponible sur ces images. Le modèle Generalized Distillation de Lopez-Paz et al. (2016) est par ailleurs une méthode particulièrement intéressante, puisqu'elle allie le cadre LUPI avec la distillation de Hinton et al. (2014) dans un cadre d'application très général, c'est pourquoi on s'intéresse particulièrement à cette méthode par la suite. Le modèle très similaire proposé par Gupta et al. (2016) résulte de travaux simultanés, et montre l'intérêt d'une telle approche sur des tâches de reconnaissance visuelles comme la détection d'actions dans les vidéos. Le modèle LIR (*Loss Inequality Regularization*) de Wang and Ji (2015) considère le score dans l'espace privilégié comme un score maximal à ne pas excéder dans l'espace ciblé, sous peine de considérer qu'on sur-apprend sur cet exemple. La similarité SVM$\ominus$ de Wolf and Levy (2013), qui force l'information de similarité entre deux visages (information ciblée) à être décorrélée de l'information de pose 3D de ces visages (information privilégiée).

Dans la suite, on présente deux méthodes principales de l'état de l'art pour l'apprentissage avec information privilégiée basées sur la pondération des exemples, citées plus haut : SVM+ (Vapnik and Vashist, 2009) et Margin Transfer (Sharmanska et al., 2014). Après une discussion sur le positionnement de ces deux méthodes, on présente ensuite Generalized Distillation (Lopez-Paz et al., 2016), une méthode générale imposant un coût imposant une ressemblance entre les modèles appris dans chacun des espaces.

### 4.2.1 SVM+ : approche historique

L'utilisation d'information privilégiée est historiquement introduite par Vapnik and Vashist (2009). Dans leur approche SVM+, l'information privilégiée est considérée comme un approximateur des variables ressorts du SVM binaire (ou *slack variables*). Autrement dit, on ne cherche plus à résoudre le problème d'optimisation de SVM à marge souple

binaire classique (*cf.* eq. 2.3), mais le problème suivant :

$$\min_{w,w^*,b,b^*} \frac{1}{2}\big(\|w\|^2 + \gamma\|w^*\|^2\big) + C\sum_{i=1}^{N}\big(\langle w^*, x_i^*\rangle + b^*\big) \tag{4.1}$$
$$\text{s.c. } \forall i = 1..N,\ y_i(\langle w, x_i\rangle + b) \geq 1 - \big(\langle w^*, x_i^*\rangle + b^*\big),\ \langle w^*, x_i^*\rangle + b^* \geq 0$$

où $w^*$ est un classifieur appris dans l'espace privilégié, $y_i$ est le label (binaire) de l'image $i$, et $C$ et $\gamma$ sont des hyper-paramètres. Les paramètres $w, w^*, b, b^*$ sont optimisés de manière jointe entre les deux espaces. Plusieurs articles se sont intéressés à l'optimisation de ce système (Pechyony and Vapnik, 2010, 2011). Une formulation quadratique a par ailleurs récemment été proposée (Li et al., 2016).

Dans cette formulation, les variables $\xi_i$ traduisant la capacité du SVM à violer la contrainte de la marge s'expriment comme un score dans l'espace privilégié $\langle w^*, x_i^*\rangle + b^*$. Ici, l'espace privilégié est donc une forme d'oracle, qui définit la valeur maximale de violation de la marge de 1 imposée par le classifieur. Tant que cette marge $1-\big(\langle w^*, x_i^*\rangle+b^*\big)$ est respectée, la condition de (4.1) est satisfaite, donc $w$ n'a pas besoin de se déformer. En revanche, dès que l'exemple $i$ est classé avec une marge inférieure à cette valeur, il intervient dans la déformation de $w$.

Il convient de noter que cette formulation ne s'applique qu'aux cas de classification binaires. Face à un problème multi-classe, il faut donc nécessairement se ramener artificiellement à un ensemble de problèmes binaires, en traduisant le problème en *un contre tous* ou *un contre un*.

Par ailleurs, à l'instar de l'optimisation SVM à marge souple standard (2.3), cette formulation ne tient pas compte des exemples très bien classés tant qu'ils le sont avec une marge supérieure à 1. SVM+ ne se focalise que sur les exemples les plus proches de la frontière. D'ailleurs, les travaux de Lapin et al. (2014) mettent en évidence le fait que toute solution d'un système SVM+ est une solution à un SVM pondéré. De surcroît, les auteurs montrent que dans ce SVM pondéré, les poids les plus importants sont nécessairement affectés aux exemples pour lesquels les variables ressorts sont les plus élevées, *i.e.* les exemples les plus difficiles.

### 4.2.2 Margin Transfer : mettre l'accent sur les exemples faciles

Introduit par Sharmanska et al. (2013, 2014), Margin Transfer est un algorithme basé sur l'intuition que la représentation privilégiée est toujours plus informative que la représentation ciblée d'un exemple. Dans ce cadre, les exemples les plus faciles (*i.e.* les plus représentatifs de leur classe) dans l'espace ciblé seront a fortiori les exemples les plus faciles dans l'espace privilégié. De même, les exemples aberrants (ou *outliers*) dans l'espace privilégié seront nécessairement des aberrations dans l'espace ciblé.

Margin Transfer se propose donc de caractériser les exemples dans l'espace ciblé par leur difficulté à être reconnus dans l'espace privilégié. Les exemples les plus représentatifs seront les plus faciles à classer dans l'espace privilégié et devront donc impérativement être bien classés dans l'espace ciblé. En revanche, plus un exemple est difficile à reconnaître dans l'espace privilégié, plus il a de risques d'être une aberration, et doit donc être ignoré par le classifieur dans l'espace ciblé.

Concrètement, cela se traduit par une adaptation de la marge imposée à chaque exemple en fonction de sa difficulté dans l'espace privilégié. Un exemple difficile ne doit

pas avoir une influence trop forte, c'est pourquoi sa marge est relaxée. En revanche, les exemples faciles doivent impérativement être bien classés : leurs marges seront donc plus importantes.

Pour traduire ce principe, les auteurs proposent alors l'algorithme suivant :

1. Apprentissage d'un SVM binaire dans l'espace privilégié :

$$\min_{w^*,\xi_i^*} \frac{1}{2}\|w^*\|^2 + C^* \sum_{i=1}^{N} \xi_i^* \qquad (4.2)$$
$$\text{s.c. } \forall i = 1..N,\ y_i\langle w^*, x_i^*\rangle \geq 1 - \xi_i^*,\ \xi_i^* \geq 0$$

2. Calcul des coefficients de difficulté de chaque exemple

$$\rho_i = y_i\big(\langle w^*, x_i^*\rangle + b^*\big) \qquad (4.3)$$

   Seuillage et plafonnement de ces valeurs à des valeurs $\rho_{min} > 0$ et $\rho_{max}$ (qui sont des hyper-paramètres)

3. Apprentissage, dans l'espace ciblé, d'un classifieur pour lequel les marges sont adaptées à la difficulté de chaque exemple :

$$\min_{w,\xi_i} \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{N} \xi_i \qquad (4.4)$$
$$\text{s.c. } \forall i = 1..N,\ y_i\langle w, x_i\rangle \geq \rho_i - \xi_i,\ \xi_i \geq 0$$

Dans ce cadre, la définition des $\rho_i$ (4.3) reflète bien la difficulté de chaque exemple $i$ dans l'espace privilégié : pour un exemple positif ($y_i = 1$), plus le score $\langle w^*, x_i^*\rangle$ est grand, plus l'exemple a été facile à classer. En revanche, un exemple positif ayant eu un score faible voire négatif est considéré comme très difficile (et potentiellement aberrant), et son score $\rho_i$ est faible. Cette notion de difficulté est ensuite transférée vers l'espace ciblé grâce à l'équation (4.4) : un exemple difficile sera forcé à une marge $\rho_i$ faible, alors qu'un exemple facile devra rester bien classé avec une forte marge.

Il convient par ailleurs de remarquer que le système d'équations (4.4) peut se récrire comme un SVM pondéré, dès lors que les $\rho_i$ sont tous strictement positifs :

$$\min_{w,\xi_i} \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{N} \rho_i\hat{\xi}_i \qquad (4.5)$$
$$\text{s.c. } \forall i = 1..N,\ y_i\langle w, \hat{x}_i\rangle \geq 1 - \hat{\xi}_i,\ \hat{\xi}_i \geq 0$$

où $\forall i = 1..N,\ \hat{x}_i = \frac{x_i}{\rho_i}$ et $\hat{\xi}_i = \frac{\xi_i}{\rho_i}$. Ainsi, chaque exemple intervient donc dans l'optimisation du vecteur $w$ avec un poids $\rho_i$, qui traduit sa facilité à être reconnu. Il convient de noter que cette formulation fait d'autant plus ressortir le fait que les exemples les plus difficiles peuvent avoir un poids très faible dans l'apprentissage. En effet, plus $\rho_{\min}$ est petit (proche de 0), plus les exemples difficiles sont écartés de l'apprentissage du classifieur. Dans un cas limite, on peut dire que Margin Transfer supprime quasiment les exemples les plus difficiles (donc les potentielles aberrations) de l'ensemble d'apprentissage.

Dans cet algorithme, on met donc l'accent sur les exemples les plus faciles (qui sont alors vus comme les exemples les plus représentatifs de leur classe), alors que SVM+ (*cf.* section 4.2.1) ne s'attache qu'à contraindre les exemples les plus difficiles (qui sont alors vus comme les exemples les plus discriminants). Intrinsèquement, l'approche Margin Transfer est donc intuitivement plus adaptée aux ensembles contenant potentiellement des aberrations.

Il convient de noter que cette approche n'est proposée que pour le cas binaire. Comme pour SVM+, il faut donc artificiellement se ramener à un ensemble de problèmes binaires pour traiter un problème multi-classe. Par ailleurs, comme pour SVM+, la formulation est développée pour un système SVM et non pas pour d'autres types de classifieurs comme les réseaux de neurones.

### 4.2.3 Discussion sur les modèles Margin Transfer et SVM+

Ici, on a décrit SVM+ et Margin Transfer, deux méthodes utilisant l'information privilégiée comme un indicateur de la difficulté des exemples. Ces deux méthodes diffèrent néanmoins sur plusieurs aspects. D'abord, Margin Transfer s'appuie plus fortement sur les exemples faciles pour apprendre l'hyperplan de séparation dans l'espace ciblé, alors que SVM+, au même titre que SVM, se focalise sur les exemples qui violent la marge autour de l'hyperplan, *i.e.* les exemples les plus difficiles. D'autre part, dans Margin Transfer, l'information privilégiée est classée dans l'espace privilégié, puis la difficulté des exemples est calculée dans cet espace. On s'appuie ensuite sur l'hypothèse de transférabilité de cette information vers l'espace ciblé. Le modèle SVM+ au contraire ne formule aucune contrainte sur le fait de bien classer les représentations dans l'espace privilégié. La difficulté des exemples est donc estimée dans cet espace pour correspondre directement à la variable ressort dans l'espace ciblé.

Bien que basées sur des approches complémentaires à plusieurs points de vue, ces deux méthodes proposent néanmoins un cadre reposant *explicitement* sur le calcul d'un niveau de difficulté pour améliorer un modèle SVM. Une autre approche également adoptée par de nombreuses méthodes LUPI consiste à formuler un coût relatif entre les modèles appris dans chacun des deux espaces (ciblé ou privilégié), comme c'est le cas par exemple de la méthode Generalized Distillation de Lopez-Paz et al. (2016), appliquée aux réseaux de neurones et qu'on présente ci-après.

### 4.2.4 Generalized Distillation : l'information privilégiée dans les réseaux de neurones

Les méthodes d'apprentissage avec information privilégiée ont surtout exploré la possibilité d'améliorer l'apprentissage de modèles superficiels comme les SVM (Vapnik and Vashist, 2009) ou les méthodes de regroupement (Feyereisl and Aickelin, 2012), et ce n'est que très récemment que le cadre LUPI a été proposé dans les réseaux de neurones. Les méthodes d'intégration de l'information privilégiée dans les réseaux de neurones s'appuient principalement sur la vision de Distillation proposée par Hinton et al. (2014). Cette approche est avant tout basée sur les travaux de compression de structures profondes de Bucila et al. (2006) : en forçant les sorties de la dernière couche d'un petit réseau de neurones à ressembler à celle d'un réseau de neurones conséquent appris sur un grand nombre de données, le petit réseau parvient à reproduire les performances du réseau très

profond. Cette idée, plus tard reprise par les FitNets (Romero et al., 2015) et les travaux de Ba and Caruana (2014), permet d'augmenter le pouvoir de représentation d'un petit réseau tout en ne l'entraînant que sur peu de données.

Dans leurs travaux, Lopez-Paz et al. (2016) reprennent l'idée de la Distillation de Hinton et al. (2014) et font le lien avec le cadre de l'information privilégiée. Ils présentent l'algorithme Generalized Distillation, un cadre d'apprentissage très général pour l'intégration de l'information privilégiée dans l'apprentissage des réseaux de neurones. Dans leur approche, un réseau de neurones est appris dans l'espace privilégié, qui est considéré comme étant plus informatif que l'espace ciblé. Ensuite, un second réseau est appris sur les données ciblées, avec un but double : il doit parvenir à bien reconnaître le label des données ciblées, mais il doit aussi réussir à prédire le même vecteur de scores que le réseau privilégié. Autrement dit, la fonction de coût doit contraindre les scores des classes présentes à être assez élevés, mais doit également contraindre la sortie du softmax à être similaire à celle du réseau dans l'espace privilégié. Concrètement, le coût s'exprime alors de la façon suivante :

$$loss_{GD} = (1-\lambda) loss(z_i, y_i) + \lambda loss(z_i, \sigma_{\mathbf{i}}^*) \tag{4.6}$$

où à chaque image $i$ est associé un vecteur de labels $y_i$, $\sigma_{\mathbf{i}}^*$ est la matrice des sorties du softmax dans l'espace privilégié

$$\sigma_k^* = \frac{e^{z_k^*/T}}{\sum_{k'=1}^{N_{classes}} e^{z_{k'}^*/T}} \tag{4.7}$$

$\lambda$ est un paramètre réglant un compromis entre le coût dans l'espace ciblé $loss(z_i, y_i)$ et le coût dans l'espace privilégié $loss(z_i, \sigma_{\mathbf{i}}^*)$, et $T$ est un hyper-paramètre de température permettant de régler l'étalement des scores dans l'espace privilégié.

Il convient de noter que $y_i$ contient un 1 pour chaque classe présente dans l'image, et des 0 partout ailleurs. Dans le cadre de nos travaux, on ne considère que des cas mono-labels : le vecteur $y_i$ contient donc un 1 pour la classe vérité-terrain $c$ de l'image $i$, et des 0 partout ailleurs. Dans ce contexte, le coût peut s'exprimer ainsi (on détaille les calculs en annexe B) :

$$loss_{GD} = -(1-\lambda) z_c + \ln\left(\sum_{k'=1}^{N_{classes}} e^{z_{k'}}\right) - \lambda \sum_{k=1}^{N_{classes}} \sigma_k^* z_k \tag{4.8}$$

Dans les deux équations (4.6) et (4.8), on remarque bien que le cas dégénéré $\lambda = 0$ nous ramène à la fonction de coût de classification classique $loss(z_i, y_i)$. De plus, ces deux équations mettent bien en lumière le fait que cette fonction de coût est un compromis entre le terme de classification classique $loss(z_i, y_i)$ et un terme mesurant l'adéquation entre les sorties des deux réseaux. Cette fonction est donc bien une extension de la fonction de coût standard intégrant l'information privilégiée sous la forme d'une contrainte de ressemblance entre les représentations apprises par chacun des réseaux.

## 4.3 Bilan

Dans ce chapitre, on a présenté le cadre d'apprentissage avec information privilégiée, ou LUPI. Dans ce contexte, on a détaillé plusieurs méthodes, basées sur différentes approches

de l'information privilégiée. On a notamment exposé plusieurs méthodes utilisant l'information privilégiée pour calculer des pondérations sur les exemples d'apprentissage dans l'espace ciblé. Ces méthodes ont notamment été appliquées par Vapnik and Vashist (2009) sur des problèmes de reconnaissance d'images faiblement résolues, avec pour information privilégiée les versions haute résolution des images d'apprentissage, c'est-à-dire exactement dans le même cadre que le contexte des travaux présentés dans ce manuscrit. Dans ce contexte, l'hypothèse de Sharmanska et al. (2014) selon laquelle la difficulté de reconnaître un exemple dans l'espace privilégié peut directement être transférée à l'exemple ciblé est parfaitement cohérente. En s'inspirant de cette approche, on propose dans le chapitre 5 deep+, un modèle permettant d'intégrer l'information privilégiée dans les réseaux de neurones profonds.

On a également présenté une deuxième approche, s'appuyant plutôt sur une contrainte de ressemblance entre les sorties des CNN dans chacun des espaces (privilégié et ciblé). Dans le chapitre 6, on propose le modèle deep++, basé sur cette intuition. Ce modèle vise à combiner les approches basées sur la pondération des exemples et celles plutôt basées sur un coût relatif entre les deux espaces, en proposant une pénalisation supplémentaire pour exemples trop mal classés dans l'espace ciblé par rapport à l'espace privilégié.

# Chapitre 5

# Deep+

**Sommaire**



Dans le chapitre 3, nous avons montré l'intérêt des réseaux de neurones profonds pour la classification d'images dans un contexte grain-fin. Dans nos travaux, des images fortement résolues, complémentaires aux images de faible résolution, sont disponibles durant la phase d'apprentissage. Pour bénéficier de ces données, on s'appuie sur le cadre de l'apprentissage avec information privilégiée, développé dans le chapitre 4.

On propose dans ce chapitre la méthode deep+, un modèle permettant d'intégrer l'information privilégiée dans l'apprentissage des réseaux de neurones. Cette méthode s'appuie sur l'information privilégiée pour calculer un niveau de difficulté des exemples, qui est ensuite utilisé pour pondérer l'influence des exemples dans l'apprentissage du réseau de neurones dans l'espace ciblé.

On commence donc par expliciter l'intuition justifiant l'approche adoptée, puis on détaille le modèle en lui-même. Enfin, on s'attache à montrer expérimentalement l'intérêt de notre approche, en montrant l'apport de notre modèle par rapport aux méthodes de l'état de l'art utilisant une approche similaire, mais basées sur des structures peu profondes, ainsi que vis-à-vis d'une méthode sans information privilégié. On propose par ailleurs une évaluation détaillée des divers aspects de notre modèle.

## 5.1 Architecture deep+

L'architecture de notre modèle est résumée dans le schéma de la figure 5.1. Cette approche se déroule en deux étapes : d'abord, un premier modèle de reconnaissance multi-classe est appris dans l'espace privilégié (cadre vert de la figure). Ce système permet de calculer, pour chaque exemple, un coefficient $\rho_i$ reflétant la difficulté qu'il y a à reconnaître cet exemple, à l'instar de la méthode Margin Transfer de Sharmanska et al. (2014). Plus précisément, $\rho_i$ est élevé pour les exemples faciles, et faible pour les exemples difficiles. On propose deux méthodes différentes pour calculer ces coefficients.

Ensuite, un deuxième système de reconnaissance multi-classe est appris, cette fois dans l'espace ciblé (cadre gris), en bénéficiant de l'information privilégiée. Plus précisément, les coefficients $\rho_i$ calculés dans l'espace privilégié permettent de biaiser l'apprentissage de ce modèle. Pour ce faire, le système appris dans l'espace ciblé optimise un nouveau critère donné par $loss_{deep+}$, une fonction de coût intégrant les coefficients de difficulté $\rho_i$. Dans cette fonction de coût, l'erreur commise sur chaque exemple est pondérée par le coefficient $\rho_i$. Ainsi, les exemples les plus faciles (resp. les plus difficiles) ont un poids plus important (resp. plus faibles) dans l'apprentissage du modèle ciblé. Autrement dit, les erreurs commises sur les exemples les plus faciles sont amplifiées, alors qu'une erreur commise sur un exemple difficile est réduite. En somme, cette opération revient à modifier le pas de gradient avec lequel chacun des exemples met à jour le modèle appris dans l'espace ciblé.

Notre modèle s'appuie donc sur le transfert de difficulté entre l'espace privilégié (contenant les versions haute-résolution des images d'apprentissage) et l'espace ciblé (l'espace des images faiblement résolues), qui semble pertinent dans notre cadre d'étude. Dans le cadre de nos expériences, les données ciblées sont des images faiblement résolues, et l'information privilégiée associée est la version haute-résolution de chacune des images d'apprentissage : dans ce contexte, l'hypothèse de travail selon laquelle la difficulté d'un exemple peut être directement transférée de l'espace privilégié à l'espace ciblé est particulièrement bien vérifiée. De plus, par la nature même de l'information privilégiée, il est cohérent de penser que les aberrations de l'ensemble d'apprentissage sont les exemples jugés extrêmement difficiles par un système de reconnaissance d'images dans l'espace privilégié, alors que les exemples les plus faciles dans cet espace devraient être les plus représentatifs de leur classe. Ainsi, on dira qu'un exemple est facile s'il est bien classé avec un score suffisamment élevé dans l'espace privilégié. À l'inverse, les exemples les plus difficiles sont les exemples ayant un score trop faible voire qui sont mal classés par le classifieur appris dans l'espace privilégié.

Notre modèle deep+ fait écho au modèle Margin Transfer de Sharmanska et al. (2014), avec deux contributions principales : d'une part, nous proposons un modèle permettant de prendre en compte l'aspect multi-classe des problèmes de classification dans l'espace privilégié. D'autre part, notre approche permet d'apprendre des réseaux de neurones profonds dans l'espace ciblé. Cela permet non seulement de bénéficier de la chaîne de traitement performante que sont les CNN, mais surtout d'apprendre les représentations internes en intégrant l'information privilégiée.

Dans la suite de ce chapitre, on présente d'abord les différentes étapes de notre modèle deep+, en détaillant d'une part le calcul des coefficients $\rho_i$, et d'autre part la fonction de coût $loss_{deep+}$, qui intègre ces $\rho_i$ dans l'espace ciblé. Ensuite, on présente une évaluation

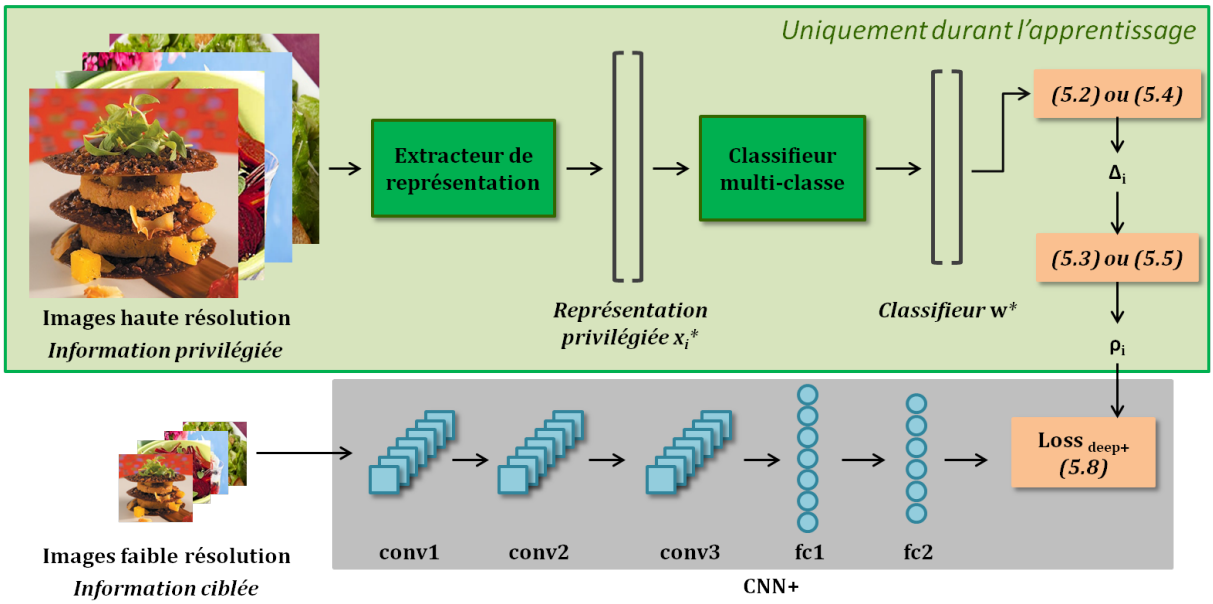



FIGURE 5.1 – Architecture de notre modèle deep+. L'information privilégiée permet de calculer un coefficient $\rho_i$ caractérisant la difficulté de chaque exemple (cadre vert). Ces coefficients sont intégrés dans la fonction de coût $loss_{deep+}$ d'un réseau appris sur les données ciblées (cadre gris). On reporte entre parenthèses les numéros des équations décrivant chaque étape.

expérimentale de notre modèle sur différentes bases de données. D'abord, on montre l'intérêt de notre modèle deep+ par rapport aux méthodes concurrentes de l'état de l'art. Ensuite, on souligne l'apport de notre méthode par rapport à un CNN sans information privilégiée. Enfin, on mène une étude plus approfondie de différents aspects de notre modèle. On finit ce chapitre par une discussion sur notre modèle.

## 5.2 Modules du deep+

### 5.2.1 Expression multi-classe des coefficients $\rho_i$

Comme illustré par la figure 5.1, notre modèle s'appuie sur le calcul de coefficients $\rho_i$ multi-classe dans l'espace privilégié. Ces coefficients sont ensuite intégrés dans la fonction de coût du réseau appris dans l'espace ciblé pour pondérer les exemples dans cet espace. Ces coefficients sont le reflet de la difficulté de chaque exemple : quel que soit le classifieur appris dans l'espace privilégié, un exemple difficile est associé à un $\rho_i$ faible, tandis que les valeurs de $\rho_i$ les plus élevées caractérisent les exemples les mieux reconnus dans l'espace privilégié. Dans cette section, on s'attache à décrire deux manières de calculer les $\rho_i$ dans l'espace privilégié. D'une part, on propose d'utiliser un modèle superficiel SVM multi-classe. On propose ensuite une solution basée sur les réseaux de neurones profonds.

#### SVM dans l'espace privilégié

Dans notre modèle deep+, nous proposons une formulation des coefficients de difficulté $\rho_i$ qui respecte l'aspect multi-classe des problèmes posés. Dans cette perspective, on

apprend un classifieur SVM multi-classe dans l'espace privilégié :

$$\min_{\mathbf{w}^*,\xi^*} \frac{1}{2}\|\mathbf{w}^*\|^2 + C^* \sum_{i=1}^{N} \xi_i^* \tag{5.1}$$
$$\text{s.t. } \forall i = 1..N,\ \langle w_c^*, x_i^* \rangle \geq \langle w_k^*, x_i^* \rangle + 1 - \xi_i^*, \forall k \neq c,\ \xi_i^* \geq 0$$

$C^*$ est un hyper-paramètre, $c$ est la classe vérité terrain de l'exemple $x_i^*$, et $\mathbf{w}^* = [w_k^*]_{k=1..D}$ est la matrice des classifieurs de chaque classe $k$ dans l'espace privilégié. Le score de l'image $x_i^*$ pour la classe $k$ est donc donné par $\langle w_k^*, x_i^* \rangle$. Ainsi, en définissant

$$\Delta_i^{SVM} = \langle w_c^*, x_i^* \rangle - \max_{k \neq c} \langle w_k^*, x_i^* \rangle \tag{5.2}$$

on a une mesure de la proximité entre le score de la véritable classe et celui de la classe la plus ambiguë. Pour les exemples jugés difficiles par l'espace privilégié, le score de la vraie classe $\langle w_c^*, x_i^* \rangle$ sera assez proche du meilleur score parmi les autres $\max_{k \neq c} \langle w_k^*, x_i^* \rangle$ classes ; un exemple difficile aura donc un score faible. En revanche, un exemple très facile dans l'espace privilégié aura un score élevé.

Cette définition prévoit néanmoins que les scores $\Delta_i^{SVM}$ puissent prendre n'importe quelle valeur réelle. En l'occurrence, un exemple très difficile dans l'espace privilégié peut y être mal classé, et le score $\Delta_i^{SVM}$ sera alors négatif. Dans un modèle SVM tel que la formulation Margin Transfer (4.4), cela signifie qu'on autorise cet exemple à être mal classé dans l'espace ciblé. Pour un modèle profond, cela signifie par ailleurs qu'on rétro-propagerait une quantité négative. Pour éviter cette situation, les $\Delta_i^{SVM}$ sont seuillés à une valeur minimale $\tau_{min} > 0$.

De même, la formulation (5.2) prévoit que les $\Delta_i^{SVM}$ puissent prendre des valeurs extrêmement grandes. Dans un réseau de neurones, cela suppose que la quantité rétro-propagée à chaque époque peut être très grande, ce qui pourrait nuire à l'apprentissage du réseau. Les $\Delta_i^{SVM}$ sont donc plafonnés à une valeur $\tau_{max}$. Finalement, on contrôle l'intervalle dans lequel vivent les coefficients $\rho_i^{SVM}$ grâce à l'opération suivante :

$$\rho_i^{SVM} = \max(\min(\Delta_i^{SVM}, \tau_{max}), \tau_{min}) \tag{5.3}$$

**CNN dans l'espace privilégié**

La capacité du modèle SVM linéaire proposé dans le paragraphe précédent est cependant discutable, et les scores calculés peuvent donc se révéler d'un intérêt limité dans le cas où on dispose d'un nombre important d'exemples d'apprentissage. En effet, comme mentionné dans la section 2.2, un classifieur ayant une capacité (*i.e.* une complexité) trop faible par rapport à la tâche traitée ne peut mathématiquement pas permettre de séparer au mieux les exemples d'apprentissage. Dans le cas qui nous intéresse, un classifieur linéaire a une capacité de l'ordre de la dimension des représentations. Si l'ensemble d'apprentissage contient beaucoup plus d'exemples d'apprentissage que la dimension de leurs représentations, le SVM linéaire n'est pas le choix le plus indiqué du fait de sa trop faible capacité.

Par ailleurs, ces coefficients sont seulement seuillés dans l'espace privilégié. Ainsi, un exemple très mal classé dans l'espace privilégié et un exemple à la limite de la bonne

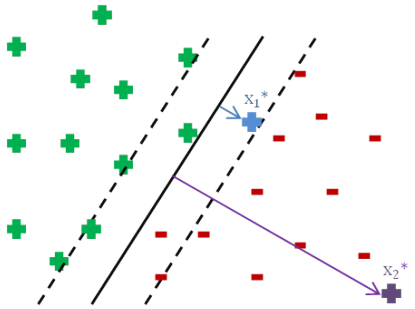



FIGURE 5.2 – Schéma illustrant les limites du seuillage des scores $\Delta_i^{SVM}$ dans l'espace privilégié. Le score $\Delta_i^{SVM}$ de l'exemple $x_2^*$ (violet) est beaucoup plus faible que celui de l'exemple $x_1^*$ (bleu), cependant comme ils sont tous les deux négatifs, après seuillage (5.3), les deux exemples ont le même score $\rho_i^{SVM} = \tau_{min}$.

reconnaissance auront tous les deux le même coefficient du fait du seuillage (eq. 5.3), alors qu'ils n'ont pas intrinsèquement la même difficulté. On représente cette situation sur la figure 5.2. Les exemples représentés en bleu ($x_1^*$) et en violet ($x_2^*$) sont tous les deux mal classés, cependant l'exemple $x_1^*$ est assez proche de la frontière de décision (donc $\Delta_1^{SVM} \simeq 0$), alors que l'exemple $x_2^*$ en est très éloigné (donc $\Delta_2^{SVM} \ll 0$). En vertu de l'équation 5.3, on aura pourtant $\rho_1^{SVM} = \rho_2^{SVM} = \tau_{min}$. On ne fait donc aucune distinction entre ces deux exemples.

On propose ici d'apprendre un réseau de neurones dans l'espace privilégié. Ainsi, dans l'équation (5.2), on pourrait naïvement remplacer les scores $\langle w_k, x_i^* \rangle$ auparavant calculés via le SVM linéaire structuré, par les sorties du softmax du CNN dans l'espace privilégié. On a alors :

$$\Delta_i^{CNN} = \tilde{z}_c(x_i^*) - \max_{k \neq c} \tilde{z}_k(x_i^*) \tag{5.4}$$

où $\tilde{z}_k(x_i^*)$ est la sortie du softmax pour la classe $k$ et pour l'image $x_i^*$ dans l'espace privilégié. Dans l'équation (5.2), les scores $\Delta_i^{SVM}$ pouvaient prendre n'importe quelle valeur réelle. Ici cependant, les scores $\tilde{z}_k(x_i^*)$ étant calculés à la sortie d'un softmax, ils sont nécessairement compris entre 0 et 1. Ainsi, la quantité $\Delta_i^{CNN}$ est toujours comprise entre -1 et 1. De fait, même les exemples les plus faciles ne peuvent avoir un poids supérieur à 1 : on voudrait pouvoir modifier cette valeur maximale. Par ailleurs, les exemples les plus difficiles pourraient avoir un $\Delta_i^{CNN}$ négatif, ce qui n'est pas souhaitable non plus. Il convient donc d'adapter la formulation des poids afin qu'ils soient tous dans un intervalle $[\tau_{min}, \tau_{max}]$ choisi par l'utilisateur, et pour lequel $\tau_{min} > 0$. Pour ce faire, on utilise un remappage linéaire des $\Delta_i^{CNN}$, qui s'exprime comme suit :

$$h_{lin} : \rho_i^{CNN} = \frac{\tau_{max} - \tau_{min}}{2} \Delta_i^{CNN} + \frac{\tau_{max} + \tau_{min}}{2} \tag{5.5}$$

Comme souligné précédemment, la première formulation deep+ (eq. 5.2) donne une même importance aux exemples extrêmement mal classés qu'aux exemples ambigus. Ici, le remappage des quantités $\Delta_i^{CNN}$ permet d'accorder moins de poids aux exemples extrêmement mal classés qu'aux exemples en bordure de la frontière de décision. En effet, un

exemple pour lequel $\rho_i^{CNN} = \tau_{min}$ correspond au cas très défavorable où le réseau accorde un score maximal pour une classe qui n'est pas la bonne. De plus, tous les exemples pour lesquels $\rho_i^{CNN} < \frac{\tau_{max}+\tau_{min}}{2}$ sont des exemples pour lesquels le score de la vraie classe est inférieur au score d'une autre classe. On a ainsi proposé une formulation permettant de hiérarchiser la difficulté des exemples, comme voulu [1].

**Autres fonctions de remappage.** Dans certains cas cependant, les valeurs $\Delta_i^{CNN}$ sont trop peu variées, si bien que la difficulté des exemples n'est que peu exploitée. Par exemple, lorsque le modèle dans l'espace privilégié est bien appris, les $\Delta_i^{CNN}$ sont en très large majorité proches de 1, alors que les exemples les plus difficiles peuvent avoir des valeurs extrêmement faibles, notamment sur une base contenant du bruit d'annotation par exemple. Il convient donc de proposer une formulation permettant de mieux répartir les valeurs des $\rho_i$ entre les deux bornes $\tau_{min}$ et $\tau_{max}$. Ainsi, on propose un remappage exponentiel :

$$h_{exp} : \rho_i^{CNN} = a_{\exp} \exp(\Delta_i^{CNN}) + b_{\exp} \tag{5.6}$$

$$a_{\exp} = \frac{\tau_{max} - \tau_{min}}{\exp(1) - \exp(-1)}$$
$$b_{\exp} = \tau_{max} - \exp(1)\frac{\tau_{max} - \tau_{min}}{\exp(1) - \exp(-1)}$$

Dans certains cas cependant, ce premier remappage peut être encore trop faible. On propose alors un remappage complémentaire :

$$h_{exp2} : \rho_i^{CNN} = a_{\exp 2} \exp(\exp(\Delta_i^{CNN})) + b_{\exp 2} \tag{5.7}$$

$$a_{\exp 2} = \frac{\tau_{max} - \tau_{min}}{\exp(\exp(1)) - \exp(\exp(-1))}$$
$$b_{\exp 2} = \tau_{max} - \exp(\exp(1))\frac{\tau_{max} - \tau_{min}}{\exp(\exp(1)) - \exp(\exp(-1))}$$

Puisque les fonctions sont toutes croissantes par rapport à $\Delta_i^{CNN}$, on conserve bien l'ordre de difficulté des exemples. De plus, les fonctions proposées permettent de s'assurer que les valeurs $\rho_i^{CNN}$ sont toujours comprises dans l'intervalle $[\tau_{min}, \tau_{max}]$. La présence des exponentielles permet cependant de mieux répartir les valeurs des $\rho_i^{CNN}$ entre les deux bornes.

Notre approche apporte ainsi une réponse à plusieurs éléments faisant défaut dans les modèles de l'état de l'art LUPI comme Margin Transfer (Sharmanska et al., 2014). En particulier, elle s'adresse à l'apprentissage de réseaux de neurones dans l'espace ciblé. Ainsi, la solution proposée est un système multi-classe de bout en bout, ce qui permet de traiter directement les tâches de reconnaissance multi-classe, contrairement aux méthodes de l'état de l'art SVM+ (Vapnik and Vashist, 2009) et Margin Transfer. D'autre part, l'apprentissage d'un CNN dans l'espace ciblé permet d'apprendre toutes les représentations profondes internes sur la base de données ciblée. Autrement dit, notre méthode permet de

1. On aurait également pu adopter une stratégie intermédiaire entre les deux modèles proposés (5.2) et (5.5).

faire bénéficier toutes les représentations intermédiaires de l'information privilégiées, alors que les méthodes de l'état de l'art Margin Transfer et SVM+ requièrent de recourir à des représentations construites à la main ou pré-apprises sur des bases de données externes.

### 5.2.2 Fonction de coût

Une fois ces coefficients $\rho_i$ calculés, notre approche consiste à les intégrer dans la fonction de coût du CNN appris dans l'espace ciblé. Ces coefficients permettent de différencier la difficulté des exemples d'après leur représentation privilégiée. Le principe de notre modèle deep+ est d'attribuer un poids plus important aux exemples les plus représentatifs de leur classe (*i.e.* les plus faciles d'après leur représentation privilégiée), et de diminuer l'impact des exemples susceptibles d'être des exemples peu représentatifs, voire des aberrations de l'ensemble de données (*i.e.* les plus difficiles d'après le modèle dans l'espace privilégié). Pour ce faire, nous proposons l'optimisation de la fonction de coût suivante :

$$loss_{deep+} = \sum_{i=1}^{N} -\rho_i \ln\left(\tilde{z}_c(x_i)\right) \tag{5.8}$$

où $\tilde{z}_c(x_i)$ est la valeur de sortie du softmax de l'exemple $x_i$ pour sa vraie classe $c$. Durant l'apprentissage, la quantité rétro-propagée est alors donnée par :

$$\frac{dloss_{deep+}}{d\tilde{z}_k(x_i)} = -\rho_i \frac{1}{\tilde{z}_c(x_i)} \delta_{k=c} \tag{5.9}$$

où $\delta_{k=c} = 1$ si $k = c$, 0 sinon. Dans cette formulation, on confère bien un poids plus important aux exemples les plus faciles, alors que le poids des exemples les plus difficiles est atténué.

Pour résumer, nous avons proposé deep+, un algorithme d'apprentissage de réseau de neurones permettant d'intégrer l'information privilégiée. Ces données complémentaires permettent de déterminer si un exemple est facile ou difficile. L'intégration de cette information dans l'apprentissage du CNN dans l'espace ciblé revient à donner un poids plus élevé aux exemples les plus faciles d'après l'espace privilégié, car ils devraient être les plus représentatifs de leur classe, et à réduire le poids des exemples les plus difficiles, supposément les exemples peu représentatifs voire les aberrations. L'intérêt principal de notre modèle est de pouvoir intégrer l'information privilégiée dans les réseaux de neurones, ce qui permet de faire bénéficier toutes les représentations internes de cette information complémentaire. L'algorithme 1 résume les différentes étapes de notre modèle deep+.

## 5.3 Résultats expérimentaux

On compare d'abord notre approche à deux méthodes de l'état de l'art utilisant l'information privilégiée comme une pondération des exemples dans l'espace ciblé. D'autre part, on met en lumière d'une part l'apport de notre méthode par rapport à un modèle CNN sans information privilégiée. On s'intéresse pour cela à plusieurs contextes de classification, notamment au sein d'une base d'images propre à Thales, afin d'illustrer l'intérêt de notre modèle dans un cadre plus opérationnel.

**Algorithm 1** Apprentissage deep+

**if** *SVM dans l'espace privilégié* **then**
  Extraction de représentations $x_i^*$ sur les images privilégiées
  Apprentissage d'un SVM multi-classe $\mathbf{w}^*$ sur les $x_i^*$ (5.1)
  Calcul des $\rho_i^{SVM}$ (5.2, 5.3)
**else if** *CNN dans l'espace privilégié* **then**
  Apprentissage d'un CNN sur les images privilégiées
  Calcul des $\rho_i^{CNN}$ (5.4, 5.5)
**end if**
Initialisation des poids $\mathbf{w}$ du CNN dans l'espace ciblé
**repeat**
  Calcul de $loss_{deep+}$ (5.8)
  Mise à jour des poids $\mathbf{w}$ par rétro-propagation de $loss_{deep+}$ (5.9)
**until** convergence du CNN dans l'espace ciblé

### 5.3.1 Description des bases de données

En s'inscrivant dans la continuité de la partie 3.3.1, on considère les deux mêmes bases orientées grain-fin que dans la partie 3.3.1 : FGVC-Aircraft (Maji et al., 2013) et UPMC-Food-101 (Wang et al., 2015b), décrites respectivement dans les sections 3.1.1 et 3.3.2. Sur ces deux bases de données orientées grain-fin, on se place dans le même cadre que pour la section 3.3.1 : on cherche à reconnaître de petites images, de taille $32 \times 32$ (données ciblées). Durant la phase d'apprentissage, on dispose également des versions haute-résolution $224 \times 224$ de ces images.

Pour valider de manière plus significative notre modèle, on se place également dans le cadre de la reconnaissance d'images proposé dans les travaux de Vapnik and Vashist (2009) et reprises par de nombreux travaux suivants (Lapin et al., 2014; Li et al., 2016; Lopez-Paz et al., 2016) : dans ce but, on s'intéresse à la base de données MNIST (LeCun et al., 1998). On décrit plus précisément la base de données ainsi que notre cadre d'étude ci-après.

#### MNIST

MNIST (LeCun et al., 1998) est une base d'images de taille $28 \times 28$ contenant des chiffres manuscrits. Chaque image contient un chiffre, centré et occupant quasiment toute l'image. Cette base de données est répartie en 10 classes (une pour chaque chiffre), et contient 60 000 images d'apprentissage pour 10 000 images de test, équiréparties entre les classes.

Sur cette base de données, Vapnik and Vashist (2009) proposent de reconnaître les images réduites à la résolution $10 \times 10$, en prenant comme information privilégiée les images à la taille $28 \times 28$ : on suit ici le même protocole. Pour cette tâche, on utilise un réseau LeNet tel que décrit dans la section 2.3.3, *i.e.* contenant deux couches de convolutions suivies de deux couches totalement connectées. Les images $10 \times 10$ sont donc ré-agrandies à la taille $28 \times 28$ pour correspondre à la taille d'entrée du réseau (on utilise le protocole de retaillage décrit dans la figure 3.3).

Dans leurs expériences, les auteurs se sont intéressés au problème binaire de la discrimination des images contenant le chiffre 5 et de celles contenant le chiffre 8. Dans le cadre

de notre approche, on s'intéresse naturellement au problème classique de discrimination à 10 classes, qui est la tâche complète proposée par MNIST et qui est abordé sans problème par notre modèle multi-classe.

### 5.3.2 Protocole expérimental et paramétrage

Dans notre approche de l'apprentissage avec information privilégiée, on a distingué deux manières principales d'aborder ce problème : une famille de méthodes s'appuie sur l'information privilégiée pour déterminer une pondération des exemples dans l'espace ciblé, tandis que d'autres approches s'intéressent plutôt à optimiser un coût relatif entre les modèles appris dans chacun des espaces, traduisant une volonté de ressemblance ou de dissemblance entre ces modèles (*cf.* Chapitre 4). Notre méthode deep+ s'appuie principalement sur la première démarche, c'est pourquoi on compare les performances de notre modèle avec celles des deux approches de l'état de l'art retenues dans la partie 4.2 : SVM+ (Vapnik and Vashist, 2009) (*cf.* section 4.2.1) et Margin Transfer (Sharmanska et al., 2014) (*cf.* section 4.2.2). On s'intéressera aux méthodes de l'état de l'art LUPI basées sur un coût de ressemblance dans le chapitre 6.

On compare trois types d'approches : les représentations d'images dans l'espace ciblé classées par un SVM linéaire multi-classe (*i.e.* sans information privilégiée, notées *Représentation ciblée*), les méthodes de l'état de l'art LUPI (SVM+ et Margin Transfer), et notre modèle deep+. Les deux premières approches sont basées sur des classifieurs SVM ; le protocole expérimental utilisé pour ces méthodes est résumé dans le tableau 5.1. Deep+ en revanche permet d'intégrer l'apprentissage d'un réseau de neurones profond, c'est pourquoi le protocole utilisé pour notre méthode est résumé dans le tableau 5.2. On détaille maintenant le protocole expérimental pour chaque type d'approche testée.

**Représentations ciblées.** Sur chacune des trois bases de données, on reporte les performances des représentations d'images ciblées sans information privilégiée, afin de montrer les résultats de cette expérience de référence. Pour ce faire, on utilise des représentations adaptées à chaque base. Sur MNIST, on apprend un LeNet sur les images ciblées. On reporte les résultats du réseau complet. Sur FGVC-Aircraft et UPMC-Food-101, on utilise des représentations VGG-M_fc1, extraites de la même manière que dans le chapitre 3, *i.e.* sur les images $32 \times 32$ ré-agrandies à $224 \times 224$ via le protocole décrit par la figure 3.3. Sur ces représentations, on apprend un SVM multi-classe linéaire avec un $C = 10^5$ sur FGVC-Aircraft, et $C = 10^3$ sur UPMC-Food-101, qui contient plus de données.

**Méthodes de l'état de l'art LUPI.** Sur chaque base de données, on apprend les modèles SVM+ et Margin Transfer sur les mêmes données ciblées, en s'aidant des mêmes données privilégiées. On prend dans l'espace ciblé les mêmes représentations que pour les représentations seules, citées ci-dessus. Sur MNIST, on prend les représentations LeNet_fc1 issu du réseau appris sur les données ciblées. Dans l'espace privilégié, on apprend un LeNet sur les images $28 \times 28$ de MNIST, et on utilise un réseau VGG-M pré-entraîné sur ImageNet pour les deux bases orientées grain-fin. Les représentations privilégiées sont la sortie de la première couche totalement connectée de LeNet (notée LeNet_fc1) sur MNIST, et la représentation VGG-M_fc1 sur les images des bases orientées grain-fin.

Pour SVM+, on utilise l'implémentation proposée par Pechyony and Vapnik (2011)[2], pour laquelle on prend un paramétrage standard $C = 10^5$ sur FGVC-Aircraft et MNIST, et $C = 10^3$ sur UPMC-Food-101, qui contient plus de données. Pour Margin Transfer, on utilise la bibliothèque LibLinear (Fan et al., 2008) pour implémenter les deux SVM (dans l'espace privilégié puis dans l'espace ciblé). On reporte les résultats en prenant $C^* = 100$ et $C = 10^5$. En suivant les travaux de Sharmanska et al. (2014), dans notre réimplémentation de Margin Transfer, on seuille tous les $\rho_i$ à 0,1. Comme ces deux algorithmes se basent sur des formulations binaires, on se ramène à une formulation *un contre un* sur MNIST, et *un contre tous* sur FGVC-Aircraft et UPMC-Food-101.

**Deep+.** Notre méthode s'appuie sur l'apprentissage d'un CNN dans l'espace ciblé. Sur chaque base de données, on utilise un réseau adapté à la tâche : LeNet sur MNIST, et LR-CNN, le réseau profond proposé dans la partie 3.3.1, sur les deux bases de données orientées grain-fin.

Dans notre modèle, les $\rho_i$ peuvent être calculés de deux manières différentes : avec un SVM dans l'espace privilégié, ou avec un CNN dans l'espace privilégié. Pour la version deep+ avec un SVM dans l'espace privilégié (notée "Deep+, $\rho_i^{SVM}$" dans les tableaux), on prend les mêmes représentations privilégiées que pour les méthodes LUPI, sur lesquelles on apprend un SVM linéaire multi-classe avec $C^* = 100$. Les $\Delta_i^{SVM}$ sont ensuite seuillés à $\tau_{min} = 0,1$ pour forcer tous les exemples à être correctement reconnus, et plafonnés à $\tau_{max} = 10$.

Pour la version deep+ avec un CNN dans l'espace privilégié (notée "Deep+, $\rho_i^{CNN}$" dans les tableaux), on utilise un réseau adapté aux images privilégiées sur chacune des bases. Sur MNIST, on utilise le réseau LeNet entraîné sur les images privilégiées. Sur UPMC-Food-101, on réinitialise les poids de la dernière couche totalement connectée d'un VGG-M pour avoir le bon nombre de sorties (*i.e.* le nombre de classes de la base), puis on adapte (*fine-tune*) les poids de toutes les couches. Autrement dit, on rétro-propage le gradient de l'erreur à-travers toutes les couches. Pour cela, on prend un pas d'apprentissage 10 fois supérieur pour la dernière couche que pour les poids des autres couches. On adopte un protocole similaire sur FGVC-Aircraft. En revanche, comme cette base contient assez peu de données, on ne rétro-propage le gradient que dans la dernière couche totalement connectée afin d'éviter le sur-apprentissage On adapte le paramétrage des $\rho_i^{CNN}$ sur chaque base de données. Sur MNIST, on prend $\tau_{min} = 0,5$ et $\tau_{max} = 2$ avec un remappage linéaire. Sur UPMC-Food-101, on fixe $\tau_{min} = 0,1$ et $\tau_{max} = 1$ avec un remappage exp2. Enfin, on prend $\tau_{min} = 0,1$ et $\tau_{max} = 1$ avec un remappage linéaire sur FGVC-Aircraft.

### 5.3.3 Résultats

On présente les résultats des méthodes testées sur les différentes bases dans le tableau 5.3. Sur MNIST, on montre que notre méthode deep+ permet de commettre moins d'erreurs que les méthodes de l'état de l'art d'apprentissage avec information privilégié testées. En effet, on commet 12 erreurs de moins que Margin Transfer, et 15 de moins que SVM+. Ces résultats prouvent donc que notre méthode parvient à intégrer l'information privilégiée avec succès dans l'apprentissage du réseau.

2. http://www.cs.technion.ac.il/~pechyony/

| | **MNIST** | **UPMC-Food-101** | **FGVC-Aircraft** |
|---|---|---|---|
| **Espace privilégié** | | | |
| Images | $28 \times 28$ | $224 \times 224$ | $224 \times 224$ |
| CNN | LeNet appris sur images privilégiées | VGG-M pré-entraîné sur ImageNet | VGG-M pré-entraîné sur ImageNet |
| Représentations | LeNet_fc1 | VGG-M_fc1 | VGG-M_fc1 |
| **Espace ciblé** | | | |
| Images | $10 \times 10$ agrandies à $28 \times 28$ | $32 \times 32$ agrandies à $224 \times 224$ | $32 \times 32$ agrandies à $224 \times 224$ |
| CNN | LeNet appris sur images ciblées | VGG-M pré-entraîné sur ImageNet | VGG-M pré-entraîné sur ImageNet |
| Représentations | LeNet_fc1 | VGG-M_fc1 | VGG-M_fc1 |

TABLE 5.1 – Tableau récapitulatif des représentations utilisées pour les méthodes de l'état de l'art LUPI basées sur des classifieurs de type SVM : SVM+ (Vapnik and Vashist, 2009) et Margin Transfer (Sharmanska et al., 2014).

| | **MNIST** | **UPMC-Food-101** | **FGVC-Aircraft** |
|---|---|---|---|
| **Espace privilégié** | | | |
| Images | $28 \times 28$ | $224 \times 224$ | $224 \times 224$ |
| CNN, $\rho_i^{SVM}$ | LeNet appris sur images privilégiées | VGG-M pré-entraîné sur ImageNet | VGG-M pré-entraîné sur ImageNet |
| CNN, $\rho_i^{CNN}$ | LeNet appris sur images privilégiées | VGG-M pré-entraîné sur ImageNet, FT sur images priv | VGG-M pré-entraîné sur ImageNet, fc3 FT sur images priv |
| Représentations (calcul des $\rho_i^{SVM}$) | LeNet_fc1 | VGG-M_fc1 | VGG-M_fc1 |
| **Espace ciblé** | | | |
| Images | $10 \times 10$ agrandies à $28 \times 28$ | $32 \times 32$ | $32 \times 32$ |
| CNN (appris sur les images ciblées) | LeNet | LR-CNN | LR-CNN |

TABLE 5.2 – Tableau récapitulatif du protocole expérimental pour les modèles basés sur des réseaux de neurones profonds : nos modèles deep+ (Chapitre 5) et deep++ (Chapitre 6), ainsi que Generalized Distillation (Lopez-Paz et al., 2016).

| Méthode | MNIST | UPMC-Food-101 | FGVC-Aircraft |
|---|---|---|---|
| Représentation ciblée | 130 erreurs | 28,9% | 32,7% |
| SVM+ | 122 erreurs | - | 31,2% |
| Margin Transfer | 119 erreurs | 30,6% | 33,5% |
| **Deep+**, $\rho_i^{SVM}$ | **108 erreurs** | **32,3%** | **40,0%** |
| **Deep+**, $\rho_i^{CNN}$ | **107 erreurs** | **31,6%** | **39,8%** |

TABLE 5.3 – Comparaison de notre méthode deep+ aux méthodes de l'état de l'art LUPI. Notre modèle deep+ (basé sur des réseaux de neurones) est appris via le protocole décrit dans le tableau 5.2. Les autres méthodes, basées sur des approches SVM, suivent le protocole décrit dans le tableau 5.1.

Sur UPMC-Food-101, notre méthode améliore de 1,7% les performances de Margin Transfer[3]. Il convient de souligner que UPMC-Food-101 est une base de données multi-classe orientée grain-fin, et qui contient donc un nombre important de classes à distinguer. Les méthodes état de l'art binaires comme Margin Transfer et SVM+ ne sont donc pas particulièrement appropriées pour ce type de tâche, alors que notre méthode permet de traiter directement les formulations multi-classes. De plus, cette base de données contient beaucoup d'images d'apprentissage, ce qui est facilement traité par notre méthode, alors que les méthodes état de l'art telles que SVM+ ne sont pas du tout adaptées à ce genre de situation. Enfin, la base de données contient un certain niveau de bruit d'annotation, que notre modèle parvient manifestement à mieux traiter que Margin Transfer.

Enfin, sur FGVC-Aircraft, notre méthode surpasse également les performances des deux méthodes concurrentes. En effet, notre modèle améliore de 6,3% les performances de Margin Transfer, et de 8,6% celles de SVM+. Comme on le montrera dans la partie 5.3.4, une partie importante de ce gain de performances est issue de l'utilisation d'un modèle profond dans l'espace ciblé pour notre méthode, alors que les méthodes de l'état de l'art sont basées sur un classifieur SVM linéaire. La possibilité d'utiliser un réseau profond permet donc à notre modèle d'améliorer les performances par rapport aux méthodes de l'état de l'art LUPI.

On montre bien que notre modèle tire réellement avantage de l'information privilégiée pour améliorer les représentations internes du réseau. Enfin, notre méthode prend directement en compte l'aspect multi-classe des problèmes traités, alors que les méthodes état de l'art testées requièrent une formulation *ad hoc*.

#### Bilan

Notre modèle deep+ permet d'améliorer les performances par rapport aux méthodes de l'état de l'art sur toutes les bases explorées. En particulier, notre modèle permet d'apprendre directement de bout en bout l'ensemble des représentations profondes de l'image (*i.e.* les représentations internes au réseau) en bénéficiant de l'information privilégiée. A contrario, les méthodes de l'état de l'art testées n'utilisent l'information privilégiée que

3. Nous ne reportons pas de performances pour SVM+, car le code en ligne utilise une optimisation dans l'espace dual, inadaptée au nombre important de données d'apprentissage de UPMC-Food-101.

dans l'étape de classification. Contrairement aux méthodes de l'état de l'art LUPI testées, notre modèle permet de faire bénéficier les représentations internes de l'information privilégiée, ce qui peut favoriser de meilleures performances de reconnaissance.

De plus, notre modèle permet de prendre en compte directement l'aspect multi-classe des problèmes abordés, alors que pour les méthodes de l'état de l'art testées, on doit avoir recours à des formulations binaires *ad hoc*. Cela peut s'avérer particulièrement coûteux, notamment dans le cas de larges bases de données comme UPMC-Food-101, que notre modèle parvient à aborder sans difficulté.

### 5.3.4 Analyse de notre modèle deep+

Comparons maintenant les performances d'un CNN standard avec celles de ce même CNN appris avec notre modèle deep+.

#### Protocole expérimental et paramétrage

Pour chaque base de données, on apprend en premier lieu un CNN standard sur les données faiblement résolues sans information privilégiée (noté "Deep") : c'est notre expérience de référence. D'autre part, on apprend le même CNN avec l'approche deep+ proposée, afin de pouvoir intégrer l'information privilégiée. Le CNN appris sur chacune des trois bases de données est adapté à la tâche ciblée. Ainsi, on prend un LeNet sur MNIST, et un LR-CNN sur les deux bases orientées grain-fin. Les hyper paramètres d'apprentissage du réseau deep+ sont les mêmes que ceux utilisés lors de l'apprentissage du réseau sans information privilégiée. Ainsi, on met bien en évidence l'apport de notre méthode par rapport à un réseau standard. Le protocole expérimental pour les deux variantes de la méthode deep+ pour chaque base de données, résumé dans le tableau 5.2, est exactement le même que dans la section 5.3.2.

#### Résultats

Les performances deep et deep+ sur les trois bases de données sont reportées dans le tableau 5.4. Sur MNIST, le réseau LeNet appris via notre méthode deep+ commet beaucoup moins d'erreurs que le LeNet standard. En effet, avec notre formulation SVM, LeNet+ commet 22 erreurs de moins que LeNet, et notre formulation CNN permet de gagner encore une erreur. On prouve donc l'intérêt de notre méthode sur cette base de données. Par ailleurs, les deux variantes de deep+ proposées sont compétitives.

Sur UPMC-Food-101, notre méthode parvient à améliorer de 1,6% les résultats de LR-CNN. Ces résultats montrent que notre modèle parvient à traiter efficacement le problème délicat de la présence d'aberrations dans l'ensemble d'apprentissage, en donnant un poids important aux exemples représentatifs de leur classe, et en attribuant un poids assez faible aux exemples les plus difficiles, *i.e.* les potentielles aberrations ou les exemples peu représentatifs de leur classe.

Sur FGVC-Aircraft, notre méthode d'intégration d'information privilégiée dans le réseau de neurones ne permet pas d'augmenter significativement les performances. Cette base de données est en effet de moyenne envergure (moins de 7 000 images d'apprentissage), ce qui peut représenter un obstacle à l'apprentissage d'un CNN dans l'espace privilégié, et donc limiter quelque peu l'apport de notre modèle deep+. De plus, comme

| Méthode | MNIST | UPMC-Food-101 | FGVC-Aircraft |
|---|---|---|---|
| Deep | 130 erreurs | 30,7% | 39,8% |
| Deep+, $\rho_i^{SVM}$ | 108 erreurs | 32,3% | 40,0% |
| Deep+, $\rho_i^{CNN}$ | 107 erreurs | 31,6% | 39,8% |

TABLE 5.4 – Apport de notre méthode deep+ par rapport à un réseau sans information privilégiée. Tous les modèles sont appris via le protocole décrit dans le tableau 5.2.

on l'a montré dans la partie 3.3.1, le fait d'apprendre un réseau de neurones sur les données ciblées permet déjà d'augmenter largement les performances.

Par ailleurs, on montre que sur les trois bases, les performances varient très peu en fonction de la méthode utilisée pour construire les $\rho_i$. Par exemple, la méthode avec $\rho_i^{CNN}$ fait une erreur de moins que la méthode avec $\rho_i^{SVM}$ sur MNIST. Ainsi, en raison du faible impact de cette méthode sur les performances de classification, on s'intéressera pour la suite de nos travaux à la formulation avec CNN dans l'espace privilégié.

#### Bilan

Notre modèle deep+ permet d'améliorer les performances d'un réseau de neurones sans information privilégiée sur la majorité des bases de données testées. Il démontre notamment de bons résultats sur UPMC-Food-101, ce qui semble indiquer la robustesse de notre approche par rapport à la présence de bruit d'annotation.

On a également montré que notre modèle deep+ permet d'améliorer les performances par rapport aux méthodes de l'état de l'art LUPI testées sur toutes les bases explorées. Plusieurs facteurs peuvent expliquer ce gain de performances. En particulier, notre modèle permet d'apprendre directement les représentations de l'image (*i.e.* les représentations internes au réseau) en bénéficiant de l'information privilégiée ; a contrario, les méthodes de l'état de l'art testées n'utilisent l'information privilégiée que dans l'étape de classification. De plus, notre modèle permet de prendre en compte directement l'aspect multi-classe des problèmes abordés, alors que pour les méthodes de l'état de l'art testées, on doit avoir recours à des formulations binaires *ad hoc*. Par ailleurs, cela peut s'avérer particulièrement coûteux, notamment dans le cas de larges bases de données comme UPMC-Food-101, que notre modèle parvient à aborder sans difficulté. Il serait intéressant d'investiguer plus en détails ces différents aspects. Enfin, on souligne que, sur MNIST, les expériences menées permettent de mesurer directement le gain de performances de notre modèle, puisque les représentations ciblées utilisées pour deep+ sont les mêmes que pour les méthodes LUPI.

### 5.3.5 Deep+ sur les images Thales

#### Description de la base de données

Dans un contexte opérationnel, Thales Optronique propose des produits pour la reconnaissance de véhicules à longue distance. L'apprentissage des algorithmes de classification requiert un nombre suffisant d'images d'apprentissage annotées, notamment dans le cadre de l'utilisation des réseaux de neurones. Le coût d'annotation étant cependant élevé pour

| | Images $32 \times 32$ | | Images $64 \times 64$ | |
|---|---|---|---|---|
| | sans FA | avec FA | sans FA | avec FA |
| CNN | 95,91% | 96,32% | 98,50% | 99,02% |
| CNN+, $\rho_i^{SVM}$ | 94,53% | 95,13% | 98,05% | 97,95% |
| CNN+, $\rho_i^{CNN}$ | 96,00% | 96,07% | 98,65% | 99,03% |

TABLE 5.5 – Apport de notre méthode deep+ sur les images simulées Thales.

les images acquises sur le terrain, il n'est cependant pas aisé d'entraîner les algorithmes sur des données réelles. Pour pallier ce problème, Thales Optronique s'est dotée d'une large base de données simulées, détaillée dans l'annexe A.

Dans cette étude, on explore deux cadres applicatifs différents au niveau de la résolution des images ciblées : dans un cas, ces images ont une taille de $32 \times 32$. On apprend alors un LR-CNN sur ces données. Dans l'autre cas, il s'agit des mêmes images, mais à résolution $64 \times 64$. Sur ces images, on utilise un CNN dont l'architecture est décrite dans l'annexe A. Les images privilégiées (*i.e.* fortement résolues) ont toutes une taille $224 \times 224$. Sur ces données, on utilise un réseau VGG-M (Chatfield et al., 2014). Pour calculer les $\rho_i^{SVM}$, on extrait les représentations VGG-M_fc1 sur les images privilégiées. Pour calculer les $\rho_i^{CNN}$, on affine les poids du réseau VGG-M sur les images privilégiées. Seuls les poids de la dernière couche sont appris sur la base de données simulées ; les poids des autres couches sont gelés. Pour le calcul des $\rho_i^{CNN}$, on prend ici $\tau_{min} = 0,1$ et $\tau_{max} = 2$.

Comme décrit dans l'annexe A, cette base de données contient des fausses alarmes (ou FA), qui sont des images de faible résolution ne contenant pas de véhicule. Puisqu'on traite dans ces travaux au problème de la reconnaissance et non pas de la détection, on s'intéresse principalement aux résultats sur la base de données sans ces images de FA. Il est néanmoins intéressant de voir les résultats en prenant en compte ces fausses alarmes. Ainsi, pour chacun des deux cadres applicatifs mentionnés plus haut, on reporte d'une part les performances sans prendre en compte les fausses alarmes de la base de données, et d'autre part les performances tenant compte de ces fausses alarmes comme une classe à part entière. Dans ce dernier cas, il convient de remarquer que l'information privilégiée n'est disponible pour aucune des images de fausses alarmes. Pour ces images-là, on fixe donc $\rho_i^{SVM} = 1$ et $\rho_i^{CNN} = 1$.

**Résultats expérimentaux**

On reporte les résultats des différentes méthodes appliquées à cette base dans le tableau 5.5. Notre modèle avec CNN dans l'espace privilégié parvient à améliorer quelque peu les performances du CNN sans information privilégiée dans les deux cadres d'application sans fausses alarmes. En effet, notre deep+ dépasse de 0,1% les performances du réseau standard sur les images $64 \times 64$, et de 0,2% dans le cas des images $32 \times 32$. On arrive donc à bénéficier quelque peu de l'information privilégiée dans notre modèle.

On s'intéresse également aux cas où des fausses alarmes se trouvent dans l'ensemble de test. Dans cette situation, on remarque que les performances de notre modèle avec CNN dans l'espace privilégié restent stables malgré la présence d'une classe supplémentaire

dans l'espace ciblé par rapport à l'espace privilégié. En effet, comme on ne dispose pas des versions hautement résolues des images de fausses alarmes, ces images n'ont pas d'information de difficulté associée dans l'espace ciblé. Malgré cela, notre modèle parvient à rester assez stable dans ce contexte.

Notons enfin que les performances de notre deep+ avec les $\rho_i^{SVM}$ sont systématiquement moins bonnes que lorsqu'on utilise les $\rho_i^{CNN}$. Une cause pouvant expliquer ce phénomène est qu'on utilise un SVM linéaire dans l'espace privilégié, sur environ 150 000 représentations de taille 4 096. Si la capacité du SVM s'avère trop faible pour le problème traité, cela peut expliquer que les résultats soient légèrement moins bons qu'avec un CNN dans l'espace privilégié. Il convient cependant de constater que les performances restent correctes, ce qui montre que notre méthode avec SVM parvient à apprendre des coefficients permettant de traiter le problème dans l'espace ciblé.

### 5.3.6 Étude approfondie de notre modèle deep+

On explore ici différents aspects de notre modèle. On valide d'abord la cohérence de nos définitions des $\rho_i$, puis la capacité de notre modèle à caractériser les bases de données difficiles de celles qui sont plus facilement traitées. On démontre ensuite l'intérêt de notre modèle lorsque le nombre d'images d'apprentissage est restreint. Enfin, on montre la capacité de notre approche deep+ à discriminer au mieux les classes les plus proches sémantiquement parlant.

#### Pertinence des $\rho_i$

On s'intéresse ici à la cohérence des coefficients de difficulté $\rho_i$ calculés via chacune de nos deux approches : $\rho_i^{SVM}$ (eq. 5.2 et 5.3) et $\rho_i^{CNN}$ (eq. 5.4 et 5.5). On reporte d'abord sur la figure 5.3 les images associées aux valeurs extrêmes de $\Delta_i$ (*i.e.* des $\rho_i$ avant remappage). Pour chacune des 10 classes de MNIST, l'image $28 \times 28$ associée à la valeur de $\Delta_i^{SVM}$ la plus élevée (resp. la plus faible) est présentée sur la ligne du haut (resp. la ligne du bas). Ainsi, la ligne du haut est censée contenir les images les plus facilement reconnaissables, alors que la ligne du bas ne devrait contenir que des images difficiles à reconnaître. Visuellement, on vérifie bien que les images de la ligne du haut sont très claires et que leur classe est très facile à déterminer. En revanche, les images de la ligne du bas sont assez difficiles à classer du fait de leur ambiguïté. Par exemple, l'image de "0" la plus difficile ressemble très fortement à l'image de "6" la plus délicate à classer ; il en va de même pour le "1" et le "7". On répète le même processus pour six classes choisies aléatoirement au sein de UPMC-Food-101, et on reporte les images sur la figure 5.4. On vérifie bien que les images censées être les plus faciles sont bien représentatives de leur classe. En revanche, les images ayant la valeur $\rho_i$ la plus faible peuvent représenter des étapes intermédiaires de la préparation (*e.g.* cannoli), être centrées sur un détail non discriminant du plat (*e.g.* l'image de soupe à l'oignon centrée sur le pain) ou être des aberrations de la base (*e.g.* les biscuits considérés comme des macarons). Cette étude permet de mettre en évidence la capacité de notre modèle à discriminer d'une part les exemples les plus représentatifs de leur classe, et d'autre part les exemples les plus difficiles voire les aberrations, qu'un œil humain peinerait lui-même à reconnaître convenablement.

De même, pour quatre classes aléatoires de UPMC-Food-101, on reporte sur la figure 5.5 les exemples associés à la valeur de $\Delta_i^{CNN}$ la plus faible (resp. la plus élevée) sur la

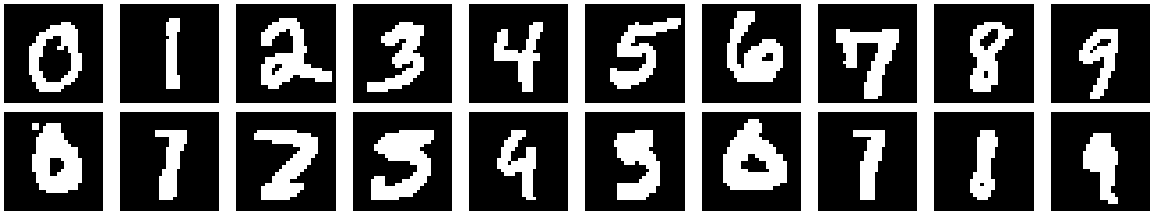


FIGURE 5.3 – Étude de la pertinence des $\rho_i^{SVM}$ calculés sur MNIST. Pour chacune des 10 classes (en colonne), la ligne du haut (resp. du bas) contient l'image $28 \times 28$ associée à la valeur de $\Delta_i^{SVM}$ la plus élevée (resp. la plus faible). Visuellement, on vérifie que les images avec des valeurs de $\Delta_i^{SVM}$ élevées sont facilement reconnaissables, alors que les images avec un $\Delta_i^{SVM}$ faible sont beaucoup plus floues ou de basse qualité, si bien qu'elles pourraient être mal classées même par un œil humain.


FIGURE 5.4 – Étude de la pertinence des $\rho_i^{SVM}$ calculés sur UPMC-Food-101. Pour six des classes de UPMC-Food-101 (*bruschetta, cannoli, caprese salad, frozen yogurt, french onion soup, macarons*), on présente l'image associée au $\Delta_i^{SVM}$ le plus élevé (resp. le plus faible) sur la ligne du haut (resp. du bas). Pour toutes les classes, l'image associée au $\Delta_i^{SVM}$ le plus élevé est très représentative de sa classe, alors que l'image associée au $\Delta_i^{SVM}$ le plus faible est soit une aberration, soit un exemple difficile à interpréter visuellement.

ligne du bas (resp. du haut). On montre ainsi que les images associées aux valeurs les plus élevées de $\rho_i^{CNN}$ sont effectivement assez facilement reconnaissables, quelle que soit la classe. En revanche, les images associées aux $\rho_i^{CNN}$ les plus faibles sont plus difficiles à catégoriser, puisqu'elles peuvent être des formes peu courantes du plat en question (*e.g.* la salade grecque ressemble beaucoup à un plat de bruschetta, une autre catégorie de cette base), voire des aberrations difficiles (*e.g.* une image de pizza catégorisée comme un plat de nachos).

#### Caractérisation des bases

On s'intéresse maintenant à montrer la capacité de notre modèle à refléter la difficulté d'une base de données. En effet, on vient de prouver la cohérence des $\rho_i$ sur des exemples individuels, mais cette cohérence doit également s'appliquer de manière plus générale à la base de données tout entière. On se penche donc ici sur la distribution des $\Delta_i$ sur les différentes bases étudiées.


FIGURE 5.5 – Étude de la pertinence des $\rho_i^{CNN}$ calculés sur UPMC-Food-101. Pour quatre classes aléatoires de UPMC-Food-101 (*beef tartare, greek salad, chocolate mousse, nachos*), on reporte l'image associée à la valeur de $\rho_i^{CNN}$ la plus élevée (resp. la plus faible) sur la ligne du haut (resp. du bas).

Sur la figure 5.6, on reporte la distribution des $\Delta_i^{SVM}$ (*i.e.* les $\rho_i^{SVM}$ avant seuillage, eq. 5.2) sur chacune des trois bases de données. On remarque en premier lieu que, sur différents aspects, les distributions sont assez similaires sur les deux bases orientées grain-fin. D'abord, une forte proportion des $\Delta_i^{SVM}$ est proche de 1. En effet, pour chaque image d'apprentissage, le SVM multi-classe dans l'espace privilégié force intrinsèquement une marge supérieure ou égale à 1 entre le score de la classe vérité-terrain et celui de toutes les autres classes (eq. 5.1). De fait, pour les exemples difficilement reconnus, la marge forcée est la plus faible possible, *i.e.* 1. Ici, la proportion de $\Delta_i^{SVM}$ ayant une valeur comprise entre 0,98 et 1,02 est de 51,7% sur FGVC-Aircraft, et de 63,8% sur UPMC-Food-101. D'autre part, les deux bases contiennent une proportion non nulle de $\Delta_i^{SVM}$ inférieurs à 1. Cela caractérise les exemples pour lesquels il n'est même pas possible de forcer une marge à 1, *i.e.* les exemples les plus difficiles et les aberrations. Par exemple, 22,3% des images de UPMC-Food-101 ont une valeur de $\Delta_i^{SVM}$ inférieure à 0,98, et 5,0% des images de cette base ont même un $\Delta_i^{SVM}$ négatif. Tous ces éléments caractérisent bien des bases de données orientées grain-fin, *i.e.* constituées d'images difficiles à reconnaître. Dans le cas de UPMC-Food-101, la présence de $\Delta_i^{SVM}$ négatifs reflète la présence de bruit d'annotation, dû au fait que les images ont été récoltées sur le web.

On montre également que notre modèle parvient à extraire une information riche sur des images plus facilement reconnaissables. Sur MNIST, 68,0% des $\Delta_i^{SVM}$ sont supérieurs à 3. Comme cette base est plus facile à traiter, les coefficients de difficulté sont plus élevés en moyenne, alors que sur les deux bases orientées grain-fin, une moindre proportion des $\rho_i$ sont supérieurs à 1 (46,8% sur FGVC-Aircraft, et 13,8% sur UPMC-Food-101). De plus, notre modèle parvient toujours à déterminer différents niveaux de facilité parmi ces exemples : 5,8% des images de UPMC-Food-101 ont un $\rho_i$ supérieur à 1,5, et cette proportion est de 10,2% sur FGVC-Aircraft. Cette étude montre bien que notre modèle parvient à extraire de l'information particulièrement utile de l'information privilégiée.

De la même manière, on présente sur la figure 5.7 la répartition des $\Delta_i^{CNN}$ (eq. 5.4) sur

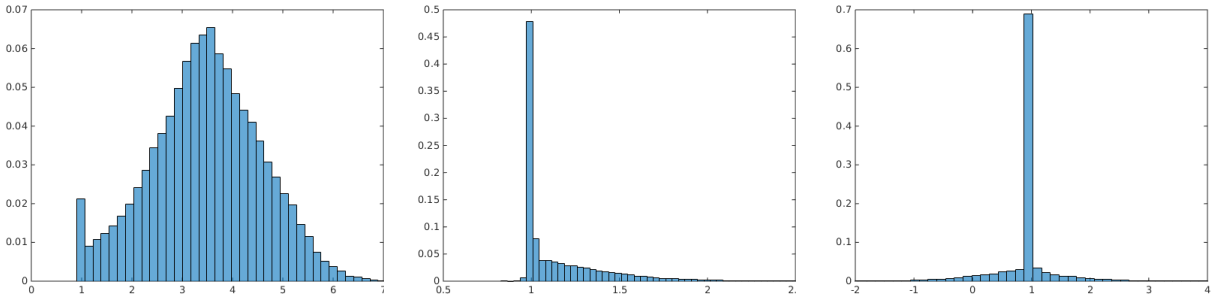



FIGURE 5.6 – Distribution des $\Delta_i^{SVM}$ sur MNIST (gauche), FGVC-Aircraft (milieu) et UPMC-Food-101 (droite). Notre modèle parvient à caractériser la difficulté de chaque base de données.

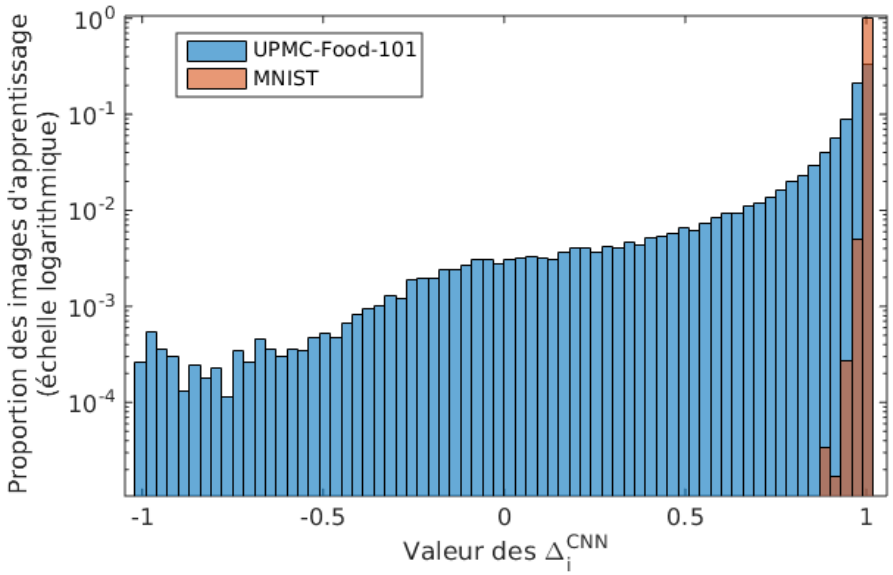



FIGURE 5.7 – Répartition des $\Delta_i^{CNN}$ sur les bases UPMC-Food-101 (en bleu) et MNIST (en orange). Note : pour plus de lisibilité, les distributions sont présentées en échelle logarithmique.

les bases UPMC-Food-101 (en bleu) et MNIST (en orange). Sur ce graphique, on montre bien que sur une base assez facile telle que MNIST, les $\Delta_i^{CNN}$ sont tous très élevés, *i.e.* les images sont toutes identifiées comme assez simples. Sur cette base, tous les $\Delta_i^{CNN}$ sont supérieurs à 0,85. En revanche, on montre bien que les $\Delta_i^{CNN}$ sont plus répartis sur la base UPMC-Food-101 : cette base étant constituée d'images récoltées sur le web, elle contient un certain bruit d'annotation, donc des images difficiles à reconnaître et des aberrations. Ici, on voit que les $\rho_i^{CNN}$ sont assez répartis entre les deux bornes ; plus de 2 000 images ont un $\Delta_i^{CNN}$ négatif. Cette étude montre bien la capacité de notre modèle à caractériser les exemples difficiles et les aberrations d'annotation.

**Effets des différents remappages des $\rho_i^{CNN}$**

On représente sur la figure 5.8 la répartition des $\rho_i^{CNN}$ suite aux différents types de remappage décrits dans la section 5.2.1. Les histogrammes notés $h_{lin}$ (en orange), $h_{exp}$ (en

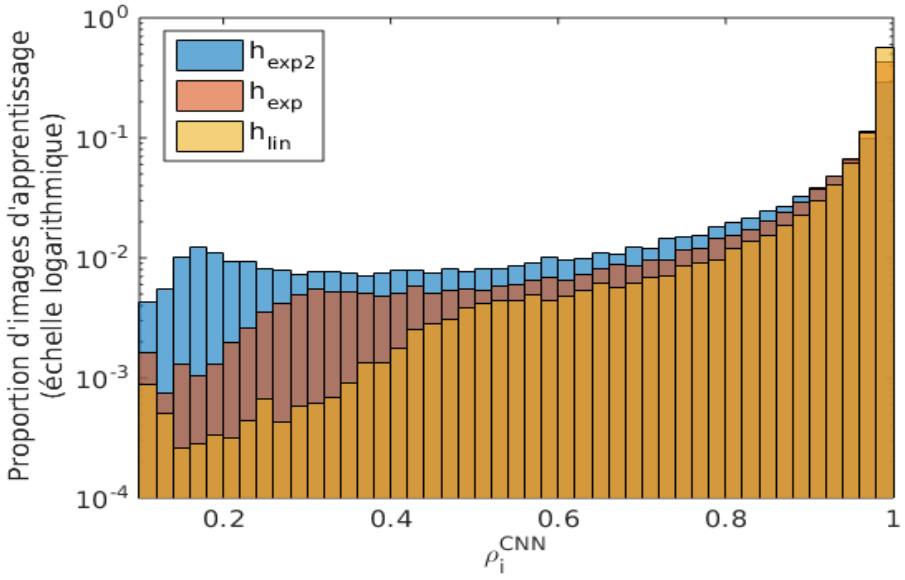



FIGURE 5.8 – Influence des différents types de remappage sur la distribution des $\rho_i^{CNN}$ sur UPMC-Food-101. Note : pour plus de lisibilité, les quantités sont présentées en échelle logarithmique.

rouge) et $h_{exp2}$ (en bleu) représentent la répartition des $\rho_i^{CNN}$ suite aux remappages décrits respectivement par les équations 5.5, 5.6 et 5.7. Pour chaque remappage, les $\rho_i^{CNN}$ finaux sont compris dans l'intervalle $[0,1;1]$. Sur ce graphique, on montre bien que le passage aux remappages non linéaires permet de mieux répartir les $\rho_i^{CNN}$ entre les bornes. En effet, avec un remappage linéaire, 70,5% des $\rho_i^{CNN}$ sont compris entre 0,95 et 1, alors que ce taux passe à 57,5% avec $h_{exp}$, puis à 42,4% avec $h_{exp2}$. Au contraire, seulement 2,4% des $\rho_i^{CNN}$ obtenus avec $h_{lin}$ sont compris entre 0,1 et 0,5, alors qu'avec $h_{exp}$ ce taux s'élève à 7,6%, et le remappage $h_{exp2}$ fait encore augmenter cette proportion à 16,2%. Au-travers de cette étude, on montre bien qu'un remappage bien choisi permet de mieux étaler le spectre des $\rho_i^{CNN}$, et permet ainsi de mieux exploiter l'information de difficulté des exemples dans l'espace privilégié.

**Influence du nombre d'exemples d'apprentissage**

On montre également l'intérêt de notre modèle dans le cas où peu d'images d'apprentissage sont disponibles. Pour ce faire, on compare les performances deep et deep+ (avec CNN dans l'espace privilégié) lorsqu'on réduit le nombre d'images d'apprentissage. Plus précisément, on exécute huit expériences indépendantes, pour lesquelles on se place dans un protocole expérimental similaire à celui de (Vapnik and Vashist, 2009) : pour chaque expérience, on sélectionne au sein de chacune des 10 classes une proportion $p$ d'images d'apprentissage de MNIST, tirées aléatoirement parmi les ensembles d'apprentissage et de validation de la base de données originale. Les deux réseaux sont appris sur ce sous-ensemble d'images d'apprentissage (avec la même structure LeNet et les mêmes paramètres d'apprentissage que dans 5.3.3), puis testés sur l'ensemble complet des images de test (10 000 images). Pour chacune des 8 expériences, on répète le processus sur 12

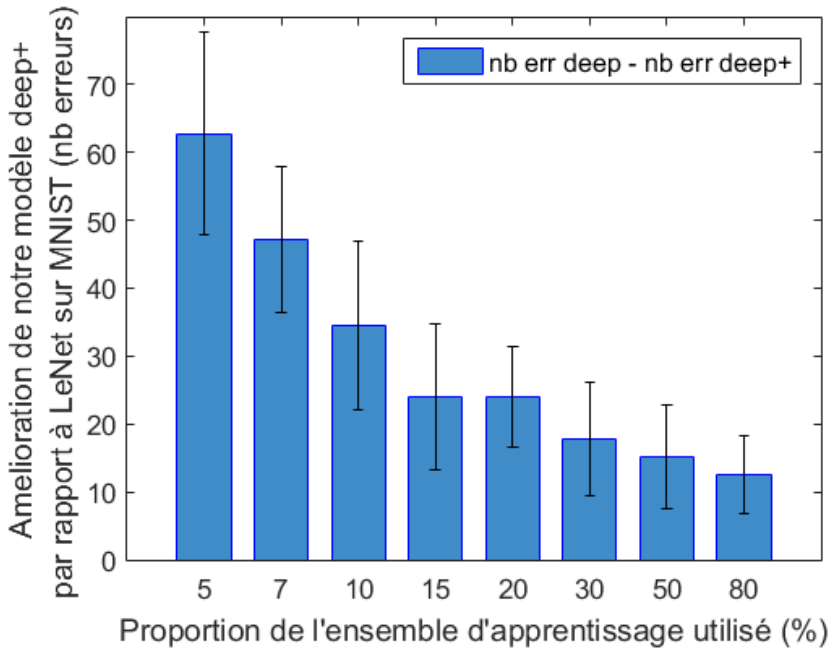



FIGURE 5.9 – Amélioration de deep+ par rapport à LeNet sur MNIST avec une proportion variable d'images d'apprentissage par classe. Quelle que soit la proportion, notre cadre deep+ permet d'apprendre un modèle plus précis que le réseau LeNet sans information privilégiée. De plus, l'amélioration de notre modèle augmente lorsque la taille de l'ensemble d'apprentissage diminue ; intégrer l'information privilégiée devient encore plus utile lorsqu'une plus petite quantité d'images d'apprentissage sont disponibles.

sous-ensembles aléatoires différents. Sur la figure 5.9, on reporte pour chaque proportion $p$ les moyennes et écart-types des valeurs $nb\_erreurs\_deep - nb\_erreurs\_deep+$. Ainsi, cette valeur est élevée lorsque LeNet fait beaucoup plus d'erreurs que notre modèle, *i.e.* lorsque notre modèle deep+ est meilleur.

Cette étude montre en premier lieu que, quel que soit le nombre d'images d'apprentissage, notre modèle atteint toujours de meilleures performances que le modèle standard. En moyenne, notre deep+ commet 12,6 erreurs de moins que le modèle LeNet avec 80% des images d'apprentissage, et 24,0 erreurs de moins lorsque seules 20% des images sont disponibles pendant l'apprentissage. De plus, les écart-types mesurés indiquent que ces valeurs sont significatives. D'autre part, on montre que l'intérêt de notre modèle croît à mesure que le nombre d'images d'apprentissage diminue. En effet, lorsque 50% des images d'apprentissage sont disponibles, notre modèle commet 15,2 erreurs de moins que LeNet ; avec 5% des données, LeNet fait 62,8 erreurs de plus que notre modèle.

Cette étude montre bien l'intérêt de l'intégration de l'information privilégiée grâce à notre modèle deep+, et à plus forte raison lorsque le nombre d'exemples d'apprentissage est limité. Cette observation rejoint l'argumentaire de Vapnik and Vashist (2009), pour qui l'intégration d'information privilégiée est avant tout un outil d'accélération de la convergence de l'algorithme d'apprentissage. Ces résultats, montrant qu'on parvient à obtenir de bien meilleures performances que le CNN sans information privilégiée avec bien moins d'exemples, offrent une perspective intéressante dans l'ère actuelle des réseaux CNN profonds, qui nécessitent beaucoup de données durant l'apprentissage.

## 5.4 Discussion

Dans ce chapitre, nous avons proposé deep+, une méthode d'intégration de l'information privilégiée dans les réseaux de neurones profonds. Cette approche s'appuie sur l'intuition que la difficulté d'un exemple dans l'espace privilégié peut être directement transférée dans l'espace ciblé. On utilise ainsi les représentations privilégiées pour calculer la difficulté de classifier un exemple d'apprentissage. Ce niveau de difficulté, traduit comme un coefficient $\rho_i$, permet de pondérer le coût de chaque exemple dans l'apprentissage d'un CNN dans l'espace ciblé, à-travers la fonction de coût $loss_{deep+}$. Les exemples les plus faciles à reconnaître ont un poids élevé, car ils sont de bons représentants de leur classe et constituent donc une information capitale. D'autre part, selon notre hypothèse de transfert, les exemples les plus difficiles dans l'espace privilégié seront très difficilement reconnaissables dans l'espace ciblé, et peuvent même être des aberrations de la base d'apprentissage. Dans ces deux cas, diminuer leur influence sur la fonction de décision apprise permet d'apprendre une fonction de décision robuste à ces aberrations.

Expérimentalement, on montre l'intérêt de notre approche sur plusieurs bases de données, et notamment dans le cadre de la classification à grain-fin, où notre modèle deep+ permet de bénéficier de l'information privilégiée pour mieux discriminer les classes les plus ambiguës. Le gain apporté par notre modèle lorsque le nombre d'exemples d'apprentissage est faible ouvre par ailleurs des perspectives intéressantes pour l'apprentissage avec une quantité restreinte d'exemples annotés.

À l'instar de Margin Transfer (Sharmanska et al., 2014), deep+ affecte un poids important aux exemples les plus faciles, et écrase le poids des exemples les plus difficiles, qui n'ont donc presque plus d'influence sur l'apprentissage. Cette approche se base sur l'intuition que de tels exemples sont de potentielles aberrations, et qu'il convient donc de réduire drastiquement leur influence. L'approche SVM+ (Vapnik and Vashist, 2009), en revanche, accorde à chaque exemple d'autant plus d'importance qu'il est mal classé. À l'inverse de Margin Transfer, cette approche se base sur l'idée que les exemples les plus difficiles sont en réalité les plus « importants » pour affiner le classifieur. Dans la suite, nous proposons d'unifier ces deux approches dans un modèle permettant de continuer à focaliser l'apprentissage sur les exemples les plus représentatifs sans pour autant oublier les exemples les plus difficiles.

# Chapitre 6

# Deep++

**Sommaire**



Dans le chapitre 5, on a proposé deep+, un modèle d'intégration de l'information privilégiée dans l'apprentissage des réseaux de neurones profonds. Ce modèle s'inscrit dans le sillon des méthodes basées sur l'utilisation de l'information privilégiée pour la pondération des exemples dans l'espace ciblé. Pour cela, on a proposé un cadre multi-classe de calcul des coefficients de pondération, donnant un poids plus important aux exemples les plus faciles selon l'espace privilégié, et diminuant l'influence des exemples les plus difficiles.

Comme décrit dans le chapitre 4, de nombreux travaux sur l'intégration de l'information privilégiée ont proposé d'autres approches du cadre LUPI. Plusieurs méthodes sont notamment basées sur un coût de ressemblance entre les modèles appris dans les deux espaces. Dans ce chapitre, on propose donc deep++, un modèle combinant les approches de l'état de l'art basées sur une formulation de coût relatif entre les espaces, avec les approches basées plutôt sur la pondération des exemples dans l'espace ciblé. On propose par ailleurs une analyse expérimentale détaillée de notre modèle, en montrant l'intérêt de cette nouvelle approche par rapport au modèle deep+ précédent et par rapport aux méthodes de l'état de l'art basées sur une approche de coût relatif. On s'attache également à analyser finement différents aspects de notre modèle.

## 6.1 Architecture deep++

Notre précédent modèle deep+ utilise l'information privilégiée pour calculer des coefficients $\rho_i$ reflétant le niveau de diffculté de chaque exemple d'apprentissage. Ces coefficients viennent ensuite pondérer les exemples dans l'espace ciblé, attribuant un poids important aux exemples les plus faciles et réduisant l'influence des exemples les plus difficiles. Ainsi, la difficulté d'un exemple est déterminée uniquement en fonction de son information privilégiée : on peut dire que cette notion de diffculté est *absolue*, dans le sens où elle ne dépend que de l'espace privilégié. À l'inverse, on peut considérer une approche *relative* de la difficulté, faisant intervenir les scores des deux espaces. Dans ce chapitre, on se propose de combiner ces deux approches de l'information privilégiée au sein d'un nouveau modèle deep++.

Afin d'affiner la prise en compte de l'information privilégiée dans notre modèle, nous proposons dans la suite deep++, une extension de notre modèle deep+ intégrant cette notion de difficulté relative entre les deux espaces de représentations. On représente sur la figure 6.1 le schéma global de notre algorithme deep++. On conserve l'architecture globale de deep+, où l'espace privilégié permet de calculer des coefficients de difficulté absolue $\rho_i$ (cadre vert). Néanmoins, on propose une nouvelle fonction de coût $loss_{deep++}$ pour intégrer l'information privilégiée dans l'apprentissage du CNN ciblé (cadre gris). À la pondération des exemples par le coefficient de difficulté absolue $\rho_i$ s'ajoute maintenant un terme supplémentaire mesurant l'éloignement entre les scores dans les deux espaces. Le but de ce terme de difficulté relative est de pénaliser les exemples dont le score de l'espace ciblé est trop faible par rapport à leur score dans l'espace privilégié. Autrement dit, on pénalise les exemples pour lesquels le scores ciblé de la bonne classe $\tilde{z}_c(x_i)$ est trop faible par rapport au score privilégié de l'exemple pour la bonne classe $\tilde{z}^*_c(x^*_i)$.

Notre idée est d'allier les deux approches absolue et relative de la difficulté des exemples, de manière à privilégier toujours plus fortement l'apprentissage des exemples les plus faciles, tout en augmentant simultanément l'influence des exemples qui ne sont pas *assez* bien classés dans l'espace ciblé par rapport à l'espace privilégié. L'intuition derrière cette pénalisation supplémentaire émane du fait que l'information privilégiée est plus riche que l'information ciblée, et qu'elle devrait ainsi permettre d'apprendre un modèle plus performant. Inciter le modèle ciblé à copier les réponses du modèle privilégié peut donc s'avérer bénéfique.

Comme souligné dans le chapitre 4, plusieurs modèles LUPI se basent sur cette intuition de ressemblance entre les deux espaces. C'est le cas par exemple de Generalized Distillation (Lopez-Paz et al., 2016), dans lequel on force les sorties du modèle ciblé à ressembler à celles du modèle privilégié. Le modèle SVM+ (Vapnik and Vashist, 2009) utilise l'information privilégiée comme un approximateur des variables ressorts $\xi_i$ de l'espace ciblé, alors remplacées par des scores dans l'espace privilégié. L'exemple $x_i$ n'a ainsi plus d'influence sur l'apprentissage tant qu'il est suffisamment bien classé, avec une marge $1-\xi_i$ qui dépend de l'espace privilégié. En revanche, dès qu'il ne respecte plus cette condition, il influence grandement l'apprentissage de la frontière de décision. Autrement dit, un exemple n'est pris en compte que s'il n'est pas assez bien classé dans l'espace ciblé selon l'espace privilégié. Cette méthode est néanmoins basée sur un classifieur SVM binaire.

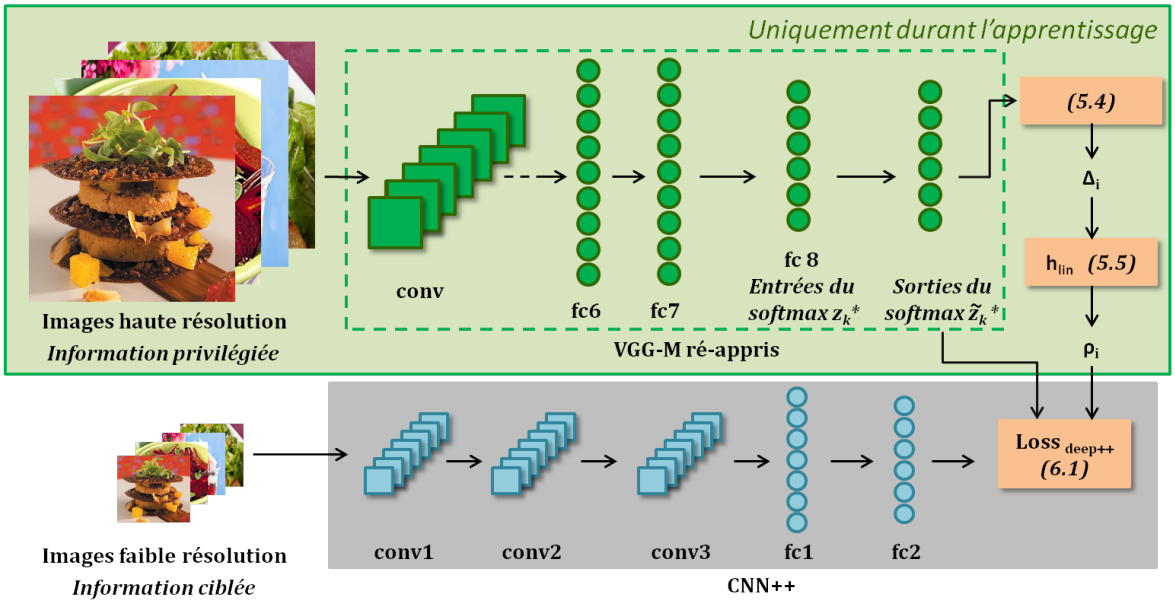



FIGURE 6.1 – Architecture de notre modèle deep++. Pour chaque exemple, le CNN appris sur les données privilégiées (cadre vert) permet de calculer un coefficient $\rho_i$, caractérisant sa difficulté, et le score $\tilde{z}_c(x_i^*)$ pour sa vraie classe. Ces deux informations sont intégrées dans la fonction de coût $loss_{deep++}$ d'un réseau appris sur les données ciblées (cadre gris). On reporte entre parenthèses les numéros des équations décrivant chaque étape.

## 6.2 Fonction de coût

Notre modèle deep++ s'appuie sur l'information privilégiée pour calculer deux termes de difficulté intervenant dans le critère d'optimisation du CNN dans l'espace ciblé. Le coefficient $\rho_i$, détaillé dans la section 5.2.1, reflète une difficulté absolue donnée par l'espace privilégié. Un deuxième terme de difficulté relative entre les deux espaces permet de pénaliser davantage un exemple tant qu'il est *trop* mal classé. Cette pénalisation supplémentaire doit cependant s'amenuiser lorsque l'exemple devient assez bien classé, *i.e.* quand son score cible pour sa vraie classe est assez élevé. Concrètement, pour chaque exemple, on cherche à rapprocher au maximum le score du CNN cible de celui du CNN privilégié. Ce nouveau terme peut donc être considéré comme une difficulté relative par rapport à l'espace privilégié, qui, de fait, varie au cours de l'apprentissage. La fonction de coût s'écrit ainsi :

$$loss_{deep++} = \sum_{i=1}^{N} -\Big(\rho_i + \gamma[\tilde{z}_c^*(x_i^*) - \tilde{z}_c(x_i)]_+\Big) \ln\left(\tilde{z}_c(x_i)\right) \tag{6.1}$$

où $\tilde{z}_c^*(x_i^*)$ (resp. $\tilde{z}_c(x_i)$) est le score en sortie du softmax de la classe vérité terrain dans l'espace privilégié (resp. ciblé), et $[.]_+$ est la fonction qui à tout réel $x$ associe $\max(x, 0)$. $\gamma$ est un hyper-paramètre régulant le compromis entre les deux termes : pour un $\gamma$ élevé, le coût relatif basé sur le gap de performances entre les deux espaces aura un impact plus important que le terme de difficulté absolue $\rho_i$.

La formulation du gradient s'écrit alors :

$$\frac{\partial loss_{deep++}}{\partial \tilde{z}_k(x_i)} = -\frac{\rho_i}{\tilde{z}_c(x_i)}\delta_{k=c} + \gamma\Big[\ln(\tilde{z}_c(x_i)) - \frac{\tilde{z}^*_c(x^*_i) - \tilde{z}_c(x_i)}{\tilde{z}_c(x_i)}\Big]\delta_{\tilde{z}^*_c(x^*_i)>\tilde{z}_c(x_i)}\delta_{k=c}. \quad (6.2)$$

Ce modèle intègre le nouveau terme $[\tilde{z}^*_c(x^*_i) - \tilde{z}_c(x_i)]_+$ par rapport à la formulation 5.8. Ce terme complémentaire est non nul uniquement lorsque $\tilde{z}_c(x_i) \leq \tilde{z}^*_c(x^*_i)$, *i.e.* lorsque l'exemple $i$ est moins bien reconnu dans l'espace cible que dans l'espace privilégié. Ce terme évolue aussi proportionnellement à l'écart entre ces deux scores. Ainsi, pour un exemple $i$ donné, plus le score dans l'espace ciblé est faible par rapport au score de l'espace privilégié, plus l'exemple aura d'influence sur l'apprentissage des poids du réseau. On rejoint ainsi l'intuition de SVM+, qui pénalise fortement les exemples mal classés par rapport à une marge, elle-même donnée par l'espace privilégié.

Le modèle proposé est également proche du modèle LIR, introduit par (Wang and Ji, 2015). Dans cette approche, le modèle appris dans l'espace ciblé est pénalisé si le coût associé à un exemple donné est inférieur à son coût dans l'espace privilégié. Autrement dit, on pénalise le modèle dans l'espace ciblé dès lors qu'il classifie *mieux* que dans l'espace privilégié. Le score dans l'espace privilégié est donc une limite au-delà de laquelle on considère que le modèle dans l'espace ciblé sur-apprend. Cette approche est assez semblable à notre modèle deep++ dans sa formulation, mais part d'une intuition tout à fait opposée, puisque notre méthode vise à forcer les scores dans l'espace ciblé à être meilleurs que dans l'espace privilégié.

**Algorithm 2** Apprentissage deep++

**Require:** $(x^*_i, x_i, y_i)_{i=1..N}$
1: Apprentissage d'un CNN sur les images privilégiées
2: Calcul des $\rho_i^{CNN}$ (5.4, 5.5)
3: Initialisation des poids du réseau dans l'espace ciblé $\mathbf{w}$
4: **for** époque $= 1..\text{epoque}_{\text{max}}$ **do**
5: Calcul de $loss_{deep++}$ (6.1)
6: Mise à jour des poids $\mathbf{w}$ par rétro-propagation du gradient de $loss_{deep++}$ (6.2)
7: **end for**

## 6.3 Résultats expérimentaux

### 6.3.1 Description des bases de données

Le modèle deep++ proposé dans ce chapitre est une extension de notre modèle deep+, proposé dans le chapitre 5. Ainsi, on s'intéresse naturellement à comparer ces deux méthodes dans des contextes similaires, et donc sur les mêmes bases de données. Comme pour le modèle deep+, on s'intéresse donc aux mêmes bases orientées grain-fin FGVC-Aircraft (Maji et al., 2013) et UPMC-Food-101 (Wang et al., 2015b), détaillées respectivement dans les sections 3.1.1 et 3.3.2, ainsi qu'à MNIST (LeCun et al., 1998), décrite dans la partie 5.3.1.

Sur ces bases de données, on s'intéresse au même type d'images ciblées et privilégiées que dans la section 5.3. De même, sur MNIST, on s'intéresse toujours au problème de classification à 10 classes.

### 6.3.2 Protocole expérimental et paramétrage

Dans cette partie, on veut comparer notre modèle deep++ par rapport à plusieurs méthodes de l'état de l'art LUPI. Pour ce faire, on présente les performances des représentations ciblées seules, celles des méthodes de l'état de l'art LUPI, puis celles de nos deux modèles deep+ et deep++. Pour les représentations ciblées seules, on reprend le même protocole que dans la partie 5.3.

**Méthodes de l'état de l'art.** Pour les méthodes basées sur une pondération des exemples Margin Transfer et SVM+, basées sur des représentations classées par un SVM, on reprend le même protocole que dans la partie 5.3.3, résumé dans le tableau 5.1.

La méthode concurrente Generalized Distillation est en revanche basée sur deux réseaux de neurones, appris respectivement dans l'espace privilégié et ciblé. Pour cette méthode, sur chaque base de données, dans chacun des espaces (privilégié et ciblé), on utilise le même réseau que deep++, *i.e.* les réseaux spécifiés dans le tableau 5.2. On prend le paramétrage standard utilisé dans les travaux de (Lopez-Paz et al., 2016), *i.e.* $T = 1$ et $\lambda = 0,7$.

**Deep++.** Pour notre modèle deep++, également basé sur des structures profondes dans les deux espaces, on se place dans les mêmes conditions d'apprentissage que pour deep+ (section 5.3.2 et tableau 5.2). En particulier, pour chaque base de données et dans chaque espace, on utilise le même réseau avec les mêmes hyper-paramètres d'apprentissage que pour deep+. De plus, on utilise les mêmes coefficients $\rho_i^{CNN}$ pour deep+ et deep++. La valeur de $\gamma$ est adaptée sur chaque base de données.

### 6.3.3 Résultats

Dans cette première partie, on compare notre modèle à plusieurs méthodes de l'état de l'art. D'une part, on reporte les résultats des méthodes déjà explorées SVM+ (Vapnik and Vashist, 2009) et Margin Transfer (Sharmanska et al., 2014), plutôt axées sur une pondération des exemples dans l'espace ciblé. D'autre part, on s'intéresse à la méthode Generalized Distillation de Lopez-Paz et al. (2016) (décrite dans la section 4.2.4), basée explicitement sur un coût de ressemblance entre les modèles appris dans chacun des deux espaces.

Les performances des différentes méthodes sont présentées dans le tableau 6.1. Sur MNIST, notre modèle deep++ parvient à faire 24 erreurs de moins que Margin Transfer, et 15 erreurs de moins que la méthode Generalized Distillation. On montre donc que notre approche de prise en compte de la difficulté des exemples est particulièrement plus efficace que les méthodes de l'état de l'art testées dans ce contexte. De plus, sur cette base de données, la méthode LIR (Wang and Ji, 2015) commet 120 erreurs, soit 25 erreurs de plus que notre modèle. Cette méthode s'appuie sur une formulation similaire à celle de deep++ puisqu'elle compare les coûts d'un exemple dans les deux espaces, mais est cependant basée sur une intuition opposée : l'exemple doit être mieux classé dans l'espace privilégié que dans l'espace ciblé. Au-travers de cette expérience, on montre donc que notre approche est plus adaptée au problème dans ce contexte.

En comparant notre méthode aux modèles de l'état de l'art sur UPMC-Food-101, on montre aussi que notre deep++ parvient à améliorer de 1,3% les performances de

| Méthode | MNIST | UPMC-Food-101 | FGVC-Aircraft |
|---|---|---|---|
| Représentation ciblée | 130 erreurs | 28,9% | 32,7% |
| SVM+ | 122 erreurs | - | 31,2% |
| Margin Transfer | 119 erreurs | 30,6% | 33,5% |
| Generalized Distillation | 110 erreurs | 31,4% | 38,9% |
| Deep+ | 107 erreurs | 31,6% | 39,8% |
| **Deep++** | **95 erreurs** | **31,9%** | **39,8%** |

TABLE 6.1 – Comparaison de notre méthode deep++ aux méthodes de l'état de l'art LUPI. Nos modèles deep+ et deep++ ainsi que l'approche Generalized Distillation (basés sur des modèles profonds) sont appris via le protocole décrit dans le tableau 5.2. Le protocole pour les autres méthodes (qui sont basées sur des SVM) est décrit par le tableau 5.1.

Margin Transfer. De plus, les performances de notre modèle sont assez proches de celles de Generalized Distillation (+0,5%). Cette observation est cohérente avec le fait que notre modèle comme celui de Generalized Distillation repose en partie sur une condition de ressemblance entre les sorties des réseaux dans chacun des deux espaces. Les approches sont donc similaires, ce qui peut expliquer la faible différence de performances entre ces modèles.

Notre méthode nous permet également d'obtenir de meilleurs résultats que les méthodes de l'état de l'art LUPI sur FGVC-Aircraft. En particulier, notre modèle permet d'obtenir des performances 0,9% meilleures que Generalized Distillation, bien que les deux méthodes soient basées sur des réseaux de neurones profonds dans l'espace privilégié aussi bien que dans l'espace ciblé.

#### Bilan

Dans cette partie, on montre que notre modèle deep++ permet d'améliorer les performances par rapport aux méthodes de l'état de l'art LUPI testées sur toutes les bases de données. Plus particulièrement, deep++ obtient de meilleurs résultats que les deux méthodes par pondération, basées sur des approches SVM, *i.e.* SVM+ et Margin Transfer. De plus, notre deep++ obtient des performances supérieures à celles de Generalized Distillation, voire assez proches dans certains cas. En effet, les approches de cette technique et de notre modèle deep++ sont assez similaires, puisqu'il s'agit de forcer l'espace privilégié à copier en partie le comportement du réseau de neurones dans l'espace privilégié.

### 6.3.4 Analyse de notre modèle deep++

Dans cette partie, on évalue l'amélioration de notre modèle deep++ par rapport aux méthodes sur lesquelles il repose. Ainsi, on analyse les performances de notre modèle deep++ comparées à celles de notre modèle deep+ (chapitre 5) et du CNN correspondant sans information privilégiée. On se place dans les mêmes conditions d'apprentissage pour les trois méthodes deep, deep+ et deep++ (voir tableau 5.2), *i.e.* les réseaux et hyper-paramètres communs aux trois expériences sont les mêmes.

| Méthode | MNIST | UPMC-Food-101 | FGVC-Aircraft |
|---|---|---|---|
| Deep | 130 erreurs | 30,7% | 39,8% |
| Deep+ | 107 erreurs | 31,6% | 39,8% |
| Deep++ | 95 erreurs | 31,9% | 39,8% |

TABLE 6.2 – Apport de notre méthode deep++ par rapport à notre méthode deep+ ou un réseau sans information privilégiée. Le protocole utilisé pour toutes ces méthodes est décrit dans le tableau 5.2.

**Résultats**

On présente les résultats des différentes méthodes dans le tableau 6.2. Sur MNIST, notre étude montre que notre modèle deep++ parvient à faire 12 erreurs de moins que notre précédent modèle deep+, et 35 de moins que le réseau de neurones sans information privilégiée. Sur cette base, on montre bien que notre modèle parvient à améliorer significativement les performances des modèles précédents.

De même, sur UPMC-Food-101, on montre que notre modèle deep++ parvient à améliorer quelque peu les performances du modèle deep+. En effet, le modèle deep+ parvient déjà à améliorer les résultats. On peut donc penser que les $\rho_i^{CNN}$ sont parvenus à filtrer la plupart des aberrations contenues dans la base de données.

Enfin, sur FGVC-Aircraft, on constate que notre modèle deep++ obtient les mêmes résultats que notre modèle précédent deep+. Sur cette base de données, le réglage de l'hyper-paramètre $\gamma$ pondérant les deux termes de la fonction de coût (eq. 6.1) révèle que de meilleures performances sont atteintes lorsqu'on annule le coût relatif, ce qui semble démontrer une certaine faiblesse du CNN dans l'espace privilégié. Cela peut s'expliquer par le fait que le CNN adapté dans l'espace privilégié est appris sur un faible nombre de données (moins de 7 000) alors qu'on cherche à discriminer un grand nombre de classes (100 classes orientées grain-fin), et qu'il a donc un risque de sur-apprentissage. En effet, le réseau VGG-M adapté sur les données FGVC-Aircraft atteint 52,3% de bonne classification sur les données de test, contre 56,3% pour les représentations pré-entraînées VGG-M_fc1, ce qui montre un léger sur-apprentissage. Par ailleurs, le fait que les performances de Generalized Distillation (38,9% dans le tableau 6.1) soient légèrement inférieures à celles du LR-CNN dans l'espace ciblé sans information privilégiée (39,8% dans le tableau 6.2) semble appuyer cette hypothèse.

**Bilan**

Dans cette partie, on a montré l'intérêt de notre modèle deep++ dans plusieurs contextes de classification d'images. Sur MNIST et UPMC-Food-101, qui contiennent un nombre suffisant de données d'apprentissage pour apprendre les réseaux de neurones dans l'espace ciblé, on montre bien que notre modèle permet d'améliorer les performances de classification.

Les expériences menées sur FGVC-Aircraft, qui contient assez peu d'exemples d'apprentissage, tendent à montrer que le manque de données peut affecter sensiblement les résultats. Par ailleurs, le réglage de l'hyper-paramètre $\gamma$ (eq. 6.1) montre que le coût re-

| | Images $32 \times 32$ | | Images $64 \times 64$ | |
|---|---|---|---|---|
| | sans FA | avec FA | sans FA | avec FA |
| CNN | 95,91% | 96,32% | 98,50% | 99,02% |
| CNN+ | 96,00% | 96,07% | 98,65% | 99,03% |
| CNN++ | 96,60% | 96,49% | 98,95% | 99,08% |
| Generalized Distillation | 96,18% | 96,29% | 98,98% | 99,14% |

TABLE 6.3 – Apport de notre méthode deep++ sur les images simulées Thales.

latif a une grande importance sur MNIST, où le nombre de données est assez large par rapport au nombre de paramètres à apprendre, alors que ce compromis est plutôt en faveur du coût deep+ sur les deux bases orientées grain-fin, sur lesquelles on apprend un grand nombre de poids. Cela semble abonder dans le sens de l'hypothèse ci-dessus. On peut donc subodorer l'intérêt des données simulées pour obtenir des bases plus grandes afin de mieux apprendre le réseau dans l'espace privilégié et donc atteindre de meilleures performances via notre modèle deep++. Dans la suite, on s'intéresse justement à l'apport de notre méthode dans le cadre des données simulées Thales.

### 6.3.5 Deep++ sur les données Thales

Dans cette partie, on évalue performances de notre modèle deep++ sur la base de données Thales. On se place dans les mêmes conditions que dans la partie 5.3.5. Comme pour la partie 6.3.4, on s'intéresse d'une part aux performances de notre modèle deep++ comparées à celles du modèle deep+ proposé dans le chapitre 5 ainsi qu'aux performances d'un CNN sans information privilégiée. D'autre part, on compare notre méthode au modèle Generalized Distillation (Lopez-Paz et al., 2016), décrit dans la partie 4.2.4.

#### Protocole expérimental et paramétrage

Pour notre modèle deep++, on utilise les mêmes $\rho_i^{CNN}$ que ceux calculés dans la partie 5.3.5, *i.e.* en prenant les mêmes hyper-paramètres $\tau_{min} = 0,1$ et $\tau_{max} = 2$. On prend une valeur de $\gamma = 1$. Comme stipulé dans la partie 5.3.5, la base de données ciblée peut contenir des fausses alarmes (*i.e.* des images ne contenant pas de véhicule), qui ne sont associée à aucune information privilégiée. Dans ce cas, les poids $\rho_i$ sont les mêmes que dans la partie 5.3.5, et on considère que les scores privilégiés associés $\tilde{z}_c^*(x_i^*)$ sont nuls. Cela correspond bien au fait que l'information est indisponible ; on n'ajoute alors aucune pénalisation supplémentaire par rapport au coût de classification pondéré.

Pour l'implémentation du modèle concurrent Generalized Distillation, on utilise le même réseau VGG-M adapté dans l'espace privilégié que pour nos modèles deep+ et deep++. Les hyper-paramètres sont réglés comme prescrit dans l'article de Lopez-Paz et al. (2016), *i.e.* $\lambda = 0,7$ et $T = 1$. Les hyper-paramètres d'apprentissage sont les mêmes pour tous les modèles testés, et sont identiques à ceux utilisés dans la partie 5.3.5. Lorsque des images de fausses alarmes sont présentes, on considère que la sortie du softmax du réseau dans l'espace privilégiée est parfaite, *i.e.* la classe de fausses alarmes a un score de 1, et les scores de toutes les autres classes sont à 0.

| | Images $32 \times 32$ | | Images $64 \times 64$ | |
|---|---|---|---|---|
| | sans FA | avec FA | sans FA | avec FA |
| CNN | 95,91% | 96,32% | 98,50% | 99,02% |
| CNN+ | 96,00% | 96,07% | 98,65% | 99,03% |
| CNN++, $\rho_i = 1$ | 96,23% | 95,79% | 98,88% | 99,12% |
| CNN++, $\rho_i = 0$ | 95,60% | 95,72% | 98,28% | 98,70% |
| CNN++ | 96,60% | 96,49% | 98,95% | 99,08% |

TABLE 6.4 – Étude des performances du coût relatif seul sur les images simulées Thales.

**Résultats**

On reporte les résultats des différentes expériences dans le tableau 6.3. Dans ce tableau, on montre que dans tous les cas de figure, notre modèle deep++ permet non seulement d'améliorer les performances par rapport au CNN standard sans information privilégiée, mais elle permet en plus de mieux classer les images de test que le CNN+. Cela montre que le terme de coût relatif ajouté par rapport à la formulation deep+ permet d'apprendre un meilleur modèle.

De plus, ce gain de performances se généralise également dans le cas où des fausses alarmes sont présentes dans l'ensemble de test. Dans ces configurations, malgré la présence de fausses alarmes pour lesquelles aucune information privilégiée n'est disponible, notre méthode deep++ parvient à améliorer les performances malgré la présence d'exemples n'ayant pas d'information privilégiée associée. Notre modèle permet donc d'intégrer facilement et efficacement cette difficulté sur cette base de données.

Par ailleurs, en comparant notre modèle à la méthode de l'état de l'art Generalized Distillation, on montre que notre deep++ atteint de meilleures performances sur les données $32 \times 32$, et est équivalent dans le cas des images $64 \times 64$. Ces deux approches sont assez similaires, dans le sens où, dans les deux cas, on impose une ressemblance entre les scores des réseaux des deux espaces. On remarquera cependant que la présence de fausses alarmes de taille $32 \times 32$ perturbe quelque peu les résultats de Generalized Distillation, puisque les performances sont alors très légèrement inférieures à celles du CNN sans information privilégiée, ce qui n'est pas le cas de notre modèle.

Dans le tableau 6.4, on s'intéresse également aux performances deep++ en n'utilisant que le terme de coût relatif, *i.e.* en fixant tous les $\rho_i$ à une valeur donnée. Ici, on a pris $\gamma = 1$ pour toutes les expériences avec $\rho_i = 1$. Dans ce tableau, on montre que dans les deux cas qui ne contiennent pas de fausses alarmes, la formulation prenant en compte le coût relatif ainsi que le coût de classification standard (cas des $\rho_i = 1$) permet d'améliorer les performances par rapport au CNN sans information privilégiée. Cela montre que notre approche semble pertinente dans le cas de la reconnaissance d'images sans fausses alarmes. D'autre part, lorsqu'on considère que tous les $\rho_i$ sont à 0, on n'a plus que le coût de classification relatif entre les deux espaces. Dans ce cas, on remarque que le réseau deep++ tend à afficher de moins bonnes performances que le réseau sans information privilégiée, mais qui restent cependant relativement élevées. Malgré le fait que le coût relatif proposé ne se suffise pas à lui seul pour la reconnaissance de ces images, il est néanmoins porteur d'une information intéressante pour cette tâche.

| Méthode de classification | MNIST | UPMC-Food-101 | FGVC-Aircraft |
|---|---|---|---|
| Deep | 130 erreurs | 30,7% | 39,8% |
| Deep+ | 107 erreurs | 31,6% | 39,8% |
| Deep++, $\rho_i = 1$ | 109 erreurs | 30,9% | 40,0% |
| Deep++, $\rho_i = 0$ | 135 erreurs | 31,1% | 37,1% |
| Deep++ | 95 erreurs | 31,9% | 39,8% |

TABLE 6.5 – Étude des performances de classification avec le coût relatif seul sur les bases de données publiques.

### 6.3.6 Étude approfondie de notre modèle deep++

#### Coût relatif seul

On s'intéresse maintenant au gain de performances apporté par le terme de coût relatif seul. Autrement dit, sur chaque base de données, on fixe les $\rho_i$ de tous les exemples d'apprentissage à une valeur donnée, ce qui permet de mettre en avant l'apport du coût supplémentaire introduit dans ce nouveau modèle. On s'intéresse à deux cas de figure en particulier : d'une part, on fixe tous les $\rho_i = 0$. Ainsi, on force seulement les exemples dans l'espace ciblé à être classés aussi bien que dans l'espace privilégié. Notamment, pour les exemples mal classés par le réseau dans l'espace privilégié, le réseau dans l'espace ciblé n'est plus pénalisé à partir du moment où il attribue un score équivalent au score privilégié. D'autre part, on se penche sur le cas où tous les $\rho_i = 1$. Dans cette expérience, on force tous les exemples à être correctement reconnus dans l'espace ciblé, en plus de pénaliser lorsque le score dans l'espace ciblé est trop faible par rapport au score dans l'espace privilégié. On reporte les résultats sur les différentes bases dans le tableau 6.5.

Dans ce tableau, on montre que lorsqu'on force le réseau dans l'espace ciblé à la fois à bien classer les exemples et à dépasser les scores du réseau dans l'espace privilégié ($\rho_i = 1$), on arrive dans tous les cas à améliorer légèrement les résultats par rapport à un CNN sans information privilégiée. Par exemple, on gagne 0,2% de bonne classification sur les deux bases de données axées grain-fin. Cette observation montre que le terme de coût relatif proposé dans notre modèle deep++ couplé au terme de classification standard permet d'améliorer quelque peu les performances de classification.

D'autre part, lorsqu'on prend $\rho_i = 0$, on ne force plus les exemples à être bien classés dans l'espace ciblé, mais seulement à avoir un score au moins aussi bon que celui de sa représentation privilégiée. Dans ce cas, sur UPMC-Food-101, cette approche permet d'améliorer les performances par rapport à un CNN standard. En effet, comme cette base contient un certain niveau de bruit d'annotation, il est possible que des exemples mal annotés perturbent la fonction apprise par un CNN sans information privilégiée. En revanche, si un CNN est correctement appris dans l'espace privilégié, pour un exemple $x_i$ de classe $c$ mal annoté, son score $\tilde{z}_c^*(x_i^*)$ devrait être assez faible, ce qui se traduit par une faible contrainte sur le score $z_c(x_i)$ dans l'espace ciblé (eq. 6.1). Ce protocole permet donc de relâcher la contrainte de classification sur les aberrations, et dans le cas de UPMC-Food-101, qui contient du bruit d'annotation, cela peut expliquer le gain de performances.

En revanche, le coût relatif seul diminue les performances de classification sur FGVC-

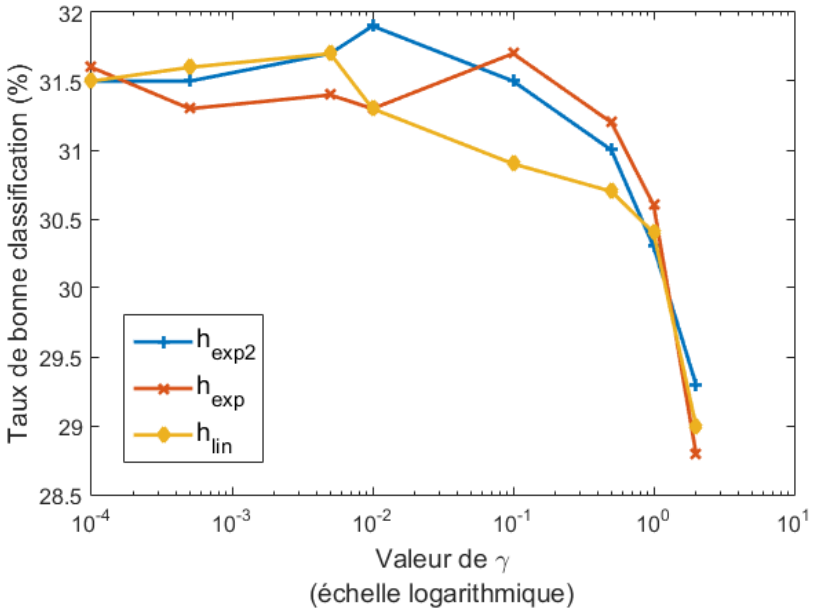



FIGURE 6.2 – Étude de l'impact de $\gamma$ sur les performances de notre modèle deep++ sur UPMC-Food-101.

Aircraft. Cette base de données ne contient qu'un nombre limité d'images d'apprentissage, ce qui rend difficile l'adaptation d'un large CNN dans l'espace privilégié. En effet, si le nombre d'images privilégiées est trop faible par rapport au nombre de poids appris, le CNN privilégié peut ne pas être suffisamment performant pour que la contrainte de ressemblance entre les deux espaces se suffise à elle seule. De même, sur MNIST, le coût relatif seul ne permet pas de mieux reconnaître les exemples de test que le LeNet entraîné sans information privilégiée. Cela tend à prouver que, dans les cas sans aberration, l'information de coût relatif seule ne suffit pas à améliorer les performances du réseau sans information privilégiée. Cependant, les performances restent tout de même assez proches de celles des réseaux sans information privilégiée, ce qui montre que ce coût relatif est porteur d'une information intéressante pour l'apprentissage du réseau.

Pour conclure, on a montré que le terme de coût relatif proposé dans notre modèle deep++ était cohérent avec la nature des différentes bases de données testées. De plus, notre terme relatif couplé avec la pénalisation standard de classification sans information privilégiée permet dans tous les cas d'améliorer les performances par rapport à un CNN sans information privilégiée, ce qui montre la légitimité de notre approche.

#### Équilibre entre les deux termes de pénalisation

On s'intéresse ici à l'impact de la valeur de $\gamma$ sur les performances d'un réseau deep++. Pour ce faire, on apprend huit réseaux deep++ indépendants, chacun avec une valeur de $\gamma$ donnée. On répète cette opération pour les trois types de remappages proposés dans la section 5.2.1. Dans toutes ces expériences, on prend $\tau_{min} = 0,1$ et $\tau_{max} = 1$, *i.e.* pour chacun des remappages, les $\rho_i$ sont les mêmes quelle que soit la valeur de $\gamma$. On reporte les résultats sur la figure 6.2.

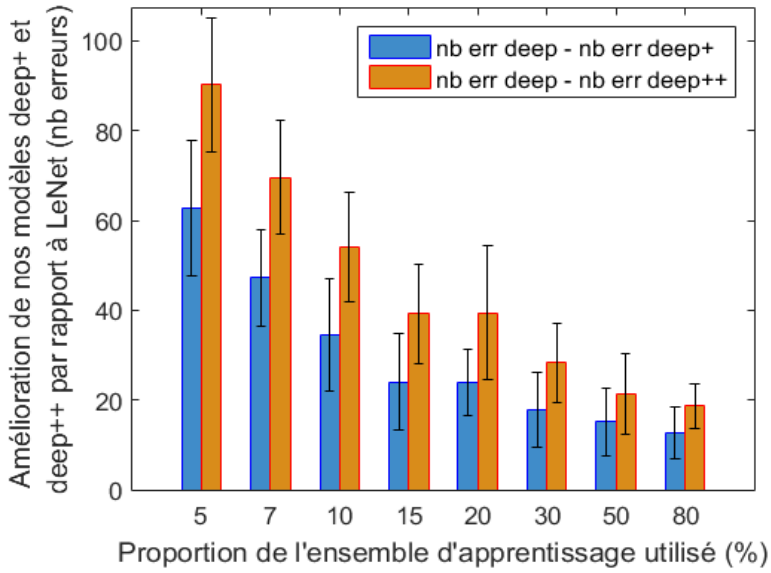



FIGURE 6.3 – Amélioration de nos modèles deep+ (en bleu) et deep++ (en orange) en fonction de la proportion d'images d'apprentissage utilisées.

Sur cette figure, deux phases majeures se dégagent. D'abord, pour des valeurs de $\gamma$ assez faibles, les performances sont relativement stables et parviennent à un pic lorsque la valeur de $\gamma$ augmente. Ensuite, quand $\gamma$ devient trop élevé, les performances chutent. Ce comportement tend à montrer que lorsque le coût relatif prend trop d'importance par rapport au coût standard de classification, le réseau parvient moins bien à généraliser. Cette observation est cohérente avec l'analyse précédente sur le coût relatif seul, montrant que le coût relatif est plus pertinent lorsqu'il est couplé à un coût de classification standard. Le maximum des performances est donc atteint lorsqu'un équilibre entre les deux termes de classification est respecté.

**Influence du nombre d'images d'apprentissage**

On s'intéresse également à l'influence du nombre d'images d'apprentissage sur les performances de notre modèle. En suivant le même protocole que dans la partie 5.3.6, on reporte sur la figure 6.3 l'amélioration de notre modèle deep++ par rapport au réseau sans information privilégiée en fonction du nombre d'images d'apprentissage. On reporte également pour comparaison l'amélioration de notre modèle deep+ par rapport au réseau sans information privilégiée, explorée dans la figure 5.9.

Sur ce graphique, on montre que notre modèle deep++ permet d'améliorer les performances du réseau sans information privilégiée quelle que soit la proportion d'images d'apprentissage. Par exemple, avec 20% des exemples d'apprentissage, deep++ commet 39,4 erreurs de moins que LeNet en moyenne. De plus, on montre que pour toutes les proportions, deep++ est systématiquement meilleur que deep+. Par exemple, avec 30 % des données d'apprentissage, deep++ améliore de 17,8 les performances de LeNet, alors que deep++ améliore de 28,3 le nombre d'erreurs de LeNet.

Le graphique montre par ailleurs que l'écart de performances de deep++ par rapport à celles de LeNet ou même de deep+ se creuse à mesure que le nombre d'images d'ap-

prentissage diminue. En effet, avec 50% des données, deep++ améliore de 21,3 erreurs les performances de LeNet, ce qui est relativement proche de l'amélioration de deep+ par rapport à LeNet (15,2 erreurs). En revanche, avec seulement 7% des images d'apprentissage, deep++ améliore de 69,6 le nombre d'erreurs commises par LeNet, là où deep+ les améliorait de 47,2. Cette observation est cohérente avec les intuitions et conclusions de Vapnik and Vashist (2009) selon lesquelles l'information privilégiée se révèle d'autant plus informative que le nombre d'exemples d'apprentissage est faible. De plus, elle corrobore l'intérêt de nos approches pour apprendre un modèle robuste avec peu de données d'apprentissage, déjà évoqué dans la partie 5.3.6, et ouvre des pistes intéressantes sur l'apprentissage des réseaux de neurones, qui requiert habituellement beaucoup d'exemples annotés.

## 6.4 Conclusion

Dans ce chapitre, nous avons proposé deep++, un algorithme d'intégration de l'information privilégiée dans les réseaux de neurones profonds. Ce modèle est une extension de notre précédente approche deep+. Notre modèle précédent ne prend en compte qu'une pondération des exemples en fonction de leur difficulté dans l'espace privilégié. Le coefficient de difficulté associé à chaque exemple peut donc être vu comme une difficulté *absolue* dans l'espace privilégié. Dans notre nouveau modèle deep++, on propose un terme complémentaire, mesurant l'éloignement entre le score dans l'espace ciblé et celui dans l'espace privilégié. En ce sens, on a donc introduit un terme mesurant une difficulté *relative* entre les deux espaces.

Expérimentalement, notre méthode a montré de bons résultats sur plusieurs bases de données publiques, notamment dans un contexte grain-fin contenant du bruit d'annotation. De plus, on a montré l'intérêt de notre méthode dans un contexte plus applicatif, sur une base de données simulées propre à Thales, y compris en présence de fausses alarmes, qui sont des données sur lesquelles on ne dispose pas d'information privilégiée. On a en outre montré que ce modèle permettait de traiter de larges bases de données, et qu'il obtenait toujours des résultats meilleurs ou équivalents aux méthodes de l'état testées. En proposant une analyse fine de notre modèle, on montre également que le terme de coût relatif est bien complémentaire du coût pondéré de deep+. On confirme également l'intérêt de notre modèle lorsqu'on dispose de peu d'exemples d'apprentissage, ce qui offre des perspectives intéressantes pour l'apprentissage de réseaux de neurones, qui s'appuie généralement sur un grand nombre de données annotées.

# Chapitre 7

# Conclusions et perspectives

**Sommaire**



## 7.1 Résumé des contributions

Dans le cadre de cette thèse, on a exploré le contexte académique de la classification d'images avec plusieurs des contraintes opérationnelles spécifiques à Thales Optronique pour la reconnaissance air-sol de véhicules lointains. On s'inscrit ainsi dans un contexte de développement rapide des méthodes d'apprentissage pour la reconnaissance d'images, en particulier avec les réseaux de neurones convolutionnels profonds qui se sont récemment imposés comme la méthode majeure de l'état de l'art pour toutes les tâches de reconnaissance visuelle. Le contexte opérationnel de ces travaux nous a amenés à aborder le problème de la reconnaissance d'images faiblement résolues. Dans le cadre méthodologique de l'apprentissage avec information privilégiée, on s'est également intéressé à l'intégration d'information complémentaire, les versions haute-résolution des images d'apprentissage, durant la phase d'entraînement.

Dans ces travaux, on a d'abord étudié l'adéquation des méthodes classiques de l'état de l'art en représentation des images dans le cadre de la reconnaissance d'images de faible résolution. Plus précisément, on a évalué la robustesse de représentations profondes pré-entraînées ainsi que de représentations Fisher Vectors face à la perte de résolution d'images de plusieurs bases de données orientées grain-fin. En particulier, les représentations profondes émanent d'un réseau dont les poids sont appris sur une large base de données externe : ce système repose donc sur le transfert des connaissances acquises sur une base extérieure vers la base de données d'intérêt. Sur cette tâche, on a montré l'intérêt des réseaux de neurones, notamment pour leur aspect compact. Le transfert semble néanmoins poser problème pour les résolutions assez faibles, trop différentes des images sur lesquelles les paramètres ont été appris. Afin d'aborder ce problème, nous avons proposé un réseau convolutionnel profond contenant assez peu de paramètres, de sorte à ce qu'il soit appris directement sur les données ciblées. Cette architecture a permis d'améliorer

les performances des représentations profondes pré-entraînées sur deux bases orientées grain-fin.

Un deuxième aspect de ces travaux a été de tirer avantage de la mise à disposition d'images haute-résolution complémentaires durant l'apprentissage du système de reconnaissance. Dans ce but, nous avons proposé deux méthodes basées sur l'approche de l'information privilégiée, permettant d'intégrer ces données additionnelles dans l'apprentissage d'un réseau de neurones convolutionnel profond. Notre premier modèle deep+ s'appuie sur l'information privilégiée afin de calculer un coefficient $\rho_i$ pour chaque exemple d'entraînement, pondérant l'influence des exemples durant l'apprentissage du réseau de neurones dans l'espace ciblé. Ces coefficients reflètent la difficulté des images d'apprentissage, et n'est basé que sur les scores calculés dans l'espace des données complémentaires. L'intuition derrière ce système de pondération est d'attribuer un poids plus important aux exemples les plus faciles : cette approche permet de focaliser l'apprentissage sur des exemples représentatifs de leur classe, et de réduire les poids des exemples les plus difficiles, *i.e.* des aberrations potentielles de la base de données. Expérimentalement, on a montré l'efficacité de notre approche par rapport aux méthodes de l'état de l'art LUPI Margin Transfer (Sharmanska et al., 2014) et SVM+ (Vapnik and Vashist, 2009), basées sur des classifieurs moins profonds, comme les SVM. Notre méthode permet d'améliorer les performances des approches de l'état de l'art dans plusieurs contextes de classification, et notamment sur une base de données contenant du bruit d'annotation. De plus, notre modèle exploitant l'information privilégiée permet également d'améliorer les performances d'un CNN standard, notamment dans un cadre expérimental proposé par Vapnik and Vashist (2009) sur MNIST. Par ailleurs, on a proposé une évaluation étendue de notre modèle, explorant les différents aspects de notre méthode. On a montré en particulier que notre deep+ permet d'améliorer fortement les performances d'un réseau profond avec très peu d'images d'apprentissage, ce qui offre des perspectives intéressantes à l'heure des réseaux de neurones profonds requérant un grand nombre d'images durant l'apprentissage.

Cette première méthode est basée sur une notion de difficulté absolue, qui ne fait intervenir que l'espace privilégié. Pour affiner la prise en compte des données complémentaires, nous avons proposé deep++, un deuxième modèle d'intégration de l'information privilégiée dans l'apprentissage d'un réseau de neurones convolutionnel profond. Ce modèle étend la méthode deep+ en intégrant une pénalisation supplémentaire, traduisant une notion de difficulté relative des exemples entre les deux espaces. L'intuition portée par cette pénalisation est de forcer les exemples à être *assez* bien classés, *i.e.* avec un score minimal, dicté par l'information privilégiée. On montre expérimentalement l'intérêt de notre modèle deep++ sur plusieurs tâches de classification d'images, atteignant des performances supérieures à celles des méthodes de l'état de l'art LUPI testées sur toutes les tâches explorées. En particulier, la comparaison de notre méthode à l'approche Generalized Distillation de Lopez-Paz et al. (2016), basée sur un coût de ressemblance entre des réseaux de neurones profonds appris dans chacun des deux espaces, montre la pertinence de notre méthode de prise en compte de l'information privilégiée dans les CNN. Dans un contexte plus opérationnel, notre deep++ permet d'améliorer à la fois les résultats de deep+ et les performances des méthodes de l'état de l'art concurrentes sur une base de données propre à Thales. Sur cette base, on montre notamment que notre modèle parvient à améliorer les performances d'un CNN sans information privilégiée dans toutes les configurations de la base de données, y compris dans les cas où certains exemples ne

sont associés à aucune information privilégiée. Par ailleurs, on confirme l'intérêt de notre approche dans des contextes où on ne dispose que de peu d'images d'apprentissage.

## 7.2 Perspectives

À la suite à ces travaux, plusieurs pistes seraient intéressantes à explorer dans des projets de plus ou moins long terme.

**Application aux données Thales réelles.** On s'est intéressé à une base de données simulées propre à Thales, afin d'illustrer l'intérêt de nos modèles dans un cadre opérationnel ainsi que pour pallier le problème du nombre limité d'images réelles. Bien que reflétant plusieurs aspects des bases d'images acquises sur le terrain, les images de cette base de données contiennent des véhicules issus d'un outil de simulation 3D, ce qui introduit nécessairement des éléments quelque peu différents de la réalité des images acquises sur le terrain. Pour répondre à ce problème, il serait intéressant d'utiliser un ensemble de données constitué en partie d'images réelles, complétées par un certain nombre d'images simulées. On pourrait ainsi évaluer la transférabilité des modèles appris sur les données simulées et appliqués aux images réelles, ou encore apprendre les modèles sur une base de données contenant les deux types d'images afin de les tester sur des images réelles.

**Intégration d'autres types de données complémentaires.** Dans nos travaux, on s'est intéressé à un type de données complémentaires particulier : la version haute-résolution des images d'apprentissage. Cette information a la particularité de permettre de classer les exemples dans l'espace privilégié, ce qui a constitué l'un des points de départ des différents modèles proposés. Pour la suite, il conviendrait de s'intéresser plus largement à d'autres types de données disponibles dans le contexte des images réelles, notamment les données de vol issues des autres capteurs du porteur. Ces données complémentaires peuvent concerner par exemple l'orientation du véhicule observé, sa distance au porteur, ou encore l'angle d'incidence de la caméra. Autrement dit, ces informations ne permettent pas de classer directement les exemples dans l'espace privilégié. Dans la littérature, plusieurs méthodes LUPI comme notamment SVM+ ont été proposées pour traiter de l'information privilégiée sans contrainte de reconnaissance dans l'espace privilégié. Il serait intéressant de proposer une nouvelle formulation pour intégrer ces données durant l'entraînement du modèle de reconnaissance. Une possibilité serait d'établir un modèle optimisant une fonction de coût multi-tâche, à l'image de la fonction de coût de Fast-RCNN (Girshick, 2015), qui vise à apprendre des classes en parallèle des coordonnées de l'objet recherché dans l'image. Dans notre contexte, on pourrait adapter cette idée afin de forcer le réseau à prédire les données du capteur en plus de la classe de l'image, ce qui pourrait améliorer la reconnaissance. Une autre approche pourrait consister à prendre en compte les images hautement résolues en plus de ces informations géométriques, et d'unifier les travaux de Su et al. (2016) sur la prise en compte du point de vue des objets avec une approche de type Generalized Distillation, où le CNN ciblé cherche à mimer les sorties du CNN privilégié. Une manière de traiter ce problème serait alors de chercher à reconnaître $N_{orientations} \times N_{classes}$ labels différents dans les deux espaces, en forçant le CNN dans l'espace ciblé à prédire une sortie similaire à celle du CNN privilégié.

**Intégration de la problématique de détection.** Dans une perspective à plus long terme, il convient de considérer le système de reconnaissance aéroporté dans sa globalité. En effet, les travaux présentés dans ce manuscrit sont focalisés sur l'aspect de classification des images, or les images qu'on cherche à reconnaître sont issues d'un module de détection, qui n'est pas parfait. Afin d'aborder le sujet dans une plus large mesure, il serait intéressant de proposer un système intégrant aussi bien la détection que la reconnaissance des véhicules. Plusieurs modèles de l'état de l'art comme les méthodes de type Faster R-CNN (Ren et al., 2015) ou les R-FCN de Dai et al. (2016) permettent d'apprendre à reconnaître ainsi qu'à localiser efficacement les objets dans les images. Couplées avec les techniques d'intégration de l'information privilégiée proposées dans ce manuscrit, on pourrait alors intégrer l'information complémentaire (qu'il s'agisse des images hautement résolues ou des données issues des autres capteurs) dans un système profond pour la localisation et la reconnaissance des objets dans la scène observée.

# Bibliographie


Fine-grained challenge 2013. https://sites.google.com/site/fgcomp2013/, 2013. 15

NATO Standardization Agreements. Stanag 3769 (about the minimal resolution needed for photographic interpretation), 1976, revised 1998. 6

Shotaro Akaho. A kernel method for canonical correlation analysis. *Proceedings of the International Meeting of the Psychometric Society*, 2001. 58

Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. *International Conference on Machine Learning (ICML)*, 2013. 58

Sandra Avila, Nicolas Thome, Matthieu Cord, Eduardo Valle, and Arnaldo de A. Araújo. Bossa : extended bow formalism for image classification. *IEEE International Conference on Image Processing (ICIP)*, pages 2966–2969, 2011. 15

Sandra Avila, Nicolas Thome, Matthieu Cord, Eduardo Valle, and Arnaldo de A. Araújo. Pooling in image representation : the visual codeword point of view. *Computer Vision and Image Understanding (CVIU)*, 117 :453–465, 2013. 15, 19, 21

Lei Jimmy Ba and Rich Caruana. Do deep nets really need to be deep ? *Advances in Neural Information Processing Systems (NIPS)*, 2014. 66

Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999. ISBN 020139829X. 3, 12

Nicolas Ballas, Li Yao, Pal Chris, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. *International Conference on Learning Representations (ICLR)*, 2016. 26

Herbert Bay, Tinne Tuytelaars, and Luc van Gool. Surf : Speeded up robust features. *European Conference on Computer Vision (ECCV)*, 2006. 13

Sean Bell, C. Lawrence Zitnick, Kavita Bala, and Ross Girshick. Inside-outside net : Detecting objects in context with skip pooling and recurrent neural networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 25, 50

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems (NIPS)*, pages 153–160, 2007. 22

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. *International Conference on Machine Learning (ICML)*, 2009. 5, 59

C.M. Bishop. *Neural Networks for pattern recognition*. 1995. 18

Michaël Blot, Matthieu Cord, and Nicolas Thome. Max-min convolutional neural networks for image classification. *IEEE International Conference on Image Processing (ICIP)*, 2016. 48

Avrim Blum and Tom Mitchell. Combining labeled and unlabeled datawith co-training. *Proceedings of the 11th annual Conference on Computational Learning Theory*, pages 92–100, 1998. 60

Mikael Boden. A guide to recurrent neural networks and backpropagation. *The Dallas project, SICS technical report*, 2002. 22

Bernhard Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. *Proceedings of the fifth annual ACM workshop on Computational learning theory*, pages 144–152, 1992. 19

Leo Breiman. Random forests. *Machine Learning*, 45(1) :5–32, oct 2001. 18

Cristian Bucila, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, August 2006. 62, 65

Micael Carvalho, Matthieu Cord, Sandra Avila, Nicolas Thome, and Eduardo Valle. Deep neural networks under stress. *IEEE International Conference on Image Processing (ICIP)*, 2016. 25, 28

Sarath Chandar, Mitesh M. Khapra, Hugo Larochelle, and Balaraman Ravindran. Correlational neural networks. *Neural Computation*, 28 :286–304, 2016. 58

Ken Chatfield, Victor Lempitsky, Andrea Vedaldi, and Adndrew Zisserman. The devil is in the details : an evaluation of recent feature encoding methods. In *British Machine Vision Conference (BMVC)*, 2011. 13, 15, 18, 21

Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details : Delving deep into convolutional nets. In *British Machine Vision Conference (BMVC)*, 2014. 24, 28, 33, 35, 50, 83

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *International Conference on Learning Representations (ICLR)*, 2015. 26

Lin Chen, Wen Li, and Dong Xu. Recognizing rgb images by learning from rgb-d data. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 61

Marion Chevalier, Nicolas Thome, Matthieu Cord, Jérôme Fournier, Gills Hénaff, and Elodie Dusch. Lr-cnn for fine-grained classification with varying resolution. *IEEE International Conference on Image Processing (ICIP)*, 2015. 9

Marion Chevalier, Nicolas Thome, Matthieu Cord, Jérôme Fournier, Gilles Hénaff, and Elodie Dusch. Low resolution convolutional neural network for automatic target recognition. *7th International Symposium on Optronics in Defence and Security*, 2016. 9

Kyunghyun Cho, Bart Van Merriënboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. 22

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *International Conference on Learning Representations (ICLR)*, 2016. 48

Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet. Torch7 : A matlab-like environment for machine learning. *BigLearn, NIPS workshop*, 2011. 26

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20 (3) :273–297, sep 1995. 3, 19

T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, IT-13(1), 1967. 18, 20

Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research (JMLR)*, 2001. 20

Nello Cristianini and John Shawe-Taylor. *An introduction to support vector machines and other kernel-based methods*. Cambridge University Press, 2000. 19

George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4) :303–314, 1989. 22

Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn : object detection via region-based fully convolutional networks. *Advances in Neural Information Processing Systems (NIPS)*, 2016. 25, 59, 108

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 886–893, June 2005. 3, 13

Olivier Delalleau, Yoshua Bengio, and Nicolas Le Roux. Efficient non-parametric function induction in semi-supervised learning. *AISTAT*, 2005. 60

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet : A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 4, 25, 28, 35

Ali Diba, Ali Mohammad Pazandeh, Hamed Pirsiavash, and Luc Van Gool. Deepcamp : Deep convolutional action & attribute mid-level patterns. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 25

Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. 2015. 26

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research (JMLR)*, 2011. 46

Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Second edition, 2001. 18

Thibaut Durand, Nicolas Thome, Matthieu Cord, and Sandra Avila. Image classification using object detectors. *IEEE International Conference on Image Processing (ICIP)*, 2013. 21

Thibaut Durand, Nicolas Thome, and Matthieu Cord. Mantra : Minimum-maximum latent structural svm for image classification and ranking. *IEEE International Conference on Computer Vision (ICCV)*, 2015. 60

Thibaut Durand, Nicolas Thome, and Matthieu Cord. Weldon : Weakly supervised learning of deep convolutional neural networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 26, 44, 60

Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 44

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR : A library for large linear classification. *Journal of Machine Learning Research (JMLR)*, 9 :1871–1874, 2008. Software available at `http://www.csie.ntu.edu.tw/~cjlin/liblinear/`. 36, 78

Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C. Platt, C. Lawrence Zitnick, and Geoffrey Zweig. From captions to visual concepts and back. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 26

Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(9), September 2010. 26

Jan Feyereisl and Uwe Aickelin. Privileged information for data clustering. *Information Sciences*, 2012. 65

Jan Feyereisl, Suha Kwak, Jeany Son, and Bohyung Han. Object localization based on structural svm using privileged information. *Advances in Neural Information Processing Systems (NIPS)*, 2014. 60, 61

Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, Philip Häusser, Caner Hazirbaş, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet : learning optical flow with convolutional networks. *arXiv :1504.06852*, 2015. 26

Shereen Fouad and Peter Tino. Ordinal-based metric learning for learning using privileged information. *International Joint Conference on Neural Networks*, 2013. 61

Shereen Fouad, Peter Tino, Somak Raychaudhury, and Petra Schneider. Incorporating privileged information through metric learning. *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, 2013. 61

Kunihiko Fukushima. Neocognitron : A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4) :193–202, 1980. 4, 22

Haoyuan Gao, Junhua Mao, Jie Zhou, Ziheng Huang, Lei Wang, and Wei Xu. Are you talking to a machine ? dataset and methods for multilingual image question answering. In *Advances in Neural Information Processing Systems (NIPS)*, 2015. 26

Peter Gehler and Sebastian Nowozin. On feature combination for multiclass object classification. *IEEE International Conference on Computer Vision (ICCV)*, 2009. 58

Ross Girshick. Fast r-cnn. *IEEE International Conference on Computer Vision (ICCV)*, 2015. 25, 44, 50, 107

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 25, 50

Ross Girshick, Forrest Iandola, Trevor Darrell, and Jitendra Malik. Deformable part models are convolutional neural networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 26

Georgia Gkioxari, Bharath Hariharan, Ross Girshick, and Jitendra Malik. R-cnns for pose estimation and action detection. 2014. 26

Georgia Gkioxari, Ross Girshick, and Jitendra Malik. Contextual action recognition with r*cnn. *IEEE International Conference on Computer Vision (ICCV)*, 2015. 25

Hanlin Goh, Nicolas Thome, Matthieu Cord, and Joo-Hwee Lim. Unsupervised and supervised visual codes with restricted boltzmann machines. *European Conference on Computer Vision (ECCV)*, 2012. 13

Hanlin Goh, Nicolas Thome, Matthieu Cord, and Joo-Hwee Lim. Top-down regularization of deep belief networks. *Advances in Neural Information Processing Systems (NIPS)*, 2013. 22

Yunchao Gong, Liwei Wang, Ruiqi Guo, and Svetlana Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. *European Conference on Computer Vision (ECCV)*, 2014. 19

Ian J. Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. *International Conference on Machine Learning (ICML)*, 2013. 49

Kristen Grauman and Trevor Darrell. The pyramid match kernel : Discriminative classification with sets of image features. *IEEE International Conference on Computer Vision (ICCV)*, 2005. 21

Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. LSTM : A search space odyssey. *CoRR*, abs/1503.04069, 2015. 22

Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid. Multimodal semi-supervised learning for image classification. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 60

Saurabh Gupta, Judy hoffman, and Jitendra Malik. Cross modal distillation for supervision transfer. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 62

David R. Hardoon, Sandor Szedmak, and John Shawe-Taylor. Canonical correlation analysis : an overview with application to learning methods. *Neural Computation*, 16(12) : 2639–2664, 2004. 58

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *European Conference on Computer Vision (ECCV)*, 2014. 25, 49

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers : surpassing human-level performance on imagenet classification. *IEEE International Conference on Computer Vision (ICCV)*, 2015. 48

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 24, 25, 42, 43, 45, 46, 47, 48, 49

Daniel Hernández-Lobato, Viktoriia Sharmanska, Kristian Kersting, Christoph H. Lampert, and Novi Quadrianto. Mind the nuisance : Gaussian process classification using privileged noise. *Advances in Neural Information Processing Systems (NIPS)*, 2014. 60

Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief networks. *Neural Computation*, 2006. 23

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky ans Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint*, 2012. 47

Geoffrey E. Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *NIPS Workshop on Deep Learning and Representation Learning*, 2014. 62, 65, 66

Judy Hoffman, Saurabh Gupta, and Trevor Darrell. Learning with side information through modality hallucination. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 61, 62

Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2) :251–257, mar 1991. ISSN 0893-6080. 2, 22

Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28 :321–377, December 1936. 58

Shaoli Huang, Zhe Xu, Dacheng Tao, and Ya Zhang. Part-stacked cnn for fine-grained visual categorization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 25

Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. jun 2010. 14

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe : Convolutional architecture for fast feature embedding. *ACM Multimedia (ACM MM)*, 2014. 26

Justin Johnson, Andrej Karpathy, and Li Fei-Fei. Densecap : Fully convolutional localization networks for dense captioning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 26, 44

Frédéric Jurie and Bill Triggs. Creating efficient codebooks for visual recognition. *IEEE International Conference on Computer Vision (ICCV)*, 2005. 13

Diederick P. Kingma and Jimmy Lei Ba. Adam : a method for stochastic optimization. *International Conference on Learning Representations (ICLR)*, 2015. 46

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael Bernstein, and Li Fei-Fei. Visual genome : Connecting language and vision using crowdsourced dense image annotations. *arXiv*, 2016. 26

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, April 2009. 25

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems (NIPS)*, 2012. 4, 24, 25, 26, 28, 42, 43, 45, 46, 47, 48, 49

Benjamin Labbé. *Machines à noyaux pour le filtrage d'alarmes : application à la discrimination multiclasse en environnement maritime*. PhD thesis, Thales Optronique S.A.S., INSA Rouen, 2011. 20

Maksim Lapin, Matthias Hein, and Bernt Schiele. Learning using privileged information : SVM+ and weighted SVM. *Neural Networks*, 53 :95–108, 2014. 61, 63, 76

Marc T. Law, Nicolas Thome, and Matthieu Cord. Hybrid pooling fusion in the bow pipeline. *ECCV Workshop on Information fusion in Computer Vision for Concept Recognition (ECCV-IFCVCR)*, 2012. 13, 19

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features : Spatial pyramid matching for recognizing natural scene categories. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2 :2169–2178, June 2006. 15, 16, 17, 21, 49

Yann LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L.D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4) :541–551, December 1989. 4, 22, 45

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, pages 2278–2324, November 1998. 4, 9, 22, 23, 24, 76, 94

Karen Lenc and Andrea Vedaldi. R-cnn minus r. *British Machine Vision Conference (BMVC)*, 2015a. 25

Karen Lenc and Andrea Vedaldi. Understanding image representations by measuring their equivariance and equivalence. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015b. 25

Hanxi Li, Yi Li, and Fatih Porikli. Deeptrack : learning discriminative feature representations by convolutional neural networks for visual tracking. *British Machine Vision Conference (BMVC)*, 2014. 26

Wen Li, Dengxin Dai, Mingkui Tan, Dong Xu, and Luc Van Gool. Fast algorithms for linear and kernel svm+. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 60, 61, 63, 76

Ming Liang and Xiaolin Hu. Recurrent convolutional neural network for object recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 25

Xiaodan Liang, Yunchao Wei, Xiaohui Shen, Zequn Jie, Jiashi Feng, Liang Lin, and Shuicheng Yan. Reversible recursive instance-level object segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 26, 44

Shengcai Liao, Xiangxin Zhu, Zhen Lei, Lun Zhang, and Stan Z. Li. Learning multi-scale block local binary patterns for face recognition. *International Conference on Biometrics*, page 828837, 2007. 13

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco : Common objects in context. *European Conference on Computer Vision (ECCV)*, 2014. 4, 26, 28

Lingqiao Liu, Lei Wang, and Xinwang Liu. In defense of soft-assignment coding. *IEEE International Conference on Computer Vision (ICCV)*, 2011. 14, 19

Stuart P. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 1982. 13

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, nov 2015. 26, 44

David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. Unifying distillation and privileged information. *International Conference on Learning Representations (ICLR)*, 2016. 5, 61, 62, 65, 66, 76, 79, 92, 95, 98, 106, 127

David G. Lowe. Object recognition from local scale-invariant features. *IEEE International Conference on Computer Vision (ICCV)*, pages 1150–1157, 1999. 3, 13, 35

Wei-Ying Ma and B. S. Manjunath. Netra : A toolbox for navigating large image databases. *ACM Multimedia Systems*, 7 :184–198, 1999. 3, 12

Andrew L. Maas, Awni Y. Hannun, and Andrew Ng. Rectifier nonlinearities improve neural network acoustic models. *International Conference on Machine Learning (ICML)*, 2013. 48

Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 25

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013. ix, 9, 32, 33, 34, 51, 53, 76, 94

Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz. Ask your neurons : A neural-based approach to answering questions about images. *IEEE International Conference on Computer Vision (ICCV)*, 2015. 26

Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan L. Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *International Conference on Learning Representations (ICLR)*, 2015. 44

Michael Mathieu, Camille Couprie, and Yann Lecun. Deep multi scale video prediction beyond mean square error. *International Conference on Learning Representations (ICLR)*, 2016. 26

Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4) :115–133, 1943. 2, 21

Frank Moosmann, Bill Triggs, and Frédéric Jurie. Fast discriminative visual codebooks using randomized clustering forest. *Advances in Neural Information Processing Systems (NIPS)*, pages 985–992, Dec 2006. 18

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. *International Conference on Machine Learning (ICML)*, 2010. 48

Romain Negrel, David Picard, and Philippe-Henri Gosselin. Compact tensor based image representation for similarity search. Orlando, United States, September 2012a. 14

Romain Negrel, David Picard, and Philippe-Henri Gosselin. Using spatial pyramids with compacted vlat for image categorization. Tsukuba, Japan, November 2012b. 14

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled : high confidence predictions for unrecognizable images. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 25

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*, December 2008. 32

Hyeonwoo Noh, Paul Hongsuck Seo, and Bohyung Han. Image question answering using convolutional neural network with dynamic parameter prediction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 26

Aude Oliva and Antonio Torralba. Modeling the shape of the scene : a holistic representation of the spatial envelope. *International Journal of Computer Vision (IJCV)*, 42 : 145175, 2001. 13

Andreas Opelt, Michael Fussenegger, Axel Pinz, and Peter Auer. Weak hypotheses and boosting for generic object detection and recognition. *European Conference on Computer Vision (ECCV)*, pages 71–84, 2004. 18, 59

Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 28

Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Is object localization for free ? - weakly-supervised learning with convolutional neural networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 25, 26, 44

Wanli Ouyang, Ping Luo, Xingyu Zeng, Shi Qiu, Yonglong Tian, Hongsheng Li, Shuo Yang, Zhe Wang, Yuanjun Xiong, Chen Qian, Zhenyao Zhu, Ruohui Wang, Chen-Change Loy, Xiaogang Wang, and Xiaoou Tang. Deepid-net : multi-stage and deformable deep convolutional neural networks for object detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 26

Dmitry Pechyony and Vladimir Vapnik. On the theory of learning using privileged information. *Advances in Neural Information Processing Systems (NIPS)*, 2010. 63

Dmitry Pechyony and Vladimir Vapnik. Fast optimization algorithms for solving svm+. *Chapter in Statistical Learning and Data Science*, 2011. 63, 78

Anastasia Pentina, Viktoriia Sharmanska, and Christoph H. Lampert. Curriculum learning of multiple tasks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 59

Florent Perronnin and Christopher Dance. Fisher kernels on visual vocabularies for image categorization. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007. 14, 15, 17

Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. *European Conference on Computer Vision (ECCV)*, 2010. 14, 15, 18, 19, 21, 35

David Picard, Nicolas Thome, and Matthieu Cord. An efficient system for combining complementary kernels in complex visual categorization tasks. *IEEE International Conference on Image Processing (ICIP)*, 2010. 58

David Picard, Nicolas Thome, Matthieu Cord, and Alain Rakotomamonjy. Learning geometric combinations of gaussian kernels with alternating quasi-newton algorithm. *European Symposium on Artificial Neural Networks (ESANN)*, 2012. 58

David Picard, Nicolas Thome, and Matthieu Cord. JKernelMachines : A Simple Framework for Kernel Machines. *Journal of Machine Learning Research (JMLR)*, 14 : 1417–1421, May 2013. 21

Matti Pietikäinen, Abdenour Hadid, Guoying Zhao, and Timo Ahonen. Computer vision using local binary patterns. *Springer*, 2011. 3, 13

Alain Rakotomamonjy, Francis Bach, Stéphane Canu, and Yves Grandvalet. Simplemkl. *Journal of Machine Learning Research (JMLR)*, pages 2491–2521, 2008. 58

Viresh Ranjan, Nikhil Rasiwasia, and C. V. Jawahar. Multi-label cross-modal retrieval. *IEEE International Conference on Computer Vision (ICCV)*, 2015. 58

MarcAurelio Ranzato, Christopher Poultney, Sumit Chopra, and Yann LeCun. Efficient learning of sparse representations with an energy-based model. *Advances in Neural Information Processing Systems (NIPS)*, 19 :1137–1144, 2006. 22

Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3546–3554, 2015. 60

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf : An astounding baseline for recognition. In *CVPR DeepVision Workshop*, pages 512–519, 2014. 25, 28, 35

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once : unified, real-time object detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 25

Scott Reed, Zeynep Akata, Honglak Lee, and Bernt Schiele. Learning deep representations of fine-grained visual descriptions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 25

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn : towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems (NIPS)*, 2015. 25, 108

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets : Hints for thin deep nets. *International Conference on Learning Representations (ICLR)*, 2015. 66

Frank Rosenblatt. The perceptron : A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386, 1958. 2, 20, 21

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. *Learning representations by back-propagating errors*, volume 1. 1986. 22, 45

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115(3) :211–252, 2015. 15, 25

Pierre-André Savalle, Stavros Tsogkas, George Papandreou, and Iasonas Kokkinos. Deformable part models with cnn features. *ECCV workshop, Parts and Attributes*, September 2014. 26

Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2) :197–227, jul 1990. 18, 59

Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann LeCun. Overfeat : Integrated recognition, localization and detection using convolutional networks. *International Conference on Learning Representations (ICLR)*, April 2014. 25

Viktoriia Sharmanska, Novi Quadrianto, and Christoph H. Lampert. Learning to rank using privileged information. In *IEEE International Conference on Computer Vision (ICCV)*, 2013. 4, 60, 62, 63

Viktoriia Sharmanska, Novi Quadrianto, and Christoph H. Lampert. Learning to transfer privileged information. In *arXiv :1410.0389*, 2014. 4, 5, 9, 60, 62, 63, 67, 70, 74, 77, 78, 79, 90, 95, 106

Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard exemple mining. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 59

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale visual recognition. *International Conference on Learning Representations (ICLR)*, 2015. 24, 25, 43, 48, 49

Ayan Sinha, Chiho Choi, and Karthik Ramani. Deephand : Robust hand pose estimation by completing a matrix imputed with deep features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 26

Josef Sivic and Andrew Zisserman. Video Google : A text retrieval approach to object matching in videos. *IEEE International Conference on Computer Vision (ICCV)*, 2 : 1470–1477, October 2003. 3, 12

Arnold W. M. Smeulders, Marcel Worring, Simone Santini, Amarnath Gupta, and Ramesh Jain. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(12) :1349–1380, December 2000. 1

Paul Smolensky. Information processing in dynamical systems : Foundations of harmony theory. In David E. Rumelhart, James L. McClelland, and PDP Research Group, editors, *Parallel Distributed Processing : Explorations in the Microstructure of Cognition, Volume 1 : Foundations*, chapter 6, pages 194–281. MIT Press, Cambridge, MA, USA, 1986. 23

Cees GM Snoek, Marcel Worring, and Arnold WM. Smeulders. Early versus late fusion in semantic video analysis. *ACM Multimedia (ACM MM)*, 2005. 4, 58

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout : a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research (JMLR)*, 2014. 46

Hao Su, Charles R. Qi, Yangyan Li, and Leonidas Guibas. Render for cnn : Viewpoint estimation in images using cnns trained with rendered 3d model views. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 107

Ilya Sutskever, James Martens, George Dahl, and Geoffrey E. Hinton. On the importance of intialization and momentum in deep learning. *International Conference on Machine Learning (ICML)*, 2015. 45

Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep neural networks for object detection. *Advances in Neural Information Processing Systems (NIPS)*, 2013. 25

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *International Conference on Learning Representations (ICLR)*, 2014. 25

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 24, 25, 42, 43, 45, 46, 47, 49

TensorFlow Development Team. TensorFlow : Large-scale machine learning on heterogeneous systems, 2015. Software available from `http://tensorflow.org/`. 26

Theano Development Team. Theano : A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016. Software available from `http://deeplearning.net/software/theano/`. 26

Tijmen Tieleman and Geoffrey E. Hinton. Lecture 6e - rmsprop : Divide the gradient by a running average of its recent magnitude. *unpublished work - COURSERA : Neural Networks for Machine Learning*, 2012. 46

Alexander Toshev and Christian Szegedy. Deeppose : Human pose estimation via deep neural networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1653–1660, 2014. 26

Vladimir Vapnik and Rauf Izmailov. Learning using privileged information : Similarity control and knowledge transfer. *Journal of Machine Learning Research (JMLR)*, 16 : 2023–2049, 2015. 61

Vladimir Vapnik and Akshay Vashist. A new learning paradigm : Learning using privileged information. *Neural Networks*, 2009. 4, 60, 61, 62, 65, 67, 74, 76, 77, 79, 88, 89, 90, 92, 95, 103, 106

Andrea Vedaldi and Brian Fulkerson. VLFeat : An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`, 2008. 35

Andrea Vedaldi and Karel Lenc. Matconvnet – convolutional neural networks for matlab. *ACM Multimedia (ACM MM)*, 2015. Software available at http://www.vlfeat.org/matconvnet/. 26, 35

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell : A neural image caption generator. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 26, 44

Paul Viola and Michael Jones. Robust real-time object detection. *IEEE International Conference on Computer Vision (ICCV)*, 2001. 18, 59

Li Wan, Matthew Zeiler, Sixin Zhang, Yann LeCun, and Rob Fergus. Regularization of neural networks using dropconnect. *International Conference on Machine Learning (ICML)*, pages 1058–1066, 2013. 46

Li Wan, David Eigen, and Rob Fergus. End-to-end integration of a convolutional network, deformation parts model and non-maximum suppression. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 25, 26

Jinjun Wang, Jianchao Yang, KaiYu, Fengjun Lv, Thomas Huang, and Yihong Gong. Locality-constrained linear coding for image classification. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 19

Lijun Wang, Wanli Ouyang, Xiaogang Wang, and Huchuan Lu. Visual tracking with fully convolutional networks. *IEEE International Conference on Computer Vision (ICCV)*, 2015a. 26

Xin Wang, Devinder Kumar, Nicolas Thome, Matthieu Cord, and Frédéric Precioso. Recipe recognition with large multimodal food dataset. *IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, 2015b. x, 9, 51, 52, 54, 58, 76, 94

Ziheng Wang and Qiang Ji. Classifier learning with hidden information. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 60, 62, 94, 95

Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-ucsd birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010. 32

Paul Werbos. *Beyond Regression : New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, 1974. 22

Lior Wolf and Noga Levy. The svm-minus similarity score for video face recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 62

Qi Wu, Peng Wang, Chunhua Shen, Anthony Dick, and Anton van den Hengel. Ask me anything : Free-form visual question answering based on knowledge from external sources. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 26

Xiaoyun Wu and Rohini Srihari. Incorporating prior knowledge with weighted margin support vector machines. *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 326–333, 2004. 59

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell : Neural image caption generation with visual attention. *International Conference on Machine Learning (ICML)*, 2015. 26

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alexander J. Smola. Stacked attention networks for image question answering. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 26

Bangpeng Yao and Li Fei-Fei. Grouplet : A structured image representation for recognizing human and object interactions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, USA, June 2010. ix, 32, 33

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks ? *Advances in Neural Information Processing Systems (NIPS)*, 2014. 25, 38

Matthew D. Zeiler. Adadelta : an adaptive learning rate method. *arXiv*, 2012. 46

Matthrew Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *European Conference on Computer Vision (ECCV)*, 2014. 25, 27

Ning Zhang, Jeff Donahue, Ross Girshick, and Trevor Darrell. Part-based r-cnns for fine-grained category detection. *European Conference on Computer Vision (ECCV)*, 2014. 25, 44

Wenchao Zhang, Shiguang Shan, Wen Gao, Xilin Chen, and Hongming Zhang. Local gabor binary pattern histogram sequence (lgbphs) : A novel non-statistical model for face representation and recognition. *IEEE International Conference on Computer Vision (ICCV)*, pages 786–791, 2005. 13

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip Torr. Conditional random fields as recurrent neural networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 26

Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. *Advances in Neural Information Processing Systems (NIPS)*, 2014. 28

Xi Zhou, Kai Yu, Tong Zhang, and Thomas S. Huang. Image classification using super-vector coding of local image descriptors. *European Conference on Computer Vision (ECCV)*, pages 141–154, 2010. 14

# Annexe A

# Données simulées Thales

Cette partie est confidentielle.

# Annexe B

# Détails sur l'algorithme Generalized Distillation

## B.1 Formulation du coût pour la classification mono-label

Dans leurs travaux, (Lopez-Paz et al., 2016) introduisent le cadre Generalized Distillation selon l'équation de coût suivante :

$$loss_{GD} = (1-\lambda)loss(z_i, y_i) + \lambda loss(z_i, \sigma^*_{\mathbf{i}}) \tag{B.1}$$

où à chaque image $i$ est associé un vecteur de labels $y_i$, qui contient un 1 pour chaque classe présente dans l'image, et des 0 partout ailleurs. Dans le cadre de nos travaux, on ne considère que des cas mono-labels : le vecteur $y_i$ contient donc un 1 pour la classe vérité-terrain $c$ de l'image $i$, et des 0 partout ailleurs. Pour chaque image $i$, $\sigma^*_{\mathbf{i}}$ est la matrice des sorties du softmax dans l'espace privilégié

$$\sigma^*_k = \frac{e^{z^*_k/T}}{\sum_{k'=1}^{N_{classes}} e^{z^*_{k'}/T}} \tag{B.2}$$

$\lambda$ est un paramètre réglant un compromis entre le coût dans l'espace ciblè $loss(z_i, y_i)$ et le coût dans l'espace privilégié $loss(z_i, \sigma^*_{\mathbf{i}})$, et $T$ est un hyper-paramètre de température permettant de régler l'étalement des scores dans l'espace privilégié.

Afin de bénéficier du détail des scores privilégiés $\sigma^*_k$, on utilise l'entropie croisée pour calculer le coût dans chacun des espaces. On présente ici le détail du calcul.

Le coût propre à l'espace ciblé est

$$loss(z_i, y_i) = -\sum_{k=1}^{N_{classes}} y_k \ln(\tilde{z}_k) \tag{B.3}$$

$$= -\sum_{k=1}^{N_{classes}} y_k \ln\left(\frac{e^{z_k}}{\sum_{k'=1}^{N_{classes}} e^{z_{k'}}}\right) \tag{B.4}$$

$$= -\ln\left(\frac{e^{z_c}}{\sum_{k'=1}^{N_{classes}} e^{z_{k'}}}\right) \tag{B.5}$$

$$= -z_c + \ln\left(\sum_{k'=1}^{N_{classes}} e^{z_{k'}}\right) \tag{B.6}$$

et le coût relatif à l'espace privilégié est

$$loss(z_i, \sigma_{\mathbf{i}}^*) = -\sum_{k=1}^{N_{classes}} \sigma_k^* \ln(\tilde{z}_k) \tag{B.7}$$

$$= -\sum_{k=1}^{N_{classes}} \sigma_k^* \ln\left(\frac{e^{z_k}}{\sum_{k'=1}^{N_{classes}} e^{z_{k'}}}\right) \tag{B.8}$$

$$= -\sum_{k=1}^{N_{classes}} \sigma_k^* \left[z_k - \ln\left(\sum_{k'=1}^{N_{classes}} e^{z_{k'}}\right)\right] \tag{B.9}$$

$$= -\sum_{k=1}^{N_{classes}} \sigma_k^* z_k + \ln\left(\sum_{k'=1}^{N_{classes}} e^{z_{k'}}\right) \sum_{k=1}^{N_{classes}} \sigma_k^* \tag{B.10}$$

$$= -\sum_{k=1}^{N_{classes}} \sigma_k^* z_k + \ln\left(\sum_{k'=1}^{N_{classes}} e^{z_{k'}}\right) \tag{B.11}$$

puisque les $\sigma_k^*$ somment à 1.

Ainsi, en intégrant dans l'équation B.1,

$$loss_{GD} = (1-\lambda)\left[-z_c + \ln\left(\sum_{k'=1}^{N_{classes}} e^{z_{k'}}\right)\right] + \lambda\left[-\sum_{k=1}^{N_{classes}} \sigma_k^* z_k + \ln\left(\sum_{k'=1}^{N_{classes}} e^{z_{k'}}\right)\right] \tag{B.12}$$

*i.e.*

$$loss_{GD} = -(1-\lambda)z_c + \ln\left(\sum_{k'=1}^{N_{classes}} e^{z_{k'}}\right) - \lambda \sum_{k=1}^{N_{classes}} \sigma_k^* z_k \tag{B.13}$$

## B.2 Détail d'implémentation

D'un point de vue calculatoire, calculer $\sum_{k'=1}^{N_{classes}} e^{z_{k'}}$ peut s'avérer risqué, car on peut alors se retrouver avec des valeurs d'exponentielles très grandes. Pour éviter ces problèmes, on peut introduire un conditionnement simple de l'équation B.13, qui s'écrit comme suit :

$$loss_{GD} = -(1-\lambda)z_c + z_{\max} + \ln\left(\sum_{k'=1}^{N_{classes}} e^{z_{k'} - z_{\max}}\right) - \lambda \sum_{k=1}^{N_{classes}} \sigma_k^* z_k \tag{B.14}$$

où $z_{\max} = \max_k(z_k)$. Cette méthode permet de recentrer les arguments en entrée des exponentielles à calculer, et ainsi d'éviter les problèmes des valeurs trop grandes. Il convient bien évidemment de remarquer que ce détail d'implémentation est applicable aussi dans le cas d'un coût softmax + log-loss classique. Dans nos expériences, on utilise toujours cette implémentation pour éviter ce problème de conditionnement.