Université ABBES LAGHROUR Khenchela
Faculté des Sciences et de la Technologie
Département de Génie Industriel
جامـــعة عبـــاس لـــــغرور خـنـشـلة
كليـــة العلـــوم والتكنـــولـــوجيـا
قسـم الهنـدسة الصنـاعية

N° Série :.....

Mémoire de fin d'étude

Pour l'obtention du diplôme de Master

Filière : Télécommunications

Spécialité : Systèmes des Télécommunications

THEME

Initiation aux réseaux de neurones convolutifs : Application à la détection d'objet

Réalisé par ***: - HAFIDI SIF EDDINE***
- RAHABI HAITHEM

Soutenu le 15/09/2020 Devant le jury compsé de :

Dr. KHEZZAR Zaki	***Président , Université Abbès Laghrour, Khenchela***
Dr. FRIHA SOUAD	***Encadreur, Université Abbès Laghrour, Khenchela***
Dr. CHERGUI Laid	***Examinateur, Université Abbès Laghrour, Khenchela***

Promotion 2019/2020

DEDICACES

GRANDE DEDICACE A LA FAMILLE, LES AMIS, LES ENSEIGNANTS, LES COLLEGUES...

REMERCIEMENTS

Ce travail est l'aboutissement d'un dur labeur et de beaucoup de sacrifices; nos remerciements vont d'abord à ALLAH qui nous a doté d'intelligence, et nous a maintenu en santé pour mener à bien cette année d'étude. Nous tenons aussi à adresser nos remerciements à nos familles respectives qui nous ont toujours soutenus et poussés à continuer nos études. Ce présent travail a pu voir le jour grâce à leur soutien.

Mes profonds remerciements sont adressés d'abord à notre encadreur FRIHA SOUAD, qui a veillé à la qualité de cette recherche qui nous l'espérons, serait à la hauteur de l'ambition souhaitée.

Nous souhaitons également remercier tous les collègues et amis dans le secteur de la télécommunication et l'informatique interrogés dans le cadre de cette étude, qui ont bien voulu répondre à nos différentes questions sur le domaine dans le but de permettre la facilité de l'élaboration de ce travail.

Nos remerciements sont aussi adressés aux honorables membres du jury qui ont accepté la lecture et l'évaluation de ce travail.

En dernier lieu je remercie l'administration de l'université ABBES LAGHROUR qui nous a fourni le cadre et les moyens pédagogiques nécessaires à une formation de qualité.

Table des matières

Chapitre 2 : Les réseaux de neurones convolutifs

Chapitre 3 : Conception et implémentation

Liste des figures

Page

Introduction générale

Ces dernières années la quantité des images qui circulent ne cesse d'augmenter dans le monde. De même le taux de croissance lui-même augmente. Info Trends [3] estime qu'en 2016 les appareils photo et les appareils mobiles ont capturé plus de 1,1 billions d'images. Selon la même estimation, en 2020, le chiffre passerait à 1,4 billions. Beaucoup de ces images sont stockées dans des services cloud ou publiées sur Internet. En 2014, plus de 1,8 milliard d'images ont été téléchargées quotidiennement sur les sites plates-formes, comme Instagram et Facebook [4].

Au-delà des appareils grand public, il existe des caméras partout dans le monde qui capturent des images à des fins d'automatisation. Ainsi, les caméras peuvent fournir des images à des robots chargés de comprendre une scène visuelle afin de construire intelligemment des appareils ou de trier les déchets. Elles peuvent être utilisées pour la surveillance de la route et

des voitures. Partout par ailleurs, des dispositifs d'imagerie sont utilisés par des ingénieurs, des médecins et des explorateurs de l'espace.

Pour gérer efficacement toutes ces images, nous devons avoir une idée sur son contenu. Le traitement automatisé du contenu de l'image est utile pour une large variété de tâches liées à l'image. Pour les systèmes informatiques, cela signifie traverser l'écart dit sémantique entre les informations de niveau de pixel stockées dans les images et la compréhension humaine de ces mêmes images. Les dits ordinateurs de vision tentent de combler cet écart.

Problématique

La détection d'objets est l'un des problèmes de base de l'ordinateur de vision. Elle consiste à localiser et identifier automatiquement les objets contenus dans les fichiers image. Les réseaux de neurones convolutifs sont actuellement la solution de pointe pour la détection d'objets [ ].

La tâche principale de ce travail consiste à examiner et à tester les méthodes de détection d'objets par réseaux de neurones convolutifs. Dans la partie théorique, nous passons en revue la littérature pertinente dans ce domaine et étudions comment les méthodes de détection d'objets par convolution se sont améliorées ces dernières années. Dans la partie expérimentale, nous étudions la mise en œuvre pratique d'un système convolutif de détection d'objets. Nous testons la qualité de ce système sur des données d'images prises des bases de données sharware (disponibles au grand public) puis sur notre propre base de donnée.

Structure du mémoire

Le mémoire commence par deux chapitres théoriques dans lesquels sont exposées les fondements théoriques de la technique appliquée dans cette étude. Nous montrons que c'est une combinaison de plusieurs domaines informatiques qui semblent à première vue différents.

Dans le premier chapitre nous commençons par une brève introduction à l'apprentissage automatique et aux réseaux de neurones. Et nous discutons de la vision par ordinateur et de la détection d'objets comme un sous-domaine en présentant les réseaux de neurones convolutifs dans le cas particulier de la vision par ordinateur.

Au deuxième chapitre nous faisons un tour d'horizon sur les méthodes les plus pertinentes citées en littérature concernant les réseaux de neurones convolutifs.

Au troisième chapitre nous passons à la partie expérimentale. Nous discutons le type de configuration expérimentale que nous avons utilisé pour construire et tester un réseau convolutif. Nous discutons non seulement les détails des expériences, mais aussi les détails des ensembles de données. De plus, nous discutons la manière dont nous évaluerons les résultats. Nous discutons la mise en œuvre pratique des expériences en discutant des logiciels et matériel et comment ils ont été utilisés.

Nous terminons enfin par un glossaire qui contient par ordre alphabétique les termes et notions cités mais non explicités dans le manuscrit. Même s'il n'est pas trop détaillé nous pensons qu'il permettrait une lecture aisée et confortable du mémoire.

Chapitre 1

Apprentissage automatique et réseaux de neurones

Introduction

Dans ce chapitre, nous fournissons le contexte théorique nécessaire pour comprendre les méthodes discutées dans le chapitre suivant. Tout d'abord, nous discutons des détails de l'apprentissage automatique, des réseaux de neurones et de la vision par ordinateur. Finalement, nous expliquons comment ces disciplines sont combinées dans des réseaux de neurones convolutifs CNN (Convolutional Neural Networks).

1.1. Apprentissage automatique

Les algorithmes d'apprentissage sont largement utilisés dans les applications de vision par ordinateur. Avant de considérer le cas particulier de l'imagerie, nous allons commencer par un bref aperçu des bases de l'apprentissage automatique en général.

L'apprentissage automatique est devenu un outil utile pour modéliser des problèmes qui sont difficiles à formuler d'une manière rigoureuse. Les programmes informatiques classiques sont des codes faits «à la main» pour effectuer une tâche donnée. Avec l'apprentissage automatique, une partie de la contribution humaine est remplacée par un algorithme d'apprentissage [22, pp. 2]. La disponibilité de la capacité de calcul et des données augmentant, l'apprentissage automatique est devenu de plus en plus pratique au fil des ans au point de devenir omniprésent.

1.1.1. Types d'apprentissage automatique

La classification et la régression sont les types de tâches les plus importants [11, pp. 3]. En classification, l'algorithme tente de prédire la classe correcte d'une nouvelle donnée basé sur les données d'entraînement. Dans la régression, au lieu des classes discrètes, l'algorithme essaie de prédire une sortie continue.

En classification, une manière typique d'utiliser l'apprentissage automatique est l'apprentissage supervisé. Un algorithme d'apprentissage montre plusieurs exemples qui ont été annotés ou étiquetés par les humains. Par exemple, dans le problème de détection d'objets, nous utilisons des images d'entraînement où les humains ont marqué les emplacements et les classes des objets pertinents. Après avoir appris des exemples, l'algorithme est capable de prédire les annotations ou les étiquettes de données précédemment invisibles. Dans l'apprentissage non supervisé, l'algorithme tente d'apprendre des propriétés utiles des données sans qu'un enseignant humain ne dise quelle sortie correcte devrait être. Un exemple classique d'apprentissage non supervisé est le clustering [11, pp. 3].

Plus récemment, notamment avec l'avènement des technologies de Deep Learning non supervisées le prétraitement est devenu un outil populaire dans l'apprentissage supervisé. Le prétraitement est une tâche incontournable en général destinée à découvrir des représentations utiles des données [9].

1.1.2. Extraction des caractéristiques

Une sorte de prétraitement est presque toujours nécessaire. Le prétraitement des données dans un nouvel espace variable plus simple est appelé extraction de caractéristiques [11, pp. 2]. Souvent, il est peu pratique ou impossible d'utiliser les données d'entraînement telles quelles en pleine dimension directement. Des détecteurs sont programmés pour extraire des caractéristiques intéressantes à partir des données, et ces caractéristiques sont utilisées comme entrées dans l'apprentissage automatique de l'algorithme. Dans le passé, les détecteurs étaient souvent conçus pour extraire des caractéristiques fixées et choisies au préalable. Le problème avec cette approche, c'est qu'on ne sait pas toujours à l'avance quelles caractéristiques sont les plus intéressantes. La tendance de l'apprentissage actuel

est d'automatiser cette tâche d'extraction des détecteurs, ce qui permet en outre d'utiliser les données complètes [22, pp. 3-5].

1.1.3. Généralisation

Étant donné que les données d'entraînement ne peuvent pas inclure toutes les instances possibles des entrées, l'algorithme d'apprentissage doit être capable de généraliser afin de gérer la partie invisible de données [11, pp. 2].

Une sous-estimation du modèle (en négligeant trop de détails) peut ne pas saisir tous les aspects importants du vrai modèle. En revanche, des méthodes trop complexes peuvent conduire à une sur-modélisation des détails et du bruit sans importance, ce qui conduit également à une mauvaise généralisation [11, pp. 9].

En règle générale, une sur-apprentissage se produit lorsqu'un modèle complexe est utilisé avec trop peu de données d'entraînement. Un modèle surestimé apprend à modéliser les exemples connus mais ne comprend pas ce qui les relie.

La performance de l'algorithme peut être évaluée à partir de la qualité et de la quantité d'erreurs. Une fonction de perte, telle que l'erreur quadratique moyenne, est utilisée pour attribuer un coût aux erreurs [11, pp. 41]. L'objectif de la phase d'entrainement est de minimiser cette perte.

1.2. Réseaux de neurones

Les réseaux de neurones sont un type populaire de modèle d'apprentissage automatique. Un cas spécial de réseaux de neurones, appelé réseau neuronal convolutif CNN (Convolutive Neural Network), est l'objectif principal de notre étude. Avant de discuter des CNN, nous devrions au préalable aborder l'origine des réseaux de neurones et expliciter le principe de leur fonctionnent.

1.2.1. Origines

Les réseaux de neurones étaient à l'origine appelés réseaux de neurones artificiels, car ils ont été développés pour imiter la fonction neurale du cerveau humain. Les travaux de

recherche innovateurs dans ce domaine comprennent le neurone binaire (Sortie 0 ou 1) à seuil de seuil de Warren McCulloch et Walter Pitts en 1943 et le perceptron de Frank Rosenblatt en 1957 [41]. Même si l'inspiration de la biologie est apparente, elle serait trompeuse, il ne faudrait pas trop insister sur le lien entre les neurones artificiels et les neurones biologiques. En effet, le cerveau humain contient environ 100 milliards de neurones fonctionnant en parallèle [37]. Les neurones artificiels sont quant à eux des fonctions mathématiques implémentées sur des ordinateurs plus ou moins en série. En outre, la recherche dans les réseaux de neurones est principalement guidée par les développements en ingénierie et les mathématiques plutôt que la biologie [22, pp. 169].

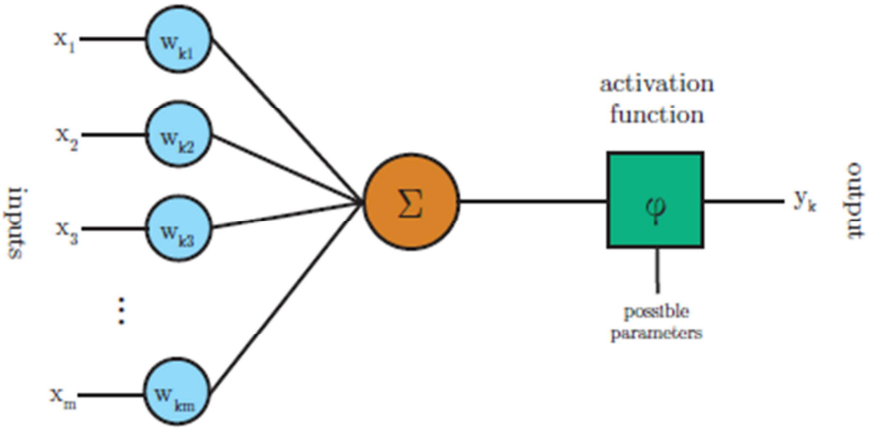


Figure 1.1. Un neurone artificiel.

Un neuronee artificiel basé sur le modèle McCulloch-Pitts [11, pp. 227-229] est présenté dans Figure 1.1. Le neurone k reçoit m paramètres d'entrée x_j. Le neurone a également m paramètres de poids w_{kj}. Les paramètres de poids souvent incluent un terme de biais. Les entrées et les poids sont linéairement combinés et additionnés. La somme est alors présentée à une fonction d'activation qui produit la sortie y_k du neurone k.

Le neurone est formé en sélectionnant soigneusement les poids pour produire une sortie souhaitée pour chaque entrée.

1.2.2. Réseaux multicouches

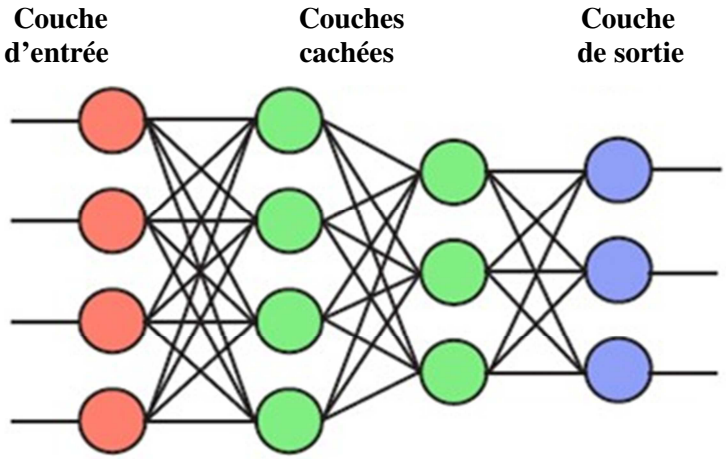


Figure 1.2. Un réseau neuronal multicouche entièrement connecté.

Un réseau de neurones est une combinaison de neurones artificiels. Les neurones sont généralement regroupés en couches. Dans le réseau multicouche entièrement connecté illustré à Figure 1.2, chaque sortie d'une couche de neurones alimente l'entrée de chaque neurone de la couche suivante. Ainsi, certaines couches traitent les données émanant de l'entrée tandis que certaines traitent les données reçues d'autres neurones. Chaque neurone a un nombre de poids égal au nombre de neurones de la couche précédente [11, pp. 227-229].

Un réseau multicouche comprend généralement trois types de couches: une couche d'entrée, une ou plusieurs couches cachées et une couche de sortie [11, pp. 227]. La couche d'entrée transmet généralement simplement les données sans les modifier. La plupart du calcul se produit dans les couches cachées. La couche de sortie convertit les activations de couches masquées vers une sortie.

Un réseau multicouche à réaction avec au moins une couche cachée peut fonctionner comme un « approximateur" universel, c'est-à-dire peut être construit pour calculer presque n'importe quelle fonction [27].

Dans ce mémoire nous abordons principalement les réseaux entièrement connectés et les réseaux convolutifs (voir section 1.3.3). Les réseaux convolutifs utilisent des paramètres de partage et ont des connexions limitées par rapport aux réseaux entièrement connectés

[22, pp. 335]. D'autres types de réseaux, tels que les réseaux récurrents sont au-delà de notre intérêt et ne sont pas inclus dans ce présent mémoire.

1.2.3 Rétro-propagation

Un réseau de neurones est entrainé en sélectionnant les poids de tous les neurones afin que le réseau apprenne à approximer les sorties cibles à partir d'entrées connues. C'est difficile de résoudre analytiquement les poids des neurones d'un réseau multicouche. L'algorithme de rétro-propagation [22, pp. 204-210] [11, pp. 241-245] fournit une solution simple et efficace pour résoudre les poids de manière itérative. La version classique utilise l'algorithme du gradient ou de la plus profonde descente (Gradient descent) comme méthode d'optimisation. La descente de gradient peut prendre beaucoup de temps et ne garantit pas de trouver le minimum global d'erreur, mais avec une configuration appropriée (connue en apprentissage automatique sous le nom d'hyper-paramètres) fonctionne assez bien dans la pratique. Dans la première phase de l'algorithme, un vecteur d'entrée est propagé vers l'avant à travers le réseau neuronal. Avant cela, les poids des neurones du réseau ont été initialisés à certaines valeurs, par exemple de petites valeurs aléatoires. La sortie reçue du réseau est comparée à la sortie souhaitée (qui doit être connue pour les exemples d'entrainement) en utilisant une fonction de perte. Le gradient de la fonction de perte est alors calculé. Ce gradient est également appelé la valeur d'erreur.

Lorsque l'erreur quadratique moyenne est utilisée comme fonction de perte, la valeur de l'erreur de la sortie est simplement la différence entre la sortie obtenue et la sortie désirée. Les valeurs d'erreur sont ensuite retournées au réseau pour calculer les valeurs d'erreur des neurones des couches cachées.

Les gradients de la fonction de perte des neurones cachés peuvent être calculés en utilisant la règle de la chaîne (Chaine's rule) des dérivées. Finalement, les poids des neurones sont mis à jour en calculant le gradient des poids et en soustrayant une proportion du gradient des poids. Ce rapport s'appelle le taux d'apprentissage [11, pp. 240]. Le taux d'apprentissage peut être fixe ou dynamique.

Une fois les poids mis à jour, l'algorithme continue à exécuter de nouveau les phases avec une entrée différente jusqu'à ce que les poids convergent. Dans la description ci-dessus,

nous avons décrit l'apprentissage en ligne qui calcule le poids mis à jour après chaque nouvelle entrée [22, pp. 277-279].

L'apprentissage en ligne peut conduire à un comportement en zigzag où l'estimation d'un point de donnée unique du gradient continue de changer de direction et ne s'approche pas du minimum directement.

Un autre moyen de calculer les mises à jour est l'apprentissage par lots complets, où nous calculons les mises à jour de poids pour l'ensemble complet des données [22, pp. 277-279]. Ceci est assez lourd en calcul et présente d'autres inconvénients. Un compromis est l'apprentissage par mini-lots, où nous n'utilisons qu'une partie de l'ensemble d'entrainement pour chaque mise à jour [54].

Les descriptions mathématiques de l'algorithme sont largement disponibles dans la littérature [42] [22, pp. 204-210].

1.2.4 Types de fonction d'activation

La fonction d'activation φ détermine la sortie finale de chaque neurone. Il est important de sélectionner correctement la fonction afin de créer un réseau efficace. Les premiers chercheurs ont découvert que les perceptrons et autres systèmes linéaires avaient des inconvénients graves, l'impossibilité de résoudre des problèmes non séparables linéairement, comme le problème XOR. Parfois les systèmes linéaires peuvent résoudre ce genre de problème en utilisant des détecteurs choisis manuellement mais ceci ne va pas avec le concept de l'apprentissage automatique. Le simple fait d'ajouter des couches n'apporte pas d'aide car un réseau composé de neurones linéaires reste linéaire peu importe le nombre de couches dont il dispose. [22, pp. 171-172].

Une manière simple et efficace de créer un réseau non linéaire consiste à utiliser des unités linéaires rectifiées ReLu (Rectified Linear units) [22, pp. 173-177]. Une fonction linéaire rectifiée génère la sortie à l'aide d'une fonction rampe telle que :

$$\varphi(s) = \max(0, s) \tag{1.1}$$

Ce type de fonction est facile à calculer et à différencier (pour la rétro-propagation). La fonction n'est pas différentiable à zéro, mais cela n'a pas empêché son utilisation dans la pratique. Les ReLus sont devenus très populaires ces derniers temps, souvent remplaçant

les fonctions d'activation sigmoïdales, qui ont des dérivés lisses mais souffrent de problèmes de saturation des gradients et de calculs plus lents.

Pour les problèmes de classification multi-classes, la fonction d'activation softmax [11, pp. 115, 203] est utilisée dans la couche de sortie du réseau :

$$\varphi(s)_j = \frac{e^{sj}}{\sum_{k=1}^{K} e^{sk}} \tag{1.2}$$

Pour tout j ∈{1, 2, …K} K étant le nombre maximal de classes.

La fonction softmax permet de générer un score de probabilité normalisé avec la somme totale des probabilités étant égale à 100%, soit 1.

1.2.5 Apprentissage profond (Deep Learning)

Les réseaux de neurones modernes sont souvent appelés réseaux de neurones profonds. Le terme «Profond» réfère aux plusieurs cachées situées entre les couches d'entrées acceptant les données à traiter et les couches de sortie destinées à délivrer le résultat du calcul. Même si les réseaux de neurones multicouches existent depuis les années 1980, plusieurs raisons ont empêché l'entrainement efficace des réseaux à plusieurs couches cachées [22, pp.226].

L'un des principaux problèmes est la « malédiction de la dimensionnalité » [22, p. 155]. Quand le nombre de variables augmente le nombre de configurations différentes des variables croit de façon exponentielle. À mesure que le nombre de configurations augmente, le nombre d'échantillons d'entrainement devrait augmenter dans une égale proportion. Collectionner un ensemble de données d'entrainement d'une taille suffisante prend du temps et est coûteux ou carrément impossible. Heureusement, les données du monde réel ne sont pas uniformément distribuées et impliquent souvent une structure, où les informations intéressantes se trouvent sur un espace de faible dimension (Low dimensionnal manifold).

Une autre façon d'améliorer la généralisation consiste à supposer qu'il y'a une constance locale [22, p. 157]. Ça signifie qu'on suppose que la fonction que le réseau neuronal apprend à approcher ne devrait pas beaucoup changer dans une petite région.

Au cours des dix dernières années, les réseaux de neurones ont connu une renaissance, principalement en raison de la disponibilité d'ordinateurs plus puissants et d'ensembles de données plus volumineux. Au début des années 2000 on a découvert que les réseaux de

neurones pouvaient être entrainés efficacement en utilisant des unités de traitement graphique, les GPU (Graphics Processing Unit) qui sont plus efficaces que les processeurs traditionnels et offrent une alternative relativement bon marché aux spécialistes du matériel [48]. Aujourd'hui, les chercheurs utilisent généralement des cartes graphiques grand public haut de gamme, telles que NVIDIA Tesla K40 [20]. D'autres avancées plus théoriques incluent le remplacement de l'erreur quadratique moyenne avec des fonctions basées sur l'entropie croisée et remplaçant l'activation sigmoïdale par des unités linéaires rectifiées [22, p. 226].

Avec l'apprentissage profond il y a moins besoin d'un apprentissage automatique « à la main » qui était la solution utilisée précédemment [22, p. 5]. Dans un contexte de détection avec le modèle classique, par exemple, le système comprend une phase de détection conçue manuellement avant la phase proprement dite de l'apprentissage automatique.

L'équivalent du Deep Learning se compose d'un seul réseau neuronal. Les couches inférieures du réseau neuronal apprennent à reconnaître les fonctionnalités de base, qui sont ensuite transmises aux couches supérieures du réseau.

1.3. Vision par ordinateur (Computer Vision)

1.3.1. Aperçu

La vision par ordinateur traite de l'extraction d'informations significatives des images numériques ou des vidéos. Ceci est distinct du traitement d'image simple qui consiste à manipuler les informations visuelles au niveau des pixels.

Les applications de la vision par ordinateur comprennent la classification d'images, la détection visuelle, la reconstruction de scènes 3D à partir d'images 2D, la récupération d'images, la réalité augmentée, la vision industrielle et l'automatisation du trafic [49].

Aujourd'hui, l'apprentissage automatique est une composante nécessaire de nombreux algorithmes de la vision par ordinateur [44]. De tels algorithmes peuvent être décrits comme une combinaison de traitement d'image et d'apprentissage automatique. Les solutions efficaces nécessitent des algorithmes qui peuvent faire face à la grande quantité d'informations contenues dans les images et pour de nombreuses applications critiques effectuer le calcul en temps réel [28].

1.3.2. Détection d'objets

La détection d'objets est l'un des problèmes classiques de la vision par ordinateur et est souvent décrit comme une tâche difficile. À bien des égards, il est similaire à d'autres tâches de vision par ordinateur, car il s'agit de créer une solution invariante à la déformation et aux changements d'éclairage.

Ce qui fait de la détection un problème distinct est qu'il implique à la fois la localisation et la classification des régions d'une image [20]. La tâche de localisation n'est pas nécessaire, par exemple, pour la classification de l'image complète. Pour détecter un objet, nous devons avoir une idée de l'endroit où l'objet pourrait être et comment l'image est segmentée. Cela crée un problème de type poulet-œuf, où, pour reconnaître la forme (et la classe) d'un objet, nous avons besoin de connaître son emplacement et pour reconnaître l'emplacement d'un objet, nous devons connaitre sa forme [53]. Certaines caractéristiques visuellement différentes, comme les vêtements et le visage d'un être humain, peuvent faire partie du même objet, mais c'est difficile pour le savoir sans reconnaître d'abord l'objet. D'un autre côté, certains objets ne ressortant que légèrement de l'arrière-plan, nécessitent une séparation avant la reconnaissance [51].

Les caractéristiques visuelles de bas niveau d'une image, telles qu'une carte de saillance, peuvent être utilisées comme guide pour localiser des objets candidats [53]. L'emplacement et la taille sont généralement définis à l'aide d'un cadre délimitant, qui est stocké sous la forme de coordonnées des coins.

Utiliser un rectangle est plus simple que d'utiliser un polygone de forme arbitraire, d'ailleurs de nombreuses opérations, telles que la convolution, sont effectuées sur des rectangles. La sous-image contenue dans le cadre de sélection est ensuite classifiée par un algorithme conçu par l'apprentissage automatique (Machine-learning) [21]. Ultérieurement, les limites de l'objet peuvent être affinées de manière itérative, après l'estimation faite initialement [49].

Au cours des années 2000, des solutions populaires pour la détection d'objets ont utilisé des descripteurs de caractéristiques tels que la transformée de caractéristique invariante par changement d'échelle SIFT (Scale-Invariant Feature Transform) [38] par David Lowe en 1999 et l'histogramme des gradients orientés HOG (Histogram of Oriented Gradients) [14] popularisé en 2005.

Dans les années 2010, il y a eu une tendance vers l'utilisation des réseaux de neurones convolutifs CNN [21] [20] [40]. Avant l'adoption à grande échelle des CNN, il y avait deux solutions concurrentes pour générer des boîtes délimitantes. Dans la première solution, un ensemble dense des propositions régionales sont générées et la plupart d'entre elles sont rejetées [36]. Ceci implique généralement un détecteur de fenêtre coulissante. Dans la deuxième solution, un ensemble peu dense de cadres délimiteurs est généré à l'aide d'une méthode de proposition de région, telle que la recherche sélective [51]. La combinaison de propositions de régions peu denses avec les réseaux de neurones convolutifs a donné de bons résultats et est actuellement populaire [20].

1.3.3. Réseaux de neurones convolutifs

1.3.3.1. Justification

Le problème de la vision par ordinateur à l'aide des réseaux de neurones traditionnels est que même une image de taille modeste contient une quantité énorme d'informations (voir section 1.2.5).

Une image monochrome 620x480 contient 297 600 pixels. Si l'intensité de chaque pixel de cette image est injectée séparément dans un réseau entièrement connecté, chaque neurone nécessite 297 600 poids. Une image complète HD 1920x1080 nécessiterait 2 073 600 poids. Si les images sont polychromes, la quantité de poids est multipliée par le nombre de canaux de couleur (généralement trois). Ainsi, nous pouvons voir que le nombre total de paramètres libres dans le réseau rapidement devient extrêmement grand à mesure que la taille de l'image augmente. Des modèles trop grands provoquent un surapprentissage (Overfitting) et des performances lentes [11, pp. 9]. En outre, de nombreuses tâches de détection de modèle nécessitent que la solution soit invariante par translation. Il est inefficace d'entraîner les neurones à reconnaître séparément le même motif dans le coin supérieur gauche et dans le coin inférieur droit d'une image. Un réseau de neurones entièrement connecté ne parvient pas à prendre ce type de structure en compte [33].

1.3.3.2. Structure de base

L'idée de base du CNN a été inspirée par un concept en biologie appelé le champ réceptif [19]. Les champs réceptifs sont une caractéristique du cortex visuel animal [29]. Ils

agissent comme des détecteurs sensibles à certains types de stimulus, par exemple, les bords.

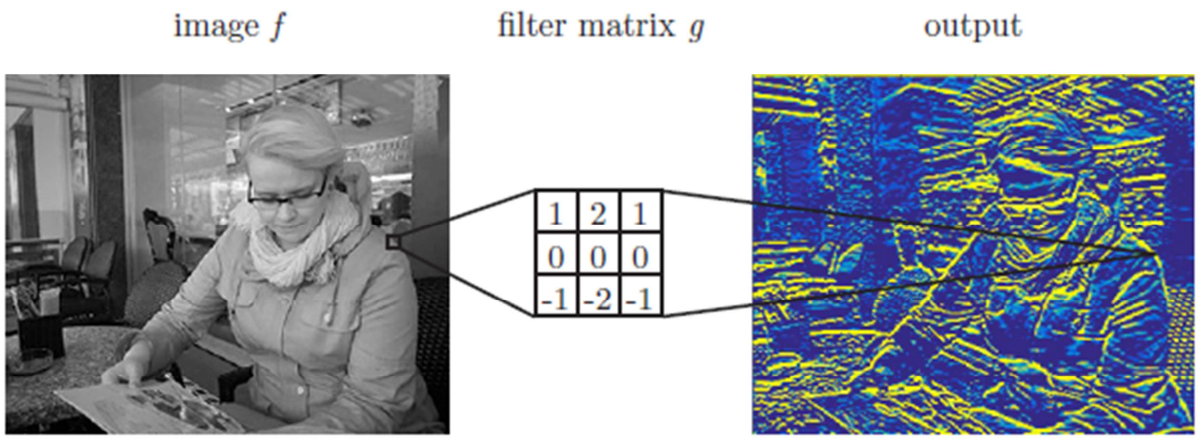


Figure 1.3. Détection des bords horizontaux d'une image
à l'aide du filtrage par convolution.

Cette fonction biologique peut être approchée dans les ordinateurs en utilisant l'opération de convolution [39]. Dans le traitement d'image, les images peuvent être filtrées en utilisant la convolution pour produire différents effets visibles. La figure 1.3 montre comment un filtre convolutif sélectionné à la main détecte les bords horizontaux d'une image, fonctionnant de manière similaire à un champ réceptif.

L'opération de convolution discrète entre une image f et une matrice de filtre g est défini par :

$$h[x,y] = f[x,y] * g[x,y] = \sum_n \sum_m f[n,m]g[x-n, y-m]. \tag{1.3}$$

En effet, le produit scalaire du filtre g et une sous-image de f (avec la même dimensions que g) centrées sur les coordonnées (x, y) produit la valeur de pixel de h aux coordonnées (x,y) [22, pp. 331-332].

La taille du champ réceptif est ajustée par la taille de la matrice du filtre. Aligner le filtre successivement avec chaque sous-image de f produit la matrice de pixels de sortie h. Dans le cas des réseaux de neurones la matrice de sortie est également appelée carte des

caractéristiques (ou carte d'activation après avoir calculé la fonction d'activation) [22, pp. 332].

Les bords doivent être traités comme un cas particulier [22, pp. 349]. Si l'image f n'est pas complétée, la taille de sortie diminue légèrement à chaque convolution. Des filtres convolutifs peuvent être combinés pour former une couche convolutive d'un réseau neuronal [19]. Les valeurs matricielles des filtres sont traitées en tant que paramètres neuronaux et sont entrainés par apprentissage automatique. L'opération de convolution remplace l'opération de multiplication d'une couche d'un réseau neuronal régulier.

La couche de sortie est généralement décrite comme un volume. La hauteur et la largeur du volume dépendent des dimensions de la carte d'activation. La profondeur du volume dépend du nombre de filtres. Étant donné que les mêmes filtres sont utilisés pour toutes les parties de l'image, le nombre des paramètres libres est considérablement réduit par rapport à une couche neuronale entièrement connectée [33].

Les neurones de la couche convolutive partagent pour leur plupart les mêmes paramètres et ne sont connectés qu'à une région locale de l'entrée. Le partage de paramètres résultant de la convolution assure l'invariance par translation [22, pp. 345 -347].

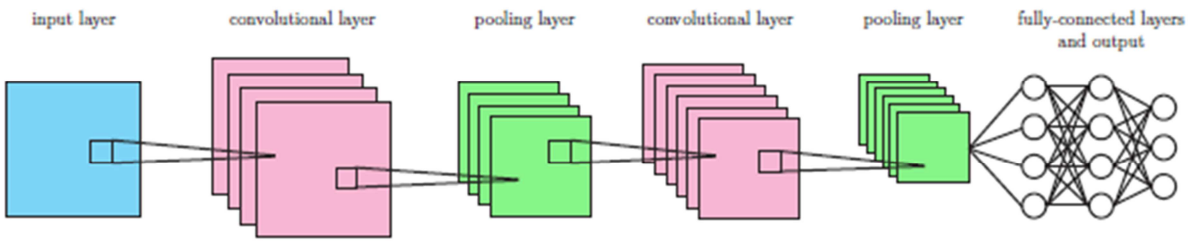


Figure 1.4. Un exemple de réseau convolutif.

Les couches convolutives successives (souvent combinées avec d'autres types de couches, comme la mise en commun décrite ci-dessous) forment un réseau neuronal convolutif (CNN).

Un exemple de réseau convolutif est présenté sur Figure 1.4. L'algorithme de l'entrainement de rétro-propagation, décrit dans la section 1.2.3, est également applicable aux réseaux convolutifs [22, pp. 372]. En théorie, les couches plus proches de l'entrée

doivent apprendre à reconnaître les caractéristiques de bas niveau de l'image, telles que les contours et les coins, et les couches plus proches de la sortie devraient apprendre à combiner ces caractéristiques pour reconnaître des formes plus significatives [19]. Dans ce mémoire, nous sommes intéressés à étudier la capacité des réseaux convolutifs à reconnaître des objets complets.

1.3.3.3. Regroupement et pas (Pooling and Stride)

Pour rendre le réseau plus gérable pour la classification, il est utile de réduire la taille de la carte d'activation dans l'extrémité profonde du réseau. Généralement, les couches profondes du réseau nécessitent moins d'informations sur l'emplacement spatial exact des entités, mais nécessitent plus de filtrage pour reconnaître plusieurs motifs de haut niveau [22, p. 342].

En réduisant la hauteur et la largeur du volume de données, nous pouvons augmenter la profondeur du volume de données et conserver le temps de calcul à un niveau raisonnable.

Il existe deux façons de réduire la taille du volume de données. Une façon est d'inclure une couche de regroupement (Pooling) après une couche convolutive [22, pp. 339-345]. La couche sous-échantillonne efficacement les cartes d'activation. Le pooling permet en plus de rendre le réseau résultant plus invariant par translation en forçant les détecteurs à être moins précis. Cependant, le pooling peut détruire l'iformations sur les relations spatiales entre les sous-parties des modèles. Un exemple typique est le Max-Pooling. Il génère simplement la valeur maximale dans un voisinage rectangulaire de la carte d'activation [22, pp. 339-42].

Une autre façon de réduire la taille du volume de données consiste à ajuster le pas (Stride) de l'opération de convolution. Ce paramètre contrôle si la sortie de convolution est calculée pour un voisinage centré sur chaque pixel de l'image d'entrée (Stride = 1) ou pour chaque nième pixel (Stride = n) [13].

La recherche a montré que les couches de pooling peuvent souvent être écartées sans perte en précision en utilisant des couches convolutives avec un pas élevé [46].

1.3.3.4. Couches supplémentaires

La couche convolutive comprend généralement une fonction d'activation non linéaire, telle qu'une fonction d'activation linéaire rectifiée (voir la sous-section 1.2.4). Les activations sont parfois décrites comme une couche séparée entre les couche de convolution et la couche de pooling. Certains systèmes,

implémentent également une couche appelée normalisation de réponse locale, qui est utilisée comme technique de régularisation. La normalisation de réponse locale imite une fonction des neurones biologiques appelée inhibition latérale, qui fait que les neurones excités diminuent l'activité des voisins les neurones. Cependant, d'autres techniques de régularisation sont actuellement plus populaires et ceux-ci sont discutés dans la section suivante. Les couches cachées finales d'un CNN sont généralement des couches entièrement connectées [11, pp. 269] [40].

Une couche entièrement connectée peut capturer des relations intéressantes que les couches convolutives partageant des paramètres ne le peuvent pas. Cependant, une couche nécessite une taille de volume de données suffisamment petite pour qu'elle soit pratique.

Les paramètres de pooling et de stride peuvent être utilisés pour réduire la taille du volume de données qui atteignent les couches entièrement connectées. Un réseau convolutif qui n'inclut pas de couches entièrement connectées, est appelé un réseau pleinement convolutif FCN (Fully Convolutional Network) [40]. Si le réseau est utilisé pour la classification, il comprend généralement une couche de sortie softmax [11, pp. 269].

Les activations des couches supérieures peuvent également être utilisées directement pour générer une représentation de caractéristiques d'une image. Cela signifie que le réseau convolutif est utilisé comme un grand détecteur de caractéristique [33].

1.3.4. Régularisation et augmentation des données

La régularisation fait référence aux méthodes utilisées pour réduire le sur-apprentissage en introduisant des contraintes ou informations supplémentaires au système d'apprentissage automatique [22, pp. 228-228]. Une manière classique d'utiliser la régularisation dans les réseaux de neurones ajoute un terme de pénalité à la fonction objectif/perte qui pénalise certains types de poids.

La fonction de partage de paramètres des réseaux convolutifs est un autre exemple de régularisation. Il existe plusieurs techniques de régularisation spécifiques aux réseaux de neurones profonds. Une technique populaire appelée décrochage ou abandon [47] tente de réduire la co-adaptation des neurones. Ceci est réalisé en abandonnant au hasard les neurones pendant l'entraînement, ce qui signifie qu'un réseau neuronal légèrement différent est utilisé pour chaque échantillon de formation ou mini lots (Minibatch). Cela empêche le système d'être trop dépendant d'un seul neurone ou d'une connexion et fournit un moyen efficace et peu coûteux de mettre en œuvre la régularisation [22, pp. 258-259]. Dans les réseaux convolutifs, l'abandon est généralement utilisé dans les couches finales entièrement connectées [45].

Le sur-apprentissage peut également être réduit en augmentant la quantité de données d'entraînement. Lorsqu'il n'est pas possible d'acquérir des échantillons plus réels, l'augmentation des données (Data augmentation) est utilisée pour générer plus d'échantillons à partir des données existantes [22, pp. 240-241]. Pour la classification utilisant des réseaux convolutifs, cela peut être réalisé en calculant des transformations des images d'entrée qui n'altèrent pas la perception des classes d'objets, mais fournissent un défi supplémentaire au système. Les images peuvent être, par exemple inversées, subir des rotations ou être sous-échantillonnées avec différents recadrages (Rognements) et échelles. De plus, du bruit peut être ajouté aux images d'entrée [22, p. 242].

1.4.6. Développement

Les réseaux de neurones convolutifs ont été parmi les premiers réseaux de neurones profonds réussis. Le Néocognitron, développé par Fukushima dans les années 1980, a fourni un modèle de réseau neuronal pour la reconnaissance d'objets invariants par translation, inspiré de la biologie [19]. Le Cun et coll. ont combiné cette méthode avec un algorithme d'apprentissage, en l'occurrence, la rétro-propagation [33]. Ces premières solutions étaient principalement utilisées pour la reconnaissance de caractères.

Après avoir fourni des résultats prometteurs, l'importance des méthodes des réseaux de neurones s'est estompée et ils ont été pour leur plupart remplacés par des machines à vecteurs de support (SVM) [21]. Par la suite en 2012, Krizhevsky et al. [32] ont obtenu d'excellents résultats sur l'ensemble de données du concours annuel ILSVRC (Imagenet Large Scale Visual Recognition Challenge) en combinant la méthode de Le Cun avec des

méthodes récentes de réglages fins pour l'apprentissage profond (Fine-tuning methods for deep learning). Ces résultats ont popularisé les CNN et ont conduit au développement de nouvelles méthodes de détection d'objets [21].

Au concours ImageNet Challenge de 2014, Simonyan et Zisserman [45] ont exploré l'effet d'augmenter la profondeur d'un réseau convolutif sur la localisation et la précision de la classification. L'équipe a obtenu des résultats qui ont amélioré l'état de l'art en utilisant des réseaux convolutifs de 16 et 19 couches de profondeur respectivement. L'architecture 16 couches comprend 13 couches convolutives (avec 3x3 filtres), 5 couches de pooling (2x2 voisinage max-pooling) et 3 couches entièrement connectées. Toutes les couches cachées utilisent des activations rectifiées (ReLu).

Les couches entièrement connectées réduisent 4096 canaux à 1000 sorties softmax et sont régularisées en utilisant la méthode de l'abandon (Dropout). Cette forme de réseau est appelée VGG-16. En 2016, le gagnant de la catégorie détection d'objets dans le défi ImageNet a également été basé sur les CNN [2]. La méthode est basée sur une génération de proposition de région CRAFT (Cascade Region-proposal-network And Fast Tr-cnn) [55] [56].

Conclusion

Étant donné que les réseaux profonds ont été à l'origine d'une grande partie de l'amélioration de détection d'objets au cours de la présente décennie, il est pertinent de se demander si le les réseaux pourraient être approfondis encore. Pour les réseaux de type VGG, il a été constaté expérimentalement que, pour un problème donné, il existe un certain nombre de couches, après quoi les erreurs d'entraînement et de test commencent à augmenter. Ceci est appelé le problème de la dégradation.

Les raisons de ce comportement ne sont actuellement pas entièrement connues. Dans le passé, l'algorithme de rétro-propagation souffrait de la disparition des gradients, mais ce problème a été atténué en remplaçant les fonctions d'activation sigmoïdales par des unités linéaires rectifiées. Il a été spéculé que les réseaux très profonds ne convergent pas dans un délai raisonnables.

L'une des méthodes développées pour permettre la formation de réseaux plus profonds est une technique de régularisation appelée normalisation par lots. Le but de La méthode

consiste à maintenir la distribution des entrées de couche stable pendant l'entraînement. En général, il est avantageux de prétraiter l'entrée du réseau neuronal pour avoir zéro variance, et cela se fait presque toujours. Dans la normalisation par lots, au lieu d'effectuer une normalisation une fois au début, elle est aussi effectuée pendant l'entraînement et entre les couches.

Une autre méthode pour créer des réseaux très profonds consiste à utiliser un réseau résiduel, également appelé ResNet. Dans un réseau résiduel, certaines couches ne fonctionnent pas directement sur l'entrée de la couche précédente. Plutôt, l'entrée passe inchangée sur plusieurs couches, et les couches opèrent sur le résidu de l'entrée passée et de la sortie de la couche résiduelle précédente. Une des raisons qui font la force des réseaux résiduels est qu'ils peuvent être formés en utilisant des algorithmes et outils d'entrainement.

Chapitre 2

Les réseaux de neurones convolutifs

Introduction

Dans ce chapitre, nous discutons et comparons différents types de réseaux de neurones convolutifs. En particulier, nous allons examiner les méthodes qui combinent les CNN avec la classification des propositions de régions. Nous discutons également comment les propositions de région, également appelées régions d'intérêt RoI (Regions of Interest) sont générées.

2.1. Les R-CNN

En 2012, Krizhevsky et al. [32] ont obtenu des résultats prometteurs avec les CNN pour la tâche générale de classification des images, comme mentionné dans la section 1.4.6 En 2013, Girshick et coll. [21] ont publié une méthode généralisant ces résultats pour la détection d'objet. Cette méthode s'appelle CNN avec proposition de région R-CNN (Region-based Convolutional Neural Network).

2.1.1. Description générale

Le calcul direct des R-CNN comporte plusieurs étapes, illustrées par Figure 2.1. Les premières régions d'intérêt sont générées. Les régions d'intérêt (RoI) sont indépendantes

des boites délimitantes qui ont une forte probabilité de contenir un objet intéressant. Dans le document, une méthode distincte appelée recherche sélective [52], est utilisée pour les générer, mais d'autres méthodes de génération de régions peuvent être utilisées à la place.

La recherche sélective, avec d'autres techniques de génération de proposition de région, est discutée plus en détail dans la section 2.3.

Par la suite, un réseau convolutif est utilisé pour extraire les caractéristiques de chaque région proposée. La sous-image contenue dans la boîte délimitante (bounding box) est déformée pour correspondre à la taille d'entrée du CNN et ensuite injectée au réseau. Une fois que le réseau a extrait les caractéristiques à partir de l'entrée, ces caractéristiques deviennent des entrées aux SVM (Support Vector Machines) pour qu'ils fournissent la classification finale.

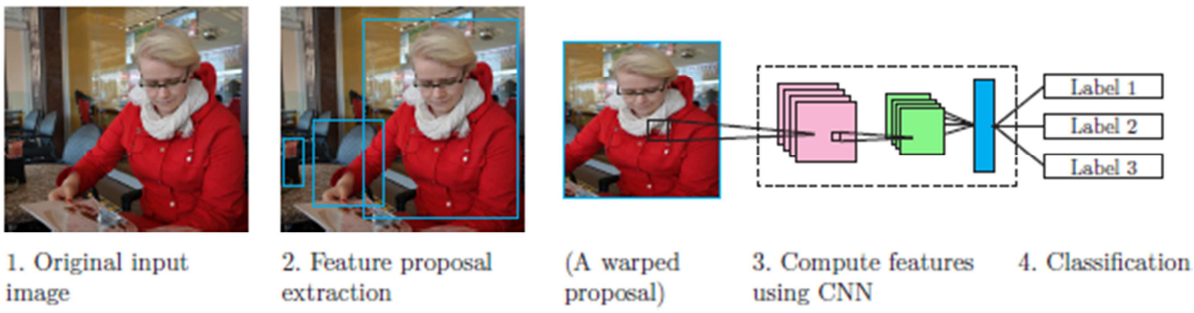


Figure 2.1. Étapes du calcul direct R-CNN.

La méthode est entraînée en plusieurs étapes, en commençant par le réseau convolutif [20]. Une fois que le CNN a été entrainé, les SVM sont adaptées aux caractéristiques CNN. Enfin, la méthode de génération de proposition de région est entrainée.

2.1.2 Inconvénients

La technique R-CNN est une méthode importante, car elle a fourni la première solution pratique pour la détection d'objets à l'aide des CNN. Étant la première, elle présente de nombreux inconvénients qui ont été améliorées par des méthodes ultérieures.

Dans son article de 2015 pour Fast R-CNN [20], Girshick énumère trois problèmes principaux de la R-CNN:

- Premièrement, la conception du réseau comprend plusieurs étapes, comme décrit ci-dessus.
- En second lieux, l'entrainement coûte cher. Pour la génération de proposition de région comme pour les SVM, les caractéristiques sont extraites de chaque proposition de région et stockées sur disque. Cela nécessite des journées de calcul et des centaines de gigaoctets d'espace de stockage.
- Troisièmement, et c'est le point le plus important, la détection des objets est lente et nécessite presque une minute pour chaque image, même sur un GPU. Ceci est dû au fait que le calcul CNN direct est effectué séparément pour chaque proposition d'objet, même si les propositions proviennent de la même image ou se chevauchent.

2.2. R-CNN rapide (Fast R-CNN)

Publiée en 2015 par Girshick [20], cette méthode permet de faire une reconnaissance d'objets. L'idée principale est d'effectuer la passe avant du CNN pour l'image entière, au lieu de l'exécuter séparément pour chaque région d'intérêt (RoI).

2.2.1. Description générale

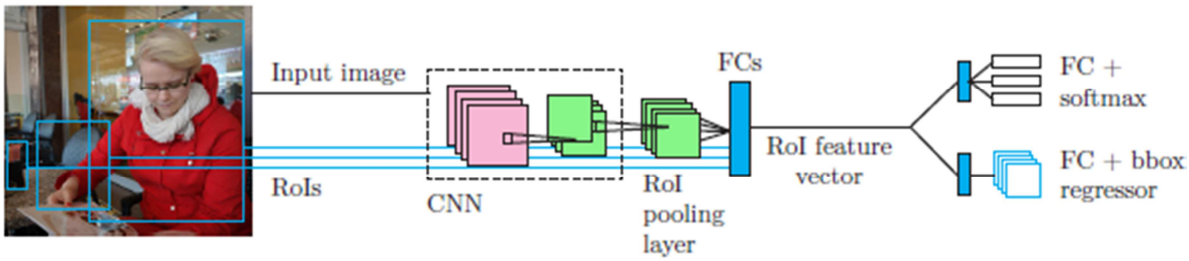


Figure 2.2. Étapes du calcul direct R-CNN rapide.

La structure générale des Fast R-CNN est illustrée par Figure 2.2. Le réseau reçoit en entrée une image plus des régions d'intérêt calculées à partir de l'image. Comme dans les R-CNN, les RoI sont généreés à l'aide d'une méthode externe.

L'image est traitée à l'aide d'un réseau CNN qui comprend plusieurs couches de regroupement convolutif et des couches de max pooling. La carte de caractéristiques convolutive qui est générée après ces couches devient une entrée pour une couche pooling RoI. Cela extrait un vecteur de caractéristiques de longueur fixe pour chaque RoI de la carte de caractéristiques.

Les vecteurs de caractéristiques sont les entrées des couches entièrement connectées qui elles sont connectées à deux couches de sortie: une couche softmax qui produit des estimations de probabilité pour les classes d'objets et une couche à valeur réelle qui produit des coordonnées d'une boîte englobante calculées par régression (ce qui signifie qu'il y'a raffinement des boîtes candidats initiales).

2.2.2. Performances de classification

Selon les auteurs, Fast R-CNN offre un temps de classification significativement plus court que le R-CNN ordinaire, prenant moins d'une seconde sur un GPU de pointe [20]. Cela est principalement dû à l'utilisation de la même carte des caractéristiques pour chaque RoI. À mesure que le temps de détection diminue, le temps de calcul global commence à dépendre fortement des performances de la génération de proposition de région méthode. La génération RoI peut ainsi former un goulot d'étranglement de calcul [40].

De plus, lorsqu'il existe de nombreux RoI, le temps consacré à l'évaluation les couches entièrement connectées peuvent dominer le temps d'évaluation des couches convolutives. Le temps de classement peut être accéléré d'environ 30% si le les couches entièrement connectées sont compressées à l'aide d'une décomposition en valeurs singulières tronquées [20]. Cela se traduit cependant par une légère diminution de la précision.

2.2.3. Entrainement

Selon les auteurs de la publication [20], Fast R-CNN est plus efficace pour entrainer que R-CNN, avec une réduction de neuf fois du temps d'entrainement. L'ensemble réseau (y compris la couche de pooling RoI et les couches entièrement connectées) peuvent être entrainé en utilisant l'algorithme de rétro-propagation et la descente de gradient stochastique. En règle générale, un réseau pré-entrainé est utilisé comme point de départ, puis affiné.

L'entrainement se fait par mini-lots de N images. R/N RoIs sont échantillonnées à partir de chaque image de mini-lot. Les échantillons RoI sont affectés à une classe, si leur intersection avec un Ground-Truth box est supérieure à 0,5. Les autres RoIs appartiennent à la classe d'arrière-plan.

Comme dans la classification, les RoI de la même image partagent le calcul et utilisation de la mémoire. Pour l'augmentation des données, l'image d'origine est inversée horizontalement.

Le classificateurr softmax et le regresseur de la boîte englobante sont ajustés ensemble à l'aide d'une fonction de perte multi-tâches, qui prend en compte à la fois la vraie classe de la RoI échantillonnée et l'écart entre l'échantillon boîte englobante et la vraie boîte délimitante.

2.3. Génération et utilisation des propositions régionales

Nous discutons dans cette section le principe le principe général de la génération des ROI et examinons de près deux méthodes populaires : la recherche sélective et les boites de contours.

2.3.1. Aperçu

Le but de la génération de proposition de région dans la détection d'objets est de générer suffisamment de régions pour que tous les vrais objets soient récupérés [57]. Le générateur est moins soucieux de la précision, car c'est la tâche du détecteur d'objet d'identifier les régions correctes à partir de la sortie du générateur de proposition de région. Cependant, le nombre de propositions générées a de l'impact sur la performance.

Comme mentionné dans la section 1.3.2, il existe deux approches principales de la génération de régions : génération d'ensembles dense et génération d'ensembles clairsemés ou peu denses.

Les solutions d'ensembles denses tentent de générer par force brute un ensemble exhaustif des boîtes englobantes qui incluent chaque emplacement d'objet potentiel [51]. Ceci peut être réalisé en faisant glisser une fenêtre de détection sur l'image. Cependant, la recherche dans chaque emplacement de l'image est coûteuse en calcul et nécessite un détecteur d'objet rapide. En outre, différentes formes et tailles de fenêtres doivent être prises en

compte. Ainsi, la plupart des méthodes de fenêtre glissante limitent la quantité d'objets candidats en utilisant une taille de pas grossière et un nombre limité de facteurs de forme fixes.

La plupart des propositions de région dans un ensemble dense ne contiennent pas d'objets intéressants. Ces propositions doivent être écartées après la phase de détection d'objet. Les résultats de détection peuvent être rejetés s'ils se situent en deçà d'un seuil de confiance prédéfini ou si leur valeur de confiance est inférieure à un maximum local (Suppression non maximale) [36].

Au lieu de rejeter les régions après l'étape de détection d'objet, le générateur de proposition de région lui-même peut classer les régions d'une manière indépendante de la classe et rejeter les régions de bas classement. Cela génère un ensemble peu dense d'objets détectés [55]. D'une manière similaire aux méthodes denses, le seuillage et la suppression des non maxima peuvent être mis en œuvre après la phase de détection pour améliorer encore la qualité de détection.

Les solutions des ensembles peu denses peuvent être supervisées ou non supervisées. L'une des méthodes non supervisées les plus populaires est la recherche sélective [51] (voir section 3.3.2), qui utilise une fusion itérative de superpixels. Il y'a également d'autres méthodes qui utilisent la même approche [57].

Une autre approche est de classer le score d'objet (Objectness) d'une fenêtre glissante. Un exemple populaire de ceci est Edge Boxes [57] (voir section 3.3.3), qui calcule le score d'objet en calculant le nombre d'arêtes dans une boîte englobante et en soustrayant le nombre d'arêtes qui chevauchent la limite de la boîte. Il y a aussi un troisième groupe de méthodes basées sur la segmentation [57]. Les méthodes supervisées traitent la génération de proposition de région comme une classification ou un problème de régression. Cela signifie utiliser un algorithme d'apprentissage automatique, tel que les machines à vecteurs de support S [55]. Il est également possible d'utiliser un réseau convolutif pour générer les régions d'intérêt.

Un exemple d'utilisation d'un CNN pour le calcul des boites englobantes est MultiBox [16]. Certaines méthodes avancées de détection d'objets, telles que Faster R-CNN [40] décrit en 2.4.1, utilisent des parties du même réseau convolutif à la fois pour générer les propositions de région et pour la détection. Ces méthodes sont dites méthodes intégrées.

2.3.2. Recherche sélective

La recherche sélective [51] utilise un partitionnement hiérarchique d'une image pour créer un ensemble restreint d'emplacements d'objets. La principale philosophie de conception est de ne pas utiliser une stratégie unique, mais de combiner les meilleures caractéristiques de la segmentation ascendante et une recherche exhaustive.

Les auteurs avaient trois considérations de conception principales :

- la recherche doit saisir toutes les échelles, être diversifiée, c'est-à-dire ne pas utiliser une seule stratégie pour regrouper des régions et être rapide à calculer. L'algorithme commence par créer un ensemble de petites régions initiales à l'aide d'une méthode appelée Segmentation d'image basée sur le graphe [18] conçue par Felzenszwalb et Huttenlocher. La méthode crée un ensemble de régions appelées superpixels.
- Combinés, les superpixels s'étendent sur l'image entière, mais individuellement, ils ne doivent pas s'étendre sur des objets différents. La recherche sélective se poursuit ensuite en regroupant de manière itérative les régions ensemble en utilisant un algorithme glouton, en commençant par les deux régions les plus similaires.
- De nombreuses mesures complémentaires sont utilisées pour calculer la similitude. Ces mesures prennent en compte la similitude des couleurs (en calculant un histogramme des couleurs), la similarité de texture (en calculant une mesure de type SIFT), taille des régions (Les petites régions devraient être fusionnées plus tôt) et dans quelle mesure les régions s'assemblent(les sauts devraient être évités). La phase de regroupement se termine lorsque chaque région a été combinée.

Les emplacements des objets hypothétiques ainsi générés sont ensuite classés par probabilité que l'emplacement contienne un objet. En pratique, les emplacements sont classés en fonction de l'ordre dans lequel ils ont été regroupés par les différentes mesures. Un certain élément aléatoire est ajouté pour éviter que les gros objets soient trop favorisés. Les emplacements classés aux rangs inférieurs sont supprimés.

La méthode de génération de région et les mesures de similarité ont été sélectionnées car elles sont rapides à calculer, ce qui rend la méthode rapide en général. En outre à l'aide de diverses mesures de similitude, la recherche peut être diversifiée davantage en utilisant des espaces colorimétriques complémentaires (pour garantir l'invariance à l'éclairage) et en utilisant des régions de départ complémentaires.

2.3.3. Boîtes de contours (Edge boxes)

Comme son nom l'indique, cette méthode [57] est basée sur la détection d'objets à partir de cartes de bords. La principale contribution des auteurs de la méthode est l'observation que le nombre de contours entièrement délimités par une boîte est corrélé avec la probabilité que la boîte contienne un objet.

Tout d'abord, la carte des bords est calculée à l'aide d'une méthode des mêmes auteurs appelé « Détecteur de bords structurés » [15]. Ensuite, les lignes de bord épaisses sont amincies utilisant une suppression non maximale. Au lieu d'opérer sur les pixels de bord directement, les pixels sont regroupés à l'aide d'un algorithme glouton. Une mesure d'affinité est conçue pour calculer si les groupes d'arêtes font partie du même contour.

Les propositions de région sont trouvées en scannant l'image en utilisant la traditionnelle méthode de fenêtre glissante et calcul d'un score d'objet à chaque position, rapport de forme et échelle.

2.4. Détection d'objets convolutifs avancés

Au-delà des Fast RCNN il existe plusieurs algorithmes de pointe avec un temps de calcul ou précision meilleurs. Nous décrivons deux de ces algorithmes.

2.4.1. Faster R-CNN (R-CNN plus rapide)

Faster R-CNN [40] par Ren et al. est une méthode intégrée. L'idée principale consiste à utiliser des couches convolutives partagées pour la génération de proposition de région et pour la détection. Les auteurs ont découvert que les cartes de caractéristiques générées par objet des réseaux de détection peuvent également être utilisées pour générer les propositions de région. La partie entièrement convolutive du réseau Faster R-CNN qui génère les propositions de fonctionnalités est appelée réseau de proposition de région (RPN).

Les auteurs ont utilisé l'architecture Fast R-CNN pour le réseau de détection. Un réseau R-CNN plus rapide est formé en alternant les formations pour génération et détection des RoI. Premièrement, deux réseaux distincts sont formés. Ensuite, ces réseaux sont combinés

et ajustés. Pendant le raffinement, certaines de ces couches sont maintenues fixes et certaines couches sont entraînées à leur tour. Le réseau formé reçoit une seule image en entrée.

Le réseau entrainé reçoit une seule image en entrée. Les couches convolutives pleinement partagées génèrent des cartes de caractéristiques à partir de l'image. Ces cartes de caractéristiques sont transmises au RPN. Le RPN produit des propositions de région, qui sont entrée, conjointement avec les dites cartes de caractéristiques, aux couches de détection finales. Ces couches incluent une couche de pooling RoI et produisent les classifications finales.

En utilisant des couches convolutives partagées, le calcul des propositions de région ne coute presque rien. Le calcul des propositions de région sur un CNN a en plus l'avantage d'être réalisable sur un GPU. Les méthodes traditionnelles de génération de RoI, telles que la recherche sélective, sont implémentées à l'aide d'un processeur CPU.

Pour traiter différentes formes et tailles de fenêtre de détection, La méthode utilise des boîtes d'ancrage spéciales au lieu d'utiliser une pyramide d'images mises échelonnées ou une pyramide de tailles de filtre différentes. Des boîtes spéciales fonctionnent comme des points de référence pour différentes propositions de région centrées sur le même pixel.

2.4.2. SSD

Le détecteur MultiBox Single Shot [36] (SSD) emmène la détection intégrée même plus loin. La méthode ne génère pas du tout des propositions, ni implique aucun ré-échantillonnage des segments d'image. Il génère des détections d'objets en utilisant un seul passage d'un réseau convolutif.

Ressemblant un peu à une méthode de fenêtre glissante, l'algorithme commence avec un ensemble par défaut de boites englobantes. Ceux-ci incluent différents rapports de forme et échelles. Les prédictions d'objet calculées pour ces boîtes incluent des paramètres d'offset qui prédisent de combien la zone correcte de délimitation entourant l'objet diffère de la boîte par défaut.

L'algorithme traite différentes échelles en utilisant des cartes de caractéristiques de plusieurs couches convolutives différentes (c'est-à-dire des cartes de caractéristiques plus grandes et plus petites) comme entrée dans le classifieur.

Puisque la méthode génère un ensemble dense de boites englobantes, le classifieur est suivi d'une étape de suppression des non maxima qui élimine la plupart des cases en dessous d'un certain seuil de confiance.

2.5. Comparaison des méthodes

Ci-dessus, nous avons décrit comment les Fast R-CNN sont plus rapides et plus précis que les R-CNN réguliers. Mais comment Fast R-CNN fonctionnent-t-ils par rapport à ce qui méthodes avancées précédentes?

Liu et coll [36] ont comparé les performances de Fast R-CNN, Faster R-CNN et SSD sur l'ensemble de test PASCAL VOC 2007. Lors de l'utilisation de réseaux entrainés sur les données d'entraînement PASCAL VOC 2007, Fast R-CNN a atteint une précision moyenne mAP (mean average precision) de 66,9. Faster R-CNN a fait légèrement mieux, avec un mAP de 69,9. SSD a atteint un mAP de 68,0 avec une taille d'entrée 300 x 300 et 71,6 avec une taille d'entrée 512 x 512.

Les implémentations standard de Fast R-CNN et Faster R-CNN utilisent 600 comme longueur de la dimension la plus courte de l'image d'entrée, le SSD semble fonctionner mieux avec des images de taille similaire. Cependant, le SSD nécessite une utilisation intensive de l'augmentation des données pour atteindre ce résultat [36]. Fast R-CNN et Faster RCNN utilisent uniquement l'inversement horizontal, et il est actuellement inconnu, s'ils bénéficieraient d'une augmentation supplémentaire.

Alors que les méthodes avancées sont plus précises que Fast R-CNN, le vrai les améliorations viennent de la vitesse. Lorsque la plupart des détections à faible probabilité sont éliminées en utilisant le seuillage et la suppression des non maxima, SSD512 peut fonctionner à 19 FPS sur un Titan X GPU. Pendant ce temps, Faster R-CNN avec une architecture VGG-16 fonctionne à 7 FPS. [36]

Les auteurs originaux de Faster R-CNN [40] rapportent un temps d'exécution de 5 FPS soit 0,2 s par image. Fast R-CNN a approximativement la même évaluation vitesse, mais nécessite un temps supplémentaire pour calculer les propositions de région. Le temps de génération de la région dépend de la méthode, la recherche sélective nécessitant 2 secondes par image sur un processeur CPU et des boîtiers Edge (Edge boxes) nécessitant 0,2 seconde par image [40].

Conclusion

Nous avons commencé le mémoire par une revue du contexte théorique. Nous avons expliqué comment fonctionnent les réseaux de neurones et ce qu'implique la détection d'objets. Nous avons démontré pourquoi les réseaux de neurones réguliers sont insuffisants et comment les réseaux convolutifs invariants par translation fournissent une solution à de nombreux problèmes de vision par ordinateur.

Ensuite, nous avons montré comment la détection d'objets convolutive a évolué du CNN relativement lent aux méthodes optimisées actuelles. Ce développement n'est généralement pas lié à la structure du réseau convolutif lui-même. Il est plutôt lié à la façon dont le réseau convolutif est utilisé et au calcul qui a lieu avant et après le réseau convolutif.

Dans les méthodes précédentes, il y avait beaucoup plus de phases distinctes impliquant prétraitement, génération de régions, calcul des couches entièrement connectées et la classification finale. Dans les dernières méthodes, ces phases ont été de plus en plus intégrées dans le réseau convolutif lui-même, tout en conservant le modèle de base CNN intact. En revanche, le lauréat 2016 du ImageNet challenge est à nouveau un modèle composé de nombreux composants séparés. Néanmoins, plusieurs goulots d'étranglement de calcul ont disparu.

Chapitre 3

Conception et implémentation

Introduction

La principale tâche de la partie expérimentale était de mettre en œuvre un détecteur d'objet à base d'un réseau de neurones convolutif profond CNN.

Les problèmes majeurs limitant l'utilisation des méthodes d'apprentissage en profondeur sont la capacité de calcul et les données d'entraînement. Pour ce mémoire nous n'avions pas accès à des serveurs ou à un GPU haut de gamme spécialisés pour les fins de recherche. Nous devions nous suffir d'implémenter les méthodes sur un ordinateur portable grand public. Former un réseau convolutif du début à la fin sur un tel matériel prendrait énormément de temps. C'est pourquoi que nous avons favorisé au début l'exploitation de réseaux pré-entrainés et leur utilisation comme point de départ. Nous avons également privilégié les méthodes qui avaient des implémentations disponibles sur MATLAB puisque c'est un outil que nous maitrisons plus ou moins.

3.1. Configuration expérimentale

3.1.1. Matériel

Le matériel réalisé est un laptop Sony Vaoi Intel® Pentium® CPU 2117U @ 1.80GHz avec 4Go de mémoire RAM, et un système d'exploitations Windows 10 édition professionnelle, service pack 1, type système 64 bits.

3.1.1. Environnement de programmation

Nous avons dans un premier temps été tentés de programmer avec Python car la plupart des méthodes sur les CNN ont été développées avec. Mais nous avons fini par opter pour Matlab car c'est un outil assez maitrisé au cours de notre formation et il fournit toutes les fonctionnalités requises.

La version utilisée est MATLAB 2018b. Nous avons implémenté le réseau convolutif en utilisant MatConvNet, qui est une boîte à outils MATLAB facile à installer et spécialement développée à cet effet. C'est une collection de fonctions MATLAB qui implémentent différents blocs de construction d'un réseau convolutif.

3.2. Approche expérimentale

Dans un premier temps et pour s'initier aux CNN nous avons commencé par tester nos images sur un réseau pré-entrainé. Nous avons fini par opter pour AlexNet. C'est un réseau neuronal convolutif de 8 couches de profondeur. On peut charger une version pré-entraînée du réseau entrainée sur la base de données ImageNet [1]. En conséquence, le réseau a appris de riches représentations de caractéristiques pour une large gamme d'images. Le réseau admet en entrée des images de taille 227 x 227.

Dans un deuxième temps nous avons construit notre propre réseau que nous avons entrainé sur notre base de données dont la description est donnée dans la section suivante.

3.3.Bases de données

3.3.1. La base ImageNet

Nous avons utilisé la base ImageNet [1] au début pour tester quelques échantillons à titre comparatif. La base de données ImageNet contient plus d'un million d'images réparties en 1000 catégories d'objets différents tels que le clavier, la souris, le crayon et de nombreux animaux (cf. Figure 3.1).

Figure 3.1. Quelques échantillons de la base de données ImageNet

3.3.2. Notre base

Pour entrainer notre réseau de neurones convolutif nous aurions pu utiliser une grande base d'images à accès libre comme ImageNet. Cela aurait rendu notre réseau plus riche. Mais ce n'est point une tâche facile à cause du matériel dont nous disposons. Le temps nécessaire pour ça est énorme sans parler des risques de plantages ou d'échauffement du matériel.

Pour y remédier nous avons créé notre propre base de données. Elle contient un total de 2537 images réparties en 10 classes se rapportant à 10 catégories d'objets et baptisées respectivement : ipod, cap, wall clock , headphone, vase, cellular phone, cowboy hat, modem, lotion, sunglasses. Ces classes correspondent à des catégories qui sont incluses dans la base ImageNet.

Certaines images représentent le même objet mais elles sont prises à des angles et positions différents. Des exemples sont illustrés par Figure 3.2 et Figure 3.3 ci-dessous.

Figure 3.2. Echantillons de la classe ‘cap’

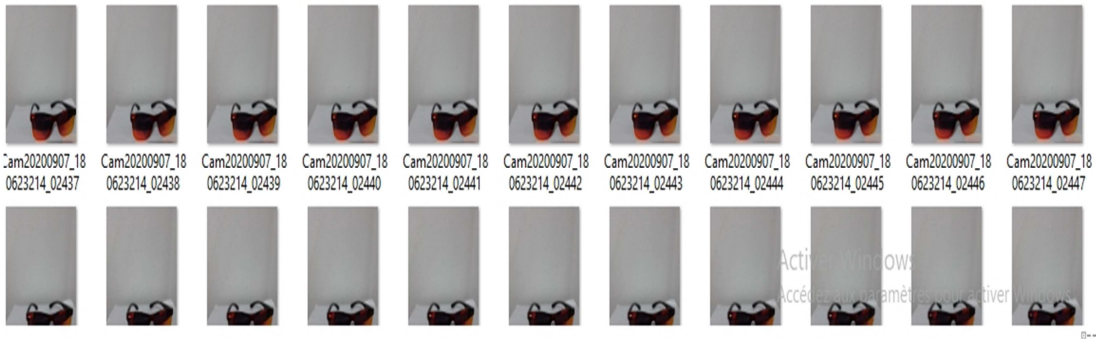


Figure 3.3. Echantillons de la classe ‘sunglasses’

3.4. Notre réseau

Le réseau que nous avons construit possède en plus de la couche d’entrée et de sortie quatre couches de convolution, deux couches de maxpooling et deux couches fully connected.

La couche d’entrée reçoit une image de taille 32*32, l’image passe d’abord par la première couche de convolution (Fonction Matlab ‘convolution2dLayer’). Cette couche est composée de 32 filtres de taille 3*3 et un padding de valeur 1. La fonction d’activation ReLU (Fonction Matlab ‘reluLayer’) est utilisée pour forcer les neurones à retourner des valeurs positives.

Les trois couches suivantes sont identiques à la première (32 filtres de taille 3*3, un padding de 1 et suivie d’une fonction d’activation . ReLU).

Un Maxpooling (Fonction Matlab ’maxPooling2dLayer ’) de fenêtre 3*3 et un stride de valeur 2 est appliqué après pour réduire la taille de l’image.

Après ces quatre couches de convolution, nous utilisons un réseau de neurones composé de deux couches fully connected (Fonction Matlab ‘fullyConnectedLayer’). La première couche est composée de 10 neurones où la fonction d’activation utilisée est une ReLU, la deuxième couche utilise la fonction d’activation softmax (Fonction Matlab ‘softmaxLayer’) qui permet de calculer la distribution de probabilité des 10 classes. Et enfin une couche de sortie de classification (Fonction Matlab ‘classificationLayer’). La configuration des couches créées précédemment décrites est illustrée par Figure 3.4 ci-dessous.

```
>> layers

layers =

  16x1 Layer array with layers:

     1   ''   Image Input             32x32x3 images with 'zerocenter' normalization
     2   ''   Convolution             32 3x3 convolutions with stride [1  1] and padding [1  1  1  1]
     3   ''   ReLU                    ReLU
     4   ''   Convolution             32 3x3 convolutions with stride [1  1] and padding [1  1  1  1]
     5   ''   ReLU                    ReLU
     6   ''   Convolution             32 3x3 convolutions with stride [1  1] and padding [1  1  1  1]
     7   ''   ReLU                    ReLU
     8   ''   Convolution             32 3x3 convolutions with stride [1  1] and padding [1  1  1  1]
     9   ''   ReLU                    ReLU
    10   ''   Max Pooling             3x3 max pooling with stride [2  2] and padding [0  0  0  0]
    11   ''   Max Pooling             3x3 max pooling with stride [2  2] and padding [0  0  0  0]
    12   ''   Fully Connected         1 fully connected layer
    13   ''   ReLU                    ReLU
    14   ''   Fully Connected         1 fully connected layer
    15   ''   Softmax                 softmax
    16   ''   Classification Output   crossentropyex
>>
```

Figure 3.4.Configuration du réseau

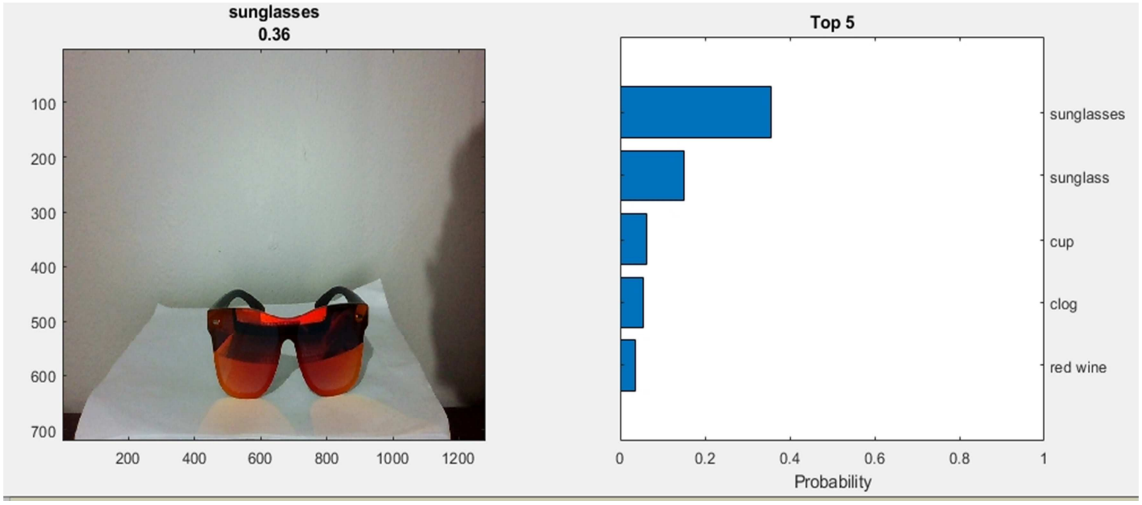
3.5. Tests et résultats

3.5.1. Tests avec le réseau pré-entrainé AlexNet

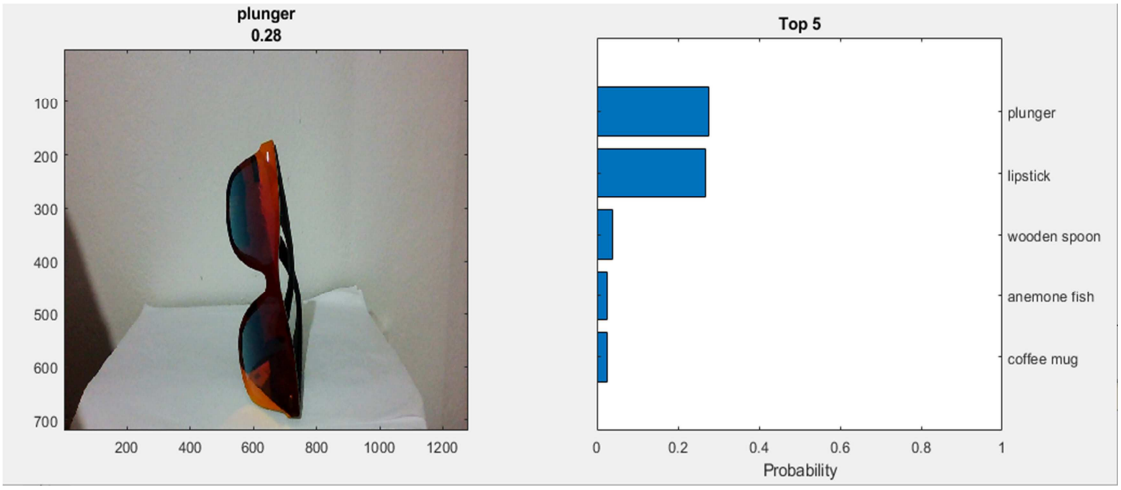
Les figures ci-dessous illustre le test fait avec un échantillon de chaque classe de notre base avec le réseau pré-entrainé AlexNet. Chaque image montre à gauche l'échantillon testé et à sa droite les classes des cinq premières probabilités d'appartenance par ordre décroissant.

Figures 3.5 montre en (a) un échantillon d'image de la classe ' sunglasses ' correctement classée et un autre échantillon en (b) de la même classe qui ne l'a pas été.

De la même façon Figure 3.6 et Figure 3.7 montrent des exemples de classement correct versus classement incorrect respectivement pour les classes



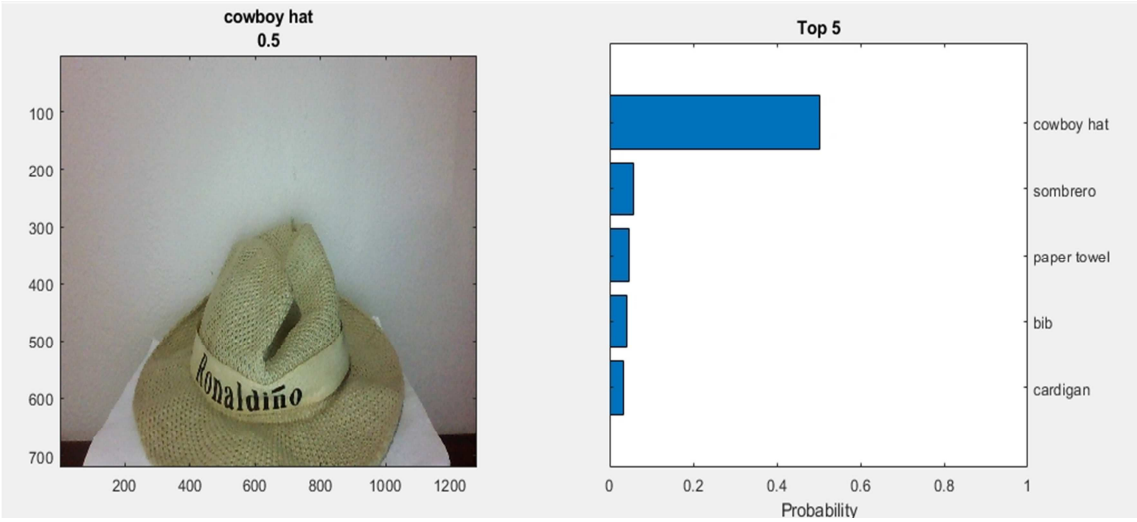
(a). Image correctement classée



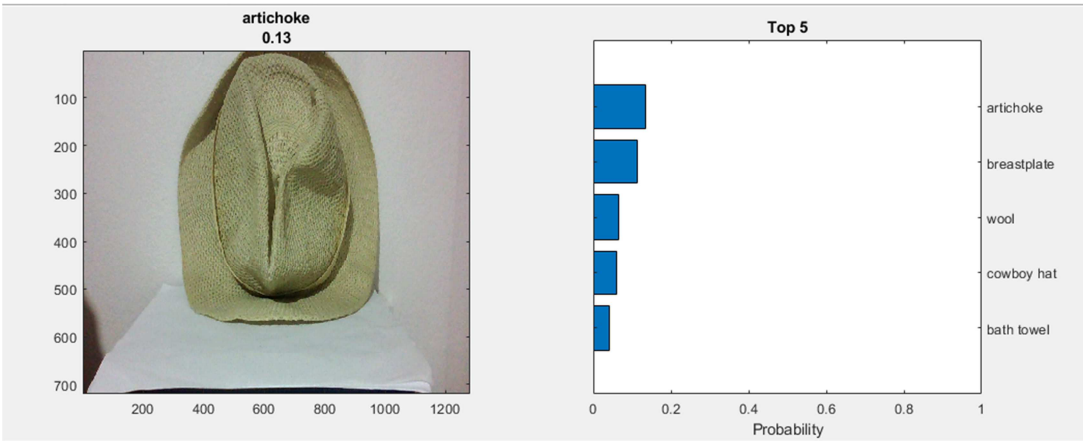
(b).

Image pas correctement classée

Figure 3.5. Exemples de tests avec deux images de la classe 'sunglasses'



(a). Image correctement classée



(b). Image pas correctement classée

Figure 3.6. Exemples de tests avec deux images de la classe ‘cowboy hat’

Les résultats de l'ensemble des tests effectués sont résumés par Tableau 3.1 ci-dessous.

Réseau CNN	Classe	Nombre d'images reconnues	Nombre d'images inconnues	Nombre total des images testées	Taux de reconnaissance
réseau pré-entrainé AlexNet	cap	28	206	234	**0,12**
	Cellulaire phone	96	172	268	**0,36**
	Cowboy hat	131	131	262	**0,50**
	headphone	29	240	269	**0,11**
	ipod	117	75	192	**0,61**
	lotion	81	174	255	**0,32**
	Modem	112	117	229	**0,49**
	sunglasses	103	184	287	**0,36**
	vase	207	98	305	**0,68**
	Wall clock	101	135	236	**0,43**

Tableau 3.1. Résultats de la classification avec le réseau pré-entrainé AlexNet

Le taux de reconnaissance moyen sur tous les tests est de **39,8%.**

3.5.2. Tests avec notre réseau

Le réseau que nous avons construit comme décrit à la section 3.4 précédente est entrainé sur une portion de notre base de donnée décrite à la section 3.3, l'autre portion est utilisée pour le test. Il est de coutume d'utiliser 80% de la base pour l'entrainement et les 20% restantes pour les tests.

Les résultats de l'ensemble des tests effectués sont résumés par Tableau 3.2 et illustrés par Figure 3.7 ci-dessous.

Réseau CNN	Classe	Taux de reconnaissance
Notre CNN	cap	**0,76**
	Cellulaire phone	**0,87**
	Cowboy hat	**0,81**
	headphone	**0,69**
	ipod	**0,91**
	lotion	**0,57**
	Modem	**0,65**
	sunglasses	**0,72**
	vase	**0,68**
	Wall clock	**0,82**

Tableau 3.1. Résultats de la classification avec le réseau pré-entrainé AlexNet

Le taux de reconnaissance moyen sur tous les tests est de **75%.**

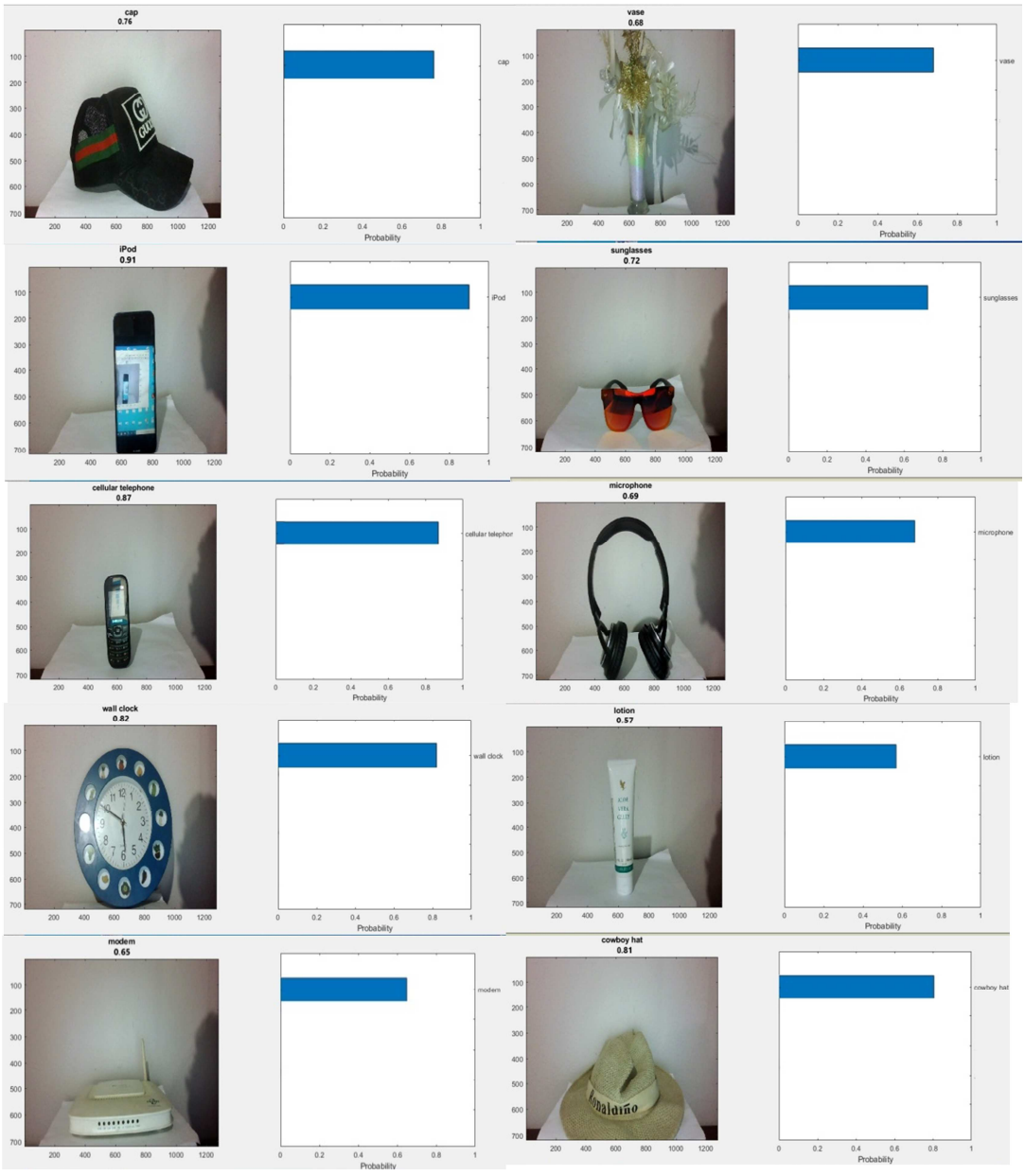


Figure 3.7. Taux de reconnaissance pour chaque classe avec notre réseau

3.6. Commentaires et interprétations

Nous pouvons remarquer (cf. Figure 3.5 et Figure 3.6) qu'une modification d'orientation ou de position peut influencer très nettement le résultat final de la classification. Ceci est visible surtout quand la base de données concerne des objets de nature différente mais d'apparence semblable comme c'est le cas pour la base de donnée ImageNet. Par contre

quand les images d'un objet donnée ne change pas trop d'orientation ou de position, comme c'est le cas de notre base de donnée, les taux de reconnaissance s'améliorent.

CONCLUSION

Nous avons implémenté un réseau de neurone convolutif et l'avons expérimenté dans un contexte de détection d'objet. MATLAB en a fourni la plateforme de programmation.

Le plus difficile de la tâche consiste à collecter les données puis à faire l'entrainement du réseau. On peut toujours utiliser des bases de données accessibles au public ou à la limite en utiliser une partie.

La configuration du réseau est une autre face du problème. En général il n'y a pas une configuration optimale générale préconisée dans la littérature, tout dépend du problème à modéliser et de l'ensemble des données.

Conclusion générale

En termes de ce travail nous pouvons dire que nous avons atteint le but fixé au début, en l'occurrence la conception d'un réseau de neurone pour la détection d'objet. L'étude bibliographique a montré que les réseaux convolutifs quelles que soit leurs variantes sont les mieux adaptés pour cette tâche.

Il nous a fallu apprendre à utiliser les outils disponibles sur Matlab pour la programmation d'un tel réseau. Les résultats sont prometteurs et les taux de reconnaissance acceptables étant donné la nature des objets à détecter.

Tout au long de ce travail nous avons pu noter l'importance du prétraitement (réduction de la dimension, padding, …) des images pour aboutir à une classification rigoureuse.

Le temps pour tester une image est négligeable par rapport à celui de l'entrainement. Nous ne nous sommes pas trop penchés sur le temps d'exécution comme paramètre d'évaluation vu

que cela dépend du matériel utilisé. L'évaluation du temps d'exécution exigerait une analyse de l'environnement, de préférence plusieurs ordinateurs différents de différents niveaux de performance, pour fournir des résultats ayant une valeur scientifique. Cela constituerait une voie intéressante pour la recherche concernant les applications temps réel.

Pour notre part l'apport de ce travail est notre familiarisation avec un outil très puissant de la vision automatique qui dispense l'opérateur de la tâche difficile de choisir, de rechercher et d'étiqueter lui-même les vecteurs discriminants de ses données.

Références

[1] Imagenet database statistics. http://image-net.org/about-stats. Accessed: 2017-04-07.

[2] Imagenet large scale visual recognition challenge 2016. http:// image-net.org/challenges/LSVRC/2016. Accessed: 2017-03-18.

[3] Infotrends - how long does it take to shoot 1 trillion photos? http://blog.infotrends.com/?p=21573. Accessed: 2017-06-20.

[4] Kpcb internet trends report 2014. http://www.kpcb.com/blog/ 2014-internet-trends. Accessed: 2017-06-20.

[5] Matconvnet: Cnns for matlab. http://www.vlfeat.org/matconvnet/. Accessed: 2017-06-21.

[6] Nvidia cuda gpu listing. https://developer.nvidia.com/cuda-gpus. Accessed: 2017-07-28.

[7] Piotr's computer vision matlab toolbox. https://pdollar.github.io/ toolbox/. Accessed: 2017-04-10.

[8] Software at the personal website of derek hoiem. http://dhoiem.cs. illinois.edu/software/. Accessed: 2017-04-21.

[9] Bengio, Y., et al. Deep learning of representations for unsupervised and transfer learning. ICML Unsupervised and Transfer Learning 27(2012), 17–36.

[10] Bileschi, S. M. StreetScenes: Towards scene understanding in still images. PhD thesis, Citeseer, 2006.

[11] Bishop, C. M. Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag New York, Inc., Secaucus,NJ, USA, 2006.

[12] Bodla, N., Singh, B., Chellappa, R., and Davis, L. S. Improving object detection with one line of code. arXiv preprint arXiv:1704.04503(2017).

[13] Chatfield, K., Simonyan, K., Vedaldi, A., and Zisserman, A.Return of the devil in the details: Delving deep into convolutional nets.arXiv preprint arXiv:1405.3531 (2014).

[14] Dalal, N., and Triggs, B. Histograms of oriented gradients for human detection. In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on (2005), vol. 1,IEEE, pp. 886–893.

[15] Dollar, P., and Zitnick, C. L. ´ Fast edge detection using structured forests. IEEE transactions on pattern analysis and machine intelligence 37, 8 (2015), 1558–1570.

[16] Erhan, D., Szegedy, C., Toshev, A., and Anguelov, D. Scalable object detection using deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2014),

pp. 2147–2154.

[17] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The pascal visual object classes (voc) challenge.International Journal of Computer Vision 88, 2 (June 2010), 303–338.

[18] Felzenszwalb, P. F., and Huttenlocher, D. P. Efficient graphbased image segmentation. International journal of computer vision 59, 2 (2004), 167–181.

[19] Fukushima, K. Neocognitron: A hierarchical neural network capable of visual pattern recognition. Neural networks 1, 2 (1988), 119–130.

[20] Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision (2015), pp. 1440–1448.

[21] Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and

pattern recognition (2014), pp. 580–587.

[22] Goodfellow, I., Bengio, Y., and Courville, A. Deep Learning. MIT Press, 2016. http://www.deeplearningbook.org.

[23] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016), pp. 770–778.

[24] Hoiem, D., Efros, A. A., and Hebert, M. Automatic photo popup. ACM transactions on graphics (TOG) 24, 3 (2005), 577–584.

[25] Hoiem, D., Efros, A. A., and Hebert, M. Geometric context from a single image. In Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on (2005), vol. 1, IEEE, pp. 654–661.

[26] Hoiem, D., Efros, A. A., and Hebert, M. Putting objects in perspective. International Journal of Computer Vision 80, 1 (2008), 3–15.

[27] Hornik, K. Approximation capabilities of multilayer feedforward networks. Neural networks 4, 2 (1991), 251–257.

[28] Huang, T. Computer vision: Evolution and promise. CERN EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH-REPORTS CERN (1996), 21–26.

[29] Hubel, D. H., and Wiesel, T. N. Receptive fields and functional architecture of monkey striate cortex. The Journal of Physiology 195, 1 (1968), 215–243.

[30] Ioffe, S., and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. CoRR abs/1502.03167 (2015).

[31] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014).

[32] Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (2012), pp. 1097–1105.

[33] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. Backpropagation applied to handwritten zip code recognition. Neural computation 1, 4 (1989),541–551.

[34] Li, Y., He, K., Sun, J., et al. R-fcn: Object detection via regionbased fully convolutional networks. In Advances in Neural Information Processing Systems (2016), pp. 379–387.

[35] Lin, T.-Y., Dollar, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. Feature pyramid networks for object detection. arXiv preprint arXiv:1612.03144 (2016).

[36] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. Ssd: Single shot multibox detector. In European Conference on Computer Vision (2016), Springer, pp. 21–37.

[37] Long, L. N., and Gupta, A. Scalable massively parallel artificial neural networks. Journal of Aerospace Computing, Information, and Communication 5, 1 (2008), 3–15.

[38] Lowe, D. G. Object recognition from local scale-invariant features. In Computer vision, 1999. The proceedings of the seventh IEEE international conference on (1999), vol. 2, Ieee, pp. 1150–1157.

[39] Marr, D., and Hildreth, E. Theory of edge detection. Proceedings of the Royal Society of London B: Biological Sciences 207, 1167 (1980), 187–217.

[40] Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems (2015), pp. 91–99.

[41] Rojas, R. Neural Networks - A Systematic Introduction). SpringerVerlag, Berlin, New-York, 1996.

[42] Rumelhart, D. E., Hinton, G. E., Williams, R. J., et al. Learning representations by back-propagating errors. Cognitive modeling 5, 3 (1988), 1.

[43] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. ImageNet Large Scale Visual Recogni-

tion Challenge. International Journal of Computer Vision (IJCV) 115, 3 (2015), 211–252.

[44] Sebe, N. Machine learning in computer vision, vol. 29. Springer Science & Business Media, 2005.

[45] Simonyan, K., and Zisserman, A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014).

[46] Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. A. Striving for simplicity: The all convolutional net. CoRR abs/1412.6806 (2014).

[47] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research 15

(2014), 1929–1958. [48] Steinkraus, D., Buck, I., and Simard, P. Using gpus for machine

learning algorithms. In Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on (2005), IEEE, pp. 1115–1120.

[49] Szeliski, R. Computer vision: algorithms and applications. Springer Science & Business Media, 2010.

[50] Torralba, A., Freeman, W. T., and Murphy, K. P. Graphical model for recognizing scenes and objects. In Advances in Neural Information Processing Systems 16, S. Thrun, L. Saul, and B. SchA˜¶lkopf, Eds. MIT Press, Cambridge, MA, 2003, p. None.

[51] Uijlings, J. R. R., van de Sande, K. E. A., Gevers, T., and Smeulders, A. W. M. Selective search for object recognition. International Journal of Computer Vision 104, 2 (2013), 154–171.

[52] Van de Sande, K. E., Uijlings, J. R., Gevers, T., and Smeulders, A. W. Segmentation as selective search for object recognition. In Computer Vision (ICCV), 2011 IEEE International Conference on (2011), IEEE, pp. 1879–1886.

[53] Walther, D., Itti, L., Riesenhuber, M., Poggio, T., and Koch, C. Attentional selection for object recognition - a gentle way. In International Workshop on Biologically Motivated Computer Vision (2002), Springer, pp. 472–479.

[54] Wilson, D. R., and Martinez, T. R. The general inefficiency of batch training for gradient descent learning. Neural Networks 16, 10 (2003), 1429–1451.

[55] Yang, B., Yan, J., Lei, Z., and Li, S. Z. Craft objects from images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016), pp. 6043–6051.

[56] Zeng, X., Ouyang, W., Yan, J., Li, H., Xiao, T., Wang, K., Liu, Y., Zhou, Y., Yang, B., Wang, Z., et al. Crafting gbd-net for object detection. arXiv preprint arXiv:1610.02579 (2016).

[57] Zitnick, C. L., and Dollar, P. ´ Edge boxes: Locating object proposals from edges. In European Conference on Computer Vision (2014), Springer, pp. 391–405.

GLOSSAIRE

Augmentation de données (Data augmentation)

Elle consiste à faire subir des transformations aux données avant de les utiliser pour l'entraînement tout en conservant leur structure. Pour chaque observation, on crée plusieurs variations. Ainsi le volume de données est artificiellement multiplié.

Carte de saillance

Un objet est saillant s'il est facilement remarqué.
La carte de saillance est une image dans laquelle la luminosité d'un pixel représente la saillance du pixel et donc est directement proportionnelle à sa saillance. Il s'agit généralement d'une image en niveaux de gris.
Elle est créée en utilisant les étapes suivantes :

- ••Les fonctionnalités de base comme la couleur, l'orientation, l'intensité sont extraites de l'image.
- Les images obtenues permettent de créer des pyramides gaussiennes pour créer des caractéristiques de la carte.
- La carte de saillance est créée en prenant la moyenne de toutes les cartes de caractéristiques.

Clustering (ou partitionnement des données)

Cette méthode de classification non supervisée rassemble un ensemble d'algorithmes d'apprentissage dont le but est de regrouper entre elles des données non étiquetées présentant des propriétés similaires. Les objets du même cluster (ou regroupement) sont similaires mais dissimilaires aux objets appartenant aux autres clusters.

Codage à la main (Hand coding)

En informatique, le codage manuel signifie éditer la représentation sous-jacente d'un programme informatique, lorsque des outils permettant de travailler sur une représentation de niveau supérieur existent également.

Descente du gradient

C'est un algorithme d'optimisation qui permet de trouver le minimum de n'importe quelle fonction convexe en convergeant progressivement vers celui-ci.

Note : Une fonction convexe est une fonction dont l'allure ressemble à celle d'une belle vallée avec au centre un minimum global.

Entropie croisée

Fonction de coût utilisée pour la classification automatique qui a pour but de quantifier la différence entre deux distributions de probabilités.

Goulot d'étranglement

Un goulot d'étranglement est un point d'un système limitant les performances globales, et pouvant avoir un effet sur les temps de traitement et de réponse.

Perceptron

Le perceptron est l'un des tout premiers algorithmes de Machine Learning, et le réseau de neurones artificiels le plus simple. C'est un algorithme d'apprentissage supervisé de classifieurs binaires (séparant deux classes). Il s'agit alors d'un type de classifieur linéaire.

Réalité augmentée

La réalité augmentée est la superposition de la réalité et d'éléments (sons, images 2D, 3D, vidéos, etc.) calculés par un système informatique en temps réel. Elle désigne souvent les différentes méthodes qui permettent d'incruster de façon réaliste des objets virtuels dans une séquence d'images.

Régression

Dans l'apprentissage automatique, le but de la régression est d'estimer une valeur (numérique) de sortie à partir des valeurs d'un ensemble de caractéristiques en entrée.

Score d'objet (Objectness)

Valeur de confiance qu'un objet existe dans le rectangle englobant. Cette mesure agit comme un détecteur d'objet générique de classe. Elle quantifie la probabilité qu'une fenêtre d'image contienne un objet de n'importe quelle classe, comme des voitures et des chiens, par opposition aux arrière-plans, tels que l'herbe et l'eau.

Segmentation d'image

C'est un traitement d'image qui a pour but de rassembler des pixels entre eux suivant des critères prédéfinis. Les pixels ainsi regroupés constituent une partition de l'image. On distingue principalement quatre classes de segmentation :

- La segmentation fondée sur les régions (Region-based segmentation). On y trouve par exemple : la croissance de région (Region-growing), décomposition/fusion (split and merge).
- La segmentation fondée sur les contours (Edge-based segmentation).
- La segmentation fondée sur la classification ou le seuillage (Thresholding) des pixels en fonction de leur intensité.
- La segmentation fondée sur la coopération entre les trois premières segmentations.

Seuillage

Le seuillage d'image est une technique simple de binarisation d'image, elle consiste à transformer une image en niveau de gris en une image dont les valeurs de pixels ne peuvent avoir que la valeur 1 ou 0. On parle alors d'une image binaire ou image en noir et blanc.

Superpixel

Un superpixel peut être défini comme un groupe de pixels qui partagent des caractéristiques communes
Un exemple type d'algorithme de génération de superpixels est l'Algorithme SLIC (Simple Linear Iterative Clustering). Cet algorithme génère des superpixels en regroupant les pixels en fonction de leur similitude de couleur et de leur proximité dans le plan de l'image.

Surapprentissage (Overfitting)

Il désigne le fait que le modèle prédictif produit par l'algorithme de Machine Learning s'adapte « trop » bien au Training Set. Par conséquent, le modèle prédictif capturera tous les "aspects" et détails qui caractérisent les données du Training Set. Dans ce sens, il capturera toutes les corrélations généralisables ainsi que le bruit produit par les données. On dit que la fonction prédictive se généralise mal. Le modèle prédictif pourra donner de très bonnes prédictions sur les données du Training Set (les données qu'il a déjà "vues" et auxquelles il s'y est adapté), mais il prédira mal sur des données qu'il n'a pas encore vues.

SVM (Support Vector Machines)

Les machines à vecteurs de support ou séparateurs à vaste marge sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre des problèmes de discrimination et de régression

ملخص

تعد تقنية التعرف على الاشياء مجالًا فرعيًا من رؤية الكمبيوتر و يعتمد □اليًا بشكل كبير على التعلم الآلي. على مدى العقد الماضي ، كان مجال التعلم الآلي يسيطر عليها ما يسمى بالشبكات العصبية العميقة ، والتي استفادت من التـ□سن في قوة الـ□وسبة وتوافر البيانات. نوع فرعي من الشبكة العصبية تسمى الشبكة العصبية التلافيفية (CNN) وهي مناسبة تمامًا للمهام المتعلقة بالصور. يتم تدريب الشبكة على الب□ث عن ميزات مختلفة ، مثل الـ□واف والزوايا واختلافات اللون عبر الصورة ودمجها في اكثرالأشكال تعقيدًا. بالنسبة لاكتشاف الكائن ، يجب على النظام تقدير مواقع الأشياء المـ□تملة وتصنيفها.

بالنسبة لأطرو□ة الماستر هذه ، قمنا بمراجعة الأدبيات الـ□الية □ول العصب التلافيفي وقمنا باختبار إنشاء تنفيذ وظيفي لـشبكة عصبية عميقة. و قمنا باختباره على مجموعة بيانات و قارناه بشبكة مدربة مسبقا

الكلمات المفتاحية: الرؤية الـ□اسوبية ، الكشف عن الأشياء ، التعلم الآلي ، الشبكات العصبية التلافيفية

Résumé

La détection d'objet est une sous discipline de la vision par ordinateur. Elle est basée aujourd'hui sur l'apprentissage automatique. Le long de la décennie précédente il y'a eu une prédominance des réseaux de neurones profonds qui ont tiré profit de l'avancée de la puissance de calcul et la disponibilité des bases de données. Un cas particulier des réseaux de neurones sont les réseaux de neurones convolutifs (CNN). Ces derniers se sont montrés particulièrement efficace dans le domaine de l'imagerie. Le réseau est entrainé pour rechercher les coins, les arrêtes et les nuances de couleurs relatifs à des structures complexes. Dans le cas de la détection d'objet le système estime les régions susceptibles de contenir des objets et les étiquette.

Dans ce mémoire de master nous avons fait une étude bibligraphique sur les CNN et avons conçu un réseau de neurone fonctionnel profond que nous avons utilisé avec une base de donnée. Les résultats sont comparés avec ceux d'un réseau pré-entrainé.

Mots clé : vision par ordinateur, détection d'objet, apprentissage automatique, réseau de neurones convolutifs.