

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE



UNIVERSITE Dr. TAHAR MOULAY SAIDA
FACULTE : TECHNOLOGIE
DEPARTEMENT : INFORMATIQUE



MÉMOIRE DE MASTER

Option : Sécurité informatique et Cryptographie

**Reconnaissance de l'empreinte digitale a base de
l'apprentissage profond**

Présenté par :

DEGHBADJ SLIMANE

ET

HADDIOUI ZANA

Encadré par :

DOCTEUR. MOHAMED EL HADI RAHMANI

Deep learning based fingerprint recognition

Abstract — The mark left by the fingerprint grooves and edges on the surface of the dictated mark The mark left by the fingerprint grooves and edges on the surface is smooth enough for Aldeo Aldeo. Finally, the imprint is removed from the sixth month of pregnancy. It is fixed and immutable. It was actually recreated symmetrically, even with an injury or burn. The design formed by fingerprints is unique to each of us. The probability that two people will have the same microfingerprints : one chance in 64 billion 1 This is why it is used to identify people in the computing field and this is what we will discuss in our note where we will try to exploit the security feature in the fingerprint through techniques Image processing such as computer vision, deep learning and CNN.

Keywords :fingerprint, computer vision, deep learning, convolutional neural network.

Reconnaissance des empreintes digitales basée sur l'apprentissage en profondeur

Résumé — La marque laissée par les rainures et les bords des empreintes digitales sur la surface de la marque dictée La marque laissée par les rainures et les bords des empreintes digitales sur la surface est suffisamment lisse pour Aldeo Aldeo. Enfin, l'empreinte est retirée dès le sixième mois de grossesse. Il est fixe et immuable. Il a en fait été recréé de manière symétrique, même avec une blessure ou une brûlure. Le design formé par les empreintes digitales est unique à chacun de nous. La probabilité que deux personnes aient les mêmes empreintes digitales : une chance sur 64 milliards 1 C'est pourquoi elle est utilisée pour identifier des personnes dans le domaine informatique et c'est ce dont nous parlerons dans notre note où nous tenterons d'exploiter la fonction de sécurité dans l'empreinte digitale à travers des techniques Traitement d'images tel que la vision par ordinateur, l'apprentissage profond et CNN.

Mots clés :empreinte digitale, vision par ordinateur, apprentissage en profondeur,convolutional neural network.

التعرف على بصمات الأصابع القائم على التعلم العميق

الملخص --- العلامة التي خلفتها الأخاديد وحواف بصمات الأصابع على سطح العلامة التي تم إملأها العلامة التي تركتها الأخاديد وحواف البصمات على السطح ناعمة بدرجة كافية لـ أخيراً ، يتم إزالة الانطباع من الشهر السادس من الحمل. إنه ثابت ولا يتغير. تم إعادة إنشائه بشكل متماثل في الواقع ، حتى مع وجود إصابة أو حرق. التصميم الذي تشكله بصمات الأصابع فريد من نوعه لكل منا. احتمالية أن يكون لشخصين نفس بصمات الأصابع: واحد من كل ٤٦ مليار فرصة ١ هذا هو سبب استخدامه للتعرف على الأشخاص في مجال الكمبيوتر وهذا ما سنتحدث عنه في مذكرتنا حيث سنحاول استغلال وظيفة البصمة الأمان من خلال تقنيات معالجة الصور مثل رؤية الكمبيوتر والتعلم العميق و الشبكات العصبية التلافيفية .

الكلمات المفتاحية: البصمة ، رؤية الكمبيوتر ، التعلم العميق ، الشبكة العصبية التلافيفية.

REMERCIEMENT

Nous sommes très reconnaissants à notre gestionnaire de mémoire M. Rahmani Mohamed El Hadi dans Le service informatique de l'UTMS, sans son initiative, ce projet n'aurait pas été possible. Nous tenons également à lui exprimer notre gratitude pour son dévouement et La confiance qu'il nous accorde, la justesse et la qualité des commentaires et Partagez vos suggestions M. Rahmani Mohamed El Hadi était très bon ancien ; Toujours ouvert aux nouvelles idées, toujours plein d'enthousiasme, Toujours prêt à nous remotiver. Merci à tous les membres du jury mémoire qui ont pris leur temps A lire et à juger ce livre ainsi que lors d'une excursion d'une journée de défense.

Nous remercions ensuite tous nos collègues du domaine de la sécurité informatique pour Leur sympathie et leur sens de l'humour. Merci à nos parents et à nos familles élargies pour leur soutien et leur aide dans Préparez ce mémoire et encouragez-les. En conclusion, nous tenons à remercier tous ceux qui ont participé de près ou de loin à la faire le travail.

TABLE DES MATIÈRES

| | |
|--|-------------|
| Table des matières | 6 |
| | Page |
| Liste des tableaux | 8 |
| Table des figures | 9 |
| 1 Vision par ordinateur | 3 |
| 1.1 Introduction au computer vision | 3 |
| 1.1.1 Qu'est-ce que la vision par ordinateur? | 4 |
| 1.1.2 Le défi de la vision par ordinateur | 4 |
| 1.1.3 Systèmes de vision par ordinateur à l'ancienne | 5 |
| 1.1.4 La vision par ordinateur et l'essor de l'apprentissage automatique | 6 |
| 1.1.5 Algorithmes d'IA / Machine Learning | 9 |
| 1.1.6 Abondance de données | 10 |
| 1.1.7 Connectivité | 10 |
| 1.1.8 Puissance de calcul | 12 |
| 1.1.9 Dynam.AI Computer Vision Solutions | 12 |
| 1.1.10 Les applications de la computer vision | 13 |
| 1.1.11 Conclusion | 13 |
| 2 SCIENCE DE EMPREINTE DIGITALE | 15 |
| 2.1 Introduction aux sciences empreinte digitale | 15 |
| 2.1.1 Étude anatomique des dermatoglyphes | 16 |
| 2.1.2 Utilisation des traces et empreintes digitales | 16 |
| 2.1.3 Traitement informatique des données relatives aux empreintes digitales | 18 |
| 2.1.4 Agencement des traces digitales | 19 |
| 2.1.5 Applications et restrictions | 20 |
| 2.1.6 Problématique | 21 |
| 2.1.7 Conclusion | 21 |
| 3 convolutional neural network(CNN) | 23 |

| | | |
|----------|---|-----------|
| 3.1 | Introduction | 23 |
| 3.1.1 | Historique du réseau de neurones convolutifs | 24 |
| 3.1.2 | Conception de réseau de neurones convolutifs | 24 |
| 3.1.3 | Exemple de couches de réseau neuronal convolutif expliquées | 25 |
| 3.2 | les couches qui composent le CNN | 26 |
| 3.2.1 | Couche convolutive | 26 |
| 3.2.2 | Réglage des paramètres | 30 |
| 3.2.3 | Max Pooling(couche de mise en commun) | 31 |
| 3.2.4 | Fully connected (couche entièrement connectée) | 32 |
| 3.2.5 | Dropout | 34 |
| 3.2.6 | Fonctions d'activation | 36 |
| 3.2.7 | Architecture CNN LeNet-5 | 40 |
| 3.3 | Applications des réseaux de neurones convolutifs | 42 |
| 3.3.1 | Réseaux de neurones convolutifs pour les voitures autonomes | 42 |
| 3.3.2 | Réseaux de neurones convolutifs pour la classification de texte | 43 |
| 3.3.3 | Réseaux de neurones convolutifs pour la découverte de médicaments | 44 |
| 3.4 | Apprentissage par transfert | 44 |
| 3.4.1 | Apprentissage par transfert pour la reconnaissance d'images | 45 |
| 3.4.2 | Comment utiliser des modèles pré-entraînés | 45 |
| 3.4.3 | Modèles d'apprentissage par transfert | 46 |
| 3.4.4 | Conclusion | 47 |
| 4 | Implementation et résultats | 49 |
| 4.1 | Introduction | 49 |
| 4.2 | Principes généraux de conception | 51 |
| 4.3 | Factorisation des convolutions avec un grand filtre Taille | 52 |
| 4.3.1 | Factorisation en plus petites convolutions | 53 |
| 4.3.2 | Factorisation spatiale en convolutions asymétriques | 54 |
| 4.4 | Utilité des classificateurs auxiliaires | 55 |
| 4.5 | Réduction efficace de la taille de la grille | 57 |
| 4.6 | Inception-v2 | 58 |
| 4.7 | Régularisation de modèle via le lissage d'étiquettes | 59 |
| 4.8 | Méthodologie de formation | 61 |
| 4.9 | Résultats Obtenus | 61 |
| 4.10 | Conclusion | 65 |
| | Bibliographie | 69 |

*

LISTE DES TABLEAUX

| TABLE | Page |
|---------------------------------|-------------|
| 4.1 Résultats obtenus | 62 |

TABLE DES FIGURES

| FIGURE | Page |
|---|-------------|
| 1.1 vision par ordinateur | 3 |
| 1.2 Caractéristiques de l'image | 5 |
| 1.3 flux de machine learning et deep learning | 7 |
| 1.4 Comment définir les propriétés | 8 |
| 1.5 Clasification des ingrédients | 8 |
| 1.6 L'évolution des réseaux de neurones | 10 |
| 1.7 volume de données | 11 |
| 1.8 appareils connectés | 11 |
| 1.9 histoire des prix de la puissance de calcul | 12 |
| | |
| 2.1 La forme de l'empreinte sur une surface lisse | 16 |
| 2.2 système d'étude d'empreintes digitales | 16 |
| 2.3 système d'identification | 17 |
| 2.4 henri fauld | 18 |
| 2.5 motifs d'empreintes digitales | 19 |
| 2.6 les minuties des empreintes digitales | 19 |
| 2.7 scanner d'identification d'empreinte digital | 21 |
| 2.8 Un guichet automatique pour certificat administratif à authentification basée sur les empreintes digitales | 21 |
| | |
| 3.1 réseaux de neurones convolution | 25 |
| 3.2 exemple de couche de réseau neuronal convolutif | 25 |
| 3.3 exemple de convolution | 26 |
| 3.4 filter ou kernel de convolution | 27 |
| 3.5 image sous form pixeles | 27 |
| 3.6 exemple de convolution avec un filter de 3x3 | 28 |
| 3.7 exemple de convolution sur un image | 28 |
| 3.8 un chat | 29 |
| 3.9 exemlpe de convolution sur un chat | 29 |
| 3.10 dimensions d'un filter | 30 |

TABLE DES FIGURES

| | | |
|------|--|----|
| 3.11 | stride | 30 |
| 3.12 | réglage de paramaetres | 31 |
| 3.13 | réglage de parametres | 31 |
| 3.14 | la couche pooling | 32 |
| 3.15 | fully connected | 33 |
| 3.16 | la complexité du modèle | 33 |
| 3.17 | dropout | 35 |
| 3.18 | dropout | 36 |
| 3.19 | fonction d'activation(non linear data | 37 |
| 3.20 | la fonction sigmoïde | 37 |
| 3.21 | la fonction de tanah | 38 |
| 3.22 | la fonction d'activation relu | 38 |
| 3.23 | leaky relu | 39 |
| 3.24 | les fonctions d'activation | 39 |
| 3.25 | | 40 |
| 3.26 | différenciation dans presque toutes les parties du Machine Learning et du Deep Learning. | 40 |
| 3.27 | Lenet5 | 41 |
| 3.28 | les couches de LeNet5 | 41 |
| 3.29 | résumé de LeNet5 | 42 |
| 3.30 | classification de texte | 43 |
| 4.1 | Figure 1. Mini-réseau remplaçant les 5×5 convolutions. | 52 |
| 4.2 | Figure 1. Mini-réseau remplaçant les 5×5 convolutions. | 53 |
| 4.3 | Figure 3. Mini-réseau remplaçant les 3×3 convolutions. le couche inférieure de ce réseau se compose d'une convolution 3×1 avec 3 unités de sortie. | 54 |
| 4.4 | Figure 4. Module de création d'origine tel que décrit dans [20]. | 55 |
| 4.5 | Figure 5. Modules de démarrage où chaque convolution 5×5 est remplacée par deux convolutions 3×3 , comme suggéré par le principe 3 de Section 2. | 55 |
| 4.6 | Figure 6. Modules de démarrage après la factorisation des $n \times n$ circonvolutions. Dans notre architecture proposée, nous avons choisi $n = 7$ pour la grille 17×17 . (Les tailles de filtres sont sélectionnées selon le principe 3). | 56 |
| 4.7 | Figure 7. Modules de lancement avec des sorties de banc de filtres étendues. Cette architecture est utilisée sur les grilles les plus grossières (8×8) pour favoriser représentations de grande dimension, comme suggéré par le principe 2 de Section 2. Nous utilisons cette solution uniquement sur la grille la plus grossière, puisque c'est l'endroit où la production de haute dimension clairsemée la représentation est la plus critique car le rapport de traitement local (de 1×1 convolutions) est augmentée par rapport à l'agrégation spatiale. | 57 |

| | | |
|------|--|----|
| 4.8 | Figure 8. Classificateur auxiliaire au-dessus de la dernière couche 17×17. Grouper la normalisation[7] des couches dans la tête latérale se traduit par un 0,4gain absolu dans la précision top-1. L'axe inférieur indique le nombre d'itérations effectuées, chacune avec une taille de lot de 32. | 58 |
| 4.9 | Figure 9. Deux manières alternatives de réduire la taille de la grille. La solution de gauche viole le principe 1 de ne pas introduire de goulot d'étranglement représentationnel de la section 2. La version de droite est 3 fois plus cher en calcul. | 58 |
| 4.10 | Figure 10. Module de lancement qui réduit la taille de la grille tout en élargissant les bancs de filtres. Il est à la fois bon marché et évite le goulot d'étranglement de représentation comme le suggère le principe 1. Le diagramme sur la droite représente la même solution mais du point de vue de tailles de grille plutôt que les opérations . . | 58 |
| 4.11 | Tableau 1. Aperçu de l'architecture de réseau proposée. la taille de sortie de chaque module est la taille d'entrée du suivant. Nous utilisons des variantes de la technique de réduction illustrée à la figure 10 pour réduire les tailles de grille entre les blocs de lancement, le cas échéant. Nous avons marqué la convolution avec 0-padding, qui est utilisé pour maintenir la taille de la grille. 0-rembourrage est également utilisé à l'intérieur des modules Inception qui ne réduisent pas la taille de la grille. Tous les autres les calques n'utilisent pas de rembourrage. Les différentes tailles de bancs de filtres sont choisies d'observer le principe 4 de la section 2. | 59 |
| 4.12 | Training Accuracy obtenus selon le nombre d'itération | 63 |
| 4.13 | Validation Accuracy obtenus selon le nombre d'itération | 63 |
| 4.14 | Cross Entropy obtenus selon le nombre d'itération | 64 |
| 4.15 | Test Accuracy obtenus selon le nombre d'itération | 65 |

INTRODUCTION GÉNÉRALE

La vision par ordinateur est un domaine scientifique interdisciplinaire qui traite de la façon dont les ordinateurs acquièrent une compréhension de haut niveau des images ou des vidéos numériques. Du point de vue de l'ingénierie, il cherche à comprendre et à automatiser les tâches que le système visuel humain peut effectuer. Les tâches de vision par ordinateur comprennent des méthodes d'obtention, de traitement, d'analyse et de « compréhension » d'images numériques et d'extraction de données pour produire des informations numériques ou symboliques, par exemple sous forme de décisions. Une empreinte digitale est un outil biométrique largement utilisé pour identifier les individus en médecine légale et médico-légale car elle est unique et immuable, elle ne change donc pas dans le temps. La probabilité de trouver deux empreintes digitales identiques est de 1 sur 64 milliards. La classification des empreintes digitales est basée sur la topographie générale de l'empreinte digitale et permet l'identification de familles telles que les anneaux, les arcs et les fentes. Dans chacune de ces catégories se trouvent un grand nombre d'éléments qui distinguent de manière unique chaque empreinte digitale. L'apprentissage en profondeur ou l'apprentissage en profondeur est un sous-domaine de l'intelligence artificielle (IA). Ce terme désigne l'ensemble des techniques d'apprentissage automatique, c'est-à-dire une forme d'apprentissage basée sur les méthodes mathématiques utilisées dans la modélisation des données. Qui se caractérise par un vaste réseau de neurones artificiels. Ces neurones sont interconnectés pour traiter et mémoriser des informations, comparer des problèmes ou des situations avec des situations passées similaires, analyser des solutions et résoudre le problème de la meilleure façon possible. Comme chez les humains, l'apprentissage en profondeur consiste à apprendre à partir d'expériences vécues ou, dans le cas des machines, d'informations enregistrées.

Problématique

Le problème auquel nous sommes confrontés est de savoir comment exploiter la fonction de sécurité des empreintes digitales en informatique, pour apprendre à la machine à comparer les images, à remarquer les changements et à classer les images en fonction de ces différences entre les images, afin de protéger notre logiciel et nos informations, et nous le fera en exploitant des techniques et des algorithmes d'apprentissage profond, en particulier la technologie Convolutional Neural Network, qui relève du domaine de la vision par ordinateur.

Organisation du mémoire

Notre mémoire est organisé comme suite :

- *Chapitre 01 : Vision par ordinateur donne une vue globale sur le domain de traitement d'image et la vision par ordinateur.*
- *Chapitre 02 : Science de l'empreinte digitale parle sur la reconnaissance d'empreinte digitale, ses processus et domaines d'application.*
- *Chapitre 03 : Convolutional neural network discute les notions de bases des réseaux de neurones convolutionnels.*
- *Chapitre 04 : Implémentation et résultats détaille l'approche proposée ainsi les résultats obtenus.*

VISION PAR ORDINATEUR

1.1 Introduction au computer vision

La computer vision désigne une technique d'intelligence artificielle permettant d'analyser des images captées par un équipement tel qu'une caméra. Concrètement, la computer vision se présente comme un outil basé sur l'IA capable de reconnaître une image, de la comprendre, et de traiter les informations qui en découlent. Pour beaucoup, la vision par ordinateur est l'équivalent, en termes d'IA, des yeux humains et de la capacité de notre cerveau à traiter et analyser les images perçues. La reproduction de la vision humaine par des ordinateurs constitue d'ailleurs l'un des grands objectifs de la computer vision.



FIGURE 1.1: vision par ordinateur

1.1.1 Qu'est-ce que la vision par ordinateur?

L'apprentissage automatique, en particulier l'apprentissage profond, a transformé la vision par ordinateur en quelques années seulement. La technologie de vision par ordinateur est l'un des domaines de recherche les plus prometteurs en intelligence artificielle et en informatique, et offre d'énormes avantages pour les entreprises de l'ère moderne.

En son cœur, le domaine de la vision par ordinateur se concentre sur la conception de systèmes informatiques capables de capturer, de comprendre et d'interpréter des informations visuelles importantes contenues dans des données d'image et de vidéo. Les systèmes de vision par ordinateur traduisent ensuite ces données, en utilisant les connaissances contextuelles fournies par les êtres humains, en informations utilisées pour orienter la prise de décision. Transformer les données d'images brutes en concepts de niveau supérieur afin que les humains ou les ordinateurs puissent les interpréter et agir en conséquence est l'objectif principal de la technologie de vision par ordinateur.

Une distinction importante doit être faite entre la vision par ordinateur et le traitement d'images. Le traitement d'image est la science consistant à apporter des modifications à une image de manière à produire une nouvelle image avec certaines caractéristiques améliorées. Ces changements incluent une résolution accrue, une luminosité et un contraste normalisés, un recadrage, un flou ou toute autre transformation numérique nécessaire à un usage spécifique. Le traitement d'image numérique ne prend pas en considération le contenu réel de l'image - il s'agit simplement d'une série de transformations mécaniques entreprises pour modifier l'image dans un but défini. Une définition moderne de la vision par ordinateur, d'autre part, signifie permettre aux ordinateurs de traiter les images visuelles, données et extraire des informations à partir de ces données. Le contenu est important, tout comme la capacité de traduire les pixels bruts en une forme interprétable par l'homme ou d'autres systèmes informatiques. L'objectif est d'enseigner aux ordinateurs comment identifier, classer et catégoriser le monde visuel comme nous le faisons.

[KALANTARI, Mahzad et KASSER, Michel].

1.1.2 Le défi de la vision par ordinateur

Les systèmes de vision humaine ont l'énorme avantage d'être informés par une vie de connaissances expérientielles qui aident à contextualiser les données dans votre champ de vision. Vos globes oculaires capturent des informations visuelles - l'image d'un chat, par exemple - et votre expérience antérieure interprète cette collection de lumière réfléchi et la relie au concept de chat. La complexité de notre système de perception visuelle et sa relation étroite avec notre mémoire et nos capacités de raisonnement supérieures donnent à ces données visuelles le contexte dont elles ont besoin pour apporter de la valeur dans les activités quotidiennes.

Ces facultés humaines, bien qu'indisponibles pour les ordinateurs, peuvent être imitées efficacement grâce à des algorithmes d'apprentissage automatique. Mais il s'avère qu'apprendre à des machines à imiter cette fonction humaine de base, fièrement démontrée par des enfants de

cinq ans du monde entier, est exceptionnellement difficile. La résolution de ce problème occupe continuellement les esprits les plus brillants de la recherche sur l'IA.

1.1.3 Systèmes de vision par ordinateur à l'ancienne

Avant 2012, la conception des systèmes de vision par ordinateur était remarquablement différente de ce qu'elle est aujourd'hui.

Si nous devons séparer notre compréhension d'un chat, nous verrions qu'un chat est en réalité un amalgame de plusieurs "caractéristiques" différentes. Ces caractéristiques comprennent une tête, des oreilles, un corps, quatre pattes et une queue - qui se combinent pour déclencher nos systèmes de mémoire et nos fonctions cognitives d'ordre supérieur pour produire la compréhension conceptuelle que nous voyons un chat. Nous pouvons les décomposer encore plus. Une tête est composée de deux yeux, un nez. Les jambes comprennent de longues formes à peu près cylindriques avec des pattes attachées, chacune avec quatre orteils de forme oblongue.

La formation des systèmes de vision par ordinateur impliquait de suivre ce processus jusqu'aux plus petites unités granulaires de données visuelles - le pixel. Le système enregistre et évalue les images numériques sur la base de ses seules données brutes, où des différences minimales de densité de pixels, de saturation des couleurs et de niveaux de luminosité et d'obscurité déterminent la structure et donc l'identité de l'objet plus grand. Un arrangement particulier de pixels peut indiquer une moustache sur le nez d'un chat ou une oreille humaine, par exemple. L'image ci-dessous montre les caractéristiques constitutives d'une image décomposées en un ensemble de densités de pixels. Les premières techniques de vision par ordinateur reposaient sur

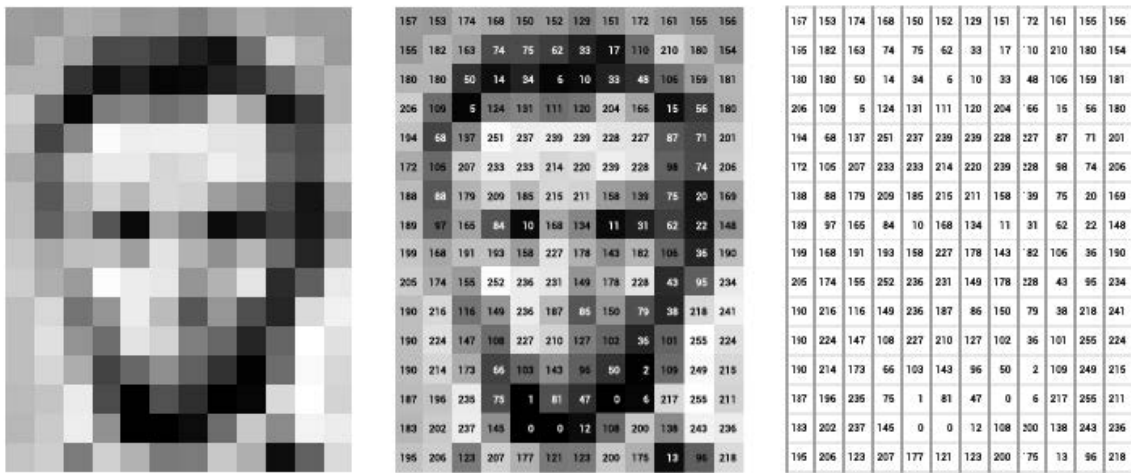


FIGURE 1.2: Caractéristiques de l'image

un effort manuel considérable pour créer des techniques de classification basées sur des règles afin de détecter et de classer certains groupes de ces arrangements de pixels. Les êtres humains

sélectionnaient manuellement ce qu'ils croyaient être les caractéristiques pertinentes des objets individuels. Ils ont explicitement dit à la machine : « les chats sont faits de jambes, les jambes sont faites de cuisses et de pattes, et les pattes sont faites d'orteils.

Les ingénieurs ont codifié chacun de ces composants d'un chat dans un ordinateur en tant que règles rigides qui pourraient détecter ces caractéristiques dans l'image. Pendant 30 ans, le domaine de la technologie de vision par ordinateur s'est appuyé sur ces détecteurs de caractéristiques lourds et fabriqués manuellement pour trier et classer les données d'image.

Ces mécanismes étaient rigides, difficiles à améliorer ou à modifier, et très chronophages à produire manuellement pour chaque nouvelle application ou objet nécessitant une détection. De plus, lorsque le nombre de classes que le modèle tente de classer augmente ou que la clarté de l'image diminue, les techniques traditionnelles de vision par ordinateur ont souvent tendance à échouer. De simples changements dans la taille, la rotation ou l'orientation de l'objet briseraient ces systèmes. Pour atteindre des performances élevées, ces systèmes de vision doivent être indépendants de ces facteurs.

[Lakhal, Nihad, Nada Rouimel, and Mourad Encadreur Grimes.]

1.1.4 La vision par ordinateur et l'essor de l'apprentissage automatique

Les progrès de l'apprentissage automatique ont modifié à jamais le destin de la technologie de vision par ordinateur. L'apprentissage en profondeur, en particulier, a rendu les algorithmes de vision par ordinateur très efficaces dans le monde réel. L'avènement du réseau de neurones convolutifs a rendu la vision par ordinateur réalisable pour les applications industrielles et a cimenté la technologie comme un investissement valable pour les entreprises cherchant à automatiser les tâches.

Les techniques traditionnelles de vision industrielle commencent par une prescription descendante des composants qui constituent l'image – ses «caractéristiques». Les modèles d'apprentissage en profondeur renversent tout ce processus. Le processus de formation du réseau de neurones profonds utilise des ensembles de données massifs et d'innombrables cycles de formation pour enseigner à la machine, de bas en haut, à quoi ressemble un chat. Pendant le processus de formation, l'algorithme extrait automatiquement les caractéristiques pertinentes des « chats » en général. Ce processus produit un modèle qui peut être appliqué à des images inédites pour produire une classification précise. L'image ci-dessous décrit le processus d'apprentissage automatique traditionnel pour la reconnaissance d'images et la détection d'objets par rapport à une approche basée sur l'apprentissage en profondeur. L'apprentissage en profondeur

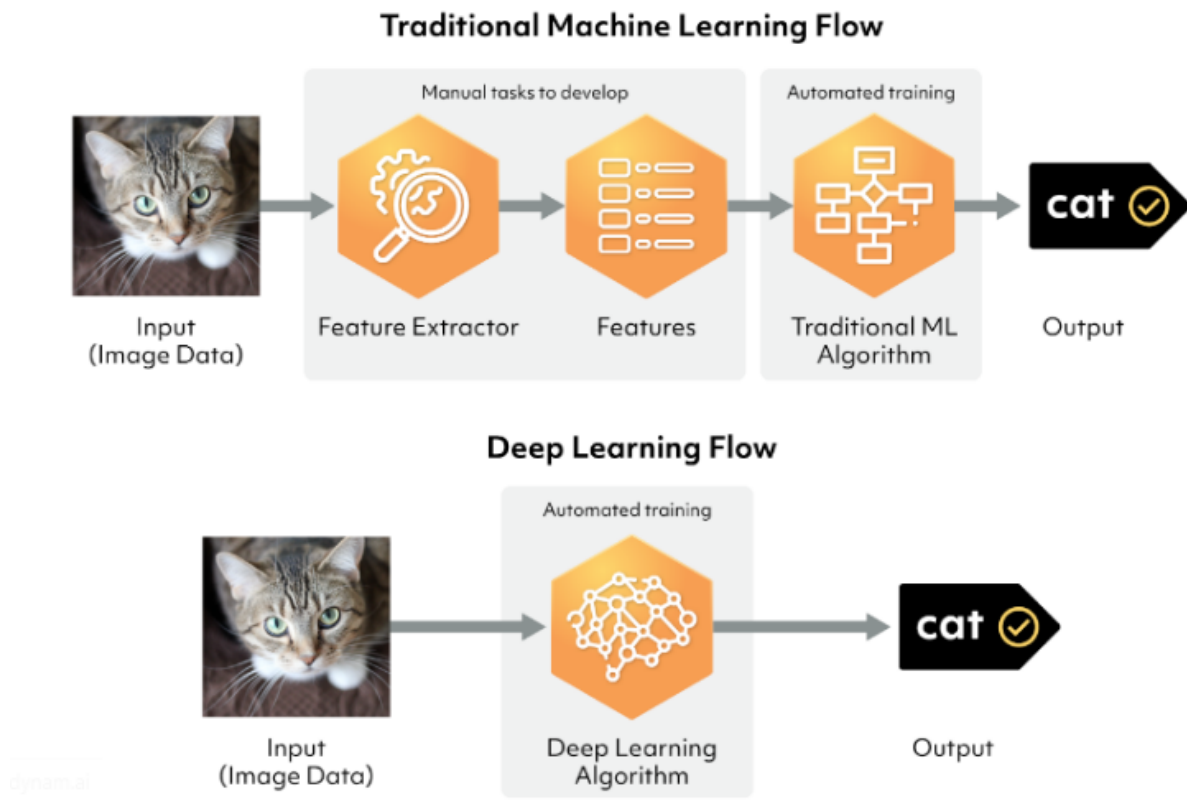


FIGURE 1.3: flux de machine learning et deep learning

de la vision par ordinateur a été rendu possible grâce à l'abondance de données d'images dans le monde moderne et à une réduction du coût de la puissance de calcul nécessaire pour les traiter. Des ensembles d'images à grande échelle comme ImageNet, CityScapes et CIFAR10 ont rassemblé des millions d'images avec des fonctionnalités étiquetées avec précision pour que les algorithmes d'apprentissage en profondeur se régalaient. Apparemment du jour au lendemain, les performances des algorithmes d'apprentissage en profondeur ont dépassé trente ans de travail sur les détecteurs de caractéristiques manuels.

L'alimentation d'un nombre suffisant d'images bien étiquetées dans un système visuel basé sur l'apprentissage en profondeur lui permet de comprendre les nuances exactes au niveau des pixels qui définissent les composants individuels de l'image plus grande. Il apprendra automatiquement où se trouvent les bords et comment des combinaisons particulières de bords, dont la couleur et le contraste différent les uns des autres et de l'arrière-plan, se combinent pour former certaines caractéristiques. L'image ci-dessous montre comment un système d'apprentissage en profondeur peut identifier les caractéristiques d'un chat. Les couches convolutives d'ordre supérieur du

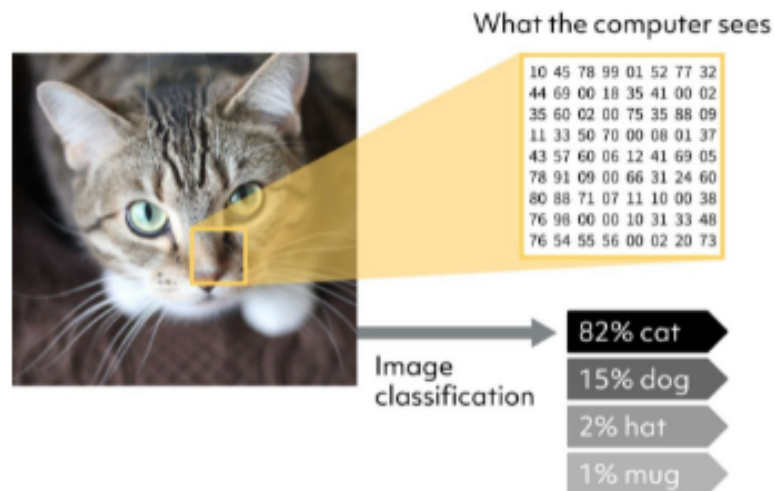


FIGURE 1.4: Comment définir les propriétés

réseau de neurones commenceront à comprendre que si vous avez 4 pattes, une tête, une queue et un corps, l'image en question peut contenir un chat. À partir de données visuelles brutes au niveau des pixels, la machine renvoie un concept d'ordre supérieur - "chat" - basé sur l'addition et la classification séquentielles de ces composants individuels. L'image ci-dessous démontre cette compréhension progressive dans le contexte de la reconnaissance faciale humaine. La

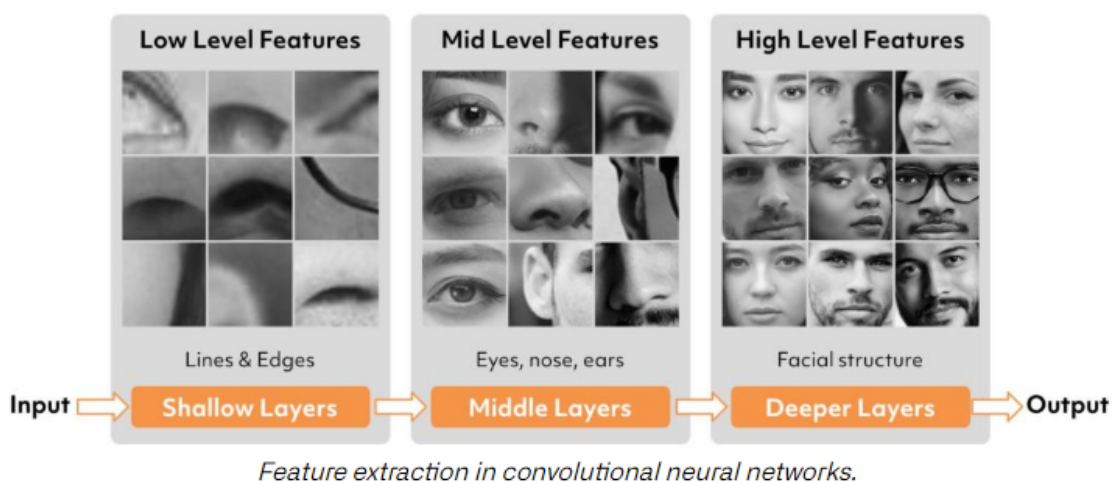


FIGURE 1.5: Clasification des ingrédients

différence est que les systèmes de vision traditionnels impliquent un humain disant à une machine ce qui devrait être là par rapport à un algorithme d'apprentissage en profondeur extrayant automatiquement les caractéristiques de ce qui est là. L'approche ascendante est

beaucoup plus efficace pour certains types de problèmes d'analyse d'images, dont beaucoup sont fréquemment utilisés dans notre vie quotidienne.

La différence est subtile, et peut-être pas évidente dans le contexte de notre chat. Cependant, lors du diagnostic d'un échantillon de tissu, ces différences deviennent précieuses. De petites fluctuations imperceptibles de la densité de pixels peuvent signifier l'apparition précoce d'un cancer - des détails que même les pathologistes experts pourraient manquer. La perception visuelle humaine atteint une barrière de résolution d'image à environ 2290 pixels par pouce et atteint un plafond pour la perception du mouvement à environ 30 images par seconde.

De plus, les humains se fatiguent lorsqu'ils effectuent des observations répétitives sur de nombreuses images. Cette fatigue entraîne des résultats commerciaux médiocres, comme c'est le cas avec l'inspection visuelle dans le contrôle de la qualité de la fabrication. La fatigue présente également des risques pour la vie humaine, comme cela est courant dans de nombreuses disciplines médicales ou dans diverses professions telles que la maintenance des infrastructures ou des aéronefs. Les processus d'inspection automatisés peuvent économiser de l'argent et des vies dans ces domaines.

La capacité des systèmes de vision par ordinateur à fonctionner avec une précision au niveau du pixel, à itérer rapidement et à fonctionner de manière cohérente dans le temps offre un potentiel incroyable pour augmenter ou surpasser la perception humaine.

1.1.5 Algorithmes d'IA / Machine Learning

Les progrès de l'IA et des algorithmes d'apprentissage automatique, en particulier les techniques d'apprentissage en profondeur, ont permis d'analyser les montagnes d'informations présentes à l'ère moderne. Portés par la baisse du coût de la puissance de calcul, les algorithmes d'apprentissage en profondeur peuvent analyser des milliards de données pour produire des modèles d'ordres de grandeur plus complexes que leurs prédécesseurs. L'automatisation algorithmique du processus de formation des réseaux neuronaux a produit d'énormes gains d'efficacité.

La prolifération de modèles d'apprentissage automatique pré-entraînés et open source au sein de la communauté de la science des données a démocratisé l'accès aux dernières techniques. Ces avancées ont étendu la capacité des ingénieurs en apprentissage automatique à expérimenter rapidement avec de grands ensembles de données pour résoudre des problèmes de vision par ordinateur de plus en plus complexes. Ci-dessous, nous pouvons voir l'évolution de simples réseaux de neurones vers des architectures spécialement conçues pour les applications de vision.

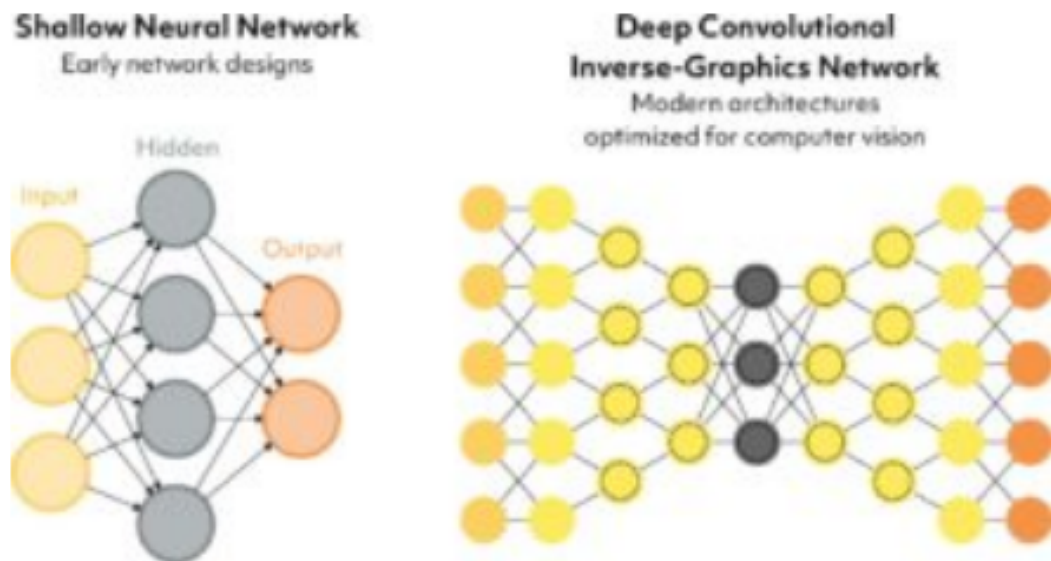


Chart showing the evolution of neural network design.

FIGURE 1.6: L'évolution des réseaux de neurones

1.1.6 Abondance de données

Le volume considérable de données dans le monde moderne peut fournir aux modèles d'apprentissage automatique toutes les données brutes dont ils ont besoin pour se régaler. Les données des médias sociaux, les données mobiles, les données de streaming des appareils IoT et la numérisation croissante de secteurs entiers sont le carburant dont les réseaux d'apprentissage en profondeur ont besoin pour devenir hyper-efficaces dans certaines tâches. L'infrastructure sous-jacente qui prend en charge les initiatives de transformation numérique des entreprises est désormais commercialisée, accessible et adoptée à grande échelle. Le monde de l'entreprise est désormais capable d'organiser son énorme quantité de données de manière sophistiquée et centralisée.

1.1.7 Connectivité

Des niveaux exponentiels de connectivité permettent aux données d'être transférées rapidement et efficacement. Les capteurs IoT, les téléphones portables, les caméras et les drones ont créé d'énormes quantités de nouvelles données d'images. Internet et le réseau de câbles à fibres optiques qui le soutiennent connectent presque toutes les personnes sur Terre. L'accès omniprésent à la connectivité sans fil a créé de puissantes artères pour l'échange de données

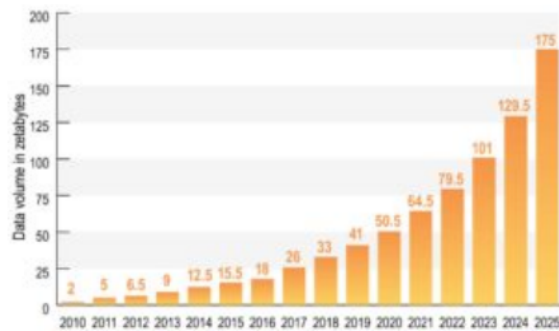


FIGURE 1.7: volume de données

d'images à grande échelle qui convergent au sein d'un point d'accès centralisé dans le cloud. Un logiciel de vision par ordinateur déployé localement « à la périphérie » peut recevoir des entrées et transmettre des données de sortie partout dans le monde.

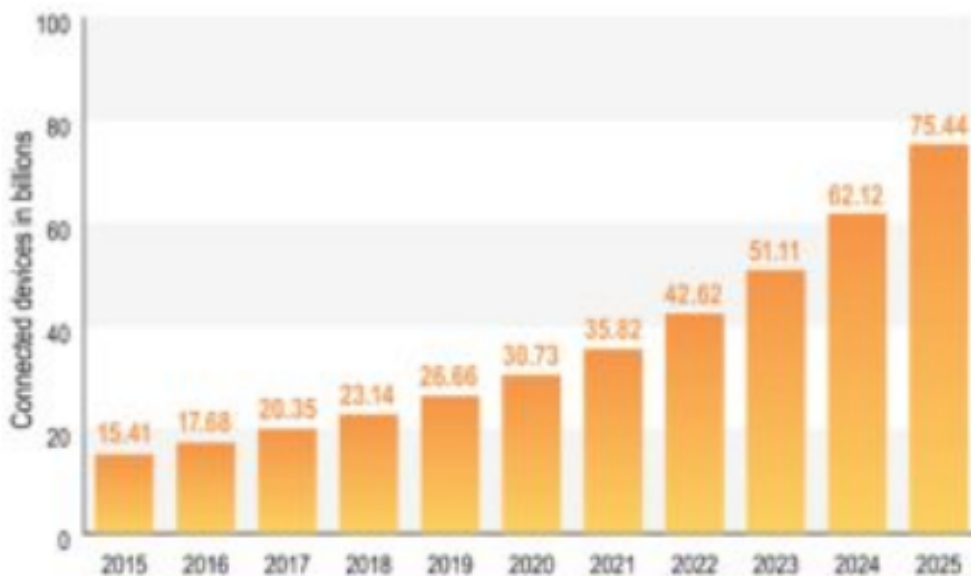
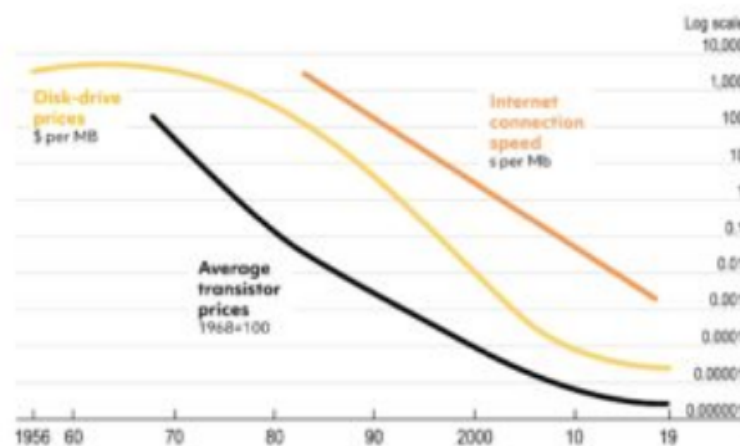


FIGURE 1.8: appareils connectés

1.1.8 Puissance de calcul

Le coût de l'informatique a chuté de façon exponentielle au cours des dernières décennies. Une estimation indique que la quantité de puissance de calcul disponible par dollar a été multipliée par dix environ tous les quatre ans au cours du dernier quart de siècle (mesurée en FLOPS ou MIPS). Cette réduction des coûts trouve son origine dans la commercialisation généralisée de centres de données à grande échelle, les progrès des matériaux semi-conducteurs et le développement de matériel spécialisé spécialement conçu pour la formation de réseaux de neurones (TPU, par exemple). Le graphique ci-dessous met en évidence la baisse exponentielle du coût de l'informatique de 1956 à 2019. Ce sont les marées montantes dont la force collective a ouvert la



History of the price of computing power. Source: Nielsen

FIGURE 1.9: histoire des prix de la puissance de calcul

voie à des applications commerciales modernisées de la technologie de vision par ordinateur. Les techniques d'apprentissage automatique telles que l'apprentissage en profondeur ont accéléré le développement de la technologie de vision par ordinateur au-delà du seuil requis pour trouver une valeur commerciale. Les tâches visuelles telles que la classification d'images, la reconnaissance d'objets et la segmentation d'images peuvent désormais atteindre des performances élevées de manière rentable pour le déploiement en entreprise. L'adoption généralisée est une réalité inévitable.

1.1.9 Dynam.AI Computer Vision Solutions

Dynam.AI propose des solutions d'IA de bout en bout pour les entreprises qui cherchent à capitaliser sur les forces qui balaient le paysage commercial moderne. Notre équipe multidisciplinaire d'experts en IA, d'ingénieurs en apprentissage automatique et de scientifiques des données

a produit des solutions innovantes pour les plus grandes entreprises du pays. Nous avons une expérience pertinente dans les domaines de la santé, des services financiers, des infrastructures et de la fabrication. Notre expertise la plus profonde réside dans l'apprentissage en profondeur pour la vision par ordinateur et l'intégration de nouvelles technologies d'IA innovantes dans les opérations commerciales traditionnelles.

Demandez une consultation gratuite à Dynam.AI dès aujourd'hui pour découvrir comment la vision par ordinateur et l'apprentissage automatique peuvent transformer votre entreprise.

1.1.10 Les applications de la computer vision

Ces dernières années, les plus grandes entreprises internationales (Google, Facebook, Amazon, Apple) ont massivement investi dans le deep learning et dans la computer vision. Dans le secteur automobile, le constructeur de véhicules autonomes Tesla a depuis plusieurs années mis l'accent sur la computer vision, plus que sur l'IoT. Le postulat qui justifie cette prise de position : les caméras connectées capables de traiter l'information en temps réel proposent une plus grande fiabilité que les différents capteurs électroniques.

Dans l'énergie, Suez utilise la computer vision dans l'eau et les déchets, notamment pour détecter les objets qui ne sont pas destinés à entrer dans l'incinérateur. Autre exemple dans l'industrie, où la start-up Prophesee entend utiliser les images pour assurer une maintenance prédictive. Par ailleurs, avec la crise du coronavirus, le spécialiste des solutions de sécurité Dahua Technology a réadapté ses caméras pour détecter par computer vision les personnes ayant de la fièvre.

1.1.11 Conclusion

Ces dernières années, les plus grandes entreprises internationales (Google, Facebook, Amazon, Apple) ont massivement investi dans le deep learning et dans la computer vision. Dans le secteur automobile, le constructeur de véhicules autonomes Tesla a depuis plusieurs années mis l'accent sur la computer vision, plus que sur l'IoT. Le postulat qui justifie cette prise de position : les caméras connectées capables de traiter l'information en temps réel proposent une plus grande fiabilité que les différents capteurs électroniques. Dans l'énergie, Suez utilise la computer vision dans l'eau et les déchets, notamment pour détecter les objets qui ne sont pas destinés à entrer dans l'incinérateur. Autre exemple dans l'industrie, où la start-up Prophesee entend utiliser les images pour assurer une maintenance prédictive. Par ailleurs, avec la crise du coronavirus, le spécialiste des solutions de sécurité Dahua Technology a réadapté ses caméras pour détecter par computer vision les personnes ayant de la fièvre. Tout cet intérêt dans le domaine de la vision par ordinateur est dû à la puissance des algorithmes et des techniques de traitement des images telles que la technologie d'apprentissage profond, qui permet aux humains de surmonter les difficultés et les obstacles dans le domaine du tri et de la classification de grandes bases de

données qui ne peuvent pas être triées manuellement. Analyser les données dans des périodes de temps acceptables.

SCIENCE DE EMPREINTE DIGITALE

2.1 Introduction aux sciences empreinte digitale

Une empreinte digitale ou dactylogramme est le dessin formé par un doigt sur un support suffisamment lisse pour qu'y restent marqués les dermatoglyphes.

Les empreintes digitales sont uniques à chaque individu et chaque doigt a son empreinte propre. La probabilité que deux personnes aient les mêmes empreintes digitales est infinitésimale : une chance sur 64 milliards¹.

Les empreintes digitales commencent à se former entre la 10^e et la 16^e semaine de vie du fœtus, par un plissement des couches cellulaires². Si l'expression des gènes joue un rôle, les circonvolutions des crêtes qui leur donne leur dessin caractéristique dépendent de nombreux facteurs externes, notamment la vitesse de croissance des doigts, l'alimentation du fœtus, sa pression sanguine, etc. Ainsi, deux vrais jumeaux révéleront une grande similarité mais seront différentes. À 24 semaines, la géométrie des empreintes est fixée définitivement pour toute la vie de l'individu et les seules déformations qui se produiront ensuite viendront de la croissance des doigts¹.

Les procédés d'identification des individus par leurs empreintes digitales, sans l'aide d'ordinateur, sont désignés sous le nom de « dactylotechnie »³. L'étude des dessins digitaux s'appelle la « dactyloscopie » (du grec daktylos, « doigt », et scopie, « examen »), tandis que l'étude des dessins palmaires s'appelle la "chiroscopie" (du grec ancien , kheír, « main », et scopie, « examen »). Le caractère quasi-unique d'une empreinte digitale ou palmaire en fait un outil biométrique très utilisé en médecine légale et par la police scientifique pour l'identification des individus.



FIGURE 2.1: La forme de l’empreinte sur une surface lisse

2.1.1 Étude anatomique des dermatoglyphes

Au xvii^e siècle, l’anatomiste Marcello Malpighi identifie les papilles dermiques et les pores exocrines des crêtes dermiques⁶, tandis qu’elles sont représentées sur une planche d’un ouvrage d’anatomie de Govard Bidloo⁷. En 1678, le botaniste et morphologiste anglais Nehemiah Grew décrit scientifiquement les dessins formés par les crêtes et les plis dermiques dans son rapport pour les Philosophical Transactions de la Royal Society.

Le physiologiste tchèque Jan Evangelista Purkinje publie en 1823 une thèse⁸ dans laquelle il classe ces dessins en neuf groupes, ce qui est très proche du système utilisé de nos jours.

[Toufik, Hafs.]

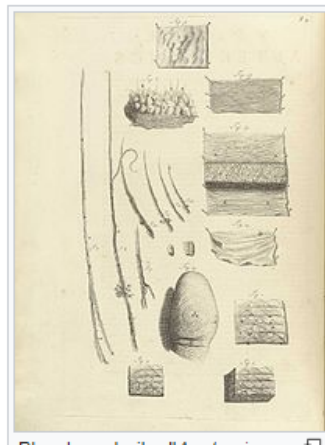


FIGURE 2.2: système d’étude d’empreintes digitales

2.1.2 Utilisation des traces et empreintes digitales

En 1877, aux Indes, le Britannique William James Herschel utilise les empreintes digitales pour éviter que les bénéficiaires de pension de l’armée ne la touchent plusieurs fois. À cette époque, elles servent aussi à identifier les marchands locaux qui refusent de remplir les termes

de leurs contrats : Herschel fait apposer leurs empreintes digitales sur ces contrats⁹. Le médecin écossais Henry Faulds (en), en poste au Japon, publie en 1880, dans la revue *Nature* un article¹⁰ dans lequel il discute de l'utilité des empreintes pour l'identification notamment des criminels, et où il propose une méthode pour enregistrer celles-ci avec de l'encre d'imprimerie ; il indique qu'il a confondu ainsi deux cambrioleurs¹¹. Il est aussi le premier à identifier des traces laissées sur un flacon. Il écrit à Charles Darwin pour lui expliquer sa méthode, mais le naturaliste, âgé et malade, transmet le courrier à son cousin Francis Galton, l'un des fondateurs de l'eugénisme et de la méthode statistique.

S'intéressant surtout à l'anthropologie, Francis Galton se penche à partir de 1888, à l'occasion d'une conférence qu'il est amené à faire à la Royal Society à propos de l'identification des individus et, en particulier, au sujet de la méthode Bertillon¹², sur l'étude des dermatoglyphes et publie en 1892 un ouvrage, *Finger Prints (Empreintes digitales)*¹³, dans lequel il établit l'unicité et la permanence de ces figures cutanées. Il conçoit un système de classification détaillé et estime alors à 1 sur 64 milliards la probabilité que deux individus puissent laisser les mêmes traces. C'est à la suite des travaux de Galton qu'on redécouvre l'utilisation des empreintes digitales comme moyen d'identification.

En 1891, après avoir étudié les écrits de Galton, Juan Vucetich, fonctionnaire de police, crée en Argentine le premier fichier d'empreintes. Il sera, l'année suivante, le premier à identifier un criminel – en l'occurrence une criminelle – sur la base des empreintes digitales. L'utilisation du terme « méthode vuceticienne » pour désigner la dactyloscopie est toujours employée dans la police. Deux années plus tard, Edward Henry (en), inspecteur britannique affecté au Bengale, met au point un système d'identification similaire à celui de Vucetich, système qui est toujours utilisé dans les pays anglophones. Ce « système Henry » définit des familles de dessins papillaires : boucles, arches, tourbillons... De retour à Londres, Henry fait adopter, dès 1897, cette technique par Scotland Yard. Il crée le premier fichier d'empreintes digitales en 1901 ; celui-ci vient alors compléter le bertillonage.



FIGURE 2.3: système d'identification



FIGURE 2.4: henri fauld

2.1.3 Traitement informatique des données relatives aux empreintes digitales

La technologie de tomodynamométrie, développée dans les années 1980, permet d'identifier les empreintes latentes difficiles. La lophoscopie étudie les motifs trouvés le long de chaque sillon grâce au scanner d'empreinte ou à des fiches de qualité.

Jusque dans les années 1980, les policiers doivent recouper manuellement des milliers de fiches cartonnées réparties dans différents fichiers régionaux.

Ainsi, en France, pour l'affaire du juge Michel, il faut plusieurs mois pour trouver à qui appartiennent les traces laissées sur une moto. En 1987, lors de l'affaire Thierry Paulin, on s'aperçoit après l'arrestation de celui-ci que ses empreintes, bien que répertoriées dans un fichier de la police de Toulouse, n'avaient pas été comparées. C'est cette dernière affaire qui accélère la création du Fichier automatisé des empreintes digitales (FAED) qui est institué par le décret du 8 avril 1987. Ce fichier est géré par la police scientifique. Depuis la loi d'orientation pour la sécurité intérieure de 2002, il s'étend aux empreintes palmaires. Ce fichier est issu du logiciel Morpho System de SAGEM22, qui automatise les photographies, la numérisation, le traitement et la comparaison des empreintes digitales et palmaires. Le principe de reconnaissance d'empreinte palmaire reste identique : un logiciel quadrille la paume de la main en seize zones de la taille d'une empreinte digitale. Comme pour la reconnaissance des empreintes digitales, il faut, pour qu'un résultat soit jugé positif en France, qu'au moins douze points d'une trace correspondent parfaitement à ceux d'une empreinte recensée dans le FAED, sans aucun point de discordance non explicable. Le 20 janvier 2010, une première est réalisée par la police technique et scientifique : un voleur est démasqué par ses empreintes palmaires²³. Autorisé le 28 octobre 2016 par décret, le Fichier des titres électroniques sécurisés (TES) inclut les empreintes digitales de tous les détenteurs d'une carte nationale d'identité ou d'un passeport français²⁴.

2.1.4 Agencement des traces digitales

L'agencement des empreintes digitales soulève un paradoxe : leur forme est spécifique d'un individu, mais elles se ressemblent beaucoup dans leurs structures. En effet, on définit trois motifs partagés par 95 pour cent de la population : la boucle (60 % des cas), la spirale (30 % des cas, appelée aussi verticilles, spires ou tourbillons) et l'arche, plus rare (5 % des cas, appelée aussi arc ou tente)^{25,1}. Les cinq pour cent restants appartiennent à une catégorie plus complexe d'agencements avec de multiples boucles¹.

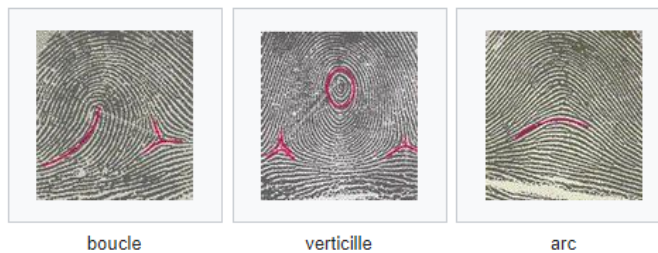


FIGURE 2.5: motifs d'empreintes digitales

Ainsi on parle d'adelte, de bidelte, de tridelte (rare). Les monodeltes se divisent en sous-groupes : les normales, externes, composites. De même pour les bideltes : à verticilles concentriques, à verticilles en « z »... [pas clair]

On différencie les motifs entre eux à l'aide de points singuliers sur les boucles, verticilles ou arcs :

points singuliers globaux : noyau ou centre : lieu de convergences des stries ; delta : lieu de divergences des stries ; points singuliers locaux (appelés aussi minuties) : points d'irrégularité se trouvant sur les lignes papillaires (terminaisons, bifurcations, îlots-assimilé à deux terminaisons, lacs).

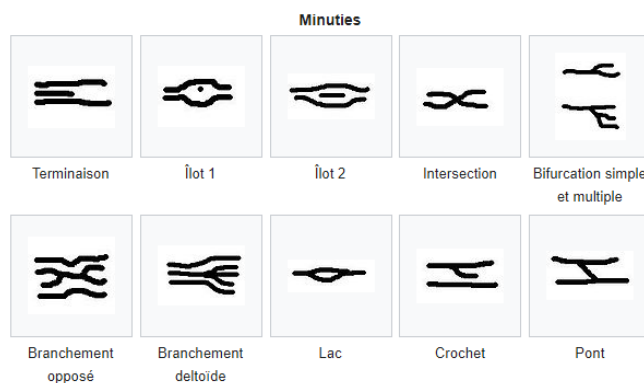


FIGURE 2.6: les minuties des empreintes digitales

On estime qu'il y a plus de cent points de convergence entre deux empreintes identiques. Les

points de convergence sont des irrégularités sur les lignes papillaires. En France, il est d'usage d'utiliser douze points (appelés minuties)²⁶ relevés sans contrariété pour authentifier l'empreinte d'un suspect. Entre huit et dix points, une forte présomption est établie grâce à des algorithmes. En Suisse, un système probabiliste est utilisé dans les comparaisons : on calcule la probabilité du dessin en se basant sur les statistiques d'apparition des différentes minuties : îlots, divisions. . .

La probabilité que deux personnes aient la même empreinte digitale est un sur 64 milliards²⁷, ce qui est très faible à l'échelle de la population humaine. De plus, son caractère aléatoire s'affranchit des risques de ressemblance entre individus partageant un même patrimoine génétique : des individus monozygotes comme des jumeaux ou des quadruplés par exemple auront chacun un jeu d'empreintes digitales qui leur sera propre et différent de celui des autres individus de la même fratrie, de même pour les empreintes légèrement différentes entre la main gauche et la main droite.

En effet, les gènes sont responsables de l'architecture générale des empreintes (les trois grands motifs) alors que le développement embryonnaire et l'environnement de la vie intra-utérine, différents d'un fœtus à l'autre, influent sur les points singuliers²⁸. Parmi ces facteurs externes, on recense, les forces de frottement des doigts en cours de formation sur divers éléments, tels le liquide amniotique et les structures utérines, la formation osseuse sous-jacente (la taille et la vitesse de croissance de l'os), le suçage du pouce in utero ou encore les mouvements des mains modèlent l'épiderme¹. L'environnement extérieur lui-même joue un rôle majeur. Par exemple, plusieurs études suggèrent que des stress chimiques (l'exposition à des agents toxiques, l'alcool), biologiques (des infections virales ou bactériennes) ou physiologiques (l'hypertension, le manque d'oxygène), mais aussi psychologiques, augmentent l'altération des dessins digitaux (interruption des crêtes, arête dédoublée)¹. À ce titre, les empreintes digitales peuvent être considérées comme des marques indélébiles de ce qui s'est produit au cours du développement entre la 10^e et la 24^e semaines de gestation¹.

2.1.5 Applications et restrictions

Par défaut, l'empreinte de l'index gauche est utilisée pour établir une carte d'identité française³⁰. Les scanners d'empreintes, autrefois utilisés uniquement pour les systèmes de fermeture des énormes coffres bancaires, deviennent à présent des éléments de sécurité intégrés par exemple sur des ordinateurs portables, des téléphones ou des guichets automatiques.

Certains ordinateurs et smartphones sont pourvus de lecteurs d'empreintes, permettant d'éviter la saisie de mot de passe.

Le 26 septembre 2008, la CNIL refuse les dispositifs biométriques de reconnaissance d'empreintes digitales à des fins de contrôle d'accès (et de présence des élèves) des établissements scolaires³¹. De manière générale, la CNIL restreint l'utilisation biométrique des empreintes digitales dans les entreprises, qui n'est acceptée qu'en présence de « fort impératif de sécurité » lorsque les empreintes sont stockées sur un système informatique central³². En revanche, elle se

montre beaucoup plus tolérante lorsque les empreintes sont stockées sur des supports individuels (carte magnétique, carte à puce, clé USB, etc.).

En effet, la CNIL indique qu'il est facile de prélever à l'insu de la personne concernée ses empreintes digitales, et donc d'usurper son identité. Ceci est d'autant plus facile si les empreintes sont stockées dans des bases de données, vulnérables à l'indiscrétion éventuelle d'employés ou au piratage informatique.



FIGURE 2.7: scanner d'identification d'empreinte digital



FIGURE 2.8: Un guichet automatique pour certificat administratif à authentification basée sur les empreintes digitales

2.1.6 Problématique

Essayer de résoudre le problème de l'identification des empreintes digitales en utilisant la technologie d'apprentissage en profondeur pour mieux protéger nos données et nos applications.

2.1.7 Conclusion

Nous avons appris l'histoire de la science des empreintes digitales et comment elle a été découverte, et les scientifiques qui ont aidé à découvrir ses propriétés, et nous avons conclu que l'empreinte digitale a un élément de sécurité merveilleux, qui est la possibilité de similitude entre les empreintes digitales des individus, qui atteint 1 dans 64 milliards, et c'est ce qui a fait de l'empreinte digitale l'outil le plus important pour identifier les individus et utilisé par toutes les Institutions, banques et pays, et cette technologie a pris d'autres dimensions après le grand développement dans le domaine de la vision par ordinateur récemment.

CONVOLUTIONAL NEURAL NETWORK(CNN)

3.1 Introduction

Au cours des dernières années de l'industrie informatique, il y a eu une énorme demande pour un ensemble de compétences particulières connu sous le nom de Deep Learning. Deep Learning un sous-ensemble de Machine Learning qui se compose d'algorithmes inspirés du fonctionnement du cerveau humain ou des réseaux de neurones.

Ces structures sont appelées réseaux de neurones. Il apprend à l'ordinateur à faire ce qui vient naturellement aux humains. Apprentissage profond, il existe plusieurs types de modèles tels que les réseaux de neurones artificiels (ANN), les auto-encodeurs, les réseaux de neurones récurrents (RNN) et l'apprentissage par renforcement. Mais il y a eu un modèle particulier qui a beaucoup contribué dans le domaine de la vision par ordinateur et de l'analyse d'images, ce sont les réseaux de neurones convolutifs (CNN) ou les ConvNets.

Les CNN sont une classe de réseaux de neurones profonds capables de reconnaître et de classer des caractéristiques particulières à partir d'images et sont largement utilisés pour analyser des images visuelles. Leurs applications vont de la reconnaissance d'images et de vidéos, à la classification d'images, à l'analyse d'images médicales, à la vision par ordinateur et au traitement du langage naturel.

Le terme "Convolution" dans CNN désigne la fonction mathématique de la convolution qui est un type spécial d'opération linéaire dans laquelle deux fonctions sont multipliées pour produire une troisième fonction qui exprime comment la forme d'une fonction est modifiée par l'autre. En termes simples, deux images qui peuvent être représentées sous forme de matrices sont multipliées pour donner une sortie qui est utilisée pour extraire des caractéristiques de l'image.

3.1.1 Historique du réseau de neurones convolutifs

Dans les années 1950 et 1960, l'Américain David Hubel et le Suédois Torsten Wiesel ont commencé à étudier le système visuel des chats et des singes à la Johns Hopkins School of Medicine. Ils ont publié une série d'articles présentant la théorie selon laquelle les neurones du cortex visuel sont chacun limités à des parties particulières du champ visuel. Ils ont en outre postulé que le traitement visuel se déroule en cascade, des neurones dédiés aux formes simples vers les neurones qui captent des motifs plus complexes. Leurs découvertes leur ont valu le prix Nobel 1981 de physiologie ou médecine. En 1980, l'informaticien japonais Kunihiko Fukushima invente le neocognitron, sorte de réseau de neurones constitué de couches convolutives et de couches de sous-échantillonnage, en s'inspirant des découvertes de Hubel et Wiesel. Le neocognitron pourrait effectuer certaines tâches de traitement d'image de base telles que la reconnaissance de caractères. De 1987 à 1990, de nombreux chercheurs, dont Alex Waibel et Kouichi Yamaguchi, ont encore adapté le neocognitron, en introduisant des innovations telles que la rétropropagation qui ont facilité l'entraînement. Puis, en 1998, Yann LeCun a développé LeNet, un réseau de neurones convolutifs à cinq couches convolutives capable de reconnaître les chiffres manuscrits du code postal avec une grande précision. Une autre étape importante a été le réseau de neurones convolutifs AlexNet du doctorant ukrainien-canadien Alex Krizhevsky, publié en 2012. AlexNet a battu tous les autres candidats au concours de reconnaissance d'images ImageNet de plus de 10 points de pourcentage, marquant le début d'une nouvelle ère dans la vision par ordinateur. Jusqu'en 2015 environ, les tâches d'image telles que la reconnaissance faciale étaient généralement effectuées au moyen de programmes codés à la main laborieux qui capturaient les traits du visage tels que les sourcils et le nez. De nombreuses applications d'OCR ou de reconnaissance faciale n'utilisaient pas du tout l'apprentissage automatique. L'accélération rapide de la puissance de calcul et la grande disponibilité de grands ensembles de données, de GPU et de logiciels d'apprentissage en profondeur ont signifié que vers le milieu des années 2010, les réseaux de neurones convolutifs étaient capables d'offrir une bien meilleure précision que les méthodes traditionnelles et sont soudainement devenus la norme pour près de toutes les tâches liées à la vision par ordinateur dans les universités et l'industrie. Maintenant, l'utilisateur moyen de smart-phone a probablement une ou deux applications exécutant des réseaux de neurones convolutifs dans sa poche, un concept qui aurait été impensable en 2010.

3.1.2 Conception de réseau de neurones convolutifs

L'architecture d'un réseau de neurones convolutifs est un réseau de neurones à anticipation multicouche, constitué en empilant de nombreuses couches cachées les unes sur les autres en séquence. C'est cette conception séquentielle qui permet aux réseaux de neurones convolutifs d'apprendre des caractéristiques hiérarchiques.

Les couches cachées sont généralement des couches convolutives suivies de couches d'activation, certaines d'entre elles suivies de couches de mise en commun.

Un simple réseau de neurones convolutifs qui aide à comprendre les principes de conception de base est le premier réseau de neurones convolutifs LeNet-5, publié par Yann LeCun en 1998. LeNet est capable de reconnaître les caractères manuscrits.

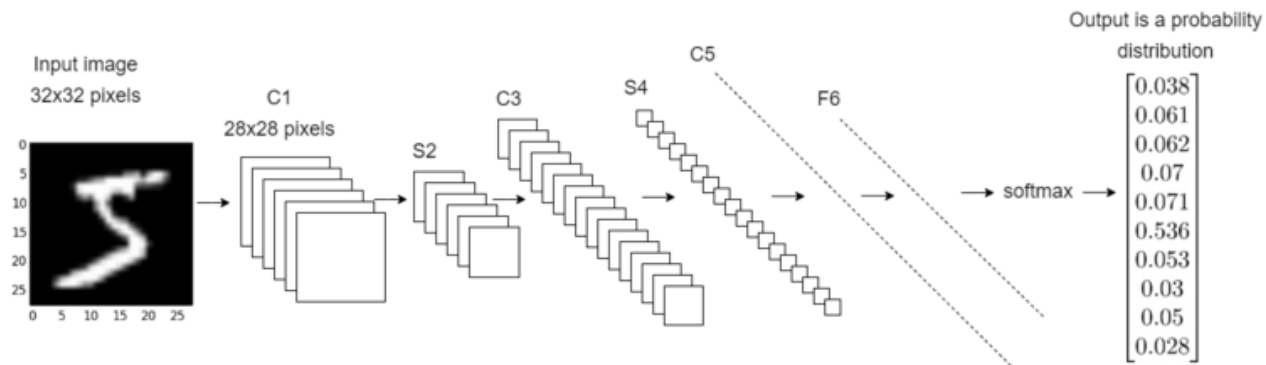


FIGURE 3.1: réseaux de neurones convolution

3.1.3 Exemple de couches de réseau neuronal convolutif expliquées

LeNet prend une image d'entrée d'un chiffre manuscrit d'une taille de 32x32 pixels et la fait passer à travers une pile des couches suivantes. Chaque couche, à l'exception de la dernière, est suivie d'une fonction d'activation de tanh.

| | |
|--------------|---|
| C1 | The first convolutional layer. This consists of six convolutional kernels of size 5x5, which "walk over" the input image. C1 outputs six images of size 28x28. The first layer of a convolutional neural network normally identifies basic features such as straight edges and corners. |
| S2 | A subsampling layer, also known as an average pooling layer. Each square of four pixels in the C1 output is averaged to a single pixel. S2 scales down the six 28x28 images by a factor of 2, producing six output images of size 14x14. |
| C3 | The second convolutional layer. This consists of 16 convolutional kernels, each of size 5x5, which take the six 14x14 images and walk over them again, producing 16 images of size 10x10. |
| S4 | The second average pooling layer. S4 scales down the sixteen 10x10 images to sixteen 5x5 images. |
| C5 | A fully connected convolutional layer with 120 outputs. Each of the 120 output nodes is connected to all of the 400 nodes (5x5x16) that came from S4. At this point the output is no longer an image, but a 1D array of length 120. |
| F6 | A fully connected layer mapping the 120-array to a new array of length 10. Each element of the array now corresponds to a handwritten digit 0-9. |
| Output Layer | A softmax function which transforms the output of F6 into a probability distribution of 10 values which sum to 1. |

FIGURE 3.2: exemple de couche de réseau neuronal convolutif

LeNet-5 est l'un des réseaux de neurones convolutifs les plus simples, avec six couches. Cela lui donne assez de puissance pour distinguer les petits chiffres manuscrits mais pas, par exemple, les 26 lettres de l'alphabet, et surtout pas les visages ou les objets. Aujourd'hui, les réseaux les plus sophistiqués peuvent avoir plus de 30 couches et des millions de paramètres, et impliquent également des branchements, mais les blocs de construction de base des noyaux convolutifs restent les mêmes.

3.2 les couches qui composent le CNN

3.2.1 Couche convolutive

Le bloc de construction clé dans un réseau de neurones convolutifs est la couche convolutive. Nous pouvons visualiser une couche convolutive comme autant de petits gabarits carrés, appelés noyaux convolutifs, qui glissent sur l'image et recherchent des motifs. Lorsque cette partie de l'image correspond au modèle du noyau, le noyau renvoie une grande valeur positive, et lorsqu'il n'y a pas de correspondance, le noyau renvoie zéro ou une valeur inférieure.

[Goodfellow, Ian, Yoshua Bengio, and Aaron Courville.].

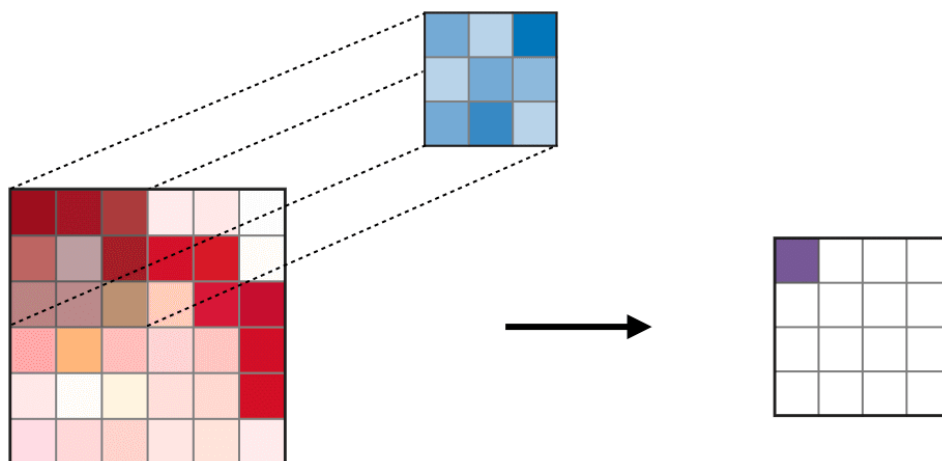


FIGURE 3.3: exemple de convolution

3.2.1.1 Exemple de calcul d'une convolution sur une matrice

Mathématiquement, le noyau est une matrice de poids. Par exemple, le noyau 3x3 suivant détecte les lignes verticales.

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

FIGURE 3.4: filter ou kernel de convolution

Imaginons une image d'entrée 9x9 d'un signe plus. Cela a deux types de lignes, horizontales et verticales, et un croisement. Au format matriciel, l'image ressemblerait à ceci :

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

FIGURE 3.5: image sous form pixels

Imaginez que nous voulions tester le noyau du détecteur de ligne verticale sur l'image du signe plus. Pour effectuer la convolution, nous glissons le noyau de convolution sur l'image. A chaque position, nous multiplions chaque élément du noyau de convolution par l'élément de l'image qu'il recouvre, et additionnons les résultats.

Comme le noyau a une largeur de 3, il ne peut être positionné qu'à 7 positions différentes horizontalement dans une image de largeur 9. Ainsi le résultat final de l'opération de convolution sur une image de taille 9x9 avec un noyau de convolution 3x3 est une nouvelle image de taille 7x7 .

Ainsi, dans l'exemple ci-dessus, le noyau est d'abord placé dans le coin supérieur gauche et chaque élément du noyau est multiplié par chaque élément dans la case rouge en haut à gauche de l'image d'origine. Étant donné que ces valeurs sont toutes 0, le résultat pour cette cellule est 0 en haut à gauche de la matrice de sortie.

Considérons maintenant la position de la boîte bleue dans l'exemple ci-dessus. Il contient une partie d'une ligne verticale. Lorsque le noyau est placé sur cette ligne verticale, il correspond et renvoie 3.

Rappelons que ce noyau de convolution est un détecteur de ligne verticale. Pour les parties de l'image originale qui contenaient une ligne verticale, le noyau a renvoyé une valeur 3, alors qu'il a renvoyé une valeur de 1 pour la ligne horizontale, et 0 pour les zones vides de l'image.

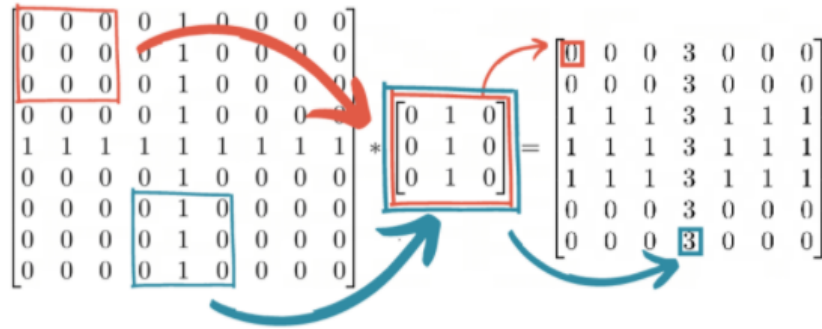


FIGURE 3.6: exemple de convolution avec un filter de 3x3

En pratique, un noyau de convolution contient à la fois des poids et des biais, similaire à la formule de régression linéaire. Ainsi, un pixel d'entrée est multiplié par le poids, puis le biais est ajouté.

3.2.1.2 Exemple de convolution sur une image

Considérons le noyau de convolution 9x9 suivant, qui est un détecteur de lignes verticales légèrement plus sophistiqué que le noyau utilisé dans le dernier exemple :

$$\begin{bmatrix} 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \end{bmatrix}$$

FIGURE 3.7: exemple de convolution sur un image

et nous pouvons prendre l'image suivante d'un chat tigré de dimensions 204x175, que nous pouvons représenter comme une matrice avec des valeurs comprises entre 0 et 1, où 1 est blanc et 0 est noir.

En appliquant la convolution, nous constatons que le filtre a effectué une sorte de détection de ligne verticale. Les rayures verticales sur la tête du chat tigré sont mises en évidence dans la sortie. L'image de sortie est 8 pixels plus petite dans les deux dimensions en raison de la taille du noyau (9x9).

Malgré sa simplicité, la capacité de détecter des lignes verticales ou horizontales, des coins, des courbes et d'autres caractéristiques simples, est une propriété extrêmement puissante du noyau de convolution. On rappelle qu'une couche convolutive est constituée d'une série de noyaux de convolution. Typiquement, la première couche d'un réseau neuronal convolutif contient un détecteur de ligne verticale, un détecteur de ligne horizontale et divers détecteurs de diagonale,

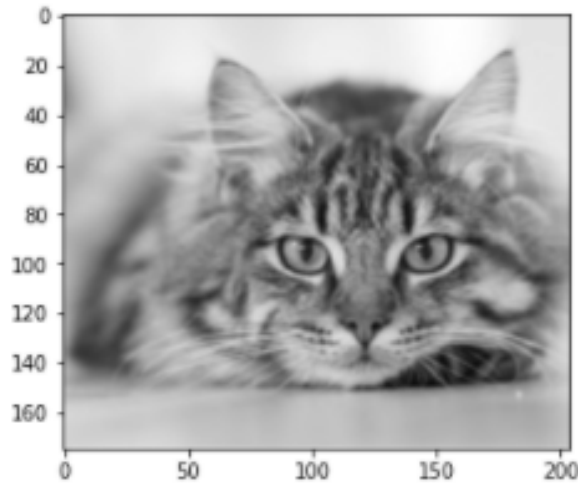


FIGURE 3.8: un chat

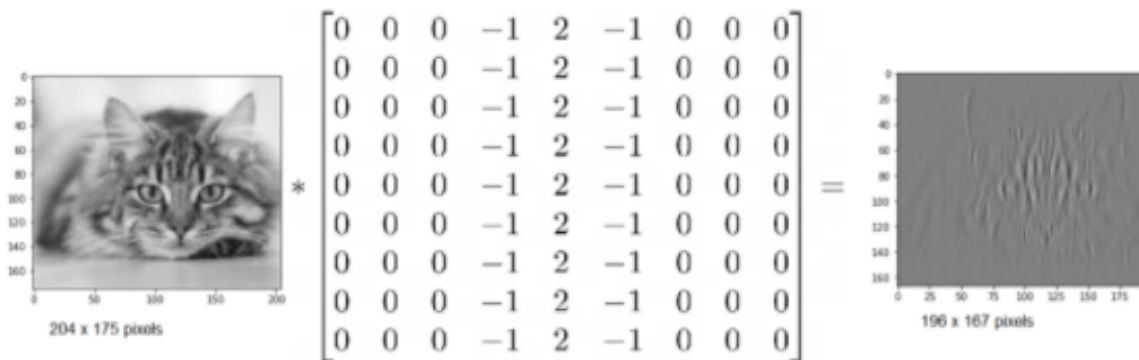


FIGURE 3.9: exemple de convolution sur un chat

de courbe et de coin. Ces noyaux de détecteurs de caractéristiques ne sont pas programmés par un humain mais sont en fait appris par le réseau de neurones lors de l'apprentissage et constituent la première étape du processus de reconnaissance d'images.

Les couches ultérieures du réseau de neurones sont capables de s'appuyer sur les caractéristiques détectées par les couches précédentes et d'identifier des formes de plus en plus complexes.

[LO, Shih-Chung B., CHAN, Heang-Ping, LIN, Jyh-Shyan, et al].

3.2.1.3 Paramètres du filtre

La couche convolutionnelle contient des filtres pour lesquels il est important de savoir comment ajuster ses paramètres.

3.2.1.4 Dimensions d'un filtre

Un filtre de taille F times F times C appliqué à une entrée contenant C canaux est un volume de taille F times F times C qui effectue des convolutions sur une entrée de taille I times I times C et qui produit un feature map de sortie (aussi appelé activation map) de taille O times O times 1 .

[Torrey, Lisa, and Jude Shavlik. "Transfer learning."].

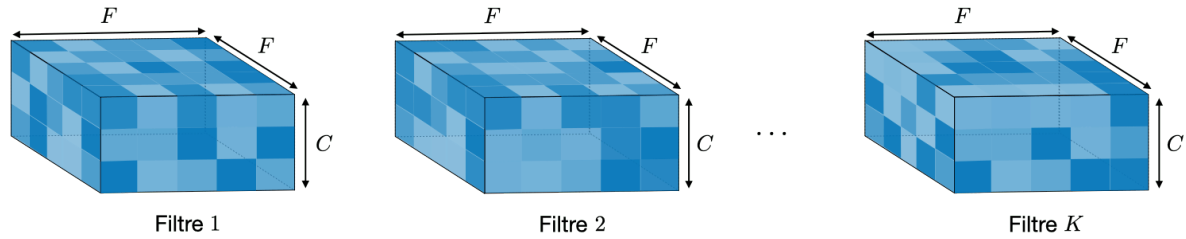


FIGURE 3.10: dimensions d'un filter

Remarque : appliquer K filtres de taille F times F times C engendre un feature map de sortie de taille O times O times K .

3.2.1.5 Stride

Dans le contexte d'une opération de convolution ou de pooling, la stride S est un paramètre qui dénote le nombre de pixels par lesquels la fenêtre se déplace après chaque opération.

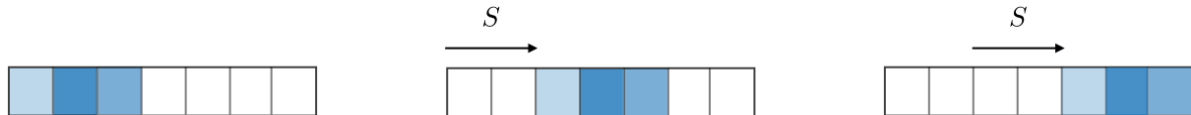


FIGURE 3.11: stride

3.2.1.6 Zero-padding

Le zero-padding est une technique consistant à ajouter PP zeros à chaque côté des frontières de l'entrée. Cette valeur peut être spécifiée soit manuellement, soit automatiquement par le biais d'une des configurations détaillées ci-dessous :

3.2.2 Réglage des paramètres

3.2.2.1 Compatibilité des paramètres dans la couche convolutionnelle

En notant I le côté du volume d'entrée, F la taille du filtre, PP la quantité de zero-padding, S la stride, la taille O de la feature map de sortie suivant cette dimension est telle que :

[Torrey, Lisa, and Jude Shavlik. "Transfer learning."].

3.2. LES COUCHES QUI COMPOSENT LE CNN

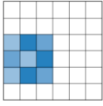
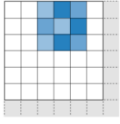
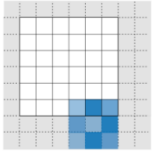
| Configuration | Valide | Pareil | Total |
|---------------|--|--|---|
| Valeur | $P = 0$ | $P_{start} = \left\lfloor \frac{S \lfloor \frac{I}{S} \rfloor - I + F - S}{2} \right\rfloor$ $P_{end} = \left\lceil \frac{S \lfloor \frac{I}{S} \rfloor - I + F - S}{2} \right\rceil$ | $P_{start} \in [0, F - 1]$ $P_{end} = F - 1$ |
| Illustration |  |  |  |
| But | <ul style="list-style-type: none"> • Pas de padding • Enlève la dernière opération de convolution si les dimensions ne collent pas | <ul style="list-style-type: none"> • Le padding tel que la feature map est de taille $\left\lfloor \frac{I}{S} \right\rfloor$ • La taille de sortie est mathématiquement satisfaisante • Aussi appelé 'demi' padding | <ul style="list-style-type: none"> • Padding maximum tel que les dernières convolutions sont appliquées sur les bords de l'entrée • Le filtre 'voit' l'entrée du début à la fin |

FIGURE 3.12: réglage de parametres

$$O = \frac{I - F + P_{start} + P_{end}}{S} + 1$$

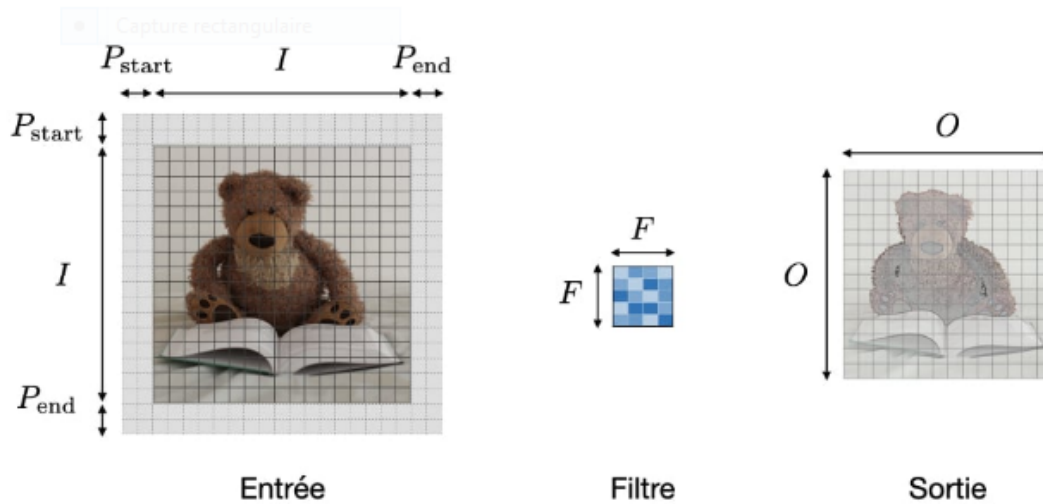


FIGURE 3.13: réglage de parametres

3.2.3 Max Pooling(couche de mise en commun)

Dans la plupart des cas, une couche convolutive est suivie d'une couche de mise en commun. L'objectif principal de cette couche est de réduire la taille de la carte des caractéristiques convoluées afin de réduire les coûts de calcul. Ceci est effectué en diminuant les connexions entre les couches et opère indépendamment sur chaque carte de caractéristiques. Selon la méthode utilisée, il existe plusieurs types d'opérations de mise en commun.

Dans Max Pooling, le plus grand élément est extrait de la carte des caractéristiques. Average

Pooling calcule la moyenne des éléments dans une section Image de taille prédéfinie. La somme totale des éléments de la section prédéfinie est calculée dans Sum Pooling. La couche de mise en commun sert généralement de pont entre la couche convolutive et la couche Fully Connected.

| Type | Max pooling | Average pooling |
|---------------------|--|--|
| But | Chaque opération de pooling sélectionne la valeur maximale de la surface | Chaque opération de pooling sélectionne la valeur moyenne de la surface |
| Illustration | | |
| Commentaires | <ul style="list-style-type: none"> • Garde les caractéristiques détectées • Plus communément utilisé | <ul style="list-style-type: none"> • Sous-échantillonne la <i>feature map</i> • Utilisé dans LeNet |

FIGURE 3.14: la couche pooling

3.2.4 Fully connected (couche entièrement connectée)

La couche entièrement connectée (FC) comprend les poids et les biais ainsi que les neurones et est utilisée pour connecter les neurones entre deux couches différentes. Ces couches sont généralement placées avant la couche de sortie et forment les dernières couches d'une architecture CNN.

Dans ce cas, l'image d'entrée des couches précédentes est aplatie et transmise à la couche FC. Le vecteur aplati subit ensuite quelques couches FC supplémentaires où les opérations des fonctions mathématiques ont généralement lieu. À ce stade, le processus de classification commence à avoir lieu.

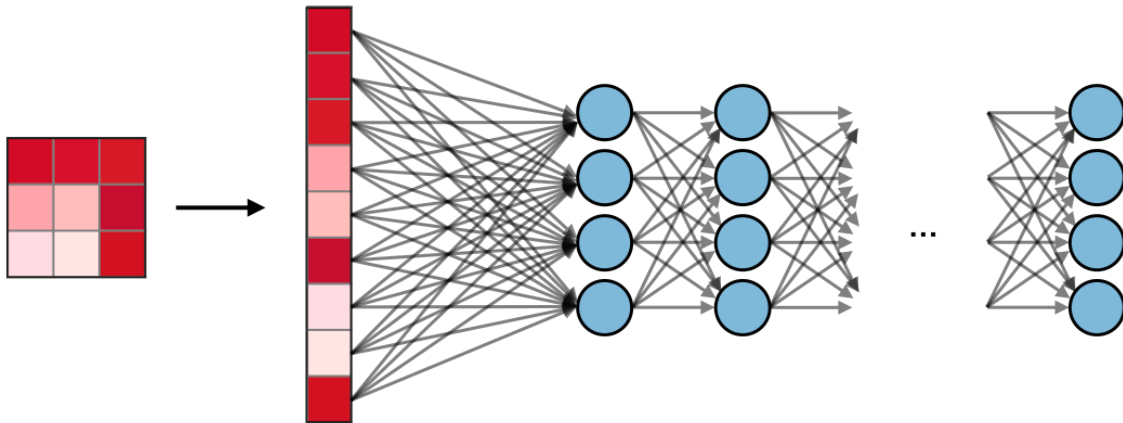


FIGURE 3.15: fully connected

3.2.4.1 Comprendre la complexité du modèle

Pour évaluer la complexité d'un modèle, il est souvent utile de déterminer le nombre de paramètres que l'architecture va avoir. Dans une couche donnée d'un réseau de neurones convolutifs, on a :

| | CONV | POOL | FC |
|----------------------|---|---|---|
| Illustration | | | |
| Taille d'entrée | $I \times I \times C$ | $I \times I \times C$ | N_{in} |
| Taille de sortie | $O \times O \times K$ | $O \times O \times C$ | N_{out} |
| Nombre de paramètres | $(F \times F \times C + 1) \cdot K$ | 0 | $(N_{in} + 1) \times N_{out}$ |
| Remarques | <ul style="list-style-type: none"> • Un paramètre de biais par filtre • Dans la plupart des cas, $S < F$ • $2C$ est un choix commun pour K | <ul style="list-style-type: none"> • L'opération de pooling est effectuée pour chaque canal • Dans la plupart des cas, $S = F$ | <ul style="list-style-type: none"> • L'entrée est aplatie • Un paramètre de biais par neurone • Le choix du nombre de neurones de FC est libre |

FIGURE 3.16: la complexité du modèle

3.2.5 Dropout

Généralement, lorsque toutes les entités sont connectées à la couche FC, cela peut entraîner un surapprentissage dans l'ensemble de données d'apprentissage. Le surapprentissage se produit lorsqu'un modèle particulier fonctionne si bien sur les données d'entraînement, ce qui a un impact négatif sur les performances du modèle lorsqu'il est utilisé sur de nouvelles données.

Pour surmonter ce problème, une couche de suppression est utilisée dans laquelle quelques neurones sont supprimés du réseau neuronal pendant le processus d'apprentissage, ce qui réduit la taille du modèle. En passant un abandon de 0,3, 30% des nœuds sont abandonnés au hasard du réseau de neurones.

3.2.5.1 Qu'est-ce que le Dropout dans les réseaux de neurones ?

Le terme « abandon » fait référence à l'abandon d'unités (à la fois cachées et visibles) dans un réseau de neurones. En termes simples, l'abandon fait référence au fait d'ignorer les unités (c'est-à-dire les neurones) pendant la phase d'apprentissage d'un certain ensemble de neurones qui est choisi au hasard. Par « ignorer », je veux dire que ces unités ne sont pas prises en compte lors d'une passe avant ou arrière particulière. Plus techniquement, à chaque étape d'apprentissage, les nœuds individuels sont soit retirés du réseau avec une probabilité $1-p$, soit conservés avec une probabilité p , de sorte qu'il reste un réseau réduit ; les bords entrants et sortants vers un nœud abandonné sont également supprimés.

3.2.5.2 Pourquoi avons-nous besoin de Dropout ?

La réponse à ces questions est « éviter le sur-ajustement ». Une couche entièrement connectée occupe la plupart des paramètres et, par conséquent, les neurones développent une co-dépendance entre eux pendant l'entraînement, ce qui réduit la puissance individuelle de chaque neurone, ce qui entraîne un surajustement des données d'entraînement.

3.2.5.3 Mode d'action

Dans cette section, je vais aborder un peu plus de technicité. En apprentissage automatique, la régularisation est un moyen d'éviter le surajustement. La régularisation réduit le surajustement en ajoutant une pénalité à la fonction de perte. En ajoutant cette pénalité, le modèle est entraîné de telle sorte qu'il n'apprenne pas un ensemble interdépendant de poids de caractéristiques. Ceux d'entre vous qui connaissent la régression logistique connaissent peut-être les pénalités L1 (laplacienne) et L2 (gaussienne). Dropout est une approche de régularisation dans les réseaux de neurones qui aide à réduire l'apprentissage interdépendant entre les neurones. Phase d'apprentissage : pour chaque couche cachée, pour chaque échantillon d'apprentissage, pour chaque itération, ignorez (remettez à zéro) une fraction aléatoire, p , de nœuds (et les activations correspondantes). Phase de test : Utilisez toutes les activations, mais réduisez-les d'un facteur p (pour tenir compte

des activations manquantes pendant l'entraînement). Quelques remarques : 1-Dropout force un

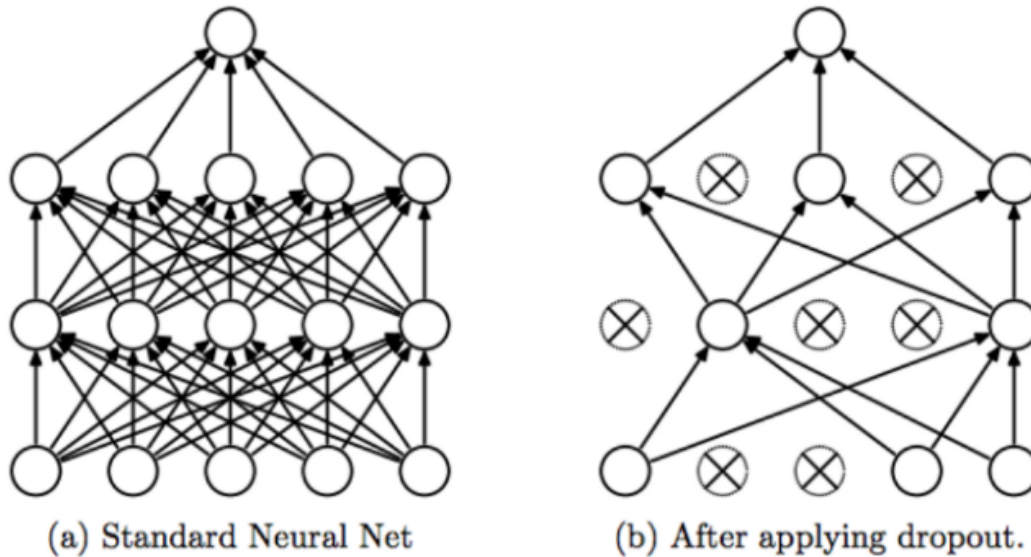


FIGURE 3.17: dropout

réseau de neurones à apprendre des fonctionnalités plus robustes qui sont utiles en conjonction avec de nombreux sous-ensembles aléatoires différents des autres neurones. 2-Dropout double à peu près le nombre d'itérations nécessaires pour converger. Cependant, le temps de formation pour chaque époque est moindre. 3-Avec H unités cachées, chacune pouvant être supprimée, nous avons 2^H modèles possibles. En phase de test, l'ensemble du réseau est considéré et chaque activation est réduite d'un facteur p

3.2.5.4 Expérience à Keras

Essayons cette théorie en pratique. Pour voir comment fonctionne le décrochage, j'ai construit un réseau profond à Keras et j'ai essayé de le valider sur l'ensemble de données CIFAR-10. Le réseau profond est constitué de trois couches de convolution de taille 64, 128 et 256 suivies de deux couches densément connectées de taille 512 et d'une couche de sortie couche dense de taille 10 (nombre de classes dans l'ensemble de données CIFAR-10). J'ai pris ReLU comme fonction d'activation pour les couches cachées et sigmoïde pour la couche de sortie (ce sont des normes, je n'ai pas beaucoup expérimenté pour les changer). De plus, j'ai utilisé la perte d'entropie croisée catégorique standard. Enfin, j'ai utilisé l'abandon dans toutes les couches et augmenté la fraction d'abandon de 0,0 (pas d'abandon du tout) à 0,9 avec un pas de 0,1 et j'ai exécuté chacun d'eux jusqu'à 20 époques. Les résultats ressemblent à ceci : À partir des graphiques ci-dessus, nous pouvons conclure qu'avec l'augmentation du décrochage, il y a une certaine augmentation de la précision de la validation et une diminution de la perte initialement avant que la tendance

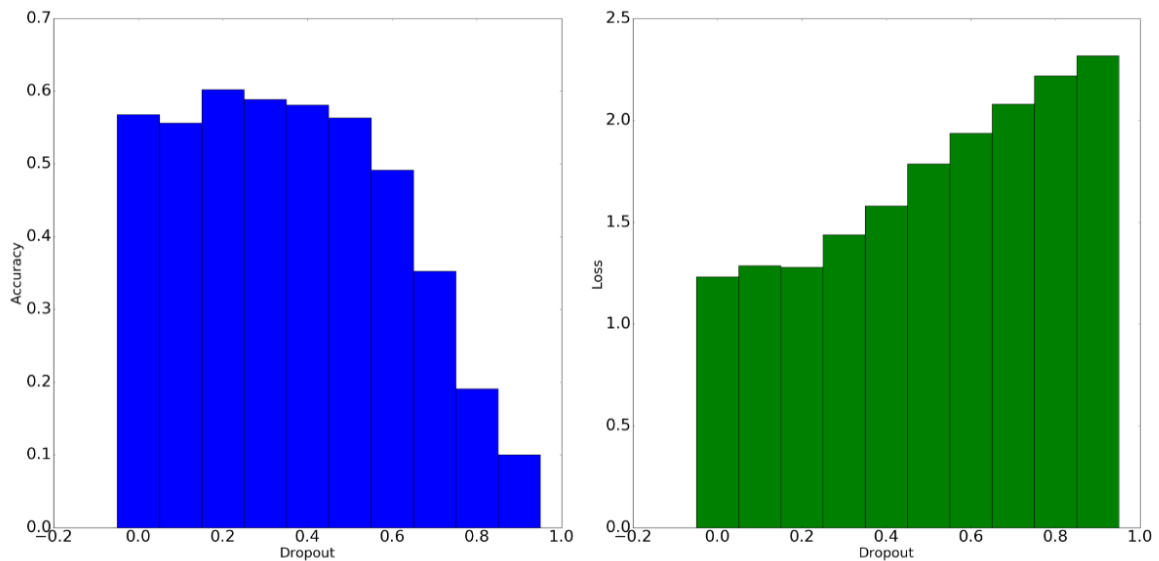


FIGURE 3.18: dropout

ne commence à baisser. Il pourrait y avoir deux raisons à la baisse de la tendance si la fraction d'abandon est de 0,2 : 0,2 est le minimum réel pour cet ensemble de données, le réseau et les paramètres définis utilisés Il faut plus d'époques pour former les réseaux.

3.2.6 Fonctions d'activation

Enfin, l'un des paramètres les plus importants du modèle CNN est la fonction d'activation. Ils sont utilisés pour apprendre et approximer tout type de relation continue et complexe entre les variables du réseau. En termes simples, il décide quelles informations du modèle doivent être déclenchées dans le sens direct et lesquelles ne doivent pas être transmises à la fin du réseau.

Il ajoute de la non-linéarité au réseau. Il existe plusieurs fonctions d'activation couramment utilisées telles que les fonctions ReLU, Softmax, tanH et Sigmoid. Chacune de ces fonctions a un usage spécifique. Pour un modèle CNN de classification binaire, les fonctions sigmoïde et softmax sont préférées et pour une classification multi-classe, généralement softmax est utilisée.

3.2.6.1 Fonction d'activation sigmoïde ou logistique

La courbe de la fonction sigmoïde ressemble à une forme de S. La principale raison pour

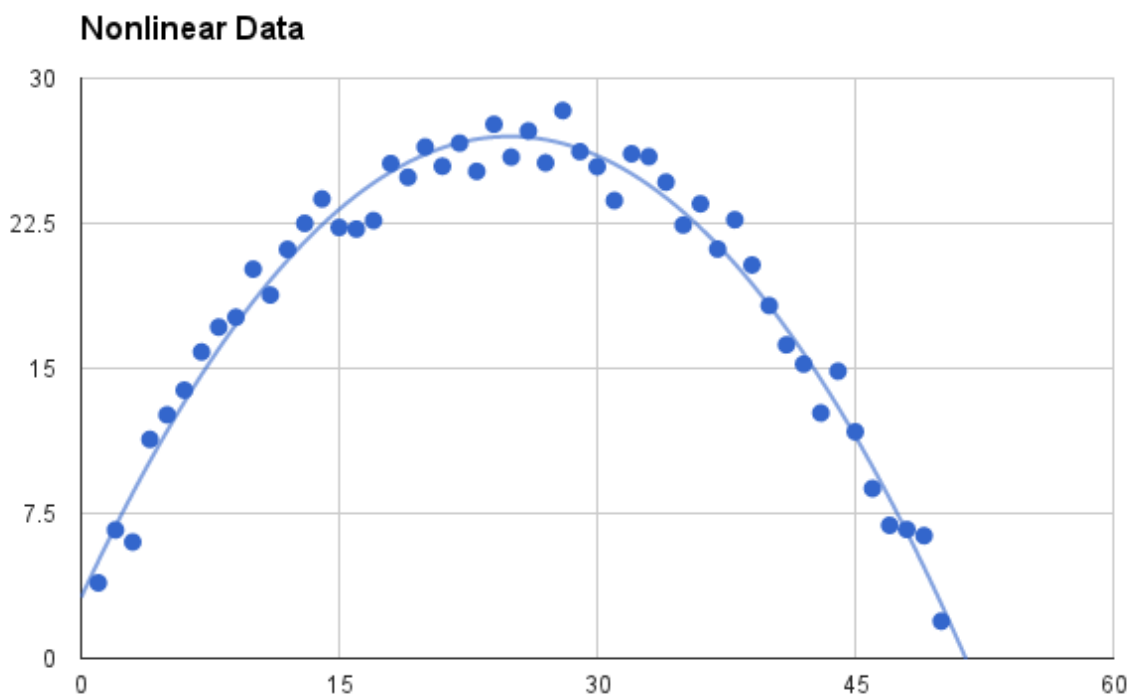


FIGURE 3.19: fonction d'activation(non linear data)

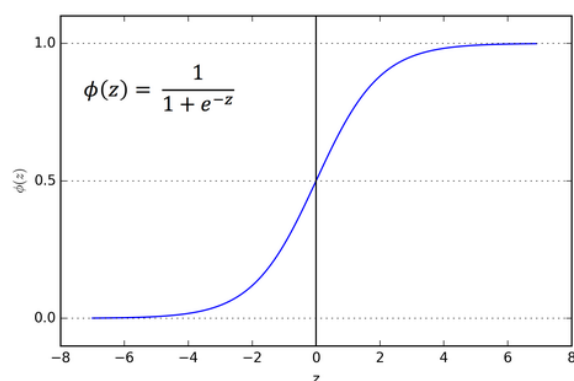


FIGURE 3.20: la fonction sigmoïde

laquelle nous utilisons la fonction sigmoïde est qu'elle existe entre (0 et 1). Par conséquent, il est particulièrement utilisé pour les modèles où nous devons prédire la probabilité en tant que sortie. Étant donné que la probabilité de quoi que ce soit n'existe qu'entre 0 et 1, sigmoïde est le bon choix. La fonction est dérivable. Cela signifie que nous pouvons trouver la pente de la courbe sigmoïde en deux points quelconques. La fonction est monotone mais la dérivée de la fonction ne l'est pas. La fonction logistique sigmoïde peut bloquer un réseau de neurones au moment de la formation. La fonction softmax est une fonction d'activation logistique plus généralisée qui est

utilisée pour la classification multiclassé.

3.2.6.2 Fonction d'activation Tanh ou tangente hyperbolique

Tanh est aussi comme sigmoïde logistique mais en mieux. La plage de la fonction tanh est de (-1 à 1). tanh est également sigmoïde (en forme de s). L'avantage est que les entrées négatives

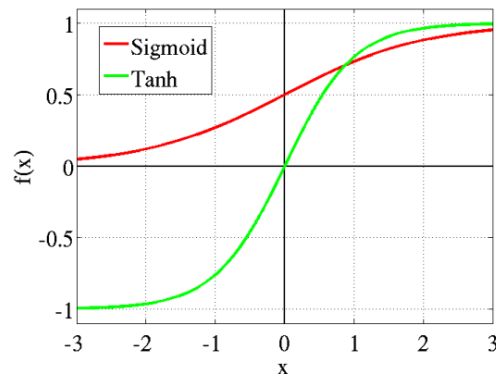


FIGURE 3.21: la fonction de tanah

seront mappées fortement négatives et les entrées zéro seront mappées près de zéro dans le graphique tanh. La fonction est dérivable. La fonction est monotone alors que sa dérivée n'est pas monotone. La fonction tanh est principalement utilisée pour la classification entre deux classes. Les fonctions d'activation tanh et logistique sigmoïde sont utilisées dans les filets feed-forward.

3.2.6.3 Fonction d'activation ReLU (Unité linéaire rectifiée)

La ReLU est actuellement la fonction d'activation la plus utilisée au monde. Depuis, elle est utilisée dans presque tous les réseaux de neurones convolutifs ou le deep learning.

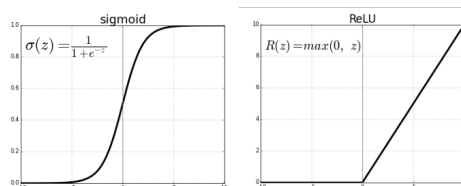


FIGURE 3.22: la fonction d'activation relu

Comme vous pouvez le voir, le ReLU est à moitié rectifié (par le bas). $f(z)$ est égal à zéro lorsque z est inférieur à zéro et $f(z)$ est égal à z lorsque z est supérieur ou égal à zéro. Plage : [0 à l'infini) La fonction et sa dérivée sont toutes les deux monotones. Mais le problème est que toutes les valeurs négatives deviennent immédiatement nulles, ce qui diminue la capacité du modèle à s'adapter ou à s'entraîner correctement à partir des données. Cela signifie que toute

entrée négative donnée à la fonction d'activation ReLU transforme immédiatement la valeur en zéro dans le graphique, ce qui à son tour affecte le graphique résultant en ne mappant pas les valeurs négatives de manière appropriée.

3.2.6.4 Leaky ReLU

C'est une tentative de résoudre le problème de la mort de ReLU. La fuite permet d'augmenter

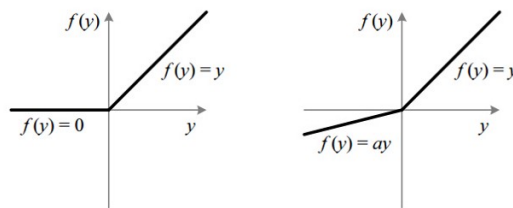


FIGURE 3.23: leaky relu

la portée de la fonction ReLU. Habituellement, la valeur de a est de 0,01 environ. Lorsque a n'est pas 0,01, il est appelé ReLU aléatoire. Par conséquent, la plage du Leaky ReLU est (-infini à l'infini). Les fonctions Leaky et Randomized ReLU sont de nature monotone. De plus, leurs dérivés sont également de nature monotone.

| ReLU | Leaky ReLU | ELU |
|---|---|---|
| $g(z) = \max(0, z)$ | $g(z) = \max(\epsilon z, z)$ with $\epsilon \ll 1$ | $g(z) = \max(\alpha(e^z - 1), z)$ with $\alpha \ll 1$ |
| | | |
| <ul style="list-style-type: none"> Complexités non-linéaires interprétables d'un point de vue biologique | <ul style="list-style-type: none"> Répond au problème de <i>dying ReLU</i> | <ul style="list-style-type: none"> Dérivable partout |

FIGURE 3.24: les fonctions d'activation

3.2.6.5 Pourquoi la dérivée/différenciation est-elle utilisée ?

Lors de la mise à jour de la courbe, pour savoir dans quelle direction et dans quelle mesure modifier ou mettre à jour la courbe en fonction de la pente. C'est pourquoi nous utilisons la

différenciation dans presque toutes les parties du Machine Learning et du Deep Learning.

| Naac | Plot | Equation | Derivative |
|--|------|--|---|
| Identity | | $f(x) = x$ | $f'(x) = 1$ |
| Binary step | | $f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$ |
| Logistic (s.k.a Soft step) | | $f(x) = \frac{1}{1 + e^{-x}}$ | $f'(x) = f(x)(1 - f(x))$ |
| Tanh | | $f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$ | $f'(x) = 1 - f(x)^2$ |
| ArcTan | | $f(x) = \tan^{-1}(x)$ | $f'(x) = \frac{1}{x^2 + 1}$ |
| Rectified Linear Unit (ReLU) | | $f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ |
| Parameteric Rectified Linear Unit (PReLU) ^[2] | | $f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ |
| Exponential Linear Unit (ELU) ^[3] | | $f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ |
| SoftPlus | | $f(x) = \log_e(1 + e^x)$ | $f'(x) = \frac{1}{1 + e^{-x}}$ |

FIGURE 3.25:

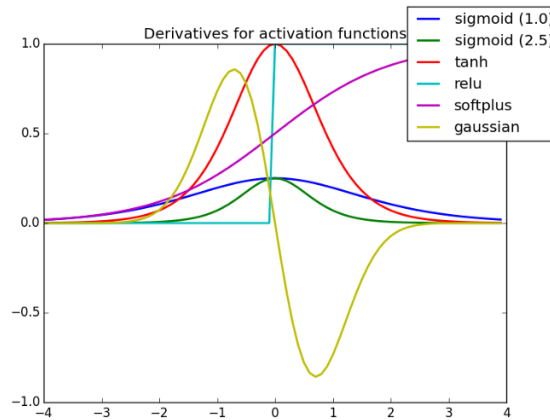


FIGURE 3.26: différenciation dans presque toutes les parties du Machine Learning et du Deep Learning.

3.2.7 Architecture CNN LeNet-5

En 1998, l'architecture LeNet-5 a été introduite dans un document de recherche intitulé « L'apprentissage basé sur le gradient appliqué à la reconnaissance de documents » par Yann

LeCun, Leon Bottou, Yoshua Bengio et Patrick Haffner. C'est l'une des architectures CNN les plus anciennes et les plus basiques.

Il se compose de 7 couches. La première couche est constituée d'une image d'entrée de dimensions 32x32. Il est convolué avec 6 filtres de taille 5x5 résultant en une dimension de 28x28x6. La deuxième couche est une opération de pooling qui filtre la taille 2x2 et la foulée de 2. Par conséquent, la dimension de l'image résultante sera de 14x14x6.

De même, la troisième couche implique également une opération de convolution avec 16 filtres de taille 5x5 suivie d'une quatrième couche de mise en commun avec une taille de filtre similaire de 2x2 et une foulée de 2. Ainsi, la dimension de l'image résultante sera réduite à 5x5x16.

Une fois la dimension de l'image réduite, la cinquième couche est une couche convolutive entièrement connectée avec 120 filtres chacun de taille 5x5. Dans cette couche, chacune des 120 unités de cette couche sera connectée aux 400 unités (5x5x16) des couches précédentes. La sixième couche est également une couche entièrement connectée avec 84 unités.

La septième couche finale sera une couche de sortie softmax avec « n » classes possibles en fonction du nombre de classes dans l'ensemble de données.

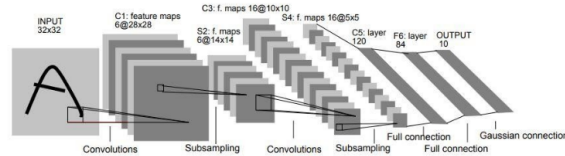


FIGURE 3.27: Lenet5

Le diagramme ci-dessus est une représentation des 7 couches de l'architecture LeNet-5 CNN.

Vous trouverez ci-dessous les instantanés du code Python pour créer une architecture LeNet-5 CNN à l'aide de la bibliothèque keras avec le framework TensorFlow.

```
#LeNet-5 CNN Architecture
model = keras.Sequential()
model.add(layers.Conv2D(filters=6, kernel_size=(5, 5), activation='relu', input_shape=(32,32,1)))
model.add(layers.AveragePooling2D())
model.add(layers.Conv2D(filters=16, kernel_size=(5, 5), activation='relu'))
model.add(layers.AveragePooling2D())
model.add(layers.Flatten())
model.add(layers.Dense(units=120, activation='relu'))
model.add(layers.Dense(units=84, activation='relu'))
model.add(layers.Dense(units=10, activation = 'softmax'))
```

FIGURE 3.28: les couches de LeNet5

En programmation Python, le type de modèle le plus couramment utilisé est le type Sequential. C'est le moyen le plus simple de créer un modèle CNN dans Keras. Il nous permet de construire un modèle couche par couche. La fonction 'add()' est utilisée pour ajouter des couches au modèle. Comme expliqué ci-dessus, pour l'architecture LeNet-5, il existe deux paires Convolution et

Pooling suivies d'une couche Flatten qui est généralement utilisée comme connexion entre Convolution et les couches Dense.

Les couches denses sont celles qui sont principalement utilisées pour les couches de sortie. L'activation utilisée est le "Softmax" qui donne une probabilité pour chaque classe et ils totalisent totalement à 1. Le modèle fera sa prédiction en fonction de la classe avec la probabilité la plus élevée.

Le résumé du modèle est affiché comme ci-dessous.

```

Model: "sequential"
-----
Layer (type)                Output Shape                Param #
-----
conv2d (Conv2D)              (None, 28, 28, 6)          156
-----
average_pooling2d (AveragePo (None, 14, 14, 6)          0
-----
conv2d_1 (Conv2D)            (None, 10, 10, 16)         2416
-----
average_pooling2d_1 (Average (None, 5, 5, 16)          0
-----
flatten (Flatten)            (None, 400)                 0
-----
dense (Dense)                 (None, 120)                 48120
-----
dense_1 (Dense)               (None, 84)                  10164
-----
dense_2 (Dense)               (None, 10)                  850
-----
Total params: 61,706
Trainable params: 61,706
Non-trainable params: 0
-----

```

FIGURE 3.29: résumé de LeNet5

3.3 Applications des réseaux de neurones convolutifs

Les réseaux de neurones convolutifs sont les plus connus pour l'analyse d'images, mais ils ont également été adaptés pour plusieurs applications dans d'autres domaines de l'apprentissage automatique, tels que le traitement du langage naturel.

3.3.1 Réseaux de neurones convolutifs pour les voitures autonomes

Plusieurs entreprises, telles que Tesla et Uber, utilisent des réseaux de neurones convolutifs comme composant de vision par ordinateur d'une voiture autonome.

Le système de vision par ordinateur d'une voiture autonome doit être capable de localiser, d'éviter les obstacles et de planifier le chemin.

Considérons le cas de la détection de piétons. Un piéton est une sorte d'obstacle qui se déplace. Un réseau de neurones convolutifs doit être capable d'identifier l'emplacement du piéton et d'extrapoler son mouvement actuel afin de calculer si une collision est imminente.

Un réseau de neurones convolutifs pour la détection d'objets est légèrement plus complexe qu'un modèle de classification, en ce sens qu'il doit non seulement classer un objet, mais également renvoyer les quatre coordonnées de sa boîte englobante.

De plus, le concepteur de réseau de neurones convolutifs doit éviter les fausses alarmes inutiles pour les objets non pertinents, tels que les déchets, mais également prendre en compte le coût élevé de la mauvaise catégorisation d'un véritable piéton et de provoquer un accident mortel.

Un défi majeur pour ce type d'utilisation est la collecte de données d'entraînement étiquetées. Le système Captcha de Google est utilisé pour s'authentifier sur des sites Web, où un utilisateur est invité à classer les images comme des bouches d'incendie, des feux de circulation, des voitures, etc. C'est en fait un moyen utile de collecter des images de formation étiquetées à des fins telles que les voitures autonomes et Google Street View.

3.3.2 Réseaux de neurones convolutifs pour la classification de texte

Bien que les réseaux de neurones convolutifs aient été initialement conçus comme un outil de vision par ordinateur, ils ont été adaptés au domaine du traitement du langage naturel avec un grand succès.

Le principe de leur utilisation sur du texte est très similaire à celui des images, à l'exception d'une étape de prétraitement. Pour utiliser un réseau de neurones convolutifs pour la classification de texte, la phrase d'entrée est segmentée puis convertie en un tableau d'inclusions de vecteurs de mots à l'aide d'une recherche telle que word2vec. Il est ensuite passé à travers un réseau de neurones convolutifs avec une couche softmax finale de la manière habituelle, comme s'il s'agissait d'une image.

Considérons un modèle qui consiste à classer la phrase « Cour suprême à envisager la publication des documents du grand jury Mueller au Congrès » dans l'une des deux catégories, « politique » ou « sport ».

Chacun des 12 mots de la phrase est converti en un vecteur, et ces vecteurs sont réunis dans une matrice. Ici, nous utilisons une taille de vecteur de mot de 5 mais en pratique, de grands nombres tels que 300 sont souvent utilisés.

$$\begin{bmatrix} 0.417 & 0.97 & 0.484 & 0.471 & 0.327 & 0.954 & 0.037 & 0.208 & 0.916 & 0.41 & 0.357 & 0.573 \\ 0.007 & 0.729 & 0.393 & 0.436 & 0.346 & 0.032 & 0.808 & 0.457 & 0.839 & 0.825 & 0.286 & 0.552 \\ 0.541 & 0.764 & 0.398 & 0.637 & 0.798 & 0.74 & 0.086 & 0.405 & 0.831 & 0.637 & 0.999 & 0.871 \\ 0.752 & 0.529 & 0.849 & 0.665 & 0.557 & 0.463 & 0.998 & 0.529 & 0.789 & 0.373 & 0.716 & 0.85 \\ 0.274 & 0.545 & 0.007 & 0.572 & 0.922 & 0.365 & 0.598 & 0.979 & 0.071 & 0.11 & 0.751 & 0.653 \end{bmatrix}$$

FIGURE 3.30: classification de texte

Cette matrice 2D peut être traitée comme une image et passée à travers un réseau neuronal convolutif régulier, qui génère une probabilité pour chaque classe et qui peut être entraînée à l'aide de la rétropropagation comme dans l'exemple « chat » contre « chien ».

Étant donné que la longueur des phrases peut varier, mais que la taille de l'image d'entrée vers un réseau doit être fixe, si une phrase est plus courte que la taille maximale, les valeurs inutilisées de la matrice peuvent être complétées par une valeur appropriée telle que des zéros.

3.3.3 Réseaux de neurones convolutifs pour la découverte de médicaments

La première étape d'un programme de développement de médicaments est la découverte de médicaments, où une société pharmaceutique identifie des composés candidats qui sont plus susceptibles d'interagir avec le corps d'une certaine manière. Tester des molécules candidates dans des essais précliniques ou cliniques est coûteux, il est donc avantageux de pouvoir cribler les molécules le plus tôt possible.

Les protéines qui jouent un rôle important dans une maladie sont appelées « cibles ». Il existe des cibles qui peuvent provoquer une inflammation ou favoriser la croissance des tumeurs. L'objectif de la découverte de médicaments est d'identifier des molécules qui interagiront avec la cible d'une maladie particulière. La molécule de médicament doit avoir la forme appropriée pour interagir avec la cible et s'y lier, comme une clé insérée dans une serrure.

La startup Atomwise basée à San Francisco a développé un algorithme appelé AtomNet, basé sur un réseau de neurones convolutifs, capable d'analyser et de prédire les interactions entre les molécules. Sans avoir appris les règles de la chimie, AtomNet a pu apprendre les interactions chimiques organiques essentielles.

Atomwise a pu utiliser AtomNet pour identifier les principaux candidats pour les programmes de recherche sur les médicaments. AtomNet a identifié avec succès un traitement candidat pour le virus Ebola, qui n'avait auparavant aucune activité antivirale connue. La molécule a ensuite fait l'objet d'essais précliniques.

3.4 Apprentissage par transfert

L'apprentissage par transfert fait généralement référence à un processus dans lequel un modèle formé sur un problème est utilisé d'une certaine manière sur un deuxième problème connexe.

Dans l'apprentissage en profondeur, l'apprentissage par transfert est une technique par laquelle un modèle de réseau de neurones est d'abord formé sur un problème similaire au problème en cours de résolution. Une ou plusieurs couches du modèle entraîné sont ensuite utilisées dans un nouveau modèle entraîné sur le problème d'intérêt. L'apprentissage par transfert a l'avantage de réduire le temps d'entraînement pour un modèle de réseau neuronal et peut entraîner une erreur de généralisation plus faible.

Les poids dans les couches réutilisées peuvent être utilisés comme point de départ pour le processus d'apprentissage et adaptés en réponse au nouveau problème. Cet usage traite l'apprentissage par transfert comme un type de schéma d'initialisation de poids. Cela peut être utile lorsque le premier problème lié a beaucoup plus de données étiquetées que le problème d'intérêt et la similitude dans la structure du problème peut être utile dans les deux contextes.

[Yang, Qiang, et al. Transfer learning.].

3.4.1 Apprentissage par transfert pour la reconnaissance d'images

Apprentissage par transfert pour la reconnaissance d'images Une gamme de modèles hautement performants a été développée pour la classification des images et démontrée lors du défi annuel de reconnaissance visuelle à grande échelle ImageNet, ou ILSVRC.

Ce défi, souvent appelé simplement ImageNet, compte tenu de la source de l'image utilisée dans le concours, a entraîné un certain nombre d'innovations dans l'architecture et la formation des réseaux de neurones convolutifs. De plus, de nombreux modèles utilisés dans les compétitions ont été publiés sous licence permissive.

Ces modèles peuvent être utilisés comme base pour l'apprentissage par transfert dans les applications de vision par ordinateur.

Ceci est souhaitable pour un certain nombre de raisons, notamment : 1) Fonctionnalités apprises utiles : les modèles ont appris à détecter des caractéristiques génériques à partir de photographies, étant donné qu'ils ont été formés sur plus de 1 000 000 d'images pour 1 000 catégories. 2) Performances de pointe : les modèles ont atteint des performances de pointe et restent efficaces pour la tâche de reconnaissance d'images spécifique pour laquelle ils ont été développés. 3) Facilement accessible : les poids des modèles sont fournis sous forme de fichiers téléchargeables gratuitement et de nombreuses bibliothèques fournissent des API pratiques pour télécharger et utiliser les modèles directement. Les pondérations du modèle peuvent être téléchargées et utilisées dans la même architecture de modèle à l'aide d'une gamme de différentes bibliothèques d'apprentissage en profondeur, y compris Keras.

3.4.2 Comment utiliser des modèles pré-entraînés

L'utilisation d'un modèle pré-entraîné n'est limitée que par votre créativité.

Par exemple, un modèle peut être téléchargé et utilisé tel quel, tel qu'intégré dans une application et utilisé pour classer de nouvelles photographies.

Alternativement, les modèles peuvent être téléchargés et utilisés comme modèles d'extraction de caractéristiques. Ici, la sortie du modèle d'une couche antérieure à la couche de sortie du modèle est utilisée comme entrée dans un nouveau modèle de classificateur.

Rappelez-vous que les couches convolutives plus proches de la couche d'entrée du modèle apprennent des caractéristiques de bas niveau telles que des lignes, que les couches au milieu de la couche apprennent des caractéristiques abstraites complexes qui combinent les caractéristiques

de niveau inférieur extraites de l'entrée et des couches plus proches de la sortie interpréter les caractéristiques extraites dans le contexte d'une tâche de classification.

Fort de cette compréhension, un niveau de détail pour l'extraction de caractéristiques à partir d'un modèle pré-entraîné existant peut être choisi. Par exemple, si une nouvelle tâche est assez différente de la classification d'objets sur des photographies (par exemple, différente d'ImageNet), alors peut-être que la sortie du modèle pré-entraîné après les quelques couches serait appropriée. Si une nouvelle tâche est assez similaire à la tâche de classification des objets dans les photographies, alors peut-être que la sortie de couches beaucoup plus profondes dans le modèle peut être utilisée, ou même la sortie de la couche entièrement connectée avant la couche de sortie peut être utilisée. Le modèle pré-entraîné peut être utilisé comme un programme d'extraction de caractéristiques distinct, auquel cas l'entrée peut être prétraitée par le modèle ou une partie du modèle en une sortie donnée (par exemple un vecteur de nombres) pour chaque image d'entrée, qui peut puis utiliser comme entrée lors de la formation d'un nouveau modèle.

Alternativement, le modèle pré-entraîné ou la partie souhaitée du modèle peut être intégré directement dans un nouveau modèle de réseau neuronal. Dans cette utilisation, les poids du pré-entraîné peuvent être gelés afin qu'ils ne soient pas mis à jour lorsque le nouveau modèle est entraîné. Alternativement, les poids peuvent être mis à jour pendant l'apprentissage du nouveau modèle, peut-être avec un taux d'apprentissage inférieur, permettant au modèle pré-entraîné d'agir comme un schéma d'initialisation de poids lors de l'apprentissage du nouveau modèle.

Nous pouvons résumer certains de ces modèles d'utilisation comme suit :

Classificateur : Le modèle pré-entraîné est utilisé directement pour classer les nouvelles images. **Extracteur de caractéristiques autonome :** le modèle pré-entraîné, ou une partie du modèle, est utilisé pour pré-traiter les images et extraire les caractéristiques pertinentes. **Extracteur de caractéristiques intégré :** le modèle pré-entraîné, ou une partie du modèle, est intégré dans un nouveau modèle, mais les couches du modèle pré-entraîné sont gelées pendant l'entraînement. **Initialisation du poids :** le modèle pré-entraîné, ou une partie du modèle, est intégré dans un nouveau modèle, et les couches du modèle pré-entraîné sont entraînées de concert avec le nouveau modèle. Chaque approche peut être efficace et faire gagner un temps considérable dans le développement et la formation d'un modèle de réseau de neurones à convolution profonde.

Il peut ne pas être clair quelle utilisation du modèle pré-entraîné peut donner les meilleurs résultats sur votre nouvelle tâche de vision par ordinateur, par conséquent, certaines expérimentations peuvent être nécessaires.

[Albatayneh, Omar, Lars Forsslöf, and Khaled Ksaibati].

3.4.3 Modèles d'apprentissage par transfert

Il existe peut-être une douzaine ou plus de modèles de reconnaissance d'images les plus performants qui peuvent être téléchargés et utilisés comme base pour la reconnaissance d'images et les tâches de vision par ordinateur associées.

Les trois modèles les plus populaires sont peut-être les suivants : VGG (e.g. VGG16 or VGG19). GoogLeNet (e.g. InceptionV3). Residual Network (e.g. ResNet50). Ces modèles sont à la fois largement utilisés pour l'apprentissage par transfert à la fois en raison de leurs performances, mais aussi parce qu'ils étaient des exemples qui ont introduit des innovations architecturales spécifiques, à savoir des structures cohérentes et répétitives (VGG), des modules de démarrage (GoogLeNet) et des modules résiduels (ResNet).

Keras donne accès à un certain nombre de modèles pré-entraînés les plus performants qui ont été développés pour les tâches de reconnaissance d'images.

Ils sont disponibles via l'API Applications et incluent des fonctions pour charger un modèle avec ou sans les poids pré-entraînés, et préparer les données d'une manière qu'un modèle donné peut attendre (par exemple, mise à l'échelle de la taille et des valeurs de pixels). [PRUKSACHATKUN, Yada, PHANG, Jason, LIU, Haokun, et al]. La première fois qu'un modèle pré-entraîné est chargé, Keras télécharge les poids de modèle requis, ce qui peut prendre un certain temps compte tenu de la vitesse de votre connexion Internet. Les poids sont stockés dans le répertoire `.keras/models/` sous votre répertoire personnel et seront chargés à partir de cet emplacement la prochaine fois qu'ils seront utilisés.

[Albatayneh, Omar, Lars Forsslöf, and Khaled Ksaibati].

3.4.4 Conclusion

Tout comme à l'intérieur du cerveau humain, les signaux sont transmis entre les neurones du cerveau artificiel. Le secret de cet exploit réside en grande partie dans les algorithmes. Dans le cas de la reconnaissance visuelle, pour être efficace, un algorithme de deep learning doit être capable d'identifier toutes les formes présentes et sous tous les angles. Ainsi, il pourra détecter la présence d'une voiture sur la route au milieu du paysage. Ceci n'est possible que si la machine a suivi une formation approfondie. Cela comprend la visualisation de milliers de photos de la voiture, sous toutes ses formes et sous tous les angles possibles. Lorsque la nouvelle image apparaît, elle est envoyée au réseau de neurones chargé de l'analyser et de déterminer si l'objet au milieu de la prise de vue est bien une voiture. Et tout cela se fait au moyen d'algorithmes complexes qui forment un modèle pour l'entraînement machine et ces algorithmes relèvent du domaine de l'apprentissage en profondeur tels que les réseaux de neurones convolutifs, dont nous avons discuté dans cette partie de la note.

[PRUKSACHATKUN, Yada, PHANG, Jason, LIU, Haokun, et al].

IMPLEMENTATION ET RÉSULTATS

4.1 Introduction

Les réseaux convolutifs sont au cœur de la plupart des solutions de vision par ordinateur de pointe pour une grande variété de Tâches. Depuis 2014, des réseaux convolutifs très profonds ont commencé de devenir grand public, ce qui a permis d'obtenir des gains substantiels dans divers repères. Bien que l'augmentation de la taille du modèle et des coûts de calcul aient tendance à se traduire par des gains de qualité pour la plupart des tâches (tant que suffisamment de données étiquetées sont fournies pour la formation), efficacité de calcul et faible paramètre count sont toujours des facteurs habilitants pour divers cas d'utilisation tels que vision mobile et scénarios big data. Ici, nous explorons les moyens d'étendre les réseaux de manière à utiliser le calcul ajouté aussi efficacement que possible en circonvolutions factorisées et régularisation agressive. Nous benchmarker nos méthodes sur la classification ILSVRC 2012 l'ensemble de validation de défi démontre des gains substantiels sur l'état de l'art : 21,2évaluation de trame unique à l'aide d'un réseau avec un coût de calcul de 5 milliards de multiplication-ajouts par inférence et avec utilisant moins de 25 millions de paramètres. Avec un ensemble de 4 modèles et évaluation multi-cultures, nous rapportons 3,5erreur et 17,3ment très performant au challenge de classification ILSVRC [16] 2014. Une observation intéressante était que les gains dans la classification, les performances ont tendance à se traduire par des gains de qualité significatifs dans une grande variété de domaines d'application. Cela signifie que les améliorations architecturales dans l'architecture convolutive profonde peuvent être utilisées pour améliorer les performances de la plupart des autres tâches de vision par ordinateur qui dépendent de plus en plus de fonctionnalités visuelles apprises de haute qualité. Aussi, les améliorations de la qualité du réseau ont abouti à de nouveaux domaines d'application pour les réseaux convolutifs dans les cas où Les fonctionnalités AlexNet ne pouvaient pas rivaliser avec celles conçues à la main, solutions sur mesure, par ex. génération

de propositions en détection[4]. Bien que VGGNet [18] ait la caractéristique convaincante de simplicité architecturale, cela a un coût élevé : évaluer le réseau nécessite beaucoup de calculs. Sur le d'autre part, l'architecture Inception de GoogLeNet [20] a également été conçu pour bien fonctionner même sous des contraintes strictes sur la mémoire et le budget de calcul. Par exemple, GoogleNet n'utilisait que 5 millions de paramètres, ce qui représentait une réduction de 12 fois par rapport à son prédécesseur AlexNet, qui utilisait 60 millions de paramètres. De plus, VGGNet a utilisé environ 3 fois plus de paramètres que AlexNet.

Depuis l'entrée gagnante du concours ImageNet 2012 [16] par Krizhevsky et al [9], leur réseau « AlexNet » a été appliqué avec succès à une plus grande variété d'ordinateurs tâches de vision, par exemple pour la détection d'objets [5], la segmentation [12], l'estimation de la pose humaine [22], la classification vidéo [8], le suivi d'objets [23] et la superrésolution [3]. Ces succès ont stimulé une nouvelle ligne de recherche axée sur la découverte de neurones convolutifs plus performants réseaux. À partir de 2014, la qualité des architectures réseau s'est considérablement améliorée en utilisant des réseaux. VGGNet [18] et GoogLeNet [20] ont donné simi-Le coût de calcul d'Inception est également beaucoup plus faible que VGGNet ou ses successeurs plus performants [6]. Cette a rendu possible l'utilisation des réseaux Inception dans le big data scénarios[17], [13], où une énorme quantité de données était nécessaire pour être traité à un coût raisonnable ou des scénarios où la mémoire ou la capacité de calcul est intrinsèquement limitée, par exemple dans les paramètres de vision mobile. Il est certainement possible d'atténuer parties de ces problèmes en appliquant des solutions spécialisées pour cibler l'utilisation de la mémoire [2], [15] ou en optimisant l'exécution de certaines opérations via des astuces de calcul [10]. cependant, ces méthodes ajoutent une complexité supplémentaire. De plus, ces Des méthodes pourraient également être appliquées pour optimiser l'architecture Inception, élargissant à nouveau l'écart d'efficacité. Pourtant, la complexité de l'architecture Inception rend il est plus difficile d'apporter des modifications au réseau. Si l'architecture est agrandie naïvement, une grande partie des gains de calcul peut être immédiatement perdue. De plus, [20] ne fournir une description claire des facteurs contributifs qui conduisent aux différentes décisions de conception du GoogLeNet architecture. Il est donc beaucoup plus difficile de l'adapter aux nouvelles cas d'utilisation tout en maintenant son efficacité. Par exemple, s'il est jugé nécessaire d'augmenter la capacité de certains Modèle de style Inception, la simple transformation de juste doubler le nombre de toutes les tailles de bancs de filtres conduira à un Augmentation de 4 fois du coût de calcul et du nombre de paramètres. Cela peut s'avérer prohibitif ou déraisonnable dans un beaucoup de scénarios pratiques, surtout si les gains associés sont modestes. Dans cet article, nous commençons par décrire quelques principes généraux et idées d'optimisation qui se sont avérés être utile pour étendre les réseaux de convolution de manière efficace façons. Bien que nos principes ne se limitent pas aux réseaux de type Inception, ils sont plus faciles à observer dans ce contexte car la structure générique des blocs de construction de style Inception est suffisamment flexible pour intégrer ces contraintes naturellement. Ceci est rendu possible par l'utilisation généreuse

de la réduction dimensionnelle et des structures parallèles des modules Inception qui permet d'atténuer l'impact des changements structurels sur composants voisins. Encore faut-il être prudent sur ce faisant, car certains principes directeurs doivent être observés pour maintenir la haute qualité des modèles. ité fournit simplement une estimation approximative de l'information contenu. 2. Les représentations de dimension supérieure sont plus faciles à traiter localement au sein d'un réseau. L'augmentation des activations par tuile dans un réseau convolutif permet de des traits plus démêlés. Les réseaux résultants s'entraînera plus vite. 3. L'agrégation spatiale peut être effectuée sur des plongements de dimension inférieure sans beaucoup ou aucune perte de pouvoir de représentation. Par exemple, avant d'effectuer une convolution plus étalée (par exemple 3×3), on peut réduire la dimension de la représentation d'entrée avant l'agrégation spatiale sans s'attendre à des effets négatifs graves. Nous émettons l'hypothèse que la raison de cette est la forte corrélation entre les résultats des unités adjacentes dans beaucoup moins de pertes d'informations lors de la réduction de dimension, si les sorties sont utilisées dans un contexte d'agrégation spatiale. Étant donné que ces signaux doivent être facilement compressible, la réduction de dimension favorise même apprentissage plus rapide. 4. Équilibrez la largeur et la profondeur du réseau. Optimal les performances du réseau peuvent être atteintes en équilibrant le nombre de filtres par étage et la profondeur de le réseau. Augmenter la largeur et la profondeur du réseau peut contribuer à des réseaux de meilleure qualité. Cependant, l'amélioration optimale pour une quantité constante de calcul peut être atteinte si les deux sont augmenté en parallèle. Le budget de calcul doit donc être réparti de manière équilibrée entre les profondeur et largeur du réseau. Bien que ces principes puissent avoir du sens, il n'est pas simple à utiliser pour améliorer la qualité des réseaux prêts à l'emploi. L'idée est de les utiliser judicieusement dans situations ambiguës seulement.

4.2 Principes généraux de conception

Nous décrivons ici quelques principes de conception basés sur l'expérimentation à grande échelle de diverses architectures choix avec des réseaux convolutifs. À ce stade, l'utilité des principes ci-dessous est spéculative et supplémentaire des preuves expérimentales futures seront nécessaires pour évaluer leur exactitude et leur domaine de validité. Pourtant, de graves écarts par rapport à ces principes ont eu tendance à entraîner une détérioration dans la qualité des réseaux et régler les situations où ces écarts ont été détectés ont entraîné des architectures améliorées en général. 1. Éviter les goulots d'étranglement de représentation, surtout au début le réseau. Les réseaux feed-forward peuvent être représentés par un graphique acyclique de la (des) couche(s) d'entrée à le classificateur ou le régresseur. Cela définit une direction claire pour le flux d'informations. Pour toute coupure séparant les entrées des sorties, on peut accéder à la quantité de l'information passant par la coupe. Il faut éviter goulots d'étranglement avec une compression extrême. En général le la taille de la représentation devrait diminuer doucement des entrées aux

sorties avant d'atteindre la représentation finale utilisée pour la tâche à accomplir. Théoriquement, le contenu de l'information ne peut pas être évalué simplement par la dimensionnalité de la représentation car il écarte des facteurs importants comme la structure de corrélation ; le dimensionnel.

4.3 Factorisation des convolutions avec un grand filtre Taille

Une grande partie des gains originaux du réseau GoogLeNet [20] découlent d'une utilisation très généreuse de la réduction de dimension. Cela peut être considéré comme un cas particulier de factorisation convolutions d'une manière efficace du point de vue informatique. Considérons par exemple le cas d'une couche convolutive 1×1 suivi d'une couche convolutive 3×3 . Dans un réseau de vision, on s'attend à ce que les sorties des activations à proximité sont fortement corrélées. On peut donc s'attendre à ce que leur activations peuvent être réduites avant l'agrégation et que cette devrait aboutir à des représentations locales expressives similaires. Ici, nous explorons d'autres façons de factoriser les convolutions dans divers contextes, notamment afin d'augmenter l'efficacité de calcul de la solution. Les réseaux Inception étant entièrement convolutifs, chaque poids correspond à.

[PRUKSACHATKUN, Yada, PHANG, Jason, LIU, Haokun, et al].

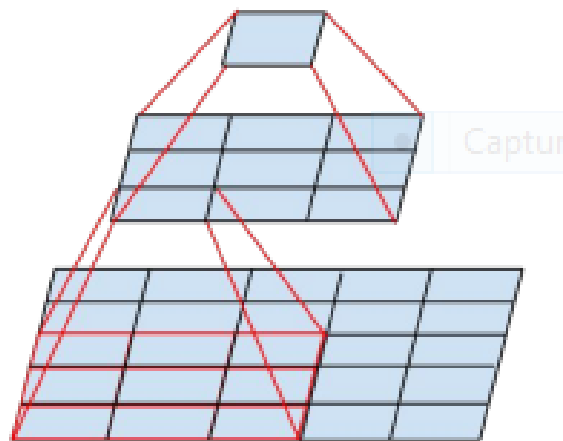


FIGURE 4.1: Figure 1. Mini-réseau remplaçant les 5×5 convolutions.

une multiplication par activation. Par conséquent, toute réduction dans le coût de calcul se traduit par un nombre réduit de paramètres. Cela signifie qu'avec une factorisation appropriée, nous pouvons nous retrouver avec des paramètres plus démêlés et donc avec une formation plus rapide. On peut aussi utiliser le calcul et des économies de mémoire pour augmenter la taille des bancs de filtres de nos réseau tout en maintenant notre capacité à former chaque modèle réplique sur un seul ordinateur.

4.3.1 Factorisation en plus petites convolutions

Convolutions avec des filtres spatiaux plus grands (par exemple 5×5 ou 7×7) ont tendance à être disproportionnellement coûteux en termes de calcul. Par exemple, une convolution 5×5 avec n filtres sur une grille avec m filtres est $25/9 = 2,78$ fois plus calculatoirement coûteux qu'une convolution 3×3 avec le même nombre de filtres. Bien sûr, un filtre 5×5 peut capturer les dépendances entre les signaux entre les activations de unités plus loin dans les couches antérieures, donc une réduction de la la taille géométrique des filtres a un coût élevé en expressivité. Cependant, on peut se demander si une convolution 5×5 pourrait être remplacé par un réseau multicouche avec moins de paramètres avec la même taille d'entrée et la même profondeur de sortie. Si nous zoomer sur le graphe de calcul de la convolution 5×5 , nous voyons que chaque sortie ressemble à un petit entièrement connecté réseau glissant sur 5×5 tuiles sur son entrée (voir Figure 1). Puisque nous construisons un réseau de vision, il semble naturel exploiter à nouveau l'invariance de traduction et remplacer composant connecté par une architecture convolutive à deux couches : la première couche est une convolution 3×3 , la seconde est une convolution couche entièrement connectée au-dessus de la grille de sortie 3×3 du première couche (voir Figure 1). Faire glisser ce petit réseau sur la grille d'activation des entrées se résume au remplacement du 5×5 convolution avec deux couches de convolution 3×3 (comparer Figure 4 avec 5). Cette configuration réduit clairement le nombre de paramètres en partageant les poids entre les tuiles adjacentes. Pour analyser

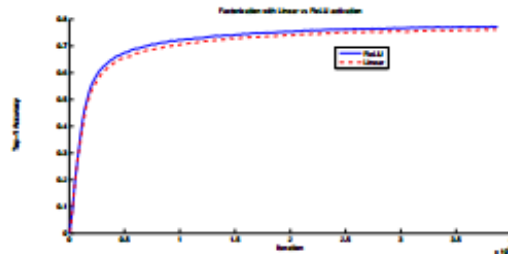


FIGURE 4.2: Figure 1. Mini-réseau remplaçant les 5×5 convolutions.

Figure 2. Une des nombreuses expériences de contrôle entre deux modèles Inception, l'un d'eux utilise la factorisation en linéaire + ReLU couches, l'autre utilise deux couches ReLU. Après 3,86 millions d'opérations, le premier s'établit à 76,2top-1 Précision sur l'ensemble de validation.

économies de coûts de calcul attendues, nous ferons quelques hypothèses simplificatrices qui s'appliquent aux situations typiques : On peut supposer que $n = m$, c'est-à-dire qu'on veut changer le nombre d'activations/unité par un alpha constant facteur. Puisque la convolution 5×5 est agrégée, α est généralement légèrement supérieur à un (environ 1,5 dans le cas de GoogLeNet). Avoir un remplacement à deux couches pour le 5×5 couche, il semble raisonnable d'atteindre cette expansion en deux étapes : augmenter le nombre de filtres de dans les deux pas.

Afin de simplifier notre estimation en choisissant $\alpha = 1$ (pas d'expansion), si on glissait naïvement un réseau sans en réutilisant le calcul entre tuiles de grille voisines, nous augmenterait le coût de calcul. glisser ce réseau peut être représenté par deux couches convolutives 3×3 qui réutilise les activations entre les tuiles adjacentes. De cette façon, nous se retrouver avec une réduction nette $9 \times 9 \times 25 \times$ du calcul, résultant en un gain relatif de 28 même économie pour le nombre de paramètres car chaque paramètre est utilisé exactement une fois dans le calcul de l'activation de chaque unité. Néanmoins, cette configuration soulève deux questions générales : Ce remplacement entraîne-t-il une perte d'expressivité? Si notre objectif principal est de factoriser la partie linéaire du calcul, cela ne suggérerait-il pas de garder les activations linéaires dans le première couche? Nous avons effectué plusieurs expériences de contrôle (par exemple, voir la figure 2) et l'utilisation de l'activation linéaire a toujours été inférieur à l'utilisation d'unités linéaires rectifiées à toutes les étapes de la factorisation. Nous attribuons ce gain à l'espace accru de variations que le réseau peut apprendre surtout si on normalise par lots [7] les activations de sortie. On peut voir pareil effets lors de l'utilisation d'activations linéaires pour les composants de réduction de dimension.

[PRUKSACHATKUN, Yada, PHANG, Jason, LIU, Haokun, et al].

4.3.2 Factorisation spatiale en convolutions asymétriques

Les résultats ci-dessus suggèrent que les convolutions avec filtres plus grands 3×3 a pourraient ne pas être généralement utiles car ils peuvent toujours être réduit en une suite de 3×3 convolutives couches.

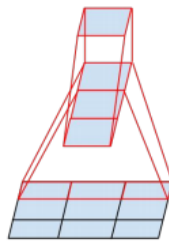


FIGURE 4.3: Figure 3. Mini-réseau remplaçant les 3×3 convolutions. le couche inférieure de ce réseau se compose d'une convolution 3×1 avec 3 unités de sortie.

On peut quand même se poser la question de savoir s'il faut factorisez-les en plus petites, par exemple 2×2 convolutions. Cependant, il s'avère que l'on peut faire encore mieux que 2×2 en utilisant des convolutions asymétriques, par ex. $n \times 1$. Par exemple en utilisant une convolution 3×1 suivie d'une convolution 1×3 équivaut à faire glisser un réseau à deux couches avec le même champ récepteur comme dans une convolution 3×3 (voir figure 3). Encore la solution à deux couches est 33 de filtres de sortie, si le nombre de filtres d'entrée et de sortie est égal. Par comparaison, factoriser une convolution 3×3 en une convolution deux 2×2 ne représente qu'une

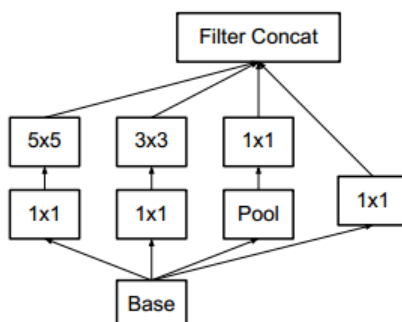
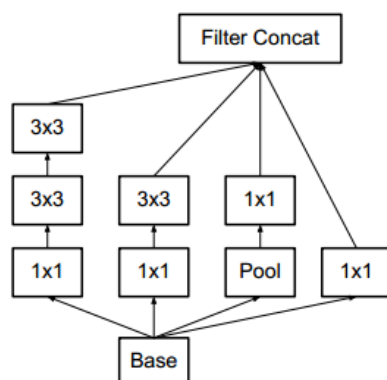


FIGURE 4.4: Figure 4. Module de création d'origine tel que décrit dans [20].

économie de 11 calcul. En théorie, nous pourrions aller encore plus loin et affirmer qu'un peut remplacer n'importe quelle convolution $n \times n$ par une convolution $1 \times n$ suivie d'une convolution $n \times 1$ et le calcul les économies de coûts augmentent considérablement à mesure que n augmente (voir figure 6).

FIGURE 4.5: Figure 5. Modules de démarrage où chaque convolution 5×5 est remplacée par deux convolutions 3×3 , comme suggéré par le principe 3 de Section 2.

En pratique, nous avons constaté que l'utilisation de cette factorisation ne fonctionne pas bien sur les premières couches, mais donne de très bons résultats sur des grilles de taille moyenne (sur les cartes de caractéristiques $m \times m$, où m est compris entre 12 et 20). A ce niveau, de très bons résultats peuvent être obtenus en utilisant 1×7 convolutions suivies par 7×1 circonvolutions.

4.4 Utilité des classificateurs auxiliaires

Aintroduit la notion de classificateurs auxiliaires pour améliorer la convergence des réseaux très profonds. La motivation initiale était de pousser les gradients utiles vers les couches inférieures pour les rendre immédiatement utiles et améliorer la convergence pendant l'entraînement en combattant le problème des gradients de disparition dans les réseaux très profonds. De plus,

Lee et al[11] soutiennent que les classificateurs auxiliaires favorisent un apprentissage plus stable et une meilleure convergence. Fait intéressant, nous avons trouvé que l'auxiliaire Les classificateurs n'ont pas entraîné d'amélioration de la convergence au début de la formation : la progression de la formation du réseau avec et sans tête latérale semble pratiquement identique avant que les deux modèles n'atteignent une précision élevée. Vers la fin de la formation, le réseau avec les branches auxiliaires commence à dépasser la précision du réseau sans branche auxiliaire et atteint un plateau légèrement plus élevé. Également[20] a utilisé deux têtes secondaires à différents stades du réseau. La suppression de la branche auxiliaire inférieure n'a pas eu d'effet négatif sur la qualité finale du réseau. Avec l'observation précédente dans le paragraphe précédent.

[PRUKSACHATKUN, Yada, PHANG, Jason, LIU, Haokun, et al].

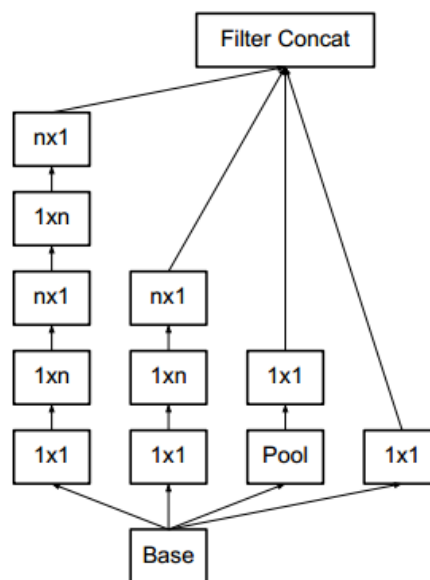


FIGURE 4.6: Figure 6. Modules de démarrage après la factorisation des $n \times n$ circonvolutions. Dans notre architecture proposée, nous avons choisi $n = 7$ pour la grille 17×17 . (Les tailles de filtres sont sélectionnées selon le principe 3).

cela signifie qu'à l'origine l'hypothèse de [20] que ces branches aident à faire évoluer les fonctionnalités de bas niveau est le plus probablement égaré. Au lieu de cela, nous soutenons que les classificateurs auxiliaires agissent comme régularisateur. Ceci est soutenu par le fait que le classificateur principal du réseau fonctionne mieux si le côté branche est normalisée par lot [7] ou a une couche de décrochage. Cette donne également une faible preuve à l'appui de la conjecture cette normalisation par lots agit comme un régularisateur.

4.5 Réduction efficace de la taille de la grille

Traditionnellement, les réseaux convolutifs utilisaient certains pools opération pour réduire la taille de la grille des cartes de caractéristiques. Dans afin d'éviter un goulot d'étranglement représentationnel, avant d'appliquer la mutualisation maximale ou moyenne de la dimension activation des filtres réseau est étendu. Par exemple, démarrer un grille $d \times d$ avec k filtres, si l'on veut arriver à $a \times d \times d \times 2$ grille avec $2k$ filtres, nous devons d'abord calculer une convolution stride-1 avec $2k$ filtres puis appliquer un pooling supplémentaire marcher. Cela signifie que le coût de calcul global est dominé par la convolution coûteuse sur la plus grande grille utilisant Opérations $2d^2k^2$. Une possibilité serait de passer à mise en commun avec convolution et donc résultant en $2(d^2)2k^2$.

[PRUKSACHATKUN, Yada, PHANG, Jason, LIU, Haokun, et al].

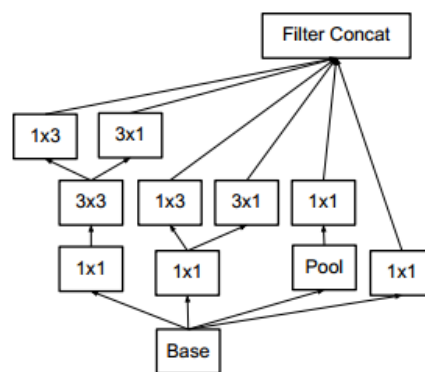


FIGURE 4.7: Figure 7. Modules de lancement avec des sorties de banc de filtres étendues. Cette architecture est utilisée sur les grilles les plus grossières (8×8) pour favoriser représentations de grande dimension, comme suggéré par le principe 2 de Section 2. Nous utilisons cette solution uniquement sur la grille la plus grossière, puisque c'est l'endroit où la production de haute dimension clairsemée la représentation est la plus critique car le rapport de traitement local (de 1×1 convolutions) est augmentée par rapport à l'agrégation spatiale.

réduire le coût de calcul d'un quart. Cependant, ce crée des goulots d'étranglement représentationnels car la dimensionnalité globale de la représentation chute à $(d^2)2k$ résultant en des réseaux moins expressifs (voir Figure 9). Au lieu de le faire, nous suggérons une autre variante qui réduit le calcul coûter encore plus cher tout en supprimant le goulot d'étranglement de la représentation. (voir la figure 10). On peut utiliser deux foulées parallèles 2 blocs : P et C. P est une couche de mise en commun (moyenne ou mise en commun maximale) l'activation, les deux sont foulée 2 dont les bancs de filtres sont concaténés comme sur la figure 10

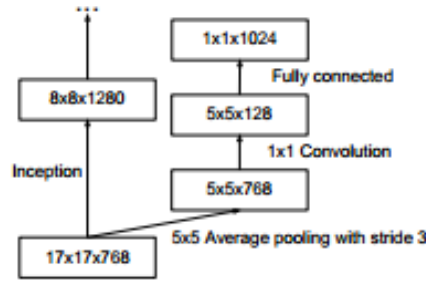


FIGURE 4.8: Figure 8. Classificateur auxiliaire au-dessus de la dernière couche 17×17 . Grouper la normalisation[7] des couches dans la tête latérale se traduit par un 0,4gain absolu dans la précision top-1. L'axe inférieur indique le nombre d'itérations effectuées, chacune avec une taille de lot de 32.

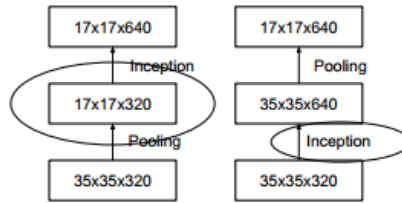


FIGURE 4.9: Figure 9. Deux manières alternatives de réduire la taille de la grille. La solution de gauche viole le principe 1 de ne pas introduire de goulot d'étranglement représentationnel de la section 2. La version de droite est 3 fois plus cher en calcul.

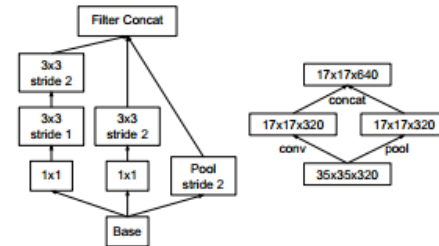


FIGURE 4.10: Figure 10. Module de lancement qui réduit la taille de la grille tout en élargissant les bancs de filtres. Il est à la fois bon marché et évite le goulot d'étranglement de représentation comme le suggère le principe 1. Le diagramme sur la droite représente la même solution mais du point de vue de tailles de grille plutôt que les opérations

4.6 Inception-v2

Ici, nous connectons les points d'en haut et proposons une nouvelle architecture avec des performances améliorées sur le Référentiel de classification ILSVRC 2012. L'aménagement de notre réseau est donné dans le tableau 1. Notons que nous avons factorisé la convolution traditionnelle 7×7 en trois convolutions 3×3 basées sur les mêmes idées que celles décrites

dans la section 3.1. Pour la partie Inception du réseau, nous avons 3 traditionnels modules de création au 35×35 avec 288 filtres chacun. Cette est réduit à une grille de 17×17 avec 768 filtres utilisant la grille technique de réduction décrite dans la section 5. Ceci est suivi par 5 instances des modules de création factorisés comme représenté sur la figure 5. Ceci est réduit à une grille de $8 \times 8 \times 1280$ avec la technique de réduction de grille illustrée à la figure 10. À le niveau 8×8 le plus grossier, nous avons deux modules Inception comme illustré à la figure 6, avec un banc de filtres de sortie concaténés taille de 2048 pour chaque tuile. La structure détaillée du réseau, y compris les tailles des bancs de filtres à l'intérieur de l'Inception modules, est donné dans le matériel supplémentaire, donné dans le model.txt qui se trouve dans le fichier tar de cette soumission.

[Albatayneh, O., Forslöf, L., Ksaibati, K. (2020)].

| type | patch size/stride or remarks | input size |
|-------------|---------------------------------|----------------------------|
| conv | $3 \times 3 / 2$ | $299 \times 299 \times 3$ |
| conv | $3 \times 3 / 1$ | $149 \times 149 \times 32$ |
| conv padded | $3 \times 3 / 1$ | $147 \times 147 \times 32$ |
| pool | $3 \times 3 / 2$ | $147 \times 147 \times 64$ |
| conv | $3 \times 3 / 1$ | $73 \times 73 \times 64$ |
| conv | $3 \times 3 / 2$ | $71 \times 71 \times 80$ |
| conv | $3 \times 3 / 1$ | $35 \times 35 \times 192$ |
| 3×Inception | As in figure 5 | $35 \times 35 \times 288$ |
| 5×Inception | As in figure 6 | $17 \times 17 \times 768$ |
| 2×Inception | As in figure 7 | $8 \times 8 \times 1280$ |
| pool | 8×8 | $8 \times 8 \times 2048$ |
| linear | logits | $1 \times 1 \times 2048$ |
| softmax | classifier | $1 \times 1 \times 1000$ |

FIGURE 4.11: Tableau 1. Aperçu de l'architecture de réseau proposée. le la taille de sortie de chaque module est la taille d'entrée du suivant. Nous utilisons des variantes de la technique de réduction illustrée à la figure 10 pour réduire les tailles de grille entre les blocs de lancement, le cas échéant. Nous avons marqué la convolution avec 0-padding, qui est utilisé pour maintenir la taille de la grille. 0-rembourrage est également utilisé à l'intérieur des modules Inception qui ne réduisent pas la taille de la grille. Tous les autres les calques n'utilisent pas de rembourrage. Les différentes tailles de bancs de filtres sont choisies d'observer le principe 4 de la section 2.

However, we have observed that the quality of the network is relatively stable to variations as long as the principles from Section 2 are observed. Although our network is 42 layers deep, our computation cost is only about 2.5 higher than that of GoogLeNet and it is still much more efficient than VGGNet.

4.7 Régularisation de modèle via le lissage d'étiquettes

Nous proposons ici un mécanisme pour régulariser le classifieur couche en estimant l'effet marginalisé de l'abandon de l'étiquette pendant la formation. Pour chaque exemple d'entraînement x , notre modèle calcule la probabilité de chaque étiquette $k \in 1 \dots K$: $p(k|x) = \exp(z_k) / \sum_{i=1}^K \exp(z_i)$

$\exp(z_i)$. Ici, z_i sont les logits ou probabilités de log non normalisées. Considérez la distribution de la vérité terrain sur étiquettes $q(k|x)$ pour cet exemple d'apprentissage, normalisé de sorte que $\sum_k q(k|x) = 1$. Par souci de concision, omettons la dépendance de p et q sur l'exemple x . Nous définissons la perte pour l'exemple comme l'entropie croisée : $J = - \sum_{k=1}^K \log(p(k)q(k))$. Minimiser cela équivaut à maximiser la valeur attendue log-vraisemblance d'une étiquette, où l'étiquette est sélectionnée en fonction de sa distribution de vérité fondamentale $q(k)$. Perte d'entropie croisée est différentiable par rapport aux logits z_k et peut donc être utilisé pour la formation de gradient de modèles profonds. Le dégradé a une forme assez simple : $\frac{\partial J}{\partial z_k} = p(k) - q(k)$, qui est bornée entre -1 et 1. Considérons le cas d'une seule étiquette de vérité fondamentale y , donc que $q(y) = 1$ et $q(k) = 0$ pour tout $k \neq y$. Dans ce cas, minimiser l'entropie croisée équivaut à maximiser la log-vraisemblance de l'étiquette correcte. Pour un exemple particulier x avec l'étiquette y , la log-vraisemblance est maximisée pour $q(k) = \delta_{k,y}$, où $\delta_{k,y}$ est le delta de Dirac, qui vaut 1 pour $k = y$ et 0 sinon. Ce maximum n'est pas réalisable pour z_k fini mais est approché si $z_y \gg z_k$ pour tout $k \neq y$ – c'est-à-dire si le logit correspondant à l'étiquette de vérité terrain est beaucoup plus grand que tous les autres logit. Ceci, cependant, peut poser deux problèmes. Premièrement, cela peut entraîner un surajustement : si le modèle apprend à attribuer une probabilité totale à l'étiquette de vérité fondamentale pour chaque exemple d'entraînement, il n'est pas garanti généraliser. Deuxièmement, il encourage les différences entre le plus gros logit et tous les autres à devenir gros, et ce, combiné au gradient borné $\frac{\partial J}{\partial z_k}$, réduit la capacité d'adaptation du modèle. Intuitivement, cela se produit parce que le modèle devient trop confiant quant à ses prédictions. Nous proposons un mécanisme pour encourager le modèle à être moins confiant. Bien que cela ne soit pas souhaitable si l'objectif est de maximiser la log-vraisemblance des étiquettes d'entraînement, régulariser le modèle et le rendre plus adaptable. La méthode est très simple. Considérons une distribution sur des étiquettes $u(k)$, indépendant de l'exemple d'apprentissage x , et un paramètre de lissage λ . Pour un exemple d'entraînement avec vérité terrain label y , on remplace la distribution de label $q(k|x) = \delta_{k,y}$ par $q_0(k|x) = (1 - \lambda)\delta_{k,y} + \lambda u(k)$ qui est un mélange de la distribution originale de la vérité terrain $q(k|x)$ et la distribution fixe $u(k)$, de poids $1 - \lambda$ et λ , respectivement. Cela peut être vu comme la distribution de l'étiquette k obtenue comme suit : d'abord, définissez-la sur l'étiquette de vérité fondamentale $k = y$; puis, avec probabilité λ , remplacer k par un échantillon tiré de la distribution $u(k)$. Nous proposons de utiliser la distribution a priori sur les étiquettes comme $u(k)$. Dans nos expériences, nous avons utilisé la distribution uniforme $u(k) = 1/K$, donc cette $q_0(k) = (1 - \lambda)\delta_{k,y} + \lambda/K$. Nous appelons ce changement dans la distribution des étiquettes de vérité terrain régularisation de lissage des étiquettes, ou LSR. Notez que LSR atteint l'objectif souhaité d'empêcher le plus grand logit de devenir beaucoup plus grand que tous les autres. En effet, si cela se produisait, alors un seul $q(k)$ s'approcheraient de 1 alors que tous les autres s'approcheraient de 0. résulter en une grande entropie croisée avec $q_0(k)$ car, contrairement $q(k) = \delta_{k,y}$, tous les $q_0(k)$ ont une borne inférieure positive. Une autre interprétation de LSR peut être obtenue en considérant l'entropie croisée :

$H(q, p) = -\sum_{k=1}^K q(k) \log p(k) = (1-\alpha)H(q, p) + \alpha H(u, p)$ Ainsi, LSR équivaut à remplacer une seule entropie croisée perte $H(q, p)$ avec une paire de telles pertes $H(q, p)$ et $H(u, p)$. La deuxième perte pénalise la déviation de l'étiquette prédite distribution p du a priori u , avec le poids relatif $1-\alpha$. Notez que cet écart pourrait être capturé de manière équivalente par la divergence KL, puisque $H(u, p) = D_{KL}(u||p) + H(u)$ et $H(u)$ est fixe. Lorsque u est la distribution uniforme, $H(u, p)$ est une mesure de la dissemblance de la distribution prédite p est uniforme, qui pourrait également être mesurée (mais pas de manière équivalente) par entropie négative $-H(p)$; Nous n'avons pas expérimenté cette approche. Dans nos expériences ImageNet avec $K = 1000$ classes, nous avons utilisé $u(k) = 1/1000$ et $\alpha = 0,1$. Pour ILSVRC 2012, nous avons trouvé une amélioration constante d'environ 0,2 absolu à la fois pour l'erreur top-1 et l'erreur top-5.

[Albatayneh, O., Forsl f, L., Ksaibati, K. (2020)].

4.8 M thodologie de formation

Notre objectif est d'utiliser la technique du transfert learning afin de d velopper un syst me de reconnaissance des empreintes digitales. Transfert learning est une des techniques les plus r centes dans le domaine du deep learning, elle consiste   donner des images  tiquet es d'empreinte digitale   un syst me de reconnaissance d'image, ce dernier peut s'adapter automatiquement   ces donn es. L'avantage de telle technique est au lieu de d velopper une nouvelle architecture de r seaux de neurone, on profite des architectures qui ont d j  prouv  leur efficacit  dans la classification des images, sachant que le d veloppement d'une nouvelle architecture peut  tre valid e qu'avec les exp rimentations, aucune technique scientifique peut dire combien de couches cach es on met, et combien de neurone dans chaque couche, et m me quels param tres meilleurs que les autres param tres. L'architecture utilis e s'appelle Inception version 3. Elle contient 8 couches en totale. Les 5 premi res couches pr sente un r seau de neurones convolutionnel, qui permet   transf rer l'image d'une matrice 5D   un vecteur de taille 2048  l ments. Puis les sorties de ce r seaux deviennent les entr es d'un r seaux de neurones fully connected avec 3 couches, ce dernier permet de classifier les vecteurs en deux classes (autoris , non_autoris ). Cette approche a comme avantage majeur d' viter l'extraction de caract ristiques des images. Notre r seau de neurone prend les images telles qu'elles sont et les classifie.

[Albatayneh, O., Forsl f, L., Ksaibati, K. (2020)].

4.9 R sultats Obtenus

Dans ce qui suit, nous discutons des r sultats obtenus, dans lesquels nous nous sommes bas s sur la meilleure pr cision d'entra nement, la meilleure pr cision de validation, la meilleure entropie crois e et la pr cision des tests pour  valuer l'effet du nombre d'it rations sur la reconnaissance de l'empreinte digitale. [Hussain, Mahbub, Jordan J. Bird, and Diego R. Faria]

TABLE 4.1: Résultats obtenus

| Number of iteration | Training Accuracy | Cross Entropy | Validation Accuracy | Test Accuracy |
|---------------------|-------------------|---------------|---------------------|---------------|
| 100 | 95.99 | 0.33 | 93.99 | 85.7 |
| 200 | 98 | 0.24 | 95.99 | 91.4 |
| 500 | 100 | 0.14 | 100 | 88.6 |
| 1000 | 100 | 0.10 | 100 | 91.4 |
| 1500 | 100 | 0.07 | 100 | 91.4 |
| 2000 | 100 | 0.06 | 99 | 91.4 |
| 2500 | 100 | 0.04 | 100 | 91.4 |
| 4000 | 100 | 0.02 | 100 | 91.5 |

Comme on le voit dans le tableau 4.1, le nombre d'itérations a affecté les résultats obtenus de telle sorte que si nous ajoutons plus d'itérations, le réseau de neurones construit de meilleurs modèles. Et cela est clair au niveau des trois premières mesures :

- *En termes de Training accuracy* : Cette mesure est la précision de l'application du modèle sur les données d'apprentissage, elle est utilisée pour évaluer le modèle lors des étapes de rétropropagation afin d'améliorer le modèle. Dans le tableau 4.1, nous avons cité la meilleure précision d'entraînement obtenue, comme on le voit, le modèle est parfait puisqu'il a correctement classé 95.99% à 100% des données d'entraînement. La figure 4.12 montre une mise à jour détaillée de la précision de l'entraînement au cours de chaque itération où (a), (b), (c), (d), (e), (f), (g) et (h) présentent respectivement 100, 200, 500, 1000, 1500, 2000, 2500 et 4000 itérations.

Pour tous les cas, nous voyons dans la figure 4.12 comment la précision de l'entraînement commence avec de petites valeurs (environ 5%), puis elle s'améliore itération après itération pour atteindre 100%.

- *En termes de Validation accuracy* : Cette mesure est la précision de l'application du modèle sur les données de validation, en tant que précision d'entraînement, elle est également utilisée pour évaluer le modèle lors des étapes de rétropropagation afin d'améliorer le modèle. Dans le tableau 4.1, nous avons cité la meilleure précision de validation obtenue, comme on le voit, le modèle est parfait puisqu'il a correctement classé 93% à 100% des données d'apprentissage. La figure 4.13 montre une mise à jour détaillée de la précision de validation au cours de chaque itération où (a), (b), (c), (d), (e), (f), (g) et (h) présentent respectivement 100, 200, 500, 1000, 1500, 2000, 2500 et 4000 itérations.

Pour tous les cas, nous voyons dans la figure 4.13 comment la précision de validation commence avec de petites valeurs (environ 5%), puis elle s'améliore itération après itération pour atteindre 100%. De plus, comme la précision de validation est inférieure à la précision d'entraînement, nous avons validé que notre modèle ne souffrait pas de problème de sous-ajustement.

- *En termes de Cross Entropy* : Lorsque nous utilisons la perte d'entropie croisée lors de

4.9. RÉSULTATS OBTENUS

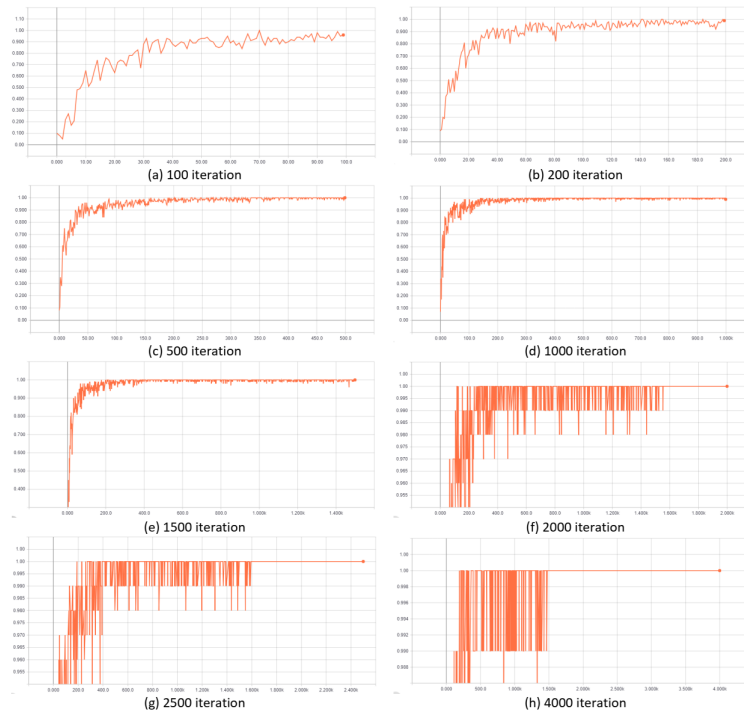


FIGURE 4.12: Training Accuracy obtenus selon le nombre d'itération



FIGURE 4.13: Validation Accuracy obtenus selon le nombre d'itération

l'entraînement des réseaux de neurones, nous calculons en fait la fonction de score à chaque fois lors du calcul des gradients pour les poids dans le réseau. Ainsi, l'objectif est de minimiser cette mesure, comme le montre le tableau 4.1, lorsque nous avons ajouté plus d'itérations, le réseau de neurones a minimisé l'entropie croisée, ce qui signifie que nous avons obtenu de meilleurs modèles. La figure 4.14 montre une mise à jour détaillée de l'entropie croisée au cours de chaque itération où (a), (b), (c), (d), (e), (f), (g) et (h) présentent respectivement 100, 200, 500, 1000, 1500, 2000, 2500 et 4000 itérations.

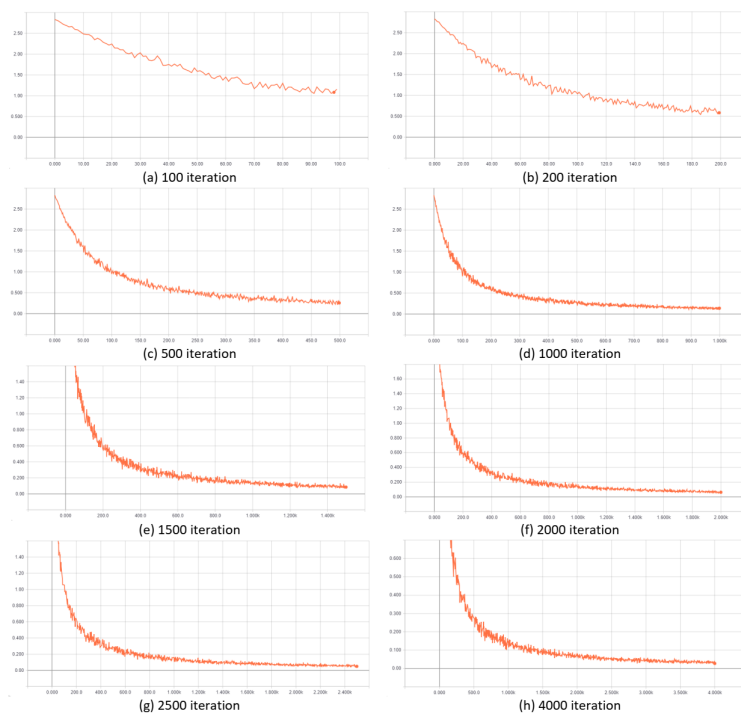


FIGURE 4.14: Cross Entropy obtenus selon le nombre d'itération

Pour tous les cas, nous voyons dans la figure 4.14 comment l'entropie commence avec des valeurs élevées (environ 0.33), puis elle a amélioré l'itération après l'itération pour converger vers 0. En détaillant les figures, nous voyons que plus nous avons ajouté des itérations, l'entropie croisée est devenue plus faible, ce qui signifie que nous avons réduit la perte d'informations en améliorant le modèle.

- *En termes de Test accuracy* : Cette mesure est la précision de l'application du modèle final sur les données de test, elle est utilisée pour évaluer la prédiction de nouvelles images. La figure 4.15 montre la comparaison de la précision du test en fonction du nombre d'itérations.

Dans notre cas (figure 4.15, le modèle a reconnu 85.7% des images de test lorsque nous avons construit un modèle en 100 itérations, alors qu'il s'est amélioré lorsque nous avons augmenté le nombre d'itérations à 200 itérations pour reconnaître 91,4% des empreintes

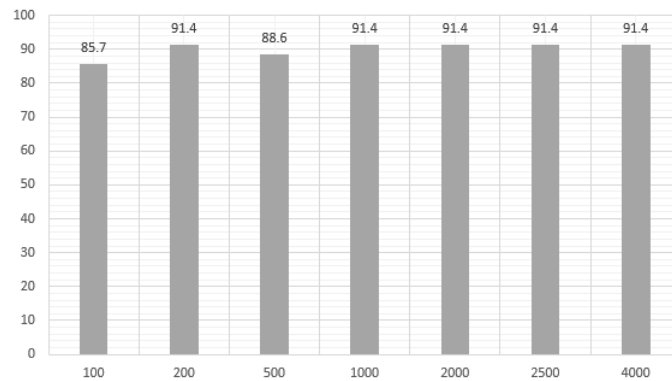


FIGURE 4.15: Test Accuracy obtenus selon le nombre d'itération

digitales, puis il est devenu fixe malgré le modèle a été amélioré sur la base des mesures précédentes.

4.10 Conclusion

Dans ce travail, nous avons appliqué un réseau neuronal convolutif profond pour l'identification des empreintes digitales à partir de leurs images. L'approche a été divisée en deux processus, le premier processus était l'extraction de caractéristiques à l'aide du calcul convolutif des non linéarités ReLU, où 5 couches en étaient responsables, puis, la dimension de ces caractéristiques obtenues a été réduite à l'aide de la méthode de mise en commun pour obtenir des vecteurs de taille 2048 élément i dont chaque vecteur représente une image dans l'ensemble de données. Après avoir obtenu les vecteurs, ils ont été utilisés comme entrées dans 3 couches entièrement connectées pour la classification. L'évaluation était basée sur quatre mesures : la précision de la formation, la précision de la validation, l'entropie croisée et la précision du test. Comme on le voit dans l'article, les valeurs des mesures obtenues ont prouvé que les réseaux de neurones convolutifs profonds peuvent être utilisés comme une bonne solution pour automatiser la classification des grains de pollen. De plus, nous avons vu que l'approche proposée évite le sous-apprentissage, ceci est prouvé par la précision de validation qui était inférieure à la précision d'entraînement.

CONCLUSION GÉNÉRALE

L'apprentissage en profondeur est d'une grande utilité dans le monde des technologies de l'information et de la communication, Il est utilisé dans les systèmes de reconnaissance faciale et vocale à bord de certains smartphones, et dans les robots pour que les équipements intelligents puissent avoir la réaction attendue dans une situation donnée (par exemple un réfrigérateur intelligent qui déclenche une alarme s'il détecte une porte laissée ouverte ou une température anormale à l'intérieur des compartiments). Vous vous demandez comment Facebook reconnaît vos amis dans les photos que vous publiez? Vous avez maintenant la réponse : l'apprentissage en profondeur. Les chercheurs, en particulier ceux qui étudient et/ou manipulent l'ADN, utilisent l'apprentissage en profondeur pour mener leurs recherches.

Ces technologies se retrouvent aussi dans les systèmes de traduction automatique, dans les voitures et autres véhicules autonomes, en médecine pour établir un diagnostic à partir d'un test d'imagerie (radio, IRM, scanner), en physique pour rechercher des particules et dans le domaine technique pour reproduire le travail.

Ce domaine a constitué un grand saut qualitatif, surtout après l'émergence du concept de transfert de connaissances et des modèles entraînés qui s'appuient sur les couches du réseau alexnet, dont nous avons déjà parlé dans cette note. point de départ de nombreuses recherches et domaines tels que le domaine du tri et de la classification des images, autour duquel s'articule cette note.

Dans ce travail, nous avons appliqué un réseau neuronal convolutif profond pour l'identification des empreintes digitales à partir de leurs images. L'approche a été divisée en deux processus, le premier processus était l'extraction de caractéristiques à l'aide du calcul convolutif des non linéarités ReLU, où 5 couches en étaient responsables, puis, la dimension de ces caractéristiques obtenues a été réduite à l'aide de la méthode de mise en commun pour obtenir des vecteurs de taille 2048 élément i dont chaque vecteur représente une image dans l'ensemble de données. Après avoir obtenu les vecteurs, ils ont été utilisés comme entrées dans 3 couches entièrement connectées pour la classification. L'évaluation était basée sur quatre mesures : la précision de la formation, la précision de la validation, l'entropie croisée et la précision du test. Comme on le voit dans l'article, les valeurs des mesures obtenues ont prouvé que les réseaux de neurones convolutifs profonds peuvent être utilisés comme une bonne solution pour automatiser la classification des grains de pollen. De plus, nous avons vu que l'approche proposée évite le sous-apprentissage, ceci est prouvé par la précision de validation qui était inférieure à la précision d'entraînement.

BIBLIOGRAPHIE

Kalantari, M, Kasser, M(2008)

Photogrammétrie et vision par ordinateur. Revue XYZ, 49-54.

KALANTARI, Mahzad et KASSER, Michel. Photogrammétrie et vision par ordinateur. Revue XYZ, 2008, p. 49-54.

Kalantari, Mahzad, and Michel Kasser. "Photogrammétrie et vision par ordinateur." Revue XYZ (2008) : 49-54.

Lakhal, N., Rouimel, N., Grimes, M. E. (2020). Reconnaissance d'empreintes digitales en utilisant les minuties (Doctoral dissertation, Université de Jijel).

LAKHAL, Nihad, ROUIMEL, Nada, et GRIMES, Mourad Encadreur. Reconnaissance d'empreintes digitales en utilisant les minuties. 2020. Thèse de doctorat. Université de Jijel.

Lakhal, Nihad, Nada Rouimel, and Mourad Encadreur Grimes. Reconnaissance d'empreintes digitales en utilisant les minuties. Diss. Université de Jijel, 2020.

Toufik, H. (2016). Reconnaissance Biométrique Multimodale basée sur la fusion en score de deux modalités biométriques : l'empreinte digitale et la signature manuscrite cursive en ligne (Doctoral dissertation, Ph. D. thesis, Université Badji Mokhtar-Annaba).

TOUFIK, Hafs. Reconnaissance Biométrique Multimodale basée sur la fusion en score de deux modalités biométriques : l'empreinte digitale et la signature manuscrite cursive en ligne. 2016. Thèse de doctorat. Ph. D. thesis, Université Badji Mokhtar-Annaba.

Toufik, Hafs. Reconnaissance Biométrique Multimodale basée sur la fusion en score de deux modalités biométriques : l'empreinte digitale et la signature manuscrite cursive en ligne. Diss. Ph. D. thesis, Université Badji Mokhtar-Annaba, 2016.

Learning, D. (2020). Deep learning. High-Dimensional Fuzzy Clustering. LEARNING, Deep. Deep learning. High-Dimensional Fuzzy Clustering, 2020. Learning, Deep. "Deep learning." High-Dimensional Fuzzy Clustering (2020).

Goodfellow, I., Bengio, Y., Courville, A. (2016). Deep learning. MIT press. GOODFELLOW, Ian, BENGIO, Yoshua, et COURVILLE, Aaron. Deep learning. MIT press, 2016.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. Deep learning. MIT press, 2016. LeCun, Y., Bengio, Y., Hinton, G. (2015). Deep learning. nature, 521(7553), 436-444.

LECUN, Yann, BENGIO, Yoshua, et HINTON, Geoffrey. Deep learning. nature, 2015, vol. 521, no 7553, p. 436-444. LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning."

nature 521.7553 (2015) : 436-444.

Lo, S. C. B., Chan, H. P., Lin, J. S., Li, H., Freedman, M. T., Mun, S. K. (1995). Artificial convolution neural network for medical image pattern recognition. *Neural networks*, 8(7-8), 1201-1214.

LO, Shih-Chung B., CHAN, Heang-Ping, LIN, Jyh-Shyan, et al. Artificial convolution neural network for medical image pattern recognition. *Neural networks*, 1995, vol. 8, no 7-8, p. 1201-1214.

Lo, Shih-Chung B., et al. "Artificial convolution neural network for medical image pattern recognition." *Neural networks* 8.7-8 (1995) : 1201-1214.

Lo, S. C., Lou, S. L., Lin, J. S., Freedman, M. T., Chien, M. V., Mun, S. K. (1995). Artificial convolution neural network techniques and applications for lung nodule detection. *IEEE transactions on medical imaging*, 14(4), 711-718.

LO, S.-CB, LOU, S.-LA, LIN, Jyh-Shyan, et al. Artificial convolution neural network techniques and applications for lung nodule detection. *IEEE transactions on medical imaging*, 1995, vol. 14, no 4, p. 711-718.

Lo, S-CB, et al. "Artificial convolution neural network techniques and applications for lung nodule detection." *IEEE transactions on medical imaging* 14.4 (1995) : 711-718.

Guo, X., Chen, L., Shen, C. (2016). Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis. *Measurement*, 93, 490-502.

GUO, Xiaojie, CHEN, Liang, et SHEN, Changqing. Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis. *Measurement*, 2016, vol. 93, p. 490-502.

Guo, Xiaojie, Liang Chen, and Changqing Shen. "Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis." *Measurement* 93 (2016) : 490-502.

Torrey, L., Shavlik, J. (2010). Transfer learning. In *Handbook of research on machine learning applications and trends : algorithms, methods, and techniques* (pp. 242-264). IGI global.

TORREY, Lisa et SHAVLIK, Jude. Transfer learning. In : *Handbook of research on machine learning applications and trends : algorithms, methods, and techniques*. IGI global, 2010. p. 242-264.

Torrey, Lisa, and Jude Shavlik. "Transfer learning." *Handbook of research on machine learning applications and trends : algorithms, methods, and techniques*. IGI global, 2010. 242-264.

Yang, Q., Zhang, Y., Dai, W., Pan, S. J. (2020). *Transfer learning*. Cambridge University Press.

YANG, Qiang, ZHANG, Yu, DAI, Wenyuan, et al. *Transfer learning*. Cambridge University Press, 2020.

Yang, Qiang, et al. *Transfer learning*. Cambridge University Press, 2020. Marcelino, P. (2018). *Transfer learning from pre-trained models. Towards Data Science*. MARCELINO, Pedro. *Transfer learning from pre-trained models. Towards Data Science*, 2018.

Marcelino, Pedro. "Transfer learning from pre-trained models." *Towards Data Science* (2018).
Pruksachatkun, Y., Phang, J., Liu, H., Htut, P. M., Zhang, X., Pang, R. Y., ... Bowman, S. R. (2020).

Intermediate-task transfer learning with pretrained models for natural language understanding : When and why does it work?. arXiv preprint arXiv :2005.00628.

PRUKSACHATKUN, Yada, PHANG, Jason, LIU, Haokun, et al. Intermediate-task transfer learning with pretrained models for natural language understanding : When and why does it work?. arXiv preprint arXiv :2005.00628, 2020.

Pruksachatkun, Yada, et al. "Intermediate-task transfer learning with pretrained models for natural language understanding : When and why does it work?." arXiv preprint arXiv :2005.00628 (2020).

Albatayneh, O., Forslöf, L., Ksaibati, K. (2020). Image retraining using TensorFlow implementation of the pretrained inception-v3 model for evaluating gravel road dust. *Journal of Infrastructure Systems*, 26(2), 04020014.

ALBATAYNEH, Omar, FORSLÖF, Lars, et KSAIBATI, Khaled. Image retraining using TensorFlow implementation of the pretrained inception-v3 model for evaluating gravel road dust. *Journal of Infrastructure Systems*, 2020, vol. 26, no 2, p. 04020014.

Albatayneh, Omar, Lars Forslöf, and Khaled Ksaibati. "Image retraining using TensorFlow implementation of the pretrained inception-v3 model for evaluating gravel road dust." *Journal of Infrastructure Systems* 26.2 (2020) : 04020014. Hussain, M., Bird, J. J., Faria, D. R. (2018, September). A study on cnn transfer learning for image classification. In *UK Workshop on computational Intelligence* (pp. 191-202). Springer, Cham. HUSSAIN, Mahbub, BIRD, Jordan J., et FARIA, Diego R. A study on cnn transfer learning for image classification. In : *UK Workshop on computational Intelligence*. Springer, Cham, 2018. p. 191-202.

Hussain, Mahbub, Jordan J. Bird, and Diego R. Faria. "A study on cnn transfer learning for image classification." *UK Workshop on computational Intelligence*. Springer, Cham, 2018. Lin, C., Li, L., Luo, W., Wang, K. C., Guo, J. (2019). Transfer learning based traffic sign recognition using inception-v3 model. *Periodica Polytechnica Transportation Engineering*, 47(3), 242-250.

LIN, Chunmian, LI, Lin, LUO, Wenting, et al. Transfer learning based traffic sign recognition using inception-v3 model. *Periodica Polytechnica Transportation Engineering*, 2019, vol. 47, no 3, p. 242-250.

Lin, Chunmian, et al. "Transfer learning based traffic sign recognition using inception-v3 model." *Periodica Polytechnica Transportation Engineering* 47.3 (2019) : 242-250.