Détection de concepts et annotation automatique d'images médicales par apprentissage profond

Mamitiana Ignace Randrianarivony






Université d'ANTANANARIVO
Domaine Sciences et Technologies
Mention Mathématiques et Informatique

Mémoire en vue de l'obtention du diplôme de Master 2 en
Mathématiques Informatique et Statistique Appliquées

Détection de concepts et annotation automatique d'images médicales par apprentissage profond

Présenté le 09 mars 2018 par :
RANDRIANARIVONY Mamitiana Ignace

Devant le jury composé de :

Président du jury :	*M. Joelson* SOLOFONIAINA	Université d'Antananarivo
Examinateur :	*M. Fenohery Tiana* ANDRIAMANAMPISOA	Université d'Antananarivo
Encadrante :	*M^me Josiane* MOTHE	Université de Toulouse
Co-encadrant :	*M. Olivier* ROBINSON	Université d'Antananarivo

Remerciements

Je souhaite adresser mes remerciements les plus sincères aux personnes qui m'ont aidé de près ou de loin pendant mon stage.

Je tiens à remercier sincèrement Madame Josiane MOTHE, qui, en tant qu'encadrante de stage, m'a aidé et orienté pendant le déroulement de mon stage

Mes remerciements s'adressent à Monsieur Olivier ROBINSON, pour son encadrement, son suivi, sa disponibilité et ses conseils.

J'adresse mes remerciements aux membres de jury qui ont accepté de juger mon travail en particulier Monsieur Joelson SOLOFONIAINA d'être le président du jury et Monsieur Fenohery Tiana ANDRIAMANAMPISOA d'être l'examinateur.

Je tiens à remercier aussi GRID 5000 et l'Université de Toulouse pour m'avoir permis d'accéder à des matériels performant pour l'élaboration de mon travail.

Je tiens à remercier Monsieur Tahiry ANDRIAMAROZAKANIAINA, pour son écoute et pour avoir accepté à répondre à toutes mes questions et aussi contribué à l'élaboration de mon travail, m'ayant guidé et corrigé durant le stage.

J'exprime ma gratitude à tous les enseignants de la MISA pour m'avoir aidé à améliorer mes connaissances. Je remercie particulièrement mes parents et ma famille pour m'avoir toujours supporté pendant tout mon stage et aussi ma famille.

Mes remerciements s'adressent aussi à ma promotion de Master, pour leur aide et leur partage durant mon stage.

Enfin, je remercie mes proches et mes amis, qui m'ont toujours aidé au cours de mon travail.

Merci

Table des matières

Table des figures

Liste des tableaux

Acronymes

Liste des acronymes utilisés dans le rapport :

CNN ou ConvNet : Convolutional Neural Network, Réseau de neurones convolutifs
CUI : Concept Unique Identifier
LSTM : Long Short-Term Memory
MSCOCO : Microsoft Common Objects in COntext
ReLU : rectified linear unit
resNet : deep residual network
RNA : Réseau de Neurones Artificiels
RNN : Recurrent Neural Network
SGD : Descente du gradient stochastique
TALN : Traitement Automatique du Langage Naturel
VO : Vision par Ordinateur
VGG : Visual Geometry Group

Chapitre 1

Introduction

Ce mémoire se situe dans le domaine de la recherche d'informations pour des images médicales. Il s'intitule "détection de concepts et annotation automatique d'images médicales par apprentissage profond". Il a pour but de concevoir des modèles pour automatiser l'extraction des informations pertinentes dans des images médicales. Ces informations sont les concepts médicaux et une description textuelle. Les concepts médicaux sont des termes biomédicaux présentés sous forme de codes, qui caractérisent le contenu visuel et abstraits dans l'image. Le but est d'identifier la présence des concepts pertinents dans l'image. D'un autre côté, l'annotation automatique d'images consiste à attribuer automatiquement, un texte ou une légende qui décrit l'image, à la manière dont nous, humains, décrivons une image à partir de sa visualisation.

Résoudre ces deux problèmes peut améliorer la productivité dans les institutions médicales. En général, le traitement des images médicales peut prendre beaucoup de temps car les systèmes d'acquisition et d'analyse des images nécessitent généralement l'intervention d'un spécialiste dans le domaine.

Ces deux problèmes (extraction de concepts et génération de légendes) ont été proposés dans le cadre du programme international ImageCLEF 2017 [1][40, 19] de la conférence CLEF (Conference and Labs of the Evaluation Forum [2], anciennement connue sous le nom de Cross-Language Evaluation Forum) qui s'est déroulé en septembre 2017. Dans le cadre de cette conférence, nous avons publié notre premier papier sur la détection automatique de concepts[62].

ImageCLEF a été lancée en 2003 dans le but de faire des recherches sur extraction d'information basée sur la combinaison du contenu visuel de l'image comme l'annotation automatique d'images ou la détection de concepts. ImageCLEF opère dans des recherches académiques et industrielles dans le domaine de recherche d'information, vision par ordinateur, informatique médicale, bio-informatique.

1. http ://www.imageclef.org/2017
2. http ://clef2017.clef-initiative.eu/

1.1 Contribution

Nous nous sommes inspirés des récentes avancées dans le domaine de la génération automatique de légendes pour des images naturelles[89, 33, 91] afin d'adopter un modèle pour des images médicales avec un apprentissage de bout en bout. Nous présentons aussi deux modèles réalisant un transfert d'apprentissage pour un problème de classification multi-label, dans le but pour détecter les concepts des images.

1.2 Organisation du document

Le chapitre 2 présente l'état de l'art. Il comporte 4 sections.

— La Section 2.1 indique les généralités de l'apprentissage automatique,

— la Section 2.2 introduit l'apprentissage profond avec ces différents variantes,

— la Section 2.3 présente les différentes méthodes de génération automatique de légendes,

— enfin la Section 2.4 cite les applications de l'apprentissage profond dans le domaine de l'imagerie médicale.

Le troisième chapitre correspond à nos contributions théoriques. Il comporte trois sections.

— la Section 3.1 décrit les prétraitements et les transformations que nous réalisons sur les images,

— la Section 3.2 présente nos analyses sur les données et deux modèles pour la détection de concepts,

— la Section 3.3 présente la génération automatique de légende avec les prétraitements des légendes.

Le chapitre quatre concerne l'implémentation des différentes méthodes décrites dans l'expérience suivie de la présentation et évaluation des résultats obtenus. Il comporte deux sections.

— la Section 4.1 présente les librairies et matériels utilisés

— la Section 4.2 décrit les résultats pour l'extraction de concepts

—

Le cinquième chapitre conclut le document sur une discussion globale des méthodologies et des résultats obtenus et aborde les travaux en perspective.

Chapitre 2

État de l'art

Ce chapitre présente l'apprentissage profond avec des réseaux de neurones et ses différentes applications. Dans la première partie nous aborderons les différentes techniques de l'apprentissage profond. Nous étudierons ensuite la génération automatique de légende. Enfin présenterons ensuite les applications de l'apprentissage profond dans le domaine médical.

2.1 Apprentissage automatique

L'apprentissage est un concept pour définir l'acquisition de connaissance et réutiliser ces nouvelles connaissances. Pour nous les humains, notre apprentissage se fait tout au long de notre vie. Nous apprenons à partir de perception de l'environnement avec les cinq sens, les expériences de la vie, de répétition des événements avec la mémoire, et de notre jugement (libre arbitre ou intelligence). Pour les machines, comme elles ne sont pas dotées de sens ni de jugement dynamique, elles obéissent aux instructions dans un programme avec des données d'entrée et une donnée sortie ou réponse du programme.

Arthur Samuel a décrit l'apprentissage automatique comme un domaine qui permet à une machine d'avoir des connaissances sans être formellement programmé ou sans intervention humaine [75]. [58] Tom Mitchell quant à lui a défini l'apprentissage comme un programme informatique qui apprend "à partir d'une donnée d'expérience E une tâche spécifique T avec une mesure de performance P". Par exemple dans une étude météo. $E =$ sera l'historique météo de plusieurs jours, $T =$ l'estimation de la météo du lendemain et $P =$ la probabilité de la météo estimée par le programme.

L'apprentissage supervisé peut être mis en parallèle avec un enseignant qui utilise ses connaissances pour enseigner et corriger les erreurs d'un élève. Cette analogie est utilisée par les algorithmes d'apprentissage pour apprendre à partir d'une donnée. Lorsque l'algorithme fait une prédiction sur un exemple, sa précision peut être calculée car la réponse correspondant à l'exemple est connue.

2.1.1 Classification

La classification supervisée est une variante de l'apprentissage supervisé. Elle consiste à faire correspondre à une entrée X, un ensemble discret de y qui représente sa catégorie. $\widehat{y} = f(X)$ avec $\widehat{y}$ est fini et représente la catégorie de X.

2.1.2 Fonction d'erreur et optimisation des hyper-paramètres

Les fonctions d'erreur sont des fonctions mathématiques qui servent à pénaliser le classifieur (modèle, algorithme) en cas de mauvaise prédiction. C'est donc aussi une fonction objective car, pour avoir un bon classifieur il faut que les pénalités reçues par le classifieur soient minimums.

Soit L la fonction d'erreur, on peut écrire :

$$\widehat{y} = model(X, W) \tag{2.1}$$

$$min_W L(\widehat{y}, y) \tag{2.2}$$

où $\widehat{y}$ est la prédiction de y pour X par le modèle,
W : hyper-paramètre qui va déterminer la prédiction du modèle. C'est toujours la valeur de cet hyper-paramètre que l'on cherche à trouver dans les algorithmes d'apprentissage.

Le choix de la fonction d'erreur est primordial en apprentissage automatique.

2.1.3 Descente stochastique de gradient

La descente stochastique de gradient (SGD) est une technique d'optimisation itérative [11] qui est la plus utilisée dans le domaine de l'apprentissage profond. On l'utilise pour minimiser la fonction d'erreur (équation 2.2) afin de chercher les hyper-paramètres.

Son algorithme est le suivant :

Algorithm 1 Descente stochastique de gradient

1. Initialiser l'hyper-parametre W
2. Choisir le pas d'apprentissage η

repeat

3. Prendre aléatoirement un sous ensemble des données x
4. Prédire $\widehat{y}$, $\widehat{y} = model(x, W)$
5. Calculer l'erreur $L(\widehat{y}, y)$
6. *Mettre a jour de l'hyper-paramètre.* $W := W + \eta \Delta W$

until Le minimum approximatif est obtenu

Le "momentum" est une variante de la descente stochastique de gradient pour augmenter la vitesse de l'optimisation [45, 84], il change sur la mise à jour du gradient : au lieu de faire directement $W := W + \eta \Delta W$, la mise à jour se fait en deux étapes comme présenté ci-dessous.

Algorithm 2 mise à jour de l'hyper-paramètre

1. $v = \mu * v - \eta \Delta W$
2. $W = W + v$

v : vélocité

μ : momentum est une constante généralement de valeur 0,9, d'après [45].

Dans l'algorithme de descente stochastique de gradient, on prend un petit sous-ensemble aléatoire des données à chaque itération (voir Algorithme 3 , étape 3) que l'on appelle **lot** (batch). Une *époque* c'est quand on a fait passer l'ensemble de toutes les données dans le modèle (étape 3).

Dans le cadre de ma recherche, j'ai travaillé avec des algorithmes d'apprentissage profond avec des réseaux de neurones. Dans la sous-section suivante nous aborderons des études spécifiques sur ces algorithmes.

2.2 Apprentissage profond

Le réseau de neurones artificiel est un algorithme d'apprentissage automatique qui s'inspire de quelques aspects des neurones animaux (voir Figure 2.1). Il s'agit d'un des plus puissants classifieurs aujourd'hui. Il est modélisé mathématiquement par un réseau dans un graphe, plus ou moins complexe, hiérarchique sous forme de couches dont les nœuds élémentaires sont appelés neurones.

Il est composé généralement de trois types de couches : la couche d'entrée (input layer), couche(s) cachée(s) (hidden layer) et la couche de sortie (output layer). La couche d'entrée est composée de neurones qui correspondent aux caractéristiques des données d'entrée représentées par une grille multidimensionnelle (par exemple la matrice de pixels de l'image ou la forme vectorielle de la donnée). La couche de sortie représente les résultats de la tache assignée au réseau. Prenons l'exemple d'une classification de 1 000 classes, les 1 000 neurones de la couche de sortie représentent la probabilité ou le score pour chaque classe. Les couches cachées sont les couches intermédiaires entre l'entrée et la sortie. L'ensemble du réseau ainsi formé est généralement vu comme une boîte noire.

La taille d'un réseau de neurones est mesurée par le nombre de couches, nombre de nœuds et la façon les nœuds/neurones sont connectés (voir la figure 2.2).

Theorem 2.1. *Théorème d'Hornik [36, 15, 29, 16] : Un réseau de neurones avec 1-couche cachée possède la propriété d'approximation universelle, c'est-à-dire qu'il peut approximer n'importe quelle fonction continue avec une précision arbitraire, à condition de disposer de suffisamment de neurones sur sa couche cachée.*

Comme on peut faire approximer n'importe quelle fonction continue avec un réseau de neurones à 1-couche cachée (cf théorème d'Hornik), l'idée de l'apprentissage profond vient du fait d'augmenter le nombre de couches cachées du réseau, d'où le terme *profond*, pour pouvoir apprendre un niveau d'abstraction plus élevé [5, 14].

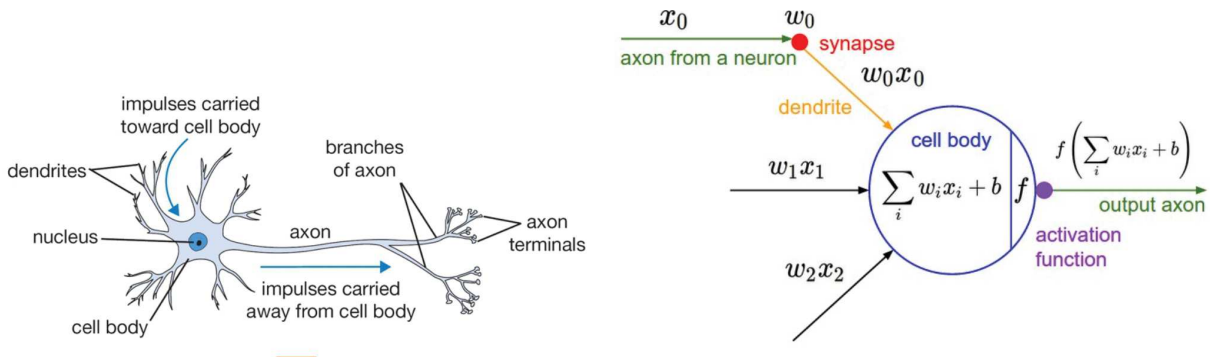


FIGURE 2.1 – Comparaison entre un neurone animal et le modèle mathématique d'un neurone artificiel. Dans les réseaux biologiques, les données d'entrée proviennent d'un autre neurone qui passe dans la dendrite, le corps cellulaire fait les calculs et le résultat sort par l'axone. Le fonctionnement d'un neurone artificiel est calqué sur celui d'un neurone biologique. Image prise de [45].

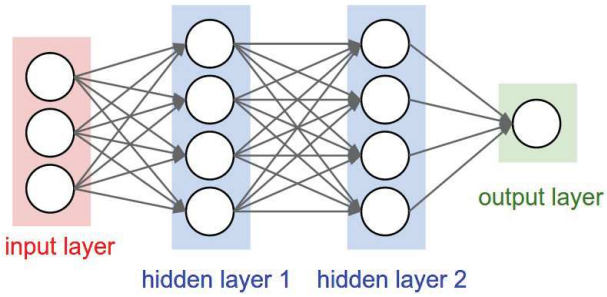


FIGURE 2.2 – 3-réseau de neurones ou réseaux de neurones avec deux couches cachées ; réseau de neurones pleinement connecté. Image prise de [45]

Le développement de l'apprentissage profond a été influencé par les échecs des autres algorithmes à apprendre un niveau de représentation élevé et à bien généraliser une forme d'intelligence artificielle [58, 38]. Par exemple les autres algorithmes d'apprentissage automatique ne parvenaient pas à avoir des résultats convenables dans le domaine de l'intelligence artificielle comme la reconnaissance d'objets, la segmentation d'images ou la reconnaissance des voix. On peut qualifier un réseau de neurones de profond si le nombre de couches cachées est supérieure ou égale à 2 (voir par exemple les figures 2.2 et 2.3). Dans la sous-section suivante nous citerons les points importants dans l'histoire de l'apprentissage profond et des réseaux de neurones, puis son fonctionnement et les autres types de réseaux de neurones que nous avons exploités pour cette recherche.

2.2.1 Historique des réseaux de neurones

Les grandes lignes de l'histoire des réseaux de neurones sont les suivantes :

— Le réseau de neurones est apparu dans les années 1940 sous le nom "multilayer perceptrons" (perceptron multicouche) par [5, 68] les travaux de Warren McCulloch et Walter Pitts qui ont montré que l'on pouvait approximer n'importe quelle fonction arithmétique ou logique, avec des perceptrons multicouche".

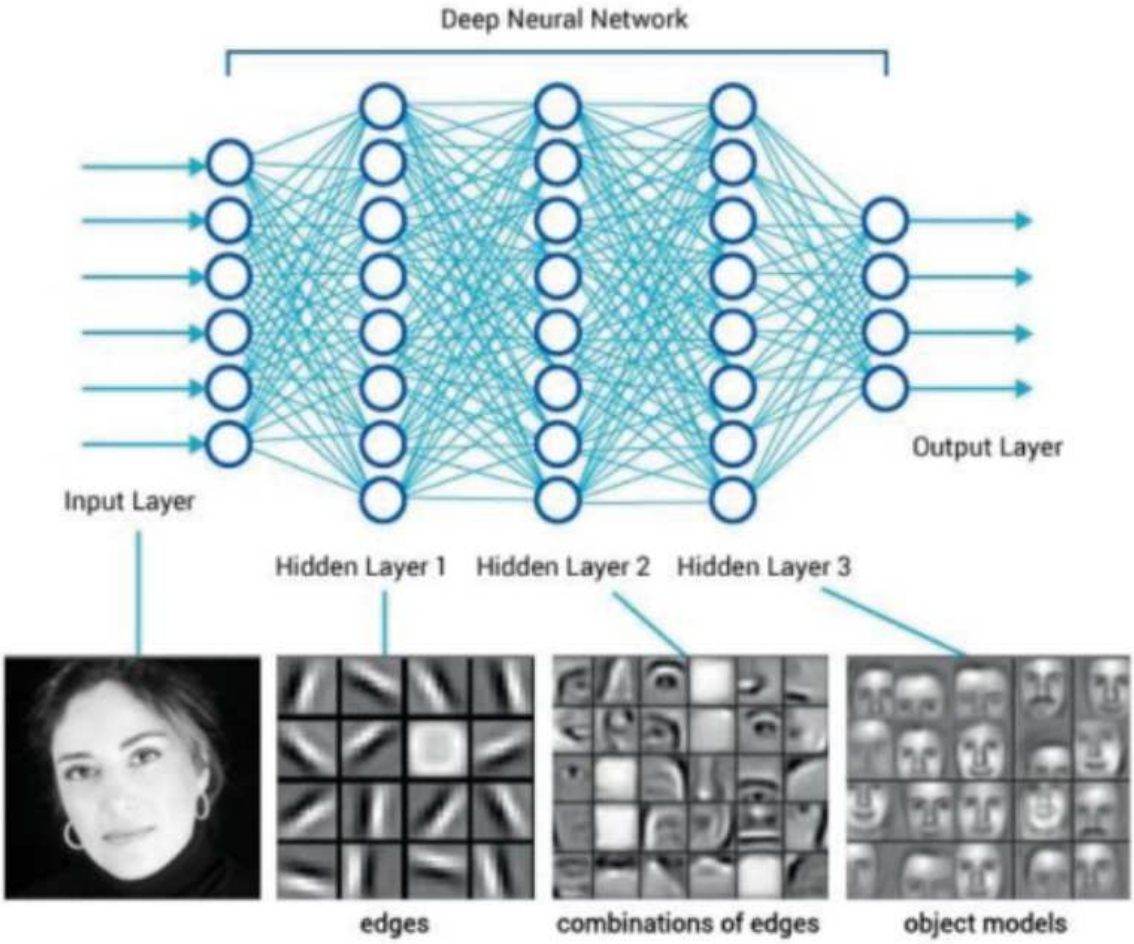


FIGURE 2.3 – Illustration du fonctionnement d'un réseau de neurones à 3 couches cachées pour le problème de la détection faciale. On voit que sur la première couche le réseau apprend les traits caractéristiques, sur la deuxième couche il combine les traits pour apprendre des formes plus concrètes comme les yeux ou le nez et sur la troisième couche il combine les formes précédentes pour avoir un visage. Image prise de [51].

— Puis, de 1958-1969 les recherches sur les perceptrons multicouches se sont multipliées avec l'apparition des premiers ordinateurs qui ont permis de faire plusieurs calculs.

— 1969 : les perceptrons ne peuvent faire que des classifications binaires et par contre ils rencontrent des difficultés avec la classification de la porte logique XOR.

— 1980 : achèvement de l'algorithme de rétro-propagation par David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams [45, 65] .

— Yann LeCun a popularisé les réseaux de neurones avec LeNet le premier réseau de neurones convolutif pour la reconnaissance de caractère en 1998 [50, 45] .

2.2.2 Fonctionnement

Un neurone biologique se caractérise par (cf figure 2.1) :

— synapse : le point de connexion avec les autres neurones,

— dendrites : les points d' entrées du neurone,

— axone : le point de sortie,

— le noyau : qui active les sorties en fonction des stimulations en entrée.

Par ressemblance à un neurone biologique, un neurone artificiel se définit par (cf figure 2.1) :

— un vecteur poids w de même taille que les signaux d'entrées,

— des signaux d'entrées $X = x_1, ..., x_n$,

— une fonction d'activation g,

— le neurone calcule le produit scalaire entre les signaux d'entrée et le vecteur poids en ajoutant un biais. Le résultat est appliqué dans la fonction d'activation.

$$s = g(X.w + b) = g(\sum_{i=1}^{n} x_i * w_i + b) \tag{2.3}$$

s : la valeur ou état interne, ou score interne du neurone.
b : variable biais du neurone.

La fonction d'activation g sert à introduire une opération de non-linéarité après l'opération de produit scalaire (cf équation 2.3). Cette non-linéarité permet d'avoir différentes variations de l'état interne sur un objet de la même classe (voir un aspect particulier de la classe) [45]. Les fonctions d'activation les plus utilisées sont :

— sigmoid : $\sigma(x) = \frac{1}{1+e^x}$

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

— Rectified-Linear unit (ReLU) : $relu(x) = max(0, x)$

Les modèles tanh, sigmoïde, ReLU, sont bien adaptés aux algorithmes d'optimisation comme la descente stochastique de gradient (cf. ci-dessus) car elles sont différentiables. Mais actuellement c'est le ReLU qui est le plus utilisé dans les réseaux de neurones profonds à cause d'un problème que l'on appelle **"vanishing gradient"**. Comme les autres fonctions d'activation (sigmoïde, tanh) produisent des valeurs entre 0 et 1, le gradient va se tendre vers 0 au fur et à mesure que l'on a des réseaux profonds (plusieurs couches). Le ReLU permet d'amortir le "vanishing gradient" car sa valeur de sortie est invariante si x est >0. De plus le calcul des exponentielles présent dans les fonctions sigmoïde et tanh est lent par rapport au calcul du max() dans ReLU qui est donc préféré.

Ce sont (a) la valeur du vecteur poids et (b) le biais pour chaque neurone qui seront les **hyper-paramètres** estimés pendant la phase d'apprentissage.

Pour prédire à partir d'un modèle de réseaux de neurones artificiels (cf. équation 2.2), on parcourt le réseau en partant de la couche d'entrée puis, on calcule successivement les valeurs des états internes de chaque neurone dans les couches cachées jusqu'à la couche de sortie. Par la suite on applique aux valeurs internes de la couche de sortie ce qu'on appelle une fonction d'entropie. Par définition, une fonction d'entropie est fonction mathématique qui sert à transformer les valeurs des états internes des neurones de la couche de sortie en probabilité. La prédiction est la classe correspondant au neurone qui possède la plus

grande probabilité. Généralement on utilise softmax entropie ou sigmoïde est utilisée avec comme fonction d'erreur respective softmax loss ou sigmoid entropy loss.

$$softmax = \frac{e^{s_y}}{\sum_{j=1}^{K} e^{s_j}} \tag{2.4}$$

$$softmax\_loss = -\log(softmax) \tag{2.5}$$

$$\sigma = \frac{1}{1+e^{s}} \tag{2.6}$$

$$sigmoidEntropyLoss = -\sum_{k=1}^{K}[y_k \log(\sigma_k) + (1-y_k)\log(1-\sigma_k)]. \tag{2.7}$$

K : nombre de classes
y : valeur réelles des classes, de taille k, dont la valeur est 1 si y_k est la classe de l'entrée et 0 sinon pour k variant de 1 à K
s : score obtenu par la variable d'entrée (image i) sur la couche de sortie , vecteur de même taille que y
s_y : valeur du score à l'indice de y où il y a un 1 .
σ : sigmoid, vecteur de taille K, car on fait une opération sur les éléments du vecteur. La différence entre softmax et sigmoid est que softmax est normalisé alors que sigmoid ne l'est pas.

Par exemple dans le cas du modèle présenté dans la figure 2.2, nous avons une entrée X de taille 3, deux couches cachées $h1$ et $h2$ de taille 4 et une couche de sortie de taille 1. Cet exemple correspond à une classification binaire donc la sortie est soit 0 soit 1. Si on veut prédire à partir de ce réseau, on suit l'équation 2.10.

$$h1 = relu(X * W1 + b1) \tag{2.8}$$

$$h2 = relu(h1 * W2 + b2) \tag{2.9}$$

$$predit = softmax(relu(h2 * W3 + b3)) \tag{2.10}$$

2.2.3 Différence par rapport au cerveau humain

Bien que les réseaux de neurones artificiels soient puissants dans le domaine de l'intelligence artificielle, voici quelques remarques sur les réseaux de neurones artificiels et leurs limites par rapport au cerveau humain :

- Un cerveau humain possède environ 100 milliards de neurones et 100 trillions de connexions (synapses), alors qu'il faut encore une super machine pour pouvoir supporter un tel nombre de neurones artificiels.
- Les données d'entrée du cerveau sont les 5 sens mais non un vecteur ou une matrice.
- Un enfant n'a pas besoin d'apprendre en regardant 100 000 images de sandwich pour savoir s'il s'agit d'un sandwich ou pas (une classification d'images), alors que les algorithmes d'apprentissage automatique oui.

2.2.4 Réseaux de neurones convolutif

Le réseau de neurones convolutif (*Convolutional Neural Network* ou *ConvNet* ou CNN) est très similaire à un réseau de neurones du type de ceux que nous avons évoqué précédemment. Les ConvNet sont très utilisés dans le domaine de la vision par ordinateur si on n'évoque qu' *imageNet, et pascal voc* qui sont des problèmes de détection et de classification d'objets dans une image. Les CNN ont aussi beaucoup de succès dans les reconnaissances faciales, la détection d'objets très utilisée dans les robots et les voitures automatiques. En gros, tout ce qui concerne la vision par ordinateur et les images. De plus on peut utiliser les convNet dans tous les problèmes ayant en entrée une matrice. Par exemple, Gehring [24] a utilisé une matrice de texte dans une tâche de traduction automatique de langue. Le réseau de neurones convolutif est spécialisé dans le traitement des données matricielles et du signal.

On note que le terme "convolutional" vient de l'opération de convolution de matrices utilisée dans le traitement de signal. Dans les convNet, 2 nouveaux types de couche ont été ajoutés dans le réseau : la couche convolution (convolutional layer) et la couche de mise en commun (pool layer). Nous les décrivons dans les parties suivantes.

2.2.4.1 Couche de convolution

Dans la couche de convolution, au lieu de faire un produit scalaire entre les valeurs internes et les poids de chaque neurone (équation 2.3), on applique un produit de convolution. Ce produit de convolution sert à extraire des caractères spécifiques dans le signal ou de l'image traitée. Dans un réseau de neurones convolutif on applique une succession de filtres de convolution dans le but d'extraire petit à petit les informations caractéristiques sur l'image.

L'opération de convolution est illustrée dans la figure 2.4. On la calcule par l'équation 2.11 où S est le produit de convolution résultant [18].

$$S = \sum_{k=0}^{M'} \sum_{l=0}^{N'} \sum_{i=1}^{R} \sum_{j=1}^{C} I_{(i+k),(j+l)} F_{(R+i-1),(C-j+1)} \tag{2.11}$$

avec
I : matrice de pixels représentant l'image
F : matrice du filtre de convolution à appliquer sur l'image
S : résultante du produit de convolution , appelée **feature map**
R : nombre de lignes de la matrice filtre
C : nombre de colonnes de la matrice filtre
M' : $\frac{M-R}{P} + 1$, avec M est le nombre de lignes de la matrice d'images, et P le nombre de pas du filtre suivant la ligne et colonne
N' : $\frac{N-R}{P} + 1$, avec N le nombre de colonnes de la matrice d'images.

Les filtres de convolution varient en fonction des effets désirés comme exemple : détecter de contour de l'image, rendre floue l'image, etc... Mais en terme d'apprentis-

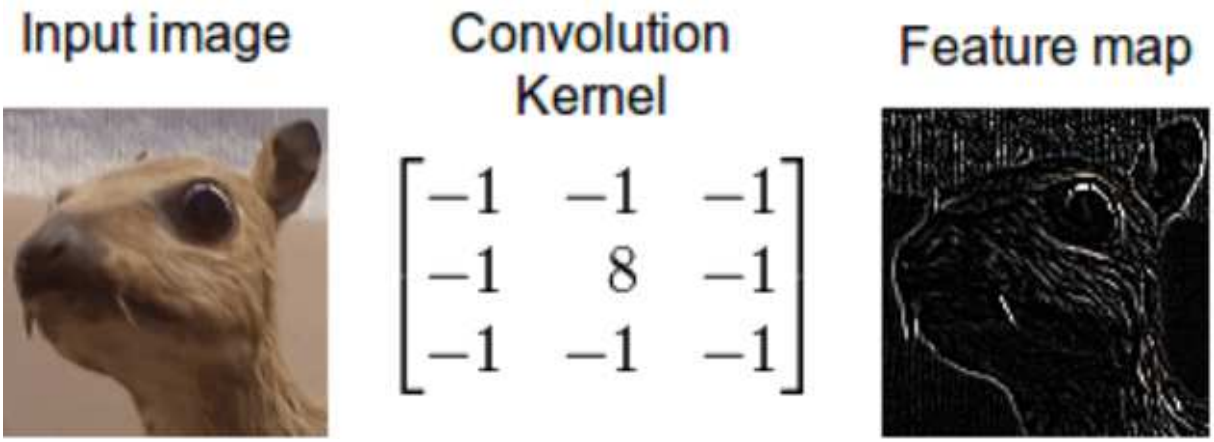


FIGURE 2.4 – Produit de convolution, image source : [18]

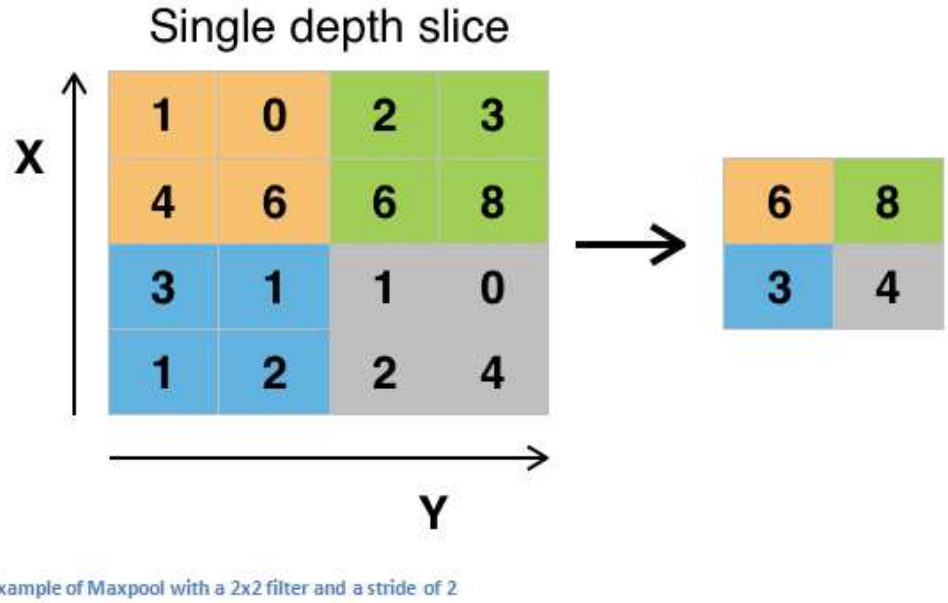


FIGURE 2.5 – Illustration de la couche de mise en commun, avec la méthode maxpooling

sage, les valeurs filtres de convolution sont les hyper-paramètres à apprendre durant la phase d'apprentissage.

2.2.4.2 Couche de mise en commun

La couche de mise en commun est une sorte de sous échantillonnage non linéaire. Elle sert à réduire la dimension spatiale dans un réseau ConvNet.

C'est une couche qui prend en entrée chaque "feature map" [1] et la simplifie. Par exemple le "maxpooling" prend une région et donne en sortie le maximum de cette région (figure 2.5).

Par exemple, si la couche cachée à partir des "feature map" est de 4x4, en prenant une région de 2x2 pour le "maxpooling", il ne restera plus en sortie que 2x2 unités.

1. feature map : résultat du produit de convolution

2.2.5 Transfert d'apprentissage

Le transfert d'apprentissage [45] est une stratégie qui cherche à optimiser les performances sur une machine d'apprentissage à partir des connaissances et d'autres tâches faites par une autre machine d'apprentissage [90].

En pratique d'après Yosinski *et al.* [92] , entraîner un modèle ConvNet depuis le début (avec les initialisations) n'est pas recommandé parce que l'entraînement d'un modèle de ConvNet nécessite une grande masse de données et prend beaucoup de temps. Par contre, il est plus usuel d'utiliser des modèles de ConvNet déjà entraîner et de les réadapter pour le problème, c'est ce que l'on appelle le transfert d'apprentissage. Il s'agit de transférer l'apprentissage d'un modèle traitant un problème vers un autre type de problème.

Il y a deux types de transfert d'apprentissage.

— L'extraction de variables du ConvNet : ici, le ConvNet est utilisé comme un extracteur[79, 45], c'est à dire qu'un vecteur est extrait à partir d'une certaine couche du modèle sans rien modifier à sa structure ou son poids et le vecteur précédemment extrait est utilisé pour une nouvelle tâche.

— Le réglage fin du modèle de ConvNet [78, 81, 38, 55] : ici, le nouveau ConvNet est initialisé avec les poids et la structure du modèle pré-entraîné à utiliser. La structure du modèle pré-entraîné est légèrement modifiée pour la nouvelle tâche, et enfin le nouveau modèle est entraîné pour la nouvelle tâche.

2.2.6 Réseau de neurones récurrents et "long short-term memory"

Les convNets et réseaux de neurones connectés sont conçus pour traiter des problèmes dont les variables en entrée ont une instance indépendante dans le temps. Par contre, il y a certains problèmes où l'ordre des événements compte. Comme dans le cas des traitements vidéo qui sont des séquences d'images dans le temps, ou des traitements de textes qui sont une suite de mots successifs et interdépendants. Pour cela, le réseau de neurones récurrent (RNN, Récurrent Neural Network), une autre extension du réseau de neurones connecté, a été introduit. Le RNN est un réseau de neurones avec mémoire où les informations du passé importent dans l'algorithme. Donc il est fait pour traiter le problème de donnée séquentielle.

Un simple RNN se construit juste en prenant la couche sortie de la précédente étape et le concaténé avec l'entrée de l'étape courante. Tout cela dans le but d'avoir la prédiction de l'étape courante (équation 2.12), d'où le nom de récurrent pour la récurrence.

$$y_t = f_W(x_t, y_{t-1}) \tag{2.12}$$

x_t : entrée à l'instant t
y_{t-1} : sortie prédite pour l'instant t-1
y_t : sortie prédite pour l'instant t

W : poids

f : la fonction d'activation dans le réseau de neurones connectés (tanh, sigmoïde, relu).

Dans le cas général on n'utilise pas cette forme simple de RNN car cette forme ne suffit pas pour avoir les meilleurs résultats. De plus, l'information apportée par la valeur de sortie à $t-1$ n'est pas très riche [43]. Ainsi, au lieu d'utiliser la valeur de sortie de l'instant t-1, on met en paramètres les valeurs de la couche cachée à t-1 qui sont plus signifiantes en termes neuronaux. [77, 13, 44]. La formulation de RNN est dans l'équation 2.14

$$h_t = f_W(x_t, h_{t-1}) \tag{2.13}$$

$$y_t = W.h_t \tag{2.14}$$

x_t : entrée à l'instant t

h_{t-1} : couche cachée pour l'instant t-1

h_t : couche cachée prédite pour l'instant t

y_t : sortie prédite pour l'instant t

f : la fonction d'activation dans le réseau de neurones connecté.

La Figure 2.6 présente l'illustration de l'équation 2.14 sous la forme d'un graphe.

D'après cette figure et l'équation 2.14, le problème avec le RNN est le fait de garder en mémoire toutes les informations de chaque instant si l'on veut qu'il soit assez flexible. Par exemple (pris de [47]) dans le texte "J'ai grandi en France ...(2000mots)... Je parle couramment le xxxx". On voudrait prédire le xxxx. Le plus logique est xxxx soit égal à "français". Mais pour qu'un RNN trouve que xxxx est égal à français, il doit mémoriser 2000 instants, ce qui est conséquent.

Tout cela va créer un très profond réseau de neurones au niveau de la récurrence (voir figure 2.6) et cela pénalise l'apprentissage en matière de mémoire (de l'ordinateur) et de temps d'apprentissage.

Comme solution, Hochreiter et Schmidhuber [34] ont inventé le LSTM (Long-Short Term Memory Cell) en 1997. Le LSTM est juste une autre forme de RNN. Le LSTM est conçu dans le but de supporter les problèmes aux longs termes de dépendances parce que sa plus grande particularité est de mémoriser beaucoup d'informations. Dans une couche LSTM, on a 4 neurones que l'on appelle *porte* alors que dans un RNN on a un seul neurone. Ces neurones de LSTM ont chacun leurs rôles et interagissent entre eux de manière spécifique. Le modèle mathématique de LSTM est dans l'équation 2.20

$$f_t = \sigma(W_f[x_t, h_{t-1}] + b_f) \tag{2.15}$$

$$i_t = \sigma(W_i[x_t, h_{t-1}] + b_i) \tag{2.16}$$

$$\tilde{C}_t = tanh(W_{\tilde{C}}[x_t, h_{t-1}] + b_{\tilde{C}}) \tag{2.17}$$

$$C_t = f_t * Ct - 1 + i_t * \tilde{C}_t \tag{2.18}$$

$$o_t = \sigma(W_f[x_o, h_{t-1}] + b_o) \tag{2.19}$$

$$h_t = o_t * tanh(C_t) \tag{2.20}$$

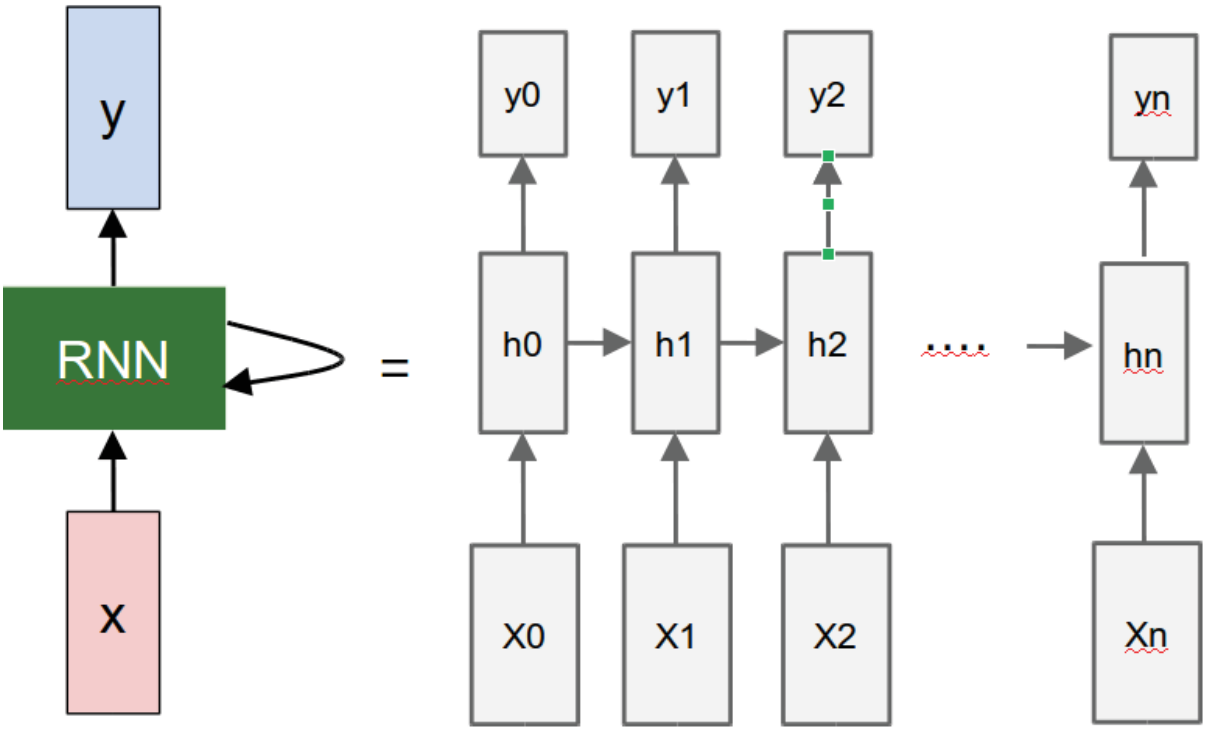


FIGURE 2.6 – Réseau de neurones récurrent - illustration, image prise de [44].

f_t : forget gate (porte d'oubli), c'est ce nœud qui décide quelle information sera supprimée/oubliée et quelle information sera gardée.
i_t : input gate (porte d'entrée), c'est ce nœud qui décide quelle valeur sera mise à jour,
$\tilde{C}_t$: new memory cell sert à stocker les informations. Combinée avec i_t, on y stocke les i informations qui viennent d'être mises à jour,
C_t : final memory cell supprime les anciennes informations oubliées de l'état précédent $(f_t * C_{t-1})$ et ajoute les nouvelles informations mises à jour $(i_t * \tilde{C}_t)$,
o_t : output gate : décide quelle information va à la sortie,
h_t : hidenstate : est la sortie, on filtre le final memory cell C_t avec l'output gate,
Les W_f, W_i, $W_{\tilde{C}}$, W_o et b_f, b_i, $b_{\tilde{C}}$, b_o sont les poids et biais respectifs des neurones $f, i, \tilde{C}, o$. Ils sont indépendants du temps mais ce sont toujours eux que l'on va apprendre dans la phase d'apprentissage .

2.3 Génération automatique de légende

L'annotation automatique d'images consiste en général à générer automatiquement la description du contenu visuel de l'image avec des mots, une phrase ou même avec une histoire. Alors on combine des techniques de vision par ordinateur pour le traitement du contenu de l'image et des techniques de traitement automatique de langage naturel (TALN) pour la génération des phrases. Bien que la génération de textes soit une tache courante, ce qui rend la génération automatique de légende difficile c'est la liaison entre le modèle graphique avec la vision par ordinateur et le modèle textuel. On peut aussi assimiler la génération automatique de légende en un problème de traduction au-

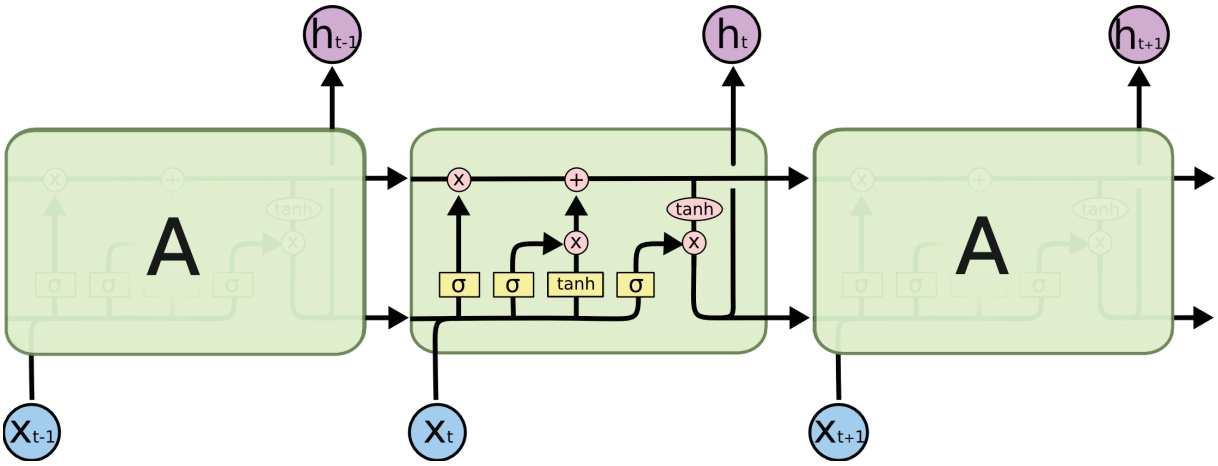


FIGURE 2.7 – Module répété d'un LSTM, avec les 4 neurones (portes) en jaune , image source : [47].

FIGURE 2.8 – Exemple de génération automatique de légende pour des images naturelles de l'ensemble MS-COCOajouter une note pour dire ce qu est MS Coco

tomatique. La traduction automatique vise à traduire un texte d'une langue à une autre (français vers malagasy par exemple) mais à la différence de ce type de traduction, dans la génération automatique de légende, il s'agit de traduire une image numérique en du texte. La génération automatique de légende est très utilisée pour indexer les images dans les moteurs de recherche comme Google image par exemple. De plus, c'est un pas pour donner aux ordinateurs un sens et une description des images. Enfin c'est utile pour les gens qui ne peuvent pas voire directement l'image, on peut leur donner la légende correspondante pour qu'ils aient une idée du contenu de l'image.

Une légende est une phrase avec une succession de variétés de classes de mots : adjectif, nom, préposition. Il y a deux grandes approches utilisées en annotation automatique d'image : l'approche par système d'extraction et l'approche par génération.

2.3.1 Approche par système d'extraction

Les études de ce groupe posent le problème de générer automatiquement la description d'une image en récupérant des images similaires à l'image à décrire [66, 82].

L'un des premiers modèles qui suivent cette approche a été celui proposé par Ordonez *et al.* [66] dans Im2Text. Les vecteurs de variables caractéristiques utilisés sont les

Algorithm 3 Génération de description par extraction

1. Transformer l'image en son vecteur de variables caractéristiques
2. Récupérer un ensemble d'images candidates à partir de l'ensemble d'apprentissage en fonction d'une mesure de similarité dans l'espace de caractéristiques utilisé.
3. Reclasser les descriptions des images candidates en utilisant davantage les informations visuelles et / ou textuelles contenues dans l'ensemble de récupération, ou bien combiner des fragments des descriptions des images candidates selon certaines règles ou schémas.

points d'intérêts de SIFT [66, 82], Haar [40], les vecteurs de caractéristique d'un réseau de neurones convolutif [81, 82, 8].

Pour l'étape de reclassement, les auteurs appliquent à l'image une variété de classifieur (classification d'objets, activités, scène ...) pour avoir le contenu visuel de l'image. Finalement, le reclassement est effectué par un classifieur formé sur ce contenu visuel et caractéristiques sémantiques des descriptions des images candidates.

Ce système nécessite une grande taille et bonne qualité des données d'entraînement pour qu'il soit flexible par rapport aux nouvelles images. Cependant il rencontre un problème de passage à l'échelle quand il y a beaucoup d'images à comparer pour trouver les plus similaires.

2.3.2 Approche par génération

L'approche par génération consiste à générer les légendes mot par mot. C'est une méthode plus flexible par rapport à l'approche par système d'extraction car elle ne répète pas les légendes déjà existantes dans les données d'entraînement. Il s'agit de l'approche la plus utilisée depuis quelques années.

Vinalys *et al.* [89] ont créé le premier modèle de génération en 2014. Ce modèle est inspiré des techniques en TALN pour les traductions automatiques de langue où on traduit une phrase d'une langue S vers une autre langue T avec un système d'encodeur-décodeur et en cherchant la phrase traduite T telle que la probabilité $P(T/S)$ soit maximale.

Le modèle de Vinalys *et al.* est divisé en deux modules : l'**encodeur d'image** et le *modèle de langue* (Figure 2.9). L'image brute entre dans l' **encodeur d'image** qui est généralement un ConvNet (le convNet nommé inceptionV3 [2]) pour faire un transfert d'apprentissage dans le but d'extraire un vecteur caractéristique. Ce vecteur se trouve à l'avant-dernière couche du CNN inception Puis l'image précédemment encodée passe dans le *modèle de langue* (les auteurs utilisent LSTM) où le modèle de langue prédit mot par mot la légende [3]. C'est dans la partie modèle de langue que l'on va faire **l'apprentissage**. D'une manière plus mathématique, voici l'expression de cette méthode.

2. `https://github.com/n3011/Inception_v3_GoogLeNet`

3. Le modèle de langue est un modèle probabiliste qui génère les légendes mot par mot connaissant l'image et les mots précédemment générés

$$x_{-1} = CNN(image) \tag{2.21}$$

$$\hat{y}_t = LSTM(x_{-1}, \hat{y}_{t-1}, ..., \hat{y}_0) \tag{2.22}$$

$$P(\hat{y}|I) = \sum_t P(\hat{y}_t|I, \hat{y}_{t-1}, ..., \hat{y}_0) \tag{2.23}$$

$$\theta = argmax_\theta P(\hat{y}|I) \tag{2.24}$$

x_{-1} : image encodée

$\hat{y}_t$: mot prédit pour l'instant t

C'est à partir de ce modèle de base que des améliorations ont été apportées.

You *et al.* [93] ont ajouté la classe des images comme information supplémentaire au modèle de langue. Ces classes sont associées chacune à un sous ensemble de mot dans le dictionnaire. Ainsi, le modèle de langue cherchera mieux dans un sous ensemble de mots plus précis. On parle ici d'**attention sémantique**. Ensuite Nomena [33] a utilisé des vecteurs de catégorie des images dans un réseau de neurones multimodaux.

Pour continuer, la notion d'**attention visuelle** est introduite par Li *et al.* [52], l'attention visuelle est un concept humain qui regarde une petite région de l'image contenant l'élément le plus attirant dans l'image. Li utilise deux ConvNet dans la partie encodeur d'image, un pour faire la détection d'objet et l'autre pour extraire la couche mise en commun (feature map, image extraite par le ConvNet) puis les résultats de ces ConvNet seront passés au modèle de langue. Pour Xu [91], à partir de la couche de mise en commun extraite il prend une partie (région) de la couche et pour chaque partie, il génère un mot qui correspond à une partie de l'image initiale.

Tous ces modèles précédemment cités sont appelés architectures en **pipeline** parce qu'elles traitent séparément le modèle de langue et l'encodeur d'image.

Karpathy *et al.* [46] casse l'indépendance des deux modules en combinant les deux modules avec une approche entraînement de bout-en-bout. L'entraînement de bout-en-bout signifie qu'une seule fonction d'erreur est utilisée pour optimiser le modèle en entier mais pas seulement le modèle de langue. Cette méthode d'entraînement offre plus de relation dans le modèle de langue et l'encodeur d'image parce que ils sont interdépendants. L'erreur est rétro-propagée sur le modèle de langue puis dans l'encodeur d'image pour faire le réglage fin. Le modèle de *Karpathy* a comme objectif de générer des légendes par région. Il y a à d'abord une détection d'objets puis une génération de légendes pour les objets détectés

Tous ces modèles sont entraînés et évalués pour les collections flickr8k, flick30k [35], et MS-COCO [54] Ce sont les collections de données les plus utilisées dans la recherche en génération automatique de légende pour les images naturelles. Les mesures d'évaluation les plus utilisées dans la génération automatique de légende sont BLEU-n et Meteor. Elles sont aussi très utilisées dans les problèmes de génération automatique de phrase en TALN. BLEU-n est utilisée dans l'évaluation de la traduction automatique où il calcule la qualité d'une phrase traduite connaissant quelques phrases de référence. BLEU est l'une des mesures qui a une forte corrélation avec le jugement humain. Le calcul de BLEU est

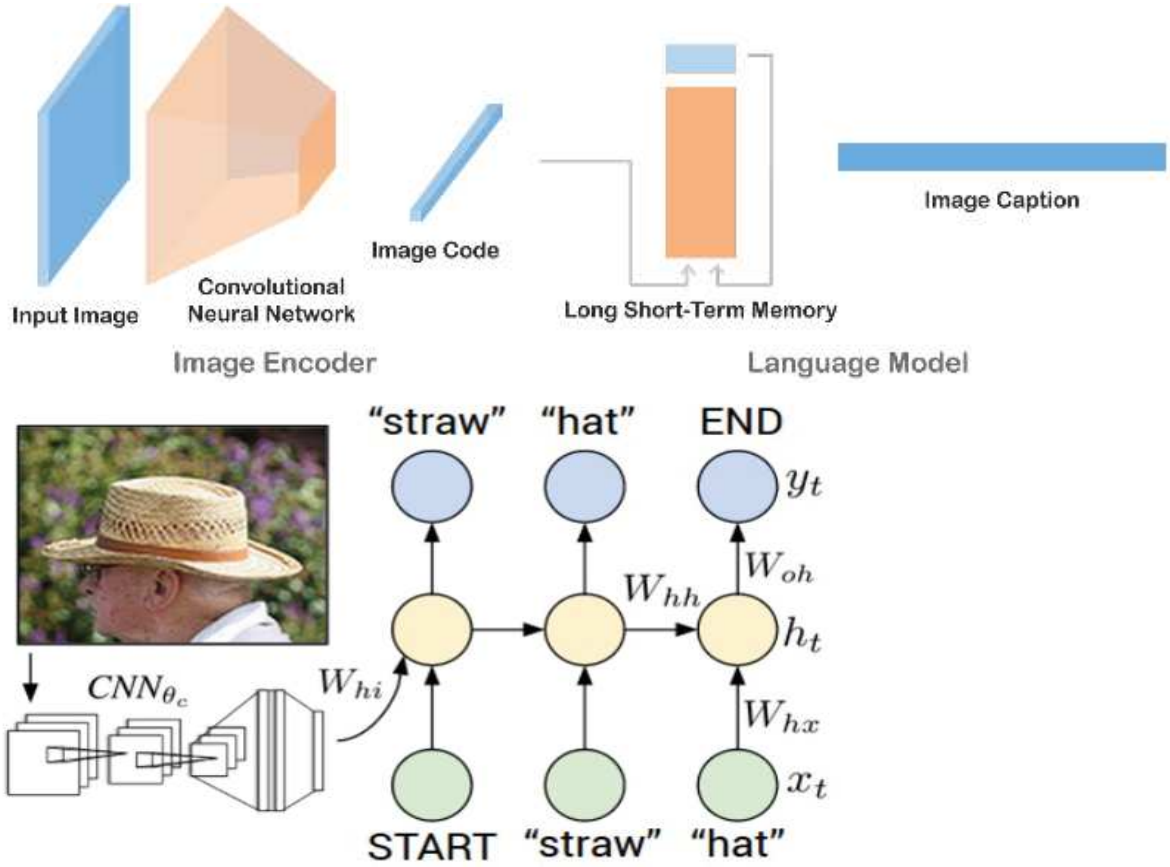


FIGURE 2.9 – Architecture du modèle de base de génération automatique de légende. source [64, 89].

basé sur la précision de n-gram. Tandis que Meteor est basée sur la moyenne pondérée de la précision et du rappel uni-gram sur les phrases de référence et celles prédites.

2.4 Apprentissage profond et imagerie médicale

De nombreuses techniques d'acquisition permettent d'obtenir des images internes du corps humain comme les imageries par résonance magnétique (IRM) en 1971, les ultrasons en 1952, les scanners en 1979 et les tomographies d'émission monophonique et par émission de positons, etc. Ces représentations numériques du corps humain doivent être interprétées pour en faire des diagnostics qui sont généralement longs et prennent beaucoup de temps aux médecins.

Les chercheurs ont construit des systèmes pour automatiser les analyses. Initialement, l'analyse des images médicale était faites avec des traitements d'image basiques sur des pixels comme filtre, détection de contour, dilatation de région, binarisation, érosion ou avec des méthodes de modélisation mathématique (cf. cours [22])

À partir de ces techniques les experts font une récurrence de type "si .. alors ... sinon" sur l'image traitée pour pouvoir automatiser les analyses. Il était difficile de travailler avec ces images à cause de la résolution de l'image en ce temps-là où de l'importance du bruit interférentiel dû aux matériels d'acquisition utilisés ou juste parce que les informations

Dataset	Model	BLEU				METEOR
		BLEU-1	BLEU-2	BLEU-3	BLEU-4	
Flickr8k	Google NIC(Vinyals et al., 2014)$^{\dagger\Sigma}$	63	41	27	—	—
	Log Bilinear (Kiros et al., 2014a)$^{\circ}$	65.6	42.4	27.7	17.7	17.31
	Soft-Attention	**67**	44.8	29.9	19.5	18.93
	Hard-Attention	**67**	**45.7**	**31.4**	**21.3**	**20.30**
Flickr30k	Google NIC$^{\dagger\circ\Sigma}$	66.3	42.3	27.7	18.3	—
	Log Bilinear	60.0	38	25.4	17.1	16.88
	Soft-Attention	66.7	43.4	28.8	19.1	**18.49**
	Hard-Attention	**66.9**	**43.9**	**29.6**	**19.9**	18.46
COCO	CMU/MS Research (Chen & Zitnick, 2014)^{a}	—	—	—	—	20.41
	MS Research (Fang et al., 2014)$^{\dagger a}$	—	—	—	—	20.71
	BRNN (Karpathy & Li, 2014)$^{\circ}$	64.2	45.1	30.4	20.3	—
	Google NIC$^{\dagger\circ\Sigma}$	66.6	46.1	32.9	24.6	—
	Log Bilinear$^{\circ}$	70.8	48.9	34.4	24.3	20.03
	Soft-Attention	70.7	49.2	34.4	24.3	**23.90**
	Hard-Attention	**71.8**	**50.4**	**35.7**	**25.0**	23.04

FIGURE 2.10 – Résultat des différents modèles de génération de légende sur les collections MS-COCO , flickr8k et flickr30k par les systèmes d'évaluation bleu-n et meteor. source [91]

intéressantes sur l'image étaient petites voire même microscopiques.

Après l'apparition de l'apprentissage automatique, il a été possible de réaliser des classifications et segmentations plus ou moins bonnes et automatiques. Mais la procédure est encore difficile car il faut faire manuellement beaucoup d' extraction de variables clefs **feature engenering** pour avoir les caractéristiques pertinentes des données (comme les points d'intérêt de SIFT , histogramme de couleurs, GIST) [61, 87] avant de les mettre dans un classifieur et un régresseur (SVM, logistique régression, Randomforest) pour améliorer la performance de ces algorithmes.

Actuellement avec l'apprentissage profond, on arrive à automatiser la phase d'extraction de variables clefs, l'apprentissage profond est le choix le plus tendance dans l'analyse des images médicales. Dans la suite nous allons voir quelques utilisations de l'apprentissage profond.

2.4.1 Classification

La classification est l'application la plus directe de l'apprentissage profond non seulement dans l'imagerie médicale mais aussi dans d'autres domaines aussi. Dans les examens, on utilise la classification pour détecter la présence d'une maladie, d'un organe dans une ou une série d'images. On utilise généralement du **transfert d'apprentissage** car il n'y a pas de données suffisantes pour entraîner un modèle dans la plupart dans cas. Par exemple, Antony *et al.* [4] a obtenu une efficacité **(accuracy)** [4] de 57.6% pour déterminer les grades de l'entorse du genou en utilisant un transfert d'apprentissage de type réglage fin avec le modèle d'alexNet. Kim *et al.* [39] a obtenu une efficacité (accuracy) de 70.5% avec transfert d'apprentissage pour l'extraction de vecteur caractéristique pour la classifi-

4. Accuracy ou pertinence est une mesure de classification désignant le pourcentage de bonnes prédictions

cation cytopathologie. D'autres ont ré-implémentés leurs propres architectures de réseau de neurones ConvNet au lieu de faire un transfert d'apprentissage. Les résultats ne sont satisfaisants que si l'on dispose de suffisamment de données [92, 81, 67].

2.4.2 Détection et localisation

Il s'agit de localiser un élément (organe, lésion) dans le temps (suivre) et dans l'espace de l'image. D'habitude on ajoute un contour comme un cadre ou un cercle, ou mieux, une courbe qui convient bien à l'objet d'intérêt cadre de contour ou (bounding box). À la différence de la précédente classification, la détection est plus difficile car on veut des informations beaucoup plus précises. Le plus grand challenge de la détection dans l'imagerie médicale c'est la vitesse de la détection, la pertinence du cadre de contour et de pouvoir détecter les petits éléments [55, 88] car d'habitude on applique le modèle dans un flux d'images (vidéo) en direct que ce soit en 2D ou en 3D.

2.4.3 Segmentation

La segmentation a beaucoup d'importance en analyse d'image médicale surtout dans des domaines spécifiques comme la cardiologie ou la neurologie. Cette segmentation consiste à décider si un pixel ou voxel appartient à un élément ou son contour. U-Net [73] est un modèle de ConvNet spécialement créé pour faire de la segmentation des images biomédicales. Son architecture est une succession de couches de convolution avec différentes tailles et filtre down-conv, pour rétrécir, et de crop puis une succession de "déconvolution" up-conv pour agrandir et revenir à la taille initiale de l'image. Si l'on fait passer juste une image dans U-Net on a directement l'image segmentée à la sortie. De plus, une extension appelée V-Net [60] de U-Net supporte les images 3D.

2.4.4 Détection de concepts médicaux

Dans le cadre de la tâche Image CLEF 2017, plusieurs méthodes ont été proposées pour identifier les concepts des images. Ce sont des méthodes à base d'apprentissage profond qui ont été essentiellement proposées pour cette tâche. Pour représenter les informations visuelles des images, l'utilisation des CNN a été fréquente [17, 56, 2, 83].

D'autres travaux utilisent des méthodes plus conventionnelles d'extraction d'information des images comme SIFT, "bag of color" [2, 87]. Ils appliquent des méthodes de recherche de similarité pour identifier les images candidates et extraire les concepts à partir de ces images candidates.

Hassan *et al.* [28] a développé un modèle qui s'appuie sur la notion d'attention visuelle qui présente une amélioration considérable par rapport au simple transfert d'apprentissage de CNN Dans leur modèle les auteurs ont combiné un CNN et un RNN . Le RNN sert à générer les concepts un par un à partir du CNN.

Le tableau 4.3 présente les résultats obtenus sur l'ensemble de test de la collection Image CLEF 2017 lorsque la mesure de F1-Score est utilisée. Les modèles proposés par

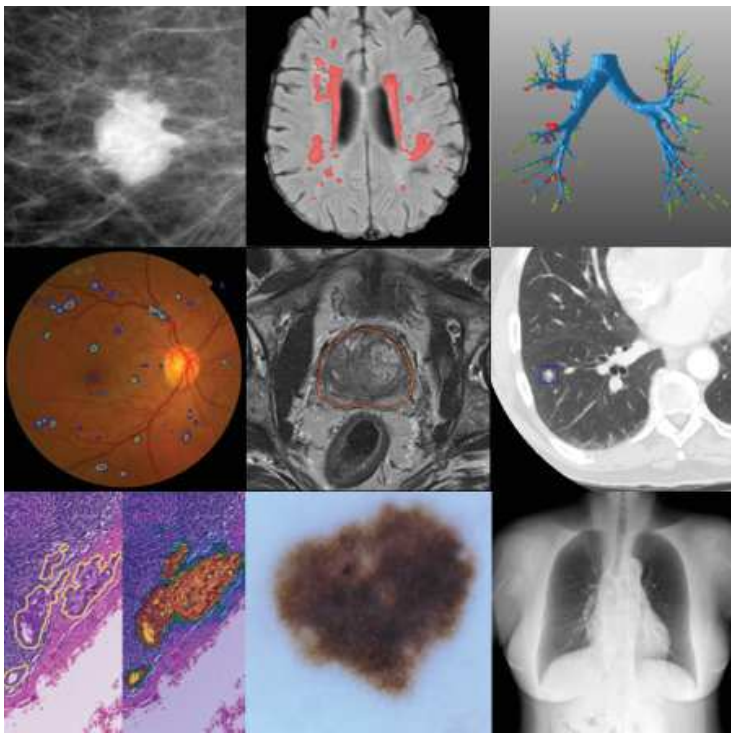


FIGURE 2.11 – Les applications de l'apprentissage profond dans l'imagerie médicale. Détection de lésion, segmentation. Image source : [55]

des participants à CLEF sont indiqués par une *.

2.4.5 Génération automatique de légende des images

C'est après l'apparition de la génération automatique de légende pour des images naturelles que l'on a commencées à en faire pour les images médicales.

Kilisev *et al* [48] utilise un CNN à plusieurs erreurs pour détecter et décrire une lésion ou une zone suspecte. Les légendes générées sont sur la description visuelle comme la taille, la forme ou la position des lésions.

Dans le cadre d'image CLEF 2017, les méthodes de génération automatique de légendes proposées par les participants sont les suivantes :

— Dans [71], les auteurs utilisent la génération de description par système d'extraction. Le système reprend les descriptions des images similaires à l'image qui est en cours de traitement.

— Les méthodes proposées dans [70, 2, 28, 53] sont une combinaison d'un CNN et d'un LSTM ou un SVM [5] pour le langage modèle. Leur méthode d'apprentissage est par *pipeline* (cf section 2.3)

Entre 2015 et 2016 plus de 300 papiers dans le domaine de l'apprentissage profond et du bio-médical ont été publiés, la plupart d'entre eux sont sur la segmentation et la détection [55] .

5. SVM :Support Vector Machine,C'est une algorithme d'apprentissage automatique

Tout au long de ce chapitre, nous avons introduit les concepts de base de l'apprentissage automatique ainsi que ceux de l'apprentissage profond. Puis nous avons vu les applications de l'apprentissage profond dans le domaine de l'imagerie médicale comme les détections et segmentations des organes dans une image. En dernier, nous avons fait une étude approfondie sur les méthodes de génération automatique de légendes pour des images naturelles.

Dans le chapitre suivant, nous appliquons ces connaissances dans l'expérience de génération automatique de légende pour des images médicales et la détection de concepts pour les mêmes images.

Chapitre 3

Nos propositions de modèles dans le domaine des images médicales

La recherche s'est déroulée en deux parties : (a) la recherche sur la détection de l'existence des concepts médicaux où on tente de trouver la présence de concepts dans une image et (b) la génération automatique de légende pour les mêmes images.

Ces deux problèmes étaient proposés par la campagne d'évaluation Image CLEF [19] à l'occasion de la conférence CLEF 2017 qui s'est déroulée à Dublin le 11 à 15 septembre 2017 [1].

Avant de passer aux détails des deux tâches, les images que nous utilisons doivent d'abord être prétraitées, ce que nous allons détailler dans la section suivante. Nous présenterons ensuite le détail de nos méthodes de transfert d'apprentissage pour la détection de concepts et enfin nous terminerons par la génération automatique de légende avec le modèle génératif entraîné par apprentissage de bout-en-bout.

3.1 Prétraitement des images

Le prétraitement des images est une étape importante pour la suite car il permet d'uniformiser les images à traiter pour ne pas faire des traitements spécifiques pour certaines images. En effet les images étaient sur différentes formes, différentes résolutions, des images en RGB et des images en niveau de gris. Les images sont toutes redimensionnées en taille 256*256 pixels par l'outil torchvision de pytorch [2]. Ce traitement d'images aidera aussi à gagner quelques places sur le disque. Les images en niveau de gris seront transformées en leur équivalent en forme RGB (par l'algorithme 4), originaire de [31].

Nous avons effectué une augmentation de données en appliquant des déformations géométriques aux images : symétrie horizontale et verticale. Ceci permet aux modèles de mieux distinguer les caractéristiques invariantes et d'augmenter le nombre d'images ayant les mêmes concepts et légendes.

1. http ://clef2017.clef-initiative.eu/
2. http ://pytorch.org/docs/master/torchvision/

Algorithm 4 Niveau de gris en RGB

Entrée : image img en niveau de gris 256*256
for c in $[r, g, b]$ **do**
  for $i = 0$ to 255 **do**
    for $j = 0$ to 255 **do**
      imgRgb[c,i,j] = img[i,j]
    end for
  end for
end for
Sortie : image en niveaux RGB de 256*256

3.2 Détection de concepts

La National Library of Medecine a instauré les UMLS (Unified Medical Language System). Le UMLs est un ensemble d'outils pour le développer des systèmes informatiques capables de comprendre le vocabulaire spécialisé, utilisé dans la biomédecine [9]. Les concepts médicaux font partie des UMLS-Metathesaurus qui est un ensemble de base de données des termes et vocabulaire biomédicaux sous forme de code. Il existe plusieurs sortes de base de données de concepts dans l'UMLS-Metathesaurus comme : MeSh (Medical Subject Headings) qui est le plus utilisé, SNOMED CT ou CUI. Dans notre cas, nous avons utilisé la base de données CUI.

Dans cette tâche, nous voulons détecter la présence des concepts pertinents pour des images médicales. La détection de concepts est une tâche intéressante car ces concepts peuvent représenter des informations sur la légende de l'image. Contrairement aux problèmes de détection d'organes ou d'objets, la détection de concepts est plus difficile car les concepts ne sont pas des contenus visuels dans l'image la plupart du temps. Les concepts peuvent désigner l'ensemble de l'image, son contexte, ou même l'environnement dans lequel l'image a été prise. Alors, une image donnée peut avoir un ou plusieurs concepts, dans ce cas on peut traiter ce problème comme une classification multi-label, c'est à dire qu'il faut prédire un ensemble de labels $Y_i = y_{i1}, .., y_{il_i}$ pour un X_i en entrée, mais non comme un problème de détection (localisation).

Dans cette section se trouve en première partie une analyse statistique de la donnée avec les relations entre images et concepts. Puis on passe par la manière d'associer les images à ses concepts pour les données entraînement. Enfin dans les deux dernières parties, nous élaborerons nos méthodes de détection de concepts, par les deux types de transfert d'apprentissage. Une avec le réglage fin de resNet50 et l'autre en utilisant VGG19 comme extracteur de variable caractéristique qu'on entraîne les variables caractéristiques dans un nouveau réseau de neurones. Les deux méthodes que nous avons utilisées sont tous les deux issues des deux types de transfert d'apprentissage et avec deux modèles de ConvNet différente parce que nous étions confrontés à des problèmes au niveau de la taille de la donnée d'apprentissage et matérielle.

3.2.1 Exploration de la données

La collection de données qu'nous avons utilisée était délivrée par l'image CLEF Lab en collaboration avec U.S. National Library of Medicine (NLM) [40, 19]. La collection contient en total 184,614 images associé à leur propre concept . La collection est séparée en 3 sous-ensembles :

— les données d'entraînement (training set) contiennent de 164 614 images est avec en tout, 20 463 concepts distincts

— les données de validation (validation set) contiennent 10 000 images et 7 070 concepts dont 309 de ces concepts ne sont pas inclue dans les concepts du training

— les données de tests (test set) contiennent 10 000 images ,les concepts sont à prédire.

La table 3.1 présente les caractères du training set. Nous avons trouvé aussi que 3,9% des images du training set et 3,79% des images de la validation n'ont pas de concept.

Nous avons approximativement 10-20 % des images qui sont composées [19] (voir par exemple la figure 3.1 ou ne sont pas des images médicales. Cela créé du bruit dans l'ensemble de données et rend la tâche plus difficile, sans la rendre forcément plus réaliste. Nous avons toutefois conservé des images qui sont dans la collection de référence pour permettre des comparaisons simples avec les autres études utilisant les mêmes données.

	Nombre de concept par image
mean	5.58
std	4.47
min	0.00
25%	3.00
50%	4.00
75%	7.00
max	75.00

	Nombre d'images par label
mean	44.95
std	320.06
min	1.00
25%	1.00
50%	3.00
75%	13.00
max	17,998.00

TABLE 3.1 – Analyse sur les images et concept pour la donnée d'apprentissage

3.2.2 Prétraitement des concepts

Les concepts associés aux images sont représentés sous forme de code (C1696103, C0040405,). D'après l'exploration de données, il y a 20463 concepts distincts dans la donnée d'entraînement. Nous avons ainsi réindexé ces concepts afin de les identifier. Pour associer une image à ses concepts, dans un vecteur appelé cible de taille 20463 initialiser à 0, on met un 1 à tous les indices des concepts associer à l'image.

Concept Id	UMLS terminologie	UMLS Type	nombre d'images associé
C1696103	Image-dosage form	Intellectual Product	17 998
C0040405	X-Ray Computed Tomography	Diagnostic Procedure	16 217
C0221198	Lesion	Finding	14 219
C1306645	Plain x-ray	Diagnostic Procedure	10 926
C0577559	Mass of body structure	Finding	9 769
C0027651	Neoplasms	Neoplastic Process	9 570
C0441633	Scanning	Diagnostic Procedure	9 289

TABLE 3.2 – Les concepts les plus fréquents dans la collection d'apprentissage

Donc, on aurait en moyenne 5.58 de 1 dans le vecteur cible et le reste est de 0 pour chaque image.

Remarque : Sous cette notation vectorielle de concepts, pour chaque image, on aurait $\|\{0,1\}\|^{20463} = 2^{20463}$ combinaisons de concepts possible. Or que le nombre de donnée d'entraînement 164614 est négligeable par rapport à 2^{20463} ($164614 \langle\langle 2^{20463}$)

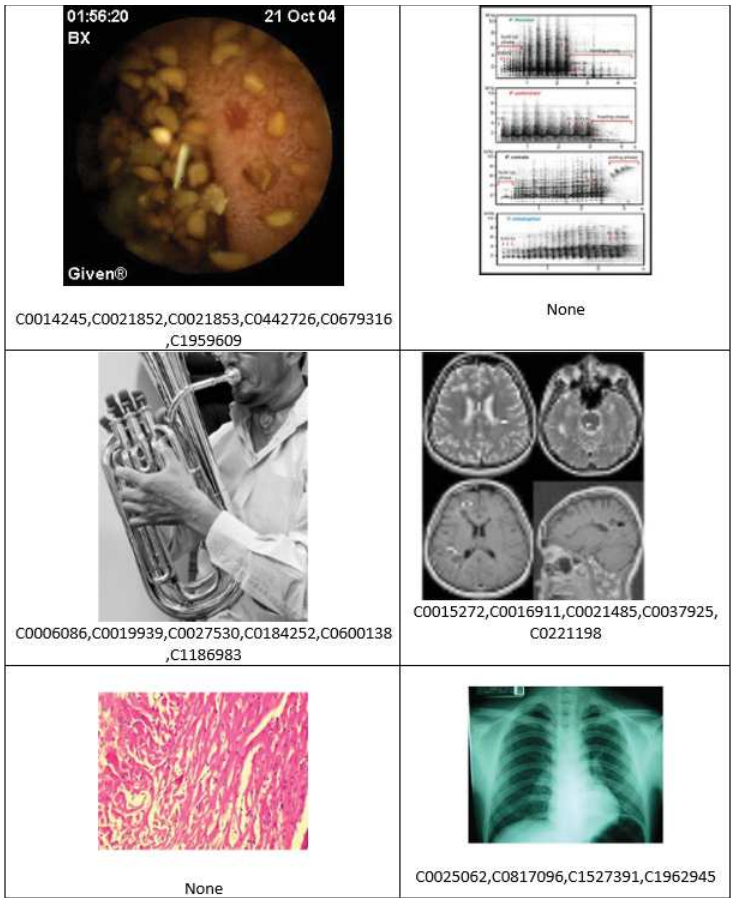


FIGURE 3.1 – Exemple d'image dans les données d'apprentissage d'ImageCLEF

3.2.3 Modèle 1 : Oxford VGG19 comme extracteur de variable

Dans ce premier modèle développé, nous avons utilisé le principe d'encodeur-décodeur qui est une stratégie utilisée pour générer automatiquement les légendes des images [89]. Dans notre cas, cette stratégie a été appliqué pour la classification multi-label des images. Nous avons développé est un transfert d'apprentissage avec Oxford VGG19 comme modèle de CNN de base [81] [3]

Nous avons choisi parce que l'Oxford VGG19 a déjà fait ces preuves [81] avec 7.3% d'erreur pour la classification d'image sur 1000 catégorie ImageNet Large Scale Visual Recognition Challenge (ILVRC 2012 [74]). [4] Comme l'illustre la figure 3.2, le CNN VGG19 contient en total 19 couches, avec une succession de couches de convolution et de mise en commun ; il est facile à implémenter sur **caffe**, et se finit avec une couche de sortie de 1000 classes. Le VGG19 est fait pour classifier des images dans 1000 classes mono-label. Nous avons donc adapté ce réseau au problème de classification multi-label que nous souhaitons ici résoudre (nous avons 20463 labels).

La création de ce modèle se fait en 2 étapes.

La première étape, la **phase d'encodage**, est d'extraire les vecteurs de représentation des variables pour chaque image. Cette extraction sera réalisée quelle que soit la provenance de l'image (ensemble d'apprentissage, de validation ou de test). Cette réduction de dimension permet de gagner beaucoup de temps et d'espace sur le disque. Ce vecteur de représentation se trouve à l'avant dernière couche du VGG19 (fully connected FC7 dans la Figure) ; la couche correspondante dans le réseau est composée de 4096 neurones. Ce vecteur résume les caractéristiques de l'image encodée, mais il est difficile d'interpréter ce vecteur comme le soulignent Sharif *et al.* [79]. De plus, d'après Bengio *et al.*, entraîner un nouveau modèle à partir de ces vecteurs offre des grandes potentialités pour des problèmes reconnaissance ou de classification [7, 14].

Par la suite, la deuxième étape consiste à utiliser ces vecteurs précédemment extraits dans un nouveau réseau de neurones, **phase de décodage**, où nous traitons le problème de la classification multi-label proprement dite. L'architecture de ce second réseau est la suivante : L'architecture du réseau est comme suit :

— Couche d'entrée : c'est cette couche qui reçoit les vecteurs de représentation de taille 4096, sortie du précédent réseau (étape de codage) ;

— Couche cachée : cette couche de taille 20463 que nous avons nommée **multi-label layer** permet de traiter l'aspect multi-label du problème ;

— Couche d'erreur (sigmoidCrossEntropy loss) : il s'agit d'une couche visant à optimiser le modèle. Avec la fonction d'activation sigmoïde sur la couche de sortie, le réseau de neurones modélise la probabilité d'un label l_j (voir équation 3.2).

Nous avons minimisé cette fonction d'erreur avec l'algorithme de descente de gradient

3. Le modèle VGG19 appris sur ILSRVC-2014 est disponible à `http://www.robots.ox.ac.uk/\%7Evgg/research/very_deep/`

4. The VGG19 model trained on ILSRVC-2014. `http://www.robots.ox.ac.uk/\%7Evgg/research/very_deep/`

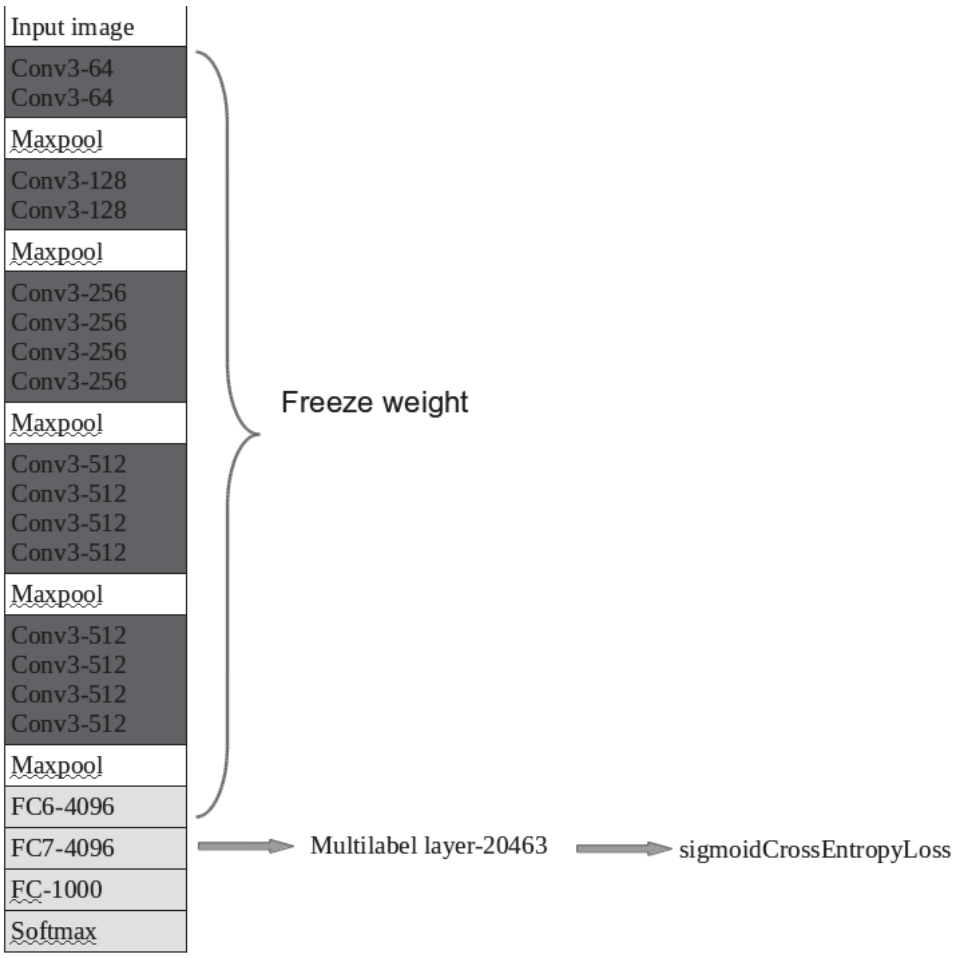


FIGURE 3.2 – Architecture du transfert d'apprentissage avec le convNet VGG19.

stochastique momentum avec les paramètres suivants : la taille du traitement par lot est de 256 images, le pas d'apprentissage est fixé à 0,0001, nous avons gardé la valeur par défaut du momentum (0,9).

3.2.4 Modèle 2 : réglage fin avec resNet50

Les résultats n'étaient optimums. Nous pensons que soit les vecteurs caractéristiques 4 096 n'apportent pas assez d'information pour le décodage, soit que le réseau VGG19 n'est pas adapté pour des images médicales. Nous avons développé un autre modèle à partir de Residual Network50 (resNet50) qui est aussi un CNN issu la collection ImageNet [5] avec seulement 3. 6% d'erreur [30]. De plus resNet50 est aussi utilisé en imagerie médicale [23, 41, 49].

La particularité du resNet est l'apprentissage par bloc résiduel. C'est dire que la valeur de sortie d'un bloc est la somme de la valeur entrée du bloc avec celui des résultats des calculs des couches dans le bloc comme les couche de convolution, relu (cf : 3.4). Ce qui rend son architecture difficile à implémenter, mais cela permet de conserver les informations d'origine.

Ce second modèle est très simple et ne nécessite qu'une seule étape. Dans la définition

5. `http://www.image-net.org/`

de l'architecture du resNet50, nous avons modifié la définition de la couche FC-1000 en **multi-label layer** comme précédemment et la couche d'erreur softmax loss en sigmoid entropy loss.

Ce modèle utilise les paramètres suivants lors de l'apprentissage : la taille du lot est de 50 images, le pas d'apprentissage est diminué selon l'équation 3.1 à chaque tranche de 7 époques [6].

Cette réduction du pas d'apprentissage est requise pour un réglage fin [45, 92].

$$lr = init\_lr * (0.1^{\frac{curent\_epoch}{decay\_epoch}}) \tag{3.1}$$

lr : pas d'apprentissage
$inti\_lr$: pas d'apprentissage initial = 0,001
$decay\_epoch$: tranche d'époques pour la réduction du pas d'apprentissage = 7
$curent\_epoch$: numéro de l'époque courante

6. Une époque est terminée lorsque l'ensemble des données est traité dans l'algorithme d'apprentissage.

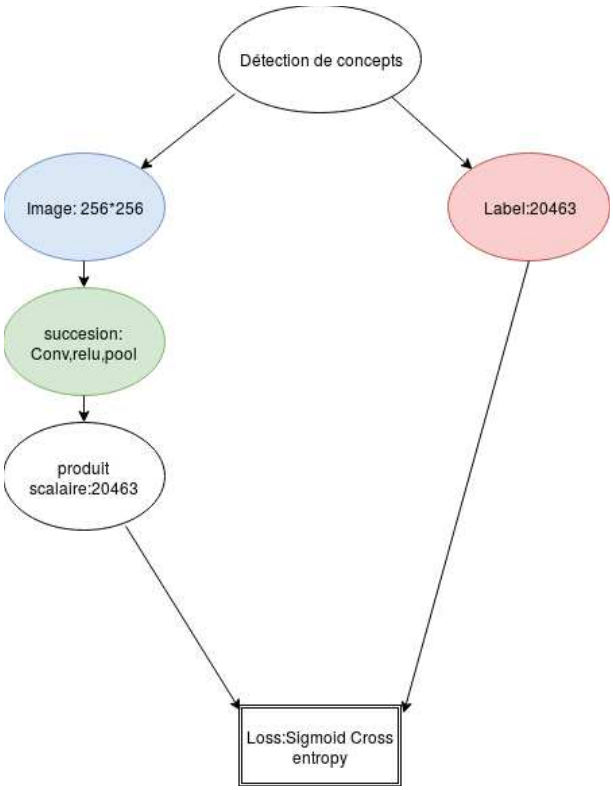


FIGURE 3.3 – Architecture du transfert d'apprentissage avec le convNet resNet50.

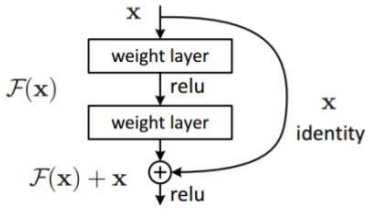


FIGURE 3.4 – Block résiduel, Image issue de [30]

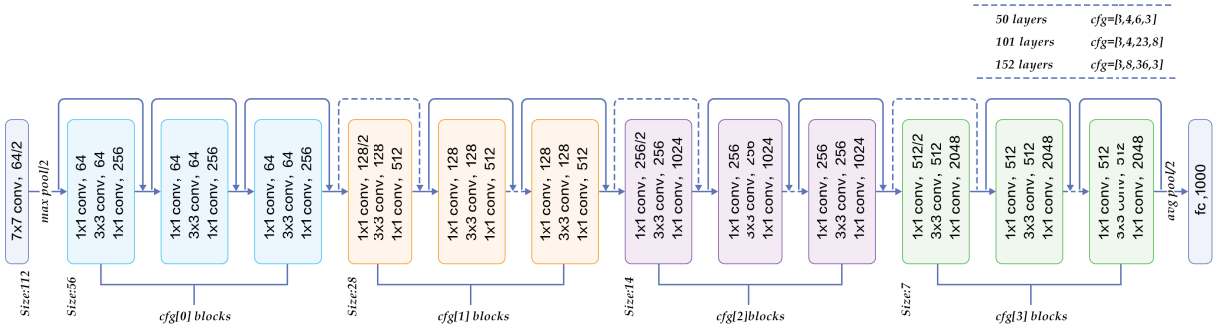


FIGURE 3.5 – ResNets avec 50, 101, 152 couches, respectivement. Tous les trois réseaux utilisent les blocs résiduel avec différents de répétitions. Image source : http ://book.paddlepaddle.org

Les deux modèles que nous avons développés ont été entraîné pour optimiser la fonction d'erreur sigmoide cross entropy (cf équation 3.2) ; en effet, la probabilité de chaque label ou concept étant considéré comme indépendant, cette fonction est adaptée [57, 90, 85, 11]

$$E = -\sum_{n=1}^{N} p(c_i)\log(\hat{p}(c_i)) + q(c_i)\log(\hat{q}(c_i)) \tag{3.2}$$

$p(c_i)$: 1 si le labels appartient à l'image 0 sinon. $q(c_i)) = 1 - p(c_i)$
$\hat{p}(c_i)$:c_i ième élément de $[1/(1+\exp(-s))] \in [0,1]$ avec s la valeur interne du neurone.

À part ces deux modèles, nous avons entraîné d'autres modèles mais ils n'ont pas eu de résultats concluants. Ils ont eu surtout des problèmes sur optimisation du sigmoïde Entropy sols parce que cela divergeait au lieu de chercher le minimum.

— : créé notre propre CNN ,

— VGG19 comme CNN extracteur suivie d'un SVM pour classifier les labels.

— traité le problème comme une classification mono-labels à 20463 classes et prendre les n meilleures classes. Mais les 20463 classes sont trop vastes et le modèle prédit toujours les mêmes classes

3.3 Annotation automatique d'image

L'objectif principal de cette tâche est de créer un modèle qui génère automatiquement les légendes des images médicales. Bien que les recherche sur ce même sujet ont toujours été destinées pour des images naturelles. Grâce à la conférence imageclef, on a pu étendre cette possibilité pour des images médicales. Dans le but d'améliorer les assistances médicales en créant un outil d'aide qui analyse/ décrit le contenu des images.

Pour le faire, on a adapté une approche par génération avec l'architecture de base de Vinalys parce que cette méthode est plus flexible et adaptative. Dans la partie suivante nous présenterons une brève exploration de la donnée que nous avons travaillée. Après nous passerons en détail les prétraitements des légendes du donnée entraînement. Enfin, on passera sur une étude approfondie du le modèle qu'on a utilisé.

3.3.1 Exploration de la donnée

Les données sur lesquelles on a travaillé sont ceux de l'imageCLEF, la même collection que pour la détection de concept. Chaque image dans les données d'entraînement est associée à ses légendes équivalentes qui, en moyenne avec 2.06 phrases (cf : Tableau 3.3).

	Nombre de légende par image
mean	2.06
std	1.83
min	0.00
25%	1.00
50%	1.00
75%	2.00
max	120.00

TABLE 3.3 – Analyse sur les légendes par image

3.3.2 Prétraitement des légendes

En se concentrant sur les légendes dans la donnée d'entraînement, en leurs états actuels elles sont encore inutilisables car elles sous forme de texte alors on ne peut pas faire des opérations mathématiques avec elles. Elles doivent aussi nettoyer, il faut enlever certains contenus indésirables pour ne pas fausser les données d'apprentissages.

3.3.2.1 Préparation des légendes

Dans cette partie, on présente les nettoyages apportés aux légendes. Les légendes auxquelles on avait travaillé présentaient beaucoup d'anomalie à traiter et qui sont volontairement mises par les organisateurs d'images clef pour traiter le problème.

Détecter les phrases (tokenise) : avec NLTK tokeniser, on a séparé les phrases des légendes.

Caractère spéciaux : telle que les ponctuation, symboles, smiley ont été enlevés, car elles posent des problèmes au niveau de l'encodage ascii de ces caractères. Et les chiffres aussi ont été enlevés.

Étiquettes : on a ajouté les étiquettes
$\langle UNK \rangle$: unknown pour les images qui n'ont pas de légende
$\langle BOS \rangle$ (begin of sentence) et $\langle EOS \rangle$ (end of sentence), au début et à la fin de chaque légende pour les délimiter.

À partir de là, on a construit notre dictionnaire de vocabulaire composé de vocabulaire anglais et des termes spécifiques en biomédical. Ce dictionnaire contient 162303 mots y compris les nouvelles étiquettes $\langle UNK \rangle$, $\langle BOS \rangle$, $\langle EOS \rangle$.

3.3.2.2 Représentation vectoriel de mot(word embedding)

Une fois les légendes complètement nettoyées, on transforme les mots en une valeur numérique. La manière la plus simple de faire cet encodage est d'utiliser un vecteur de sac de mots à valeur binaire. C'est-à-dire qu'on met un 1 à l'indice du mot du vocabulaire dans un vecteur de taille V, la taille du dictionnaire.

$$\overbrace{[0, 0, \ldots, 1, \ldots, 0, 0]}^{\text{—V—elements}}$$

Cette approche présente deux principaux problèmes :

— **sur la mémoire** : au fur et à mesure qu'on a un dictionnaire de grande taille,

— **sur la cooccurrence de mots** : il n'y a pas de notion de cooccurrence, de contexte, de relation entre les mots dans ce système d'encodage.

Parce que si on prend par exemple la mesure de similarité cosinus. Il n'y aurait pas de similarité entre deux mots quelque que soit les mots.

$mot1 = [0, 0, 0, 0, ..., 1, ..., 0, 0, 0, 0]$ le 1 se trouve à la i-eme position.

$mot2 = [0, 0, 0, ..., 1, ..., 0, 0, 0, 0, 0]$ 1 en j-eme position.

$cosine\_similarite(mot1, mot2) = (mot1.mot2)/||mot1||.||mot2||$ donc on aurait 0 de similarité car $i \neq j$.

Pour résoudre ces deux problèmes, on utilise le **word embedding**. C'est une technique pour représenter les qualités contextuel et "sémantique " des "mots" dans un vecteur réel de dimension réduite.

$$W : mots \mapsto \mathbb{R}^n \tag{3.3}$$

W : word embedding

Ainsi avec cette application, on peut utiliser des opérateurs arithmétiques vectoriels et la similarité sémantique des mots seront gardées.

Par exemple :
$W("king") + W("woman") = W("queen")$ et
similarité$(W("king"), W("woman")) \neq 0$

Dans notre cas, on a utilisé le word embedding de pytorch [69], C'est un modèle de réseaux de neurones à une couche cachée qui prend en entrée la taille du dictionnaire vocabulaire 162303 et la sortie la dimension désirer, pour notre cas on a choisi n=256.

3.3.3 Modèle pour la génération automatique de légende

Notre modèle de génération de légende suit toujours l'architecture de base proposé par Vinalys qui prend le problème en deux modules : le module encodeur d'image avec un réseau de neurones et un modèle de langue avec une RNN. (Figure 2.9 et 3.7)

Pour le module encodeur d'image, j'ai utilisé le même principe que dans la méthode deux de la détection de concept. Nous avons utilisé le précédent resNet50 comme convNet dont la définition de la dernière couche a été changée en 256 ; il s'agit de la taille du vecteur représentant un mot sous sa forme vectorielle.

C'est pour dire intuitivement qu'à chaque étape de l'apprentissage du langage-modèle, l'encodeur d'image encode l'image avec une représentation d'un mot encore inconnu et c'est le travail du modèle de langue de trouver ce mot encore inconnu.

Le module modèle de langue est un LSTM à 2 couche interne qui prend en entrée directement l'image précédemment encodé en vecteur de 256 et la sortie est une la représentation vectorielle du mot. Cet LSTM avec multiple couche permet de réaliser un traitement hiérarchique sur les tâches temporelles difficiles et de capturer plus naturellement la structure des séquences (la structure des descriptions) [32].

Après, on retransforme les représentations vectorielles des mots prédits à chaque instant du LSTM en vrais mots ou du moins à leur indice dans le dictionnaire de vocabulaire. Pour prédire le mot suivant, le LSTM prend en paramètre l'image, et les mots déjà prédites. (cf état de l'art section 2.3.2 , équation 2.24)

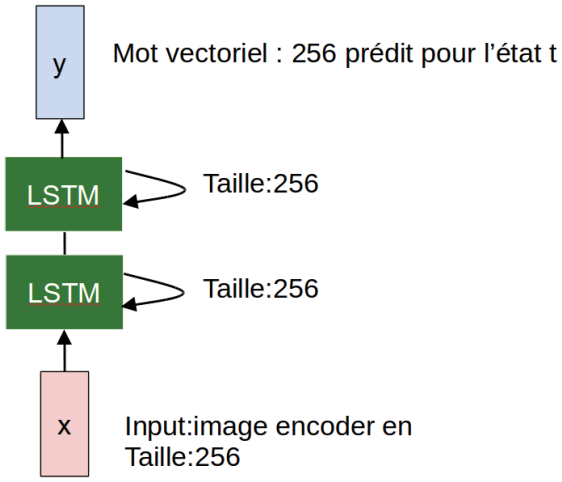


FIGURE 3.6 – modèle de langue : LSTM à 2 couches interne.

Par conséquent, on peut comparer si les légendes prédites par le modèle est égale aux légendes réelles de l'image (3.7).

L'architecture de modèle étant mis en place, la question qui se pose maintenant, c'est "comment on va entraîner les deux modules ?"

Est-ce qu'on va entraîner seulement le langage-modèle et utiliser le module d'encodeur d'image comme simple'encodeur ? (entraînement par pipeline) ou est-ce qu'on va entraîner en même temps les deux modules ensemble ?

Compte tenu des résultats des deux méthodes de détection de concepts où on a beaucoup travaillé avec les réseaux de neurones convolutifs, ces derniers représentent l'encodeur d'image dans un modèle génération de légende (cf résultats dans le tableau 4.3).

L'approche par extraction de variable n'a pas eu de très bons résultats pour la détection de même si on a entraîné le réseau très longtemps Par contre avec la méthode de réglage fin, on a eu une petite amélioration même si on ne l'a entraîné qu'avec quelques itérations.

Si on part de ces hypothèses, une approche par réglage fin dans la partie encodeur d'image est le mieux adapté pour notre donnée (de plus les modèles de convNets qu'on a utilisés étaient entraînés à partir d'images naturelles et non d'images biomédicales.)

Donc si on revient à notre modèle de génération automatique de légende, on a opté pour l'apprentissage de bout en bout. C'est-à-dire qu'on fait à la fois un réglage fin du réseau de neurones convolutif et un apprentissage du LSTM en même temps.

L'ensemble du modèle est alors entraîné pour minimiser la fonction **log de vraisemblance (cf équation 3.4)**. Cela permet au modèle de langue de chercher à chaque instant le mot suivant sachant l'image et les mots précédemment prédits.

$$Loss_i = \sum_t \log P(w_t | I^{(i)}, w_{t-1} ... w_{t-j}) \tag{3.4}$$

$Loss_i$: erreur pour l'image i
w_t : mot prédit pour l'instant t
I : image I

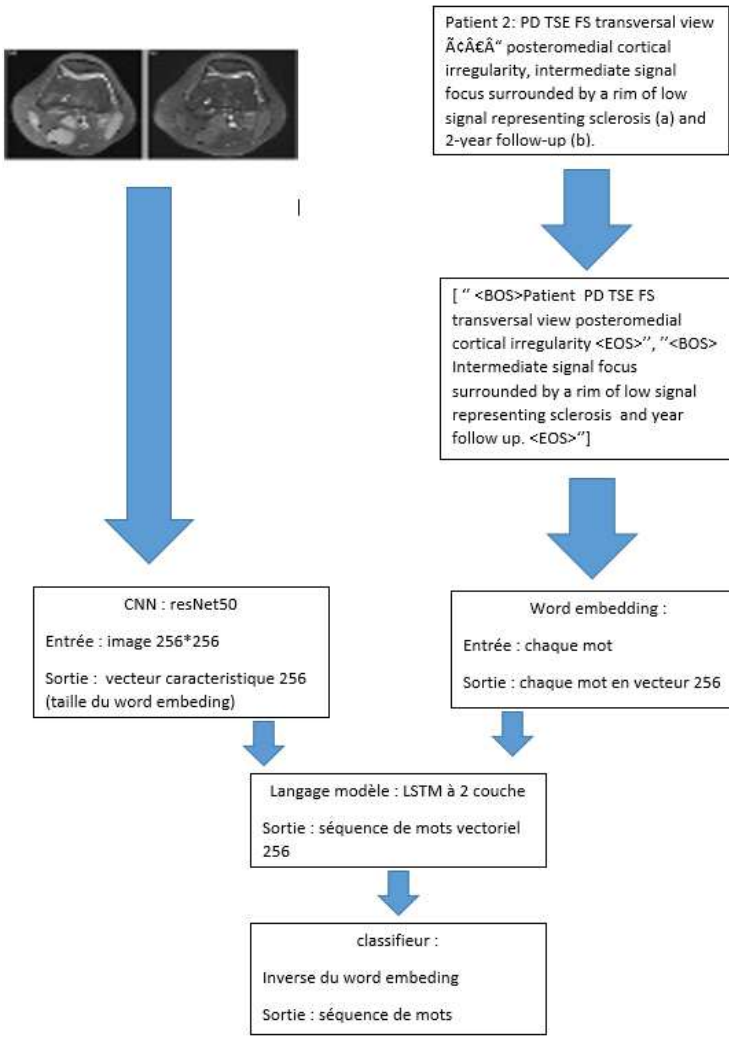


FIGURE 3.7 – Flux de travail pour la génération automatique de légende.

Chapitre 4

Implémentation ,résultat et évaluation

Dans ce chapitre, nous parlerons de l'implémentation des approches proposées en présentant les outils utilisés, les choix technologiques et le résultat des expérimentations.

4.1 Langage, librairie et matériels utilisés

Le langage de programmation choisi pour les 3 méthodes présentées précédemment est python. Python est un langage de programmation interprété, un langage de script et qui est un peu lent par rapport aux langages compilés comme C ou C++ pour faire des calculs. Python offre plusieurs libraires et est doter d'une communauté active. En ce qui concerne le traitement de données, les librairies numpy, pandas, et matplotlib, sont des librairies de calculs matriciels, d'analyses et de visualisation de données. Pour l'apprentissage profond, python possède des environnements de travail comme caffe, tensorflow, mxnet, keras, pytorch. Ces environnements de travail sont très utiles car ils permettent de gérer facilement l'algorithme de rétro-propagation [1] pour les réseaux de neurones de grande taille, les convNet, RNN, LSTM. Ils assurent aussi la parallélisation des calculs sur la GPU.

La méthode de détection de concepts avec extraction VGG19 a été faite avec caffe [42]. Caffe est spécialisé surtout pour l'entraînement des réseaux de neurones convolutif. Les deux autres méthodes ont été faites avec l'environnement pytorch. Pytorch est fait pour les réseaux dynamiques comme pour le cas de la génération de légende avec le CNN+ LSTM. Caffe et pytorch offrent des accès à la GPU. La GPU est très utile pour l'apprentissage profond grâce à sa capacité de faire des calculs en parallèle avec les mini processeurs appelés CUDA qui vont beaucoup accélérer la phase d'apprentissage parce qu'ils sont étroitement liés et partagent une mémoire unique. Bien que les GPUs soient difficiles à avoir sur le marché en particulier à Madagascar, grâce à notre collaboration avec l'Université de Toulouse, nous avons pu profiter de leur utilisation grâce à GRID5000 [2].

1. rétro-propagation : algorithme pour optimiser un réseau de neurone avec le SGD
2. https ://www.grid5000.fr/

	Méthode avec VGG19	Réglage fin resNet50	Génération automatique de légende
Origine matériel	Local	Grid5000-nancy	Grid5000-nancy
Matériel	MSI-core i5 16 Gb Ram	Grimani - 64Gb Ram NVIDIA Tesla K40	Grêle 128Gb ram NVIDIA GTX 1080

TABLE 4.1 – Matériel utilisé pour les différentes tâches

C'est une plate-forme qui offre des machines virtuelles pour les universités de France.

4.2 Résultats et évaluations

4.2.1 Détection de concepts

Pour évaluer la performance des modèles de détection de concepts, les organisateurs d'ImageCLEF 2017 ont proposé la mesure **F1-score**. F1-score représente la moyenne harmonique entre la précision et le rappel. La précision calcule le pourcentage des labels prédits qui sont pertinents, et le rappel calcule le pourcentage des labels pertinents qui sont prédits pour une image i.

$$rappel_i = \frac{\mid \widehat{y_i} \cap y_i \mid}{\mid y_i \mid} \tag{4.1}$$

$$precision_i = \frac{\mid \widehat{y_i} \cap y_i \mid}{\mid \widehat{y_i} \mid} \tag{4.2}$$

$$F1\_score_i = \frac{2 * rappel_i * precision_i}{rappel_i + precision_i} \tag{4.3}$$

On a ajouté la mesure de distance de **Hamming**, en plus de F1-score pour les données de validation seulement car on peut l'évaluer grâce aux concepts réels.

Le Hamming loss mesure le taux de labels qui sont mal classifiés dans une classification multi-label. C'est à dire les labels (concepts) réels qui ne sont pas prédits et les non labels réels qui sont prédits.

Le tableau 4.2 présente les différentes évaluations sur les différentes mesures pour la détection de concepts.

$$HamLoss = \frac{1}{N} \sum_{i=1}^{N} \frac{xor(x_i, y_i)}{L}. \tag{4.4}$$

où : N : nombre d'images

L : nombre de labels

y_i labels réels
x_i labels prédits

Nom du mesure	modèle	valeur
F1-score (moyenne)	extraction VGG19	0.047
	réglage fin resNet50	0.067
precision (moyenne)	VGG19	0.040
	resNet50	0.049
rappel (moyenne)	VGG19	0.07
	resNet50	0.014
Hamming loss	VGG19	0.0002
	resNet50	0.0002

TABLE 4.2 – Évaluation des prédictions sur les données de validation

On remarque que la valeur du Hamming loss est très petite pour les deux méthodes. Comme cette mesure définit le taux de mauvaise classification on pourrait conclure que ce taux est faible et que nos méthodes sont très efficaces. Mais c'est (partiellement) faux, comme on prédit un vecteur de taille 20463 et qu'il y a beaucoup de 0 et peu de 1 dans ce vecteur. Le Hamming loss prend les 0 prédits comme étant des bonnes prédictions, et c'est normal puis ce que ces concepts ne sont pas pertinents pour l'image.

Le F1-score est la moyenne entre les labels prédits et labels pertinents. Nos modèles arrivent à trouver 4% des concepts en général ; ils trouvent une ou deux concepts par images seulement.

Les figures 4.1, 4.2, 4.3 présentent des exemples de prédiction des concepts par les deux modèles.

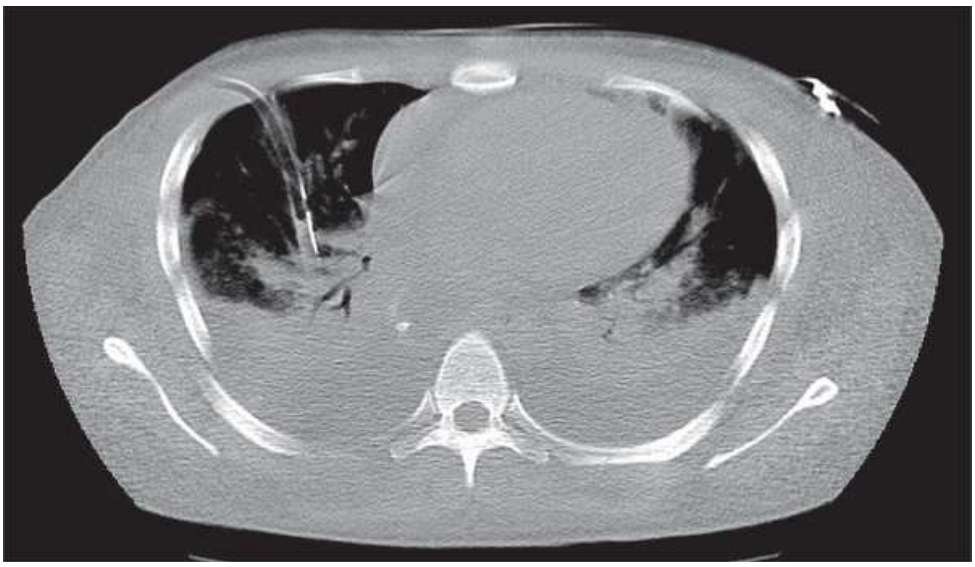


FIGURE 4.1 – **Prédit** : C0040405(Computed Tomography) C0027651(Neoplasms) ,C0441633(Scan) . **original** :C0008034(Chest Tube), C0032227(Pleural Effusion), C0040405(Computed Tomography), C0441587(Insertion)

FIGURE 4.2 – **Prédit :** C0577559 (mass of body structure) , C0000726(Abdomen) .**original** :C0000726(Abdomen)

	F1-Score
NLM [2] *	0.1718
IPL [87] *	0.1436
PRNA[28] *	0.1208
BMET[56] *	0.0958
resNet50	**0.0663**
VGG19	**0.0462**
PRNA [28] *	0.0234
NLM [2] *	0.0162
MPUN[83]*	0.0028

TABLE 4.3 – F1-score évalué par ImageCLEF sur les données de test. Les valeurs en gras sont celles que nous avons obtenues

FIGURE 4.3 – **Prédit :** C0040405(Diagnostic Procedure), C0087111(Therapeutic procedure),C0543467,C0577559(mass of body structure),C0817096(Body Location or Region),C1306645(X-ray NOS). **original :** C0035412(A malignant solid tumor), C0577559 (mass of body structure)

4.2.2 Génération automatique de légende

L'apprentissage de notre modèle d'apprentissage s'est arrêté au bout de 3 époques soit 2000 itérations de 50 lots d'images sur l'optimisation du sigmoid cross entropie loss avec la descente stochastique de gradient, à cause d'une limitation de l'utilisation de grid 5000. La figure 4.4 présente cette optimisation pour les données d'apprentissage et la figure présente les erreurs évaluées sur les données de validation à la fin de chaque époque.

On peut utiliser ce résultat d'apprentissage même si c'est seulement pour 3 époque parce que d'après la figure, l'erreur sur l'optimisation stagne aux alentours de 350 depuis l'époque 2.

L'évaluation des prédictions sur les données de validation et de test est BLEU-1 (cf Chapitre 2, section 2.3). La table 4.4 présente les résultats des autres équipes marqués par *, évaluer sur les données de test sur ImageCLEF. Malheureusement, nous n'avons pas pu soumettre nos prédictions à ImageCLEF mais on aurait 0.226 pour les données de test et 0.2561 de BLEU pour la donnée de validation. On pourrait interpréter ce résultat comme 25.61% des légendes prédites sont partiellement correctes pour la validation et 22.6% pour le test .

Les figures 4.5, 4.6, 4.7, 4.8, 4.9 illustrent des exemples de légendes générées par notre modèle et des légendes proposées par des spécialistes. La figure 4.9 est un cas de mauvaise prédiction parce que la phrase générée n'a pas de relation avec l'image et la légende orignale.

Dans la figure 4.8, nous rencontrons un cas où le modèle se perd dans la génération de légendes. Ceci est montré par le boucle "in the right lower lobe of the right lower lobe and thes..". Ce cas se présente beaucoup dans les images composées.

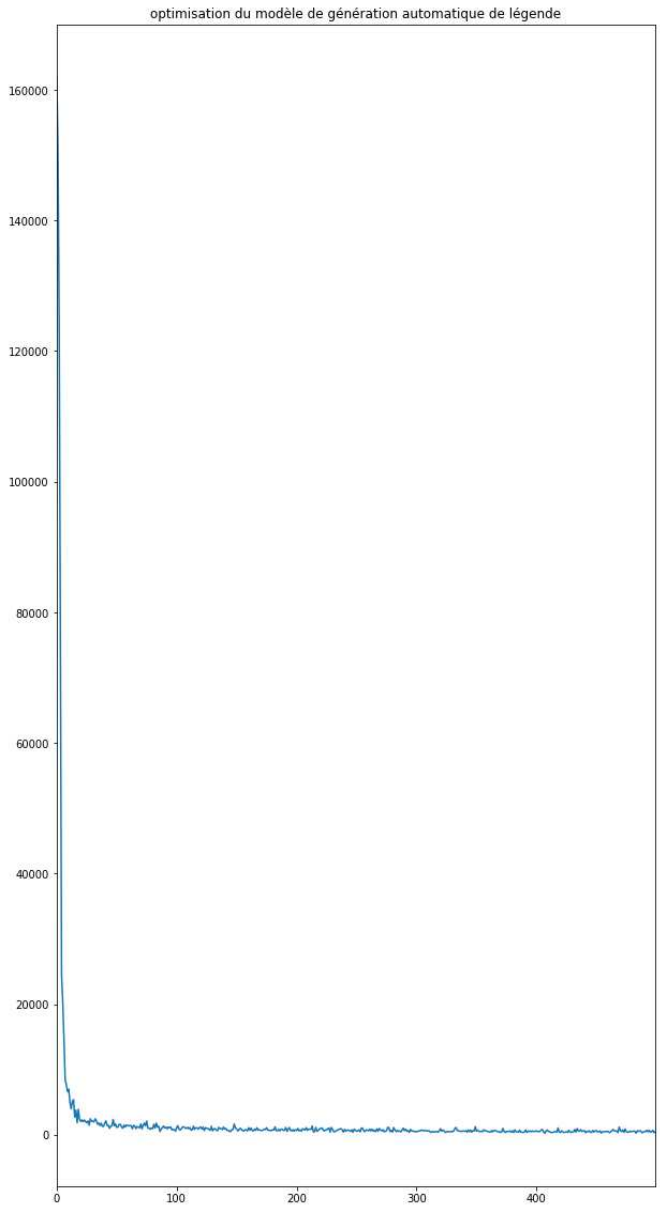


FIGURE 4.4 – Historique des 500 premières itérations de l' optimisation du log de vraisemblance pour le modèle de génération de légende sur les données d'entraînement ImageCLEF 2017. L'axe des X correspond au nombre d'itération alors que Y au log de vraisemblance des images

Team	BLEU
ISIA [53]*	0.2600
MAMI	**0.226**
NLM [2]*	0.2247
ISIA [53]*	0.2240
NLM [2]*	0.1384
NLM [2]*	0.1131
BMET [56]*	0.0982
BCSG [70]*	0.0749

TABLE 4.4 – Résultat de la génération de légende pour ImageCLEF caption citeImage-ClefCaption017 évaluer pour les données de test. Notre modèle est en gras.

FIGURE 4.5 – **Prédit** : clinical picture. **original** : Resected small bowel of three level of jejunoval

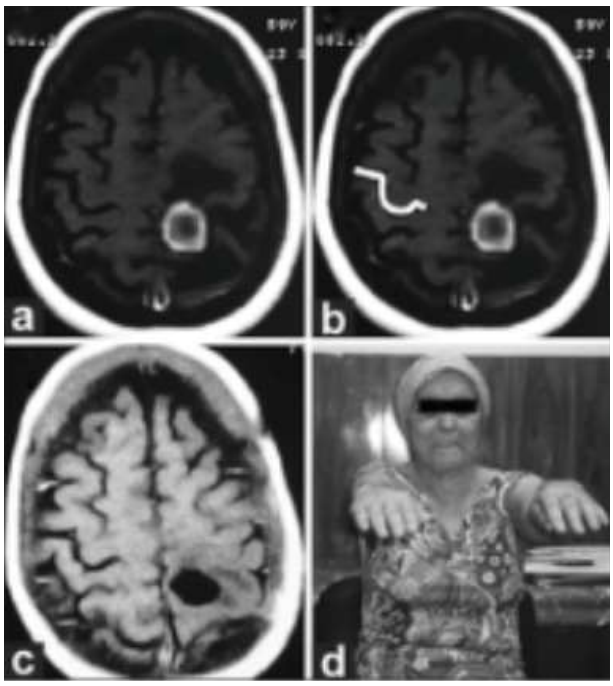


FIGURE 4.6 – **Prédit** : mri of the brain in a patient with high signal intesity in the right lobe. **Original** :(a and b) Pre-op axial MRI. Imaging reveals a lesion involving most of the left post-central gyrus and distorting normal left-brain anatomy. On the opposite side, the Omega shape is highlighted in white. (c) Post-op axial MRI. Imaging demonstrates appropriate excision. Histologically, a breast cancer metastasis was found. (d) Follow-up. No motor deficits seen at neurological examination

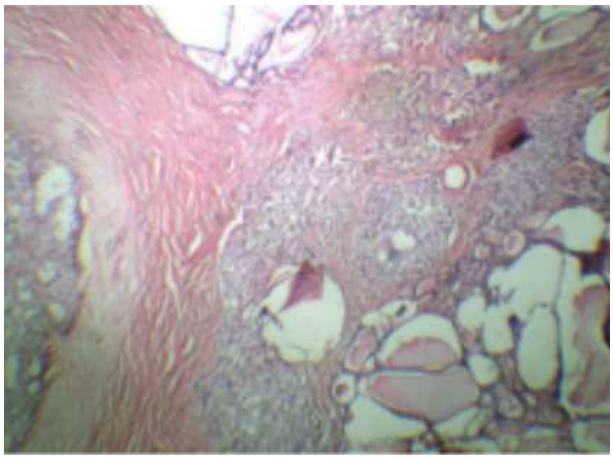


FIGURE 4.7 – **Prédit** : the tumor cells are seen in the dermis **Original** : Papillary cancer in the left lobe of the thyroid.

FIGURE 4.8 – **Prédit** : the arrow shows the presence of the lesion in the right lower lobe of the right lower lobe and the.the arrow shows the presence of the lesion in the right lower lobe of the right lower lobe with a. **Original** : Distal lesion. (A) Coronal and (B) sagittal magnetic resonance images demonstrating distal avulsion of the semitendinosus. (C) Incision and identification of the tendon. (D) Stripping of the tendon.

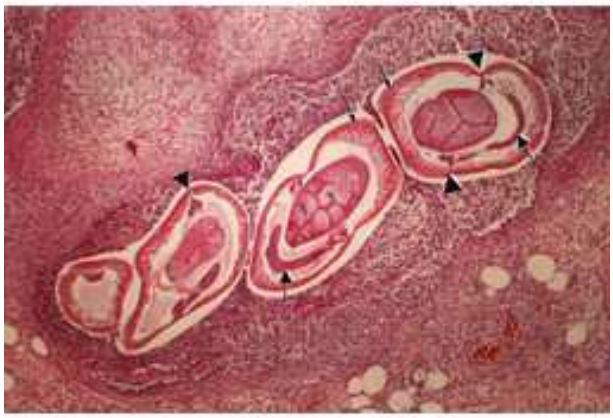


FIGURE 4.9 – **Prédit** : histological section of the resected specimen showing a giant cell carcinoma in the dermis . **Original** :Transverse sections of the Anisakis larva in the submucosa of the intestinal wall. Polymyarian muscle cells (thin arrows), separated into four quadrants by the chords, showing two wing-like distal lobes (arrow heads). Excretory cells are banana shaped (thick arrows) and situated ventrally to the intestine (staining ; original magnification

BLEU-1	0.5300
BLEU-2	0.35
BLEU-3	0.24
BLEU-4	0.17
METEOR	0.17

TABLE 4.5 – Résultats des évaluations des prédictions de légendes sur la collection MS-COCO

Nous avons aussi évalué notre modèle d'annotation automatique d'images pour la collection MS-COCO qui est une collection pour la génération automatique de légende pour des images naturelles. Les résultats des évaluations sur les scores bleus et météore sont dans la table 4.5. Les scores obtenus par le modèle sur MS-COCO sont bien meilleurs que sur les données médicales pour BLEU (Table 4.5 et 4.4). Ceci est dû à la qualité donnée d'entraînement .

FIGURE 4.10 – **Prédit** : a large elephant standing next to a baby elephant. **Original** : a mother elephant and calf walking with a herd of zebras in the background. an elephant leading a baby through a savannah plain. a grown elephant and a young elephant roam freely together in an open field.

FIGURE 4.11 – **Prédit** : a building with a clock on top of it. **Original** : a large clock tower with a wind indicator on top. blue and orange stone clock tower with a small clock. a clock tower has a weather vane on top of it

Chapitre 5

Conclusion et perspective

La finalité de ce stage est de concevoir un système de génération automatique de légende pour des images médicales de la base de données de la campagne d'évaluation internationale CLEF. Mais avant de réaliser cette tâche, nous avons dû passer beaucoup de temps sur la détection de concepts médicaux comme préliminaire. Ainsi nous nous sommes intéressés surtout aux techniques d'apprentissage profond pour y arriver car ce sont les techniques les plus en vogue aujourd'hui.

Concernant la détection des concepts médicaux, appelés CUI, nous avons transformé ce problème en un problème de classification multi-label. Nous avons expérimenté essentiellement 2 méthodes de transfert d'apprentissage de réseaux de neurones convolutif.

— Extraction de VGG19 puis extension d'un nouveau réseau de neurones pour la classification multi-label ;

— Adaptation des réseaux du neurones convolutif resNets50 pour la classification multi-label puis faire un réglage fin du modèle ;

Nous avons remarqué que les deux méthodes avaient tendance à prédire les CUIs les plus présents dans la collection d'entraînement. Nous avons pu améliorer notre résultat mais nous pensons que plus de recherche reste à faire comme établir un système qui extrait automatiquement les figures composées ou utiliser des CNN spécialisés dans les détections d'objets par région. Nous pourrions aussi envisager de réduire les concepts à prédire en les classifiant ou envisager des techniques de classification de grand vecteur qui est utilisé dans les traitements automatiques de la langue naturelle.

Concernant la génération automatique de légende des images, le modèle que nous avons proposé était une combinaison de CNN et LSTM parce que nous avons travaillé à la fois sur des images et des textes.

Comme la méthode par réglage fin a obtenu une amélioration des résultats dans la détection de concepts, nous avons gardé le resNet 50 comme CNN de base. Nous avons entraîné l'ensemble du modèle de bout en bout. C'est-à-dire que d'une manière plus simple, lors de l'apprentissage de l'ensemble du modèle (CNN+LSTM), nous avons fait à fois un réglage fin des poids CNN et l'apprentissage des poids du LSTM.

Nous pensons qu'utiliser les modèles de génération automatique de description avec notion d'attention visuelle serait très intéressant et utiles pour continuer ce projet car ces

modèles regardent les zones où il faut avoir de l'attention dans l'image [72, 52, 46]. En outre nous pourrions aussi utiliser les concepts comme information supplémentaire pour le modèle de langue [33, 93]. Nous pourrions aussi envisager d'utiliser l'apprentissage par renforcement pour avoir un modèle évolutif de génération de légende.

Pour terminer, la génération automatique de légendes pour les images est une tâche très intéressante et présente un axe de recherche qui nécessite d'être développé parce qu'elle peut présenter de nombreux avantages si on la propose dans les institutions médicales.

Ces travaux font suite aux travaux de master de M. Nomena NY HOAVY [33]. En outre, ces travaux ont été publiés dans les actes de la conférence ImageCLEF 2017 et présentés sous forme de poster [62]. Finalement, nous avons soumis un papier à la conférence nationale CORIA [1].

Annexes

Concept detection with Transfer learning

Nomena Ny Hoavy, Josiane Mothe, Mamitiana Ignace Randrianarivony

INTRODUCTION

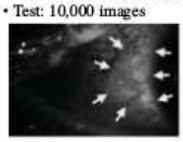
- Concept detection and captioning for medical image such radiology, IRM can improve the rapidity of a treatement at medical institution. Thus there is a need to automated those two task. The main goal of the concept detection task is to identify the existence of relevant biomedical concepts in the medical images. We develop an automated concept detection using methode based a transfert learning with an pretrained model of CNN

Problem Statement

- Identify the existence of relevant biomedical concepts in the medical images
- Classification problem where each class corresponds to a concept label. An image may contain one or many concepts

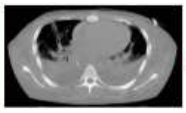
Datasets

- Image Clef provides a dataset of medical images
- Training data:164,614 images and 20,463 concept labels
- Validation :10,000 images and 7,070 concept labels
- Test: 10,000 images



C0229089: Right Eye
C1512346: Patient Visit
C1761613: Conjunctival redness

Results



Predicted concept
C0040405: Computed Tomography
C0000726: Abdomen
C0027651: Neoplasms
C0441633 :Scan

Model for multilabel classification

Original concept
C0008034:Chest Tube
C0032227:Pleural Effusion
C0040405: Computed Tomography
C0441587:Insertion

F1-score	0.047
MR	0.041
HL	0.0002

HL=Hamming loss
MR=Matching ratio

Experimentation on validation dataset.

F1-score	0.046

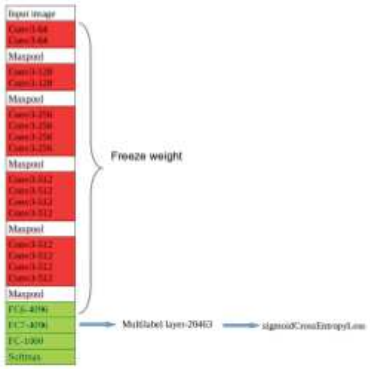
Result on test data release

Model

•An architecture based on VGG19 by Oxford Model
. Extended it to a multilabel classification for concept detection

Because Oxford VGG19 model performed well at ILVRC 2012 for classifying images over 1,000 classes

- Train above the VGG19 new layer for multilabel classification from the FC7-4096



Architecture of the network CNN

References

[1] Russakovsky, Olga and Deng , Imagenet large scale visual recognition challenge, International Journal of Computer Vision, 2015
[2] Simonyan, Karen and Zisserman, Andrew, Very deep convolutional networks for large-scale image recognition, 2014
[3] Wei, Yunchao and Xia,CNN: Single-label to multi-label, 2014

Exemples de code

```
imgs = Variable(imgs, volatile=not training)
captions = Variable(captions, volatile=not training)
input_captions = captions[:-1]
target_captions = pack_padded_sequence(captions, lengths)[0]

pred, _ = model(imgs, input_captions, lengths)
err = loss(pred, target_captions)
perplexity.update(math.exp(err.data[0]))

if training:
    optimizer.zero_grad()
    err.backward()
    clip_grad_norm(model.rnn.parameters(), grad_clip)
    optimizer.step()

# measure elapsed time
batch_time.update(time.time() - end)
end = time.time()
if i % print_freq == 0:
    logging.info('{phase} - Epoch: [{0}][{1}/{2}]\t'
                 'Time {batch_time.val:.3f} ({batch_time.avg:.3f})\t'
                 'Data {data_time.val:.3f} ({data_time.avg:.3f})\t'
                 'Loss {perp.val:.4f} ({perp.avg:.4f})'.format(
                     epoch, i, len(data),
                     phase='TRAINING' if training else 'EVALUATING',
                     batch_time=batch_time,
                     data_time=data_time, perp=perplexity))
```

FIGURE 5.1 – Partie code pour l'entraînement

```
pathModel ="/home/mamitiana/pydevWorkspace/captionGeneration/results/2017-08-30_14-51-16/checkpoint_epoch_3.pth.tar"

valDset=ImageClefDataset(config.Configuration.valImages,config.Configuration.valext,'val')

batch_size =50
model= torch.load(pathModel)['model']

for i  in range ( len(valDset)):
    img ,_,_ = valDset[i]
    print(i)
    captions = model.generate(img)
    print(captions)
    valDset.setCaption(i,captions)

valDset.exportPrediction('valAut.csv')
```

FIGURE 5.2 – Partie code pour prédiction

Bibliographie

[1] Coria-taln-rjc. `https://project.inria.fr/coriataln2018/fr/`. Last accessed : 2018-02-22.

[2] Abacha, A. B., de Herrera, A. G. S., Gayen, S., Demner-Fushman, D., and Antani, S. (2017). NLM at ImageCLEF 2017 Caption Task.

[3] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al. (2016). Tensorflow : Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv :1603.04467*.

[4] Antony, J., McGuinness, K., O'Connor, N. E., and Moran, K. (2016). Quantifying radiographic knee osteoarthritis severity using deep convolutional neural networks. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 1195–1200. IEEE.

[5] Aparicio IV, M., Levine, D. S., and McCulloch, W. S. (1994). Why are neural networks relevant to higher cognitive function. *Neural networks for knowledge representation and inference*, pages 1–26.

[6] Beckham, C. and Pal, C. (2016). A simple squared-error reformulation for ordinal classification. *arXiv preprint arXiv :1612.00775*.

[7] Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning : A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8) :1798–1828.

[8] Bernardi, R., Cakici, R., Elliott, D., Erdem, A., Erdem, E., Ikizler-Cinbis, N., Keller, F., Muscat, A., and Plank, B. (2016). Automatic description generation from images : A survey of models, datasets, and evaluation measures. *J. Artif. Intell. Res.(JAIR)*, 55 :409–442.

[9] Bodenreider, O. (2004). The unified medical language system (umls) : integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1) :D267–D270.

[10] Bolze, R., Cappello, F., Caron, E., Daydé, M., Desprez, F., Jeannot, E., Jégou, Y., Lanteri, S., Leduc, J., Melab, N., et al. (2006). Grid'5000 : A large scale and highly reconfigurable experimental grid testbed. *The International Journal of High Performance Computing Applications*, 20(4) :481–494.

[11] Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer.

[12] Bozal Chaves, A. (2017). Personalized image classification from eeg signals using deep learning. B.S. thesis, Universitat Politècnica de Catalunya.

[13] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv :1406.1078*.

[14] Coates, A., Lee, H., and Ng, A. Y. (2010). An analysis of single-layer networks in unsupervised feature learning. *Ann Arbor*, 1001(48109) :2.

[15] Csáji, B. C. (2001). Approximation with artificial neural networks. *Faculty of Sciences, Etvs Lornd University, Hungary*, 24 :48.

[16] Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4) :303–314.

[17] Dimitris, K. and Ergina, K. (2017). Concept detection on medical images using deep residual learning network.

[18] Dumoulin, V. and Visin, F. (2016). A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv :1603.07285*.

[19] Eickhoff, C., Schwall, I., García Seco de Herrera, A., and Müller, H. (2017). Overview of ImageCLEFcaption 2017 - image caption prediction and concept detection for biomedical images. *CLEF working notes, CEUR*.

[20] Ekins, S. (2016). The next era : Deep learning in pharmaceutical research. *Pharmaceutical research*, 33(11) :2594–2603.

[21] Elliott, D. and Keller, F. (2014). Comparing automatic evaluation measures for image description. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2 : Short Papers)*, volume 2, pages 452–457.

[22] Fenohery (2016). Imagerie médicale. MISA.

[23] Filippov, S., Moiseev, A., and Andrey, A. (2017). Deep diagnostics : Applying convolutional neural networks for vessels defects detection. *arXiv preprint arXiv :1705.06264*.

[24] Gehring, J., Auli, M., Grangier, D., and Dauphin, Y. N. (2016). A convolutional encoder model for neural machine translation. *arXiv preprint arXiv :1611.02344*.

[25] Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587.

[26] Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. (2017). Lstm : A search space odyssey. *IEEE transactions on neural networks and learning systems*.

[27] Gupta, A., Verma, Y., Jawahar, C., et al. (2012). Choosing linguistics over vision to describe images. In *AAAI*, page 1.

[28] Hasan, S. A., Ling, Y., Liu, J., Sreenivasan, R., Anand, S., Arora, T. R., Datla, V., Lee, K., Qadir, A., Swisher, C., et al. (2017). Prna at imageclef 2017 caption prediction and concept detection tasks.

[29] Hassoun, M. H. (1995). *Fundamentals of artificial neural networks*. MIT press.

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

[31] Helland, T. Seven grayscale conversion algorithms (with pseudocode and vb6 source code). `http://www.tannerhelland.com/3643/grayscale-image-algorithm-vb6/`. Last accessed : 2018-02-22.

[32] Hermans, M. and Schrauwen, B. (2013). Training and analysing deep recurrent neural networks. In *Advances in neural information processing systems*, pages 190–198.

[33] Hoavy, N. N. (2016). Annotation automatique d'images par apprentissage profond :génération automatique de descriptions d'une image. Thèse de Master, MISA.

[34] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8) :1735–1780.

[35] Hodosh, M., Young, P., and Hockenmaier, J. (2013). Framing image description as a ranking task : Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47 :853–899.

[36] Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5) :359–366.

[37] Hosang, J., Benenson, R., and Schiele, B. (2014). How good are detection proposals, really ? *arXiv preprint arXiv :1406.6962*.

[38] Huval, B., Coates, A., and Ng, A. (2013). Deep learning for class-generic object detection. *arXiv preprint arXiv :1312.6885*.

[39] Hwang, S., Kim, H.-E., Jeong, J., and Kim, H.-J. (2016). A novel approach for tuberculosis screening based on deep convolutional neural networks. *Medical Imaging*, 9785 :97852W–1.

[40] Ionescu, B., Müller, H., Villegas, M., Arenas, H., Boato, G., Dang-Nguyen, D.-T., Dicente Cid, Y., Eickhoff, C., Garcia Seco de Herrera, A., Gurrin, C., Islam, B., Kovalev, V., Liauchuk, V., Mothe, J., Piras, L., Riegler, M., and Schwall, I. (2017). Overview of ImageCLEF 2017 : Information Extraction from Images. In *CLEF 2017 Proceedings*, volume 10456 of *Lecture Notes in Computer Science*, pages 315–337, Dublin, Ireland. Springer.

[41] Islam, M. T., Aowal, M. A., Minhaz, A. T., and Ashraf, K. (2017). Abnormality detection and localization in chest x-rays using deep convolutional neural networks. *arXiv preprint arXiv :1705.09850*.

[42] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014). Caffe : Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM.

[43] Jozefowicz, R., Zaremba, W., and Sutskever, I. (2015). An empirical exploration of recurrent network architectures. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 2342–2350.

[44] Karpathy, A. (2015). The unreasonable effectiveness of recurrent neural networks. *Andrej Karpathy blog*.

[45] Karpathy, A. (2016). Cs231n : Convolutional neural networks for visual recognition. *Neural networks*, 1.

[46] Karpathy, A. and Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3128–3137.

[47] Karpathy, A., Johnson, J., and Fei-Fei, L. (2015). Visualizing and understanding recurrent networks. *arXiv preprint arXiv :1506.02078*.

[48] Kisilev, P., Sason, E., Barkan, E., and Hashoul, S. (2016). Medical image description using multi-task-loss cnn. In *International Workshop on Large-Scale Annotation of Biomedical Data and Expert Label Synthesis*, pages 121–129. Springer.

[49] Laina, I., Rieke, N., Rupprecht, C., Vizcaíno, J. P., Eslami, A., Tombari, F., and Navab, N. (2017). Concurrent segmentation and localization for tracking of surgical instruments. *arXiv preprint arXiv :1703.10701*.

[50] LeCun, Y., Jackel, L., Bottou, L., Cortes, C., Denker, J. S., Drucker, H., Guyon, I., Muller, U., Sackinger, E., Simard, P., et al. (1995). Learning algorithms for classification : A comparison on handwritten digit recognition. *Neural networks : the statistical mechanics perspective*, 261 :276.

[51] Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. (2009). Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pages 609–616. ACM.

[52] Li, L., Tang, S., Deng, L., Zhang, Y., and Tian, Q. (2017). Image caption with global-local attention. In *AAAI*, pages 4133–4139.

[53] Liang, S., Li, X., Zhu, Y., Li, X., and Jiang, S. (2017). Isia at the imageclef 2017 image caption task.

[54] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco : Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

[55] Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., van der Laak, J. A., van Ginneken, B., and Sánchez, C. I. (2017). A survey on deep learning in medical image analysis. *arXiv preprint arXiv :1702.05747*.

[56] Lyndon, D., Kumar, A., and Kim, J. (2017). Neural captioning for the imageclef 2017 medical image challenges.

[57] Mencía, E. L. and Fürnkranz, J. (2010). Efficient multilabel classification algorithms for large-scale problems in the legal domain. In *Semantic Processing of Legal Texts*, pages 192–215. Springer.

[58] Michalski, R. S., Carbonell, J. G., and Mitchell, T. M. (2013). *Machine learning : An artificial intelligence approach*. Springer Science & Business Media.

[59] Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Interspeech*, volume 2, page 3.

[60] Milletari, F., Navab, N., and Ahmadi, S.-A. (2016). V-net : Fully convolutional neural networks for volumetric medical image segmentation. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 565–571. IEEE.

[61] Min, S., Lee, B., and Yoon, S. (2017). Deep learning in bioinformatics. *Briefings in bioinformatics*, 18(5) :851–869.

[62] Mothe, J., Hoavy, N. N., and Randrianarivony, M. I. (2017). IRIT & MISA at ImageCLEF 2017 - Multi label classification. In *Working Notes of CLEF 2017 - Conference and Labs of the Evaluation Forum, Vol-1866*. CEUR-WS.

[63] Mothe, J., Hoavy, N. N., and Randrianarivony, M. I. (2018). Classification multi-label à grande dimension pour la détection de concepts médicaux. In *Conférence en Recherche d'Information et Applications, CORIA'18 (soumis)*.

[64] Najman, M. Image captioning with convolutional neural networks.

[65] Ng, A. (2016). Machine learning.

[66] Ordonez, V., Kulkarni, G., and Berg, T. L. (2011). Im2text : Describing images using 1 million captioned photographs. In *Advances in Neural Information Processing Systems*, pages 1143–1151.

[67] Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10) :1345–1359.

[68] Parizeau, M. (2004). Réseaux de neurones. *GIF-21140 et GIF-64326*, 124.

[69] Paszke, A., Gross, S., Chintala, S., et al. (2017). Pytorch.

[70] Pelka, O. and Friedrich, C. M. Keyword generation for biomedical image retrieval with recurrent neural networks.

[71] Rahman, M. M., Lagree, T., and Taylor, M. (2017). A cross-modal concept detection and caption prediction approach in imageclefcaption track of imageclef 2017.

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

[73] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net : Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer.

[74] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3) :211–252.

[75] Samuel, A. L. (2000). Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, 44(1.2) :206–226.

[76] Schlegl, T., Waldstein, S. M., Vogl, W.-D., Schmidt-Erfurth, U., and Langs, G. (2015). Predicting semantic descriptions from medical images with convolutional neural networks. In *International Conference on Information Processing in Medical Imaging*, pages 437–448. Springer.

[77] Schuster, M. and Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11) :2673–2681.

[78] Shankar, S., Garg, V. K., and Cipolla, R. (2015). Deep-carving : Discovering visual attributes by carving deep neural nets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3403–3412.

[79] Sharif Razavian, A., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). Cnn features off-the-shelf : an astounding baseline for recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 806–813.

[80] Shin, H.-C., Roberts, K., Lu, L., Demner-Fushman, D., Yao, J., and Summers, R. M. (2016). Learning to read chest x-rays : recurrent neural cascade model for automated image annotation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2497–2506.

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

[82] Singha, M. and Hemachandran, K. (2012). Signal & image processing : An international journal (sipij). *Content Based Image Retrieval using Color and Texture*, 3(1) :39–57.

[83] Stefan, L.-D., Ionescu, B., and Müller, H. (2017). Generating captions for medical images with a deep learning multi-hypothesis approach : Medgift–upb participation in the imageclef 2017 caption task.

[84] Sutskever, I., Martens, J., Dahl, G., and Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147.

[85] Tsoumakas, G., Katakis, I., and Vlahavas, I. (2009). Mining multi-label data. In *Data mining and knowledge discovery handbook*, pages 667–685. Springer.

[86] Uijlings, J. R., Van De Sande, K. E., Gevers, T., and Smeulders, A. W. (2013). Selective search for object recognition. *International journal of computer vision*, 104(2) :154–171.

[87] Valavanis, L. and Stathopoulos, S. (2017). IPL at ImageCLEF 2017 concept detection task. In *Working Notes of CLEF 2017 - Conference and Labs of the Evaluation Forum, Vol-1866*. CEUR-WS.

[88] van Grinsven, M. J., van Ginneken, B., Hoyng, C. B., Theelen, T., and Sánchez, C. I. (2016). Fast convolutional neural network training using selective data sampling : Application to hemorrhage detection in color fundus images. *IEEE transactions on medical imaging*, 35(5) :1273–1284.

[89] Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. (2015). Show and tell : A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164.

[90] Wei, Y., Xia, W., Huang, J., Ni, B., Dong, J., Zhao, Y., and Yan, S. (2014). Cnn : Single-label to multi-label. *arXiv preprint arXiv :1406.5726*.

[91] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. (2015). Show, attend and tell : Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057.

[92] Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks ? In *Advances in neural information processing systems*, pages 3320–3328.

[93] You, Q., Jin, H., Wang, Z., Fang, C., and Luo, J. (2016). Image captioning with semantic attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4651–4659.

[94] Yu, Y., Ko, H., Choi, J., and Kim, G. (2017). End-to-end concept word detection for video captioning, retrieval, and question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3165–3173.

[95] Zaremba, W., Sutskever, I., and Vinyals, O. (2014). Recurrent neural network regularization. *arXiv preprint arXiv :1409.2329*.

[96] Zhou, Z.-H., Zhang, M.-L., Huang, S.-J., and Li, Y.-F. (2012). Multi-instance multi-label learning. *Artificial Intelligence*, 176(1) :2291–2320.

Résumé

Dans ce document, nous présentons les méthodes d'extraction d'information que nous proposons à partir d'images médicales. L'objectif de ces méthodes est d'associer automatiquement des concepts issus d'un vocabulaire (concepts UML) à une image donnée mais également de générer automatiquement une légende pour cette image. Nous avons sommes focalisés sur les techniques d'apprentissage profond et sur la collection de données de référence ImageCLEF Caption 2017. Nous avons traité le problème de la détection de concepts par une classification multi-label à grande dimension. L'aspect multi-label réfère au fait qu'une image peut être associée à plusieurs classes ou concepts ; la grande dimension s'explique elle par le fait qu'il a une grande variété de concepts dans la base de données. Dans ce travail, nous proposons deux méthodes de transfert d'apprentissage à partir des réseaux de neurones convolutif Visual Geometry Group 19 et Residual network 50 pour extraire les concepts de chaque image. Pour la partie annotation automatique d'images, la tâche est de générer un texte qui correspond à la description de l'image. Nous avons utilisé l'architecture classique du type encodeur-décodeur proposée par Vinalys pour la génération de légendes. L'ensemble de ce modèle est entraîné par apprentissage de bout en bout dans lequel l'erreur est rétro-propagée sur le modèle de langue puis sur le réseau de neurones convolutif. Les résultats obtenus sur les collections internationales ImageCLEF 2017 et MS-COCO sont assez concluants.

Mots clés :apprentissage profond, réseau de neurones artificiels, annotation automatique d'images, génération de légendes, large classification

Abstract

In this report, we present methods for extracting information from medical images. The purpose of these methods is to associate automatically concepts from a vocabulary to a given image, but also to generate automatically a caption for this image. We focused on deep learning techniques and the ImageCLEF Caption 2017 dataset. Concept detection is a high dimension multi-label classification because (1) each image can be associated to various concepts or classes, (b) there is a wide variety of concepts in the database. In this work, we present two transfer learning methods based on convolutional neural networks : Visual Geometry Group 19 and Residual network 50 to extract concepts from each image. For the automatic image annotation part, the task is to generate a text that corresponds to the description ofthe image. We used the classical encoder-decoder architecture proposed by Vinalys for the generation of legends. The model is driven by an end-to-end learning where the error is backpropagated on the language model and then on the image encoder. The result is quite conclusive on ImageCLEF 2017 and MS-COCO datasets

Keywords : deep learning, artificial neural networks, automatic image annotation, image captioning, large classification

Titre :Détection de concept et annotation automatique d'images médicales par apprentissage profond

Auteur : Mamitiana Ignace RANDRIANARIVONY
Tel : +261 34 86 522 17
Email : mamitianaignace@gmail.com
Encadreur : Madame Josiane MOTHE