



**HAL**  
open science

# Localisation pour la stimulation magnétique transcrânienne : comparaison d'un réseau de neurones convolutifs hiérarchique avec un réseau de neurones convolutifs traditionnel

Enora Giffard

► **To cite this version:**

Enora Giffard. Localisation pour la stimulation magnétique transcrânienne : comparaison d'un réseau de neurones convolutifs hiérarchique avec un réseau de neurones convolutifs traditionnel. Sciences du Vivant [q-bio]. 2021. dumas-03594449

**HAL Id: dumas-03594449**

**<https://dumas.ccsd.cnrs.fr/dumas-03594449v1>**

Submitted on 2 Mar 2022

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial - NoDerivatives 4.0 International License

<p>Année universitaire : 2020-2021.</p> <p>Spécialité : Mathématiques Appliquées Statistiques.</p> <p>Spécialisation (et option éventuelle) : Science des données pour la biologie option statistiques bayésiennes.</p>	<p><b>Mémoire de fin d'études</b></p> <p><input type="checkbox"/> d'ingénieur d'AGROCAMPUS OUEST (École nationale supérieure des sciences agronomiques, agroalimentaires, horticoles et du paysage), école interne de L'institut Agro (Institut national d'enseignement supérieur pour l'agriculture, l'alimentation et l'environnement)</p> <p><input type="checkbox"/> de master d'AGROCAMPUS OUEST (École nationale supérieure des sciences agronomiques, agroalimentaires, horticoles et du paysage), école interne de L'institut Agro (Institut national d'enseignement supérieur pour l'agriculture, l'alimentation et l'environnement)</p> <p><input type="checkbox"/> de Montpellier SupAgro (étudiant arrivé en M2)</p> <p><input checked="" type="checkbox"/> d'un autre établissement (étudiant arrivé en M2)</p>
---	--

## **Localisation pour la stimulation magnétique transcrânienne - Comparaison d'un réseau de neurones convolutifs hiérarchique avec un réseau de neurones convolutifs traditionnel**

Par : Enora GIFFARD

*Soutenu à Rennes le 08.09.2021.*

**Devant le jury composé de :**

Président :

Maître de stage : John S.H. Baxter.

Enseignant référent : David Causeur.

Sébastien Lê.

*Les analyses et les conclusions de ce travail d'étudiant n'engagent que la responsabilité de son auteur et non celle d'AGROCAMPUS OUEST*



# Table des matières

<b>Introduction</b>	<b>page 1</b>
<b>La localisation pour la stimulation magnétique transcrânienne répétitive</b>	<b>pages 2 à 3</b>
<b>Les réseaux de neurones artificiels</b>	<b>pages 3 à 7</b>
<b>Les deux architectures comparées</b>	<b>pages 7 à 10</b>
Traditionnelle	pages 7 à 8
Hiérarchique	pages 8 à 10
<b>Comparaison</b>	<b>pages 10 à 19</b>
Consommation de mémoire vive	pages 10 à 14
Précision	pages 15 à 19
<b>Discussion, travaux futurs et conclusion</b>	<b>pages 19 à 20</b>
<b>Bibliographie</b>	

# Localisation pour la stimulation magnétique transcrânienne - Comparaison d'un réseau de neurones convolutifs hiérarchique avec un réseau de neurones convolutifs traditionnel

## Introduction

Penser, ressentir, se mouvoir... Plusieurs dizaines de milliers de neurones établissent plusieurs centaines de millions de connexions entre eux pour permettre à notre cerveau de mener à bien ces fonctions. Dans cette machinerie complexe, il arrive parfois que des neurones soient excités de façon anarchique ou encore inhibés au point qu'ils ne remplissent plus leurs fonctions. Que l'influx nerveux circule trop, pas assez ou mal, cela peut parfois provoquer des désordres fonctionnels qui atteignent la qualité de vie d'une personne, qui devient alors un patient. Des exemples bien connus de tels désordres sont la dépression, les hallucinations, l'épilepsie, certaines douleurs chroniques d'origine neurologique ou la maladie de Parkinson.

Les thérapies les plus couramment utilisées en neurologie ainsi qu'en psychiatrie impliquent des molécules modulant l'action de neuromédiateurs qui vont, directement ou indirectement, exciter ou inhiber certains neurones. Certains patients répondent bien à ces traitements quand d'autres sont insatisfaits. L'observance en psychiatrie, notamment, est extrêmement faible, en grande partie à cause de nombreux effets secondaires (Palazzolo, 2009). L'accoutumance, quant à elle, est un inconvénient majeur du traitement médicamenteux de la douleur. Il arrive également que certains patients répondent peu à certains traitements ou présentent des contre-indications qui en empêchent l'usage. Il est donc important que nous puissions proposer d'autres thérapies, non-médicamenteuses, à ces patients et ainsi augmenter l'offre de soins. La stimulation magnétique transcrânienne répétitive, dont nous décrivons le fonctionnement, s'inscrit dans ce contexte. Une source de variabilité dans l'efficacité de cette méthode semble être la précision avec laquelle les zones corticales ciblées sont localisées (Kim et al., 2014). Plusieurs méthodes sont utilisées pour cela en pratique clinique et seront abordées dans le chapitre suivant. Les réseaux de neurones, en particulier convolutifs, très utilisés en traitement de l'image, sont prometteurs et leur fonctionnement fera l'objet d'un chapitre dédié. Les stratégies de localisation de cibles utilisées par le cerveau humain et étudiées en neuropsychologie (Foulsham et al., 2013) ont inspiré la création d'une architecture de réseaux de neurones convolutifs hiérarchique (Baxter et al., 2021). L'objet de ce mémoire est de comparer les performances de cette architecture avec celles d'une architecture de réseau de neurones convolutifs traditionnelle. Après description de cette expérience et de ses résultats, nous discuterons plus avant leurs limites.

# La localisation pour la stimulation magnétique transcrânienne répétitive

La stimulation magnétique transcrânienne consiste en l'application d'une bobine à la surface du crâne qui génère un champ magnétique puis par induction et d'après la loi de Lenz-Faraday une dépolarisation des centres neuronaux à l'intérieur du cerveau. La répétition de cette stimulation à des fréquences inférieures à 1 Hz semble avoir un effet inhibiteur tandis qu'à des fréquences supérieures à 5 Hz un effet excitateur peut être obtenu (Fitzgerald et al., 2006). La bobine peut être appliquée de manière à stimuler certaines zones d'intérêt (Tab. 1.), selon que l'on cherche à traiter la dépression, la schizophrénie, le syndrome de stress post-traumatique, les troubles obsessionnels compulsifs, la maladie de Parkinson, les membres fantômes ou les névralgies (Hodaj et al., 2016).

Régions fonctionnelles	Indications
Gyrus temporal transverse (gyrus de Heschl) gauche	Troubles psychiatriques
Cortex orbitofrontaux gauche et droit	
Cortex préfrontaux dorsolatéraux gauche et droit	
Régions faciales des cortex moteurs primaires gauche et droit	Douleurs chroniques des zones correspondantes controlatérales
Régions des membres inférieurs des cortex moteurs primaires gauche et droit	
Régions des membres supérieurs des cortex moteurs primaires gauche et droit	

Tab. 1. Cibles pour la stimulation magnétique transcrânienne répétitive (Baxter et al., 2021).

Plusieurs techniques sont utilisées afin de localiser approximativement ces régions fonctionnelles.

L'une d'entre elles est la distance au point moteur de la main. Le praticien procède par essais et erreurs jusqu'à trouver la position de la bobine qui déclenche une réponse motrice de la main controlatérale. Les localisations des régions fonctionnelles d'intérêt sont alors estimées relativement à ce point moteur de la main. Par exemple, George et al. (2000) placent une bobine sur le crâne de façon à obtenir la plus forte réponse du muscle court abducteur du pouce droit puis la déplacent de 5 cm en direction rostrale sur un plan parasagittal, c'est-à-dire vers l'avant, dans le but d'atteindre le cortex préfrontal dorsolatéral gauche, cible pour le traitement de la dépression par stimulation magnétique transcrânienne répétitive.

On peut également utiliser le système EEG 10-20, qui consiste en l'utilisation d'un bonnet perforé de points situés à des distances de 10% ou 20% de la distance entre l'inion et le nasion, normalement utilisé pour les électro-encéphalogrammes (EEG). Ainsi, le point F3 du bonnet semble par exemple correspondre au cortex préfrontal dorsolatéral gauche. La portée est en général de 20 mm ou moins et dans environ 10% des cas une aire différente de l'aire ciblée est atteinte. (Herwig et al., 2003).

Ces méthodes ne prennent pas en compte la taille, la forme du crâne ainsi que la configuration interne du cerveau et sont donc soumises à une importante variabilité inter-patient.

Le recalage d'images se base sur un atlas anatomique standard, dit fixe, et une image issue d'IRM volumétrique du patient, dite flottante. Un vecteur de déformation associant un voxel de l'image flottante à un voxel de l'image fixe est élaboré par optimisation d'une fonction de coût (Rusjan et al., 2010). Cette méthode n'est pas limitée par la taille ou la forme du crâne. Néanmoins il est très possible que l'atlas ne soit pas représentatif des anatomies individuelles des patients et la variabilité inter-patient reste importante. De plus, malgré l'optimisation, un degré d'erreur subsiste. Un autre inconvénient majeur est la durée relativement longue du processus, de l'ordre de dix minutes (Baxter et al., 2021).

L'utilisation de réseaux de neurones artificiels convolutifs, particulièrement bien adaptés au traitement d'images, est prometteuse pour la localisation de régions fonctionnelles à partir d'IRM volumétrique. Ces modèles sont inspirés du fonctionnement du cerveau et il est donc intéressant de continuer à s'inspirer de la recherche en psychologie et en neurosciences afin d'en développer de nouvelles versions.

## La recherche visuelle active en psychologie

La localisation de cibles est une tâche effectuée couramment par notre cerveau. Les processus en action mais aussi les stratégies mises en œuvre lors de la recherche visuelle active d'un objet sont étudiés extensivement en sciences cognitives. L'idée ici est pour notre réseau de neurones artificiels d'imiter les stratégies du cerveau humain. Nous en introduirons deux dans ce chapitre, aussi opposées que complémentaires, la méthode *bottom-up* et la méthode *top-down*.

La stratégie la plus évidente et la plus simple est la méthode *bottom-up*, en anglais du bas vers le haut. Le sujet ne dispose pas de connaissances préalables sur l'objet ou sa localisation, autres qu'une description simple ou une représentation lui permettant de le distinguer des distracteurs et qui est inhérente à la tâche. La cible comme les distracteurs sont observés en parallèle. Les objets sortant du lot, dits saillants, sont trouvés plus rapidement. De même, si le nombre de traits pouvant différencier la cible d'un distracteur est faible, de l'ordre d'un ou deux, la méthode *bottom-up* est couronnée de succès. En effet, chaque objet ne nécessite pas une attention individuelle et un traitement en parallèle peut suffire.

Lorsque le nombre de traits à observer pour identifier la cible est trop élevé ou que cette dernière est peu saillante, la méthode *bottom-up* est peu efficace et la méthode *top-down* est privilégiée. Cette méthode consiste en l'utilisation de connaissances préalables sur l'objet ou sa localisation ou en l'élaboration d'une stratégie d'allocation de l'attention. Le sujet peut émettre l'hypothèse que la cible se trouve en haut à gauche de l'image et concentrer son attention en fonction. S'il sait que l'objet est rouge, il peut également, après un premier traitement en parallèle pour identifier tous les objets rouges, choisir de les observer plus attentivement un à un.

Ainsi, la méthode *bottom-up* part du bas, au niveau du stimulus, et alloue l'attention en parallèle pour parvenir à localiser la cible. La méthode *top-down*, quant à elle, part du haut, au niveau cognitif, et alloue l'attention en série à chaque objet ou à chaque portion d'image.

On a longtemps cru que la méthode *bottom-up* était priorisée dans la recherche active, comme le montraient les différentes expériences réalisées en laboratoire. Mais ces résultats sont-ils généralisables? Une expérience classique consiste en une cible disposée parmi des distracteurs dont elle peut être différenciée par un ou deux traits sur un écran d'ordinateur, entièrement dans le champ de vision du sujet. Comme nous l'avons vu, dans ce genre de situations, la méthode *bottom-up* suffit à localiser efficacement la cible. Cependant, dans la vie quotidienne, nous évoluons dans et interagissons avec, un monde tridimensionnel et l'objet que nous cherchons n'est pas nécessairement dans notre champ de vision. Nous devons bouger la tête voire la totalité du corps, soulever d'autres objets, ouvrir des portes. Il semble évident dans ce cas que la méthode *bottom-up* ne peut pas être suffisante. De plus, nous disposons le plus souvent de connaissances préalables sur la localisation de l'objet que nous cherchons, par exemple du dernier endroit où l'on se souvient avoir vu ses clés, du réceptacle où on les range habituellement ou du trajet effectué. Dans ces situations de la vie courante, la méthode *top-down* tend à être privilégiée, bien que complémentée par la méthode *bottom-up* (Foulsham et al., 2013).

Quel est le lien entre l'étude de la recherche visuelle active humaine et notre problématique? Nous verrons par la suite que les deux stratégies discutées dans ce chapitre correspondent en fait aux stratégies utilisées par les deux réseaux de neurones que nous comparerons.

## Les réseaux de neurones artificiels

Un réseau de neurones artificiels consiste en un assemblage de couches, chacune contenant un nombre prédéterminé de neurones. Les neurones de la première correspondent aux variables prédictives et ceux de la dernière aux variables prédites. Les couches intermédiaires prennent en entrée la sortie de la couche précédente. Chacun de leurs neurones applique une opération sur les valeurs des neurones de la couche précédente, puis une fonction d'activation est appliquée sur le résultat.

La fonction d'activation la plus utilisée est la *Rectified Linear Unit* ou *ReLU* qui associe à  $x$  le maximum de  $x$  et de 0, ce qui a pour effet de maintenir les valeurs positives tout en annulant les valeurs négatives. Sa dérivée est extrêmement simple à calculer et vaut 0 ou 1, sauf en 0 où elle n'est pas dérivable. Il existe d'autres fonctions d'activation comme la fonction sigmoïde et la tangente hyperbolique, avec des dérivées un peu plus complexes, ce qui a pour effet d'ajouter du temps de calcul à un algorithme déjà souvent très coûteux en temps comme en mémoire comme nous allons le voir.

Chaque individu d'un *batch*, c'est-à-dire un sous-échantillon dont la taille est un hyperparamètre du réseau, traverse l'ensemble des couches une première fois puis le résultat est mesuré grâce à une fonction de coût qui dépend du problème traité. Il peut s'agir par exemple de l'erreur quadratique moyenne, ou fonction de coût L2, souvent utilisée pour les tâches de localisation.

La rétropropagation permet ensuite de calculer le gradient. Il s'agit d'un vecteur contenant les dérivées partielles de la fonction de coût par rapport à chacun des paramètres. Par exemple, la dérivée partielle de la fonction de coût  $C$  par rapport à un paramètre  $w$  de la dernière couche est

$$\frac{\partial C}{\partial w} = \frac{\partial C}{\partial a} \frac{\partial a}{\partial z} \frac{\partial z}{\partial w}$$

Avec  $a$  l'activation du neurone de la dernière couche dont le paramètre permet le calcul, et  $z$  tel que  $a = \sigma(z)$  avec  $\sigma$  la fonction d'activation.

Le gradient est noté  $\nabla_{\theta}C(\theta)$ , avec  $\theta$  les paramètres du modèle.

Les paramètres peuvent ensuite être mis à jour grâce à la descente de gradient. Cette dernière applique l'opération suivante (Hansen, 2019).

$$\theta_{t+1} = \theta_t - \eta \nabla_{\theta}C(\theta)$$

Les paramètres mis à jour lors de l'itération  $t + 1$  sont égaux aux anciens paramètres mis à jour lors de l'itération  $t$  auxquels on soustrait le gradient pondéré par un taux d'apprentissage  $\eta$ . Si la dérivée partielle de la fonction de coût par rapport au paramètre est négative, alors augmenter la valeur du paramètre augmente les performances, on soustrait donc le produit de cette dérivée partielle négative et du taux d'apprentissage au paramètre, ce qui a pour effet d'augmenter la valeur de ce dernier. Si la dérivée partielle est positive, alors augmenter la valeur du paramètre réduit les performances du réseau, et soustraire le produit de cette dérivée partielle positive et du taux d'apprentissage a pour effet de réduire la valeur du paramètre.

Il existe différents optimiseurs de descente de gradient, qui utilisent des versions améliorées de cette formule. Nous utilisons l'optimiseur *Adam*, pour *Adaptive Moment Estimation* (Kingma et al., 2015).

L'idée de l'optimiseur *Adam* est d'adapter le taux d'apprentissage aux gradients précédents pour chaque paramètre. Dans les faits, il permet d'avoir un taux d'apprentissage important au début pour ne pas rester coincé autour d'un minimum local dès le début et augmenter la vitesse. Le taux d'apprentissage a tendance à décroître à mesure qu'on s'approche d'un minimum local pour permettre des pas plus précis mais peut augmenter de nouveau en s'adaptant aux gradients. Cet optimiseur utilise deux taux de décroissance  $\beta_1$  et  $\beta_2$  dont les valeurs changent à chaque itération et qui sont généralement initialisés à 0,9 et 0,999 (Fig. 1.).  $m$  et  $v$  sont initialisées à 0. A chaque itération elles sont divisées respectivement par  $(1 - (\beta_1)^t)$  et  $(1 - (\beta_2)^t)$ , ce qui revient à les multiplier par un facteur 10 ou 1000 à la première itération, qui décroît à chaque itération et a pour limite 1, ce qui permet d'augmenter leurs valeurs rapidement au début malgré l'initialisation à 0.



**Algorithm 1:** *Adam*, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation.  $g_t^2$  indicates the elementwise square  $g_t \odot g_t$ . Good default settings for the tested machine learning problems are  $\alpha = 0.001$ ,  $\beta_1 = 0.9$ ,  $\beta_2 = 0.999$  and  $\epsilon = 10^{-8}$ . All operations on vectors are element-wise. With  $\beta_1^t$  and  $\beta_2^t$  we denote  $\beta_1$  and  $\beta_2$  to the power  $t$ .

---

**Require:**  $\alpha$ : Stepsize  
**Require:**  $\beta_1, \beta_2 \in [0, 1)$ : Exponential decay rates for the moment estimates  
**Require:**  $f(\theta)$ : Stochastic objective function with parameters  $\theta$   
**Require:**  $\theta_0$ : Initial parameter vector  
 $m_0 \leftarrow 0$  (Initialize 1<sup>st</sup> moment vector)  
 $v_0 \leftarrow 0$  (Initialize 2<sup>nd</sup> moment vector)  
 $t \leftarrow 0$  (Initialize timestep)  
**while**  $\theta_t$  not converged **do**  
     $t \leftarrow t + 1$   
     $g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$  (Get gradients w.r.t. stochastic objective at timestep  $t$ )  
     $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$  (Update biased first moment estimate)  
     $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$  (Update biased second raw moment estimate)  
     $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$  (Compute bias-corrected first moment estimate)  
     $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$  (Compute bias-corrected second raw moment estimate)  
     $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$  (Update parameters)  
**end while**  
**return**  $\theta_t$  (Resulting parameters)

---

Fig. 1. Pseudo-code de l'optimiseur de descente de gradient développé par Kingma et al. (2015).

La rétropropagation et la descente de gradient ont lieu après qu'un *batch* a traversé le réseau. Lorsque tous les individus du jeu de données d'apprentissage répartis en plusieurs *batches* ont donné lieu à une descente de gradient, on parle d'une *epoch*. Le nombre d'*epochs* est aussi un hyperparamètre. Dans notre cas, nous l'optimisons grâce à un autre hyperparamètre, appelé patience. A chaque *epoch*, l'erreur sur l'échantillon de validation est calculée. Si, au bout du nombre d'*epochs* défini par la patience, l'erreur n'a pas diminué, alors le programme s'arrête et les paramètres de la meilleure *epoch* sont sauvegardés.

Il existe plusieurs types de neurones, qui composent plusieurs types de couches qui elles-mêmes forment plusieurs types de réseaux. Les neurones d'une couche linéaire entrent dans leur fonction d'activation une combinaison linéaire, ou une somme pondérée, des neurones de la couche précédente à laquelle on ajoute un biais. Les coefficients et les biais sont les paramètres appris par le réseau. Le nombre de ces paramètres est pour chaque couche linéaire le produit des nombres de neurones de cette couche et de la couche précédente.

Les neurones d'une couche convolutive appliquent une opération de convolution. Chaque neurone a un *kernel* qui agit comme un filtre. Il est composé de coefficients permettant, comme dans le cas des couches linéaires, de réaliser une combinaison linéaire des variables d'entrée. La différence est que le kernel, possédant généralement moins de coefficients que le nombre de variables qu'il multiplie, ne les prend pas toutes en compte en une fois et navigue le long du signal. Notons d'ailleurs que le cas particulier d'un kernel de même taille que l'entrée de la couche opèrerait une combinaison linéaire. L'opération de convolution est notée  $*$ .

Dans le cas du traitement d'un signal continu en une dimension, la *feature map*, c'est-à-dire le produit de convolution peut être définie comme ceci.

$$s(t) = (x * k)(t) = \int x(t - a)k(a)da$$

Avec  $s$  la *feature map*,  $t$  une coordonnée, par exemple en secondes dans le cas d'un signal dont l'unique dimension est le temps,  $x$  le signal en entrée de la couche et  $k$  le *kernel*.

Or, les signaux et images numérisés sont discrets. Ils ne sont donc pas intégrables. Il est alors nécessaire de définir l'opération de convolution autrement.

On obtient ainsi pour une dimension la définition suivante.

$$s(t) = \sum_a x(t - a)k(a)$$

Voici la généralisation de cette définition pour le traitement d'images en trois dimensions.

$$S(i, j, k) = (I * K)(i, j, k) = \sum_l \sum_m \sum_n I(i - l, j - m, k - n)K(l, m, n)$$

La corrélation croisée dont la définition est ci-dessous peut également être utilisée comme alternative à la convolution par certains outils (Goodfellow et al., 2016). C'est le cas du module torch que nous utiliserons.

$$S(i, j, k) = \sum_l \sum_m \sum_n I(i + l, j + m, k + n)K(l, m, n)$$

## Les deux architectures comparées

### Traditionnelle

Afin d'évaluer l'apport du réseau de neurones convolutifs hiérarchique de Baxter et al. (2021) dont l'architecture est présentée plus bas (Fig. 3.), nous avons créé un réseau de neurones convolutifs traditionnel (Fig. 2.), d'architecture plus simple.

Ce dernier est composé de six couches convolutives suivies chacune d'une activation *ReLU*. Le nombre de *kernels* de la première est de 32 puis augmente deux à deux jusqu'à 42. Après chaque couche convolutive à l'exception de la dernière, une opération de *pooling* par le maximum de taille de *kernel* 2 et de pas 2 est effectuée. Une couche linéaire à 512 neurones suivie d'une activation *ReLU* suit. L'optimiseur choisi est Adam et le taux d'apprentissage est de 0.001.

La résolution des images, et leur taille par conséquent, diminue rapidement à chaque couche de *pooling*, qui remplace un cube de huit voxels par leur maximum. Le réseau de neurones convolutifs a ainsi accès d'abord aux détails en haute résolution répartis sur toute l'image, puis aux éléments plus grossiers, toujours sur l'image entière, en plus faible résolution. Ce mode de fonctionnement le rapproche de la méthode de recherche visuelle active *bottom-up*.

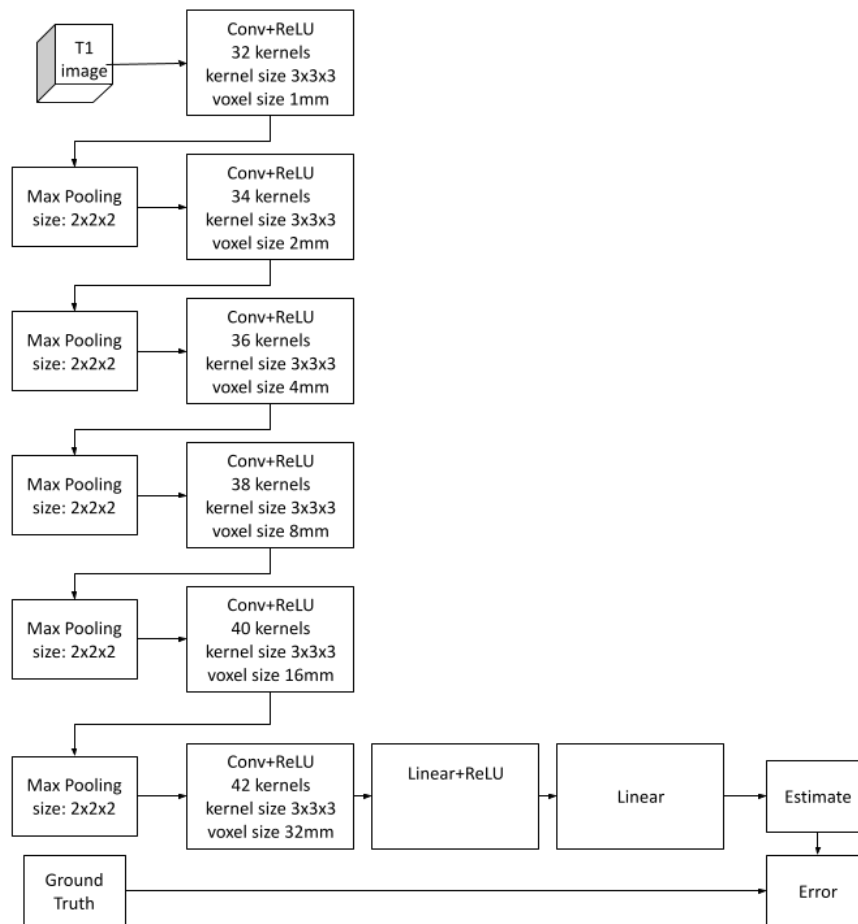


Fig. 2. Architecture de réseaux de neurones convolutifs traditionnel *bottom-up* créée pour cette étude.

## Hierarchique

Le réseau de neurones convolutifs multirésolution (Fig. 3.), développé par Baxter et al. en 2021, propose quant à lui une architecture divisée en six sous-réseaux d'architectures identiques. La résolution de l'image ainsi que sa taille sont diminuées cinq fois grâce à une opération de *pooling* par la moyenne, ce qui permet d'obtenir six images de résolutions différentes en remplaçant un cube de huit voxels par leur moyenne. L'image ayant la plus faible résolution est entrée dans un des sous-réseaux. Le résultat, appelé centroïde, permet de définir une zone de forme cubique qui représente un huitième de l'image et dans laquelle il est probable que la cible se trouve. L'image de résolution supérieure, initialement huit fois plus grande que la précédente, est coupée pour ne conserver que la zone cible. L'image résultante, de résolution supérieure et de même taille que celle entrée dans le sous-réseau précédent, est entrée dans le sous-réseau suivant. Cet enchaînement d'opérations est effectué cinq fois jusqu'à arriver à la résolution la plus fine et un centroïde de zone cible très précis, qui sera l'estimation finale de la localisation. Les estimations intermédiaires seront cependant conservées dans un vecteur, ce qui permettra de calculer la fonction de coût. En effet, la coupe

n'étant pas dérivable, l'erreur d'apprentissage ne doit pas être calculée à partir de la sortie de la dernière couche seule pour que la descente de gradient puisse s'effectuer. Les erreurs quadratiques moyennes entre les centroïdes et la localisation indiquée dans le jeu de données sont calculées pour chaque résolution.

Ce réseau a d'abord accès à des éléments de contexte en résolution grossière avant de porter son attention sur une zone plus réduite et des détails en résolution plus fine. Ce mode de fonctionnement le rapproche de la méthode *top-down* de recherche visuelle active.

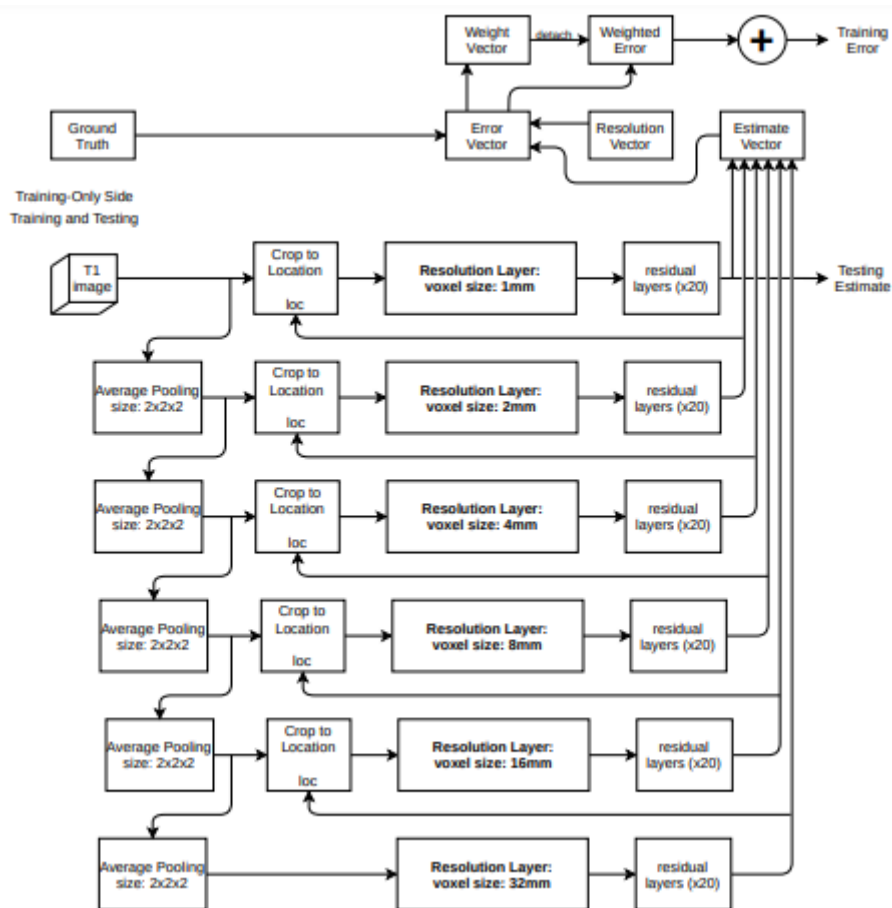


Fig. 3. Architecture de réseaux de neurones convolutifs hiérarchique *top-down* objet de cette étude. Réseau et schéma réalisés par Baxter et al. (2021).

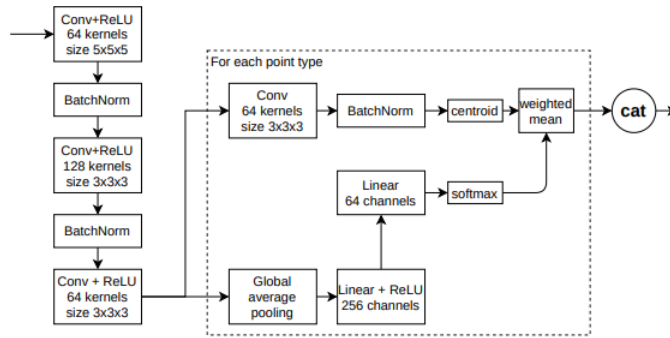


Fig. 4. Architecture d'une couche de résolution de l'architecture de réseaux de neurones convolutifs hiérarchique *top-down*. Réseau et schéma réalisés par Baxter et al. (2021).



Fig. 5. Architecture d'une couche résiduelle de l'architecture de réseaux de neurones convolutifs hiérarchique *top-down*. Réseau et schéma réalisés par Baxter et al. (2021).

## Comparaison

### Consommation de mémoire vive

La consommation de mémoire d'un réseau de neurones dépend du nombre de paramètres et de la taille et du nombre des données, *feature maps* ou images, s'il s'agit d'images.

Intéressons-nous d'abord au nombre de paramètres.

Le nombre de paramètres d'une couche convolutive est de  $n_i(n_{i-1}k^d + 1)$ , avec  $d$  le nombre de dimensions de convolution,  $k$  la taille du *kernel*,  $n_i$  le nombre de *kernels* de la couche à laquelle on s'intéresse et  $n_{i-1}$  le nombre de *kernels* de la couche précédente. En effet, le nombre de paramètres d'un *kernel* est  $k^d$ . On le multiplie par le nombre de *kernels* de la couche précédente et on ajoute le biais. On obtient le nombre de paramètres pour un *kernel* de la couche à laquelle on s'intéresse. Il faut ensuite le multiplier par le nombre de *kernels*.

Le nombre de paramètres d'une couche linéaire est  $n_l(n_{l-1} + 1)$ , avec  $n_{l-1}$  la taille de la sortie de la couche précédente et  $n_l$  le nombre de neurones de la couche à laquelle on s'intéresse.

Une couche de normalisation *Batch Norm* utilise deux paramètres appris ainsi qu'une moyenne mobile et une variance mobile, soit 4 paramètres. Une couche de *pooling* n'a pas besoin de stocker des paramètres puisque ses paramètres sont fixes.

Intéressons-nous maintenant à la mémoire vive nécessaire au stockage des images.

Pour chaque couche de convolution, elle est de  $sn_l$ , avec  $s$  la mémoire utilisée par une image ou *feature map* de la couche précédente et  $n_l$  le nombre de *kernels* de la couche de convolution.

Chaque couche de *pooling* avec une taille de *kernel* 2 et un pas de 2 divise la consommation de mémoire par  $2^d$ , avec  $d$  le nombre de dimensions de *pooling*. Les couches de normalisation *Batch Norm* ne modifient pas la taille des données.

Pour les couches linéaires, c'est la taille des activations, égale au nombre de neurones, qui compte.

Les paramètres sont des nombres à virgule flottante et sont traditionnellement codés sur 4 octets. Nos images sont à l'origine cubiques de 256 voxels de côté et nous effectuons notre apprentissage sur 13 cibles. Chaque voxel est également codé sur 4 octets.

Notre réseau *bottom-up* a  $187\,358 + 256 \times 256 \times 256 / 2^5 \times 21 + 1\,539 \times 13 = 11\,217\,413$  paramètres au total (Tab. 2.), qui représentent donc environ 45 Mo. La totalité de ces paramètres doit être stockée dans la mémoire vive, puisque ce sont eux qui constituent les connaissances du réseau et devront être mis à jour.

En ce qui concerne la consommation de mémoire par les images et *feature maps*, ce qui compte n'est pas tant la consommation totale ni les sorties des couches prises séparément que les éventuels goulots d'étranglement. Une couche utilise son entrée et sa sortie simultanément et c'est donc la somme de l'entrée et de la sortie d'une couche qui risque de consommer une quantité de mémoire vive trop importante pour notre système, et créer un goulot d'étranglement.

Nous pouvons voir ici que c'est au niveau de la première couche de *pooling* que la consommation de mémoire dédiée aux *feature maps* est la plus importante. Cette couche a une entrée de 32 fois la taille des images originales et une sortie de 4 fois cette taille. Elle a donc besoin de 36 fois plus de mémoire qu'une image originale. Le nombre de paramètres des différentes couches ainsi que la taille de *batch* viennent bien sûr s'ajouter à l'équation. Une de nos images de 256x256x256 voxels utilise environ 67 Mo. Notre goulot d'étranglement mobilise donc environ 2,4 Go de mémoire vive. Chaque *kernel* de plus dans la première couche convolutive nécessiterait environ 75 Mo de mémoire vive supplémentaire.

Nous voyons bien ici que le problème est ce goulot d'étranglement, qui nécessite à lui seul une quantité importante de mémoire vive pour le stockage des *feature maps*, et non les paramètres qui représentent une quantité de mémoire négligeable en comparaison.

Architecture <i>bottom-up</i>		
Couche	Paramètres	Images et <i>feature maps</i>
Entrée	0	$s$
Convulsive 32 <i>kernels</i> (3x3x3)	$(1*3^3 + 1)*32 = 896$	$s*32$
<i>Pooling</i> (2x2x2)	0	$s/8*32 = s*4$
Convulsive 34 <i>kernels</i> (3x3x3)	$(32*3^3 + 1)*34 = 29\ 410$	$s/8*34 = s*4,25$
<i>Pooling</i> (2x2x2)	0	$s/8^2*34 \simeq s*0,531$
Convulsive 36 <i>kernels</i> (3x3x3)	$(34*3^3 + 1)*36 = 33\ 084$	$s/8^2*36 \simeq s*0,563$
<i>Pooling</i> (2x2x2)	0	$s/8^3*36 \simeq s*0,070$
Convulsive 38 <i>kernels</i> (3x3x3)	$(36*3^3 + 1)*38 = 36\ 974$	$s/8^3*38 \simeq s*0,074$
<i>Pooling</i> (2x2x2)	0	$s/8^4*38 \simeq s*0,009$
Convulsive 40 <i>kernels</i> (3x3x3)	$(38*3^3 + 1)*40 = 41\ 080$	$s/8^4*40 \simeq s*0,010$
<i>Pooling</i> (2x2x2)	0	$s/8^5*40 \simeq s*0,001$
Convulsive 42 <i>kernels</i> (3x3x3)	$(40*3^3 + 1)*42 = 45\ 402$	$s/8^5*42 \simeq s*0,001$
Linéaire 512 neurones	$(s/8^5*42 + 1)*512$	512
Sortie	$(512 + 1)*c*3$	$c*3$
<b>Nombre de paramètres total/ goulot d'étranglement</b>	$187358 + s/2^5*21 + 1539*c$	$s*36$

Tab. 2. Nombre de paramètres et consommation de mémoire des images et *feature maps* utilisés par notre architecture *bottom-up*. Avec  $s$  la taille originale d'une image et  $c$  le nombre de cibles.

Notre réseau *top-down* a un total de  $3\ 523\ 272 + (1\ 033\ 704 + 3 \times 256 \times 256 \times 256/8) \times 13 = 106\ 717\ 872$  paramètres (Tab. 3.), ce qui représente environ 427 Mo.

Le goulot d'étranglement de notre réseau *top-down* se situe entre les deux premières couches de convolution de chaque couche de résolution (Fig. 4.). Chaque couche de résolution prend en entrée une image de 8x8x8 voxels, que ce soit l'image entière en résolution d'un voxel pour 32 mm ou une partie de l'image en résolution d'un voxel pour 1 mm. Cette image représente environ 2,048 ko. Si on additionne la sortie première couche de convolution de 64 *kernels* qui nécessite environ 131 ko pour son stockage et celle de la deuxième couche de convolution de 128 *kernels* qui nécessite environ 262 ko, on arrive à un total d'environ 393 ko, ce qui est bien plus faible. Si on considère que le réseau doit garder en mémoire les six résolutions de l'image entière, avant l'application des couches de résolution, on peut ajouter à cela  $4(256 \times 256 \times 256 + 128 \times 128 \times 128 + 64 \times 64 \times 64 + 32 \times 32 \times 32 + 16 \times 16 \times 16 + 8 \times 8 \times 8)$ , soit environ 77 Mo.

L'architecture *top-down* utilise 427 Mo pour le stockage de ses paramètres alors que l'architecture *bottom-up* n'en utilise que 45. En revanche, les goulots d'étranglement provoqués par le stockage des images et *feature maps* des réseaux *top-down* et *bottom-up* sont respectivement de 77 Mo et 2,4 Go. L'architecture *top-down* est donc largement plus efficace en termes de mémoire vive que l'architecture *bottom-up*, leurs consommations de mémoire au niveau de leurs goulots d'étranglement respectifs se situant sur des ordres de grandeur différents (environ 2,5 Go contre environ 504 Mo).

Architecture <i>top-down</i>			
Couche	Paramètres	Images et <i>feature maps</i>	Répétitions
Entrée	0	$s$	1
<i>Pooling</i> (2x2x2)	0	$s/8$	1
<i>Pooling</i> (2x2x2)	0	$s/8^2$	1
<i>Pooling</i> (2x2x2)	0	$s/8^3$	1
<i>Pooling</i> (2x2x2)	0	$s/8^4$	1
<i>Pooling</i> (2x2x2)	0	$s/8^5$	1
Sous-réseau	$450632 + (127364 + s/16)*c$	goulot d'étranglement : $s/8^5*256$	6
Couches résiduelles	$7779*c + 1296$	goulot d'étranglement : $1296 + c*3$	120
<b>Nombre de paramètres total/ goulot d'étranglement</b>	<b><math>2859312 + (1697664 + 3*s/8)*c</math></b>	<b><math>s + s/8 + s/8^2 + s/8^3 + s/8^4 + s/8^5</math></b>	

Tab. 3. Nombre de paramètres et consommation de mémoire des images et *feature maps* utilisés par notre architecture *top-down*. Avec  $s$  la taille originale d'une image et  $c$  le nombre de cibles. Détails des sous-réseaux (Tab. 4.) et couches résiduelles (Tab. 5.) ci-dessous.



Sous-réseau			
Couche	Paramètres	Images et <i>feature maps</i>	Répétitions
Entrée	0	$s/8^5$	1
Convulsive 64 <i>kernels</i> (5x5x5)	$(1*5^3 + 1)*64 = 8064$	$s/8^5*64$	1
<i>BatchNorm</i>	4	$s/8^5*64$	1
Convulsive 128 <i>kernels</i> (3x3x3)	$(64*3^3 + 1)*128 = 221312$	$s/8^5*128$	1
<i>BatchNorm</i>	4	$s/8^5*128$	1
Convulsive 64 <i>kernels</i> (3x3x3)	$(128*3^3 + 1)*64 = 221248$	$s/8^5*64$	1
<i>Pooling</i>	0	$s/8^6*64$	$c$
Linéaire 256 neurones	$(s/8^6*64 + 1)*256$	256	$c$
Linéaire 64 neurones	$(256 + 1)*64 = 16448$	64	$c$
Convulsive 64 <i>kernels</i> (3x3x3)	$(64*3^3 + 1)*64 = 110656$	$s/8^5*64$	$c$
<i>BatchNorm</i>	4	$s/8^5*64$	$c$
<b>Nombre de paramètres total/ goulot d'étranglement</b>	<b><math>450632 + (127364 + s/16)*c</math></b>	<b><math>s/8^5*256</math></b>	

Tab. 4. Nombre de paramètres et consommation de mémoire des images et *feature maps* utilisés par un sous-réseau (Fig. 4.), appelé couche de résolution, de notre architecture *top-down*. Avec  $s$  la taille originale d'une image et  $c$  le nombre de cibles.

Couches résiduelles		
Couche	Paramètres	Images et <i>feature maps</i>
Entrée	0	$c*3$
Linéaire 1296 neurones	$(c*3 + 1)*1296$	1296
Sortie	$(1296 + 1)*c*3$	$c*3$
<b>Nombre de paramètres total/ goulot d'étranglement</b>	<b><math>7779*c + 1296</math></b>	<b><math>1296 + c*3</math></b>

Tab. 5. Nombre de paramètres et consommation de mémoire des images et *feature maps* utilisés par une couche résiduelle (Fig. 5.) de notre architecture *top-down*. Avec  $s$  la taille originale d'une image et  $c$  le nombre de cibles.

## Précision

Nous posons l'hypothèse que les erreurs d'estimation obtenues par nos deux architectures de réseaux de neurones convolutifs proviennent de distributions différentes.

Si cette première hypothèse est validée, nous posons une deuxième hypothèse selon laquelle les erreurs d'estimation obtenues par notre architecture *top-down* seraient globalement plus faibles que celles obtenues par notre architecture *bottom-up*.

Nous nous interrogerons également sur une différence de sensibilité de nos deux architectures de réseaux de neurones convolutifs à la taille de l'échantillon d'apprentissage.

Les données utilisées sont vingt-six images issues d'imagerie par résonance magnétique volumétrique, pondérées en T1 et dont un voxel est isotropique de taille 1mm. Chaque image correspond à un patient. Les images proviennent de différents hôpitaux qui utilisent différents équipements. Une normalisation Min-Max est effectuée, utilisant le cinquième percentile comme minimum et le quatre-vingt-quinzième percentile comme maximum. Le jeu de données comporte pour chaque image les coordonnées dans le repère défini par les axes antéro-postérieur, inférieur-supérieur et droite-gauche de treize cibles d'intérêt. Ces coordonnées sont obtenues grâce aux annotations d'un expert pour le gyrus temporal transverse gauche, les cortex orbitofrontaux gauche et droit et les cortex préfrontaux dorsolatéraux gauche et droit, soit les cibles du traitement des troubles psychiatriques en stimulation magnétique transcrânienne. Pour les régions faciales des cortex moteurs primaires gauche et droit, les régions des membres inférieurs des cortex moteurs primaires gauche et droit et les régions des membres supérieurs des cortex moteurs primaires gauche et droit, cibles du traitement des douleurs chroniques, ce sont trois experts qui ont fourni des annotations pour chacune des images. Afin de déterminer un point de consensus, la cohérence des annotations a été vérifiée visuellement. Ensuite, pour les annotations cohérentes, seules celles pour lesquelles il existait un accord entre au moins deux des experts ont été retenues. Le centre de gravité des annotations des experts en accord est le point de consensus utilisé par la suite (Baxter et al., 2021).

Il existe des données manquantes. Trois images n'ont pas de coordonnées dans le jeu de données pour le gyrus temporal transverse gauche et pour deux d'entre elles les cortex préfrontaux dorsolatéraux gauche et droit ne sont pas renseignés non plus.

Nous utilisons une évaluation croisée. Pour ce faire, vingt-six jeux de données divisés chacun en un échantillon d'apprentissage, un échantillon de validation et un échantillon de test sont créés en utilisant la méthode suivante. Selon la taille souhaitée pour l'échantillon de test, les individus sont tirés aléatoirement sans remise parmi les individus qui ont le moins été utilisés dans les échantillons de test des jeux de données précédents. Il en résulte que chaque individu est utilisé le même nombre de fois plus ou moins une dans un échantillon de test, cela permet d'avoir un nombre de prédictions par patient comparable. Quand un échantillon de test est constitué, quatre individus sont tirés aléatoirement sans remise parmi les individus restants et servent de base à l'échantillon de validation, les autres servent de base à l'échantillon d'apprentissage. Aux échantillons d'apprentissage et de validation sont ajoutées à des fins d'augmentation des données des images symétriques par rapport au plan sagittal de celles déjà présentes. Les coordonnées sont adaptées, le cortex orbitofrontal gauche étant le symétrique

par rapport au plan sagittal du cortex orbitofrontal droit de l'image d'origine, par exemple. Au cours de l'apprentissage, une augmentation des données sous forme de rotations et translations aléatoires est également effectuée.

De plus, afin d'évaluer les sensibilités des architectures de réseaux de neurones convolutifs aux tailles des échantillons d'apprentissage, cette évaluation croisée est faite trois fois avec chaque fois des tailles d'échantillon d'apprentissage et de test différentes. Dans un cas, la taille des échantillons d'apprentissage est de 16 individus avant augmentation des données et la taille des échantillons de test de 6. Dans un autre cas la taille des échantillons d'apprentissage est de 11 avant augmentation des données comme la taille des échantillons de test. Dans un autre cas encore la taille des échantillons d'apprentissage est de 6 avant augmentation des données et la taille des échantillons de test est de 16. Chacune des opérations suivantes sera effectuée séparément pour chacune des trois expériences.

Les mêmes jeux de données sont utilisés pour les apprentissages, les validations et les tests des deux architectures de réseaux de neurones.

Le cortex moteur primaire droit n'étant annoté que dans les échantillons d'apprentissage et de validation à des fins d'augmentation des données par symétrie selon l'axe sagittal, nous ne le considérerons pas par la suite et ne prendrons donc en compte que douze cibles.

Un vecteur d'erreur est calculé à partir des estimations de chacune des deux architectures sur les patients des échantillons de test, avec pour chaque estimation une erreur égale à la distance euclidienne entre la localisation de la cible renseignée dans le jeu de données et son estimation par le réseau de neurones. L'unité de cette distance est le millimètre.

$$\|c - e\|_2 = \sqrt{(c_x - e_x)^2 + (c_y - e_y)^2 + (c_z - e_z)^2}$$

Avec  $c$  le consensus pour la localisation de la cible et  $e$  l'estimation de la localisation de cette même cible.

Les moyennes de ces erreurs par patient sont ensuite calculées.

Pour comparer nos deux architectures, et afin de nous affranchir des hypothèses de normalité du test des échantillons appariés de Student, nous effectuons pour chaque cible un test des rangs signés de Wilcoxon, sur les différences entre les moyennes des erreurs par patient de l'architecture *top-down* et celles de l'architecture *bottom-up*. L'hypothèse nulle est que ces différences proviennent d'une population dont la médiane est 0. Ainsi, les différences entre les erreurs de chacune des architectures se compenseraient et aucune des architectures ne serait globalement supérieure à l'autre. Les p-values de ces tests sont présentées pour chaque cible dans les tableaux Tab. 6., Tab. 7. et Tab. 8. pour des tailles d'échantillon d'apprentissage respectivement de 16, 11 et 6. Puisque nous faisons 3 x 12 tests, nous appliquons la correction des comparaisons multiples de Bonferroni. Le seuil de significativité pour la p-value est alors de 5/36%, c'est-à-dire environ  $1,4 \cdot 10^{-3}$ .

Nous pouvons voir que l'hypothèse nulle est rejetée pour toutes les cibles sauf le cortex préfrontal dorsolatéral gauche et la région du membre supérieur gauche pour des tailles d'échantillons d'apprentissage de 16 et de 11 et le gyrus temporal transverse gauche pour des tailles d'échantillons d'apprentissage de 11 et de 6. Pour toutes les cibles pour lesquelles cette hypothèse nulle est rejetée, nous ne pouvons pas raisonnablement rejeter notre première hypothèse selon laquelle les erreurs d'estimation obtenues par nos deux architectures de

réseaux de neurones convolutifs proviennent de distributions différentes. Pour toutes les cibles, la moyenne des erreurs de l'architecture *top-down* est inférieure à celle de l'architecture *bottom-up*. Pour les cibles pour lesquelles nous ne rejetons pas notre première hypothèse, nous ne pouvons pas non plus raisonnablement rejeter notre deuxième hypothèse selon laquelle les erreurs d'estimation obtenues par notre architecture *top-down* seraient globalement plus faibles que celles obtenues par notre architecture *bottom-up*.

Apprentissage : 16, Test : 6					
Cible	P-value	Moyenne		Ecart-type	
		Top-down	Bottom-up	Top-down	Bottom-up
Gyrus temporal transverse gauche	<b>3,5*10<sup>-4</sup></b>	6,11	10,39	3,75	4,10
Cortex orbitofrontal gauche	<b>2,9*10<sup>-5</sup></b>	5,99	11,37	4,95	5,34
Cortex orbitofrontal droit	<b>2,1*10<sup>-5</sup></b>	6,54	11,36	4,95	4,91
Cortex préfrontal dorsolatéral gauche	2,3*10 <sup>-2</sup>	9,20	11,03	9,05	5,34
Cortex préfrontal dorsolatéral droit	<b>4,9*10<sup>-4</sup></b>	7,24	11,07	4,38	5,54
Cortex moteur primaire gauche	<b>8,2*10<sup>-5</sup></b>	7,78	13,15	3,62	6,15
Régions faciales du cortex moteur primaire gauche	<b>4,6*10<sup>-5</sup></b>	6,45	12,00	4,06	5,30
Régions faciales du cortex moteur primaire droit	<b>1,9*10<sup>-5</sup></b>	8,30	13,96	5,53	6,15
Régions du membre inférieur du cortex moteur primaire gauche	<b>5,1*10<sup>-5</sup></b>	8,77	14,72	5,63	6,40
Régions du membre inférieur du cortex moteur primaire droit	<b>1,6*10<sup>-4</sup></b>	10,02	15,22	7,71	7,71
Régions du membre supérieur du cortex moteur primaire gauche	2,8*10 <sup>-3</sup>	10,10	13,84	5,47	6,84
Régions du membre supérieur du cortex moteur primaire droit	<b>8,7*10<sup>-5</sup></b>	9,63	14,90	5,54	6,26

Tab. 6. Résultats des tests des rangs signés de Wilcoxon sur les différences entre les moyennes des erreurs en millimètres par patient pour l'architecture *top-down* et l'architecture *bottom-up*. En **gras**, les p-values inférieures au seuil de significativité de 5/36%. Validation croisée réalisée avec des échantillons d'apprentissage de 16 patients avant augmentation des données et des échantillons de test de 6 patients.

Apprentissage : 11, Test : 11					
Cible	P-value	Moyenne		Ecart-type	
		Top-down	Bottom-up	Top-down	Bottom-up
Gyrus temporal transverse gauche	1,9*10 <sup>-3</sup>	7,15	10,83	4,29	4,51
Cortex orbitofrontal gauche	<b>4,6*10<sup>-5</sup></b>	6,74	12,43	6,79	6,01
Cortex orbitofrontal droit	<b>4,6*10<sup>-5</sup></b>	7,04	12,32	6,66	5,63
Cortex préfrontal dorsolatéral gauche	3,1*10 <sup>-3</sup>	9,94	12,47	10,38	5,66
Cortex préfrontal dorsolatéral droit	<b>1,3*10<sup>-4</sup></b>	7,81	12,07	5,56	5,58
Cortex moteur primaire gauche	<b>6,3*10<sup>-5</sup></b>	7,58	13,54	3,94	6,72
Régions faciales du cortex moteur primaire gauche	<b>3,7*10<sup>-5</sup></b>	7,45	12,73	4,44	5,41
Régions faciales du cortex moteur primaire droit	<b>2,9*10<sup>-5</sup></b>	8,52	14,39	5,56	6,75
Régions du membre inférieur du cortex moteur primaire gauche	<b>8,7*10<sup>-5</sup></b>	9,07	14,89	5,90	7,17
Régions du membre inférieur du cortex moteur primaire droit	<b>4,1*10<sup>-5</sup></b>	10,44	15,79	7,85	7,85
Régions du membre supérieur du cortex moteur primaire gauche	1,4*10 <sup>-3</sup>	10,24	14,54	6,05	7,00
Régions du membre supérieur du cortex moteur primaire droit	<b>6,3*10<sup>-5</sup></b>	9,98	15,33	5,90	6,45

Tab. 7. Résultats des tests des rangs signés de Wilcoxon sur les différences entre les moyennes des erreurs en millimètres par patient pour l'architecture *top-down* et l'architecture *bottom-up*. En **gras**, les p-values inférieures au seuil de significativité de 5/36%. Validation croisée réalisée avec des échantillons d'apprentissage de 11 patients avant augmentation des données et des échantillons de test de 11 patients.

Apprentissage : 6, Test : 16					
Cible	P-value	Moyenne		Ecart-type	
		Top-down	Bottom-up	Top-down	Bottom-up
Gyrus temporal transverse gauche	3,9*10 <sup>-3</sup>	8,09	11,07	5,13	4,54
Cortex orbitofrontal gauche	<b>5,1*10<sup>-5</sup></b>	7,26	13,27	7,59	6,13
Cortex orbitofrontal droit	<b>1,8*10<sup>-4</sup></b>	8,04	13,27	8,24	5,85
Cortex préfrontal dorsolatéral gauche	<b>1,2*10<sup>-3</sup></b>	10,14	13,22	9,85	6,51
Cortex préfrontal dorsolatéral droit	<b>4,4*10<sup>-5</sup></b>	8,21	12,80	5,48	6,17
Cortex moteur primaire gauche	<b>3,6*10<sup>-5</sup></b>	8,28	14,26	4,11	6,77
Régions faciales du cortex moteur primaire gauche	<b>4,1*10<sup>-5</sup></b>	8,48	13,93	5,34	5,94
Régions faciales du cortex moteur primaire droit	<b>1,9*10<sup>-5</sup></b>	9,55	15,48	5,82	7,33
Régions du membre inférieur du cortex moteur primaire gauche	<b>1,2*10<sup>-4</sup></b>	10,28	16,24	6,27	7,33
Régions du membre inférieur du cortex moteur primaire droit	<b>3,3*10<sup>-5</sup></b>	11,26	16,90	7,70	8,07
Régions du membre supérieur du cortex moteur primaire gauche	<b>2,0*10<sup>-4</sup></b>	10,81	15,70	5,63	7,06
Régions du membre supérieur du cortex moteur primaire droit	<b>2,6*10<sup>-5</sup></b>	10,46	16,63	6,14	7,16

Tab. 8. Résultats des tests des rangs signés de Wilcoxon sur les différences entre les moyennes des erreurs en millimètres par patient pour l'architecture *top-down* et l'architecture *bottom-up*. En **gras**, les p-values inférieures au seuil de significativité de 5/36%. Validation croisée réalisée avec des échantillons d'apprentissage de 6 patients avant augmentation des données et des échantillons de test de 16 patients.

## Discussion, travaux futurs et conclusion

Une limite de cette étude est l'optimisation de ces deux architectures et de leurs hyperparamètres. Si nous tenté d'optimiser l'architectures *bottom-up* de la façon qui nous a paru la meilleure, de la même façon que Baxter et al. (2021) ont tenté d'optimiser l'architecture *top-down*, nous ne pouvons être certains qu'il n'était pas possible de parvenir à une meilleure optimisation. Un compromis a également dû être trouvé entre consommation de mémoire vive et performance, notamment pour le réseau *bottom-up*, en raison de sa consommation limitante techniquement.

Nos données sont également imparfaites. En effet, la localisation des cibles de la stimulation magnétique transcrânienne est une tâche difficile et les experts ne sont pas toujours en accord.

Les cibles utilisées en psychiatrie sont les plus difficiles à déterminer, car les effets ne sont pas immédiatement mesurables. La vérité terrain pour ces cibles est obtenue à partir des annotations d'un expert uniquement, contre trois pour les cibles utilisées en traitement de la douleur.

Une étude statistique de la sensibilité au nombre de patients dans les échantillons d'apprentissage exploitant les résultats précédemment exposés pourrait être menée.

Afin de parfaire la comparaison entre méthodes psychologiques et réseaux de neurones artificiels, nous pourrions utiliser des méthodes d'interprétabilité, qui nous permettraient d'identifier plus précisément où et comment les réseaux de neurones portent leur attention.

Dans le futur, nous pouvons imaginer l'intégration d'un réseau de neurones convolutifs hiérarchique *top-down* à un système de neuronavigation pour la stimulation magnétique transcrânienne répétitive. Cela permettrait un gain de temps pour les neurochirurgiens. Afin de s'assurer que ces derniers aient un contrôle sur les processus, nous pouvons imaginer un système avec lequel le neurochirurgien interagirait afin de déterminer la meilleure position de la bobine.

En conclusion, la stimulation magnétique transcrânienne répétitive cible des régions fonctionnelles particulières du cerveau, en fonction de son objectif. Ces zones sont difficiles à localiser précisément, car les anatomies cérébrales des patients diffèrent grandement. Les techniques utilisées jusqu'alors sont peu précises ou coûteuses en temps. L'idée d'utiliser des réseaux de neurones convolutifs se heurte à des limites techniques. En effet, la consommation de mémoire vive d'un réseau de neurones convolutifs en trois dimensions est très importante, en grande partie en raison de la taille des images et *feature-maps*. Pour contourner ce problème, l'idée a alors été de s'inspirer de la recherche en psychologie. Deux méthodes d'allocation de l'attention en recherche visuelle active se démarquent, la méthode *bottom-up* et la méthode *top-down*. La méthode *bottom-up* semble se rapprocher du fonctionnement des réseaux de neurones convolutifs traditionnels, qui identifient d'abord les détails en haute résolution puis des éléments plus grossiers et enfin le contexte en basse résolution. La méthode *top-down* a, quant à elle, inspiré la création d'une architecture de réseaux de neurones convolutifs hiérarchique qui identifie d'abord grâce au contexte en basse résolution une zone à cibler, puis augmente la résolution sur cette zone seulement, sans jamais changer la taille de l'image, et ceci plusieurs fois jusqu'à arriver à une zone très précise. L'intérêt est que l'image entière en haute résolution n'est pas utilisée, seulement un fragment de celle-ci, ce qui réduit grandement la consommation de mémoire. Afin de déterminer si ce type d'architecture représente une avancée par rapport aux réseaux de neurones convolutifs traditionnels, nous avons créé une architecture de réseaux de neurones convolutifs traditionnels, et nous avons comparé ses erreurs d'estimation et celles de l'architecture hiérarchique en utilisant une validation croisée et les mêmes données pour chacun. Il en résulte que pour neuf des douze cibles, l'architecture hiérarchique obtient des erreurs d'estimation significativement plus faibles que l'architecture traditionnelle. Pour trois des cibles, l'hypothèse selon laquelle les erreurs des deux réseaux se compensent ne peut pas toujours être rejetée. L'architecture hiérarchique consomme ainsi moins de mémoire vive et obtient des erreurs d'estimation plus faibles ou équivalentes à celles de l'architecture traditionnelle. Elle obtient également des erreurs d'estimation plus faibles ou équivalentes à celles des méthodes utilisées en stimulation magnétique transcrânienne répétitive (Baxter et al., 2021). Elle semble donc représenter une avancée par rapport aux techniques traditionnelles.

# Bibliographie

Baxter John S.H., Quoc Anh Bui, Maguet Ehouarn, Croci Stéphane, Delmas Antoine, Lefaucheur Jean-Pascal, Bredoux Luc, Jannin Pierre. Automatic Cortical Target Point Localisation in MRI for Transcranial Magnetic Stimulation via a Multi-Resolution Convolutional Neural Network. In : *International Journal of Computer Assisted Radiology and Surgery*. n°16. pp. 1077–1087 (2021).  
D.O.I. : [10.1007/s11548-021-02386-1](https://doi.org/10.1007/s11548-021-02386-1).

Fitzgerald Paul B., Fountain Sarah, Daskalakis Zafiris J.. A comprehensive review of the effects of rTMS on motor cortical excitability and inhibition. In : *Clinical Neurophysiology*. n°117. pp. 2584-2596 (2006).  
D.O.I. : [10.1016/j.clinph.2006.06.712](https://doi.org/10.1016/j.clinph.2006.06.712).

Foulsham Tom, Chapman Craig, Nasiopoulos Eleni, Kingstone Alan. (2013). Top-Down and Bottom-Up Aspects of Active Search in a Real-World Environment. In : *Canadian journal of experimental psychology*. n°68. (2013).  
D.O.I. : [10.1037/cep0000004](https://doi.org/10.1037/cep0000004).

George Mark S., Nahas Ziad, Molloy Monica, Speer Andrew M., Oliver Nicholas C., Li Xing-Bao, Arana George W., Risch S.Craig, Ballenger James C.. A controlled trial of daily left prefrontal cortex TMS for treating depression. *Biological Psychiatry*. n°48. pp. 962-970 (2000).  
D.O.I. : [10.1016/S0006-3223\(00\)01048-9](https://doi.org/10.1016/S0006-3223(00)01048-9).

Goodfellow Ian, Bengio Yoshua, Courville Aaron. Convolutional Networks. In : *Deep Learning*. M.I.T. Press (éd.). Cambridge, Massachusetts. pp. 326-366 (2016).  
Disponible à l'adresse : <https://www.deeplearningbook.org/contents/convnets.html>.

Hansen Casper. Neural Networks: Feedforward and Backpropagation Explained & Optimization. In : [mlfromscratch.com](https://mlfromscratch.com) (2019). [en ligne].  
Disponible à l'adresse : <https://mlfromscratch.com/neural-networks-explained/>. (Consulté le 16.06.2021).

Herwig Uwe, Satrapi Peyman, Schönfeldt-Lecuona Carlos. Using the International 10-20 EEG System for Positioning of Transcranial Magnetic Stimulation. In : *Brain Topography*. n°16. pp. 95–99 (2003).  
D.O.I. : [10.1023/B:BRAT.0000006333.93597.9d](https://doi.org/10.1023/B:BRAT.0000006333.93597.9d).




Hodaj Hasan, Alibeu Jean-Pierre, Maindet-Dominici Caroline, Dumolard Anne, Payen Jean-François. Pratique de la rTMS dans le traitement des syndromes douloureux chroniques : l'expérience grenobloise. In : *Clinical Neurophysiology*. n°46. pp. 86-87 (2016).  
D.O.I. : [10.1016/j.clinph.2006.06.712](https://doi.org/10.1016/j.clinph.2006.06.712).

Kim Woo Jin, Min Yu Sun , Yang Eun Joo, Paik Nam-Jong. Neuronavigated vs. conventional repetitive transcranial magnetic stimulation method for virtual lesioning on the Broca's area. In : *Neuromodulation*. n°17. pp. 16-21 (2014).  
D.O.I. : [10.1111/ner.12038](https://doi.org/10.1111/ner.12038).

Kingma Diederik P., Ba Jimmy Lei. Adam: A Method for Stochastic Optimization. 3rd. International Conference on Learning Representations. San Diego. (2015).  
Disponible à l'adresse : <https://arxiv.org/pdf/1412.6980v9.pdf>.

Palazzolo Jérôme. Medical compliance and relapses in schizophrenia: From classical neuroleptics to APAP. In : *Annales Médico-psychologiques, revue psychiatrique*. n°167. pp. 308-317 (2009).  
D.O.I. : [10.1016/j.amp.2009.03.009](https://doi.org/10.1016/j.amp.2009.03.009).

Rusjan Pablo M., Barr Mera S., Farzan Faranak, Arenovich Tamara, Maller Jerome J., Fitzgerald Paul B., Daskalakis Zafiris J. Optimal Transcranial Magnetic Stimulation Coil Placement for Targeting the Dorsolateral Prefrontal Cortex Using Novel Magnetic Resonance Image-Guided Neuronavigation. In : *Human brain mapping*. n°31. pp. 1643–1652 (2010).  
D.O.I. : [10.1002/hbm.20964](https://doi.org/10.1002/hbm.20964).

 	<p>Diplôme : Master.</p> <p>Spécialité : <b>Mathématiques Appliquées Statistiques.</b></p> <p>Spécialisation / option : Science des données pour la biologie option statistiques bayésiennes.</p> <p>Enseignant référent : David Causeur.</p>
<p>Auteur(s) : Enora Giffard.</p> <p>Date de naissance* : 29.05.1998.</p>	<p>Organisme d'accueil : MediCIS (U.M.R. 1099 L.T.S.I.)</p> <p>Adresse :</p> <p>2, avenue du Pr. Léon Bernard CS 34317 35043 RENNES Cedex</p> <p>Maître de stage : John S.H. Baxter (C.R.).</p>
<p>Nb pages : 20                      Annexe(s) : x</p>	
<p>Année de soutenance : 2021.</p>	
<p>Titre français : Localisation pour la stimulation magnétique transcrânienne - Comparaison d'un réseau de neurones convolutifs hiérarchique avec un réseau de neurones convolutifs traditionnel.</p> <p>Titre anglais : Localisation for transcranial magnetic stimulation – A comparison between a hierarchical convolutional neural network and a traditional convolutional neural network.</p>	
<p>Résumé :</p> <p>La stimulation magnétique transcrânienne répétitive est une thérapie ciblant des régions cérébrales fonctionnelles. Différentes méthodes existent pour localiser ces dernières mais sont peu précises. Les réseaux de neurones convolutifs se révèlent prometteurs mais leur consommation de mémoire vive importante lors du traitement d'images volumétriques restreint leur utilisation. Ce mémoire se propose de comparer l'architecture de réseaux de neurones convolutifs hiérarchique développée par Baxter et al. (2021) avec une architecture de réseaux de neurones convolutifs traditionnelle sur deux points : leur consommation de mémoire vive et leur précision. L'architecture hiérarchique se révèle plus efficace, avec une consommation de mémoire moindre et une précision meilleure que ou égale à l'alternative.</p>	
<p>Abstract :</p> <p>Repetitive transcranial magnetic stimulation is a therapy which targets functional brain regions. There exist several methods in order to localise these regions but they don't achieve high accuracy. Convolutional neural networks are promising but their high R.A.M. usage is limiting their use. This thesis offers a comparison between Baxter et al.'s hierarchical convolutional neural network architecture (2021) and a traditional convolutional neural network architecture based on two points : R.A.M. usage and accuracy. The hierarchical architecture proves to be more efficient, with a lesser R.A.M. usage and an accuracy better than or equal to the alternative.</p>	
<p>Mots-clés : stimulation magnétique transcrânienne répétitive, cerveau, réseau de neurones.</p> <p>Key Words: rTMS , brain, deep learning.</p>	

\* Élément qui permet d'enregistrer les notices auteurs dans le catalogue des bibliothèques universitaires