CentraleSupélec

IAEM

A propos des architectures de cartes auto-organisatrices stylées

THÈSE

présentée et soutenue publiquement le le plus tard possible en 2022

pour l'obtention du

Doctorat CentraleSupelec

(mention informatique)

par

Noémie Gonnier

Composition du jury

Président :	Le président	du jury
Rapporteurs :	Le rapporteur 1	du laboratoire
	Le rapporteur 2	
	Le rapporteur 3	
Examinateurs :	L'examinateur 1	
	L'examinateur 2	

Laboratoire Lorrain de Recherche en Informatique et ses Applications

Loria
Laboratoire lorrain de recherche en informatique et ses applications

Mis en page avec la classe thesul.

Sommaire

Introduction

Cette thèse propose une contruction d'une architecture modulaire

Chapitre 1

Approche modulaire des réseaux de neurones

Chapitre 2

Cartes de Kohonen et modèle d'architecture CxSOM

Idée du chapitre :

"Qu'est ce qu'on veut faire avec des cartes de Kohonen ?" " A quoi servent les cartes de Kohonen ?" ok on les utilise pour de la visualisation, de la réduction de dimension. La visualisation est bien pour un observateur humain, la réduction de dimension peut impliquer qu'on va utiliser un algorithme derrière. Mais les cartes de Kohonen vont plus loin dans l'apprentissage : on a une approximation de l'espace d'entrée par un graphe. Cela veut dire qu'une entrée est associée à un prototype dans la carte, mais inversement : un prototype est associé à un ensemble d'entrée continu ou contigu. Une entrée est alors représentée par notamment sa position dans la carte : un nombre donc, ou une paire. Il est possible de faire du calcul sur ces positions au sein d'algoithmes.

Dans cette thèse, on a pensé à utiliser cette propriété pour construire un réseau de cartes auto-organisatrices. Par ce réseau, on peut exploiter les positions pour générer des dynamiques au sein de la carte qui permettront une prise de décision, ou des représentation de donnée différentes.

// Kohonen : il faut surprendre encore ! Par quel bout le prendre ? → Appuyer sur les cartes 1D → Comment ca se fait qu'on les utilise pas de ouf ? → Intérêt de la topologie de la carte. Dans une carte seule, est ce que c'est vraiment utile ? → Questionnement informatique : qu'est ce qui se passe en fait dedans, mais c'est quand même rigolo.

2.1 De la biologie au calcul

De la biologie au calcul : patterns temporels des neurones impulsionnels vs SOM

2.2 Algorithme Général

Une carte de Kohonen est un graphe dans lequel chaque noeud possède un poids ω appartenant à l'espace des entrées. L'algorithme repose ensuite sur l'adaptation de ces poids, en prenant en compte les connexions dans le graphe, afin de représenter les données d'entrées. Ainsi, n'importe quel graphe pourrait être considéré ; le plus souvent, une grille 2D est utilisée.

Mettre ici algo

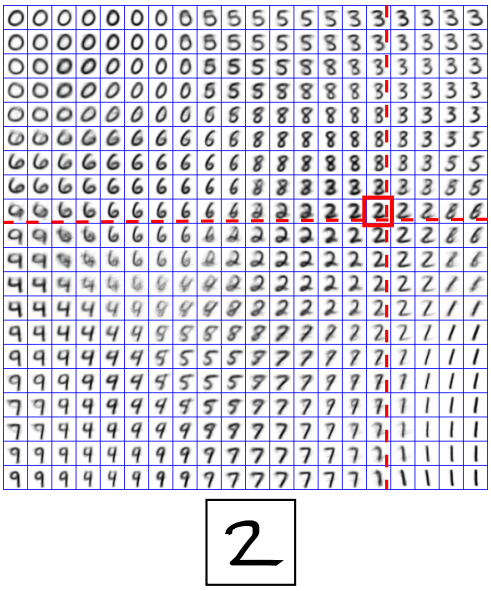

FIGURE 2.1 – Une carte de Kohonen s'organise en zones dont les poids sont proches dans l'espace des entrées. Chaque entrée présentée à la carte peut alors être représentée par la valeur de la position du BMU correspondant dans la carte. Les entrées sont projetées sur le carré $[0, 1] \times [0, 1]$.

2.3 Approche topologique des cartes de Kohonen

La notion de voisinage et de topologie est un élément clé des cartes de Kohonen. Le voisinage est en effet pris en compte lors de l'apprentissage et lors de l'interpretation des cartes. Cependant, ce voisinage est généralement défini, dans les applications des cartes, comme un bonus par rapport aux KMeans, une aide à la convergence et à la vitesse de dépliement. Pourtant c'est la l'essence même d'une carte de Kohonen : projeter des éléments sur un graphe, ce qui nous permet de faire des calculs sur des positions plutot que des données de grandes dimensions.

2.4 Description de l'algorithme

Le but de cette thèse est de proposer un modèle permettant d'associer des cartes auto-organisatrices dans n'importe quel type d'architecture, comme une sorte de brique de base. En particulier, on cherchera à construire des architectures non-hiéarchiques de cartes, exemple en figure 2.2. Nous nous placons donc dans le cadre de modules pré-établis, dont les entrées ont été connectées par avance. Les poids de chaque carte seront quant à eux appris, avec comme objectif que les cartes apprennent leurs entrées mais puissent également distinguer un état global de l'architecture. Pour les entrées, nous nous placons dans un cadre de multi-modalité, détaillé au chapitre suivant. Les différentes cartes prendront des données d'entrées sur différents espaces.

2.4.1 Carte de Kohonen classique

Rappelons les notations concernant une carte de Kohonen standard. Prenons un ensemble de données d'entrées, dans lequel chaque élément est un vecteur d'un espace D. On a définit une distance d sur D, généralement la distance euclidienne. La carte de Kohonon construite sur ces entrées est un graphe, généralement une ligne 1D ou une grille 2D de N noeuds. Chaque noeud

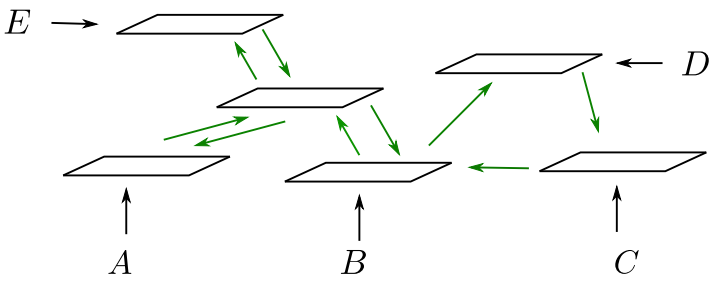

FIGURE 2.2 – Exemple d'architecture modulaire *non-hiérarchique* de cartes de Kohonen. Les entrées sont A, B, C, D, E quelconques. Chaque carte peut ou non prendre une entrée ; les connexions sont réciproques ou non.

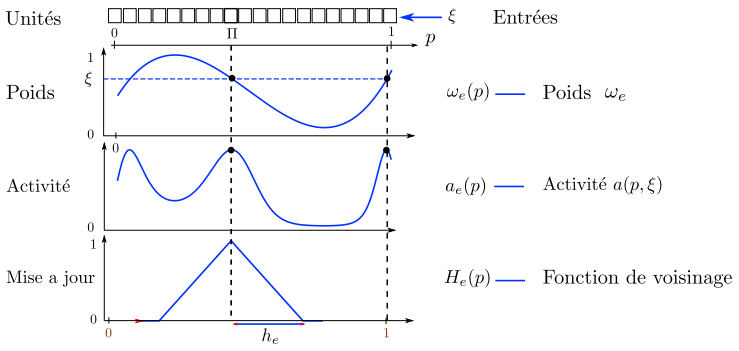

FIGURE 2.3 – Notations utilisées dans une carte de Kohonen simple

possède un poids associé $\omega_e inD$ ou *prototype*, du même espace que les entrées. et une *position* i dans la carte. Ces positions sont ensuite indexées entre 0 et 1 par $p = \frac{i}{N}$ pour l'homogénéité des calculs. L'ensemble des poids est noté $\omega_e(p), p \in [0, 1]$. L'algorithme se décompose de la façon suivante :

1. Une entrée ξ_t est présentée à la carte.
2. L'unité ayant le poids le plus proche de ξ_t selon la distance d est choisie comme *Best Matching Unit* de la carte. Sa position est notée Π.
3. Chaque poids ω_e est déplacé vers l'entrées ξ, en fonction de sa distance dans la carte à la best matching unit :

$$\omega_e(p, t+1) = \omega_e(p, t) + \alpha h(\Pi, p)(\xi - \omega_e(p, t)) \tag{2.1}$$

$h(\Pi, p)$ est la *fonction de voisinage*. Elle est maximale en $p = \Pi$ et décroissante autour de cette position. Dans notre étude, les fonctions de voisinage sont triangulaire, donc maximale en Π, décroissante sur le rayon de voisinage h_e et nulle après.

Lors de l'étape 2 de l'algorithme, une activité peut être calculée, au lieu d'une distance pour choisir le BMU. Ce dernier est alors choisi comme $\Pi = \arg\max_p(a(\xi, p)$. Nous utiliserons cette solutions dans notre modèle. Les notations au sein d'une carte sont résumées en figure 2.3.

2.4.2 Modèle : CxSOM

Décrivons maintenant le modèle CxSOM étudié dans cette thèse. Dans ce modèle, l'algorithme original de Kohonen est modifié afin de connecter des cartes entre elles, et d'autoriser des connexions non-hiérarchiques. Définissons la connexion entre deux cartes. Une carte A est connectée à une carte B lorsque la carte B prend en entrée la position du BMU de la carte A. Considérons G, le graphe de connexions des cartes. Ce graphe est *orienté* et les *boucles* sont autorisées. C'est ce qu'on appelera *architecture non-hiérarchique* de cartes, par opposition à des architectures comme HSOM dans laquelle le BMU d'une carte A nourrit une carte B de façon unidirectionnelle. Chaque carte aura ainsi plusieurs entrées : une entrée *externes* dans un espace d'entrée, facultative, et k entrées *contextuelles* qui sont les positions des BMU des cartes qui lui sont connectées. Par ailleurs, la recherche du BMU doit être modifiée par rapport à l'originale : les rétroactions entre les cartes sont autorisées, la position du BMU de la carte A va donc influencer la position du BMU de la carte B, lequel modifie à nouveau le BMU de la carte A, etc. Notre algorithme présente donc deux modifications principales :

— Les cartes possèdent plusieurs entrées, externes et contextuelles. Le calcul de l'activité est donc modifié afin de prendre en compte ces différentes couches d'entrées.
— La recherche du BMU est modifiée afin de gérer les rétroactions entre cartes.

La description du modèle CxSOM est détaillée en figure 2.5, dans un cas ou une carte reçoit deux connexions, et l'algorithme explicité en **??**.

Gestion des entrées externes et contextuelles

A un pas d'apprentissage t, une carte M reçoit en entrée une entrée *externe* notée ξ_t et K entrées *contextuelles* notées $\gamma_{0t}, \cdots, \gamma_{Kt}$, qui sont les BMU Π des cartes qui lui sont connectées. La carte possède donc $k+1$ couches de poids. ω_e correspond à l'entrée externe et $\omega_{c0}, \cdots, \omega_{cK}$ aux entrées contextuelles. On calcule une activité séparément sur chaque couche de poids selon la formule suivante :

$$a(p,x) = \exp(\frac{(\omega(p)-x)^2}{2\sigma^2} \; x = \xi_t \text{ ou } \gamma_{kt}, \; \omega = \omega_e \text{ ou } \omega_{ck} \tag{2.2}$$

Les activités contextuelles sont moyennées en une activité $a_c(p, \gamma_t)$, avec $\gamma_\mathbf{t} = (\gamma_{0t}, \cdots, \gamma Kt)$. Les activités externes et contextuelles sont enfin fusionnées en une activité globale :

$$a_g(p, \xi_t, \gamma_\mathbf{t}) = \sqrt{a_e(p,\xi_t)(\beta a_e(p,\xi_t) + (1-\beta)a_c(p,\gamma_\mathbf{t})} \tag{2.3}$$

Une convolution est appliquée sur cette activité globale. Cela évite les effets de plateau. Cette activation globale est utilisée pour déterminer le BMU de la carte.

Gestion des rétroactions dans l'architecture

Contrairement à une carte simple, on ne peut pas calculer tous les BMUs de l'architecture en prenant l'argmax de a_g dans chaque carte. A cause des influences mutuelles entre cartes, calculer le BMU d'une des cartes modifie les entrées des autres cartes de l'architecture, et donc leur BMU. Cette recherche est donc réalisée par un processus dynamique que l'on appelera *relaxation*, menant à un consensus entre cartes : on cherche le point, s'il existe, où chaque BMU maximise l'activité globale de chaque carte.

Le processus de relaxation est donc une boucle imbriquée dans un pas d'apprentissage de l'architecture, indexée par τ. Notons $\Pi^{[}i]$ la position du BMU de la carte i, et $\mathbf{\Pi} = (\Pi^{[}0], \cdots, \Pi^{[}n])$,

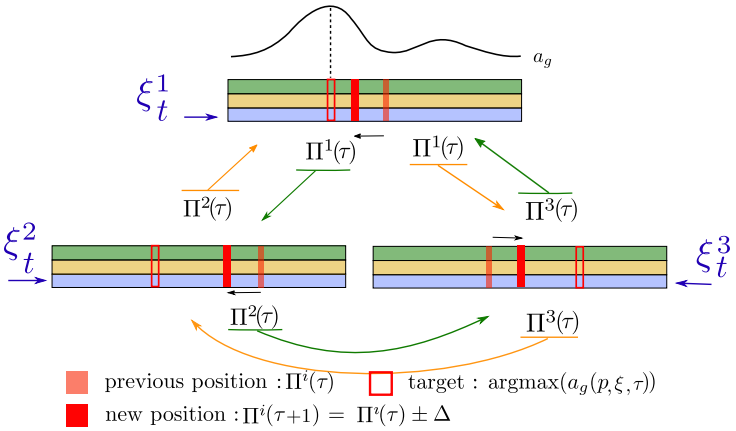

FIGURE 2.4 – description d'une étape de la relaxation dans l'architecture, aboutissant à un consensus entre cartes. Au sein d'une même itération t, les position des BMU Π sont légèrement déplacées jusqu'à ce que toutes les positions Π des cartes de l'architecture soient stable. Ces positions maximisent collectivement les activités globales de chaque carte.

avec n le nombre de cartes de l'architecture. Au début d'un pas d'apprentissage, chaque carte est nourrie avec une entrée externe ξ_t^i, et les activités externes $a_e^i(\xi_t^i, p)$ de chaque carte peuvent être calculées. La recherche du BMU suit donc le processus de relaxation suivant :

1. Dans chaque carte i, la position Π^i est initialisée à $\arg\max_p(a_e^i(\xi_t^i, p)$. Les entrées contextuelles sont alors initialisées en prenant le BMU correspondant aux connexions de l'architecture.
2. Tant que toutes les positions Π^i ne sont pas stables,
   (a) Dans chaque carte i, calculer les activités contextuelles et globales, définissant ainsi $p^{\star i} = \arg\max_p(a_g(p, \gamma^{\mathbf{i}}, \xi^i)$
   (b) Déplacer Π^i vers $p^{\star i}$: $\Pi^i \leftarrow \Pi^i \pm \Delta$ si $|\Pi^i - p^{\star i}| \geq \Delta$, $\Pi^i \leftarrow p^{\star i}$ sinon
3. Le BMU de chaque carte est pris comme la valeur finale stable de ce processus dynamique. Cette valeur est utilisée pour les mise a jour des poids.

Il peut arriver que les positions se stabilisent sur un cycle limite. Dans ce cas, on arrêtera la relaxation arbitrairement ; ce phénomène étant ponctuel, il n'influencera pas l'apprentissage. Les paramètres des cartes de l'architecture sont choisis pour éviter de telles situations.

Mise à jour des poids

Les poids sont mis à jour par rapport à leurs entrées respectives suivant l'équation 2.1. Le BMU d'une carte est ainsi commun à toutes les couches. Les rayons de voisinage h_e et h_c ont des valeurs différentes ; celles-ci seront détaillée en partie suivante.

Tests

Les expériences faites sur l'architecture se décomposent en une phases d'apprentissage et phases de test. Pendant les tests, la mise à jour des poids des cartes est gelée et seuls le calcul des activités et le processus dynamique de sélection du BMU sont effectués.

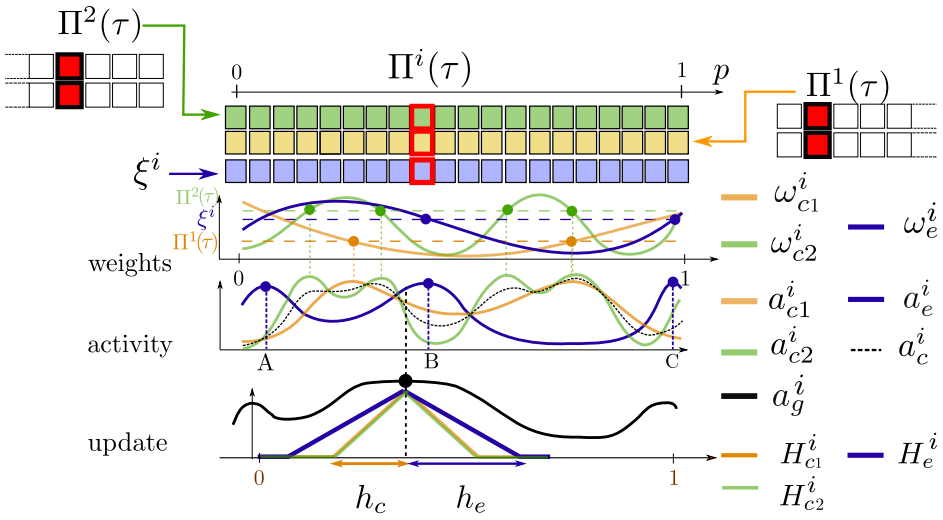

FIGURE 2.5 – Description d'une carte au sein d'une architecture CxSOM. La carte recoit deux connexions de cartes voisines, et possède donc deux couches contextuelles

2.5 A trier

.

Chapitre 3

Représentation des cartes de Kohonen

Sommaire

3.1 Problématique

La problématique de la représentation d'un algorithme d'apprentissage est un défi posé depuis quelques années pour, par exemple, les algorithmes de deep learning. Autant de nombreuses métriques existent pour qualifier la qualité d'un apprentissage supervisé, autant le problème est largement ouvert quand il s'agit de comprendre les mécanismes d'apprentissage. Cette question de représentation est notamment soulevée dans l'étude de l'explicabilité de l'intelligence artificielle.

Les cartes de Kohonen,de leur coté, sont généralement associées à une facilité de représentation et de visualisation. En effet, leur nombre réduit de prototypes et leur aspect topologique permet d'en tracer une représentation visuelle facilement interprétable. Cette facilité d'interprétation, cependant, est limitée à un domaine d'utilisation : celui dans lequel les éléments qui nous intéressent sont les distances euclidiennes entre les données, ou plus généralement dans lequel la distance considérée pour la mise a jour des cartes possède un pendant graphique directement interprétable à l'oeil humain.

Dans un cas plus général de cartes auto-organisatrices, telles que celles agissant dans CxSOM, l'apprentissage repose sur des calculs d'activités et un processus de relaxation. Ces activités n'ont

FIGURE 3.1 – Représentations possible des poids d'une carte de Kohonen classiques, dans le cas d'entrées sous forme d'imagettes ou de points en deux dimensions.

pas forcément un pendant graphique. De ce fait, leur représentation graphique mérite une analyse plus approfondie que dans le cas de cartes classiques.

Enfin , "Est ce que les prototypes ont extrait une information pertinente des données" n'a en fait que des réponses partielles dans la littérature. Il s'agit d'abord de déterminer ce qu'on cherche à apprendre dans un cas spécifique. Nous nous poserons ainsi cette question pour l'architecture CxSOM.

3.1.1 Représentation classique des cartes de Kohonen

La manière la plus répandue de représenter une carte de Kohonen est de tracer les poids de ses prototypes, disposés dans le graphe qu'est la carte. En fonction des dimensions des entrées, cette représentation prennent plusieurs formes. Deux exemples courants de représentation sont les suivants :

- Le graphe qu'est la carte de Kohonen est représenté dans l'espace de ses positions (la grille d'indices (i, j), ou une ligne indexée par i. Sur chaque noeud est tracé le poids correspondant. C'est le cas par exemple en figure 3.1.1 dans lequel les poids des prototypes, qui sont des imagettes, sont affichés en chaque point de la grille. Si la dimension d'un poids est trop grande pour être représentée graphiquement, il est également courant de labeliser chaque prototype et d'afficher ces labels sur les noeuds de la carte, en tant que représentation.
- Lorsque les données traitées sont des points deux ou trois dimensions, les poids des prototypes peuvent être directement tracés dans l'espace R^2 ou R^3. Ces poids sont alors reliés en fonction des positions des noeuds dans la carte, montrant ainsi la déformation de la carte dans l'espace d'entrée, c'est le cas en figure 3.1.1.

Ces représentations sont particulièrement adaptées à un observateur humain. Il est ainsi alors facile de labeliser les prototypes, de visualiser les clusters ... Ceci étant conditionné à ce que les cartes se soient organisées en suivant une distance. Lorsqu'il regarde la carte, l'oeil humain se demande : "est-ce que la carte est bien dépliée sur toutes les données ? Est-ce qu'un prototype représente correctement une donnée ?" Intellectuellement, il imagine en fait une donnée, par exemple une imagette d'un chiffre à main levée, et reproduit le processus de sélection du BMU qui a eu lieu lors de l'apprentissage de la carte pour trouver le poids qui lui correspond le mieux.

3.1.2 Limites de cette représentation dans CxSOM

Dans CxSOM, le choix du BMU se fait suivant plusieurs activités, et plus encore, suivant un processus de relaxation. Il est bien entendu possible de tracer les poids d'une carte après

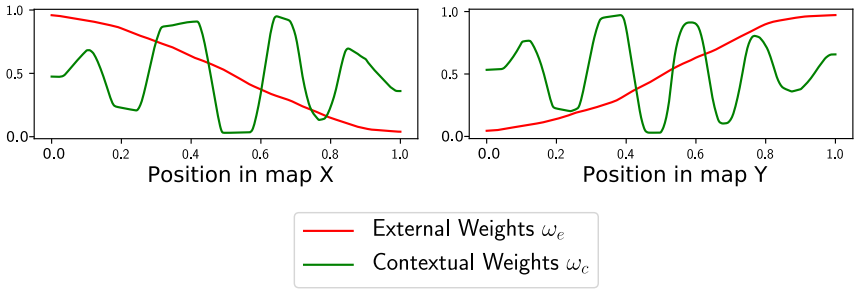

FIGURE 3.2 – Représentation des valeurs des poids d'une carte au sein de CxSOM. La seule représentation de ces poids ne suffit pas à savoir comment la carte se comporte.

apprentissage, comme représenté en figure 3.1.2. Cependant, le processus intellectuel menant à la représentation mentale d'une carte, en regardant les poids, n'est plus possible. En imaginant une donnée, on ne pourra pas trouver le BMU selectionné. La simulation du processus d'activité et relaxation est nécessaire pour la représentation compréhensible par l'humain d'une carte au sein de CxSOM. Par ailleurs, chaque unité d'une carte a plusieurs poids. Il est donc compliqué de comprendre directement le rôle de ces poids en regardant leur valeur.

De plus, l'intéret de CxSOM réside dans la communication entre cartes. Représenter les cartes une à une laisse donc de coté leur connexion. Il est donc nécessaire de trouver un moyen de représenter l'architecture comme un tout. Enfin, la représentation visuelle des cartes est limitée par la dimension des entrées et la dimension des cartes. La représentation visuelle d'une carte classique est seulement limitée par la dimension des entrées. Ici s'ajoute à la dimension des entrées la dimension d'une carte et le nombre de carte. Il sera difficile de représenter graphiquement des architectures de plus de trois cartes, et encore plus lorsque les entrées sont en grande dimension. Cette difficulté de représentation soulève la nécessité de définir des valeurs indicatrices du fonctionnement de la carte, calculables en grande dimension.

Ce chapitre questionne donc la façon de représenter une carte de Kohonen, et plus particulièrement la façon de représenter une carte au sein d'une architecture. Nous présenterons donc en premier lieu un formalisme pour la carte et les entrées multimodales associées, et à partir de ce formalisme nous proposerons plusieurs représentations et indicateurs cherchant à comprendre ce que l'architecture apprend sur les données d'entrée, et de quelle façon.

3.2 Formalisme : variables aléatoires

Nous introduisons dans cette section un formalisme traitant les éléments des cartes et les entrées en tant que variables aléatoires. Ce formalisme a l'avantage de à la fois clarifier les représentations, et de permettre le développement d'indicateurs statistiques sur les cartes.

3.2.1 Représentation des entrées

Les observations multimodales que l'on cherchera à apprendre par l'architecture de cartes sont notées $X^i, i = 0 \cdots N$ où N est le nombre de modalités considérées. Lors de l'apprentissage et du test, elles sont échantillonnées ; ainsi, à chaque pas de temps, l'architecture se voit présentée un vecteur $(X_t^0, \cdots, X_t^N)$. Lorsqu'elles sont considérées en tant que *entrée externe* d'une carte, on les notera plutôt $\xi^i, i = 0 \cdots N$, avec i l'indice de la carte dont ξ^i est l'entrée. Pour tout i, X^i et ξ^i sont des variables aléatoires, et $\mathbf{X} = (X^0, \cdots, X^N)$ et $\xi = (\xi^0, \cdots, \xi^N)$ sont les vecteurs

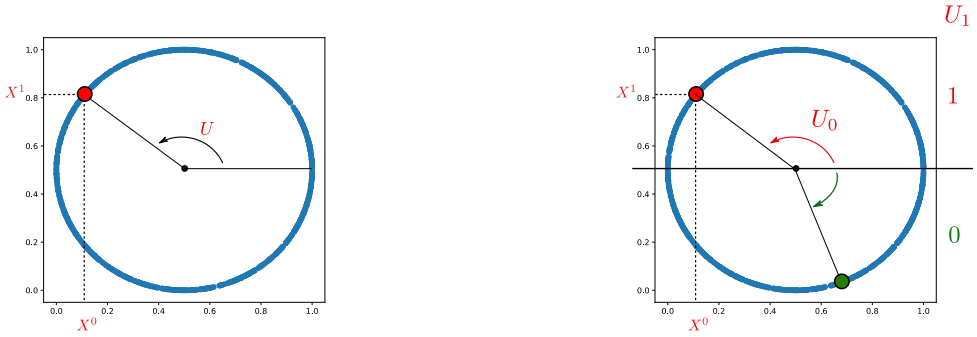

FIGURE 3.3 – Exemples de paramétrisations du cercle. La paramétrisation qui traduit le plus facilement le modèle est naturellement celle dans laquelle U est à valeurs réelles. Le modèle auxquelles appartiennent les modalités X^0 et X^1 est donc représenté par la variable cachée U.

aléatoires correspondants.

Pour les entrées CxSOM, on s'intéresse à l'apprentissage de relations entre entrées. Les variables X^i ne sont a priori donc pas des variables indépendantes. Afin de mieux comprendre comment les cartes apprennent des relations entre les entrées, on introduit une autre variable aléatoire U. Cette variable est multidimensionnelle et est choisie de façon à ce que chaque variable X^i soit une fonction quelconque de la variable aléatoire U, et uniquement de cette variable.

$$\forall t, \forall i, X_t^i = f_t^i(U_t) \tag{3.1}$$

Cette variable traduit l'existence d'un modèle reliant les observations. Prenons un exemple géométrique ; considérons des points tirés sur un cercle quelconque dans l'espace en deux dimensions. $\mathbf{X} = (X^0, X^1)$, les coordonnées cartésiennes des points du cercle, est alors une vecteur aléatoire, dont les composantes sont les variables aléatoires X^0, X^1. En définissant une variable U à valeurs réelles, chaque point du du cercle peut maintenant s'écrire, selon l'équation paramétrique du cercle :

$$\begin{cases} X_t^0 = r\cos(U_t) \\ X_t^1 = r\sin(U_t) \end{cases}.$$

U représenterait ici l'angle du point sur le cercle. U est une variable cachée qui *réduit la dimension* du modèle. ELle contient toute l'information sur l'échantillon.

U et f^i ne sont pas uniques. Elle sont choisies en fonction de ce qu'on cherche à traduire dans le modèle. Ainsi, pour le même ensemble de points sous forme de cercle, on peut aussi définir une variable U en deux dimensions, une dimension à valeur réelles paramétrisant un demi cercle, l'autre à valeurs dans $0, 1$ indiquant de quel coté de l'axe des abscisses on se situe.

Notons par contre que la plus petite dimension possible de U dépend du degré de liberté du modèle. Si toutes les observations se situent sur une courbe de dimension 1, alors il existe une variable U en une dimension satisfaisant l'équation 3.1. Si les observations se situent sur une surface de dimension 2, alors, U sera en deux dimensions, et ainsi de suite.

Cette façon de représenter les entrées est-elle générale ?

3.2.2 Représentation des éléments des cartes

Comme dans de nombreux algorithmes d'apprentissages, on peut décomposer le jeu de données d'entrée en jeu d'apprentissage et jeu de tests. Lors de la phase de test, seul le processus

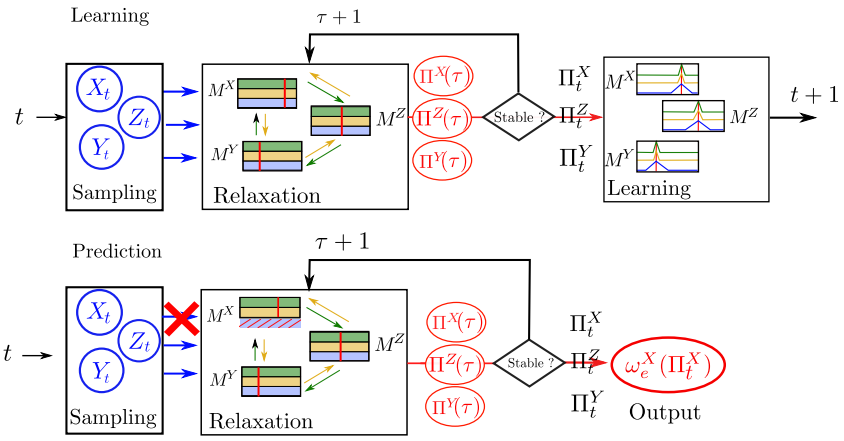

FIGURE 3.4 – Schéma descriptif de l'apprentissage et des tests.

de recherche de la best matching unit est réalisé et la partie mise à jour des cartes de Kohonen n'est plus opérée. Dans le cadre des variables aléatoires, chaque itération est alors un tirage indépendant. Les éléments des cartes peuvent donc être considérés comme des variables aléatoire et une itération de test comme la réalisation de celles-ci. La phase de test peut être réalisée après n'importe quelle itération de l'algorithme d'apprentissage. Le processus d'apprentissage et de tests est décrit en figure 3.4.

Nous considérerons alors plusieurs éléments des cartes en tant que variables aléatoires, notamment :

— Les positions des BMUs $\Pi^0, \cdots, \Pi^N$ dans chaque carte
— Les poids externes des BMUs $\omega_e^0(\Pi^0), \cdots, \omega_e^N(\Pi^N)$

Notons que tout élément d'une carte pourrait être vu de cette manière. Une phase de test est donc un grand nombre de réalisations d'une variable aléatoire jointe :

$$(\xi^0, \cdots, \xi^N, \Pi^0, \cdots, \Pi^N, \omega_e^0(\Pi^0), \cdots, \omega_e^N(\Pi^N))$$

Les composantes de cette variable jointe ne sont pas indépendantes. Les représentations et indicateurs présentés ensuite chercheront à détecter et comprendre au mieux ces dépendances statistiques.

Ainsi, dans ce formalisme par variable aléatoires, à chaque pas d'apprentissage peut-être associé un ensemble de réalisations de variables aléatoires. Ceci permet alors d'utiliser des outils et métriques issus de la théorie de l'information pour qualifier l'organisation des cartes au sein de l'architecture. Cette approche ne se limite pas à l'architecture CxSOM :

3.3 Représentations graphiques

Qu'est ce qu'une carte a appris des données ?

Que cherche t-on à représenter dans les représentations classiques des poids des cartes de Kohonen, que ce soit sous la forme d'un tableau de prototypes ou d'une projection dans l'espace d'entrée ? On cherche en fait à visualiser comment les poids sont répartis en fonction de leur *position* dans la carte. En d'autres termes, on cherche à comprendre si les positions de la carte correspondent à tous les éléments de l'espace d'entrée, si une continuité est réalisée. D'une façon similaire, on peut faire le choix de représenter le poids de la best matching unit par rapport à sa

position. Cela donne la même représentation que le fait de tracer le poids de chaque prototype par rapport à sa position dans la carte ; à la seule différence qu'elle fera la distinction entre les *unités mortes de la cartes*, c'est à dire les unités qui ne sont jamais best matching unit et qui ne seront donc pas affichée dans la représentation des tests, et les autres. Cette représentation prend en compte la façon de calculer le BMU, donc le coeur de l'algorithme.

La question de la répartition des valeurs d'une carte par rapport à la position de leur BMU va plus loin que les poids : il est intéressant d'étudier la répartition de n'importe quel élément d'une carte de cette façon, afin de comprendre De façon plus générale, on peut représenter, à partir d'un échantillon test, la dépendance de n'importe quelle variable par rapport à la position de la best matching unit correspondante. Nous détaillerons dans cette partie quelques représentations qui paraissent pertinentes.

3.3.1 Représenter les entrées par rapport à une carte

Représentation des entrées de test sur les poids de la carte

3.3.2 Représentation de U par rapport au BMU

Chercher à apprendre des relations entre les données

Un des objectifs principaux de l'architecture CxSOM est d'encoder les relatations entre les données d'entrée. Il nous faut donc tracer des valeurs qui expriment cette relation. Expliquer mieux comment U est choisi : il s'agit d'une transformation non linéaire des données sur un autre espace, dans lequel U.

TODO : cas des cartes 2D, cas des entrées

3.3.3 Représentation des cartes sous forme de "distortion" (trouver un mot)

Lors

3.4 Information mutuelle comme indicateur statistique

De nombreuses valeurs ont été développées en théorie de l'information, depuis Shannon en 1948 (citer), pour mesurer des dépendances entre variables. Lister les spécificités des mesures et dans quel cas on peut les utiliser : Mesure proba / estimation (estimation en grande dimensions, etc) Est ce que les distributions doivent etre connues ...

Exemple de domaines d'application de ces mesures

Ces mesures s'appuient sur des variables : elles ne dépendent pas du modèle.

3.4.1 Information mutuelle et entropie

Les notions d'*entropie* et les valeurs qui en sont dérivées, telle que l'*information mutuelle* entre des distributions, sont des notions fondamentales de la théorie de l'information de Shannon. Ces quantités donnent des informations concernant la distribution d'une variable aléatoire. Les formules indiquées dans ce paragraphe concernent des variables aléatoire discrètes. L'entropie de Shannon d'une variable aléatoire X, de distribution $p(X)$, est notée $H(X)$ et définie par la formule :

$$-\sum_{x \in X} p(x) \log(p(x))$$

Elle se mesure en *bit/symbole* lorsque le le logarithme est en base 2, ce qui est généralement utilisé. L'entropie est une mesure de la quantité d'incertitude, ou de surprise, sur la valeur de la variable aléatoire X. Si la la distribution de probabilité de X est concentrée autour d'un point, l'entropie est faible : lors d'une réalisation de X, l'observateur est *plutôt certain* du résultat. En revanche, l'entropie est maximale lorsque lorsque X suit une distribution de probabilité uniforme. L'entropie s'interpète également comme la quantité moyenne d'information à fournir, en bits, pour coder la valeur que prend la variable X. De la même manière, on peut définir l'entropie conjointe de deux variables, qui est l'entropie de leur distribution jointe, et l'entropie conditionnelle, qui est l'entropie de leurs distributions conditionnelles.

Outre les entropies jointes et conditionnelles, les relations statistques entre deux variables aléatoires peuvent être mesurées par *l'information mutuelle*. Elle se définit formellement par :

$$I(X,Y) = \sum_{x,y \in X,Y} p(x,y) \log(\frac{p(x,y)}{p(x)p(y)})$$

Cette valeur mesure la quantité d'information moyenne apportée par une réalisation de X sur la réalisation de Y. L'information mutuelle possède notamment les propriété suivantes :

$$I(X,Y) = 0 \Leftrightarrow \text{X et Y sont indépendantes}$$

Cette propriété se comprend dans la définition de I : si X et Y sont indépendantes, $p(x,y) = p(x)p(y)$ et terme $\log(\frac{p(x,y)}{p(x)p(y)}$ est nul pour toute les valeurs de x et y. Inversement, $I(X,Y) = 0$ ssi tous les termes (positifs) de la somme sont nuls, donc si $(p(x,y) = p(x)p(y)$ pour toutes les valeurs de X et Y. L'information mutuelle est donc aussi une mesure de la distance entre la distribution jointe de (X,Y) et leur indépendance.

Elle s'exprime à partir de l'entropie :

$$I(X,Y) = H(X) + H(Y) - H(X,Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

Elle est symétrique :

$$I(X,Y) = I(Y,X)$$

Pour toute fonction f, $I(X,Y) \geq I(X, f(Y)$. L'égalité est atteinte ssi f est *bijective*. (proof ?)

rapporter a la corrélation -> le taux de corrélation mesure les dépendances linéaires, info mutuelle tout type de dépendance statistique.

3.4.2 Indicateur

Lors de l'analyse de CxSOM, on souhaite comprendre l'information que portent les positions des BMUs d'une carte sur le modèle d'entrées. Nous avons défini les éléments de la carte en terme de variables aléatoire ; l'information mutuelle peut alors être une représentation pertinente de l'information portée par le BMU d'une carte sur le modèle. Le modèle est représenté par la variable (X,Y,Z), mais aussi par U. Dans ce sens, $I(\Pi, U)$ est l'information que porte le BMU d'une carte sur U, donc sur le modèle.

On souhaite cependant avoir un indicateur absolu, qui permettrait, sur une échelle de 0 à 1, de quantifier à quel point un BMU porte de l'information sur U. On va donc normaliser l'information mutuelle $I(\Pi, U)$ par la valeur maximale qu'elle peut prendre dans notre carte.

Propriété 1. *La valeur maximale atteinte par $I(\Pi, U)$ est $H(U)$, atteinte lorsque U est fonction de Π.*

Démonstration. Par construction, Π est une fonction de U dans une carte de Kohonen. En effet, notre algorithme est déterministe, et défini pour toute valeur de U. Par propriété de l'information mutuelle, pour toute fonction f et variable X, Y, $I(X, f(Y)) \leq I(X, Y)$. Donc, $I(U, \Pi) \leq I(U, U) = H(U)$ Cette valeur est atteinte si et seulement si U et Π sont en bijection, autrement dit, ssi U est aussi une fonction de Π. □

Nous définissons donc un indicateur de la relation entre U et un BMU comme :

$$UC(\Pi; U) = \frac{I(\Pi, U}{H(U)} \tag{3.2}$$

Ce coefficient n'est pas symétrique, et mesure donc l'information portée par le premier terme sur le second, relativement à la valeur maximale qu'elle peut prendre. Dans le cas des cartes CxSOM, $UC \in [0, 1]$.

3.4.3 Estimation

3.4.4 Expériences et Résultats

Chapitre 4

Analyse de l'architecture modulaire, champs d'application

Avant de pouvoir étudier une architecture de cartes, il est nécessaire de se pencher sur les outils de visualisation de ces cartes, ainsi que sur les indicateurs qu'on peut étudier pour qualifier les comportement. Il faut noter qu'une carte de Kohonen, malgré son fonctionnement apparemment simple, se montre compliquée lorsqu'il s'agit de l'étudier mathématiquement. On peut donc seulement citer les études proposées par (Cottrell, 2003) a propos de la convergence d'une carte 1D. Les auteurs se posent les questions suivantes :

— Est ce que la carte converge ?

— Comment savoir si une représentation apportée par la carte est pertinente ?

Dans le cas d'une carte 1D, on cherche ces réponses mathématiquement mais les représentation usuelle des cartes permet une intuition du résulat : on a de fortes chances d'avoir juste en supposant que la carte converge. Quant à la représentation, on peut proposer des interprétation visuelles : si la carte couvre toutes les données, si elle est "bien dépliée" à l'oeil, l'apprentissage semble pertinent.

Dans le cas de l'architecture CxSOM, on se trouve dans une situation épineuse : même la visualisation des poids ne permet pas de conclure et savoir si on a bien représenté les entrées.

4.1 Cas d'utilisation : les entrées multimodales

4.1.1 Définition et inspiration biologique

4.1.2 Formalisme

4.1.3 Perspectives

Le formalisme présenté, avec des entrées multimodale comme fonction de variable cachées n'est pas forcément général.

4.2 Représentation des entrées

4.3 Information apprise par une carte

Une idée est de déterminer si une carte a gagné de l'information sur le modèle générant les entrées. Dans le cas simple, ce modèle peut être entièrement reprénté par U ; chaque carte peut

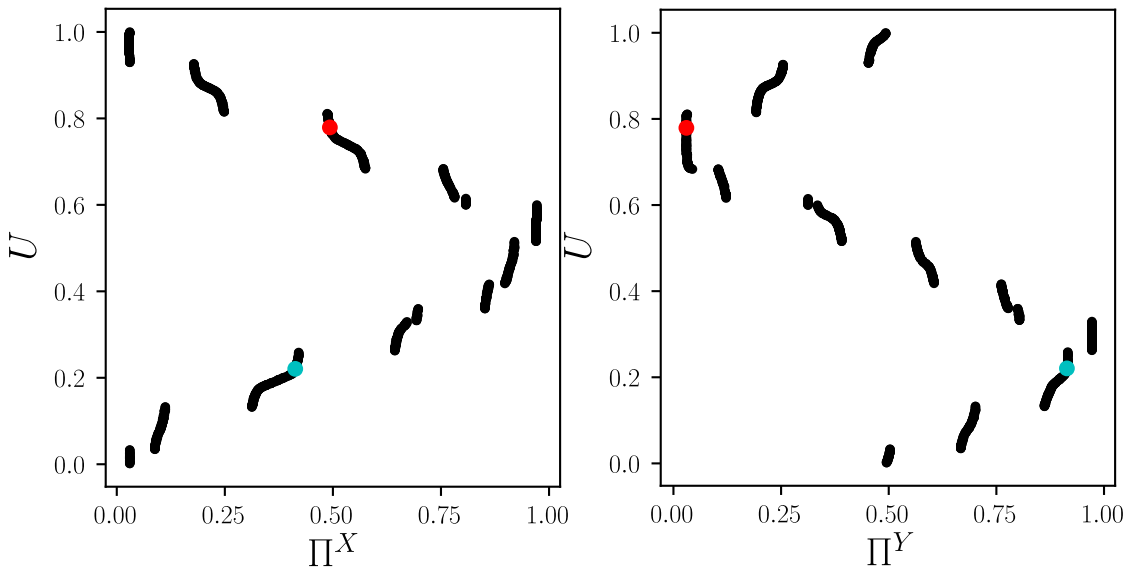

FIGURE 4.1 – Pour l'échantillon de test, valeur de U en fonction des valeurs du BMU Π dans chacune des cartes. On voit que U est une fonction du BMU dans chaque carte, contrairement au cas ou les cartes apprendraient indépendamment sur les mêmes entrées, voir figure 4.2.

être représentée par son BMU, considéré comme la seule sortie de la carte. En tracant U en fonction de Π, le BMU d'une carte, on observe directement si une carte a été capable de lever l'ambiguité sur le modèle en distinguant les entrées selon leur variable cachée U. Cette ambiguité est levée si U est une fonction de Π. Cette fonction est observée dans le cas des cartes jointes.

Cette propriété, dans le cas 1D, peut être calculée par l'information mutuelle entre U et Π. Plus précisément, par $\frac{I(U,\Pi)}{H(U)}$, avec $H(U)$ l'entropie de U. En effet, dans le meilleurs des cas, U est une fonction parfaite de Π et donc $H(U|\Pi) = 0$: en connaissant Π, on connait totalement U. Alors, $I(U,\Pi) = H(U) - H(U|\Pi) = H(U)$. Notre indicateur vaut alors 1 lorsque U est une fonction parfaite de Π. De plus Π est forcément une fonction de U car l'algorithme est déterministe : à une entrée correspond une sortie, toujours la même, donc $(I(U,\Pi) = H(\Pi) =>$ visiblement, non ... Notre indicateur estimant l'information portée par le BMU d'une carte sur la variable cachée du modèle U est donc

$$\frac{H(\Pi)}{H(U)}$$

. du coup, nope.

Cet indicateur doit être estimé en discrétisant les variables, donnant une entropie nécessairement positive et strictement supérieure à 0. L'évolution de l'indicateur au cours de l'apprentissage est donnée en figure 4.3. Cet indicateur est calculé en moyenne pour 100 réalisations de l'apprentissage, avec des poids initiaux différents.

Choses à faire

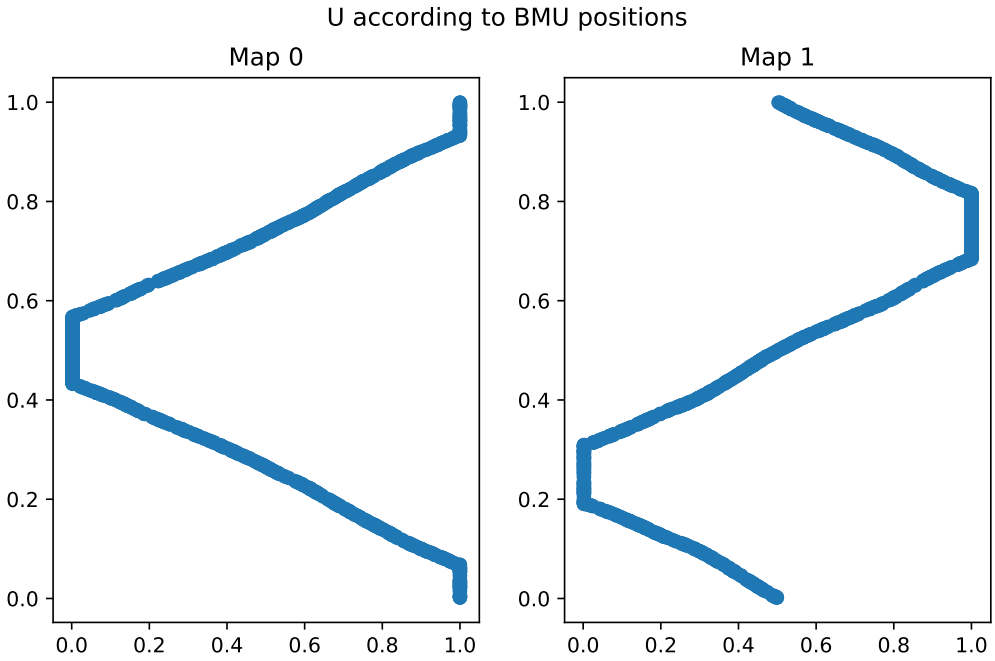

FIGURE 4.2 – Pour l'échantillon de test, entrée sur un cercle, valeur de U en fonction des valeurs du BMU Π dans chacune des cartes, lorsque les cartes M_x et M_y ne sont pas connectée. Chacune des cartes n'a aucune information de plus que celle portée par son entrée sur l'état global du système U, et Π n'est donc pas une fonction de U dans chaque carte.

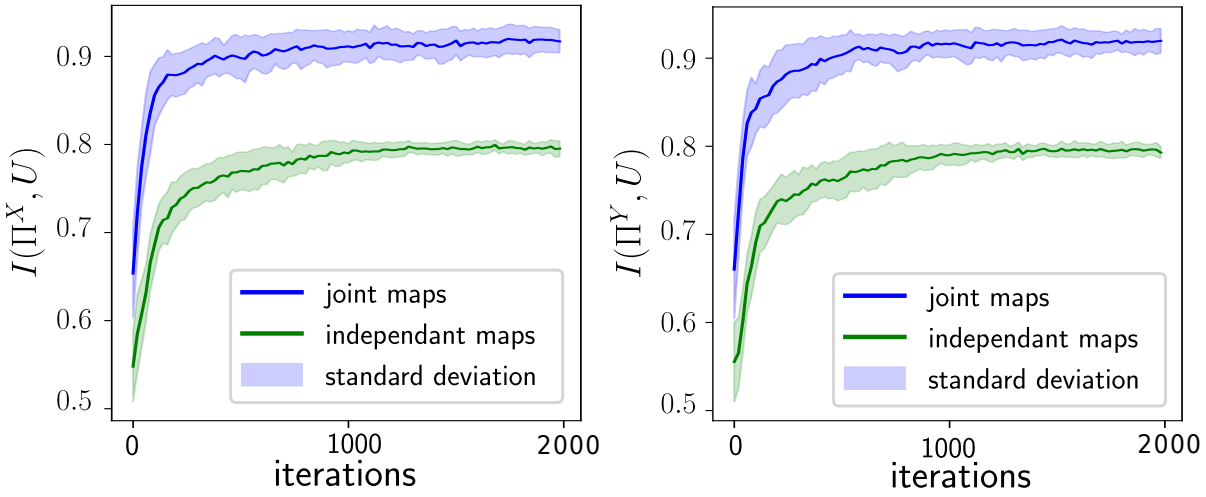

FIGURE 4.3 – Evolution de l'indicateur relatif à l'information mutuelle entre Π et U dans chaque carte au cours de l'apprentissage. Cet indicateur est comparé à celui calculé dans le cas ou les cartes apprennent séparément.

— Cette valeur est uniquement calculée pour un modèle connu, et en 1 dimension forcément. Peut on avoir des équivalents en plus de dimension ?
— Il existe des quantités mesurant l'information portée par un symbole sur une variable, une sorte d'info mutuelle locale. On sait que $I(U, \Pi) = H(\Pi)$, et on veut que $I(U, \Pi) = H(U)$, mais comment est elle répartie entre les BMUs ? Est-ce pertinent de se pencher sur ces quantités ?

4.4 Représenter une carte au sein d'une architecture

Représentation des poids, des entrées, des BMU - analyse

4.5 Choix des paramètres

4.5.1 Influence des rayons de voisinage

4.5.2 Influence des autres paramètres

4.5.3 Compatibilité en 2D

4.6 Analyse de la relaxation

L'apprentissage conjoint des cartes repose sur la relaxation au sein d'une itération. On cherche donc à vérifier si la relaxation converge vers une valeur quelle que soit l'entrée, et si elle est pertinente en large dimension avec de nombreuses cartes.

4.6.1 Analyse expérimentale

4.6.2 Champs de BMU

4.6.3 Limitations et possibilités en grande dimension

4.7 Implémentation

L'implémentation des expériences a été réalisée via l'environnement CxSOM [**?**].

4.8 Perspectives d'évolutions

Avant de présenter les performances d'un algorithme, il s'agit de définir plus précisément ce qu'on attend de ce système et comment le représenter. L'architecture CxSOM se présente comme une construction qui répond à un questionnement structurel des réseaux de neurones. Mais au juste, qu'attend t-on de ce réseau de neurones ? De la prédiction, de l'organisation ? Les cartes de Kohonen sont habituellement utilisées dans un objectif de clustering, ou associées à d'autres algorithmes de prédiction utilisant leurs propriétés structurelles. En étude préliminaire pour CxSOM, il s'agit de comprendre le comportement de l'architecture de cartes.

Chapitre 5

Expériences

5.1 Prédiction d'entrée

Prédction sur des données jouets
Prédiction sur drone
Bien se placer dans le contexte "on va chercher a omprendre ce système dynamique".
Formaliser le problème en terme de variables aléatoires

Conclusion

Bibliographie

[1] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and D. Klein. Neural module networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 39–48, 2016.

[2] A. L. Barabasi and Eric Bonabeau. Scale-free networks. *Scientific American*, 288 :60–69, 2003.

[3] J. He Biyu. Scale-free brain activity : past, present, and future. *Trends in Cognitive Sciences*, 18(9), September 2014.

[4] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE J. Robotics Autom.*, 2 :14–23, 1986.

[5] Aaron Clauset, Cristopher Moore, and Mark E. J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453 :98–101, 2008.

[6] Robert Csordas, Sjoerd van Steenkiste, and J. Schmidhuber. Are neural nets modular ? inspecting functional modularity through differentiable weight masks. *ArXiv*, abs/2010.02066, 2021.

[7] Daniel J. Felleman and David C. Van Essen. Distributed hierarchical processing in the primate cerebral cortex. 1991.

[8] Logan Harriger, Martijn P. van den Heuvel, and Olaf Sporns. Rich club organization of macaque cerebral cortex and its role in network communication. *PLoS ONE*, 7, 2012.

[9] Judit Horváth, István Szalai, and Patrick De Kepper. An experimental design method leading to chemical turing patterns. *Science*, 324 :772 – 775, 2009.

[10] M. Johnsson, C. Balkenius, and G. Hesslow. Associative self-organizing map. In *Proc. IJCCI*, 2009.

[11] Louis Kirsch, Julius Kunze, and D. Barber. Modular networks : Learning to decompose neural computation. In *NeurIPS*, 2018.

[12] J. Lampinen and E. Oja. Clustering properties of hierarchical self-organizing maps. *Journal of Mathematical Imaging and Vision*, 1992.

[13] D. Meunier, R. Lambiotte, and E. Bullmore. Modular and hierarchically modular organization of brain networks. *Frontiers in Neuroscience*, 4, 2010.

[14] S Milgram. The small world problem. *Psychology today*, 2 :60–67, 1967.

[15] Harold J. Morowitz. The mind, the brain, and complex adaptive systems. 1995.

[16] Raj Kumar Pan and Sitabhra Sinha. Modularity produces small-world networks with dynamical time-scale separation. *EPL*, 85 :68006, 2009.

[17] German I. Parisi, Jun Tani, Cornelius Weber, and Stefan Wermter. Lifelong learning of spatiotemporal representations with dual-memory recurrent self-organization. *Frontiers in Neurorobotics*, 2018.

[18] Erzsébet Ravasz, Audrey Somera, D A Mongru, Zoltán N. Oltvai, and A.-L. Barabasi. Hierarchical organization of modularity in metabolic networks. *Science*, 297 :1551 – 1555, 2002.

[19] Edmund T. Rolls and Gustavo Deco. Computational neuroscience of vision. 2002.

[20] C. Watanabe, Kaoru Hiramatsu, and K. Kashino. Modular representation of layered neural networks. *Neural networks : the official journal of the International Neural Network Society*, 97 :62–73, 2018.

[21] Stefan Wermter, Jim Austin, David Willshaw, and Mark Elshaw. Towards novel neuroscience-inspired computing. 2001.