École doctorale n° 77 : IAEM

THÈSE

pour obtenir le grade de docteur délivré par

l'Université de Lorraine

Spécialité doctorale "Informatique"

Calcul neuromorphique pour l'exploration et la catégorisation robuste d'environnement visuel et multimodal dans les systèmes embarqués.

présentée et soutenue publiquement par

Yann BERNARD

le 8 Vendémiaire An CCXXX

Directeur de thèse : **Bernard GIRAU**
Co-encadrant de thèse : **Nicolas HUEBER**
Co-encadrant de thèse : **Pierre RAYMOND**

Jury

M. Alan Turing,	Professeur	Examinateur
Mme. Margaret Hamilton,	Professeur	Rapporteur
M. Emil L. Post,	Professeur	Examinateur
M. Ken Thompson,	Professeur	Examinateur

Table des matières

Table des matières iii

Liste des figures v

Liste des tableaux vii

1 Détection de Nouveauté 1
- 1.1 Introduction . . . 2
- 1.2 Modèles Neuronaux . . . 2
- 1.3 Traitements des images . . . 3
- 1.4 Détection de nouveauté . . . 8
- 1.5 Expérimentations sur la base CDNET14 . . . 10
- 1.6 Conclusion . . . 10
- 1.7 Références . . . 10

Liste des figures

1.1 Lac de Nino 4
1.2 Représentation d'une image 5
1.3 Compression et décompression d'image 6
1.4 Représentation d'une image 7
1.5 Détection de nouveauté avec QV 8
1.6 Détection de nouveauté avec topologie 8

Liste des tableaux

Chapitre 1

Détection de Nouveauté

« *D'abord, j'observe les êtres humains car je les aime bien. J'enregistre dans ma tête tout ce que j'ai remarqué, et ensuite, avec les souvenirs de ce que j'ai vu, je dessine.* »

Hayao Miyazaki

Sommaire

1.1 Introduction 2
1.2 Modèles Neuronaux 2
1.2.1 Cartes Auto-Organisatrices 2
1.2.2 Gaz Neuronaux en Croissance 2
1.3 Traitements des images 3
1.3.1 Apprentissage et reconstruction 3
1.3.2 La gestion des canaux (couleurs) 6
1.4 Détection de nouveauté 8
1.4.1 Détection avec quantification vectorielle 8
1.4.2 Détection avec distance neurale 8
1.4.3 Considérations pour la combinaison 9
1.5 Expérimentations sur la base CDNET14 10
1.5.1 Evaluation de la qualité de reconstruction 10
1.5.2 Evaluation de la détection de nouveauté 10
1.6 Conclusion 10
1.7 Références 10

1.1 Introduction

1.2 Modèles Neuronaux

1.2.1 Cartes Auto-Organisatrices

1.2.2 Gaz Neuronaux en Croissance

1.3 Traitements des images

Il existe de nombreuses possibilités différentes pour apprendre des images avec la quantification vectorielle, dû au grand nombre de façons de découper une image en vecteurs d'entrée. Pour présenter les alternatives que l'on a notre disposition pour apprendre des images, on peut considérer deux extrême : apprendre l'image en entier ou apprendre chaque pixel individuellement. Le premier cas peut sembler absurde dans le cas d'une seule image (une base de données d'un seul élément), mais peut présenter un intérêt lorsque l'on considère une suite d'images par exemple. Dans cet exemple, l'environnement appris serait défini par l'entièreté de ce que voit le capteur et tous les changements, où qu'ils soient dans l'image, auraient de l'importance.

Le second extrême est l'apprentissage au niveau du pixel. Dans cette approche on prend chaque pixel individuel comme un vecteur d'entrée, ce qui rendrait l'espace d'entrée unidimensionnel pour une image en niveau de gris (ou tridimensionnel si l'image est en couleur, plus de détails dans la section 1.3.2). Sur un plan conceptuel, cette option considère qu'une image est définie uniquement par les couleurs ou luminosités présentes, peu importe leur positions dans celle-ci. Ce qui peut être utile dans certains cas particuliers, mais n'est pas adapté à ce que l'on souhaite faire dans cette thèse.

Il existe un très grand nombre d'autres découpages possibles entre ces deux extrêmes, chacun représentant une certaine façon de considérer une image, et par conséquent la différence ou similarité entre deux images. C'est à dire que les différences qui vont apparaître en comparant deux images seront différentes et dépendront de la façon dont celles-ci seront représentées.

Il est important donc de considérer le contexte de nos travaux pour définir la façon de représenter une image, notre application cible étant la detection de nouveauté. Dans ce contexte, nous considérons qu'une image est une combinaison de nombreux éléments plus petits. Par exemple, une photographie d'un lac de montagne 1.1 peut être décrite comme étant la combinaison d'un élément de lac (avec sa couleur, bleu sombre et sa texture uniforme), d'un élément de plaine herbeuse (verte et uniforme), d'éléments rocailleux qui sont gris et soit uniformes (dans le premier plan) soit plus contrastés en se combinant avec la verdure de la végétation (dans les bords de l'image), et ainsi de suite. Ce genre de découpage "sémantique" de l'image est ce qui permet la détection de nouveauté de fonctionner, car dans notre cas la nouveauté est par définition ce que n'est pas déjà dans l'image et donc ne faisant pas partie de ces éléments au sens large. Pour regrouper les parties d'une images appartenant au même élément, il est nécessaire d'avoir une information mise en contexte (un pixel seul ne suffit généralement pas à savoir à quel élément il appartient dans l'image), ce qui implique que les pixels doivent être pris dans leurs environnements locaux pour conserver l'information de voisinage comme la texture par exemple. Nous avons donc choisi de découper l'image en plus petites images à la façon d'une mosaique, que nous appellerons des imagettes. Ces imagettes (d'une taille arbitraire en hauteur et largeur) conservent l'environnement local tout en étant suffisament petites pour être précises dans l'espace (une imagette ne représentant qu'une partie d'un élément et non pas regroupant plusieurs éléments, ce qui réduirait sa capacité de généralisation), et permettre d'avoir une base d'apprentissage assez étoffée pour tirer parti des propriétés de nos modèles de quantification vectorielle et de leur topologie. La partie pratique de l'apprentissage d'une image, sa représentation et sa reconstruction sont abordés dans la section suivante 1.3.1.

1.3.1 Apprentissage et reconstruction

Apprentissage

La première étape nécessaire à l'apprentissage est la constitution de la base d'apprentissage à partir de l'image. L'utilisation de modèles de quantification vectorielle établissent la première contrainte pour le découpage : les imagettes doivent être d'une taille fixe. En effet il est nécessaire pour les SOM comme pour les GNG et leurs variantes que tous les vecteurs d'entrées soient de la même longueur pour que le calcul de distance avec les neurones fonctione, qu'ils représentent

FIGURE 1.1 – Exemple d'image comportant plusieurs éléments notables tels qu'un lac (bleu sombre et uniforme), une plaine herbeuse (verte et uniforme), d'éléments rocailleux qui sont gris et soit uniformes (dans le premier plan) soit plus contrastés en se combinant avec la verdure de la végétation (dans les bords de l'image), et ainsi de suite.

correctement les entrées qui leurs sont attachées.

Nous avons également choisi de limiter nos tailles d'imagettes à des valeurs carrées, pour des raisons de simplicité. Il est possible que des imagettes plus larges que hautes ou plus hautes que larges représentent mieux les éléments de l'image que l'on apprend. Cependant ce serait une préférence spécifique à chaque image et peu généralisable, car si on effectue par exemple une rotation de 90° de l'image, la préférence s'inversera. L'inversement des tailles dans les imagettes carrées ne changeant rien, elles sont pour leur part insensibles aux rotations discrètes de l'image (par pas de 90°).

Dans la version classique, le découpage de l'image se fait en mosaique, sans superpositions entre les imagettes. C'est à dire que chaque pixel n'appartient qu'à une seule imagette. Le processus est montré sur la figure **??** Si les dimensions de l'images ne sont pas un multiple de la taille des imagettes, les pixels en trop sont simplement rognés par la droite et par le bas, car en général ils ne contiennent pas beaucoup d'informations.

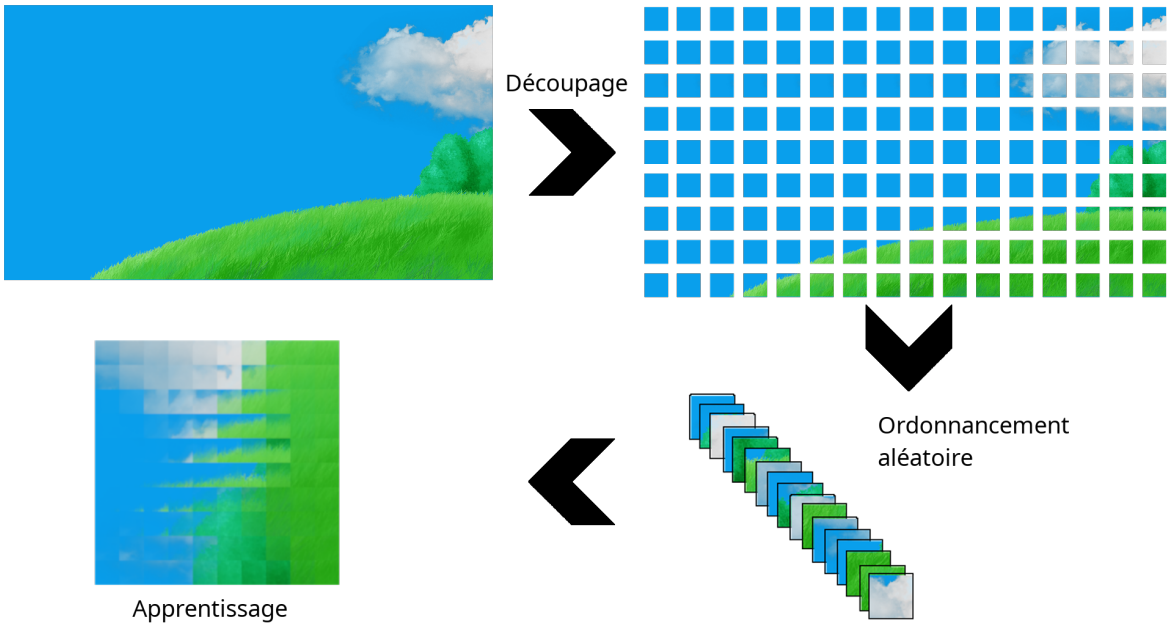

FIGURE 1.2 – Illustration du processus de représentation et d'apprentissage d'une image par une SOM.

Reconstruction

Une fois l'apprentissage terminé, il y a deux résultats. Le premier est le modèle entrainé avec les différents poids des neurones codant une imagette représentante du cluster d'imagettes associé à ce neurone. Le second, facultatif, est la liste pour chaque imagette de l'image l'indexe du neurone le plus proche de celle-ci, qui pourra être utilisée pour la reconstruction de l'image d'apprentissage.

Pour reconstruire une image à partir de la liste des indexes de BMU, il suffit de remplacer chaque index par le vecteur prototype du neurone auquel il correspond. Ces vecteurs prototypes devront être réassemblés en imagettes (dans un tableau à 2 dimensions à la place d'un vecteur à une dimension), et placées à la bonne position pour reformer l'image.

Il est aussi possible de reconstruire une image qui n'a pas été apprise. Pour cela il faut créer la liste d'indexes de neurones des imagettes de la nouvelle image, et de reconstruire ensuite l'image par le même procédé que montré précédemment. Il faut noter que l'image que l'on souhaite reconstituer doit être proche de l'image apprise pour avoir un résultat correct.

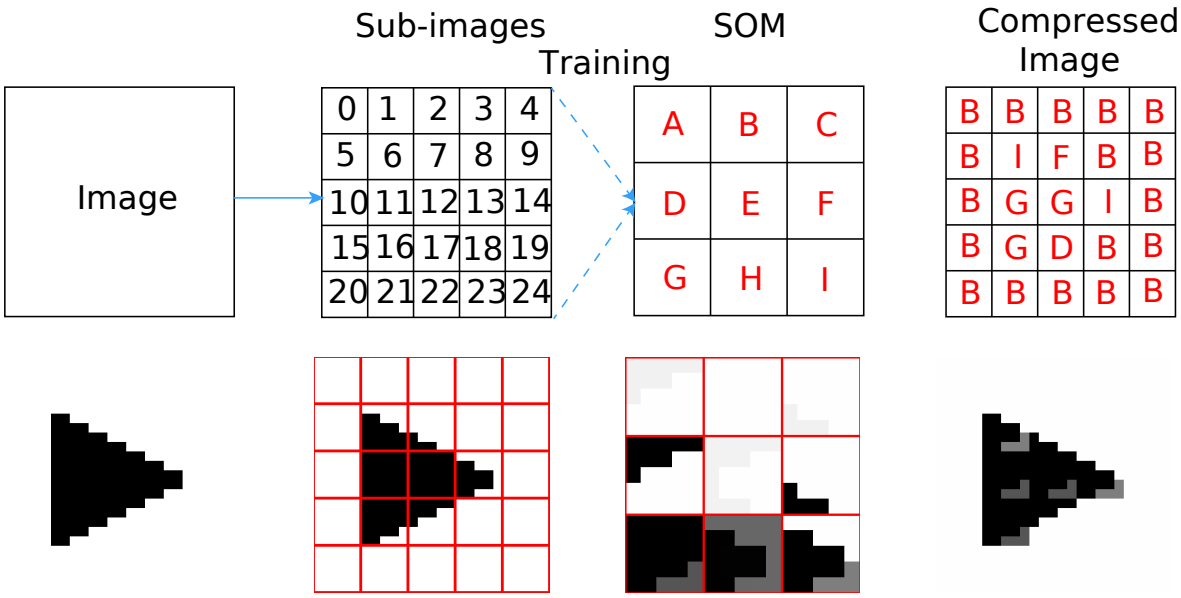

FIGURE 1.3 – Schéma simplifié du processus de compression et de reconstruction d'une image, avec ici seulement 9 neurones et 25 imagettes.

1.3.2 La gestion des canaux (couleurs)

Nous avons jusque là vu comment apprendre une image ayant une seule valeur par pixel (soit des images en nuances de gris). Cependant la majorité des dispositifs de capture actuels fournissent des images en couleur, c'est à dire trois canaux. Nous allons voir dans cette section comment transposer l'apprentissage, la compression et la reconstruction à des images à un nombre arbitraire de canaux par pixels.

Une approche possible serait de séparer l'image par composante (une image RVB par exemple donnerait trois images en niveau de gris, une R, une V et une B). Deux options s'offrent ensuite à nous. Soit utiliser une seule SOM pour apprendre toutes les images ainsi extraites en espérant que les différentes formes présentes dans chaque composante soient assez similaires entre elles pour ne pas trop surcharger la SOM de données (car on vient de multiplier la taille de notre base d'apprentissage par le nombre de canaux), et que ces données restent assez cohérentes dans l'espace d'entrée pour que la réduction dimensionnelle se fasse correctement. Soit utiliser une SOM par composante pour apprendre chaque canal séparément, et de regrouper ensuite les différents canaux reconstitués en une image couleur.

Cependant ces deux approches ont un défaut majeur pour la compression d'images (ainsi que la détection de nouveauté par conséquent), c'est la création d'aberrations chromatiques dans l'image reconstituée. En effet, les canaux étant appris séparément avant d'être recombinés, la reconstruction peut donner pour certains pixels des teintes de couleurs qui n'existaient pas dans l'image de base, et très saillants visuellement. Par exemple un pixel blanc dans l'image d'entrée (avec une forte composante R, V et B), lorsque reconstitué par la SOM peut être bien reconstitué dans deux composantes (V et B par exemple), et mal reconstitué dans la troisième (R) avec une valeur beaucoup plus faible que dans l'image de base (cela arrive car notre calcul de distance minimise). Par conséquent ce pixel aura une couleur turquoise dans l'image reconstituée à la place de blanc. Ce genre d'erreur est particulièrement mauvaise, car le résultat de la reconstruction minimise bien de façon optimale la distance avec l'image de base, et ce n'est que visuellement que les changements de teintes sont apparents et dégradent qualitativement l'image plus que ce que l'erreur mesurée ne laisse penser.

Une meilleure approche consiste à inclure tous les canaux directements dans les imagettes. Chaque imagette devient donc une imagette en couleur, et sa taille augmente donc en conséquence. Une imagette de 10 par 10 pixels par exemple qui donnerait un vecteur prototype de taille

100, passe à 300 avec les trois couleurs. L'apprentissage et la reconstruction se déroulent de la même manière que dans la SOM classique. Il faut aussi noter que l'ordre n'a pas d'importance dans les vecteurs prototypes, on peut arranger les valeurs en RGBRGBRGB tout comme RRRGGGBBB sans que cela ne change le résultat, le calcul de distance euclidienne étant indépendant de l'ordre des coordonnées.

(a) Couleurs fusionnées (b) Couleurs séparées

FIGURE 1.4 – Comparaison entre une image avec des couleurs fusionnées et la même image avec des couleurs séparées qui présente des artefacts visuels.

1.4 Détection de nouveauté

Nous avons, à partir des différents principes, objectifs et fonctionnements définis dans les sections précédentes, développé deux processus de détection de nouveauté complémentaires. Nous présentons ces deux méthodes dans cette partie. La première est basée sur la propriété de quantification vectorielle de nos modèles pour trouver de la nouveauté précisément dans des images. La seconde est basée sur la topologie de nos modèles pour détecter de la nouveauté, qui donne des résultats moins précis, mais aussi moins bruités.

1.4.1 Détection avec quantification vectorielle

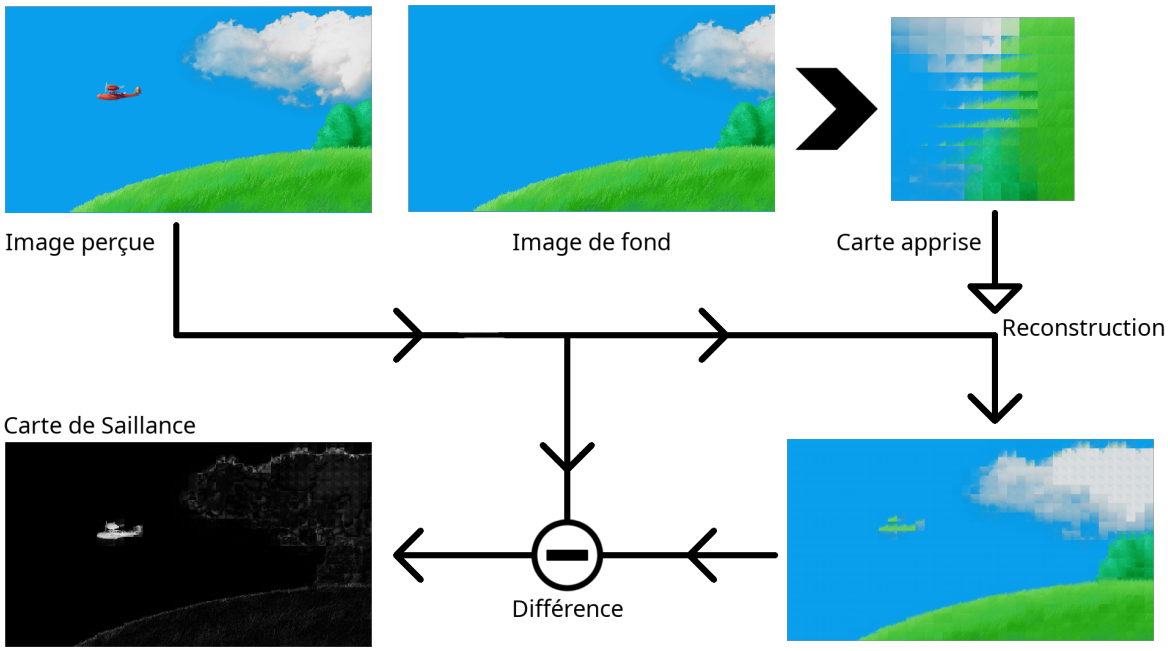

FIGURE 1.5 – Fonctionnement schématique de la détection de nouveauté avec quantification vectorielle.

1.4.2 Détection avec distance neurale

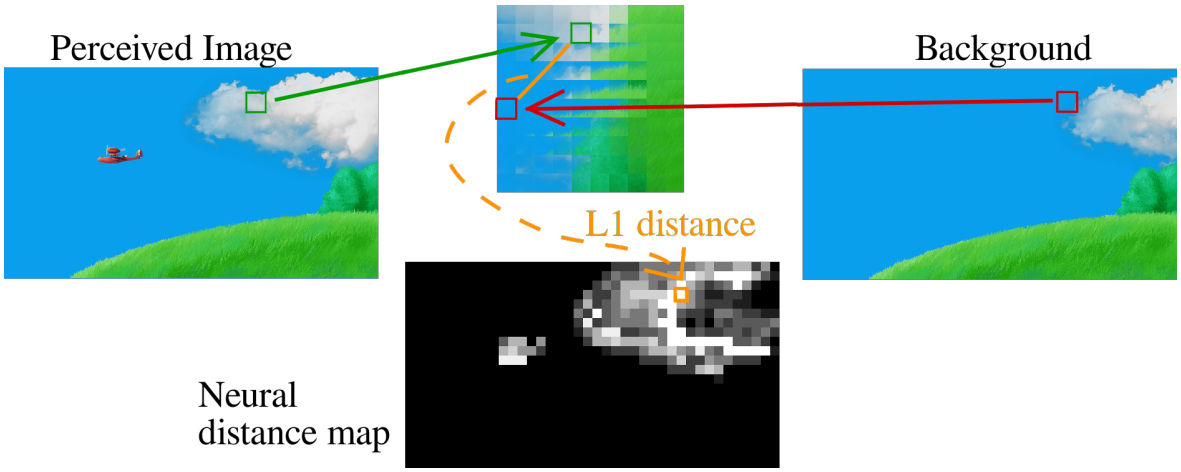

FIGURE 1.6 – Fonctionnement schématique de la détection de nouveauté avec topologie.

1.4.3 Considérations pour la combinaison

Une fois les deux cartes de saillance générées, il est nécessaire de les combiner pour n'avoir qu'un seul résultat qui représentera la sortie de notre système. Il existe un très grand nombre de façons de le faire cette combinaison, et il existe dans la littérature des modèles qui se basent sur une bonne combinaison de différentes cartes de saillance pour obtenir de meilleurs résultats . Dans notre cas, nous avons avons préféré utiliser une combinaison simple de nos deux cartes de saillance. C'est à dire qu'elle n'utilise pas de paramètres (pour ne pas ajouter encore une variable à optimiser). Nous souhaitons aussi bénéficier de la complémentarité des deux cartes de saillances que l'on a. Par conséquent la solution la plus simple est de multiplier les deux cartes ensemble. Ainsi, sera considéré comme nouveauté dans la carte de sortie, ce qui apparaît comme nouveauté en même temps dans les deux cartes de saillance (car $petit \times petit = petit$, $grand \times petit = petit$ et seulement $grand \times grand = grand$). De cette façon, le bruit qui est présent dans la carte résultant de la quantification vectorielle et qui n'est pas présent dans la carte topologique disparaît de la carte finale. Il en va de même pour les mouvements qui ne sont pas des nouveautés qui apparaissent dans la carte topologique, mais pas dans la carte de quantification vectorielle.

Un problème qui peut apparaître dans ce cas est la trop petite valeur du résultat et le déséquilibre d'impact de nos deux cartes, car nos cartes de saillance sont toutes les deux définies entre 0 et 1. Il est possible que des situations arrivent lors desquelles les deux cartes ont un impact disproportionnel sur le résultat (par exemple si l'objet à détecter sur une carte a une valeur de 0.2 sur une carte et 0.8 sur l'autre, alors la seconde aura plus d'impact sur les valeurs de la sortie finale). De même, en multipliant deux nombres compris entre 0 et 1, le résultat sera forcément inférieur à chacun des deux nombres. Ainsi, on a aussi un effet qui réduit toutes les valeurs de la carte de saillance finale, ce qui peut poser problème si l'on n'y fait pas attention. La solution que nous avons choisi à ces des problèmes, est de re-normaliser les deux cartes de saillances avant de les multiplier. C'est à dire que l'on rééchelonne l'ensemble de la carte en mettant la valeur maximum de la carte à 1 et le minimum à 0 et d'étaler les valeurs intermédiaires entre les deux pour conserver le même espacement relatif entre elles. Cela a pour effet d'éviter une trop grande disproportion d'impact entre les deux cartes (mais ne résoud pas complètement le problème, car on re-normalise avec le maximum, et non avec la possible valeur de la nouveauté detectée), et permet d'avoir un résultat avec des valeurs généralement plus hautes. Cependant, cela vient aussi avec des désavantages, comme par exemple le fait que si il n'y a pas de signal dans l'entrée, le maximum des cartes de saillance sera quand même 1 et on pourrait observer des signaux positifs dans la sortie alors que l'entrée et les cartes de saillances n'en montrent pas. En pratique, cela est peu fréquent car la valeur maximum dans un cas où il n'y a pas de signal en entrée vient du bruit, et est donc décorellée entre les deux cartes. De plus, la taille du signal d'entrée compte, et il est peu probable que du bruit seul puisse créer une zone de signal assez large pour être confondu avec une vraie nouveauté.

1.5 Expérimentations sur la base CDNET14

1.5.1 Evaluation de la qualité de reconstruction

Métriques utilisées

Préparation des données

Paramétrages des modèles

Résultats expérimentaux

Interprétations

1.5.2 Evaluation de la détection de nouveauté

Métriques utilisées

Préparation des données

Paramétrages des modèles

Résultats expérimentaux

Interprétations

1.6 Conclusion

1.7 Références