

Cours ENSTA ParisTech - IN104

Projet informatique - Réseau de Kohonen

David Filliat - ENSTA ParisTech

1 Description du projet

Dans ce projet, nous allons mettre en oeuvre une méthode de projection non-linéaire permettant d'étudier et de visualiser la distribution de données de grande dimension. Basés sur une phase d'apprentissage, les *réseaux de Kohonen* permettent de projeter ces données sur un plan en conservant leurs propriétés topologiques. Nous appliquerons cette méthode à un problème de classification d'images pour rechercher les caractéristiques des images les plus utiles pour leur catégorisation.

1.1 Problème

Nous disposons d'un grand nombre d'images appartenant à différentes catégories et nous souhaitons créer un algorithme permettant de trouver automatiquement la catégorie d'une image. Pour cela, nous allons chercher un descripteur qui sera calculé à partir d'une image et qui résumera l'information contenue dans l'image. Idéalement, ce descripteur devra être relativement constant pour toutes les images d'une même catégorie et très différent pour des images de catégories différentes afin de permettre une catégorisation simple.

Ces descripteurs seront souvent de grande dimension et il n'est pas aisé de les visualiser afin de voir si ils sont bien discriminant pour les catégories. Nous allons donc essayer d'en faire une représentation graphique en 2 dimensions qui respecte leur distribution spatiale et montre clairement si des images ont des descripteurs proches ou non.

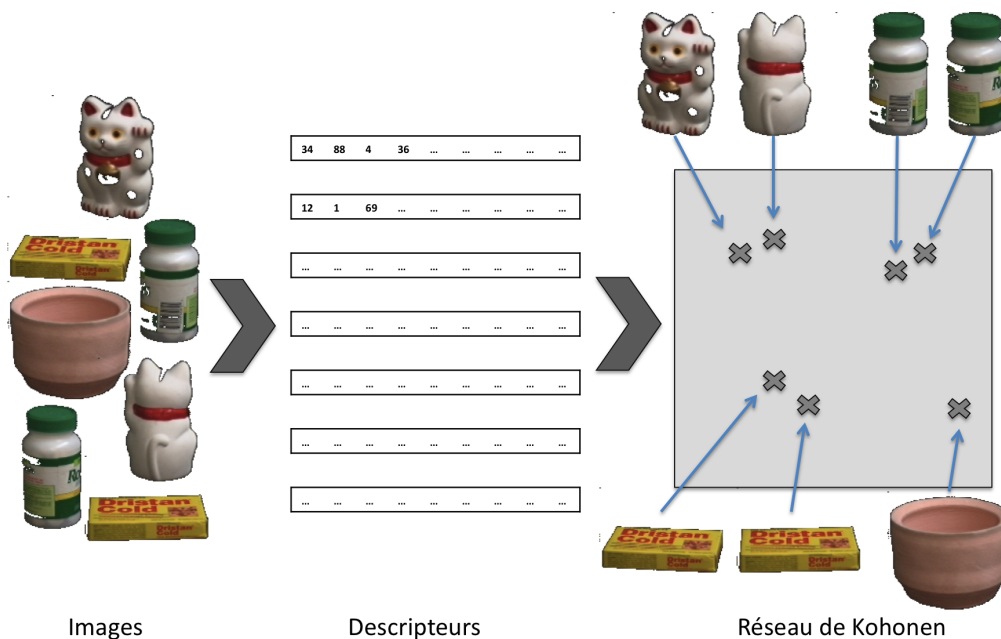


FIGURE 1 – Illustration du problème

Le problème de projection de ces descripteurs de grande dimension dans un espace de dimension 2 peut être abordé par différentes techniques, notamment par l'*analyse en composantes principales*¹ ou ses

1. http://fr.wikipedia.org/wiki/Analyse_en_composantes_principales

variantes. Dans le cadre de ce projet, nous allons mettre en oeuvre un type de réseau de neurones particulier, le réseau de Kohonen² appelé également *carte auto-organisatrice*. Après une phase d'apprentissage, ces réseaux sont capables de réaliser cette projection de manière pertinente pour le problème particulier sur lequel ils ont été entraînés. Ils réalisent de plus une projection non linéaire souvent bien adaptée aux problèmes pratiques.

1.2 Le réseau de Kohonen

1.2.1 Structure

Le réseau de Kohonen est généralement constitué d'une grille régulière de neurones en deux dimensions. Chacun des neurones j de cette grille est également connecté à des neurones d'entrée i , chaque connexion ayant un poids particulier $w_{i,j}$.

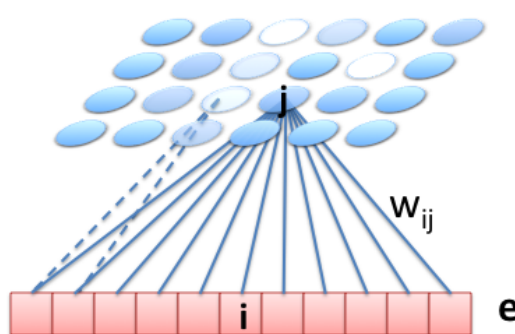


FIGURE 2 – Le réseau de Kohonen

On définit une *activation* pour chaque neurone de la grille, qui est calculée comme la distance entre les valeurs des neurones d'entrée et les valeurs des poids associés :

$$a(j) = \sqrt{\sum_i (e(i) - w_{i,j})^2}$$

Pour une entrée donnée, le neurone ayant la plus faible activation est déclaré vainqueur et correspond à la position de cette entrée dans la grille. Notons que l'ensemble des poids $w_{i,j}$ pour un neurone j donné correspond à un vecteur d'entrée particulier, pour lequel l'activation est nulle. Ce vecteur d'entrée est la position associée au neurone de sortie j dans l'espace d'entrée.

Après une phase d'apprentissage, tous les neurones de la grille seront donc associés à des valeurs particulières des entrées (via leur poids de connexions). La méthode d'apprentissage détaillée dans la section suivante permet de répartir ces valeurs dans la grille de façon à refléter aux mieux en 2 dimensions la répartition des données d'entrée dans leur espace d'origine.

1.2.2 Apprentissage

Le réseau de Kohonen est un réseau *auto-organisateur* dont l'apprentissage est *non supervisé*. L'apprentissage se réalise simplement en lui présentant les données d'entrée dans un ordre aléatoire, sans autre information et en appliquant une procédure de modification des poids.

Pour une entrée donnée, l'apprentissage se réalise en modifiant les poids des connexions pour le neurone vainqueur et ses voisins suivant la formule :

$$w_{i,j} = w_{i,j} + \alpha(t) \cdot \gamma(t, j) \cdot (e(i) - w_{i,j})$$

2. http://fr.wikipedia.org/wiki/Carte_auto_adaptative

Cette équation a pour effet de rapprocher les poids des neurones proches du neurone vainqueur des valeurs des entrées. Le coefficient $\alpha(t)$ est le coefficient d'apprentissage compris entre 0 et 1 et qui doit décroître au cours du temps pour permettre à l'apprentissage de converger.

Le coefficient $\gamma(t, j)$ permet de propager l'apprentissage aux neurones voisins du vainqueur. Il doit être maximal pour le neurone vainqueur et diminuer lorsque l'on s'éloigne de ce neurone. Le nombre de voisins touchés doit lui aussi décroître au cours du temps afin de permettre à l'apprentissage de converger. La fonction la plus utilisée est la gaussienne avec une variance σ diminuant au cours du temps :

$$\gamma(t, j) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-d^2}{2\sigma^2}\right)$$

où d est la distance entre le neurone gagnant et le neurone j .

1.3 Les descripteurs d'image

Un descripteur est un vecteur de valeurs, calculé à partir d'une image, qui résume une information contenue dans l'image (la couleur, les gradients, la texture...). C'est une représentation de moins grande taille que l'image (quelques dizaines de dimensions au lieu de quelques centaines de milliers de pixels).

Dans le cadre de ce projet, il s'agira de tester différents descripteurs d'image afin de déterminer lesquels permettent de catégoriser correctement un ensemble d'images. Il faut donc chercher des descripteurs qui permettent de grouper les images d'une même catégorie et de séparer clairement les images de catégories différentes. Pour cela, nous utiliserons une base de donnée d'images, dont nous calculerons les descripteurs. Nous utiliserons ensuite ces descripteurs pour entraîner un réseau de Kohonen, puis nous afficherons la position de quelques images dans le réseau pour étudier leur répartition.

Nous testerons au moins deux descripteurs : un histogramme de couleurs et une mesure de la texture.

1.3.1 Descripteur de couleur

Ce descripteur va caractériser une image par la répartition des couleurs des différents pixels. Chaque pixel est généralement représenté par trois valeurs donnant son niveau sur trois couleurs primaires : rouge, vert, bleu. Pour simplifier le descripteur, nous utiliserons un autre espace de représentation, l'espace HSV. La teinte H (Hue) est la couleur pure (donnant la position de la couleur sur l'arc en ciel, de 0 à 360), S (Saturation) est le niveau de couleur (il vaut 0 pour du noir ou du blanc et est maximum pour une couleur pure), V (Value) est le niveau de gris (voir http://en.wikipedia.org/wiki/HSV_color_space).

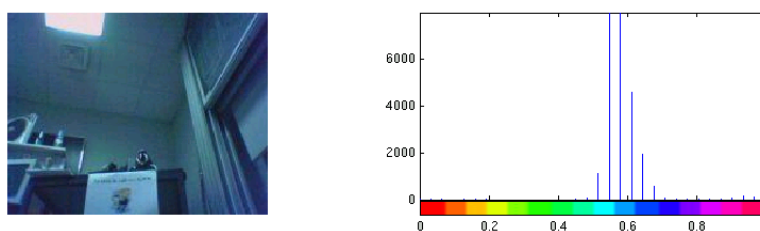


FIGURE 3 – Exemple d'histogramme de teinte (H).

Nous prendrons comme descripteur l'histogramme des valeurs de H découpé selon un nombre de cases paramétrables.

1.3.2 Descripteur de texture

Pour caractériser la texture, nous utiliserons la variance locale du niveau de gris des pixels (le V de HSV). Pour chaque pixel, nous calculerons donc la variance de l'ensemble des pixels contenus dans un voisinage de taille $n \times n$ autour du pixel ($n = 3, 5, 7, \dots$). Nous prendrons ensuite l'histogramme de ces variances comme descripteur de l'image.



FIGURE 4 – Exemple d'histogramme de texture

1.4 Cahier des charges

Le projet à réaliser devra permettre de lire une série d'images dans différentes catégories connues et de les traiter en calculant un descripteur pour chaque image. Plusieurs descripteurs devront être testés.

Ces descripteurs devront ensuite être utilisés pour entraîner un réseau de Kohonen de taille paramétrable.

Enfin le réseau de Kohonen devra permettre de visualiser la position d'un ensemble de descripteurs afin d'analyser leur pertinence pour la catégorisation des images : il s'agira de vérifier que les images d'une même catégorie sont proches et que les images de catégories différentes sont séparées.

2 Organisation du travail

2.1 Conception

La première étape est la conception de votre programme. Répartissez les différentes fonctionnalités nécessaires dans différents modules. Définissez les fonctions dont vous aurez besoin dans chaque module, définissez les structures de données.

Vous aurez à votre disposition une bibliothèque contenant un certain nombre de fonctions que vous pourrez ré-utiliser. Ces fonctions portent essentiellement sur l'affichage graphique et sur la lecture/écriture des images. La bibliothèque est disponible à l'adresse : <http://uei.ensta.fr/filliat/Courses>. La description des fonctions graphiques est disponible sur la page :

<http://moscova.inria.fr/~peskine/enseignement/iup-2002/graphics-doc.html>

La bibliothèque contient de plus les fonctions :

```
- void read_JPEG_file (char * filename, int *width, int *height, unsigned char
**R_data, unsigned char **G_data, unsigned char **B_data);
```

Cette fonction lit une image dans un fichier au format jpeg et renvoie 3 tableaux des valeurs RGB des pixels. La fonction renvoie également la taille de l'image (width et height). Chacun des tableaux contient width*height valeurs, les pixels sont rangés ligne par ligne en partant du haut à gauche de l'image.

```
- void RGBtoHSV( float r, float g, float b, float *h, float *s, float *v );
```

Cette fonction convertit le triplet r,g,b de couleurs d'un pixel en un triplet au format h,s,v.

Pour utiliser cette bibliothèque, vous aurez besoin de lier les bibliothèques dynamiques jpeg, X11 et math (options de l'éditeur de lien : -L/usr/X11R6/lib -lX11 -lm -ljpeg)

Enfin un fichier kohonen_display.cpp que vous pourrez modifier à volonté contient les fonctions :

```
- void affiche_activation (float * act, int cx, int cy);
```

Affiche une grille de taille cx*cy, le niveau de gris de chaque élément étant lu dans le tableau act.

```
- void affiche_input (float *in, int taille);
```

Affiche en bas de la fenêtre une ligne de cellules dont le niveau de gris est lu dans in.

```
- void affiche_label (char *label, int px, int py);
```

Affiche le texte de label à la position px,py.

Ces fonctions permettent de réaliser un affichage graphique très simple en utilisant la bibliothèque graphique mentionnée précédemment.

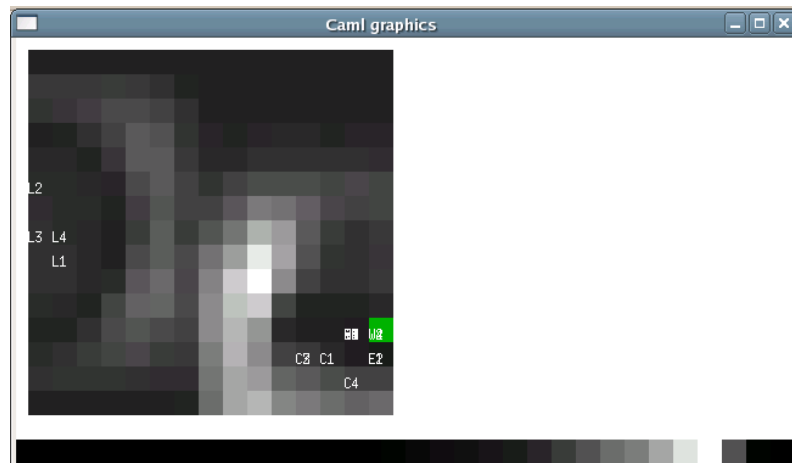


FIGURE 5 – Exemple d'utilisation de kohonen_display.cpp

La liste des images que vous utiliserez pour l'apprentissage du réseau de Kohonen vous sera fournie dans un fichier texte `Training_files.txt` que vous utiliserez en entrée de votre programme. Il contient des lignes de la forme :

```
/home/uei/filliat/IN104/obj14__0.jpg
/home/uei/filliat/IN104/obj14__10.jpg
...
```

Après apprentissage, vous afficherez la position sur la carte de Kohonen d'un certain nombre d'images dont la liste est fournie dans `Test_files.txt`. Chaque ligne contient un identifiant de 2 lettres puis le nom de l'image, séparés par un espace :

```
O14 /home/uei/filliat/IN104/obj14__155.jpg
O14 /home/uei/filliat/IN104/obj14__65.jpg
...
```

2.2 Développement

Développez les modules que vous avez identifié précédemment en commençant par les modules de plus bas niveau (qui n'utilisent aucun autre module), puis testez-les avant de développer les suivants. Privilégiez l'utilisation du debugger pour la mise au point. Testez les modules au fur et à mesure de leur écriture afin d'éviter la propagation incontrôlable de bugs.

Créez un jeu de données partiel pour pouvoir faire des tests rapidement. Créez des images spécifiques pour vérifier le fonctionnement de vos modules si nécessaire.

N'hésitez pas à modifier les fonctions d'affichage fournies pour améliorer leur aspect ou afficher d'autres informations.

2.3 Test et validation

Utilisez la base de données complète pour l'apprentissage, affichez les labels des images de test.

Manipulez les différents paramètres de l'algorithme d'apprentissage (α , γ , taille de la carte, taille des histogrammes des descripteurs...) jusqu'à obtenir un résultat cohérent et répétable. Qu'en concluez-vous sur la pertinence des différents descripteurs pour la tâche considérée ?

Proposez et évaluez d'autres descripteurs, par exemple les auto-corélogrammes, ou des histogrammes d'orientation des gradients (ask google).