# Rapport - Reconnaissance des Formes

Xudong Zhang, Ye Liu

## 0. Introduction

Dans ce projet, nous avons cherché sur la reconnaissance des chiffres manuscrits. Premièrement, nous extrayons les caractéristiques d'image dont les méthodes sont bien définies dans le rapport. Puis, on vérifie la qualité des caractéristiques par la méthode des fenêtres de Parzen et le K-plus-proche-voisin. Sur cette base, on teste deux méthodes d'apprentissage plus modernes--le SVM et le méthode BP (Back Propagation) réseaux de neurones. Naturellement, on considère le problème de la sélection des caractéristiques après. C'est-à-dire comment choisir les meilleur caractéristiques pour le système. On teste l'algorithme d'sélection avec les réseaux de neurones et discute les résultats.

## 1. Extraction de caractéristiques (173 caractéristiques)

### 1.1 Segmentation

A cause des habitudes différentes, les tailles de caractères variantes. Ça va changer beaucoup de caractéristiques, et ils ne sont pas compatible l'un l 'autre. Pour obtenir un critère compatible pour tous les scripts, nous devons découper les caractères. Nous découpons les parties blancs, et nous laissent un rectangle alors les caractères se plaquent contre les rectangles.

Parce que les dimensions des rectangles varient après le découpage, nous devons normaliser the caractéristiques. Alors nous instituons la différence entre le maximum et les minimums d'une certainecaractéristique comme une unité. Soit c une valeur d'une certaine caractéristique normalisé, alors il s'écrie:

$$c = \frac{c - \min}{\max - \min}$$

$\min$ = le minimum de tous valeurs de cette caractéristique

$\max$ = lemaximum de tous valeurs de cette caractéristique

### 1.2 Le centre de gravité et ce moment (14 caractéristiques)

Le centre de gravité de chaque numéro peut illustrer quelques particularités sur une vue global. Par exemple, le centre de gravité numéro « 1 », « 0 » ou « 8 » est bien dans le centre euclidien, mais le centre de gravité numéro « 6 » se situe dans la partie

plus haut et pour « 9 » il se situe dans la partie plus bas. Alors nous choisissons un vecteur de dimension deux, qui est la coordonnée de centre de gravité. Pour normalisation, nous transformons les données à l'échelle. Soit (i,j) les points de données avec le valeur v(i,j) égalent à « 1 » (c'est-à-dire les point noirs) ou « 0 » (les point blancs), alors le centre de gravité s'écrit :

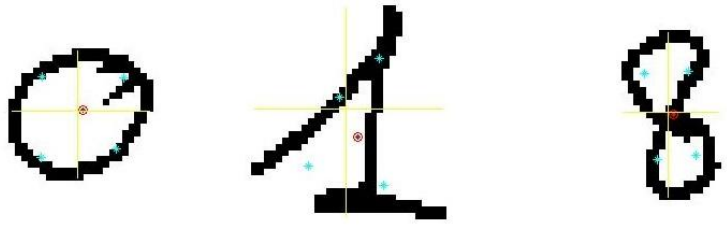
$$g = (g_x, g_y)$$

$$g_x = \frac{\sum_i \sum_j i \cdot v(i,j)}{\sum_i \sum_j v(i,j)}, g_y = \frac{\sum_i \sum_j j \cdot v(i,j)}{\sum_i \sum_j v(i,j)}$$

Et le moment s'écrit:

$$c = \sum_i \sum_j v(i,j) \cdot [(i - g_x)^2 + (j - g_y)^2]^{1/2}$$

Mais le centre globalseulement est brumeux, il ne peut pas distinguer certains échantillons, surtout « 1 », « 0 » et « 8 ». Alors nous divisons l'image en quatre quadrants, puis nous comptons les coordonnées de centre de gravité et les moments pour chaque quadrant. Ce processus fabrique un vecteur de dimension trois pour chaque quadrant. Les résultats pour « 1 », « 0 » et « 8 » sont posé au-dessous. Par conséquent, les valeurs sont bien séparées pour tous les chiffres.



| Moment | « 0 » | « 1 » | « 8 » |
|---|---|---|---|
| Quadrant 1 | 4.0952 | 1.7606 | 4.4547 |
| Quadrant 2 | 3.7069 | 4.1819 | 4.0230 |
| Quadrant 3 | 4.1754 | 6.4775 | 3.6067 |
| Quadrant 4 | 3.2482 | 5.4878 | 4.3812 |

Figure 1. Les centres de gravité et les moments pour « 1 », « 0 » et « 8 ». Les centres de gravité sont de points rouges, et Les centres de gravité pour tous les quadrants sont points bleus. Les moments ont déjà normalisé.

## 1.3 La densité (20 caractéristiques)

La répartition de la densité pour une direction arbitraire est un choix bien adapté à notre cas. Comme exposer dans l'image au-dessous, nous choisissons la direction

perpendiculaire et la direction horizontale. Cette deux directions sont le plus facile de réaliser, et ils ont autant de avantage que les autre directions.

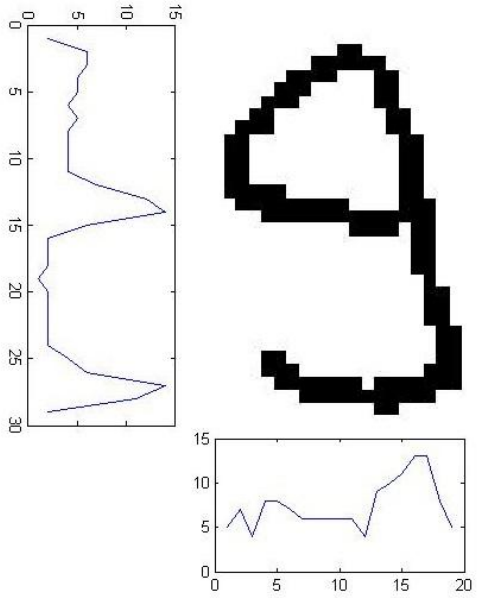



Figure 2. La répartition de la densité par exemple pour « 9 ».

Soit (i,j) les points de données avec le valeur v(i,j), les répartitions de la densité de deux directions s' écrient :

$$\rho_{pj} = \frac{\sum_i v(i,j)}{\sum_i 1}, \rho_{hi} = \frac{\sum_j v(i,j)}{\sum_j 1}$$

Le centre de gravité et la densité révèlent le caractère en vue global. Pour mieux marquer et mieux distinguer les chiffres, nous décidons de choisir quelques data qui indiquent plus de détails.Ces sont la transformation de Fourier et la méthode de Komori.

## 1.4 La transformation de Fourier (16 caractéristiques)

La transformation de Fourier est une extension pour développer une série de Fourier des fonctions périodiques. Le résultat de cette transformation représente l'état des toutes fréquences, qui sont les vitesses de changement des valeurs. L'amplitude de la transformation de Fourier d'une image se pose au-dessus, les valeurs le plus grandes se situent dans les quatre points. Ces valeurs révèlent les parties avec the changement de niveau de gris le moins fort. Comme l'indique la figure (b), l'image expose principalement le couleur. Mais comme l'indique la figure (a), l'image expose principalement ladivision desdifférents niveaux de gris.

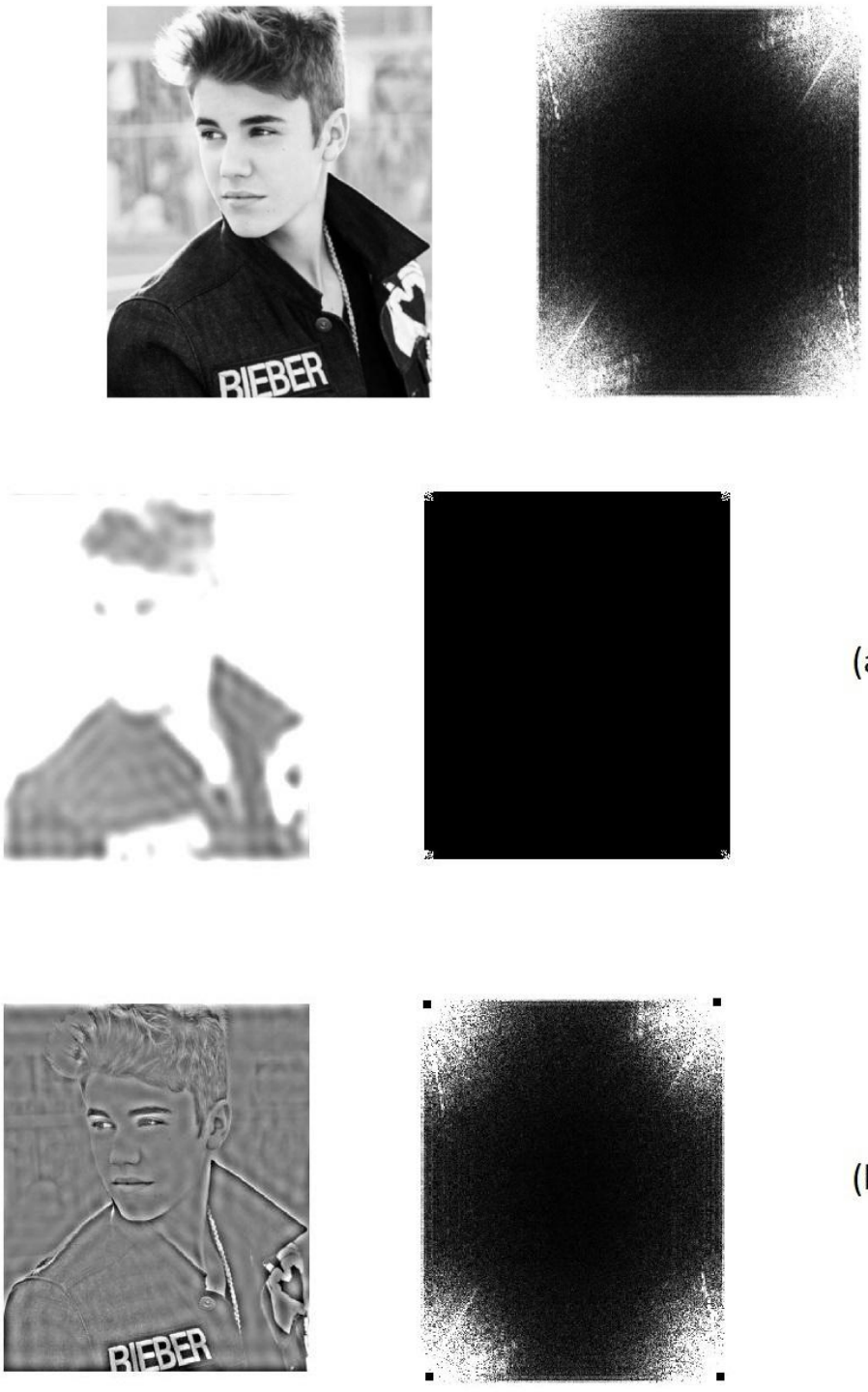



Figure 3. La premi ère ligne est l'image originale et l'image apr ès la transformation. Figure(a) est les images avec les data dans les quatre coins. Figure(b) est les images apr ès nous supprimons les data dans les quatre coins. La partie noir et rectangulaire signifie les data supprim é.

Pour obtenir les informations d'une image, on doit compter la transformation de

Fourier pour deux dimensions. Soit f(x,y) une fonctiondes variables réellesx et y sur un plan, et u et v sont les variables de la fréquence. Alors la transformation s'écrit :

$$\mathrm{F}(u,v) = \int_{-\infty}^{+\infty} f(x,y)\exp[-j2\pi(xu+yv)]dxdy$$

Dans notre programme, nous utilisons la transformation de Fourier deux-dimensionnel discrète. Soit f(x,y) une fonctiondes variables discrètes réelles x et y sur un plande la taille M×N , et u et v sont les variables de la fréquence. Alors la transformation s'écrit :

$$\mathrm{F}(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\exp[-j2\pi(\frac{xu}{M}+\frac{yu}{N})]$$

$$avec, \mathrm{x} = 0,1,...,M-1, \mathrm{y} = 0,1,...,\mathrm{N}-1$$

Nous obtenons les images après la transformation de Fourier, l'illustration plus basse est ces images de chaque numéro. Et nous choisissons les images plus caractéristiques. Avec la observation attentivement, nous découvrions que les données varient le plus c'est les données se trouvent dans les quatre coins.

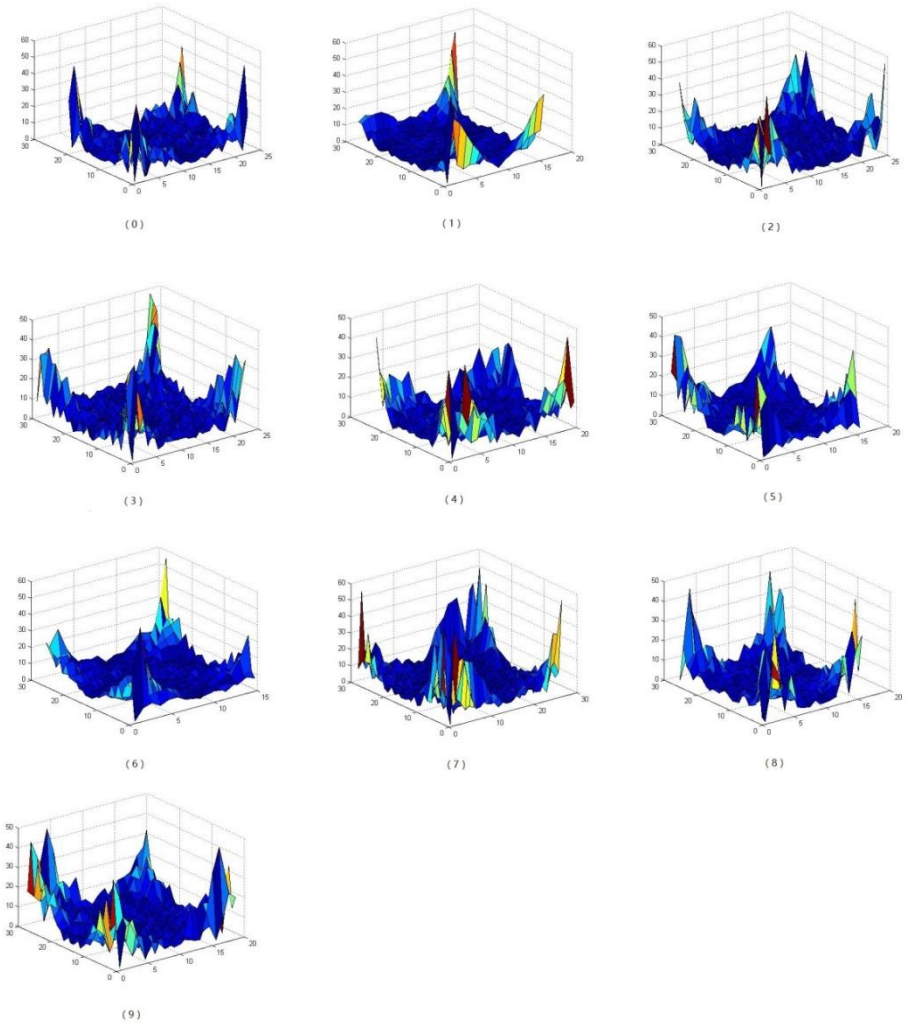



Figure 4. Les images après la transformation de Fourier de chaque chiffre. Les numérosau-dessous de chaque image signifient son chiffre.

Prenons le principe de choisir la caractéristique le plus distinguable possible, nous choisissons quatre points des proportions définies dans chaque coin.Les images obtient par les data des quatre coins s'posent :

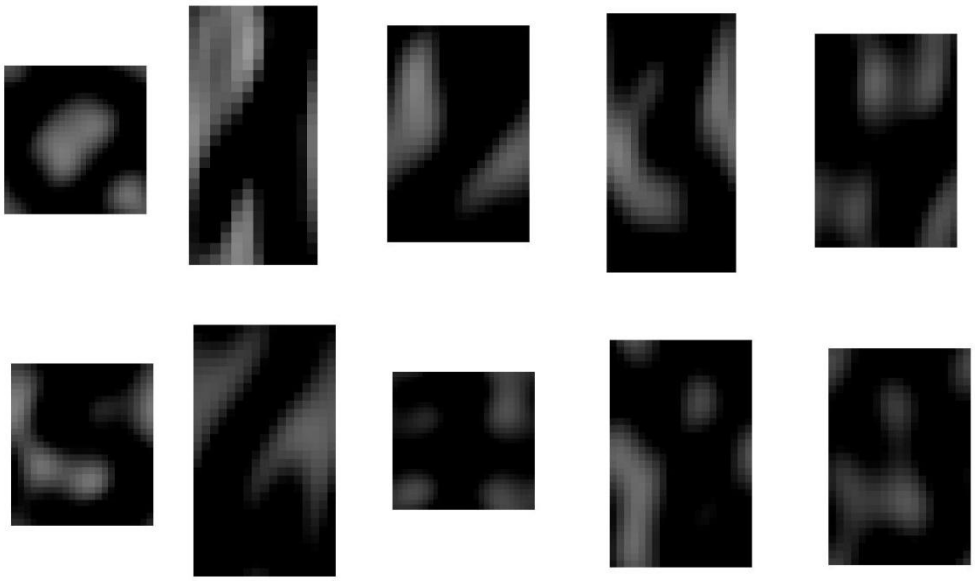


Figure 5. Figure 4. Les images de information des quarte coins après la transformation de Fourier inverse de chaque chiffre.

## 1.5 La Méthode de Komori (123 caractéristiques)

La méthode de komori est une méthode qui peut bien caractériser les différents chiffres. C'est une méthode qui s'intéresse aux concavités. Elle définit 45 symboles d'étiquetage pour 8 directions. Dans notre programme, nous citons les méthodes de Komori.

## 1.6 Conclusion

En résumé, nous choisissons 14 caractéristiques pour les centres de gravité et ses moments. 2 d'entre eux sont la coordonnée du chiffre. 8 d'entre eux sont les quatre coordonnées des quatre quadrants. Et 4 caractéristiques sont les moments des quatre quadrants.

Pour la densité, nous choisissons 20 caractéristiques (10 pour la direction perpendiculaire et 10 pour la direction horizontale).

Après la transformation de Fourier, nous prenons 4 données aux quatre coins. Ça fait 16 caractéristiques.

Enfin, nous choisissons 41 symboles d'étiquetage pour 8 directions. Et nous divisons les images en 3 régions. Ça fait 123 caractéristiques.

En tout, 14+20+16+123=173 caractéristiques. Pour un meilleur résultat, nous supprimons les caractéristiques avec trop de zéro (Si le nombre de numéro non zéro est moins que 80, on supprime cette caractéristique). Finalement il reste 141 caractéristiques.

# 2. Examen

## 2.1 Examen par les fenêtres de Parzon

Le principe de la méthode des fenêtres de Parzon est approximer la distribution de probabilités par une somme de fonctions centrée autour des points d'apprentissage.

Nous avons choisi les fonctions rectangulaires premièrement, mais le résultat est très mauvais avec un taux d'erreur plus de 10% pour quelque chiffre.

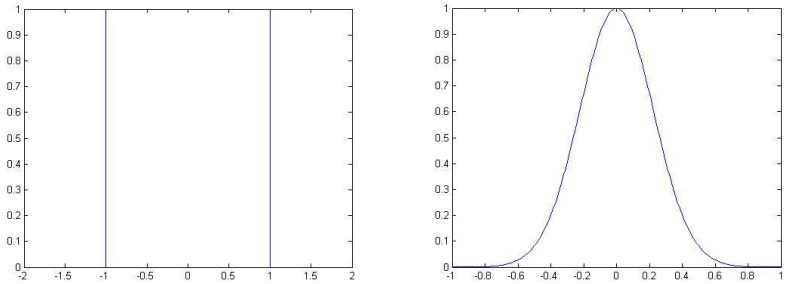


Figure 6. La fonction rectangulaire et la fonction gaussienne

Alors nous choisissons les fonctions gaussiennes, et étudions ses performances. Soit x le vecteur de caractéristiques, la fonction s'écrit :

$$\varphi_n(x_{test}) = \sum_i A \cdot \exp[-C \cdot \left| \overrightarrow{x_{test}} - \overrightarrow{y_{learn,i}} \right|]$$

Le paramètre «A » ne changent pas le résultat, alors nous nous intéressons plus à le paramètre «C ». Les résultats sont indiqués dans les tableaux. Nous découvrons que «C » avec une valeur moyenne s'exprime le résultat meilleur que les « C » avec une valeur grande ou petite. C'est parceque la fonction avec le plus grand C ressemble à la fonction de Dirac, il perd l'information. Et la fonction avec le plus petit C va grandit la influence d'erreur.

Taux d'erreur quand C = 2

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.0215 | 0.0324 | 0.0625 | 0.0512 | 0.0580 | 0.0172 | 0.0517 | 0.173 | 0.116 |

Moyenne =0.0585, écart = 0.0514

Taux d'erreur quand C = 20

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 0.0043 | 0.0301 | 0.0389 | 0.0732 | 0.0349 | 0.0394 | 0.0086 | 0.0560 | 0.059 | 0.076 |

Moyenne= 0.0421, écart = 0.0245

Taux d'erreur quand C = 200

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 0.0064 | 0.0494 | 0.0649 | 0.0948 | 0.072 | 0.0556 | 0.0107 | 0.127 | 0.0679 | 0.0717 |

Moyenne= 0.0621, écart = 0.0357

Pour obtenir le meilleur résultat, nous choisissons :

$$C = 17$$

Taux d'erreur quand C = 17

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 0.002 | 0.030 | 0.0389 | 0.0732 | 0.0349 | 0.03712 | 0.0086 | 0.056 | 0.0592 | 0.0739 |

Moyenne =0.0414, écart = 0.0245

## 2.2 Méthode du plus proche voisin

Pour comparer le résultat, nous essayons la méthode du plus proche voisin. Pour obtenir un bon résultat, nous fait d'élimination. Nous supprimons les points qui ont les deux plus proches points de différentes identifications.

Taux d'erreur de méthode du plus proche voisin

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 0.0064 | 0.030 | 0.0389 | 0.0732 | 0.0349 | 0.0394 | 0.0086 | 0.056 | 0.0614 | 0.0762 |

Moyenne =0.0426, écart = 0.0243

Le résultat est proche du résultat des fenêtres de Parzon.

## 2.3 Séparation à vaste marge(SVM)

Le départ de SVM est très simple. On veut chercher la surface de séparation qui a la vaste marge. Le problème peut être transformé en un problème d'optimisation quadratique. On peut classifier SVM en trois conditions--linéairement séparable, non linéairement séparable et non linéaire. Ici on teste la méthode qui adapte à la condition non linéairement séparable et teste les noyaux non linéaires. Ici on considère la situation avec deux classes. Pour plusieurs classes, on peut simplement classifier deux par deux.

Pour la situation non linéairement séparable, on doit introduire les variables ressorts. Le problème change en :

$$min_{\omega,b,\xi} \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{N} \xi_i \, , avec \; y_i(\omega \cdot x_i - b) \geq 1 - \xi_i, \forall i$$

$\xi_i$ sont les variables ressorts et $C$ une paramètre constante. $y_i = \pm 1$pour les deux classes respectivement. Par la méthode de Lagrange, on peut changer la forme du problème :

$$min_\alpha \boldsymbol{\psi}(\boldsymbol{\alpha}) = min_\alpha \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{n} y_i y_j K(x_i, x_j)\alpha_i\alpha_j - \sum_{I=1}^{n} \alpha_I$$

$$avec\ 0 \le \alpha_I \le C, \sum_{I=1}^{n} y_i \alpha_I = 0.$$

Après, on peut trouver que :

$$\omega = \sum_{i=1}^{N} y_i \alpha_I x_i \ , b = \ \omega \cdot x_k - y_k, pourtout, \alpha_k > 0$$

$K(x_i, x_j)$est une fonction noyau. Pour le cas linéaire, $K(x_i, x_j) = {x_i}^T x_j$. Pour déterminer la classification, on doit calculer :

$$u = \sum_{i=1}^{N} y_i \alpha_I K(x_i, x) - b$$

Son signe signifie la classe.

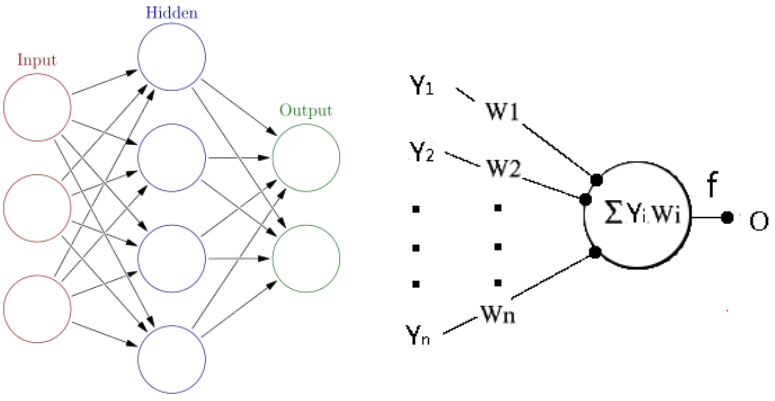
Il est très simple de changer SVM linéaire pour adapter au cas non linéaire. On peut justement changer la définition de la fonction noyau. Par exemple, on peut choisir d'utiliser :

$$K(x_i, x) = exp\left\{-\frac{|x-x_i|^2}{\sigma^2}\right\}.$$

On a testé l'algorithme sur les numéros 6 et 8, le taux d'erreur de 6 est 0.22% et le taux d'erreur de 8 est 1.54%. Mais le calcul d'apprentissage est très lourd.

## 2.4 BP (Back Propagation) réseaux de neurones

Réseaux de neurones rétropropagatice est une méthode d'utilisation de réseaux de neurones. Nous utilisonsles réseaux avec trois couches, comme l'indique la figure.



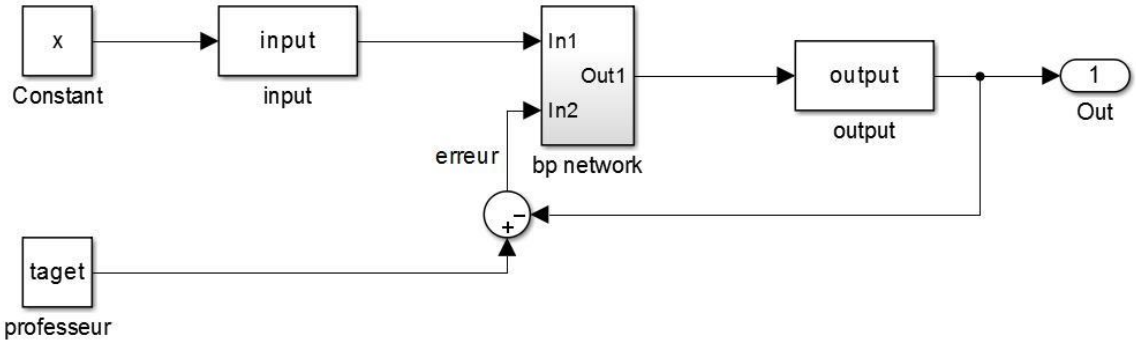



Figure 7. BP network

Nous utilisons la fonction «tansig » entre la couche «input » et la couche «hidden », puis la fonction linéaire entre la couche «hidden »et la couche «output ».

$$\text{tansig}(x) = \frac{1-e^{-x}}{1+e^{-x}}$$

Nous prenons le nombre des éléments dans la couche secrète N égal à 10 par défaut, pour choisir la meilleure méthode d'entraîner.

D'abord nous utilisons la méthode de trainbr (Bayesian regularization backpropagation), puis nous utilisons la méthode de trainlm (Leverberg-Marquardt). Ce sont les méthodes plus fréquentes à la condition que la quantité n'est pas grande.

**trainfcn = trainbr** Taux d'erreur,N=10

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 0.0301 | 0.0924 | 0.0692 | 0.0431 | 0.0466 | 0.0348 | 0.0064 | 0.0625 | 0.0986 | 0.0874 |

Moyenne =0.0571, écart = 0.0301

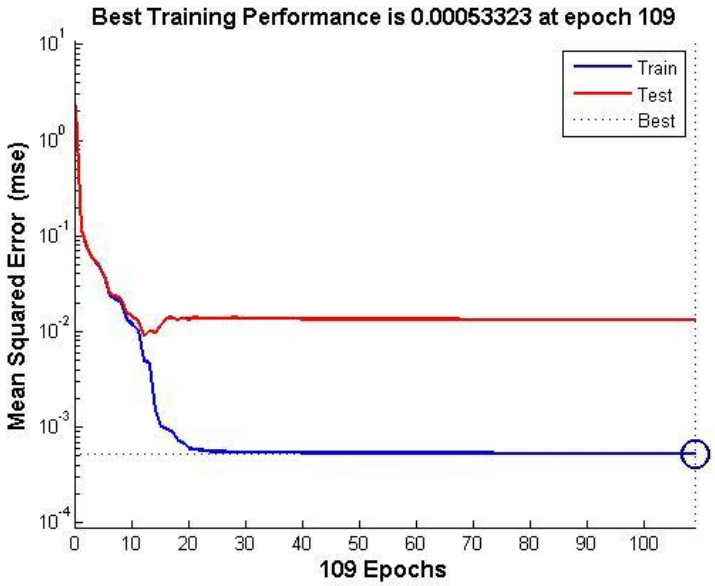



Figure 8. La performance quand trainfcn = trainbr, N = 10.

**trainfcn = trainlm** Taux d'erreur, N=10

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 0.0215 | 0.0258 | 0.0173 | 0.0387 | 0.0233 | 0.03944 | 0.0215 | 0.0301 | 0.0416 | 0.0336 |

Moyenne =0.0293, écart = 0.0087

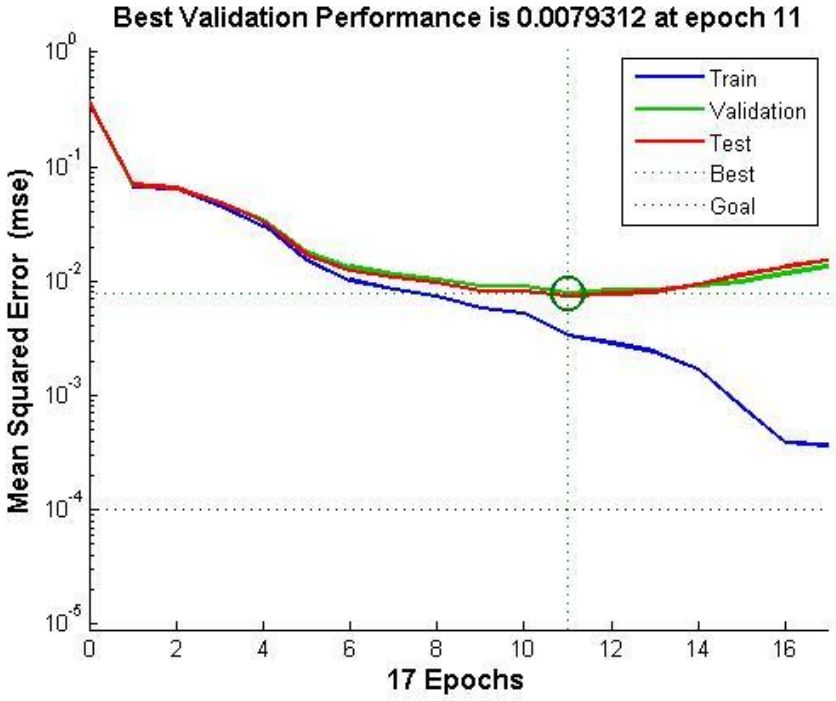



Figure 9. La performance quand trainfcn = trainlm, N = 10.

Nous découvrons que la méthode de BR fait plus de temps et sa performance est mauvaise. La méthode de LM converge plus vite, son résultat est plus précis. Alors nous choisissons la méthode de LM. Puis nous étudions la influence de N (le nombre de couche « hidden »).

Normalement, le système avec plus de neurone exprime un résultat meilleur. Mais ça coûte plus de l'espace mémoire interne. Dans notre cas, le système N = 20 dépasse déjà la extrême limite du ordinateur normale. Nous choisissions N = 15 pour obtenir un meilleur résultat.

**trainfcn = trainlm** Taux d'erreur, N=15

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 0.0043 | 0.0537 | 0.0324 | 0.0431 | 0.02097 | 0.0232 | 0.0064 | 0.0280 | 0.0175 | 0.0358 |

Moyenne =0.0266, écart = 0.0155

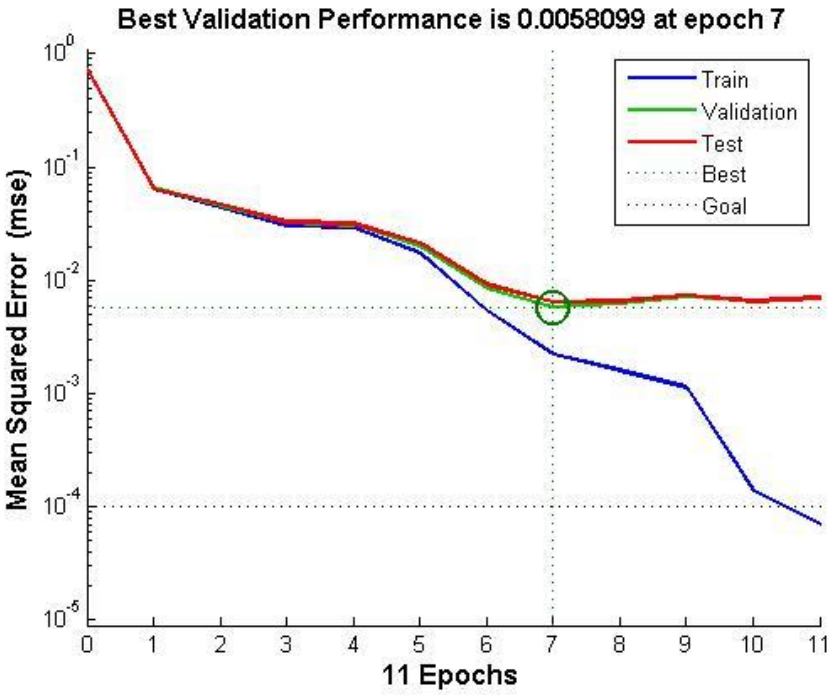



Figure 10. La performance quand trainfcn = trainlm, N = 15.

## 3. La sélection des caractéristiques

Maintenant, on a beaucoup de caractéristiques. Mais pour l'algorithme d'apprentissage, s'il y a plus de caractéristiques, le résultat de reconnaissance ne va pas certainement être meilleur. Alors, la sélection de caractéristique est importante pour choisir la meilleure combinaison de caractéristique. De plus, le plus grand nombre de caractéristique implique une calcule plus lourde. Alors la sélection des caractéristiques peut alléger la quantité de calcul, voire optimiser la qualité du système.

### 3.1 La distance entre les classes et la distance dans la classe

On considère la condition préalable de la reconnaissance des chiffres. Pour chaque chiffre manuscrit, on utilise un vecteur composé de plusieurs caractéristiques pour signifier le chiffre. Donc tous les chiffres situent dans un espace linéaire avec une dimension très haute. Si les caractéristiques sont bonnes, différents éléments de classe différente vont situent dans les zones différentes. De plus, les éléments de la même classe vont être proches. Pour décrire cette situation mathématiquement, d'abord, on définit la distance entre les classes et la distance dans la classe.

Noter $x_k^i$ et $x_l^j$ deux vecteurs caractéristiques dans la classe $\omega_i$ et la classe $\omega_j$, avec dimension $D$. On dénote $\delta(x_k^i, x_l^j)$ la distance entre les deux classes. Donc

la distance moyenne peut être écrite comme :

$$J_d(x) = \frac{1}{2}\sum_{i=1}^{c} P_i \sum_{j=1}^{c} P_j \frac{1}{n_i n_j} \sum_{k=1}^{n_i} \sum_{l=1}^{n_j} \delta(x_k^i, x_l^j)$$

Dans cette équation, $c$ dénote le nombre des classes, $n_i, n_j$ dénotent le nombre des éléments dans la classe $\omega_i$ et la classe $\omega_j$ respectivement. $P_i, P_j$ sont la probabilité priori.

Il y a beaucoup de définitions de la distance. Ici, on choisir la distance euclidienne. Alors :

$$\delta(x_k^i, x_l^j) = (x_k^i - x_l^j)^T (x_k^i - x_l^j)$$

De plus, on note :

$$m_i = \frac{1}{n_i}\sum_{k=1}^{n_i} x_k^i,\ m = \sum_{i=1}^{c} P_i m_i.$$

Avec ces notation, on peut écrire la distance comme :

$$J_d(x) = \sum_{i=1}^{c} P_i \left[ \frac{1}{n_i} \sum_{k=1}^{n_i} \left(x_k^i - x_l^j\right)^T \left(x_k^i - x_l^j\right) + (m_i - m)^T (m_i - m) \right]$$

On sépare l'équation en deux. La première partie s' écrit:

$$J_w = \sum_{i=1}^{c} \frac{P_i}{n_i} \sum_{k=1}^{n_i} \left(x_k^i - x_l^j\right)^T \left(x_k^i - x_l^j\right)$$

Il est définie comme la distance moyenne dans les classes. Puis la deuxième partie est :

$$J_b = \sum_{i=1}^{c} P_i (m_i - m)^T (m_i - m).$$

On peut aussi écrire cette distance comme :

$$J_b = \frac{1}{2}\sum_{i=1}^{c} P_i \sum_{j=1}^{c} P_j (m_i - m_j)^T (m_i - m_j).$$

On définit cette partie la distance moyenne entre les classes.

## 3.2 L'algorithme de la sélection des caractéristiques

Maintenant, on a deux distances bien définies pour décrire les qualités des caractéristiques. La question est quelle combinaison des caractéristiques est le meilleur. Intuitivement, les éléments dans la même classe sont proche mais différentes classes se situent loin entre eux. Si une combinaison des caractéristiques est bonne, alors $J_w$ va être petit et $J_b$ va être grand. Donc on définit le ratio $J_b/J_w$ comme le critère. On le dénote $J$. Le plus grand $J$ signifie la meilleure combinaison des caractéristiques.

La deuxième question est comment chercher la meilleure combinaison. Une méthode simple est de calculer toutes les situations. Mais ça n'est pas pratique parce que le calcule va être très grand. Donc on doit utiliser les algorithmes plus rapides. Ici on choisir une algorithme qui combine SFS (Sequential Forward Selection) et SBS (Sequential Backward Selection). Les algorithmes sont expliqués dessous.

3.2.1 SFS (Sélection séquentielle Forward) et GSFS (Generalized SFS)

L'algorithme SFS dit que si on a $k$ caractéristiques qui compose une groupe caractéristique $X_k$, pour les autres $D-k$ caractéristiques, on calcule :

$$J(X_k + x_i), i = 1,2,\cdots, D-k$$

Si $J(X_k + x_i)$ est le plus grand, donc on choisit $X_{k+1} \coloneqq X_k + x_i$. Finalement, on peut obtenir $n$ caractéristiques. Pour GSFS, chaque fois, on ajoute $r$ éléments dans le groupe. La fiabilité est meilleure, mais ça fait plus du calcul d'ordre($C_r^{D-k}$).

3.2.2 SBS (Sélection arrière séquentielle) et GSBS (Generalized SBS)

SBS est similaire à SFS. La différence est que chaque fois on supprimer un élément dans $X_k$. Si $J(X_k - x_i)$ est le plus grand, donc on choisir $X_{k+1} \coloneqq X_k - x_i$. On itère l'algorithme jusqu'à il y a $n$ éléments restants. Pour GSBS, on supprimer $r$ éléments dans l'itération.

3.2.3 Combinaison de GSFS et GSBS

Si on combiner GSFS et GSBS, on peut améliorer la fiabilité de résultat. C'est-à-dire on peut ajouter $l$ éléments par GSFS et après supprimer $r$ éléments par GSBS. Et finalement on peut obtenir $n = l - r$ éléments. Le processus peut être dénoté comme :

$$Z_l = (l_1, l_2, \cdots, l_i), l_1 + l_2 + \cdots + l_i = l$$

$$Z_r = (r_1, r_2, \cdots, r_j), r_1 + r_2 + \cdots + r_j = r$$

Ici on teste l'algorithme par choisir $l_1 = l_2 = \cdots = l_i = 2$ et $r_1 = r_2 = \cdots = r_j = 1$. Le résultat avec 100 caractéristiques est dessous.

**trainfcn = trainlm** Taux d'erreur, N=10

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 0.0064 | 0.0537 | 0.0476 | 0.0474 | 0.037 | 0.0255 | 0.028 | 0.0603 | 0.0833 | 0.1098 |

Moyenne =0.0567, écart = 0.0306

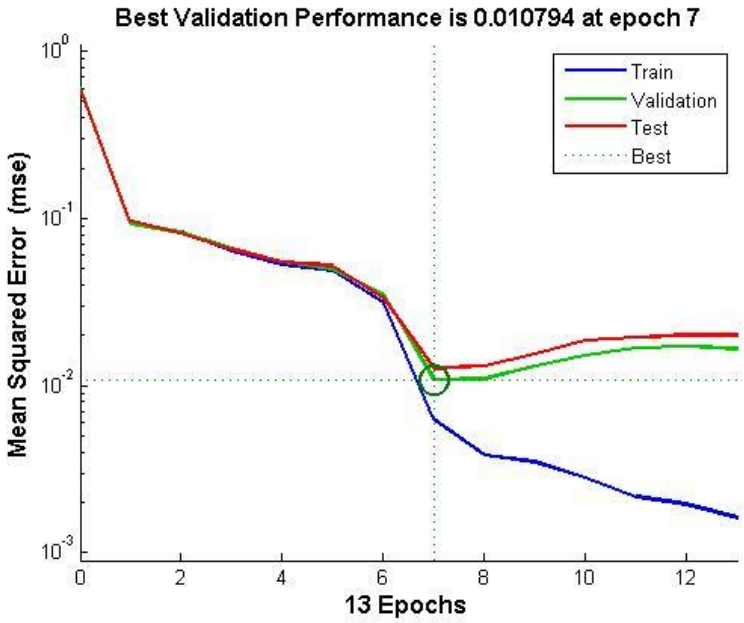



Figure 11. La performance quand trainfcn = trainbr, N = 10 avec 100 caractéristiques

Le taux d'erreur de 9 est un peu haut. Il y a deux raisons. Premièrement, cet algorithme ne peut pas choisir la meilleure combinaison. De plus, le critère $J$ n'est pas un critère optimal. Peut-être quelque classes sont très loin mais quelques classes sont assez proche. Dans ce cas-là, la moyenne devient le plus grande. Donc, il reste encore des espace pour optimiser le système.

## 4. Conclusion

Dans ce projet, nous avons cherché sur la reconnaissance des chiffres manuscrits. Avec les exames plus approfondi, pour obtenir un meilleur résultat nous choisissions la méthode de réseaux de neurones. Et nous choisissions N = 15 comme le nombre de couche « hidden », et la méthode de trainlm (Leverberg-Marquardt) pour entraîner. Nous obtenons un taux d'erreur moyenne 2.66%.

## 5. Référence

【1】 Theodoridis S, Koutroumbas K. Pattern Recognition, Academic Press[J]. New York, 1999.

【2】 Tou J T, Gonzalez R C. Pattern recognition principles[J]. 1974.

【3】 Trier Ø D, Jain A K, Taxt T. Feature extraction methods for character recognition-a survey[J]. Pattern recognition, 1996, 29(4): 641-662.

【4】 Lim J S. Two-dimensional signal and image processing[J]. Englewood Cliffs, NJ, Prentice Hall, 1990, 710 p., 1990, 1.

【5】 Sonka M, Hlavac V, Boyle R. Image processing, analysis, and machine vision[M]. Cengage Learning, 2014.