

Projet IN54 A2016

Reconnaissance de chiffres manuscrits

Hamza Jaffali

2 janvier 2017

Table des matières

1	Introduction	3
2	Localisation et extraction des chiffres manuscrits	3
2.1	Recherche des lignes hautes et basses de chaque classe	3
2.2	Recherches des lignes gauches et droites de chaque chiffre de chaque classe	4
2.3	Recherches rectangles englobants de chaque chiffre de chaque classe	4
2.4	Résultats	5
3	Premier classifieur : profil et distance euclidienne minimale	5
3.1	Extraction des profils	5
3.2	Apprentissage du classifieur	6
3.3	Décision	8
3.4	Résultats	8
4	Second classifieur : densités et K plus proches voisins	11
4.1	Calcul des densités	11
4.2	Apprentissage du classifieur	12
4.3	Décision	12
4.4	Résultats	13
5	Combinaison des classifieurs	15
6	Autres classifieurs	15
7	Conclusion	16

Table des figures

1	Résultat de la localisation et extraction pour l'image de la base d'apprentissage	6
2	Résultat de la localisation et extraction pour l'image de la base d'apprentissage	7
3	Exemple de profils gauche et droit pour le chiffre 8 à droite	7
4	Exemple de vecteur de probabilités pour le chiffre 0 à droite	9
5	Exemple de vecteur de probabilités pour le chiffre 1 à droite	9
6	Exemple de vecteur de probabilités pour le chiffre 2 à droite	10
7	Evolution du taux de reconnaissance en fonction de la taille $d/2$ des profils gauches et droits	10
8	Exemple de matrice de densités pour le chiffre 6 à droite	11
9	Exemple de vecteur de probabilités pour le chiffre 6 à droite	12
10	Evolution du taux de reconnaissance en fonction du nombre $n \times m$ de zones	13
11	Evolution du taux de reconnaissance en fonction du nombre $n \times m$ de zones	14

1 Introduction

Dans le cadre de l'Unité de Valeur IN54, dont l'intitulé est "Reconnaissance de formes", nous avons été amenés à développer un projet de reconnaissance de formes, plus précisément, un projet portant sur la reconnaissance de chiffres manuscrits.

L'objectif du projet est d'implémenter un certains nombre de méthode de classification proposées, de constater et de discuter les résultats propres à chaque méthode. Une seconde partie consistera en la combinaison de ces classifieurs selon différentes approches, et voir si ces méthodes de combinaisons apportent un avantage par rapport aux méthodes initiales.

Une dernière partie consistera en la proposition d'amélioration et de nouvelles idées de classifieur pour la reconnaissance de chiffres manuscrits.

Dans le code source fournit avec le rapport, les trois parties sont rangées par dossier. Un script *Main.m* permet de lancer les calculs pour chaque classifieur ou pour la combinaison.

2 Localisation et extraction des chiffres manuscrits

Afin de pouvoir travailler avec les images de la base d'apprentissage et de test contenant un certain nombre de chiffres manuscrits alignés, il convient tout d'abord d'extraire chacun des chiffres présents sur l'image à traiter.

2.1 Recherche des lignes hautes et basses de chaque classe

Les chiffres manuscrits étant plus ou moins alignés par classes, il convient tout d'abord d'extraire les différentes classes de chiffres initialement rangées par lignes dans les images disponibles.

Pour ce faire, on détermine tout d'abord la projection horizontale de l'image en utilisant la fonction *sum* sur la seconde dimension de la matrice d'image (c'est à dire une somme pour chaque ligne, en faisant varier les colonnes). On aurait également pu utiliser la fonction *min*. Cette somme n'est effectuée que pour les pixel dont le niveau de gris est inférieur à 50 (pixel considéré comme noir et non blanc). On obtient alors un vecteur vertical qui contient une valeur non nulle à un indice donné, seulement si sur la ligne du même indice il y avait au minimum un pixel noir. De même, si l'on obtient à une valeur nulle à un indice donné de ce

vecteur vertical, c'est que sur l'image, la ligne du même indice ne contient que des pixels blancs.

En utilisant ce processus, on détermine alors les lignes hautes et basses de chaque classe de chiffre. En effet, on parcourt l'ensemble des lignes de l'image : si une ligne avec au moins un pixel noir apparaît et que la ligne précédente est une ligne "blanche" alors la ligne blanche est la ligne haute de la classe correspondante (car on laisse une ligne vide) ; si une ligne "blanche" apparaît et que la ligne précédente est une ligne avec au moins un pixel noir alors la ligne blanche est la ligne basse de la classe correspondante.

Enfin, on stocke dans deux tableaux de dimension 10 (le nombre de classes de chiffres) les indices des lignes hautes et basses respectivement.

2.2 Recherches des lignes gauches et droites de chaque chiffre de chaque classe

Une fois la ligne haute et basse de chaque classe de chiffre connues, et donc que l'on a réussi à extraire la bande correspondant à chaque classe de chiffre, il convient à présent d'extraire les lignes des chiffres d'une même classe, toujours selon le même principe de projection.

Afin de séparer les chiffres d'une même classe, disposés horizontalement, on procède à une projection verticale de chaque bande de l'image. On procède alors à une somme sur la première dimension et selon le même principe qu'expliqué ci-dessus, on en déduit les lignes gauches et droites de chaque chiffre. On prend également soin de laisser une ligne de plus à droite et à gauche, pour anticiper l'étape de détermination du rectangle englobant.

On recommence ensuite ce processus pour les 10 classes de chiffres différentes, et on stocke les résultats dans deux matrices , de dimension $10 \times$ le nombre de chiffres dans une classe (20 pour notre image d'apprentissage), les indices des lignes gauches et droites respectivement.

2.3 Recherches rectangles englobants de chaque chiffre de chaque classe

Une fois les lignes gauches et droites connues pour chaque chiffre, il convient de déterminer les lignes hautes et basses proprement à chaque chiffre cette fois ci. Pour ce faire, et étant donné que cela fonctionne très bien, on réutilise la méthode

de projection horizontale pour mettre à jour les lignes hautes et basses de chaque chiffre de chaque classe de l'image.

On possède alors les indices des lignes hautes, droites, basses et gauches pour chaque chiffre de l'image. On peut donc déterminer les rectangles englobants pour chaque chiffre.

Nous choisissons alors de stocker les informations du rectangle englobant d'un chiffre dans une matrice construite comme suit :

$$R = \begin{pmatrix} \text{indice_haut} & \text{indice_gauche} \\ \text{indice_bas} & \text{indice_droite} \end{pmatrix}$$

On stocke alors tous les rectangles englobant dans une cellule contenant toute ces matrices. Les matrices sont rangées aux mêmes indices que les indices des chiffres sur l'image. Le chiffre en haut à gauche aura donc l'indice $\{1, 1\}$.

2.4 Résultats

3 Premier classifieur : profil et distance euclidienne minimale

Dans cette section, nous implémentons le premier classifieur basé sur l'extraction d'un profil droit et gauche de chaque image, et basé sur la distance minimale au centre d'une classe pour la décision (distance euclidienne minimum).

3.1 Extraction des profils

Afin de déterminer le profil gauche ou droit d'un chiffre, on découpe préalablement le chiffre en $\frac{d}{2}$. Ainsi, à chaque partie de l'image, en partant du bord gauche (respectivement droit), on parcourt le chiffre vers la droite (respectivement gauche) jusqu'à ce que les pixels de l'image ont un niveau de gris inférieur à 50 (ce qui est considéré comme un pixel noir). On garde alors l'indice pour lequel ceci arrive pour en déduire la distance entre ce premier pixel noir et le bord gauche (respectivement droit). On divise alors cette distance par la longueur entre bord droit et gauche, et l'on stocke ce nombre à l'indice correspondant dans le vecteur de profil gauche (respectivement droit).

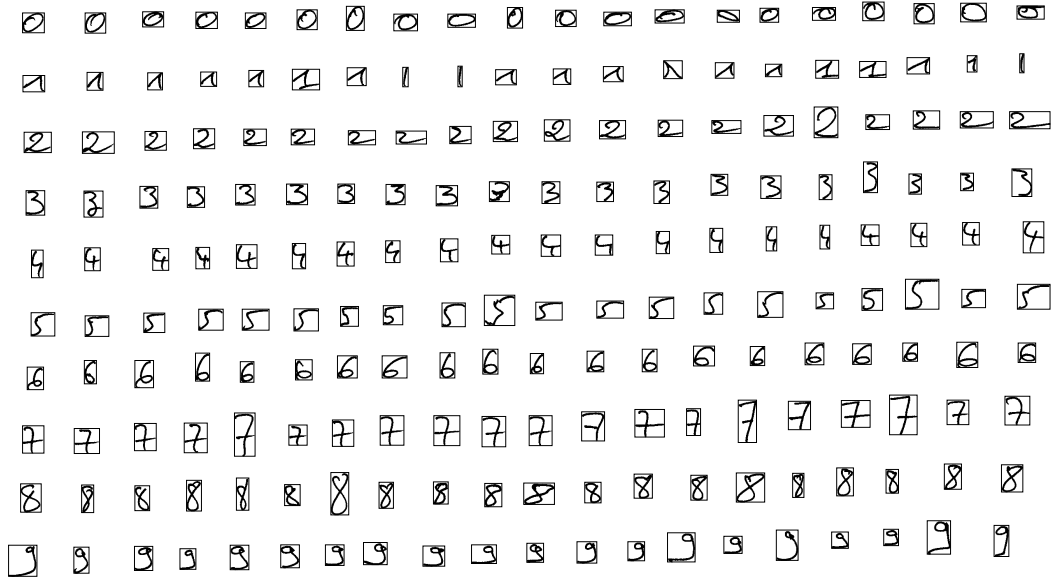


FIGURE 1 – Résultat de la localisation et extraction pour l'image de la base d'apprentissage

A la fin du traitement, on concatene les deux vecteurs en un seul, le profil gauche en haut, et le profil droit en bas, afin de former un vecteur de taille d correspondant au profil du chiffre.

On remarque pour ce chiffre, par exemple, que le fait de choisir un découpage partant du haut jusqu'au bas n'est pas toujours pertinent, car aux extrémités du chiffre, les distances aux bords peuvent être très grandes ou très courtes alors que cela ne reflète pas la forme du chiffre. Par ailleurs, les points caractéristiques du chiffres ne sont pas pris en compte lorsque l'on parcourt, du haut vers le bas à pas constant, l'image pour déterminer le profil gauche et droit.

3.2 Apprentissage du classifieur

La phase d'apprentissage consiste à déterminer, dans un premier temps, le profil de chaque chiffre de l'image *app.tif*. Pour ce faire, on localise et extrait tous les chiffres de l'image d'apprentissage, et l'on applique l'algorithme explicité ci-dessus. On obtient alors un vecteur de profil pour chacun des chiffres de l'image.

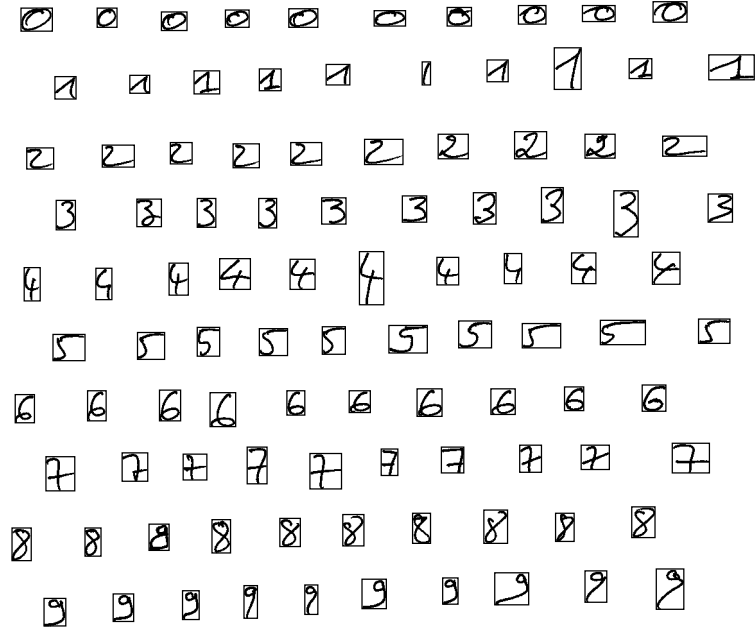


FIGURE 2 – Résultat de la localisation et extraction pour l'image de la base d'apprentissage

$$P_{gauche} = \begin{pmatrix} 0.7273 \\ 0.1591 \\ 0.3636 \\ 0.0455 \\ 0.2045 \end{pmatrix} \quad P_{droit} = \begin{pmatrix} 0.1818 \\ 0.1136 \\ 0.3182 \\ 0.0682 \\ 0.5682 \end{pmatrix}$$



FIGURE 3 – Exemple de profils gauche et droit pour le chiffre 8 à droite

Ensuite, on détermine le centre de chaque classe. Pour ce faire, on calcule la moyenne des vecteurs de profil pour tous les chiffres de la même classe. On obtient alors les 10 centres qui seront utilisés pour le processus de décision.

3.3 Décision

La phase de décision consiste à partir d'un chiffre en entrée, de lui attribuer un vecteur de probabilité, donc chacune des composantes et la probabilité d'appartenance du chiffre à la classe désignée par l'indice de la composante.

Pour construire ce vecteur, on calcule tout d'abord la distance entre le vecteur de profil du chiffre à reconnaître et les centres de la base d'apprentissage. On obtient alors un vecteur de distances de taille le nombre de classes (ici 10).

Enfin, chaque i -ème composante du vecteur de distances est transformée selon la formule suivante :

$$v_{ecteur_distance}(i) = \frac{\exp(-v_{ecteur_distance}(i))}{\sum_{j=1}^{10} \exp(-v_{ecteur_distance}(j))}$$

On obtient alors le vecteur de probabilité final pour un chiffre. On calcule alors ce vecteur de probabilité pour chacun des chiffres de la base de test, et l'on stocke tous ces vecteurs dans une cellule de même dimension que le nombre de chiffres de l'image de test.

3.4 Résultats

Nous listons alors ci-dessous quelques exemples de vecteurs de probabilité pour quelques chiffres choisis, avec $d/2 = 5$:

Après application à tous les chiffres, on peut s'intéresser à l'évolution du taux de reconnaissance en fonction de la taille du profil.

On remarque que la courbe d'évolution possède deux phases distinctes en fonction de la taille du profil gauche et droit. En effet, pour une valeur de $d/2$ comprise entre 1 et 7 environ, le taux de reconnaissance croît. On en déduit alors que l'ajout de distances dans le profil permet de différencier de manière plus précise certains chiffres et donc augmenter rapidement le taux de reconnaissance global. Dans un second temps, on rencontre une phase d'oscillation où l'ajout de distances dans le vecteur de profil n'induit pas forcément une meilleure reconnaissance des chiffres. On vérifie que le taux maximal de reconnaissance est rencontré pour $d/2 = 20$ avec un taux maximum de 84%, pour notre base d'apprentissage et de test,


$$\text{Vecteur_probabilité} = \begin{pmatrix} 0.1809 \\ 0.1007 \\ 0.1006 \\ 0.0759 \\ 0.0635 \\ 0.0829 \\ 0.0944 \\ 0.0893 \\ 0.1395 \\ 0.0722 \end{pmatrix}$$


FIGURE 4 – Exemple de vecteur de probabilités pour le chiffre 0 à droite

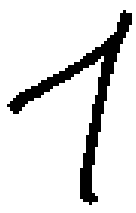
$$\text{Vecteur_probabilité} = \begin{pmatrix} 0.1037 \\ 0.1684 \\ 0.0811 \\ 0.0883 \\ 0.0778 \\ 0.0795 \\ 0.0959 \\ 0.1126 \\ 0.1078 \\ 0.0849 \end{pmatrix}$$


FIGURE 5 – Exemple de vecteur de probabilités pour le chiffre 1 à droite

mais il peut en être autrement pour une autre base d'apprentissage ou un autre échantillon de test.

Une piste d'amélioration du classifieur serait de tester différentes distances autres que la distance euclidienne, et voir si l'on prend mieux en compte la différence entre les chiffres induite par cette distance. Une autre manière d'améliorer le classifieur serait aussi de proposer une autre manière de répartir les endroits où l'on calcule les distances pour les profils : choisir une autre manière que celle d'espacer uniformément les points de mesure, par exemple selon une courbe gaussienne, ou de manière aléatoire.

$$\text{Vecteur_probabilité} = \begin{pmatrix} 0.1053 \\ 0.1139 \\ 0.1277 \\ 0.0825 \\ 0.0674 \\ 0.0758 \\ 0.1115 \\ 0.0958 \\ 0.1453 \\ 0.0748 \end{pmatrix} \quad \text{2}$$

FIGURE 6 – Exemple de vecteur de probabilités pour le chiffre 2 à droite

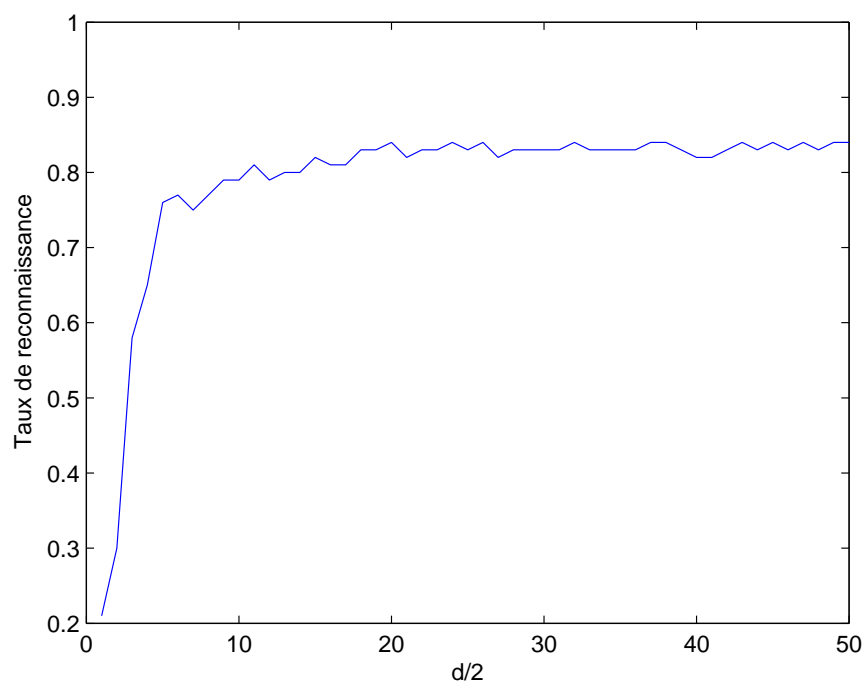


FIGURE 7 – Evolution du taux de reconnaissance en fonction de la taille $d/2$ des profils gauches et droits

4 Second classifieur : densités et K plus proches voisins

Dans cette section, nous nous attarderons sur l'implémentation du second classifieur basé sur la caractérisation des chiffres par la densité de pixel sur ses sous-parties, avec une décision prise sur le principe des K plus proches voisins.

4.1 Calcul des densités

Contrairement au classifieur 1 qui caractérisait chaque chiffre par rapport à son profil, le classifieur 2 associe à chaque chiffre une matrice de densité. On découpe dans un premier temps le rectangle englobant du chiffre selon les deux dimensions, en n zones verticalement, et m zones horizontalement. On obtient alors $n \times m = d$ zones "identiques" formant le chiffre.

On détermine alors une densité sur chacune de ces zones, en comptant le nombre de pixels noirs présents dans cette zone par rapport à la surface totale de cette dernière. On obtient alors une matrice de taille $n \times m$ de densités caractérisant le chiffre.

On trouvera en Figure 8 en exemple de matrice de densité pour un chiffre donné.

$$\text{Matrice_densité} = \begin{pmatrix} 0 & 0.3316 & 0.8673 & 0.9439 & 0 \\ 0.1531 & 0.8418 & 0.1020 & 0.1276 & 0 \\ 0.6122 & 0.1786 & 0 & 0 & 0 \\ 1.0459 & 0.8163 & 0.8163 & 0.6633 & 0 \\ 0.7143 & 0 & 0 & 0.5867 & 0 \\ 0.7908 & 0.7143 & 0.6633 & 0.7653 & 0 \end{pmatrix} \quad \text{6}$$

FIGURE 8 – Exemple de matrice de densités pour le chiffre 6 à droite

Un des reproches que l'on pourrait faire à cette méthode c'est le découpage de zones qui peut ne pas être toujours pertinent pour différencier les chiffres. Peut être qu'un découpage circulaire en partant du centre, et en augmentant au fur et à mesure le rayon pourrait être plus intéressant, mais cela est à vérifier expérimentalement. D'autres idées de découpage du rectangle englobant pourraient être envisagées, faisant intervenir plus ou moins de considérations sur le chiffre en question.

4.2 Apprentissage du classifieur

La phase d'apprentissage consiste alors à appliquer le calcul des densités pour tous les chiffres de l'image de base. On stocke ensuite les différentes matrices dans un cellule de même taille que le nombre de chiffres en lignes et colonnes (10×20 dans notre cas).

4.3 Décision

Afin de déterminer les vecteurs de probabilité d'appartenance à chacune des classes, on utilise la méthode des K plus proches voisins. Effectivement, on détermine tout d'abord la matrice de densité du chiffre que l'on veut identifier. On calcule ensuite les distances entre cette matrice et celles de tous les chiffres de la base d'apprentissage. Dans notre cas, on choisit la distance euclidienne.

A partir des distances connues entre notre chiffre et tous les chiffres de la base, on ne garde que les K chiffres les plus proches de notre chiffre à identifier. La probabilité d'appartenir à une classe est alors le nombre de chiffres, parmi ces K plus proches, qui appartiennent à cette classe, divisé par le nombre total de voisins (c'est à dire K).

On trouve Figure 9 un exemple de vecteur de probabilité pour le chiffre 6, avec $n = m = 5$ et $K = 5$:

$$\text{Vecteur_probabilité} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \text{6}$$

FIGURE 9 – Exemple de vecteur de probabilités pour le chiffre 6 à droite

4.4 Résultats

Le classifieur 2 se base sur un nombre de paramètres supérieur à celui du classifieur 1. En effet, on peut jouer sur le découpage du rectangle sur le plan vertical avec la variable n , et horizontal avec la variable m . De plus, le nombre de plus proches voisins K retenu peut également faire varier la qualité du résultat.

En fixant une valeur pour le nombre de plus proches voisins retenus à $K = 10$, on fait alors varier les valeurs de n et m entre 1 et 20, et l'on étudie l'influence de ces variations sur la qualité de reconnaissance. On remarque, après étude de la surface tracée, que plus l'on augmente le nombre de zones, plus le taux de connaissance semble meilleur. A partir d'un moment, le taux ne se stabilise pas et le maximum n'est pas forcément atteint pour des valeurs maximales de n et m . En effet, on observe que pour $K = 10$, les meilleurs valeurs sont $n = 7$ et $m = 4$, et le taux est de 66%.

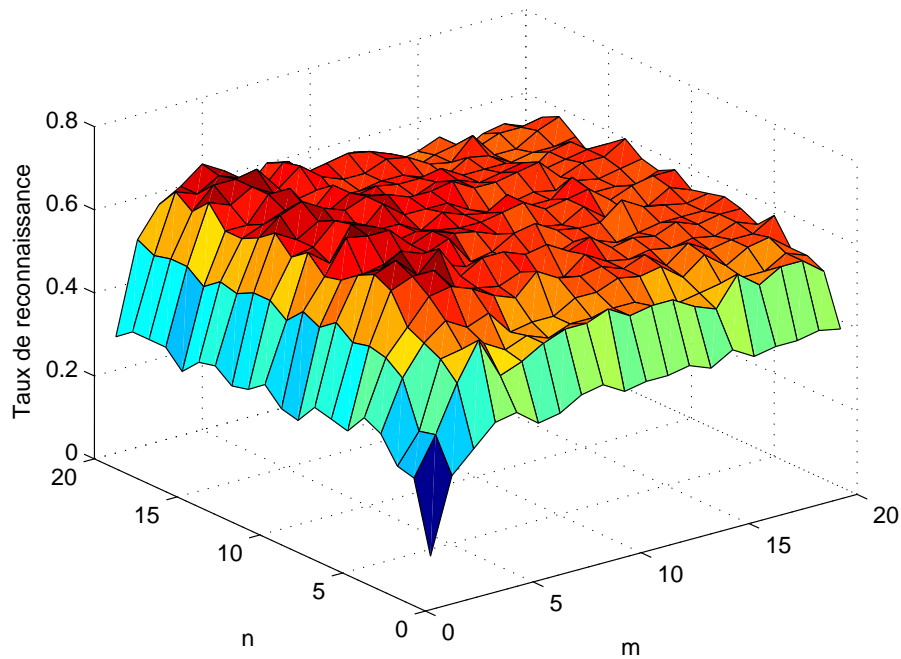


FIGURE 10 – Evolution du taux de reconnaissance en fonction du nombre $n \times m$ de zones

Si on suppose à présent que $n = m$, c'est à dire que l'on a $d = n^2$ zones, il est intéressant de voir l'évolution du taux de reconnaissance en faisant évoluer le nombre de zones $n = m$ ainsi que le nombre de plus proches voisins K , tous les deux de 1 à 20. On s'aperçoit dans un premier temps, que plus la valeur de $n = m$ augmente, plus la taux de reconnaissance s'affine. Passé une certaine valeur (4 environ), le taux semble se stabiliser entre 0.7 et 0.8 environ. La variation de K ne semble pas jouer énormément d'importance dans la taux de reconnaissance, néanmoins, plus le nombre est grand, et plus le taux diminue légèrement tout de même. En effet, on peut comprendre cela par le fait que plus on a de voisins pris en compte, plus on a de chance de tomber sur des chiffres qui ne correspondent pas. Enfin, on remarque que le meilleur taux de reconnaissance est atteint pour la valeur de $K = 1$ et $n = m = 13$ et le taux est de 83%. On comprend qu'avec $K = 1$, on prend l'unique chiffre le plus proche, ce qui revient en fait à la méthode de la distance euclidienne minimum.

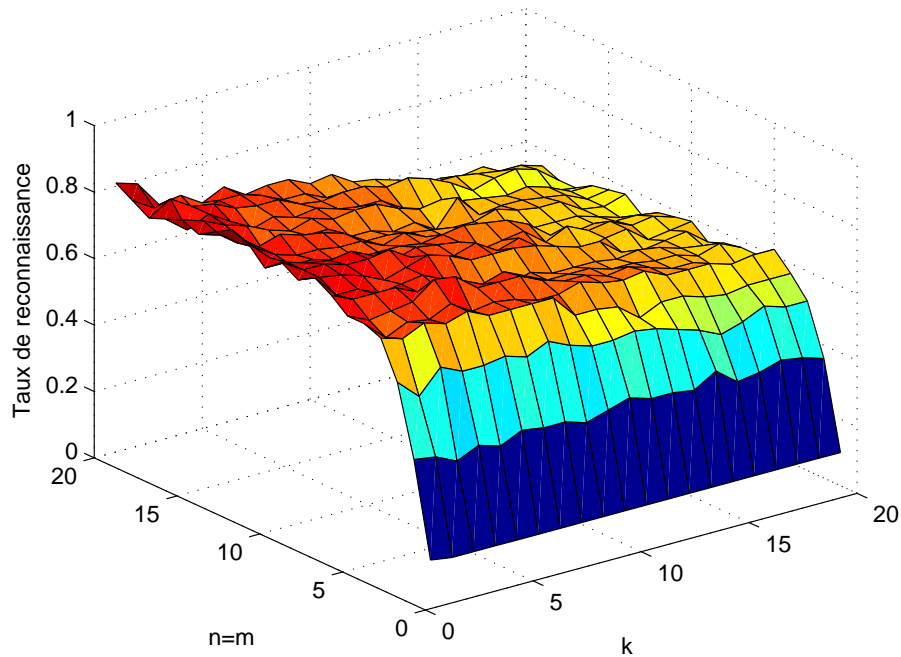


FIGURE 11 – Evolution du taux de reconnaissance en fonction du nombre $n \times m$ de zones

5 Combinaison des classifieurs

Dans cette section, nous nous intéresserons à la performance des deux méthodes de combinaison proposées : la somme ou le produits des vecteurs de probabilité.

En appliquant la première combinaison, qui est la somme, avec les meilleurs vecteurs de probabilité des classifieur 1 ($d/2 = 20$) et classifieur 2 ($n = m = 13$ et $K = 1$), on obtient un taux de reconnaissance égal à 83%, ce qui est meilleur que le classifieur 1, mais moins bon que le classifieur 2.

En appliquant la seconde combinaison, qui est le produit, pour les mêmes vecteurs de probabilités, on obtient également 83%.

En modifiant les vecteurs de probabilité, on obtient encore une fois la même probabilité pour les deux combinaisons.

Il faudrait proposer un plus grand nombre de vecteurs de probabilités pour étudier en détail laquelle des deux méthodes de combinaison est la meilleure.

6 Autres classifieurs

Dans l'optique d'améliorer toujours plus le taux de reconnaissance des chiffres manuscrits, d'autres classifieurs peuvent en effet être implémentés.

Le sujet du projet nous propose notamment un classifieur basé sur la distance de Mahalanobis, qui, à la différence de la distance euclidienne où toutes les composantes des vecteurs sont traitées indépendamment et de la même façon, accorde un poids moins important aux composantes les plus dispersées. Elle prend en compte la variance et la corrélation de la série de données [1]

On peut aussi imaginer mettre en place un réseau de neurones pour la reconnaissance des chiffres manuscrits. Cela pourrait être un réseau de neurones à plusieurs couches, ou bien un réseau de neurones entièrement connecté qui utiliserait la mémoire autoassociative pour reconnaître les chiffres à partir d'une base d'apprentissage en mémoire.

D'autres classifieurs tels que

7 Conclusion

Ce projet de reconnaissance de chiffres manuscrit fut vraiment un projet intéressant à plusieurs niveaux.

Il nous a permis, dans un premier temps, de pouvoir implémenter par nous même des principes, restés théoriques jusque là, de classification avec différentes méthodes, et d'étudier les contraintes et les influences des paramètres sur les résultats obtenus. C'est une très bonne base pour pouvoir implémenter par la suite un classifieur dans le domaine de l'image ou dans un autre domaine.

Dans un second temps, ce projet nous a permis d'appliquer ces techniques de classifications à un cas concret de reconnaissance de chiffre, qui est un cas fort intéressant. Cela nous permet d'avoir un aperçu réel des applications possibles dans le domaine de l'image des reconnaissances de formes.

C'est un projet très intéressant dans l'ensemble, et dont seulement le temps nous a empêché d'aller plus loin dans le détail du projet. Nous aurions en effet aimé implémenter d'autres types de classifieurs (notamment avec réseau de neurones), et également peaufiner et optimiser plus en détail les classifieurs déjà proposés.

Références

- [1] https://fr.wikipedia.org/wiki/Distance_de_Mahalanobis