Reconnaissance image

Lucas Marty

October 2024

Contents

1 Introduction

1.1 Présentation

L'objectif de ce projet est de développer un programme capable de reconnaître automatiquement les chiffres inscrits sur des images. Pour ce faire, nous avons accès à une base de données contenant une centaine d'images, chacune représentant un chiffre. Les chiffres présents sur chaque image sont connus et servent de référence pour entraîner le programme.

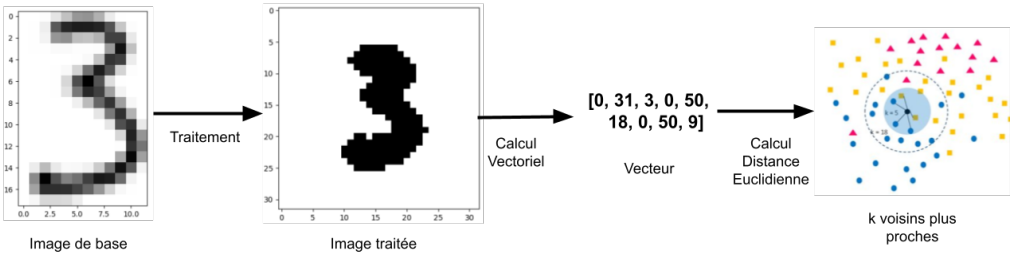

Figure 1: Etape du programme

Afin d'optimiser la reconnaissance des chiffres, nous allons prétraiter ces images pour les uniformiser. Ce prétraitement inclut des étapes telles que le redimensionnement des images à une taille fixe, la conversion en niveaux de gris, la binarisation et la dilatation de l'image. Ces étapes visent à mettre en avant les caractéristiques essentielles des chiffres, facilitant ainsi leur reconnaissance par le programme.
Une fois les images traitées, nous pouvons calculer un vecteur correspondant à chacune d'elles. Pour cela, l'image est découpée en plusieurs carrés de taille identique, et dans chacun de ces carrés, nous comptons le nombre de pixels noirs. Ce qui nous génère un vecteur pour chaque image.
Ensuite, pour déterminer le chiffre inscrit sur une image donnée, nous comparons son vecteur à ceux des images de la base de données. En utilisant un algorithme de plus proches voisins, nous identifions les vecteurs les plus similaires et attribuons à l'image testée le chiffre correspondant à celui de l'image la plus proche.

1.2 Base de Données

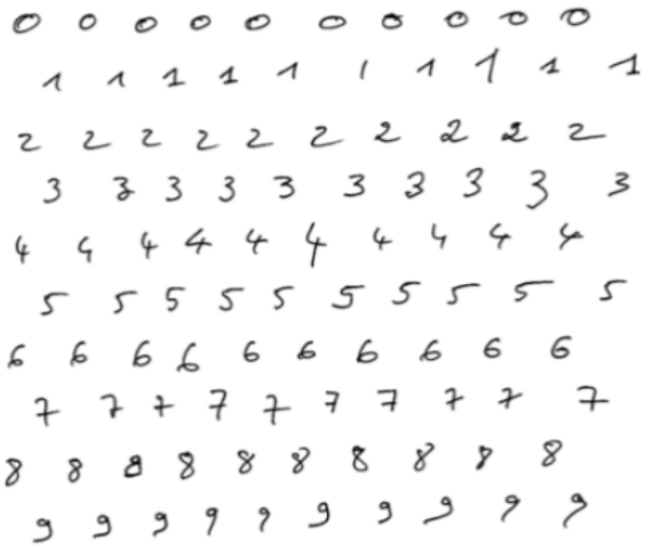

Figure 2: Base de données

La base de données fournie contient 100 images, 10 images pour chaque chiffre de 0 à 9. Ces images sont en niveaux de gris et chacune d'entres elles ont des tailles différentes. Afin de pouvoir utiliser ces images, il sera nécessaire de

les redimensionner et de les binariser rendant les futurs traitements d'image plus efficaces et uniformes.

1.3 Enjeux Scientifiques

Le premier défi auquel nous sommes confrontés est le traitement des images. Les premières opérations, telles que le redimensionnement et la binarisation, sont évidentes. Cependant, l'érosion et la dilatation des images peuvent grandement modifier des caractéristiques importantes, ce qui pourrait avoir un impact important sur l'efficacité du programme. Il est donc crucial de bien ajuster ces opérations pour éviter de perdre des détails essentiels à la reconnaissance des chiffres.

Un autre aspect crucial à analyser concerne les problèmes liés à la variance inter-classe et à la variance intra-classe des vecteurs obtenus à partir des images.

Dans notre cas, la variance inter-classe représente les différences entre la représentation graphique des différents chiffres. Ce que nous souhaitons, c'est une variance inter-classe importante, permettant de distinguer clairement deux chiffres différents. Cependant, nous pourrions rencontrer des problèmes liés à une faible variance inter-classe lorsque nous analysons les chiffres 1 et 7, ou encore 6 et 9, car ils sont proches en termes de forme.

La variance intra-classe représente les variations entre les vecteurs des images représentant un même chiffre. Idéalement, nous souhaitons une variance intra-classe faible afin de regrouper correctement les images d'un même chiffre. Dans notre base de données, on observe que le chiffre 1 peut être écrit de deux manières différentes : soit avec deux traits, soit avec un seul. Cela augmente la variance intra-classe du chiffre 1, le rendant ainsi plus difficile à identifier correctement par le programme.

2 Proposition d'une solution

2.1 Chaîne de traitement

Comme dit précédemment, nous commençons par traiter l'ensemble des images de la base de données afin de pouvoir les analyser plus efficacement par la suite.

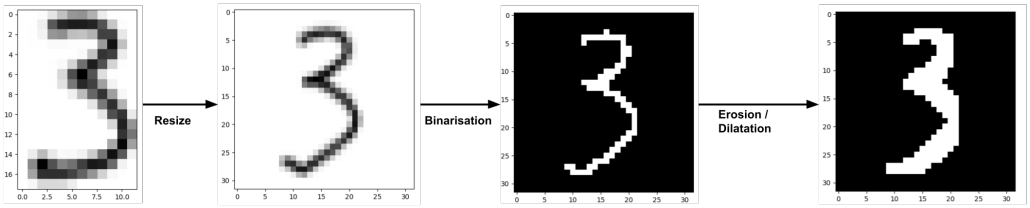

Figure 3: Chaîne de traitement

Je commence par le redimensionnement des images. J'ai choisi de les redimensionner en 32 pixels par 32 pixels, car toutes les images de la base ont une taille équivalente ou inférieure à ces dimensions.

Ensuite, nous réalisons la binarisation des images. Les images fournies sont en niveaux de gris, mais nous souhaitons travailler uniquement avec des images en noir et blanc. Pour cela, nous vérifions la valeur de chaque pixel : si son niveau de gris est inférieur à 0.5, le pixel est considéré comme noir, sinon il est blanc. Lors de cette étape de binarisation, j'inverse les couleurs, chaque pixel noir devient blanc et inversement. Cette inversion facilite l'utilisation des fonctions d'érosion et de dilatation de la bibliothèque **scipy.ndimage**.

Nous allons maintenant retoucher l'image en utilisant l'érosion et la dilatation, dans le but de développer des caractéristiques propres à chaque chiffre. La dilatation que j'utilise est basée sur une structure particulière : un carré de 3x3, tandis que l'érosion utilise une structure en croix de 2x2. Cela permet de mettre davantage l'accent sur la dilatation.

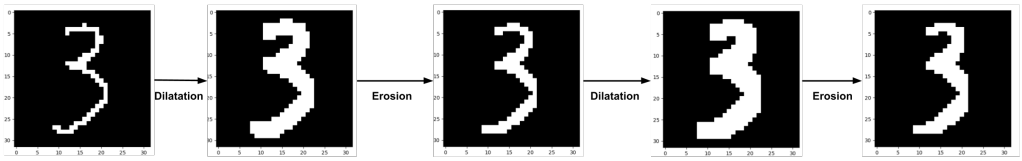

Figure 4: Application de la dilatation et de l'érosion

Je commence par appliquer une dilatation, suivie d'une érosion, puis une autre dilatation et une dernière érosion. Mon objectif est de préserver globalement la forme du chiffre tout en augmentant la surface occupée par celui-ci.

2.2 Extraction de caractéristiques

Une fois l'image traitée, je souhaite en extraire un vecteur. Pour cela, je divise l'image en 16 sections, à la fois en longueur et en largeur, ce qui donne 256 carrés de 2x2 pixels. Pour chaque carré, je compte le nombre de pixels noirs qu'il contient et j'ajoute cette valeur à une liste représentant mon vecteur.
Ces vecteurs seront utilisés pour comparer les différentes caractéristiques des images en calculant la distance euclidienne entre eux.

Une fois les distances calculées, j'utilise l'algorithme des k-plus proches voisins (k-NN) pour annalyser le chiffre de l'image. Le principe de cet algorithme est de comparer l'image avec les k vecteurs les plus proches, c'est-à-dire ceux ayant la plus petite distance euclidienne. Le chiffre associé à l'image est alors déterminé par un vote majoritaire parmi les k-plus proches voisins. Par exemple, si la majorité des voisins les plus proches correspond au chiffre 7, le programme classera l'image comme étant un 7.

3 Annalyse des Résultats

3.1 Matrice de Confusion

Afin de visualiser les résultats obtenues par l'éxécution du programme, nous pouvons réaliser une matrice de confusion regroupant l'ensemble des tests réaliser. Chaque ligne correspond à un chiffre réel et chaque colonne à un chiffre prédit. Idéalement, la matrice devrait montrer des valeurs élevées sur la diagonale (représentant les prédictions correctes) et des valeurs proches de zéro en dehors de la diagonale (indiquant peu d'erreurs).

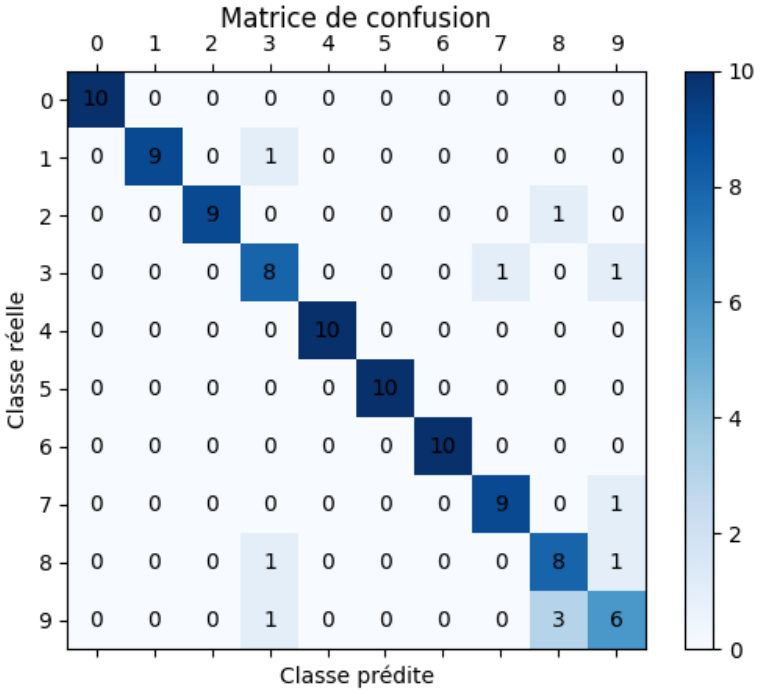

Figure 5: Matrice de Confusion

Après traitement des images et extraction des vecteurs, on obtient un taux de reconnaissance de 89%, ce qui est relativement satisfaisant. Le programme reconnaît bien les chiffres de 1 à 7 en général. Malgré cela, on observe que des erreurs surviennent notamment avec le chiffre 9. Le programme confond souvent les chiffres 3, 7 et 8 avec le chiffre 9. Ces confusions illustrent des problèmes de variance inter-classe, où les différences entre ces chiffres ne sont pas suffisamment marquées pour que l'algorithme puisse les distinguer de manière fiable.

3.2 Erreurs

Les traitements effectués précédemment permettent d'accentuer les caractéristiques des chiffres, mais dans certains cas, cela peut aussi en supprimer certaines.

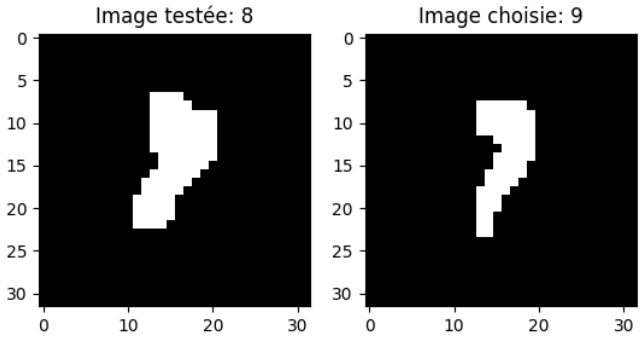

Figure 6: Confond 8 avec 9

Une caractéristique qui est perdue est le trou central des chiffres 8 et 9. Ainsi, certains chiffres 8 avec une forme particulière sont confondus avec le chiffre 9.

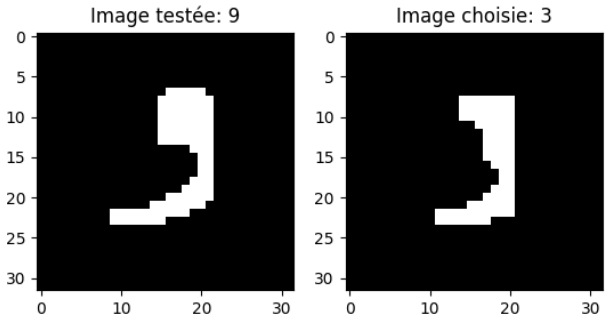

Figure 7: Confond 9 avec 3

De même, le chiffre 3 perd ses courbes distinctives et finit par ressembler à un 9.
Dans mon implémentation, je n'ai pas réussi à trouver un équilibre entre l'agrandissement des chiffres et la conservation de leurs trous distinctifs. La structure que j'ai utilisée pour la dilatation est probablement trop large, ce qui empêche de préserver ces caractéristiques importantes. Malgré cela, j'ai tout de même décidé de conserver cette structure pour la dilatation, car elle offre de bons résultats pour la reconnaissance des autres chiffres.

4 Conclusion

Pour conclure, ce projet, m'a permis de développer une méthode de reconnaissance de chiffres basée sur le traitement d'images et l'extraction de caractéristiques. En commançant par traitées l'image puis d'en extraire une valeur permettant de la comparer aux autres images afin de deviner le chiffre.

Le modèle a montré des résultats satisfaisants, avec un taux de reconnaissance de 89%. Toutefois, des erreurs ont persisté, notamment dues à la perte de certaines caractéristiques importantes, comme les trous au centre des chiffres 8 et 9, et la confusion entre les chiffres 3, 8, et 9. Ces erreurs sont en partie causées par la structure utilisée pour la dilatation, qui, bien qu'efficace pour la majorité des chiffres, change certaines formes spécifiques.

Pour améliorer ces résultats, on peut envisager d'optimiser les prétraitements, notamment en ajustant la structure utilisée pour la dilatation afin de mieux préserver les caractéristiques distinctives des chiffres.
Travailler avec une base de données plus complète permettrait également au programme de prendre en compte plusieurs variations d'un même chiffre, ce qui pourrait limiter certaines erreurs.