Études et expérimentation de la reconnaissance des chiffres

NGUYEN Van Tho - NGUYEN Quoc Khai

22 Décembre, 2013

1 Introduction

Dans ce TP, nous visons à développer un programme qui peut reconnaître des chiffres. Pour il existe plein de recherches qui donnent les résultats efficaces tels que K-plus proches voisins, Décision stumps, Classification non linéaire, SVMs, Réseaux neuraux et Réseaux de convolution[1]. Il existe aussi des méthode moins efficace comme Classification linéaire, la classification bayésienne. Pour trouver des bonnes méthodes, nous examinons quelques algorithmes qui rendent des résultats acceptables et qui sont faisables. Après ces études, nous avons choisit 2 méthodes, K-plus proches voisins et SVM. Nous expérimentons plusieurs paramètres de ces deux méthodes pour trouver la meilleure configuration.

2 Descriptions des méthodes utilisées

Dans ce TP, nous avons utilisé 2 méthodes, K-plus proches voisins et SVM

2.1 K-plus proches voisins Euclide

Cette méthode est réalisé dans des étapes suivantes.

2.1.1 Détection des blocks

Chaque image est divisée en N blocks de même taille. On vous signale que le choix du nombre de block est si important, cela influence beaucoup au résultat de reconnaissance. Après avoir testé des paramètres, on a choisi de mettre $N = 14x14 = 196$.

2.1.2 Calcul du descripteur

Un descripteur contient N éléments et une valeur d'un élément e est la somme des pixels d'un block qu'on a détecté dans la dernière étape.

2.1.3 Calcul de la distance d'Euclide entre deux descripteurs

$$D = \sqrt{\sum_{(1<=i<=N)} (e_{1i} - e_{2i})^2} \tag{1}$$

2.1.4 K-plus proches voisins (K-Nearest Neighbors)

Quand on entre une image de test, le programme calcule la distance D entre cette image avec toutes les images d'apprentissage, ils donnes K images ayant les distances les plus courtes. Choix du type le plus populaire pour le type de l'image de test. Ici, K est un paramètre d'entré. En observant le résultat des tests, nous trouvons que $K = 4$ donne le meilleur résultat.

2.2 Méthode SVM

Nous avons aussi implémenté un programme utilisé la méthode SVM. Cette méthode utilise la même structure que la méthode K-plus proches voisins. Autrement dit, l'image est divisée en N bloques. Le vecteur caractéristique est un vecteur de taille N, chaque élément de vecteur est la somme des valeurs de niveau de gris d'un bloc.

Dans ce travail, nous avons utilisé l'implémentation de SVM de OpenCV.

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0. \end{aligned}$$

Le kernel gaussien utilisé dans notre programme:

$$K(x_i, x_j) = exp(-\gamma(x_i - x_j)^2)$$

3 Présentation du programme

Ce programme contient 4 classes divisées en 8 fichiers : *channel.h channel.cpp IO.h IO.cpp training.h training.cpp control.h et control.cpp* et un fichier de la fonction *main*, *main.cpp*. Un *makefile* sert à faciliter de compiler le programme.

Ce programme a les fonctionnalités suivantes :

1. Création des descripteurs pour les images d'apprentissage ou de test et les enregistre dans un fichier.
2. A partir de deux fichiers de descripteur de test et d'apprentissage, le programme catégorise les images vers la catégorie correspondante.

Dans ce programme, aucun algorithme est réutilisé. Par contre, nous avons utilisé la librairie OpenCV pour le traitement de base.

4 Expérimentation

4.1 Évaluation

Pour évaluer le programme, nous avons utilisés deux critères: le taux de précision et le temps de calcul. Une configuration des paramètres est considérée meilleure si elle rend un résultat avec un grand taux de précision et avec un petit temps de calcul.

4.2 SVM

Une des difficultés de l'algorithme SVM est comment choisir les bonnes paramètres. Un mauvais choix de paramètres peut rendre un résultat très mauvais. Avec un kernel gaussien, on a deux paramètres à optimiser: la constance C et le gamma. Donc, pour trouver les valeurs optimales de ces paramètres, nous utilisons la méthode Grid search proposé par Chih-Wei Hsu, Chih-Chung Chang et Chih-Jen Lin [2].En effet, nous avons expérimenté les valeurs de C de 2^{-5} à 2^{15} et gamma de 2^{-19} à 2^3. De plus, pour trouver le nombre optimal de blocs, nous expérimenté plusieurs nombre de bloc: de 7 à 16.

Les bonnes valeurs de C reçues sont de 2 à 8. Les bonnes valeurs de gamma sont de 0.0001 à 0.008. Nous avons fait la deuxième série d'expérimentation avec ces valeurs de C et gamma.

4.3 Méthode KNN

Avec cette méthode, pour obtenir le bon résultat, il faut bien choisir des paramètres comme le nombre de block, le K plus proches voisins.

1. Nombre de block, ce paramètre est important. Quand le nombre de block est petit, le résultat n'est pas exact. Par contre, quand on le met très grand, la méthode peut se tromper aussi à cause des bruits dans les images.

2. K plus proches voisins, cette paramètre est aussi importante mais c'est plus facile de fixer cette paramètre. Par rapport des testes, nous trouvons que le bon est d'entre 1 et 5.

5 Analyse des résultats obtenus

5.1 Méthode KNN

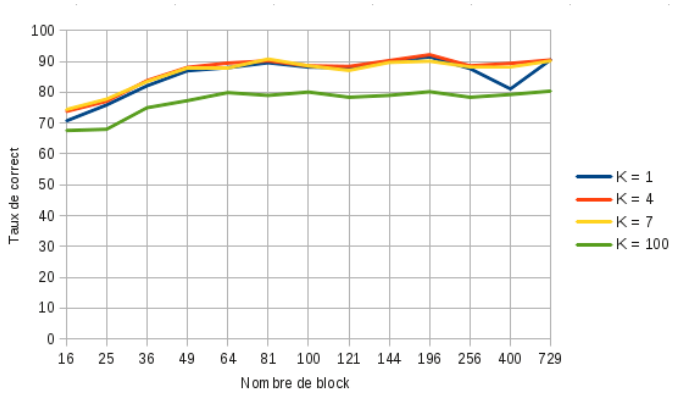

Figure 1: Taux de correct

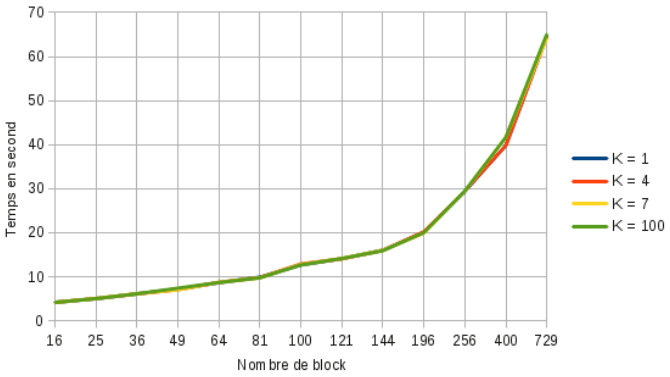

Figure 2: Temps de calcul

5.2 Méthode SVM

Comme nous avons discuté dans la section expérimentation, le point crucial de cette méthode est de bien choisir la configuration de C et gamma. Voir les figures de résultats, on constate qu'il y a un petit compromis entre le taux de précision et le temps de calcul. En fait, quand le résultat est crucial, on peut utiliser la configuration avec C = 3, gamma = 0.002 et le nombre de blocs est 14x14. Cette configuration rend le taux de précision de 94.8% et le temps de calcul est 4 seconds. Par contre,quand le temps de calcul est plus crucial, la configuration avec C = 8, gamma = 0.0004 le nombre de blocs est 7x7. Cette configuration rend le taux de précision de 92.1% et le temps de calcul est 0.8 seconds.

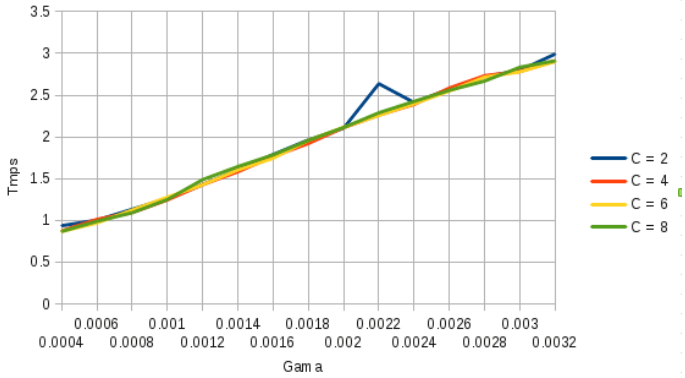

Figure 3: Taux de correct, blocs = 7x7

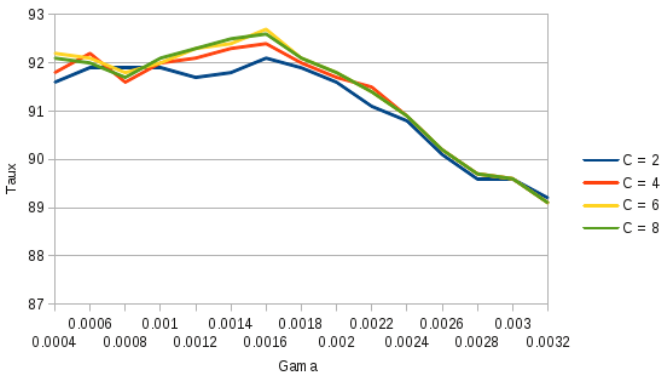

Figure 4: Temps de calcul, blocs = 7x7

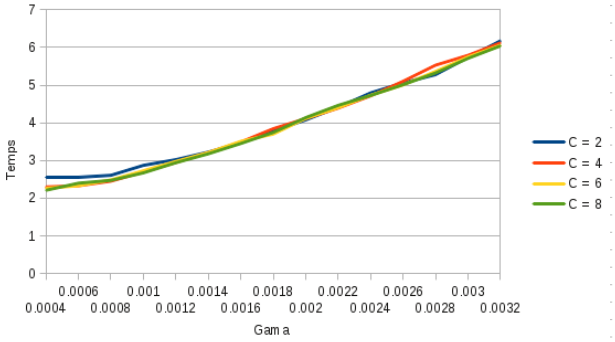

Figure 5: Taux de correct, blocs = 14x14

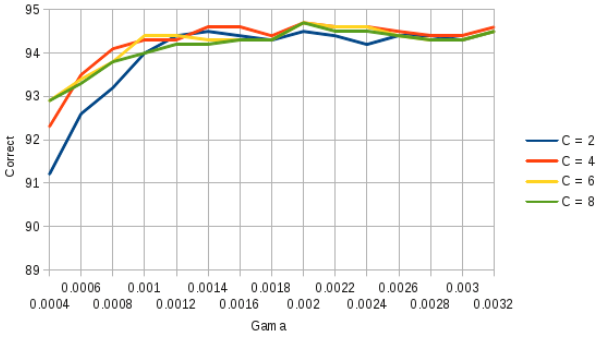

Figure 6: Temps de calcul, blocs = 14x14

5.3 Comparaison entre KNN et SVM

Nous constatons que la méthode SVM peut donner des résultats beaucoup plus vite que la méthode KNN. De plus, la précision de résultat rendu par SVM est un peu importante par rapport à celui rendu par la méthode KNN.

6 Conclusion

Dans ce projet, nous avons implémenté deux algorithmes pour la reconnaissance de chiffres manuscrit. Nous avons aussi expérimenté plusieurs configurations des paramètres pour trouver la meilleure configuration.

En général, la méthode SVM est meilleure que la méthode KNN en toutes les termes d'évaluation. Cependant, pour trouver les bonnes paramètres, on doit faire plus d'expérimentation.

References

[1] David Lowe *THE MNIST DATABASE of handwritten digits*. Yann LeCun, Courant Institute, NYU, Corinna Cortes, Google Labs, New York, Christopher J.C. Burges, Microsoft Research, Redmond, *http://www.cs.ubc.ca/ lowe/keypoints/*

[2] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin *A Practical Guide to Support Vector Classification*. Department of Computer Science National Taiwan University, Taipei 106, Taiwan *http://www.csie.ntu.edu.tw/ cjlin*