Classification d'images avec SVM, PCA et Feature Engineering

Gabriel Ferré, Geoffroy de Baritault, Yasmine EL-Ainous, Fatima Zahrae El Mokrini

Projet de 3A – Polytech Nice Sophia

Table des matières

1 Introduction et Exploration des Données

A l'issue de notre projet nous allons construire un classifieur fiable pour reconnaître des chiffres manuscrit. Le jeu de données utilisé est `digits`, fourni par la bibliothèque Scikit-learn. Il se compose de 1797 images en niveaux de gris de dimensions 8x8 pixels, chaque image étant un vecteur de 64 features correspondant à l'intensité des pixels (de 0 à 16). Chaque image est associée à un label, le chiffre qu'elle représente (de 0 à 9).

Tout d'abord, on a une distribution des classes uniformes, avec environ 180 échantillons par chiffre (Figure 1).

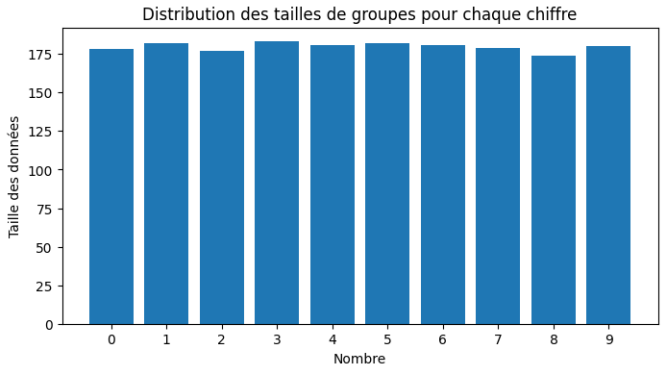

FIGURE 1 – Distribution des classes dans le jeu de données complet. L'échantillon est bien équilibré.

Maintenant, pour visualiser la structure de notre base de données, il est bon de faire une Analyse en Composantes Principales (ACP) que l'on applique aux données que nous avons normalisé (avec Min-Max Scaling). Le graphique de la variance expliquée cumulée (Figure 2) montre que plus de 90% de la variance est capturée par les 30 premières composantes principales, montrant alors qu'effectuer un PCA est pertinent. De plus, les projections 3D révèlent que certaines classes comme 0 et 4 sont naturellement bien séparées, tandis que d'autres comme 1 et 7 présentent des supperpositions, soulignant alors l'utilité de mmodèles de classification non-linéaire.

2 Pré-traitement et Feature Engineering

Au lieu, d'utiliser les 64 pixels de l'image Plutôt que d'utiliser directement les 64 pixels bruts, on effectue alors une étape appelé (*feature engineering*) permettant de construire des vecteurs plus compactes, guidant ainsi l'apprentissage du modèle. Trois types de caractéristiques ont été extraits et combinés :

— **Caractéristiques via ACP :** Les 30 premières composantes de l'ACP sont conservées. Elles représentent l'information la plus pertinente de l'image, agissant un compresseur d'information. La (Figure 3) valide que 30 composantes, conserve l'essentiel de la structure visuelle.

— **Caractéristiques Zonales :** L'image 8 par 8 est découpée en trois zones horizontales (haut, milieu, bas). La moyenne d'intensité de chaque zone est calculée, résultant en

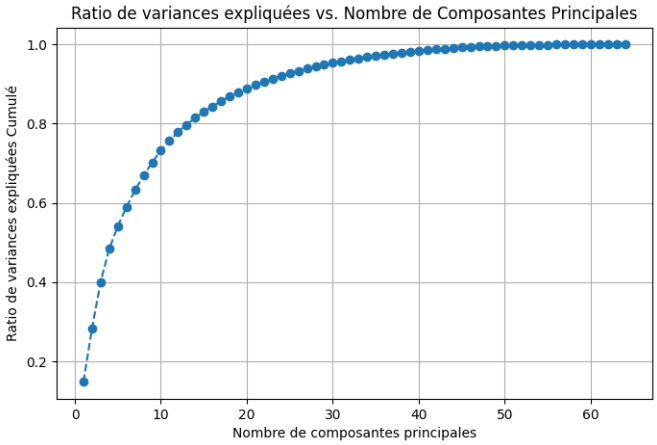

FIGURE 2 – Variance expliquée cumulée en fonction du nombre de composantes principales.

un vecteur de 3 caractéristiques. Cette technique encode une information spatiale discriminante pour des chiffres comme le '7' (zone haute vide) ou le '1' (zones haute et basse vides).

— **Caractéristiques de Contours (Filtre de Sobel) :** Le filtre de Sobel est lui appliqué pour détecter les gradients d'intensité, qui permettent de décrire les contours du chiffre. Alors, la moyenne de l'image de Sobel résultante est utilisée comme une seule caractéristique, permettant alors de mesurer la "complexité" ou la "densité" du tracé.

A savoir que l'ensemble de ces processus est implémenter dans un `Pipeline` Scikit-learn, utilisant une `FeatureUnion` pour l'extraction simultané des caractéristiques et un `StandardScaler` pour la normalisation finale. Cette approche permet de garantir la reproductibilité. Les données ont été divisées en un ensemble d'entraînement (80%) et de test (20%), un ratio assurant un apprentissage efficient pour une reconnaissance finale fiable.

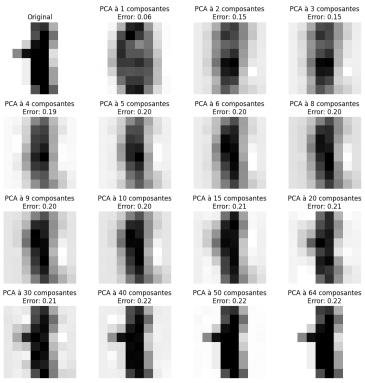

FIGURE 3 – Qualité de reconstruction d'une image en fonction du nombre de composantes ACP.

3 Expérimentation et optimisation du modèle SVM

3.1 Choix du classifieur : Support Vector Machine

Le Support Vector Machine (SVM) est un classifieur à marge maximale, particulièrement efficace pour des problèmes non linéaires lorsque couplé à un noyau. Dans notre cas, nous utilisons un noyau gaussien (RBF), permettant de projeter les données dans un espace de Hilbert de grande dimension afin de rendre les classes linéairement séparables.

3.2 Construction du pipeline de classification

Nous mettons en place une chaîne de traitement (pipeline) combinant :

— Une réduction de dimension via l'Analyse en Composantes Principales (ACP), qui permet de projeter les données sur les vecteurs propres associés à la plus grande variance. Cela a deux intérêts : la débruitisation et la réduction de la complexité.

— Des descripteurs d'intensité issus de la décomposition zonale : on découpe l'image en sous-blocs et on calcule la moyenne d'intensité par zone. Cette approche encode l'information spatiale grossière.

— Un calcul des gradients par le filtre de Sobel, qui capte l'information de texture et de bordure, cruciale pour la classification visuelle.

Ces trois types de caractéristiques sont concaténés pour former une représentation riche mais compacte de l'image.

3.3 Optimisation des hyperparamètres

Le SVM avec noyau RBF dépend de deux hyperparamètres essentiels :

— C : pénalité des erreurs de classification. Un C élevé minimise l'erreur sur l'entraînement mais augmente le risque de surapprentissage.

— γ : paramètre du noyau RBF qui contrôle l'influence d'un point d'entraînement. Un γ trop grand conduit à un modèle très complexe et peu généralisable.

Nous effectuons une recherche exhaustive (*GridSearch*) avec validation croisée pour sélectionner les meilleures valeurs de C, γ et du nombre de composantes principales k de l'ACP. Le critère d'évaluation utilisé est l'accuracy moyenne en validation croisée (5 folds).

Résultats : Les meilleurs hyperparamètres obtenus sont :

$$C = 10,\ \gamma = 1,\ k = 30$$

Accuracy moyenne en cross-validation : 99.1%

3.4 Évaluation sur le jeu de test

Le modèle entraîné avec les meilleurs paramètres est évalué sur un jeu de test indépendant. Les résultats sont les suivants :

— **Accuracy** : 98.6%

— **Erreur quadratique moyenne (MSE)** : 0.172

3.5 Influence du nombre de folds K

Nous avons étudié l'impact du choix de K dans la validation croisée. Des valeurs faibles de K (e.g. 2) donnent des estimations moins stables. Au-delà de $K = 5$, les gains deviennent marginaux.

3.6 Stratégies de classification multi-classes

Les SVM étant conçus pour la classification binaire, deux stratégies d'extension au multi-classes sont employées :

— **One-vs-One (OvO)** : $\binom{C}{2}$ classifieurs, chacun distinguant une paire de classes. Complexité quadratique en nombre de classes, mais bonne précision.

— **One-vs-Rest (OvR)** : C classifieurs binaires, chaque classifieur distinguant une classe cible du reste.

Dans notre cas, OvO a fourni une accuracy équivalente (98.6%) à celle d'OvR, avec un temps de calcul légèrement plus élevé. Cela montre que les deux approches sont viables, le choix dépendant du compromis précision/complexité.

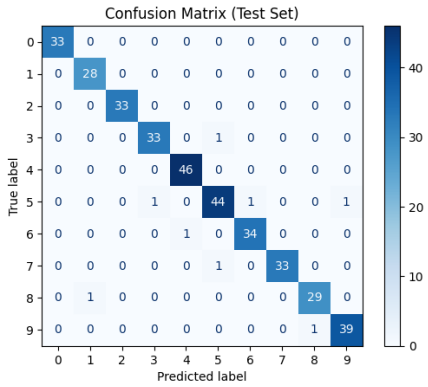

FIGURE 4 – Matrice de confusion du modèle SVM optimisé sur le jeu de test.

4 Réseaux de Neurones pour la Classification

Suite à l'analyse de l'approche SVM, nous avons étudié un modèle de réseau de neurones (NN) utilisant Keras. L'avantage de cette approche va être sa capacité à classifier directement à partir des pixels bruts, sans *feature engineering*.

Une recherche d'hyperparamètres a alors pû être menée sur le réseau de neurones et son architecture (nombre de couches, nombre de neurones), la fonction d'activation et l'optimizer. La meilleure configuration trouvée est alors un réseau avec :

— Une couche d'entrée de 64 neurones (un par pixel).

— **Trois couches cachées de 128 neurones** chacune, avec la fonction d'activation **ReLU**.

— Une couche de sortie de 10 neurones avec une activation **Softmax**.

— L'optimiseur **RMSprop** et la fonction de perte `SparseCategoricalCrossentropy`.

Comme chaque modèle il a été développé sur 50 époques atteignent alors une **accuracy de 98.9%** sur le jeu de test. Les courbes d'apprentissage (Figure 5) montrent toujout une convergence très rapide, sans signe d'overfitting, ce qui nous permet de nous rassurer sur la validité du modèle.

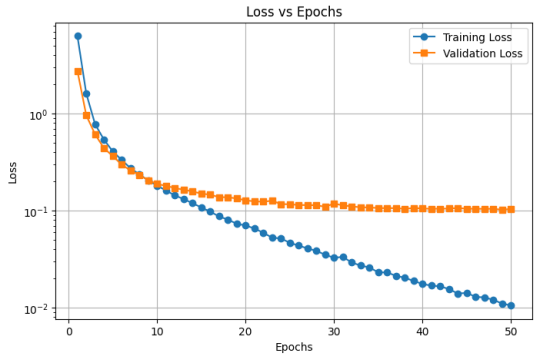

FIGURE 5 – Évolution de la fonction de perte durant l'entraînement du réseau de neurones.

5 Conclusion

Ainsi, ce projet nous a permis de comprendre et mettre en relief deux approches différentes mais pertinentes pour la classification d'imagee. Premièrement, on à étudié la façon dont nous pouvons manipuler de façon compressé les images par la méthode ACP et par la détection de contour via la méthode Sobel. Deuxièmement, nous avons implementé notre premier modèle basée sur un SVM avec des caractéristiques manuelles, a démontré qu'une bonne compréhension des données et du *feature engineering* pertinent permettent d'obtenir d'excellents résultats. Enfin, la deuxième approche repose sur un réseau de neurones, a montré la puissance des méthodes d'apprentissage profond pour extraire directement des motifs complexes sans compression où discrimination des données, étant alors aussi bon voir meilleur les approches classiques et ce facilement

TABLE 1 – Synthèse des performances des modèles finaux sur le jeu de test.

Modèle	**Ingénierie de Caractéristiques**	**Accuracy Test**
SVM + Noyau RBF	Manuelle (ACP, Zonale, Sobel)	98.6%
Réseau de Neurones Dense	Automatique (Apprentissage profond)	**98.9%**

Finalement, leréseau de neurones a été retenu comme le modèle final en raison de sa performance égal voir supérieur et de son approche plus facilement généralisable.

Par ailleurs, les limites de notre projet résident alors dans la relative simplicité du jeu de données. Pour des tâches de reconnaissances optiques plus complexes, le *feature engineering* deviendrait prohibitive. L'amélioration la plus simple à implémenter et la plus évidente serait d'explorer les **Réseaux de Neurones Convolutifs (CNN)**, qui sont l'état de l'art en traitement d'images grâce à leur capacité à exploiter les motifs locaux des pixels, et qui offriraient probablement des performances encore meilleures.