DÉPARTEMENT DE MATHÉMATIQUES ET DE GÉNIE INDUSTRIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MATHÉMATIQUES DE L'APPRENTISSAGE PROFOND
DEVOIR 1

3 octobre 2023

Directives : *Ce devoir doit être complété pour mardi le 24 octobre. Il peut être fait par équipe de deux personnes. Il doit être déposé sur le site Moodle du cours en un seul fichier, en format pdf. Vous devez utiliser le langage Julia.*

Utilisation de modules avec Julia

Des fonctionnalités peuvent être ajoutées à Julia via l'utilisation de modules. On retrouve la liste complète de ces modules via le site de Julia. Pour utiliser ces modules, on utilise la commande

```
using NomDuModule
```

Si vous n'avez jamais utilisé le module demandé, Julia va vous demander de l'installer dans un message d'erreur, à l'aide de la commande

```
import Pkg; Pkg.add("NomDuModule")
```

La prochaine fois que vous allez invoquer la commande « `using NomDuModule` », le module sera compilé. Tout sera plus rapide par la suite.

Base de données MNIST

Nous voulons concevoir un système de classification de chiffres manuscrits basé sur la SVD. La base de données MNIST sera utilisée. Nous pouvons y avoir accès avec Julia en utilisant les modules :

```
using Flux
using MLDatasets
```

On utilise ensuite la commande

```
trainX,trainY = MNIST(split=:train)[:]
```

pour aller chercher les images qui nous serviront à entraîner notre système de classification. Chaque coefficient du tenseur `trainX[i,j,k]` contient la valeur d'un réel entre 0 et 1 (le ton de gris d'un pixel). On a ainsi accès à $k = 60\,000$ images de dimensions $i \times j = 28 \times 28$ pixels comme jeu de données d'apprentissage. Le vecteur `trainY[k]` donne le chiffre associé à chaque image.

```
testX,testY = MNIST(split=:test)[:]
```

donne un jeu de 10 000 données de test.
On peut « vectoriser » les matrices des images, comme nous l'avons vu lors de la première semaine de cours, à l'aide de la commande

```
train = Flux.flatten(trainX)
```

ce qui nous donne une matrice de dimensions $784 \times 60\,000$, dans ce cas.

Traitement d'images

Nous voulons aussi être en mesure de manipuler les images provenant de ces jeux de données à l'aide de Julia. Nous utiliserons le module « `Images` » (cf. JuliaImages). Nous aurons aussi besoin du module « `Plots` » pour afficher ces images. Nous nous limiterons à l'utilisation de quelques commandes simples.
La commande `colorview` permet d'exprimer une matrice associée à une image dans un format qui peut être affiché. Par exemple, pour ce qui est de la première image du jeu de données d'apprentissage, on entre

```
Image = colorview(Gray,trainX[:,:,1]')
```

pour avoir la matrice associée à la première image en tons de gris. On peut ensuite l'afficher à l'écran :

```
plot(Gray.(Image))
```

ou la sauvegarder dans un fichier :

```
save("Image.png",Gray.(Image))
```

SVD

Finalement, le module « `LinearAlgebra` » offre la plupart des fonctionnalités liées à l'algèbre linéaire. On y retrouve la commande

```
U,sigma,V = svd(A)
```

On pourra afficher la distribution des valeurs singulières à l'aide de la commande

```
scatter(sigma,yaxis=(:log10))
```

qui provient du module « `Plots` ».

Le devoir

1. Choisir une image de `trainX` qui correspond au dernier chiffre de votre matricule (ou du matricule d'un(e) des membres de votre équipe). Afficher les valeurs singulières associées à cette image à l'aide de la commande `scatter` donnée en introduction (vous pouvez l'améliorer). Sans faire de calculs, estimer visuellement quelle approximation de rang k serait appropriée pour représenter cette image.
2. Afficher sur un graphe l'erreur des approximations de rang k provenant de cette SVD en fonction de k (utiliser la norme de Frobenius).
3. Afficher les images associées aux approximations de rang $k = 1, \ldots, 8$ de l'image choisie, provenant de cette SVD, ainsi que les images associées à chaque matrice de rang 1 utilisée. Vous devez donc donner 16 images. Commenter brièvement.
4. Après avoir complété ce petit réchauffement, voici les étapes qui nous permettront de concevoir notre système de classification de chiffres manuscrits basé sur la SVD :
   a) À l'aide de la commande `flatten` donnée en introduction, créer la matrice `train` basée sur le tenseur `trainX`. Regrouper ensuite les vecteurs colonnes des images associées à chaque chiffre $(0, 1, \ldots, 9)$ dans les matrices `train0`,...,`train9`. Ces matrices auront 784 lignes.

b) Appliquer la SVD à ces 10 matrices. Vous obtiendrez ainsi les matrices `U0`,...,`U9`, `V0`,...,`V9` ainsi que les vecteurs `sigma0`,...,`sigma9`. On espère maintenant que les vecteurs singuliers associés à `train0`,...,`train9` contiennent toute l'information qui caractérise chaque chiffre.

c) Pour le chiffre que vous avez utilisé aux numéros 1 à 3, afficher les valeurs singulières calculées à l'étape précédente. Afficher ensuite les 8 premiers vecteurs singuliers à gauche (les 8 premières colonnes de U) sous forme d'images 28×28 en tons de gris. Commenter brièvement.

d) Finalement, pour un vecteur $\boldsymbol{z}$ provenant du jeu de données de test, on va chercher la combinaison linéaire $U_k\boldsymbol{x}$ qui va donner la meilleure approximation de $\boldsymbol{z}$, où U_k contient les k premiers vecteurs singuliers à gauche pour un chiffre donné. Autrement dit, on veut calculer

$$\min_{\boldsymbol{x}\in\mathbb{R}^k} \|\boldsymbol{z} - U_k\boldsymbol{x}\|.$$

Il s'agit d'un problème de moindres carrés qu'on peut aussi exprimer comme

$$U_k^{\mathrm{T}} U_k\, \boldsymbol{x} = U_k^{\mathrm{T}} \boldsymbol{z}.$$

Puisque U_k a des colonnes orthogonales, le calcul de $\boldsymbol{x}$ peut grandement être simplifié.

Afin de vous aider à déterminer une valeur de k appropriée, toujours pour le chiffre que vous avez utilisé aux numéros 1 à 3, donner le graphe de $\|\boldsymbol{z} - U_k\boldsymbol{x}\|$ en fonction de k. Donner la valeur de k que vous utiliserez dans votre système de classification.

e) On veut finalement calculer la *matrice de confusion* associée à votre système de classification. Elle vous servira à en mesurer la qualité. On utilisera le jeu de données de test de la base de données MNIST.

Dans le cadre de notre problème, cette matrice sera de dimensions 10×10. Chaque ligne correspond à un chiffre provenant du jeu de données de test. Chaque colonne correspond à l'estimation retournée par le système de classification.

Si le système identifie bien le chiffre m, on ajoutera un 1 en position $(m\,,m)$ de la matrice. Les succès des approximations du système se retrouvent donc sur la diagonale de la matrice.

Si le système n'identifie pas correctement un chiffre, on ajoutera un 1 en position $(m\,,n)$ de la matrice, où le chiffre m aura été identifié comme étant le chiffre n.

Comment pourrait-on estimer de façon « globale » la précision du système de classification développé ?

Steven Dufour