Prelucrarea Semnalelor

Laboratorul 6.

Antrenarea Dicționarelor

1 Eliminarea zgomotului dintr-o imagine

Scopul acestei teme este implementarea și testarea algoritmului de eliminare de zgomot (denoising) dintr-o imagine pe baza antrenării unui dicționar pentru reprezentări rare. Funcțiile care vă vor ajuta la implementarea temei se regăsesc în pachetul `pip install dictlearn`. Documentația acestuia se găsește la adresa `https://unibuc.gitlab.io/graphomaly/dictionary-learning/index.html`.

Pentru eliminarea zgomotului utilizați metoda patch-urilor distincte, care nu se suprapun. În primă fază, alterați conținutul imaginii I, prin modificarea valorii fiecărui pixel: adăugați la fiecare pixel un zgomot (un scalar) cu distribuție normală de medie 0 și deviație standard σ. Veți obține astfel imaginea Inoisy. Pentru a obține semnalele $\boldsymbol{Y}$ ce vor fi utilizate ulterior în antrenare, este necesară extragerea patch-urilor din imagine. Fiecare semnal va fi un patch de dimensiune $p \times p$ vectorizat. Eliminarea zgomotului presupune învățarea unui dicționar $\boldsymbol{D}$ și a reprezentării rare corespunzătoare $\boldsymbol{X}$, care să reprezinte cât mai bine datele inițiale $\boldsymbol{Y}$. Intuiția este că dicționarul învățat va putea reprezenta numai patternuri recurente din imagini, neputând învăța zgomotul. Astfel, la reconstrucția datelor inițiale pe baza dicționarului învățat, se va elimina zgomotul.

Învățarea reprezentării rare $\boldsymbol{X}$ și a dicționarului $\boldsymbol{D}$ se realizează utilizând algoritmii OMP și K-SVD. La finalul antrenării se va calcula reprezentarea rară folosind dicționarul antrenat anterior utilizând metoda `omp`, ce găsește reprezentarea rară a semnalelor pe baza nivelului de sparsitate impus s. Obțineți astfel reprezentarea $\boldsymbol{X_c}$, în baza căreia reconstruiți imaginea Ic, care nu ar trebui să conțină zgomot.

Figura 1 prezintă imaginea originală, I, cea afectată de zgomot, Inoisy, respectiv imaginea din care a fost eliminat zgomotul, Ic.

Pentru a implementa soluția, utilizați următoarele valori ale parametrilor:

```
p = 8              # dimensiunea unui patch (numar de pixeli)
s = 6              # sparsitatea
N = 1000           # numarul total de patch-uri
```

```
n = 256             # numarul de atomi din dictionar
K =  50             # numarul de iteratii DL
sigma = 0.075       # deviatia standard a zgomotului
```

Dicționarul $\boldsymbol{D}$ se inițializează aleator și se normalizează, fiecare din cele n coloane este un vector de medie 0 și normă (lungime) 1. Dicționarul $\boldsymbol{D}$ este redundant, conține mult mai multe coloane decât cele $m = p \times p$ coloane necesare unei baze din $\mathbb{R}^m$.

1.1 Evaluarea performanței

Pentru a evalua performanța soluției folosiți măsura Peak Signal to Noise Ratio, pentru a cărui calcul puteți utiliza secvența de cod de mai jos:

```
def psnr(img1, img2):
mse = numpy.mean((img1 - img2) ** 2)
if(mse == 0):
return 0
max_pixel = 255
psnr = 20 * numpy.log10(max_pixel / numpy.sqrt(mse))
return psnr
```

Figura 1: Eliminarea zgomotului. a) Imaginea originală. b) Imaginea alterată de zgomot. c) Imaginea rezultată.

2 Ghid Python

Pentru a instala pachetul Dictionary Learning utilizați `pip install dictlearn`. Apoi încărcați `from dictlearn import DictionaryLearning` și pachetul

`from dictlearn import methods` ce conține metode pentru algoritmii de reprezentări rare și actualizare de dicționar.

Pentru a încărca o imagine utilizați `I = image.imread('nume_imagine.png')`, după ce ați importat pachetul `from matplotlib import image`.

Pentru a extrage patch-uri dintr-o imagine și apoi pentru a reconstrui o imagine din patch-uri folosiți
`from sklearn.feature_extraction.image import extract_patches_2d`, respectiv `reconstruct_from_patches_2d`. Prima primește ca parametrii imaginea și dimensiunea patch-urilor (în cazul vostru, tuple de 2 elemente, reprezentând cele 2 dimensiuni ale patch-ului), iar a doua primește ca parametrii lista de patch-uri și dimensiunea imaginii finale.

Pentru a normaliza dicționarul, utilizați
`D0 = normalize(D0, axis=0, norm='max')`, după ce ați încărcat pachetul `from sklearn.preprocessing import normalize`.

Pentru a antrena un dicționar pe datele $\boldsymbol{Y}$, utilizați secvența

```
dl = DictionaryLearning(
n_components=n,
max_iter=K,
fit_algorithm='ksvd',
n_nonzero_coefs=s,
code_init=None,
dict_init=D0,
params=None,
data_sklearn_compat=False
)
dl.fit(Y)
D = dl.D_
```

Secvența setează parametrii pentru clasa `DictionaryLearning`, apoi antrenează dicționarul pe semnalele `Y` și în final memorează dicționarul în variabila `D`.

Pentru a calcula reprezentarea rară a unui semnal, utilizați
`X, err = methods.omp(Y, D, algorithm='omp', n_nonzero_coefs=s)`.

3 Exerciții

1. Pregătirea imaginii.
   (a) Incărcați imaginea. Veți obține imaginea `I` de dimensiune $m1 \times m2$.
   (b) Adăugați zgomot cu dispersie σ imaginii, folosind secvența `Inoisy = I + sigma*np.random.randn(I.shape[0],I.shape[1])`.

(c) Extrageți patch-urile din imaginea Inoisy și memorați-le în variabila `Ynoisy`. Funcția `extract_patches_2d` va returna colecția de patch-uri, însă pentru a putea fi utilizate în continuare în antrenarea dicționarului e nevoie de ajustarea dimensiunii, respectiv de vectorizarea patch-urilor. Afișați întâi dimensiunea `Ynoisy`. Pentru a vectoriza patch-urile folosiți
`Ynoisy = Ynoisy.reshape(Ynoisy.shape[0], -1)`.
Afișați din nou dimensiunea `Ynoisy`, pentru a observa rezultatul vectorizării. Calculați media semnalelor pe linii (axa 0) și scădeți-o din Ynoisy, apoi transpuneți matricea.

(d) Selectați N patch-uri de dimensiune p la întamplare din imagine, obținând astfel semnalele `Y`.
Pentru aceasta, utilizați `numpy.random.choice(Ynoisy.shape[1],N)`.

2. Antrenarea dicționarului.

   (a) Generați un dicționar aleator și normați coloanele, obținând dicționarul `D0`.

   (b) Antrenați dicționarul `D` pornind de la dicționarul `D0` inițializat mai sus, în K iteratii utilizând patch-urile selectate `Y` ca semnale de antrenare.

3. Calcul reprezentării rare și reconstrucția imaginii.

   (a) Calculați reprezentarea rară a semnalelor `Ynoisy`, obținând `Xc`.

   (b) Obțineți patch-urile curate, `Yc`, utilizând dicționarul `D` și reprezentarea `Xc`, apoi adăugați media pe linii pe care ați scăzut-o anterior.

   (c) Reconstruiți imaginea din patch-urile `Yc`, obținând imaginea curată `Ic`.

4. Evaluarea performanței.

   (a) Vizualizați cele trei imagini (originală, alterată de zgomot și curățată de zgomot). Pentru aceasta, folosiți `plt.imshow(I)`, după ce ați încărcat pachetul `from matplotlib import pyplot as plt`.

   (b) Calculați *psnr* pentru a măsura reducerea zgomotului. Calculați atât *psnr* între imaginea originală și cea afectată de zgomot, cât și între cea originală și cea în care ați eliminat zgomotul. Dacă valoarea obținută pentru imaginea denoised este mai mare decât pentru cea noisy, metoda și-a îndeplinit scopul.