Procesarea semnalelor

Antrenarea dicționarelor. Aplicații.

Paul Irofti

Universitatea din București

Facultatea de Matematică și Informatică

Departmentul de Informatică

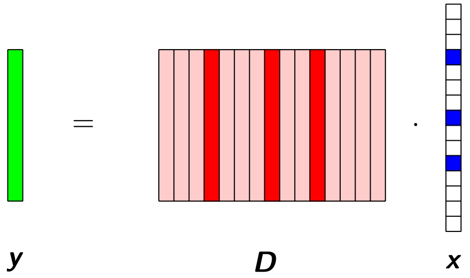
Email: paul.irofti@fmi.unibuc.ro

Reprezentarea rară

Reprezentăm un semnal $\boldsymbol{y}$ a.î.

$$\boldsymbol{y} = \boldsymbol{D}\boldsymbol{x} = \sum_{j=1}^{n} x_j \boldsymbol{d}_j = \sum_{j \in \mathcal{S}} x_j \boldsymbol{d}_j, \tag{1}$$

unde mulți x_j sunt zero, iar $\mathcal{S} = \{j \mid x_j \neq 0\}$ e suportul semnalului.



Definiție: $\boldsymbol{x}$ se numește reprezentarea rară a lui $\boldsymbol{y}$.

Problema de optimizare

Aproximare cu criteriu erorii

$$
\begin{aligned}
\min_{\boldsymbol{x}} \quad & \|\boldsymbol{x}\|_0 \\
\text{s.t.} \quad & \|\boldsymbol{y} - \boldsymbol{D}\boldsymbol{x}\| \le \varepsilon
\end{aligned}
\tag{2}
$$

unde ε este o toleranță acceptată.

Aproximare cu criteriu rar

$$
\begin{aligned}
\min_{\boldsymbol{x}} \quad & \|\boldsymbol{y} - \boldsymbol{D}\boldsymbol{x}\|^2 \\
\text{s.t.} \quad & \|\boldsymbol{x}\|_0 \le s
\end{aligned}
\tag{3}
$$

Aceste soluții se pretează foarte bine cazului în care semnalul măsurat este perturbat de un zgomot $\boldsymbol{v}$

$$
\boldsymbol{y} = \boldsymbol{D}\boldsymbol{x} + \boldsymbol{v}. \tag{4}
$$

care se pierde în urma aproximării

Algoritmul OMP

Algorithm 1: Orthogonal Matching Pursuit

Data: dictionary $\boldsymbol{D} \in \mathbb{R}^{m \times n}$
signal $\boldsymbol{y} \in \mathbb{R}^{m}$
sparsity level s
stopping error ε

Result: representation support $\mathcal{S}$, solution $\boldsymbol{x}$

Initialize $\mathcal{S} = \emptyset$, $\boldsymbol{e} = \boldsymbol{y}$
while $|\mathcal{S}| < s \text{ and } \|\boldsymbol{e}\| > \varepsilon$ **do**
  Find new index: $k = \arg\max_{j \notin \mathcal{S}} |\boldsymbol{e}^T \boldsymbol{d}_j|$
  Build new support: $\mathcal{S} \leftarrow \mathcal{S} \cup \{k\}$
  Compute new solution: $\boldsymbol{x}_{\mathcal{S}} = (\boldsymbol{D}_{\mathcal{S}}^T \boldsymbol{D}_{\mathcal{S}})^{-1} \boldsymbol{D}_{\mathcal{S}}^T \boldsymbol{y}$
  Compute new residual: $\boldsymbol{e} = \boldsymbol{y} - \boldsymbol{D}_{\mathcal{S}} \boldsymbol{x}_{\mathcal{S}}$

Apel: $\boldsymbol{x} = \text{OMP}(\boldsymbol{D}, \boldsymbol{y}, s, \varepsilon)$

Problema de antrenare

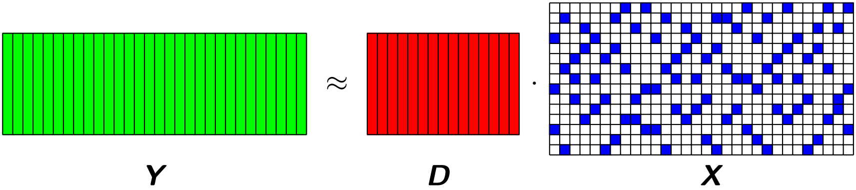
Date semnalele de antrenare $\boldsymbol{Y} \in \mathbb{R}^{m \times N}$ și s, antrenarea dicționarului $\boldsymbol{D}$ presupune rezolvarea problemei de optimizare

$$\begin{aligned} \min_{\boldsymbol{D},\boldsymbol{X}} \quad & \|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X}\|_F^2 \\ \text{s.t.} \quad & \|\boldsymbol{x}_\ell\|_0 \le s, \ \ell = 1 : N \\ & \|\boldsymbol{d}_j\| = 1, \ j = 1 : n \end{aligned} \tag{5}$$

unde variabilele sunt $\boldsymbol{D} \in \mathbb{R}^{m \times n}$ și $\boldsymbol{X} \in \mathbb{R}^{n \times N}$.

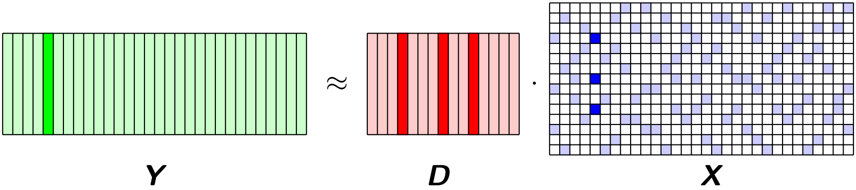
Exemplu: Problema de antrenare

Aproximarea $\boldsymbol{Y} \approx \boldsymbol{D}\boldsymbol{X}$ trebuie să fie cât mai bună.



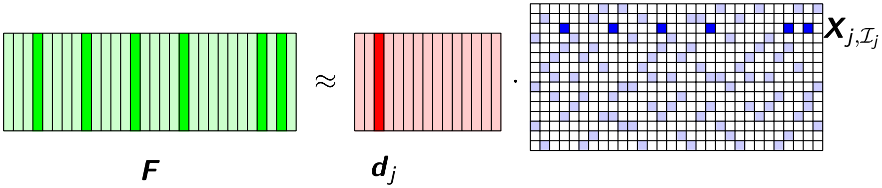
Exemplu: Problema de antrenare

Contribuția unui singur semnal



Exemplu: actualizarea atomului $\boldsymbol{d}_j$

Problema aproximării



Algoritmul K-SVD

Algorithm 2: Actualizare K-SVD

Data: signals set $\boldsymbol{Y} \in \mathbb{R}^{m \times N}$
current dictionary $\boldsymbol{D} \in \mathbb{R}^{m \times n}$
representation matrix $\boldsymbol{X} \in \mathbb{R}^{n \times N}$
Result: updated dictionary $\boldsymbol{D}$

Compute error $\boldsymbol{E} = \boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X}$
for $j = 1$ **to** n **do**
  Modify error: $\boldsymbol{F} = \boldsymbol{E}_{\mathcal{I}_j} + \boldsymbol{d}_j \boldsymbol{X}_{j,\mathcal{I}_j}$
  Compute first singular value σ_1 of $\boldsymbol{F}$ and associated singular vectors $\boldsymbol{u}_1$ and $\boldsymbol{v}_1$
  Update atom and representation: $\boldsymbol{d}_j = \boldsymbol{u}_1, \quad \boldsymbol{X}_{j,\mathcal{I}_j} = \sigma_1 \boldsymbol{v}_1^T$
  Recompute error: $\boldsymbol{E}_{\mathcal{I}_j} = \boldsymbol{F} - \boldsymbol{d}_j \boldsymbol{X}_{j,\mathcal{I}_j}$

Apel: $[\boldsymbol{D}, \boldsymbol{X}] = \text{K-SVD}(\boldsymbol{Y}, \boldsymbol{D}, \boldsymbol{X})$

Schemă de calcul cu OMP și K-SVD

Algorithm 3: Optimizare alternativă

Data: signals set $\boldsymbol{Y} \in \mathbb{R}^{m \times N}$
sparsity s
initial dictionary $\boldsymbol{D} \in \mathbb{R}^{m \times n}$
number of iterations K
Result: trained dictionary $\boldsymbol{D}$

for $k = 1 : K$ **do**
    Sparse coding: $\boldsymbol{x}_i = \mathrm{OMP}(\boldsymbol{D}, \boldsymbol{y}_i, s, \varepsilon)$, $i = 1 : N$
    Dictionary update: $[\boldsymbol{D}, \boldsymbol{X}] = \text{K-SVD}(\boldsymbol{Y}, \boldsymbol{D}, \boldsymbol{X})$

Aplicații

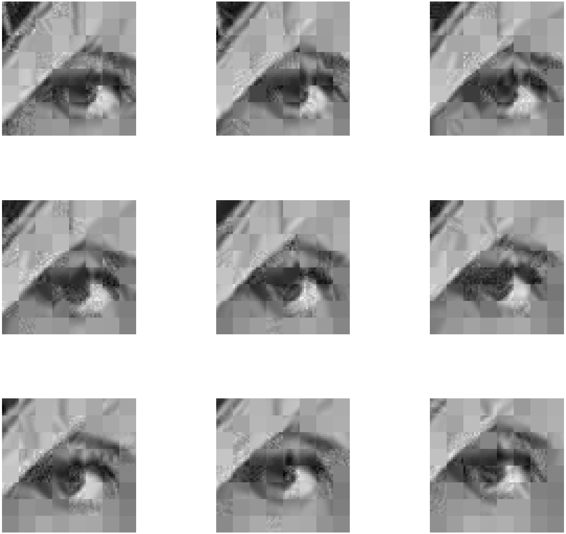
- eliminarea zgomotului (*denoising*)
- completarea unei imagini (*inpainting*)
- compresie
- clasificare
- rezumarea colecților de imagini (*image collection summarization*)
- rezumarea video (*video summarization*)
- albume foto (*photo albuming*)

Exemplu: eliminarea zgomotului

Original

Noisy

Cleaned



Overlapping Patches

Eliminare de zgomot: Obiectiv

- **Problema**: imaginea perturbată (monocromă) $\boldsymbol{I}$ cu P pixeli
- **Preprocesare**: împart imaginea în petice de $\sqrt{m} \times \sqrt{m}$ pe care le vectorizez (și eventual centrez, normez etc.)
- **Date**: semnalele perturbate (cu zgomot) $\boldsymbol{Y} \in \mathbb{R}^{m \times N}$,
- **Caut**: dicținoarul $\boldsymbol{D} \in \mathbb{R}^{m \times n}$ și reprezentările $\boldsymbol{X} \in \mathbb{R}^{n \times N}$
- **Rezultat**: $\boldsymbol{Y}_c = \boldsymbol{D}\boldsymbol{X} \approx \boldsymbol{Y}_0$
- Notații: $\boldsymbol{Y}_c$ semnalele curățate, $\boldsymbol{Y}_0$ semnalele originale

Eliminare de zgomot: Model

Presupunem că avem de a face cu zgomot alb gaussian

$$\boldsymbol{Y} = \boldsymbol{Y}_0 + \boldsymbol{V}, \tag{6}$$

și ne așteptăm să aproximăm pe $\boldsymbol{Y}$ cu antrenarea unui dicționar

$$\boldsymbol{Y} \approx \boldsymbol{D}\boldsymbol{X} = \boldsymbol{Y}_{\boldsymbol{c}}, \tag{7}$$

astfel încât reziduu $\boldsymbol{E}$ să coincidă cu zgomotul aditiv $\boldsymbol{V}$

$$\boldsymbol{E} = \boldsymbol{Y} - \boldsymbol{Y}_{\boldsymbol{c}} \approx \boldsymbol{V} \tag{8}$$

Eliminare de zgomot: Antrenarea

Procesul DL

- identifică tipare comune printre semnalele $\boldsymbol{Y}$
- aceste tipare stau la baza construcției semnalelor originale $\boldsymbol{Y}_0$
- se poate întâmpla ca unii atomi să urmeze tiparul zgomotului
- soluții: preprocesare, modificarea problemei de optimizare, structurarea dicționarului
- antrenare pe semnalele cu zgomot date
- dacă clasa din care provin semnalele cu zgomot e cunoscută: antrenare pe semnalele curate cunoscute

Eliminare de zgomot: Clasa cunoscută

Separ antrenarea de eliminarea de zgomot:

- antrenare: pe semnale curate din clasă
- denoising: reprezint semnalele cu zgomot cu dicționarul obținut
- dezavantaj: pot lipsi anumite tipare din dicționar care sunt folosite în semnalele cu zgomot
- dacă clasa este prea mare, prea generală, pot avea mai multe tipare posibile decât atomi disponibili

Eliminare de zgomot: Clasa cunoscută

Alternativă, compunerea a două dicționare

- antrenez două dicționare: extern și intern
- extern: dicționar antrenat pe semnalele curate din clasă
- intern: dicționar antrenat pe semnalele cu zgomot
- rezultat: $\boldsymbol{D}_c = [\boldsymbol{D}_e \;\; \boldsymbol{D}_i]$
- denoising: $\boldsymbol{x} = \text{OMP}(\boldsymbol{D}_c, \boldsymbol{y}, \boldsymbol{s}, \varepsilon)$

Eliminare de zgomot: Varianța zgomotului

Adesea cunoaștem canalul de comunicație și defectele aferente.

- dacă cunoaștem varianța zgomotului σ^2
- o putem folosi în criteriu de oprire OMP
- rezultă aproximări mai bune $\boldsymbol{x}$
- previne modelarea tiparului zgomotului la actualizare
- cel mai comun criteriu

$$\epsilon = C\sqrt{m}\sigma, \tag{9}$$

- m este dimensiunea semnalului și C este o constantă (denumită și *gain*)

Eliminare de zgomot: Denoising OMP

- experimentele au arătat că o alegere bună este $C = 1.15$
- criteriul devine $\epsilon = C\sqrt{m}\sigma$
- se poate ca eroarea de reprezentare să nu treacă acest prag
- ar rezulta o reprezentare cu $s = m$
- previn adăugând un criteriu suplimentar pentru $\boldsymbol{x}$ rar
- soluție comună: $\boldsymbol{x}$ să folosească cel mult $s = \frac{m}{2}$ atomi
- algoritmul OMP pentru denoising devine

$$\boldsymbol{x} = \text{OMP}\left(\boldsymbol{D}, \boldsymbol{y}, \frac{m}{2}, 1.15\sqrt{m}\sigma\right). \tag{10}$$

Eliminare de zgomot: Artefact bloc

Precum am văzut, rezultatul *denoising* pare alcătuit din blocuri.

De ce apare acest artefact?

Eliminare de zgomot: Artefact bloc

Precum am văzut, rezultatul *denoising* pare alcătuit din blocuri.

De ce apare acest artefact?
Răspuns: pentru că lucrăm cu petice vectorizate din imagine

Eliminare de zgomot: Petice distincte

Presupunem că avem o imagine pătrată monocromă

- dimensiunea imaginii este de $\sqrt{P} \times \sqrt{P}$ pixeli
- presupunem că avem de a face cu petice pătrate de $\sqrt{m} \times \sqrt{m}$
- presupunem că $\sqrt{P} \bmod \sqrt{m} = 0$
- deci putem împărți imaginea în $\sqrt{P/m} \times \sqrt{P/m}$ petice
- vecini: 4 în general, 3 pe margini, 2 în colțuri
- fiecare petic este vectorizat în $\boldsymbol{Y} \in \mathbb{R}^{m \times N}$, $N = P/m$
- în baza lui $\boldsymbol{Y}$ învățăm dicționarul $\boldsymbol{D}$
- după care aplicăm *Denoising OMP* pentru a obține $\boldsymbol{Y}_c$

Eliminare de zgomot: Petice distincte

Aplicarea *Denoising OMP* cu petice distincte

- fiecare coloană $\boldsymbol{y}_i$ este reprezentată cu OMP
- rezultă $\boldsymbol{Y}_c = \boldsymbol{DX}$, $\boldsymbol{Y}_c \in \mathbb{R}^{m \times N}$, $N = P/m$
- reconstruim imaginea inversând operația de vectorizare
- după care plasăm peticele corespunzător în imagine

Problemă

- peticele vecine au fost obținute prin apeluri distincte la OMP
- două peticele vecine pot avea inițial la graniță pixeli similari
- dar fiecare petic a folosit atomi diferiți din dicționar
- deci vor avea un suport diferit cu coeficienți diferiți
- apare fenomenul anterior, numit și *blocking effect*

Eliminare de zgomot: Petice suprapuse

Acest fenomen dispare când folosim petice suprapuse

- pornim din colțul de stânga sus al imaginii cu petice $\sqrt{m} \times \sqrt{m}$
- când am obținut petice distincte ne-am deplasat în jos pe coloană cu un pas de $p = \sqrt{m}$ până la sfârșitul coloanei
- apoi, am aplicat aceiași tehnică pe următoarea coloană aflată la $p = \sqrt{m}$ pixeli distanță la dreapta
- pentru petice suprapuse folosim un pas $p < \sqrt{m}$
- numărul total de petice va fi

$$N = \left(\left\lfloor \frac{\sqrt{P} - \sqrt{m}}{p} \right\rfloor + 1 \right)^2 \tag{11}$$

- un pixel se va repeta de cel mult $\lfloor \sqrt{m}/p \rfloor^2$ ori în $\boldsymbol{Y}$

Eliminare de zgomot: Petice suprapuse

Aplicarea *Denoising OMP* cu petice suprapuse

- fiecare coloană $\boldsymbol{y}_i$ este reprezentată cu OMP
- rezultă $\boldsymbol{Y}_c = \boldsymbol{DX}$, $\boldsymbol{Y}_c \in \mathbb{R}^{m \times N}$, N conform (11)
- la reconstrucția imaginii pixelul final va fi o medie a valorilor diferite obținute în peticele în care apare

Eliminare de zgomot: cuantificarea rezultatelor

- cei mai populari indicatori sunt PSNR și SSIM
- ambii compară semnalul curățat cu originalul
- PSNR indică diminuarea raportului dintre semnal și zgomot
- SSIM este mai apropiat de ce percepe ochiul uman

Eliminare de zgomot: PSNR

Raportul dintre puterea maximă a semnalului și a zgomotului.

$$\text{PSNR} = 20 \log_{10} \frac{\text{DR}}{\text{RMSE}}, \tag{12}$$

- DR (*dynamic range*) – raportul dintre cea mai mare și cea mai mică valoare posibilă ce poate apărea într-un semnal
- DR $=$ 255 pentru imagini monocrome de 8-biți
- RMSE (*root mean square error*) – $\frac{1}{\sqrt{mN}} \|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X}\|_F$

Eliminare de zgomot: SSIM

$$\mathrm{SSIM}(\boldsymbol{y}_0, \boldsymbol{y}_c) = \frac{(2\mu_{\boldsymbol{y}_0}\mu_{\boldsymbol{y}_c} + C_1)(2\sigma_{\boldsymbol{y}_0\boldsymbol{y}_c} + C_2)}{(2\mu^2_{\boldsymbol{y}_0} + \mu^2_{\boldsymbol{y}_c} + C_1)(\sigma^2_{\boldsymbol{y}_0} + \sigma^2_{\boldsymbol{y}_c} + C_2)}, \qquad (13)$$

- μ – media
- σ^2 – varianța
- $\sigma_{\boldsymbol{y}_0\boldsymbol{y}_c}$ – covarianța
- C_1 și C_2 – constante, aduc stabilitate numerică în urma împărțirii, se stabilesc în funcție de DR

Completarea unei imagini (*inpainting*)

Completarea unei imagini este un caz special al eliminării de zgomot.

- nu mai avem zgomot aditiv, pur și simplu lipsesc pixeli
- tipic imaginilor
  - zgârieturi
  - text suprapus
  - erori de comunicare
  - eliminare voită a unor elemente din imagine
- umplem golurile folosind informația contextuală, învecinată

Inpainting: Antrenare

- antrenăm un dicționar folosind părțile disponibile din semnale
- folosim doar peticele complete
- sau folosim și petice incomplete dar avem grijă să sărim peste părțile lipsă în momente cheie
- golurile din peticele incomplete nu pot fi considerate zerouri, de ce?

Inpainting: Antrenare

- antrenăm un dicționar folosind părțile disponibile din semnale
- folosim doar peticele complete
- sau folosim și petice incomplete dar avem grijă să sărim peste părțile lipsă în momente cheie
- golurile din peticele incomplete nu pot fi considerate zerouri, de ce?
- **Răspuns**: pentru că nu le vom putea distinge de pozițiile cu pixeli negri

Inpainting: Antrenare

- antrenăm un dicționar folosind părțile disponibile din semnale
- folosim doar peticele complete
- sau folosim și petice incomplete dar avem grijă să sărim peste părțile lipsă în momente cheie
- golurile din peticele incomplete nu pot fi considerate zerouri, de ce?
- **Răspuns**: pentru că nu le vom putea distinge de pozițiile cu pixeli negri
- putem folosi o matrice auxiliară $\boldsymbol{M} \in \{0,1\}^{m \times N}$
- $\boldsymbol{M}$ este o mască ce identifică pozițiile în care nu avem date

Inpainting: Masca

Algoritmii de reprezentare și actualizare pentru DL trebuie modificați

- dat $\boldsymbol{M}$, semnalul $\boldsymbol{y}_i$ are indicii pixelilor cunoscuți $\mathcal{I} = \boldsymbol{M}_i$
- dat s, putem reprezenta partea disponibilă $\boldsymbol{y}_{\mathcal{I}}$ cu OMP folosind doar rândurile $\mathcal{I}$ din $\boldsymbol{D}$
- această modificare se mai numește și *Masked OMP*
- la actualizare problema pentru *Masked K-SVD* devine

$$\left\| \boldsymbol{M} \odot (\boldsymbol{F} - \boldsymbol{d}\boldsymbol{x}^T) \right\|_F . \tag{14}$$

- actualizare în funcție de indicii datelor disponibile $\mathcal{M} \in \mathbb{N}^2$
- adică privim problema pe elemente: $d_i x_j = f_{ij}$, $(i, j) \in \mathcal{M}$

Inpainting: Recuperare

Dat $\boldsymbol{D}$, recuperăm semnalul $\boldsymbol{y}$ (cu pixeli cunoscuți $\mathcal{I}$) astfel:

- dacă, dat s, putem reprezenta partea disponibilă $\boldsymbol{y}_{\mathcal{I}}$ cu OMP folosind doar rândurile $\mathcal{I}$ din $\boldsymbol{D}$
- atunci, aproximăm pe $\boldsymbol{y}$ cu $\boldsymbol{D}\boldsymbol{x}$, folosind întreg dicționarul $\boldsymbol{D}$

Inpainting: Recuperarea unei imagini

Un semnal din $\boldsymbol{Y}$ este un petec vectorizat

- ordinea în care recuperăm peticele contează. De ce?

Inpainting: Recuperarea unei imagini

Un semnal din $\boldsymbol{Y}$ este un petec vectorizat

- ordinea în care recuperăm peticele contează. De ce?
- **Răspuns**: pixelii recuperați vor fi folosiți în calcularea și recuperarea următorilor pixeli lipsă

Inpainting: Recuperarea unei imagini

Un semnal din $\boldsymbol{Y}$ este un petec vectorizat

- ordinea în care recuperăm peticele contează. De ce?
- **Răspuns**: pixelii recuperați vor fi folosiți în calcularea și recuperarea următorilor pixeli lipsă
- Cum abordăm o gaură, un bloc mare de pixeli lipsă?

Inpainting: Recuperarea unei imagini

Un semnal din $\boldsymbol{Y}$ este un petec vectorizat

- ordinea în care recuperăm peticele contează. De ce?
- **Răspuns**: pixelii recuperați vor fi folosiți în calcularea și recuperarea următorilor pixeli lipsă
- Cum abordăm o gaură, un bloc mare de pixeli lipsă?
- **Răspuns**: brodăm de la exteriorul găurii către interior

Inpainting: Recuperarea unei imagini

Un semnal din $\boldsymbol{Y}$ este un petec vectorizat

- ordinea în care recuperăm peticele contează. De ce?
- **Răspuns**: pixelii recuperați vor fi folosiți în calcularea și recuperarea următorilor pixeli lipsă
- Cum abordăm o gaură, un bloc mare de pixeli lipsă?
- **Răspuns**: brodăm de la exteriorul găurii către interior
- putem folosi petice suprapuse?

Inpainting: Recuperarea unei imagini

Un semnal din $\boldsymbol{Y}$ este un petec vectorizat

- ordinea în care recuperăm peticele contează. De ce?
- **Răspuns**: pixelii recuperați vor fi folosiți în calcularea și recuperarea următorilor pixeli lipsă
- Cum abordăm o gaură, un bloc mare de pixeli lipsă?
- **Răspuns**: brodăm de la exteriorul găurii către interior
- putem folosi petice suprapuse?
- **Răspuns**: da, dar numărul de petice în care apare același pixel este limitat; suntem constrânși să folosim petice în care lipsesc doar câțiva pixeli

Inpainting: Recuperarea unei imagini

Un semnal din $\boldsymbol{Y}$ este un petec vectorizat

- ordinea în care recuperăm peticele contează. De ce?
- **Răspuns**: pixelii recuperați vor fi folosiți în calcularea și recuperarea următorilor pixeli lipsă
- Cum abordăm o gaură, un bloc mare de pixeli lipsă?
- **Răspuns**: brodăm de la exteriorul găurii către interior
- putem folosi petice suprapuse?
- **Răspuns**: da, dar numărul de petice în care apare același pixel este limitat; suntem constrânși să folosim petice în care lipsesc doar câțiva pixeli
- **Soluție**: putem face mai multe iterații asupra aceluiași petec

Inpainting: Exemplu 30% pixeli disponibili