

Sketching Datasets for Large-Scale Learning

Large-Scale High-Dimensional Clustering with Fast Sketching

Lucy Attwood, Solune Denis, Baptiste Genest, Romane Giard
(M2 Maths en action)

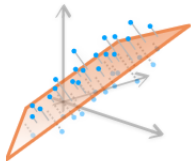
Auteurs des articles : Rémi Gribonval, Antoine Chatalic, Nicolas Keriven, Vincent Schellekens, Laurent Jacques, Philip Schniter

18/03/2024

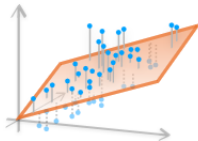
Université Claude Bernard  Lyon 1

Problématiques du machine learning

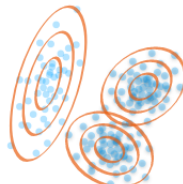
Exemples de tâches courantes en machine learning :



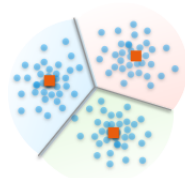
(a) PCA



(b) Linear regression



(c) Mixture modeling



(d) Clustering

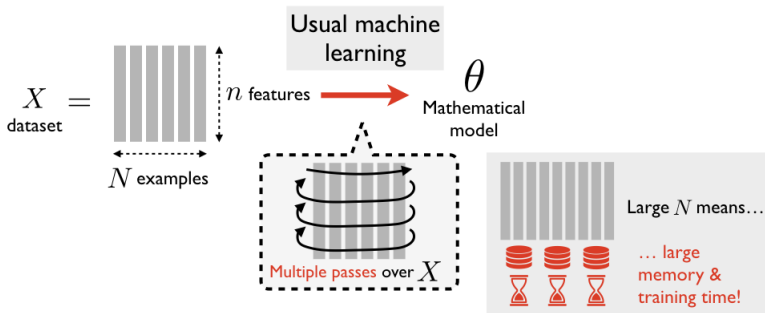
Quelques problèmes :

- Le nombre de points à considérer peut être immense et en (très) grande dimension, alors que l'information à retrouver est très petite en comparaison.
- Situation de Data Flow : la plupart des algorithmes de ML ne sont pas adaptés si on reçoit des données en continu.
- Privacy : peut-on garantir qu'il est impossible de trouver des informations sur des échantillons du dataset ?

Une idée pour résoudre ces problèmes

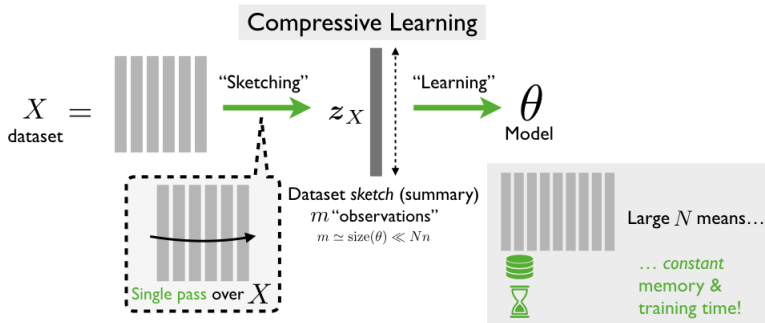
A quel point est-il possible de compresser des données ? Est-il possible d'apprendre sur des données compressées ? Si oui, *comment* les compresser ?

Une solution : le sketching



Pipeline classique de Machine Learning
(source de l'image : site de Vincent Schellekens)

Une solution : le sketching



Pipeline de Sketching
(source de l'image : site de Vincent Schellekens)

Plan

- 1 Opérateur de sketching
- 2 Compressive learning
- 3 Algorithmes
- 4 Coût calculatoire et résultats expérimentaux
 - Complexité algorithmique
 - Tests sur MNIST, problème des K-means

Objectif et différence avec le ML classique

Dans un contexte de Machine Learning, on cherche un modèle qui reproduise le plus possible les données :

$$\min_{\theta} ||A\theta - X||^2$$

Objectif et différence avec le ML classique

Dans un contexte de Machine Learning, on cherche un modèle qui reproduise le plus possible les données :

$$\min_{\theta} ||A\theta - X||^2$$

Le Compressive learning cherche un modèle p_X dont le sketch est le plus proche possible de celui des données :

$$\min_{p_X} ||\mathcal{A}(p_X) - \mathcal{A}(X)||^2$$

Comment choisir $\mathcal{A}$?

L'opérateur de Sketching $\mathcal{A}$ est choisi comme l'espérance d'une fonction **non-linéaire** Φ des données $\mathcal{X} \sim p_X$

$$\tilde{z} := \mathcal{A}(\mathcal{X}) = \frac{1}{n} \sum_{i=1}^n \Phi(x_i)$$

Comment choisir $\mathcal{A}$?

L'opérateur de Sketching $\mathcal{A}$ est choisi comme l'espérance d'une fonction **non-linéaire** Φ des données $\mathcal{X} \sim p_X$

$$\tilde{z} := \mathcal{A}(\mathcal{X}) = \frac{1}{n} \sum_{i=1}^n \Phi(x_i)$$

En passant à la limite on peut définir le sketch du modèle p_X :

$$z = \mathcal{A}(p_X) := \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n \Phi(x_i) \stackrel{\text{w.p.1}}{=} \mathbb{E}(\Phi(X))$$

Linéarité par rapport aux densités

Bien que $\mathcal{A}$ ne soit pas linéaire par rapport aux données :

$$\tilde{z} := \mathcal{A}(\mathcal{X}) = \frac{1}{n} \sum_{i=1}^n \Phi(x_i)$$

Linéarité par rapport aux densités

Bien que $\mathcal{A}$ ne soit pas linéaire par rapport aux données :

$$\tilde{z} := \mathcal{A}(\mathcal{X}) = \frac{1}{n} \sum_{i=1}^n \Phi(x_i)$$

Il l'est en les densités !

En effet pour une loi de mélange, si $X \sim p_X = \alpha p_{X_1} + (1 - \alpha)p_{X_2}$, on a :

$$\mathcal{A}(p_X) = \mathbb{E}_{X \sim p_X} [\Phi(X)] = \alpha \mathbb{E}_{X_1 \sim p_{X_1}} [\Phi(X_1)] + (1 - \alpha) \mathbb{E}_{X_2 \sim p_{X_2}} [\Phi(X_2)]$$

La non-linéarité permet une forme de mémoire.

Pas trop de pertes d'informations

Dans un contexte linéaire, on a la Propriété d'Isométrie Restreinte :

Lemme de johnson-lindenstrauss

Pour une matrice aléatoire A , on a, avec grande proba :

$$\frac{1}{1+\delta} \|Ax - Ax'\|_2^2 \leq \|x - x'\|_2^2 \leq \frac{1}{1-\delta} \|Ax - Ax'\|_2^2$$

Pas trop de pertes d'informations

Dans un contexte linéaire, on a la Propriété d'Isométrie Restreinte :

Lemme de johnson-lindenstrauss

Pour une matrice aléatoire A , on a, avec grande proba :

$$\frac{1}{1+\delta} \|Ax - Ax'\|_2^2 \leq \|x - x'\|_2^2 \leq \frac{1}{1-\delta} \|Ax - Ax'\|_2^2$$

Dans le contexte du compressive learning, on considère une **LRIP sur des opérateurs sur des distributions** :

LRIP sur $\mathcal{A}$

Sous certaines conditions, il existe une constante C_0 telle que :

$$d(p_\theta, p_{\tilde{\theta}}) \leq C_0 \|\mathcal{A}(p_\theta) - \mathcal{A}(p_{\tilde{\theta}})\|_2$$

où $d(p_X, p'_X)$, est une distance entre les distributions qui dépend du problème considéré.

Garanties

- Quand la LRIP est vérifiée, on a un contrôle sur l'excès de risque lié à l'apprentissage sur des données compressées.
- Pour les opérateurs de sketching considérés, on peut choisir m tel que la LRIP soit vérifiée **avec grande probabilité**.

Plan

- 1 Opérateur de sketching
- 2 Compressive learning**
- 3 Algorithmes
- 4 Coût calculatoire et résultats expérimentaux
 - Complexité algorithmique
 - Tests sur MNIST, problème des K-means

Comment choisir Φ ?

Ça dépend du problème..

Exemple (Analyse en composantes principales)

Définir une base avec k vecteurs propres principaux de la matrice d'auto-corrélation empirique

$$\hat{R} = \frac{1}{n} \sum_{i=1}^n x_i x_i^T = \frac{1}{n} \sum_{i=1}^n \Phi(x_i) = \mathcal{A}(\mathcal{X})$$

avec $\Phi(\mathcal{X}) = \text{vec}(\mathcal{X}\mathcal{X}^T) = (x_1x_1, x_2x_1, \dots, x_dx_d)^T$. I.e $\hat{R}$ est un sketch.

Comment choisir Φ ?

Pour du clustering, ils utilisent une transformée de Fourier aléatoire des données :

$$\mathcal{A}(x) = \frac{1}{n} \sum_{i=1}^n \exp(-j2\pi w_j^\perp x_i) = \frac{1}{n} \sum_{i=1}^n \Phi(x_i)$$

avec $\Phi(x) = \exp(-j2\pi Wx)$ et $W = (w_1, \dots, w_m) \in \mathbb{R}^{m \times d}$ une matrice aléatoire.

Apprendre depuis un sketch

Le choix de la loss est extrêmement naturel :

On souhaite trouver les paramètres du modèle tels que le sketch du modèle soit le plus proche possible du sketch des données.

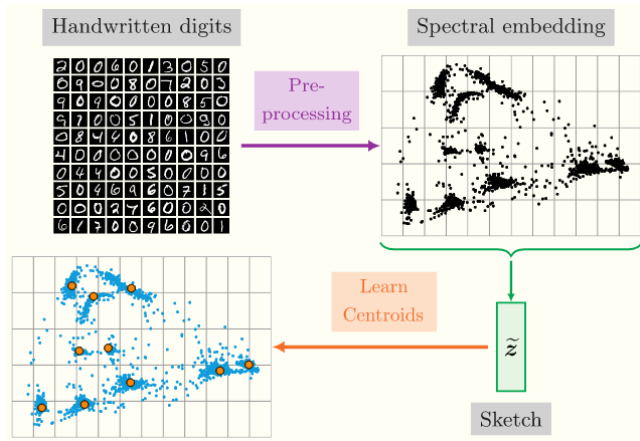
Loss associée au sketch

On cherche θ qui minimise :

$$C(\theta) := \|\mathcal{A}(\mathcal{X}) - \mathcal{A}(p_\theta)\|^2$$

Apprendre depuis un sketch : exemple du clustering

Déroulement d'une tâche de clustering en Compressive Learning :

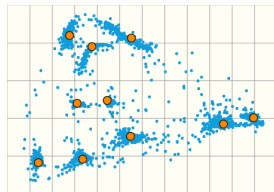


Apprendre depuis un sketch : exemple du clustering

Loss sur X

On souhaite trouver les k centroïdes de clusters $\{c_l\}_{l=1}^k$ qui minimise

$$\sum_{i=1}^n \min_l \|x_i - c_l\|^2$$



Loss associée au sketch

On souhaite trouver le k centroïdes de clusters $\{c_l\}_{l=1}^k$ qui minimise

$$C(c_l, \tilde{z}) := \|\tilde{z} - \sum_{l=1}^k \alpha_l \Phi(c_l)\|^2$$

avec $\tilde{z} = \mathcal{A}(\mathcal{X})$ et $\{\alpha_l\}_{l=1}^k$ des poids tel que $\sum_{l=1}^k \alpha_l = 1$.

Plan

- 1 Opérateur de sketching
- 2 Compressive learning
- 3 Algorithmes**
- 4 Coût calculatoire et résultats expérimentaux
 - Complexité algorithmique
 - Tests sur MNIST, problème des K-means

Apprentissage parcimonieux

Si on suppose que les données sont explicables par très peu de paramètres, il existe des algorithmes très efficaces et robustes pour apprendre, par exemple le problème du LASSO :

$$\min_{\theta} \|A\theta - X\|_2^2 + \|\theta\|_1$$

C'est un problème **convexe** ! Par exemple si on cherche à reproduire un signal X à partir d'un dictionnaire A .

Apprentissage parcimonieux

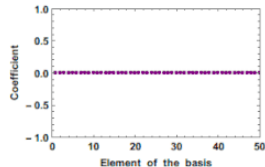
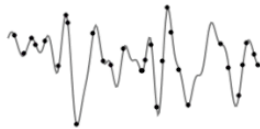
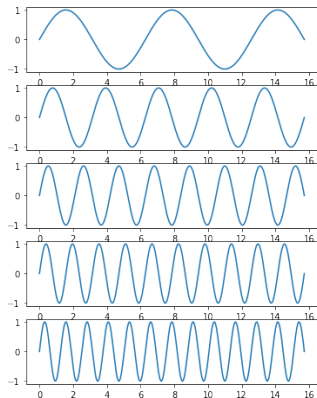
Si on suppose que les données sont explicables par très peu de paramètres, il existe des algorithmes très efficaces et robustes pour apprendre, par exemple le problème du LASSO :

$$\min_{\theta} \|A\theta - X\|_2^2 + \|\theta\|_1$$

C'est un problème **convexe** ! Par exemple si on cherche à reproduire un signal X à partir d'un dictionnaire A .

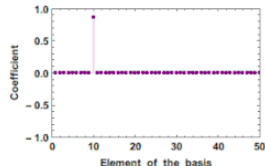
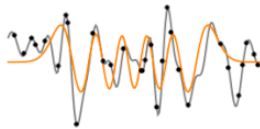
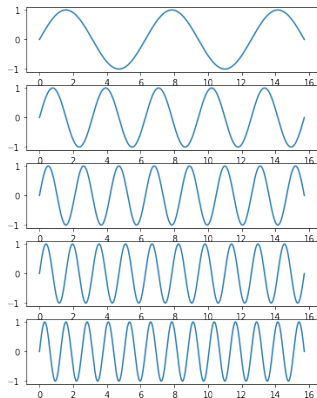
Une méthode de résolution de ce problème est l'algorithme Orthogonal Matching Pursuit (OMP) pour une solution θ k -sparse.

Rappels sur Orthogonal Matching Pursuit



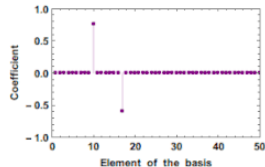
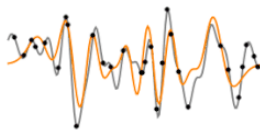
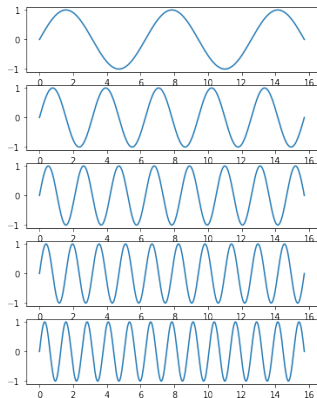
Reconstruction d'un signal avec OMP
(source de l'image : Wikipédia)

Rappels sur Orthogonal Matching Pursuit



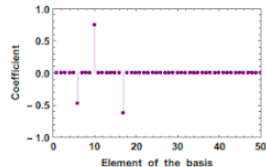
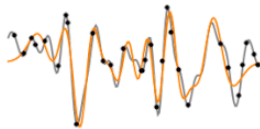
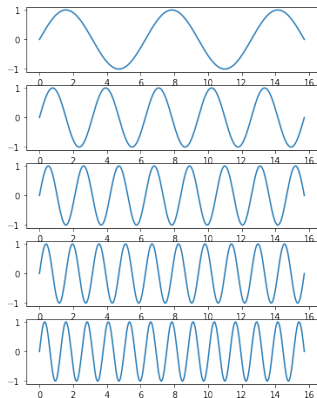
Reconstruction d'un signal avec OMP
(source de l'image : Wikipédia)

Rappels sur Orthogonal Matching Pursuit



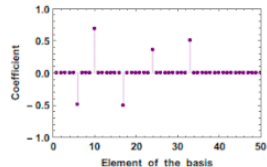
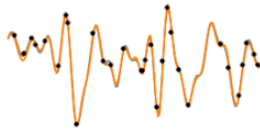
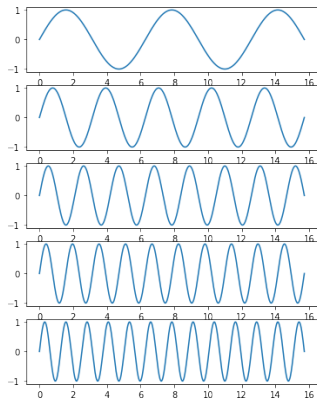
Reconstruction d'un signal avec OMP
(source de l'image : Wikipédia)

Rappels sur Orthogonal Matching Pursuit



Reconstruction d'un signal avec OMP
(source de l'image : Wikipédia)

Rappels sur Orthogonal Matching Pursuit



Reconstruction d'un signal avec OMP
(source de l'image : Wikipédia)

Orthogonal Matching Pursuit

On a $\mathbf{b} = \mathbf{A}\mathbf{x}$, on veut retrouver $\mathbf{x}$.

Algorithm 1: OMP($\mathbf{A}, \mathbf{b}$)

Input: $\mathbf{A}, \mathbf{b}$

Result: $\mathbf{x}_k$

Initialization $\mathbf{r}_0 = \mathbf{b}, \Lambda_0 = \emptyset$;

Normalize all columns of $\mathbf{A}$ to unit L_2 norm;

Remove duplicated columns in $\mathbf{A}$;

for $k = 1, 2, \dots$ **do**

Step-1. $\lambda_k = \underset{j \notin \Lambda_{k-1}}{\operatorname{argmax}} |\langle \mathbf{a}_j, \mathbf{r}_{k-1} \rangle|$;

Step-2. $\Lambda_k = \Lambda_{k-1} \cup \{\lambda_k\}$;

Step-3. $\mathbf{x}_k(i \in \Lambda_k) = \underset{\mathbf{x}}{\operatorname{argmin}} \|\mathbf{A}_{\Lambda_k} \mathbf{x} - \mathbf{b}\|_2, \quad \mathbf{x}_k(i \notin \Lambda_k) = 0$;

Step-4. $\hat{\mathbf{b}}_k = \mathbf{A} \mathbf{x}_k$;

Step-5. $\mathbf{r}_k \leftarrow \mathbf{b} - \hat{\mathbf{b}}_k$;

end

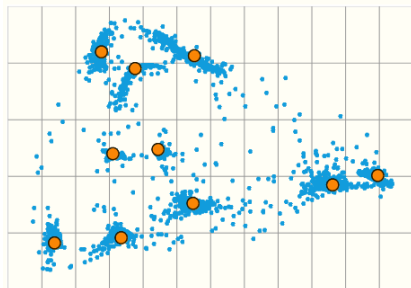
OMP dans un contexte continu

Adaptation possible car le sketch est linéaire pour notre modèle :

$$\|Ax - y\|^2 \rightsquigarrow \|\mathcal{A}(\sum \alpha_i \delta_{c_i}) - \tilde{z}\|^2$$

"OMP continu"

- Le signal qu'on souhaite reconstruire est le sketch des données : $\tilde{z}$
- Modèle statistique $\iff$
"Dictionnaire continu" :
 $\{\mathcal{A}(\delta_c), c \in \Omega\}$
- La loss n'est plus convexe ! Donc approche gloutonne moins efficace.



Ω

OMP version Compressive Learning (CLOMP)

Algorithm 1: CLOMPR for K -means (CKM)

Data: Sketch $\hat{\mathbf{z}}$, frequencies Ω , parameter K , bounds $\mathbf{l}$, $\mathbf{u}$

Result: Centroids C , weights α

$\hat{\mathbf{r}} \leftarrow \hat{\mathbf{z}}; C \leftarrow \emptyset;$

for $t \leftarrow 1$ **to** $2K$ **do**

Step 1: Find a new centroid

$\mathbf{c} \leftarrow \text{maximize}_{\mathbf{c}} \left(\text{Re} \left\langle \frac{\mathcal{A}\delta_{\mathbf{c}}}{\|\mathcal{A}\delta_{\mathbf{c}}\|}, \hat{\mathbf{r}} \right\rangle, \mathbf{l}, \mathbf{u} \right)$

end

Step 2: Expand support

$C \leftarrow C \cup \{\mathbf{c}\}$

end

Step 4: Project to find α

$\alpha \leftarrow \arg \min_{\alpha \geq 0} \left\| \hat{\mathbf{z}} - \sum_{k=1}^{|C|} \alpha_k \mathcal{A}\delta_{\mathbf{c}_k} \right\|$

end

Step 5: Global gradient descent

$C, \alpha \leftarrow$

$\text{minimize}_{C, \alpha} \left(\left\| \hat{\mathbf{z}} - \sum_{k=1}^{|C|} \alpha_k \mathcal{A}\delta_{\mathbf{c}_k} \right\|, \mathbf{l}, \mathbf{u} \right)$

end

 Update residual: $\hat{\mathbf{r}} \leftarrow \hat{\mathbf{z}} - \sum_{k=1}^{|C|} \alpha_k \mathcal{A}\delta_{\mathbf{c}_k}$

end

OMP version Compressive Learning (CLOMP)

CLOMPR : Compressive
Learning OMP with
Replacement

Algorithm 1: CLOMPR for K -means (CKM)

Data: Sketch $\hat{z}$, frequencies Ω , parameter K , bounds l, u

Result: Centroids C , weights α

$\hat{r} \leftarrow \hat{z}; C \leftarrow \emptyset;$

for $t \leftarrow 1$ **to** $2K$ **do**

Step 1: Find a new centroid

$c \leftarrow \underset{c}{\text{maximize}} \left(\text{Re} \left\langle \frac{\mathcal{A}\delta_c}{\|\mathcal{A}\delta_c\|}, \hat{r} \right\rangle, l, u \right)$

end

Step 2: Expand support

$C \leftarrow C \cup \{c\}$

end

Step 3: Enforce sparsity by Hard Thresholding if $t > K$

if $|C| > K$ **then**

$\beta \leftarrow \arg \min_{\beta \geq 0} \left\| \hat{z} - \sum_{k=1}^{|C|} \beta_k \frac{\mathcal{A}\delta_{c_k}}{\|\mathcal{A}\delta_{c_k}\|} \right\|$

 Select K largest entries $\beta_{i_1}, \dots, \beta_{i_K}$

 Reduce the support $C \leftarrow \{c_{i_1}, \dots, c_{i_K}\}$

end

end

Step 4: Project to find α

$\alpha \leftarrow \arg \min_{\alpha \geq 0} \left\| \hat{z} - \sum_{k=1}^{|C|} \alpha_k \mathcal{A}\delta_{c_k} \right\|$

end

Step 5: Global gradient descent

$C, \alpha \leftarrow$

$\underset{C, \alpha}{\text{minimize}} \left(\left\| \hat{z} - \sum_{k=1}^{|C|} \alpha_k \mathcal{A}\delta_{c_k} \right\|, l, u \right)$

end

 Update residual: $\hat{r} \leftarrow \hat{z} - \sum_{k=1}^{|C|} \alpha_k \mathcal{A}\delta_{c_k}$

end

Plan

- 1 Opérateur de sketching
- 2 Compressive learning
- 3 Algorithmes
- 4 Coût calculatoire et résultats expérimentaux
 - Complexité algorithmique
 - Tests sur MNIST, problème des K-means

Quel gain par rapport à K-means ?

Une fois le sketch calculé :

Performances

- On observe que la performance de CLOMP ne dépend pas de la taille du jeu de données !
- Quand la taille du sketch devient suffisamment grande on retrouve les performances de K-means.

Les auteurs montrent ces complexités, avec k le nombre de clusters, n et d respectivement le nombre et la dimension des données :

Complexités en temps

K-means : $\mathcal{O}(iknd)$ (i itérations)

CLOMP : $\mathcal{O}(k^3d^2)$ en ayant pris $m = \mathcal{O}(kd)$ (optimum empirique)

Et le coût du sketching ?

Les performances des algorithmes qui travaillent sur le sketch, une fois celui-ci calculé, sont très compétitives mais le calcul du sketch peut être prohibitif !

En effet, la complexité du calcul naïf de z est en $\mathcal{O}(nkd^2)$, ce qui en théorie rend l'algorithme global moins performant que K-means, **mais** :

Et le coût du sketching ?

Les performances des algorithmes qui travaillent sur le sketch, une fois celui-ci calculé, sont très compétitives mais le calcul du sketch peut être prohibitif !

En effet, la complexité du calcul naïf de z est en $\mathcal{O}(nkd^2)$, ce qui en théorie rend l'algorithme global moins performant que K-means, **mais** :

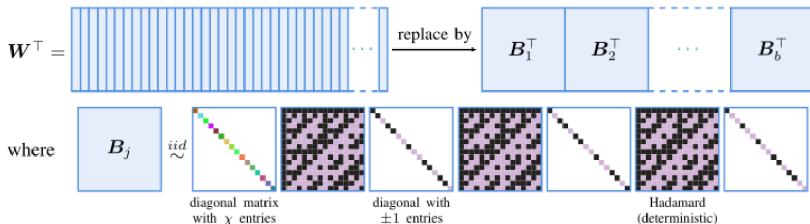
Avantages calculatoires

- Le sketching peut-être trivialement massivement parallélisé.
- La mise à jour du sketch suite à l'ajout d'une nouvelle donnée et la ré-exécution de CLOMP est très rapide (Data-Flow).
- On peut remplacer le tirage de W et la multiplication matrice-vecteurs en utilisant des matrices aléatoires structurées pour accélérer les calculs !

Fast Sketching

Rappel : $\Phi(x) = \exp(-j2\pi Wx)$, où $W = (w_j)_{1 \leq j \leq m} \in \mathbb{R}^{m \times d}$ où on choisit classiquement $w_j \sim |\mathcal{N}(0, \frac{1}{\sigma^2})| \times \mathcal{U}(\mathbb{S}^{d-1})$.

On remplace le tirage d'une grande matrice aléatoire, dense, par une **matrice aléatoire structurée** qui est un produit de matrices simples :



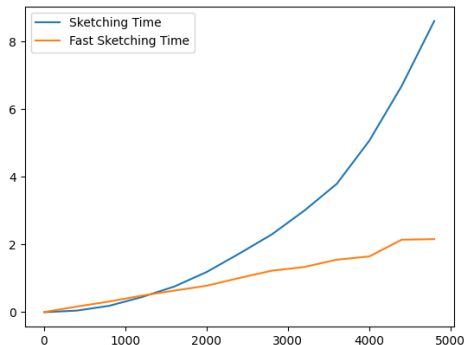
Matrice de Hadamard de taille 2^q : $H_q = \frac{1}{\sqrt{2}} \left(\begin{array}{c|c} H_{q-1} & H_{q-1} \\ \hline H_{q-1} & -H_{q-1} \end{array} \right), H_1 = (1)$.

Fast Sketching

Cette construction permet une complexité avantageuse. Avec m la taille du sketch, d la dimension des données :

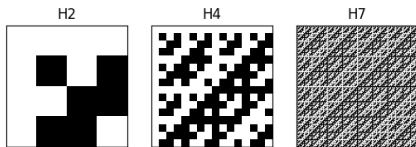
- génération d'une matrice diagonale, et calcul du produit en $\mathcal{O}(d)$;
- produit avec une matrice de Hadamard de complexité en $d \log(d)$.

Réduit la complexité du calcul du sketch à $\mathcal{O}(nm \log(d))$ au lieu de $\mathcal{O}(nmd)$.

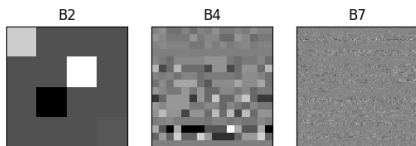


Temps du sketching en fonction de la dimension d , taille de sketch $m = 15 \times d$

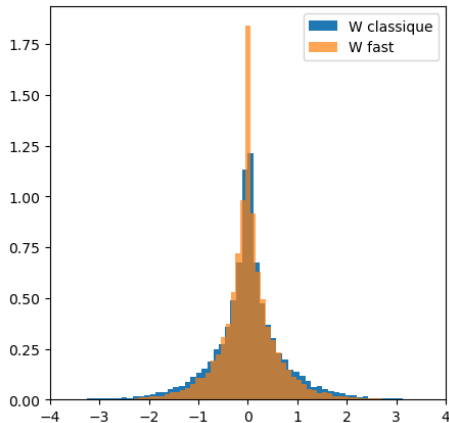
Fast Sketching



Matrices de Hadamard



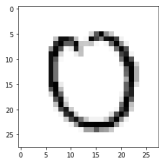
Matrices aléatoires structurées



Histogramme de W , pour le sketch ordinaire et la version "fast"

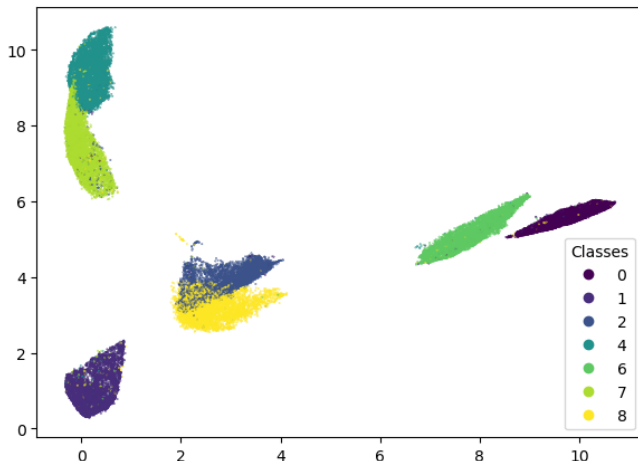
Jeu de données et problème

MNIST → Suppression de 3 classes + UMAP : projection non linéaire en dimension 20.



Nombre de
données :
42 500

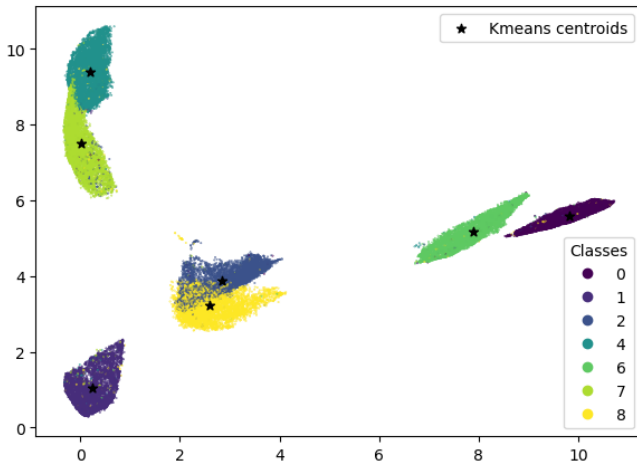
But : retrouver
7 centroïdes



MNIST, projection avec UMAP (2 premières dimensions)

Clustering avec K-means

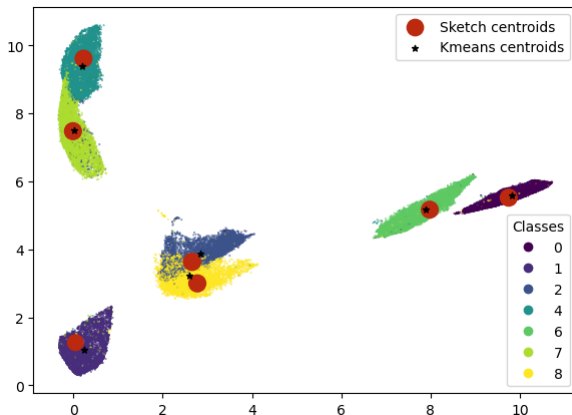
On mesure la reconstruction des centroïdes avec le SSE:= $\sum_{i=1}^n \min_l \|x_i - c_l\|^2$.



MNIST, centroïdes avec K-means (SSE= 9.3×10^4)

Clustering avec le sketch et CLOMPR

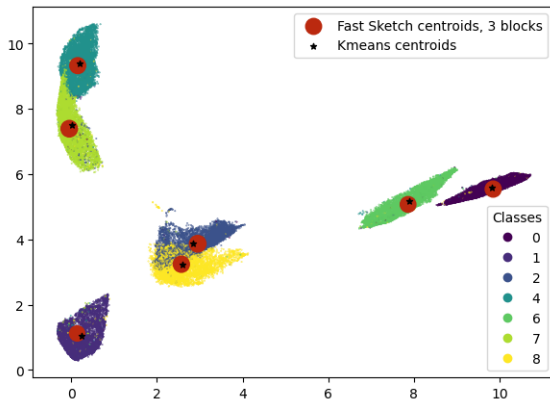
Sketch de taille $m = 10 \times K \times d$. Rappel : $\Phi(x) = \exp(-j2\pi Wx)$.
 $W = (w_j)_{1 \leq j \leq m} \in \mathbb{R}^{m \times d}$. Ici $w_j \sim |\mathcal{N}(0, \frac{1}{\sigma^2})| \times \mathcal{U}(\mathbb{S}^{d-1})$.
 σ^2 est estimé sur le jeu de données.



MNIST, centroïdes avec le sketch (SSE sketch / SSE K-means = 1.14)

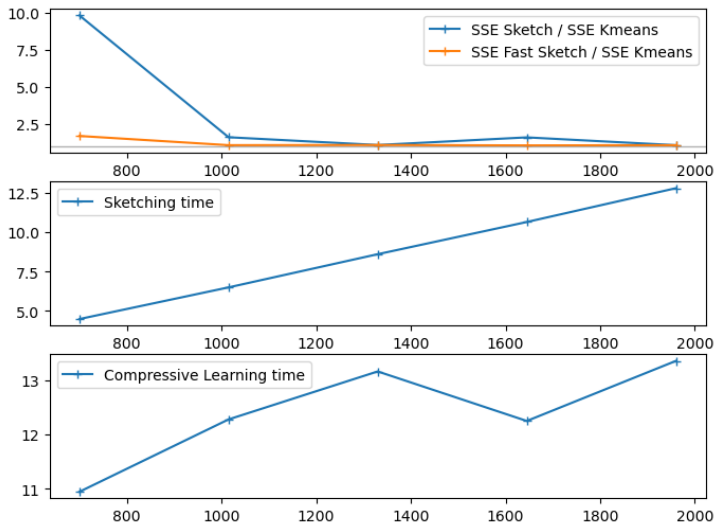
Clustering avec le "fast sketch"

"Fast sketch" de taille $m = 10 \times K \times d$, en utilisant 3 blocs de matrices pour B_j ($B_j = R B_{j,1} B_{j,2} B_{j,3}$).



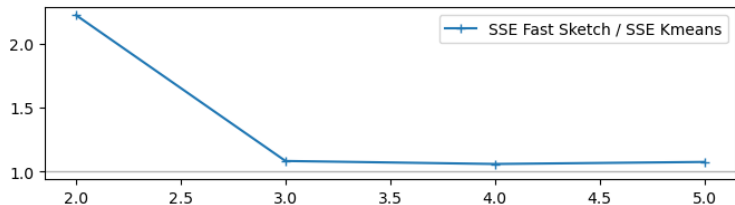
MNIST, centroïdes avec le "fast sketch" (SSE fast sketch / SSE K-means = 1.08)

Performances selon la taille du sketch



Performance du compressive learning selon la taille du sketch (moyenne sur 5 essais)

Performances selon le nombre de blocs ("fast sketch")



Performance du compressive learning selon le nombre de blocs L pour le fast sketch (moyenne sur 5 essais)

Conclusion

Principe : faire un résumé des données, et apprendre dessus.

Intérêt : réduire les coûts computationnels et de mémoire, gérer les situations de data flow, garantir la privacy.

Excès de risque : garanties données par la LRIP, pour un choix de risque, d'opérateur de sketching et d'un espace de distributions.

Inconvénients : choix d'hyperparamètres.

Limites : pour l'instant adapté à peu de tâches de machine learning.

Perspectives : garanties théoriques pour le fast sketching.

Références