Master research Internship

Bibliographic report

Towards Scalable Sketched Learning

Domains: Machine Learning

Author:

Antoine Chatalic

ENS Rennes

Supervisors:

Rémi Gribonval and Nicolas Keriven

PANAMA, Inria Rennes

Abstract: Designing efficient algorithms for learning from large-scale collections of high-dimensional elements is highly challenging. This report focuses on sketched learning, a framework which consists in compressing the whole collection into a sketch, and then performing the learning task from this sketch only. We derive from compressive sensing and random Fourier features the process of sketching, and explain which algorithms can be used to learn from the sketch for the problems of k-means clustering and mixture model estimation. Last part focuses on the computational cost of the method and discusses how structured matrices could be used to speed up the sketching process, which will be the core of the internship.

Contents

1 Introduction

The last decades have witnessed tremendous advances in the domain of machine learning, which aims at building algorithms able to extract, summarize, model and generalize the information contained in a dataset. Although the ever-increasing amount of available data opens up new opportunities, building algorithms able to cope with such large collection of often high-dimensional elements is actually highly challenging.

One approach to deal with this problem is to reduce the dimension of all elements individually, but the total number of elements would in this case stay unchanged. Other proposals include subsampling the dataset before learning, but simply discarding some elements increases the risk of loosing information.

A new approach has been proposed in the last few years under the name of sketched learning [1, 2, 3, 4, 5, 6]. It consists in first computing a compressed representation, called a sketch, of the whole dataset, and then performing the learning task from this sketch only, i.e. not relying anymore on the initial data. This global workflow is depicted in figure 1, where N denotes the number of elements in the collection, n the ambient dimension and m the size of the sketch.

This new framework has emerged from the domain of compressive sensing [7], which is built on the idea that most finite-dimensional signals can be reconstructed from far less linear measurements than their dimension because they belong to a low-dimensional subspace of the ambient space. This notion of low-dimensional model appears in machine learning as well, where it is common to make the assumption that the elements constituting the dataset have been drawn from an underlying probability distribution which has a low complexity; one can therefore generalize the information contained in the empirical distribution and avoid overfitting. Similarly to compressive sensing, the idea at the core of sketched learning is that a distribution belonging to such a low-dimensional model can be retrieved from a sketch of small size (compared to the initial dataset).

The sketch is obtained by averaging a carefully designed random function over the dataset. This function is nonlinear in the training samples, but the associated sketching operator is linear with respect to probability distributions. Once the sketch is calculated, the learning task can be formulated as finding, in a parametric family, the solution of a least squares problem. This framework has been applied with success for problems such as (Gaussian) mixture model estimation [3, 4], k-means clustering [5] or principal component analysis [6]. Some theoretical guarantees on the possibility of reconstructing from the sketch have also been proposed.

Ideally, the size of the sketch is in practice not related to the size N of the collection or the

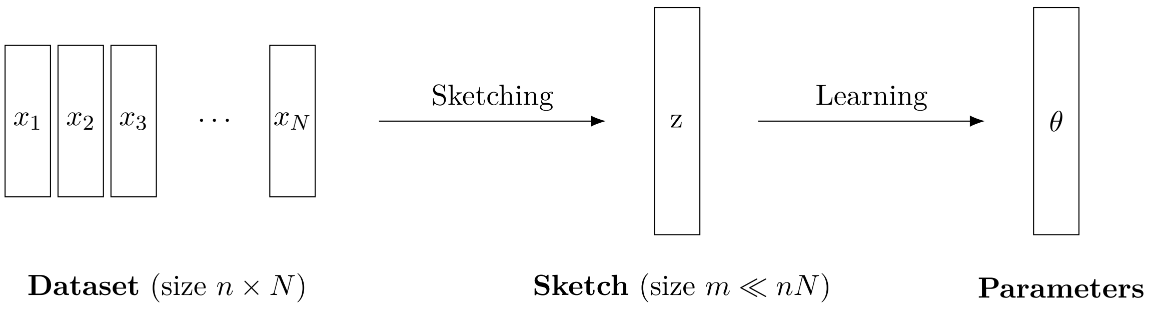

Figure 1: General principle of sketched learning. Here, θ denotes all the parameters that one wants to learn from the data.

dimension n of the training vectors, but rather to the number of parameters to learn, i.e. to the complexity of the learning task to be performed. This has been verified empirically by Keriven et al. for Gaussian mixture estimation [4] and k-means [5] using synthetic datasets. The sketch is therefore much smaller than the initial dataset, and that implies a cost for the process of learning from the sketch which is also quite reduced. However, the cost of computing the sketch is roughly $\mathcal{O}(mnN)$ and is mainly due to the multiplication by a random matrix. The goal of the internship is to reduce this complexity further by using fast transforms based on structured matrices.

Section 2 will explain how ideas from compressive sensing can be generalized to design a sketching operator for probability distributions, which bears similarities with kernel methods and random Fourier features. The produced sketch can then be used to perform the learning task (section 3), for instance estimation of mixture models or k-means clustering. The last part of this report (section 4) discusses complexity-related aspects and presents some works on fast operators relying on structured matrices that could be leveraged in order to reduce the cost of the sketching process.

2 Sketching Large Collections

2.1 From Compressive Sensing to Sketched Learning

A fundamental and early result in signal processing is the sampling theorem discovered independently by Kotelnikov, Nyquist, Shannon and Whittaker, which states that one should use a sampling rate of at least twice the highest frequency of a continuous band-limited signal to get all the information contained in it. In a related way, if $x \in \mathbb{R}^n$ is a discrete finite-dimensional signal and $y \in \mathbb{R}^m$ a linear observation obtained trough the measurement matrix $A \in \mathbb{R}^{m\times n}$ (y=Ax), basic linear algebra imposes that m should be at least n to be able to inverse the system, i.e. recover the original signal from the measurements. Compressive sensing (or compressed sensing) [7] is based on the discovery that it is actually possible to use far less linear measurements than that if the original signal is sparse, i.e. if there is a basis in which it can be represented with only a small number of nonzero coefficients.

Compressive sensing mainly consists in building linear measurement operators that indeed allow us to inverse such undetermined linear systems with the additional constraint of sparsity on the signal x, and designing algorithms to perform in practice this reconstruction. In a related way, one may be interested in approximating a signal by a sparse linear combination of elements of a dictionary; the latter can typically be overcomplete (i.e. a set of linearly dependent vectors) and possibly learned from data.

Interestingly, it has been proved that measurement operators having the desired properties could be produced with high probability using random (e.g. Gaussian or Bernouilli) matrices [7, chap. 9]. There is then a guarantee on the number of needed measurements, namely s-sparse vectors (i.e. having no more than s nonzero entries) can be reconstructed from m measurements if $m \geq Cs\ln(n/s)$, where C is a constant.

Building on the idea that sparsity is just a low-dimensional model for vectors, many proposals have been made to extend the key idea of compressive sensing to other kind of spaces and associated low-dimensional models. For instance, instead of retrieving a sparse vector, one might be interested in recovering a low-rank matrix. In machine learning, it is common to make the assumption that the dataset has been drawn from a distribution with a relatively low complexity, i.e. which belongs to a low-dimensional model. A typical use is to consider families of parametric distributions, which are defined by a small number of parameters. Sketched learning has emerged following this idea; the goal here is to build a sketching operator $\mathcal{A}$, which is linear with respect to probability distributions, such that the sketch $\hat{z} = \mathcal{A}(\hat{p})$ of the empirical distribution $\hat{p} = \frac{1}{N}\sum \delta_{x_i}$ of the dataset $X = [x_1, \dots, x_N]$ contains all the relevant information on the distribution that is needed to perform the desired learning task.

2.2 Calculation of the Sketch

We propose here to explain how the sketch is calculated in practice.

Let $\mathcal{P}$ be the set of probability distributions over some measurable space $(Y, \mathcal{B})$. The general idea to build a sketching operator $\mathcal{A} : \mathcal{P} \to \mathbb{C}^m$ consists in calculating some carefully designed generalized moments of the probability distribution. For a sketch of size m, one would calculate m such moments:

$$\mathcal{A} : p \mapsto (E_{x\sim p} f_1(x), \quad \cdots, E_{x\sim p} f_m(x))$$

where (f_i) are functions over Y taking values in $\mathbb{C}$. The principle behind reconstruction (see section 3) is that the sketch of a distribution belonging to the low-dimensional model that correctly fits the dataset should be close to the sketch of the empirical distribution $\hat{p} = \frac{1}{N}\sum_{i=1}^{N} \delta_{x_i}$. This idea is reminiscent of the generalized method of moments, in which the parameters of a (usually semiparametric) model are estimated by fitting some theoretical generalized moments of the parametric model to their empirical equivalents.

Although different functions might be used for the (f_i), we focus here on the sketching operator that has been proposed for compressive k-means [5] and Gaussian mixture estimation [3]:

$$\mathcal{A} : p \mapsto \left(\mathbb{E}_{x\sim p} e^{-i\omega_1^T x}, \quad \cdots, \quad \mathbb{E}_{x\sim p} e^{-i\omega_m^T x}\right)$$

where $(\omega_j)_{1\leq j\leq m}$ are some frequency vectors in $\mathbb{R}^N$ drawn i.i.d. from a well-chosen distribution (this choice will be explained later, see 2.3). In practice, the true data distribution is unknown, so the sketch is calculated from the empirical distribution $\hat{p} = \frac{1}{N}\sum_{i=1}^{N} \delta_{x_i}$:

$$\mathcal{A}(\hat{p}) = \left(\frac{1}{N}\sum_{i=1}^{N} e^{-i\omega_1^T x_i}, \quad \cdots, \quad \frac{1}{N}\sum_{i=1}^{N} e^{-i\omega_m^T x_i}\right)$$

We recall that to any probability distribution p can be associated its characteristic function $\varphi_p : \omega \mapsto E_{x\sim p}[e^{-i\omega^T x}]$ which uniquely defines it; it is actually the Fourier transform of the probability density function. The sketching operator therefore simply consists in sampling the characteristic function of p at the randomly chosen frequencies $(\omega_j)_{1\leq j\leq m}$. This choice can be explained by some classical results from compressive sensing, namely it is possible to reconstruct a sparse signal with high probability from a low number of randomly selected samples of its discrete Fourier transform.

In practice, if W denotes the matrix of frequencies ($W = [\omega_1, \dots, \omega_m]^T$) and X the dataset ($X = [x_1, \dots, x_N]$), the calculation consists in computing the product WX, applying an element-wise complex exponential non-linearity, and averaging. The computation of WX can therefore easily be distributed. Moreover, it is compatible with on-line learning, i.e. it is possible to account for new elements of the collection without recomputing the sketch: we can simply update it after sketching these new elements only. These properties simply come from the linearity of the sketching operator, the empirical distribution of a union of datasets being a linear combination of their respective empirical distributions.

It has been empirically suggested that m should be roughly close to the number of parameters that one wants to learn during the learning phase, i.e. in a way directly related to the complexity of the learning task. In practice, m will therefore be usually much smaller than N.

Random Neural Networks As explained before, the computation of the sketch can be seen as the succession of two different steps: first, computing the matrix product WX, and then applying an element-wise non-linear complex exponential before averaging. This calculation is similar to what would be computed in one layer of a neural network with random weights: calculating a linear output before applying a non-linear function. In a neural network however, the training examples would be used one at a time for learning, whereas we apply this calculation to the whole dataset and average to get a mean representation.

2.3 Distribution of the Frequencies

We explained that a sketch of size m was obtained by sampling the characteristic function of the empirical distribution in m frequencies drawn i.i.d. from a distribution (which will be denoted Λ in the following), that we did not specify. Because we are interested in speeding up the sketch calculation (see 4.1), it is important to understand this underlying distribution: we would like to approximate somehow the matrix W by a faster operator, but we want it to behave similarly. We give briefly the intuition of how such Λ distributions have been built for the specific case of Gaussian mixture estimation [4]. In practice, the obtained "adapted radius distribution" has been empirically shown to give good results for compressive k-means as well [5].

Gaussian Distribution In the context of Gaussian mixture estimation, it is natural to take a look at the expression of the characteristic function of the normal distribution to choose the frequencies. A normal distribution of parameters $\theta = (\mu, \Sigma)$ has the following characteristic function:

$$\varphi_\theta(\omega) = \exp(-i\omega^T\mu)\exp\left(-\frac{1}{2}\omega^T\Sigma\omega\right), \quad \text{and in particular: } |\varphi_\theta(\omega)| = \exp\left(-\frac{1}{2}\omega^T\Sigma\omega\right) \tag{1}$$

A first choice is therefore to draw the frequencies with respect to the modulus of this characteristic function, i.e. $\Lambda_\Sigma = \mathcal{N}(0, \Sigma^{-1})$. However, it is known and has been empirically verified that such a distribution undersamples low frequencies in high dimension.

Folded Gaussian Distribution The first solution that has been proposed to avoid this problem is to draw ω as follows:

$$\omega = R\Sigma^{-1/2}\delta$$

Here, $\delta \in \mathbb{R}^n$ is a direction chosen with a uniform distribution on the unit sphere, and $R \in \mathbb{R}^+$ is the radius, whose distribution should still allow us to control the amplitude $e^{-\frac{1}{2}\omega^T\Sigma\omega}$ of the characteristic function. It can be shown that the previously given characteristic function (1) corresponds, with this new distribution for ω, to the one-dimensional characteristic function of a Gaussian of parameters $\theta_R = (\mu_R = \delta^T\Sigma^{-1/2}\mu, \sigma_R = 1)$ evaluated in R, i.e. $\varphi(R\Sigma^{-1/2}\delta) = \varphi_{\theta_R}(R)$.

The previously introduced Gaussian distribution associated to this one-dimensional characteristic function is therefore chosen for R. In practice, R being positive, a folded normal distribution of parameters $(\mu = 0, \sigma = 1)$ is used according to (1).

Adapted Radius Distribution A third distribution has been proposed to reduce the number of frequencies with a low radius produced by the folded Gaussian distribution. The characteristic function indeed takes values close to 1 around $\omega = 0$, and therefore such frequencies do not carry much information.

By adding the condition that the frequencies should help to discriminate Gaussians with different parameters around the true Gaussian, the following distribution is derived for R [4]:

$$p_R(R) = C\left(R^2 + \frac{R^4}{4}\right)^{\frac{1}{2}} e^{-\frac{1}{2}R^2} \quad \text{where } C \text{ is a constant}$$

The direction is chosen similarly to the folded Gaussian distribution.

2.4 From Sketches to Embeddings of Probability Distributions

Although sketching has been introduced as an extension of compressive sensing, it is closely related to some other concepts such as embeddings of probability distributions in reproducing kernel Hilbert spaces [8] and random Fourier features [9]. We propose here to highlight these similarities.

Mean Map Embedding When working with a kernel (i.e. symmetric positive definite function) $k : Y \times Y \to \mathbb{C}$, there exists from the positive definiteness of the kernel a mapping ψ to a high-dimensional feature space $\mathcal{F}$ endowed with an inner product $\langle \cdot, \cdot \rangle_{\mathcal{F}}$, i.e. $\psi : Y \to \mathcal{F}$, such that $\forall x, y \in Y, k(x, y) = \langle \psi(x), \psi(y) \rangle_{\mathcal{F}}$. The idea of working in the feature space $\mathcal{F}$ without actually needing to compute ψ by relying on the kernel function to perform the calculations is known as the kernel trick, and is at the core of kernel methods.

In a similar way, according to Moore-Aronszajn theorem, there is for every kernel k a unique Hilbert space of functions, that we denote in the following $(\mathcal{H}, \langle \cdot, \cdot \rangle_{\mathcal{H}})$, such that $\forall x \in Y, \forall f \in \mathcal{H}, f(x) = \langle k(x, \cdot), f \rangle_{\mathcal{H}}$. Mean map embedding consists in representing a probability distribution as a mean function in this space through the following feature map:

$$\phi : p \mapsto \int_Y k(x, \cdot) dp(x)$$

A pseudometric γ_k is then naturally defined from the inner product as $\gamma_k(p, q) = \|\phi(p) - \phi(q)\|_{\mathcal{H}}$. Sriperumbudur et al. [8] gave conditions on k under which γ_k is a true metric (such k are called characteristic kernels); this ensures that the associated embedding ϕ is injective.

Connection with sketches Suppose now that $Y = \mathbb{R}^N$. A kernel k is said to be translation-invariant if it can be expressed as $k(x, y) = K(x - y)$, where $K : \mathbb{R}^n \to \mathbb{C}$. In this specific case, Bochner's theorem states that K is the Fourier transform of a non-negative measure, which under some normalization constraints is a proper probability distribution $\Lambda(\omega)$. As a consequence:

$$k(x, y) = K(x - y) = \int_{\mathbb{R}^n} \Lambda(\omega) e^{i\omega^T(x-y)} d\omega \tag{2}$$

In this case, it can be proved (see e.g. [8, corollary 4 (i)]) that γ_k can be expressed as follows:

$$\gamma_k(p, q) = \left(\int_{\mathbb{R}^n} |\varphi_p(\omega) - \varphi_q(\omega)|^2 \, d\Lambda(\omega)\right)^{\frac{1}{2}}$$

where φ_p and φ_q denote the characteristic functions of p and q. Therefore, if the sketching operator $\mathcal{A}$ relies on m frequencies $(\omega_i)_{1 \leq i \leq m}$ drawn i.i.d. from a distribution Λ, and k denotes the kernel associated to Λ through Bochner's theorem, then with a high probability:

$$\|\mathcal{A}(p) - \mathcal{A}(q)\|_2^2 = \frac{1}{m} \sum |\varphi_p(\omega) - \varphi_q(\omega)|^2 \approx \int_{\mathbb{R}^n} |\varphi_p(\omega) - \varphi_q(\omega)|^2 \, d\Lambda(\omega) = \gamma_k(p, q)^2$$

The l_2 distance between sketches is therefore directly related to the (pseudo)metric γ_k.

Random Fourier Features Kernels appear in many ways in machine learning, and various works have been done to speed up the evaluation of these kernel functions. A common way to do that is to build an approximation of the feature map, i.e. a function $z : \mathbb{R}^n \to \mathbb{R}^D$ such that $\forall x, \forall y, k(x, y) \approx z(x)^T z(y)$.

A popular such method is random Fourier features [9], which can be applied for translation-invariant kernels. The main difference with sketched learning, where one defines implicitly a kernel by choosing a distribution Λ for the frequencies, is that the process is reversed; random Fourier features consist, for a given translation-invariant kernel k (s.t. $k(x, y) = K(x - y)$) that one wishes to approximate efficiently, in building a feature map relying on frequencies drawn i.i.d. from the Fourier transform Λ_k of K according to Bochner's theorem (2). By denoting $(\omega_i)_{1 \le i \le D}$ these frequencies, the feature map is in practice defined as follows:

$$z(x) = \sqrt{1/D}\left[\cos(\omega_1^T x), \sin(\omega_1^T x), \cos(\omega_2^T x), \sin(\omega_2^T x), \cdots, \cos(\omega_D^T x), \sin(\omega_D^T x)\right]^T$$

3 Learning from the Sketch

The first algorithms that have been proposed for sketched learning are the estimation of the parameters of a mixture model, and k-means clustering. We explain in this section how to perform such tasks from the previously calculated sketch, by first having a look at reconstruction methods used in traditional compressive sensing. Note that some results have also been obtained for principal component analysis [6].

3.1 From Orthogonal Matching Pursuit to CLOMPR

Let $(Y, \mathcal{B})$ still denote a measurable space, and $\mathcal{P}$ be the set of probability distributions over Y. Given an empirical sketch $\hat{z} = \mathcal{A}(\hat{p})$, the reconstruction is performed by looking for the probability distribution q^* in the set of distributions allowed by our low-dimensional model $\mathcal{M} \subset \mathcal{P}$ whose sketch best matches the empirical sketch:

$$q^* = \underset{q \in \mathcal{M}}{\arg\min} \, \|\hat{z} - \mathcal{A}(q)\|_2 \tag{3}$$

Finding $q*$ is a usually a difficult non-convex optimization problem.

This reconstruction is again reminiscent of compressive sensing, where one seeks to reconstruct a signal x using an observation $y = Ax$, by solving $v = \arg\min_v \|y - Av\|_2$ with a sparsity assumption for v. The methods that have been developed to solve this problem can be roughly categorized in three classes: optimization-based methods that usually work on a relaxed version of the sparsity constraint, thresholding-based techniques, and matching pursuit and its variations. An adaptation of hard-thresholding algorithm has initially been proposed for sketched learning [1, 2][10, chap. 8]. In the following, algorithms based on matching pursuit are considered; we briefly explain the main idea of this method, and how it has been adapted to the case of sketched learning.

Matching Pursuit We consider from now that the reconstructed vector must be s-sparse, i.e. cannot have more than s nonzero entries. The idea, starting from $v_0 = 0$, is to iteratively and greedily widen the support (i.e. the set of the positions of the non-zero coefficients) of the vector v_k being reconstructed until reaching v_s which can have s nonzero coefficients, the maximum allowed by the s-sparsity constraint. If we denote v_k this vector at iteration k (which therefore has k nonzero entries), v_{k+1} is built to differ from v_k only by its i-th coefficient which can now have a value different from zero; the value of i is selected such that by taking a nonzero value, the i-th coefficient of v_k is the one that can reduce the most the residual, i.e. the one corresponding to the column A_i of A which has the highest correlation $|\langle r_k, A_i \rangle|$ with the residual $r_k = Av_k - y$.

Extensions A popular variant is orthogonal matching pursuit (OMP), where all coefficients are updated each time we extend the support, by performing an orthogonal projection on the space spanned by the selected columns of A. Orthogonal matching pursuit with replacement (OMPR) algorithm is obtained by combining this idea with the possibility of modifying the support even after reaching a vector with s nonzero entries: the algorithm performs $2s$ iterations instead of s, and iterations $s+1$ to $2s$, which would technically produce a reconstruction v_k with $k > s$ nonzero coefficients, are followed by a hard-thresholding step keeping only the s highest coefficients. The support can therefore be modified during these iterations, but its size cannot exceed s.

Reconstructing Distributions The algorithm that has been designed for sketched learning, named CLOMPR, is an adaptation of OMPR where the letters "CL" stand for compressive learning. The main difference is probably that the finite dictionary that was composed of the columns of A is now replaced with an infinite continuously-indexed dictionary (the parametric model). We propose to study the new difficulties arising in this context through the practical cases of mixture model estimation and k-means clustering.

3.2 Mixture Model Estimation

The framework of sketched learning has been initially used to estimate the parameters of mixture model [3, 4][10, chap. 8]. In this context, we consider a parametric model for "basic" distributions: $\mathcal{M}_b = \{p_\theta\}_\theta$, and use the notion of s-sparsity for probability distributions that are a mixture of s such basic distributions:

$$\mathcal{M} = \left\{ \sum_{i=1}^{s} \alpha_i p_{\theta_i} \;\middle|\; \sum_{i=1}^{s} \alpha_i = 1 \text{ and } (\forall i \in [\![1, s]\!],\ \alpha_i \geq 0 \text{ and } p_{\theta_i} \in \mathcal{M}_b) \right\}$$

Similarly to OMP (let consider for now that replacement is not used), we want to iteratively enlarge the support, which is now composed of distributions in $\mathcal{M}_b$ (or equivalently of their respective parameters), i.e. we want at each iteration to pick the value of θ corresponding to the distribution p_θ which has the highest correlation with the residual. A first difference with the classic version of OMP, where the nonzero coefficients of the reconstructed vector could take any value, is that the α_k need to be positive in order to yield a mixture which is a correct probability distribution. We therefore don't maximize the absolute value of the correlation, but its real part. If we denote $\theta^k = (\theta_i^k)_{1 \leq i \leq k}$ the k values of the parameters chosen until step k, then at step $k+1$ the support is update as follows:

$$\theta^{k+1} = \left(\theta^k,\ \arg\max_\theta \left[\mathrm{Re} \left\langle \frac{\mathcal{A}(p)_\theta}{\|\mathcal{A}(p_\theta)\|_2}, \hat{r}_k \right\rangle \right] \right)$$

where $\hat{r}_k$ denotes the residual after iteration k. As explained before, the dictionary $\mathcal{M}_b$ is now infinite and continuously indexed; a gradient ascent is used in order to find a maximum.

Still at step $k+1$ and after expanding the support, the coefficients $\alpha^{k+1} = (\alpha_i^{k+1})_{1 \leq i \leq k+1}$ associated to the $(\theta_i^{k+1})_{1 \leq i \leq k+1}$ can be calculated as the solution of the following non-negative least-squares optimization problem:

$$\alpha^{k+1} = \arg\min_{\alpha \geq 0} \left\| \hat{z} - \sum_{i=1}^{k+1} \alpha_i \mathcal{A}(p_{\theta_i^{k+1}}) \right\|_2$$

Eventually, a global minimization step is added at the end of each iteration compared to standard OMP algorithm. The minimization is performed jointly on α and on the support θ (for a fixed size $k+1$ at iteration $k+1$), starting from their current values $(\alpha^{k+1}, \theta^{k+1})$:

$$(\alpha^{k+1}, \theta^{k+1}) = \arg\min_{\alpha, \theta} \left\| \hat{z} - \sum_{i=1}^{k+1} \alpha_i \mathcal{A}(p_{\theta_i}) \right\|_2 \quad \text{starting from the current value of} \quad (\alpha^{k+1}, \theta^{k+1})$$

This new optimization step is due to the fact that the dictionary is not incoherent enough, i.e. different elements of the dictionary can be quite close (for close values of the parameters) and it might be necessary to adjust the support when a new element is added to it.

The obtained algorithm, CLOMP, needs s iterations for an s-sparse mixture. A version with replacement (CLOMPR) can easily be derived by performing $2s$ steps, and adding on the last s iterations a hard-thresholding step (after expanding the support) keeping only in the support the s parameters associated to the s highest weights β_i calculated as follows:

$$\beta = \arg\min_{\beta \geq 0} \left\| \hat{z} - \sum_{i=1}^{|\theta^k|} \beta_i \frac{\mathcal{A}(p_{\theta i})}{\|\mathcal{A}(p_{\theta i})\|} \right\|_2$$

Experiments are performed on synthetic data and on a database for speaker verification. The adapted radius frequency distribution (see 2.3) is shown to outperform Gaussian and folded Gaussian radius distribution. The CLOMPR algorithm gives similar results to those obtained using EM, but using significantly less time and memory for large ($N > 10^5$) datasets. Experiments for speaker verification are performed at scales that are not accessible for EM algorithm on a single laptop.

3.3 Compressive K-means

Starting from the dataset $X = \{x_1, \cdots, x_N\}$ of points in $\mathbb{R}^n$, the problem commonly known as k-means clustering consists in finding k centers $C = \{c_1, \cdots, c_k\}$ in $\mathbb{R}^n$ that minimize the sum of squared distances from each point x_i to its closest center c_j:

$$C = \arg\min_{C} \sum_{i=1}^{N} \min_{j} \|x_i - c_j\|_2^2 \tag{4}$$

The problem is NP-hard, and the simple heuristic proposed by Lloyd, which consists in iteratively associating each point to its closest cluster, and then updating the centers' positions, is often used to converge to a local optimum.

This problem has been studied from the viewpoint of sketched learning under the name of compressive k-means by Keriven et al. [5]. By denoting $p_C = \sum_{j=1}^{k} \delta_{C_j}$, one aims at finding C such that $\|\hat{z} - \mathcal{A}(p_C)\|_2^2$ is minimal according to (3). The same CLOMPR algorithm can be used to perform this reconstruction in practice. The authors also propose to associate weights to the points when sketching.

Empirical results are given for artificial data and for spectral clustering on the MNIST dataset. Multiple initialization strategies are discussed, and compressive k-means is shown to be less sensitive to initialization. Relative sum of squared errors (SSE), i.e. SSE obtained with compressive k-means divided by SSE obtained with Lloyd algorithm, are given for different values of m, n and k. A relative SSE of 2 is shown to be obtained roughly for $m \approx 5kn$, which is coherent with what one could expect. The algorithms are also executed 5 times in a row with different random initializations; this gives better SSE for k-means, but the results for compressive k-means are stable across repetitions, suggesting that only one execution of the algorithm is sufficient. In all cases (even when SSE is larger), compressive k-means yields better clustering results in terms of adjusted rank index. As regards execution time, CLOMPR is roughly 2 orders of magnitude faster than standard k-means for a dataset of $N = 10^7$ elements.

4 Scalable Sketched Learning

Although the proposed framework of sketched learning allows to perform some learning tasks with a drastically reduced computational complexity, there is still room for improvement, in particular for the calculation of the sketch. We propose to take a closer look at the complexity of the different parts of the workflow, and then review some ideas related to fast transforms that could be used to further reduce the sketching complexity.

4.1 Computational Complexity of the Framework

Lloyd algorithm for k-means scales in $\mathcal{O}(knNI)$ (where k denotes the number of clusters, I the number of iterations, and n and N still respectively the ambient dimension and the number of points), whereas the CLOMPR algorithm has complexity $\mathcal{O}(mnk^2)$ [5]. It has been empirically shown that m only needs to be roughly close to the number of parameters to estimate, i.e. in our case the positions of k clusters: $m \approx kn$, and CLOMPR alone can therefore be much faster than standard k-means. However, one also needs to account for the cost of sketching, which is $\mathcal{O}(mnN)$ due to the computation of the matrix product WX before applying the non-linearity and averaging. This multiplication has clearly the highest cost of the whole process, and this is also the case for other problems that have been subject to a “sketched” version. Even if it is easy to split the dataset over several nodes and in this way distribute the computation across them, it would be desirable to further reduce this cost. Although it will still be needed to go through the whole collection one time, it should be possible to use fast transforms to reduce the cost of processing one training vector.

Keriven et al. suggest as well to reduce the dimension of the vectors as a preprocessing step. It is indeed possible in theory to perform such a dimension reduction and preserve approximately the distances according to Johnson-Lindenstrauss lemma. Some works have been done in this direction, relying for instance on random projections [11].

4.2 Efficient Sketching Using Fast Transforms

FAμST transforms Building on the idea that most well-known fast transforms such as the fast Fourier transform or fast implementations of discrete cosine transform and discrete wavelet transform all share a common structure, Le Magoarou and Gribonval proposed [12] under the name of FAμST (Fast Approximate Multi-layer Sparse Transforms) a general framework that encompasses all such operators that either are fast or try, in a similar way, to approximate a desired matrix with a lower computation complexity. A FAμST matrix can be written as a product (hence “multi-layer”) of few sparse matrices:

$$A = \prod_{j=1}^{J} S_j$$

where the $(S_j)_{1\leq j\leq J}$ are individually sparse, and ideally the total number s of nonzero coefficients in all the factors $(S_j)_{1\leq j\leq J}$ is much lower than the size of the dense matrix A. The storage cost and the computational cost of the multiplication, which both scale in $\mathcal{O}(s)$, are therefore likewise reduced. Moreover, when an operator has to be estimated from the data, enforcing a FAμST structure yields better statistical properties; this is for instance quite interesting in the context of dictionary learning, where one tries to learn a dictionary of atoms in which elements have a representation that is e.g. as sparse as possible.

Factorization algorithm A hierarchical factorization algorithm is introduced to get a FAμST approximation of a given operator A, by minimizing a cost function as follows:

$$(S_i)_{1\leq j\leq J} = \underset{(S_j)_{1\leq j\leq J}}{\arg\min} \left\| A - \prod_{j=1}^{J} S_j \right\|^2 + \sum_{j=1}^{J} f(S_j)$$

where f is a penalty function favoring sparse factors. Some non-convex optimization methods are used, yielding an efficient algorithm with is applied with successful results to dictionary learning and acceleration of linear inverse problems.

Substituting W by a FAμST equivalent Due to the nature of our dense and random matrix of frequencies $W = (\omega_1, \ldots, \omega_m)^T$, there is no particular reason that such a factorization approach would work correctly for this specific case. However, the general framework is interesting and we indeed seek to build a substitution operator for W which would somehow have a FAμST structure. Next subsection takes a closer look at different works that use sparse matrices for building approximations of operators of interest.

4.3 Building Fast Transforms using Structured Matrices

Multiple works have proposed to approximate some transformations of interest, such as random Gaussian matrices, by relying on structured matrices, i.e. matrices having far less degrees of freedom than their size, and whose structure can be leveraged for fast multiplication. Examples include Toeplitz (diagonal-constant), circulant (special case of Toeplitz), Hankel (i.e. having constant skew-diagonals) and Vandermonde matrices. This notion is naturally closely related to the FAμST structure. Most of the articles presented here focus on kernels approximation; similarly to random Fourier features (see 2.4), the goal is to design approximate features maps of the form:

$$z(x) = \sqrt{\frac{1}{m}} \times \rho(Ax)$$

where A is a $m \times n$ matrix and ρ denotes the nonlinearity. A common case is the approximation of Gaussian kernels, i.e. having for A a random matrix whith every coefficient sampled independently from the standard normal distribution; as a consequence, most of the following works focus on the approximation of Gaussian matrices. In the following, we mean by "Gaussian" entries values that are drawn from the standard normal distribution $\mathcal{N}(0, 1)$.

Fastfood Le et al. showed that the behavior of a dense Gaussian random matrix could be well approximated by relying on Walsh-Hadamard and diagonal matrices having Gaussian values on theirs diagonals [13]. We recall that Walsh-Hadamard matrices have mutually orthogonal rows (and columns), allow to perform matrix-vector products in linearithmic time, and can be obtained recursively by the following definition:

$$H_1 = [1] \quad \text{and} \quad H_{2d} = \begin{bmatrix} H_d & H_d \\ H_d & -H_d \end{bmatrix} = H_2 \otimes H_d$$

where $\otimes$ denotes the kronecker product. The matrix A is obtained by stacking vertically multiple matrices F_i having the following shape:

$$F = \frac{1}{\sigma\sqrt{d}} SHG\Pi HB \tag{5}$$

where S, G, B are diagonal random matrices, $\Pi \in \{0,1\}^{n\times n}$ is a (e.g. randomly generated) permutation, and H a Walsh-Hadamard matrix. If the desired feature size is not a power of two, the authors propose to simply use a bigger Walsh-Hadamanrd matrix (with a dimension $n = 2^k$) and to use zero-padding. More specifically, B has binary (i.e. in $\{0,1\}$) random values on its diagonal, G has Gaussian random values, and S is a random scaling matrix. The authors show that any row of $HG\Pi HB$ is i.i.d. Gaussian. The key idea with this approach is that a single diagonal Gaussian matrix is "recycled" to produce many Gaussian (but not independent) rows.

For a final A matrix of size $m \times n$, storage cost grows in $\mathcal{O}(m)$, and the multiplication by A can be performed in $\mathcal{O}(m\log(n))$. Both theoretical approximation guarantees and empirical simulations for penalized least-square regression are provided.

$\mathcal{P}$-model Still in the context of kernels approximation, Choromansky and Sindhwani proposed a general framework for creating random structured matrices by separating randomness from structure [14]. This is similar to the idea, in the Fastfood approach, of using only a diagonal of Gaussian elements to produce many Gaussian rows, but the separation is here even more explicit.

From a random Gaussian vector g of size r and a $\mathcal{P}$-model, i.e. a set of structured matrices $\mathcal{P} = (P_i)_{1\leq i\leq m}$ s.t. $P_i \in \mathbb{R}^{r\times n}$, the random matrix of size $m \times n$ is constructed as follows:

$$A(\mathcal{P}, g) = \begin{pmatrix} g^T P_1 \\ \vdots \\ g^T P_m \end{pmatrix}$$

This framework encompasses classical structured matrices such as Hankel and Toeplitz matrices as specific cases, as well as Fastfood [13] (using for the P_i some diagonal matrices).

Different metrics on $\mathcal{P}$ are introduced to quantify the structure of a given $\mathcal{P}$-model and its ability to "recycle" the Gaussian vector, i.e. metrics measuring the level of dependence between the rows of the final matrix $A(\mathcal{P}, g)$ produced by the model. It is in the end easier to control at the same time the quality of the model with these measurements and the "budget" of randomness r.

Multiple theoretical results are provided based on the introduced metrics, including the unbiasedness of the approximation for Gaussian and arc-cosine kernels of order 0 and 1. The good results obtained with the Fastfood approach can be explained by this formalism.

Orthogonal Random Features In the continuation of random Fourier features, Yu et al. replaced the Gaussian matrix by a random matrix having orthogonal rows [15], and observed significant improvement of the approximation accuracy. This is coherent with the good results obtained with Fastfood, where HG (see eq. (5)) has orthogonal rows with a high probability.

More specifically, let A denote the desired final $m \times n$ matrix. If $m > n$, the approach consists similarly to Fastfood in building A by vertically stacking $\lceil m/n \rceil$ independently generated squared $n \times n$ matrices (F_i). In order to generate each of these F_i matrices, one starts by generating a $n \times n$ random Gaussian matrix G and calculating its QR decomposition $G = QR$, where Q is orthogonal. F_i is then defined as:

$$F_i = \frac{1}{\sigma} SQ$$

where σ is the standard deviation of the desired Gaussian kernel to estimate and S is a diagonal matrix whose entries are drawn according to the χ-distribution with n degrees of freedom. This ensures that the norms of the rows of F_i have the same distribution than in the case of a standard random Gaussian matrix. The features obtained with this approach are called orthogonal random features (ORF). The feature map defined using such a matrix A is proved to yield unbiased estimation of the Gaussian kernel, and a bound on the variance is provided.

The authors extend this idea further by proposing structured orthogonal random features (SORF) for faster estimation. The F_i have in this case the following structure:

$$F_i = \frac{\sqrt{n}}{\sigma} HD_1HD_2HD_3$$

where H is the normalized Walsh-Hadamard matrix and D_i are diagonal matrices with entries following the Rademacher distribution. The idea is again here that a Gaussian behavior can be approached using Walsh-Hadamard and diagonal matrices with high accuracy. For large values of n (empirically $n > 32$), the approximation error is shown to be similar in both cases (ORF/SORF), which empirically validates one more time this fact.

5 Discussion and Outlook

Before discussing how the different elements that have been introduced fit together, it should be mentioned that multiple other approaches have been used to deal with large-scale volumes of data. As previously mentioned (see 4.1), some methods try to reduce the dimension of the vectors individually before learning. Such dimensionality reduction can be performed using random projections [11]. Another interesting approach would be to reduce the number N of samples before learning. This is the direction taken with coresets methods, which try to identify a smaller subset having good generalization properties for the desired task. Sketching has also been used in the database community for different goals, for instance to work with continuous streams [16], in which case it is convenient to be able to simply update the sketch.

Summary The framework of sketched learning as been presented, from the process of sketching to the reconstruction algorithm. The latter has been adapted from traditional compressive sensing and has a low complexity compared to algorithms working on the original dataset. On the sketching side however, randomly sampling the characteristic function of the empirical probability distribution involves a matrix product which has an important computational cost. The latter could be reduced by leveraging approaches initially proposed in the context of kernel approximation and relying on random structured matrices.

Multiple questions arise from the different elements that have been detailed. The FAμST factorization approach is attractive, but as suggested trying to factorize directly W is unlikely to work in practice. It will therefore be necessary to manually build an operator having a FAμST design from some well-chosen structured matrices. It is also important to keep in mind that the frequencies are drawn with respect to a specific distribution (e.g. adapted radius distribution, see subsection 2.3); the fast operator should ideally approximate this distribution. In the case of the adapted radius distribution, each frequency is generated by a real radius and a direction chosen uniformly on the unit sphere. Given that is is possible to draw a vector with uniform distribution on the unit sphere by drawing its coefficients i.i.d. from a normal distribution and normalizing the resulting vector, we can hope to directly benefit from results on Gaussian matrices. The radius could be taken into account simply via a diagonal scaling matrix, similarly to the matrix S in the Fastfood approach (see equation (5)). It could also be possible to consider slightly modified distributions of the frequencies.

The different approaches proposed during the internship will be subject to an empirical validation. Implementations will rely on the MATLAB toolbox SketchMLbox [17], which already provides the (standard) sketching procedure and CLOMPR learning algorithm. Large-scale datasets such as Imagenet [18] will be used in order to evaluate the performance of the proposed method for scalable sketched clustering.

References

[1] Anthony Bourrier. *Compressed sensing and dimensionality reduction for unsupervised learning. (Échantillonnage compressé et réduction de dimension pour l'apprentissage non supervisé).* PhD thesis, University of Rennes 1, France, 2014.

[2] Anthony Bourrier, Remi Gribonval, and Patrick Perez. Compressive gaussian mixture estimation. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*.

[3] Nicolas Keriven, Anthony Bourrier, Rémi Gribonval, and Patrick Pérez. Sketching for large-scale learning of mixture models. In *ICASSP*, pages 6190–6194. IEEE, 2016.

[4] Nicolas Keriven, Anthony Bourrier, Rémi Gribonval, and Patrick Pérez. Sketching for large-scale learning of mixture models. *CoRR*, abs/1606.02838, 2016.

[5] Nicolas Keriven, Nicolas Tremblay, Yann Traonmilin, and Rémi Gribonval. Compressive k-means. *arXiv preprint arXiv:1610.08738*, 2016.

[6] Rémi Gribonval, Gilles Blanchard, Nicolas Keriven, and Yann Traonmilin. Random moments for sketched statistical learning. In *Submitted to AISTATS 2017*, October 2016.

[7] Simon Foucart and Holger Rauhut. *A Mathematical Introduction to Compressive Sensing.* Applied and Numerical Harmonic Analysis. Birkhäuser, 2013.

[8] Bharath K. Sriperumbudur, Arthur Gretton, Kenji Fukumizu, Bernhard Schölkopf, and Gert R. G. Lanckriet. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11:1517–1561, 2010.

[9] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *NIPS*, pages 1177–1184. Curran Associates, Inc., 2007.

[10] Holger Boche, Robert Calderbank, Gitta Kutyniok, and Jan Vybíral. *Compressed sensing and its applications.* Springer, 2015.

[11] Christos Boutsidis, Anastasios Zouzias, and Petros Drineas. Random projections for k-means clustering. In *Advances in Neural Information Processing Systems*, pages 298–306, 2010.

[12] Luc Le Magoarou and Rémi Gribonval. Flexible multi-layer sparse approximations of matrices and applications. *CoRR*, abs/1506.07300, 2015.

[13] Quoc Le, Tamás Sarlós, and Alex Smola. Fastfood-approximating kernel expansions in loglinear time. In *Proceedings of the international conference on machine learning*, 2013.

[14] Krzysztof Choromanski and Vikas Sindhwani. Recycling randomness with structure for sublinear time kernel expansions. In *ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2502–2510. JMLR.org, 2016.

[15] Felix X. Yu, Ananda Theertha Suresh, Krzysztof Marcin Choromanski, Daniel Holtmann-Rice, and Sanjiv Kumar. Orthogonal random features. In *NIPS*, pages 1975–1983, 2016.

[16] Nitin Thaper, Sudipto Guha, Piotr Indyk, and Nick Koudas. Dynamic multidimensional histograms. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 428–439. ACM, 2002.

[17] Nicolas Keriven, Nicolas Tremblay, and Rémi Gribonval. Sketchmlbox: A matlab toolbox for large-scale mixture learning. 2016.

[18] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. IEEE Computer Society, 2009.