Compressive Classification (Machine Learning without learning)

Vincent Schellekens* and Laurent Jacques*

Abstract—Compressive learning is a framework where (so far unsupervised) learning tasks use not the entire dataset but a compressed summary (sketch) of it. We propose a compressive learning classification method, and a novel sketch function for images.

1 Introduction and background

Machine Learning (ML)—inference models from datasets of numerous learning examples—recently showed unparalleled success on a wide variety of problems. However, modern massive datasets necessitate a long training time and large memory storage. The recent Compressive Learning (CL) framework alleviates those drawbacks by computing a compressed summary of the dataset—its sketch—prior to any learning [1]. The sketch is easily computed in a single parallelizable pass, and its required size (to capture enough information for successful learning) does not grow with the number of examples: CLs time and memory requirements are thus unaffected by the dataset size.

So far, CL focused on unsupervised ML tasks, where learning examples don’t belong to a (known) class [1, 2, 3]. We show that CL easily extends to supervised ML tasks by proposing (Sec. 2) and experimentally validating (Sec. 3) a first simple compressive classification method using only a sketch of the labeled dataset (Fig 1). We also introduce a sketch feature function leveraging a random convolutional neural network to better capture information in images. While not as accurate as ML estimation leveraging a random convolutional neural network to betting examples don’t belong to a (known) class [1, 2, 3]. We sketch storage. The recent datasets necessitate a long training time and large memory requirements are thus unaffected by the dataset size.

Compressive Learning is a framework where (so far unsupervised) learning tasks use not the entire dataset but a compressed summary (sketch) of it. We propose a compressive learning method, and a novel sketch function for images. Our method also enjoys from a nice geometric interpretation, i.e., Maximum A Posteriori classification performed in the Reproducible Kernel Hilbert Space associated with the sketch.

(Unsupervised) Compressive Learning: Unsupervised ML usually amount to estimate parameters of a distribution $P$, from a dataset $\mathcal{X} := \{x_i \sim iid \, P\}_{i=1}^N \subset \mathbb{R}^n$ of examples—associated to an empirical distribution $\hat{P}_X := \frac{1}{N} \sum_{x_i \in \mathcal{X}} \delta_{x_i}$, with $\delta_u$ the Dirac measure at $u$. While most unsupervised ML algorithms require (often multiple times) access to the entire dataset $\mathcal{X}$, CL algorithms require only access to the sketch: a single vector $z_{\mathcal{X}} \in \mathbb{C}^m$ summarizing $\mathcal{X}$. This dataset sketch $z_{\mathcal{X}}$ actually serves as a proxy for the true distribution sketch $A(P)$, i.e., a linear embedding of the “infinite-dimensional” probability distribution $P$ into $\mathbb{C}^m$, a space of lower dimension:

$$A(P) := \mathbb{E}_{x \sim P} f(x) \simeq z_{\mathcal{X}} := A(\hat{P}_X) = \frac{1}{N} \sum_{x_i \in \mathcal{X}} f(x_i),$$

where $f$ is a random nonlinear feature map to $\mathbb{C}^m$. This map defines a positive definite kernel $\kappa(u, v) := \mathbb{E}(f(u), f(v))$, and $\kappa$ in turn provides a Reproducible Kernel Hilbert Space (RKHS) $\mathcal{H}_\kappa$ to embed distributions; $A$ indirectly maps $P$ to its Mean Map $\kappa(.) ; P := \mathbb{E}_{x \sim P} \kappa(.) ; x \in \mathcal{H}_\kappa$ [4, 5]. Existing methods [2, 3] use Random Fourier Features [7] as map $f$:

$$f_{\text{RF}}(x) = \left[\exp(i \omega_j^T x)\right]_{j=1}^m \text{ with } \omega_j \sim iid \, \Lambda,$$

and $\kappa$ is then shift-invariant and the Fourier transform of the distribution $\Lambda$: $\kappa(x, x') = \varpi(x-x')$ for solving ML infers a mathematical model from a labeled dataset $\{x_i, y_i\}_{i=1}^N \subset \mathbb{R}^n$ where each signal $x_i \in \mathbb{R}^n$ belongs to a class $C_k$ as designated by its class label $y_i \in [K]$. Denoting $p_k := \mathbb{P}(x \in C_k) = \mathbb{P}(y = k)$, the signals are assumed drawn from an unknown density $P$:

$$x_i \sim iid \, P = \sum_{k=1}^K p_k \, p(x \mid x \in C_k) =: \sum_k p_k \mathcal{P}(x).$$

As illustrated in Fig 1top, our supervised compressive learning framework considers that $\mathcal{X}$ is not explicitly available but compressed as a collection of $K$ class sketches $z_{\mathcal{X}_k}$ defined as:

$$z_{\mathcal{X}_k} = A(\mathcal{P}_{\mathcal{X}_k}) \text{ where } \mathcal{X}_k := \{x_i \in C_k\}. (4)$$

*E-mail: {vincent.schellekens, laurent.jacques}@uclouvain.be. ISPGroup, ELEN/ICTEAM, UCLouvain (UCL), B1348 Louvain-la-Neuve, Belgium. VS and LJ are funded by Belgian National Foundation (F.R.S.-FNRS).
We can also require approximated a priori class probabilities \( \hat{p}_k \), e.g., \( \hat{p}_k = \frac{N_k}{m} \) if we count the class occurrences \( N_k = |X_k| \), or setting an uniform prior \( \hat{p}_k = \frac{1}{K} \) otherwise.

**Classification phase:** Under (3), the optimal classifier (minimal error probability) for a test example \( x' \) is the Maximum A Posteriori (MAP) estimator \( k_{\text{MAP}} := \arg\max_{k} \hat{p}_k \varphi_k(x') \), where \( \varphi_k \) is generally hard to estimate. In our CL framework, we classify \( x' \) from \( \{X_k\} \) and \( \hat{p}_k \) only (Fig. 1, bottom): we acquire its sketch \( \varphi_k(x') = f(x') \) and maximize the correlation with the class sketch weighted by \( \hat{p}_k \), i.e., we assign to \( x' \) the label

\[
k^* := \arg\max_k \hat{p}_k \varphi_k(x', \{X_k\})
\]

Note that this Compressive Classifier (CC) does not require parameter tuning. Interestingly, under a few approximations, this procedure can be seen as a MAP estimator in the RKHS \( \mathcal{H}_k \). Indeed, we first note that if \( m \) is large, the law of large numbers (LLN) provides the kernel approximation (KA)

\[
\langle f(u), f(v) \rangle \simeq \kappa(u, v), \quad \forall u, v \in \mathbb{R}^n. \quad (\text{KA})
\]

Assuming \( N_k \) is also large, another use of the LLN gives the mean map approximation (MMA): we have both \( \hat{p}_k \simeq p_k \) and

\[
\langle z_{u}, \kappa(X_k) \rangle \simeq \frac{1}{N_k} \sum_{x_i \in X_k} \langle f(u), f(x_i) \rangle \simeq \frac{1}{N_k} \sum_{x_i \in X_k} \kappa(u, x_i) \simeq \mathbb{E}_{x \sim P_k} \kappa(u, x) := \kappa(u, P_k) \quad \forall u \in \mathbb{R}^n. \quad (\text{MMA})
\]

Consequently, under the KA and MMA approximations,

\[
k^* \simeq \arg\max_k p_k \kappa(x', P_k), \quad (5)
\]

or in other words, we replace \( P_k \) in the MAP estimator by its Mean Map \( \kappa(\cdot, P_k) \)—its embedding in \( \mathcal{H}_k \)—such that CC computes a MAP estimation inside the RKHS \( \mathcal{H}_k \). In all generality \( \kappa(\cdot, P_k) \) is not a probability density function, but can be interpreted as a smoothing of \( P_k \) by convolution with \( \varpi(u, 0) \) if \( \kappa \) is a properly scaled shift-invariant kernel. Alternatively, (3) can be seen as a Parzen-windows classifier—a nonparametric Support Vector Machine (without weights learning)—evaluated compressively thanks to the sketch (3)(4).

**3 Experimental proof of concept**

**Synthetic datasets:** We build two datasets that are not linearly separable (Fig. 2, left), and sketch them using \( f = f_{RFF} \) with \( \Lambda \sim \mathcal{N}(0, \frac{1}{2\sigma^2}) \); therefore \( \kappa(u, v) \propto \exp(-\frac{\|u-v\|^2}{2\sigma^2}) \). As shown Fig. 2 (right), the test accuracy of CC improves with \( m \) until reaching—when the KA is good enough—a constant floor depending on the compatibility between \( \kappa \) and \( P \). Accuracy is almost optimal when \( \kappa \) is close to the constituents of \( P \) (e.g., 1st dataset, \( \sigma = 0.1 \)), but degrades when the kernel scale and/or shape mismatches the data clusters (e.g., 2nd dataset, \( \sigma = 10 \); or 2nd dataset). CC thus reaches good accuracy provided \( m \) is large enough and \( \kappa \) is well adapted to the task.

**Standard datasets:** We also test CC on some well-known “real-life” datasets from the UCI ML Repository [15]. Table 1 compares the error rates of CC and SVM, a fully learned approach. Although worse than SVM, CC is surprisingly accurate considering its compressive nature, low computational cost (especially when \( m = 50 \)), and that \( \kappa \) is a basic, non-tuned kernel.

**Image classification:** More challenging are image classification datasets: handwritten digit recognition (MNIST [16]) and vehicle/animal recognition (CIFAR-10 [17]). We use \( f = f_{\text{CNN}} \) (the default architecture provided by [18]) because it yielded better accuracy than \( f_{RFF} \), and compare CC to the same CNN architecture with a classification layer, with all weights learned in one pass over \( \mathcal{X} \) for fairness. Again CC is outperformed by the learned approach, but still achieves reasonable, non-trivial accuracy. Surprisingly, CC performs here better on the test set than on the training set.

**4 Discussion and conclusion**

We proposed a very simple and flexible compressive classification method, relying only on class sketches: accumulated random nonlinear signatures \( f(\cdot) \) of the learning examples. This classifier is cheap to evaluate (e.g., in low-power hardware, following ideas from [19]), involves no parameter tuning, and has an interesting interpretation: a MAP estimator inside the RKHS \( \mathcal{H}_k \), associated with the kernel \( \kappa \) defined by \( f \). Preliminary experimental results, relying on a basic Gaussian \( \kappa \), are an encouraging proof of concept, but indicate room for improvement if the mapping \( f \) (and associated kernel \( \kappa \)) is optimized according to the true data distribution; for example, image classification accuracy improves when \( f \) is a random CNN (defining a shift-variant \( \kappa \)). Intuitively, \( \kappa \) should be such that the Mean Maps \( \kappa(\cdot, P_k) \in \mathcal{H}_k \) of different classes \( k \) are “well separated” (ideally as much separated as the initial, unknown densities \( P_k \)). This could be done by adding some a priori assumptions on the densities \( P_k \), or by first getting a rough estimation of them through a form of distilled sensing [20]. To be reliable, compressive classification also requires precise, non-asymptotic guarantees, e.g., using results from [5] and [14].
References

[1] R. Gribonval, G. Blanchard, N. Keriven, and Y. Traonmilin, “Compressive Statistical Learning with Random Feature Moments,” ArXiv e-prints, Jun. 2017.

[2] N. Keriven, N. Tremblay, Y. Traonmilin, and R. Gribonval, “Compressive K-means,” ICASSP 2017 - IEEE International Conference on Acoustics, Speech and Signal Processing, 2017.

[3] N. Keriven, A. Bourrier, R. Gribonval, and P. Pérez, “Sketching for Large-Scale Learning of Mixture Models,” Information and Inference: A Journal of the IMA, 2017.

[4] N. Aronszajn, “Theory of reproducing kernels.” Transactions of the American Mathematical Society, no. 68, pp. 337–404, 1950.

[5] A. Smola, A. Gretton, L. Song, B. Scholkopf, “A Hilbert space embedding for distributions”, International Conference on Algorithmic Learning Theory, Springer, Berlin, Heidelberg, 2007.

[6] B. K. Sriperumbudur, A. Gretton, K. Fukumizu, B. Schölkopf, and G. R. Lanckriet, “Hilbert Space Embeddings and Metrics on Probability Measures,” Journal of Machine Learning Research, vol. 11, pp. 1517–1561, Aug. 2010.

[7] A. Rahimi and B. Recht, “Random Features for Large-Scale Kernel Machines,” in Advances in Neural Information Processing Systems 20, J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Eds. Curran Associates, Inc., 2008, pp. 1177–1184.

[8] W. Rudin, Fourier Analysis on Groups. Interscience Publishers, 1962.

[9] D. Ulyanov, A. Vedaldi, V. Lempitsky, “Deep Image Prior,” arXiv preprint, 2017.

[10] R. Giryes, G. Sapiro, A.M. Bronstein “Deep Neural Networks with Random Gaussian Weights: A Universal Classification Strategy?,” IEEE Transactions on Signal Processing, vol. 64, no. 13, pp. 3444-3457, Jul. 2016.

[11] A. Rosenfeld, J.K. Tsotsos, “Intriguing Properties of Randomly Weighted Networks: Generalizing While Learning Next to Nothing,” arXiv preprint, 2018.

[12] Y. Cho, L.K. Saul, “Kernel methods for deep learning,” Advances in neural information processing systems, 2009.

[13] R. O. Duda, P. E. Hart “Pattern Classification and Scene Analysis”, Wiley Interscience, 1973.

[14] B. Scholkopf, A. J. Smola, “Learning with kernels: support vector machines, regularization, optimization, and beyond”, MIT press, 2001.

[15] A. Asuncion, D.J. Newman, “UC Irvine Machine Learning Repository,” http://archive.ics.uci.edu/ml/index.php Accessed: 2018-05-15.

[16] Y. LeCun, C. Cortes, and C. J. Burges, “The MNIST database of handwritten digits,” http://yann.lecun.com/exdb/mnist/ Accessed: 2018-05-15.

[17] A. Krizhevsky, “The CIFAR-10 dataset,” https://www.cs.toronto.edu/~kriz/cifar.html Accessed: 2018-05-15.

[18] The MatConvNet Team, “MatConvNet: CNNs for MATLAB,” http://www.vlfeat.org/matconvnet/ Accessed: 2018-06-13.

[19] V. Schellekens, and L. Jacques, ”Quantized Compressive K-Means,” arXiv preprint, arXiv:1804.10109 2018.

[20] J. Haupt, R.M. Castro, and R. Nowak, "Distilled sensing: Adaptive sampling for sparse detection and estimation," IEEE Transactions on Information Theory, vol. 57, no. 9, 2011, pp. 6222-6235.