Giovanni Pagliarini (A0209803H)
Corentin Dumery (A0198927E)

Project Report

CS5234: Algorithms at Scale
October 30, 2019

What we have done and plan to do

Coin flips Following our plans from the previous report, we implemented the Johnson-Lindenstrauss transform described by D. Achlioptas in *Database-friendly random projections: Johnson-Lindenstrauss with binary coins*. The idea is simple: we construct a matrix R where any $R_{i,j}$ is randomly assigned a value in $\{-1, 0, 1\}$. The transform described in the article is simply the product $\frac{1}{\sqrt{k}} xR$ where x is the input vector and k the output's dimension. Note that in their proof, the authors define two parameters ϵ and β in $[0, 1]$ and assume that the reduced number of dimensions is not too small, else their proof doesn't hold. Specifically, it has to satisfy the inequality :

$$\begin{aligned} k &\geq k_0 \\ &\geq \frac{4 + 2\beta}{\epsilon^2/2 - \epsilon^3/3} \log(n) \end{aligned}$$

Taxi? We were then hoping to apply this transform to real-world data from taxi trips. Since this data has very different attributes, including some textual fields, we had to define a scaling factor for each column. Everything was scaled to $[0, 1]$. That way, no column is made arbitrarily more important than another. However, this dataset still had a major problem: it has only 17 dimensions. The constraint on k described earlier becomes, for the best values of ϵ and β, $k \geq 25 \log(n)$. Therefore, the proof that this transform satisfies the Johnson-Lindenstrauss lemma cannot hold if we try to reduce the data to less than 17 dimensions. It became obvious that we had to either find a better dataset or a better transform. To get more significant results, we decided to try changing both.

FJLT First, we implemented the transform described by Ailon N. and Chazelle, B. in *The fast Johnson-Lindenstrauss transform and approximate nearest neighbors*. Unlike the previous JLT method, the proof given in the paper describing this method holds for any k, as long as k is at least of the order $O(log(n))$. This transform is slightly more complex than the previous one; the random projection matrix Φ, here, is computed as the product of three matrices: a random matrix P, a normalized Walsh–Hadamard matrix H, and another random matrix D. Similarly, the transform of x is given by the product $\Phi x = P(HDx)$. The idea behind this approach is that, before applying the random projection P, pre-multiplying x by HD is expected to yield lower values compared with considering x alone: this allows the multiplication by P to then randomly sample and aggregate dimensions without entries getting too far apart from each other.

Digits! Since we also wanted to extend our research to higher dimensional data, we experimented with two new datasets, and intend to introduce even wider datasets in the upcoming weeks. For now, the two new datasets are ***sklearn**'s Digit Dataset* and the very popular *MNIST database*. They are very similar to each other, in which their entries consist of 2D

matrices representing handwritten digits, but differ in dimensionality. The first one consists of ~ 1800 8×8 matrices, the second one ~ 70000 28×28 matrices. With these datasets we derived interesting results using Achlioptas' method, whereas we found that using our FJL transform the quality degrades much more; we suspect there might be an implementation error, and we are currently debugging our FJLT.

Quality study We decided to limit this study to clustering, and our first and main test problem is K-means clustering; but how to exactly use it as a measure of quality degradation? What follows is a description of our process of assessing the goodness of our dimensionality reduction methods.

Given a dataset of labeled entries in R^n, we perform K-means in the original space, obtaining a clustering which approximates the ground-truth clustering (the one given by the dataset's labels/classes). We call the result of K-means on the original data our "baseline clustering". Then, for different JLT methods we apply the method to the input data, followed by K-means, obtaining a clustering in the reduced space R^k. In order to tell if the reduced clustering is good, we must: (1) decide if we want to compare it to either the ground-truth or the baseline clusterings; (2) find the right metric to compare two clusterings.

As for the metric, we initially thought of considering the value of K-means' objective function at convergence. This metric has lots of problems, the major one being that it doesn't allow straightforward comparison between results from different spaces R^n and R^k (maybe n/k-scaling does the job for L1 norm, but we have no proof of this). Ultimately, we found the two interesting measures of *completeness* and *homogeneity*, introduced by Rosenberg and Hirschberg (2007), and their harmonic mean called *v_measure* (from now on, v for simplicity). These three are measures in $[0, 1]$ that captures how much a clustering is similar to another one.

As for whether we should refer to the real or the baseline clustering, we have reasons for doing both. On one hand, the goal of dimensionality reduction is to approach a similar distribution of the original data into a lower-dimension space, regardless of any clustering algorithm. On the other hand, K-means is the lens through which we inspect the distribution of the entries (more specifically, it tells whether the right or wrong entries got closer to each other): it may occur that the baseline itself is not a good clustering, just because K-means is not compatible with the dataset, and in this case, comparing with the ground-truth is too short-sighted. Moreover, while dimensionality reduction's is a good way to reduce the CPU workload, it could also occur that the projection makes the data actually easier to cluster correctly. This hasn't happened in our study yet, however, it would be quite interesting to see some cases where K-means performs worse on the original space than it does on the reduced space; or, in case this actually cannot happen, we may look for a proof of the opposite statement. Therefore, we are keeping doors open and considering both $\frac{v_{reduced}}{v_{ground}}$ and $\frac{v_{reduced}}{v_{baseline}}$. We may consider to design an aggregated adhoc metric in the future.

What's next While, so far, K-means has been our only test problem to measure the quality of the reduction, we want to widen this project's quest by trying at least one different clustering method. Although it would be interesting to compare results with the similar algorithm that is K-medians, we are probably going to leave this for a later stage of this work, as we are more attracted to **Mean-shift** clustering, and `sklearn` already has a ready-made class for it.

In our project proposal, we proposed to combine dimensionality reduction with random sampling, but we eventually realized the questions we were asking were not interesting enough, and we changed our horizons. Instead, in order to avoid putting flesh on the bones, we are going to focus on dimensionality reduction only, and will **leave sampling for an eventual later stage**.

Indeed, sampling may be both an interesting path to discover and a tool to approach **bigger datasets**, since, as mentioned above, we are going to introduce datasets with hundreds of variables: see, for instance, *DIM-sets (high)* at https://cs.uef.fi/sipu/datasets/, or even wider databases at https://archive.ics.uci.edu/ml/datasets.php?format=&task=&att=&area=&numAtt=&numIns=&type=&sort=attDown&view=table. A further interesting study could be done by comparing the results for these datasets to those derived on lower dimensional (labeled) datasets. Note that, having changed our horizons, the "taxis" database may not perfectly fit this study anymore: being unlabeled, we are going to assess the quality of the reduction with respect of the baseline only.